TIME SERIES ANALYSIS

Chapter 10 Change point detection

1 Change point detection

Assume that you wear an iWatch to monitor your heart rate. You run for a mile, walk for twenty minutes and then run for another mile. The heart rate data show a cluster of high heart rate, followed by a cluster of low heart rate and then back to high rate like Figure 1. When a data analyst investigates the heart rate data the changes in the time series reveal there are changes in the underlying activities. Similarly, the heart rate of a patient in an Intensive Care Unit (ICU) is monitored. It is critical to detect any changes in the heart rate and alert the doctor to respond quickly. The abrupt change in the behavior of a time series are often alarming because they signal something significant. We need to detect them accurately and timely and send out alarms.

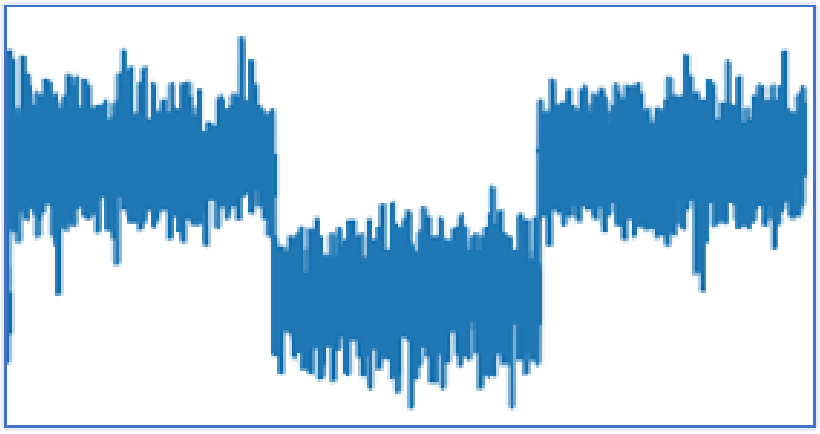


Figure 1: mean.data

Change points are abrupt variations in time series data. Such abrupt changes may represent transitions that occur between states. Detection of change points is useful in modeling and prediction of time series and is found in application areas such as medical condition monitoring, climate change detection, speech and image analysis, and human activity analysis.

Change point detection is the problem of finding abrupt changes in data when a property of the time series changes. Segmentation, edge detection, event detec-

tion, and anomaly detection are similar concepts which are occasionally applied as well as change point detection

To put it simple, a change point divides a time series into different segments where each segment has its own statistical characteristics (e.g., mean, variance, etc.).

More formally, let us assume we have an ordered sequence of data $y_{1:n} = (y_1, \cdots, y_n)$. A change point is said to occur within this set when there exists a time, $\tau \in \{1, 2, \cdots, n-1\}$ such that the statistical properties of $\{y_1, \cdots, y_\tau\}$ and $y_{\tau+1}, \cdots, y_n$ are different in some way. Extending this idea of a single changepoint to multiple changes, we will have a number of changepoints, m, together with their positions, $(\tau_1, \cdots, \tau_m)$. Consequently the m changepoints will split the data into m + 1 segments, with the ith segment containing data $y_{\tau_{i-1}:\tau_i}$. Typically we want to test how many segments are needed to represent the data, i.e., how many changepoints are present and estimate the values of the parameters associated with each segment.

2 Offline and On-line

There are two types of the change point detection algorithms, one is offline and the other is on-line or real-time. The offline algorithms use all the data points to find the change points. Why two types ? It is because the computation time of the offline methods is too long to quality for a real-time purpose, although they may detect all the change points. In contrast, the on-line algorithms can detect the change points "on the fly". With the emergence of real-time streaming data such as IoT or on-line streaming, the on-line methods can be implemented on a hardware device.

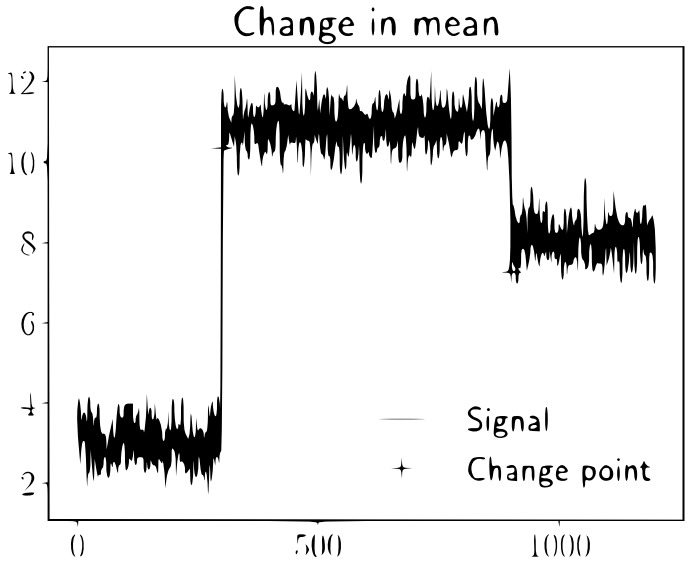
3 Different types of change points

3.1 Change in mean

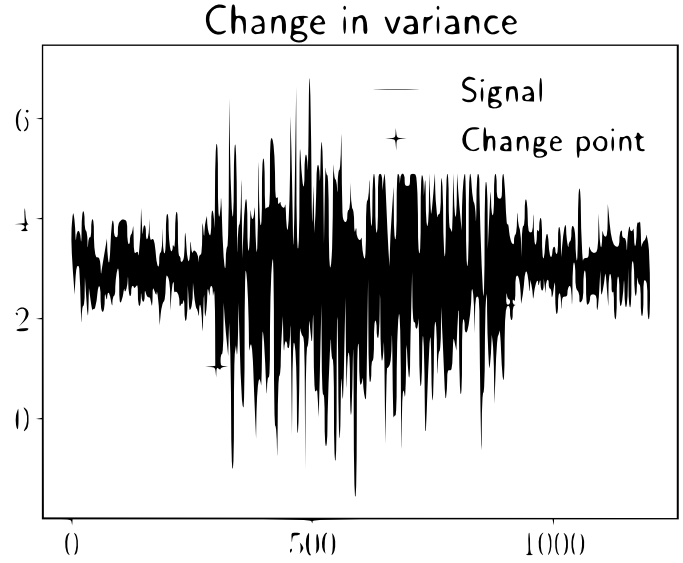
Change in mean is the most common example and also the easiest to identify. Change in mean usually occurs when a time series can be divided into different constant segments having different mean values. One of the earliest algorithms for detecting such changes is the Cumsum algorithm (Page 1954), which was developed to detect change in mean. It was applied for quality control in manufacturing.

3.2 Change in variance

Change in variance is another common example. The mean value of the signals keeps constant, but there are several segments in terms of different variance values. This can be interpreted as a sudden noise in the signals. Both changes in



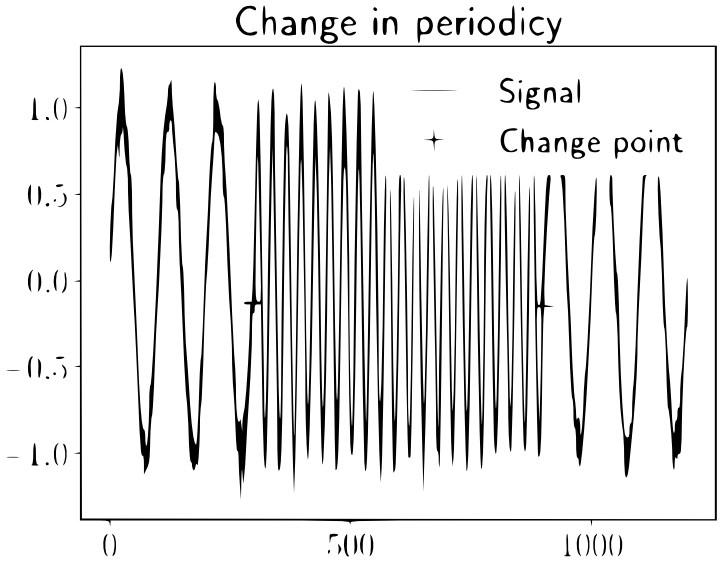
mean and change in variance can be detected by comparing statistical properties through the signal.



3.3 Change in periodicity

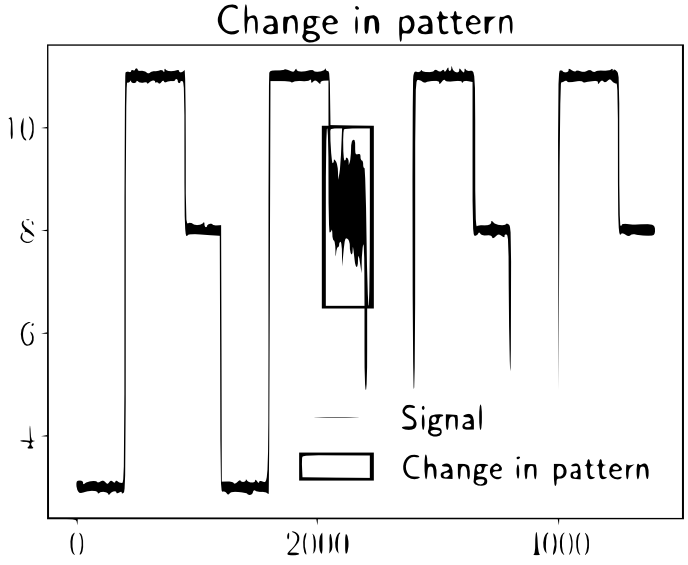
Change in periodicity (also called change in frequency) concerns time series with cyclic properties (e.g., a machine's regime). Here, the change occurs when the frequency changes suddenly. Detection of this kind of change is usually

done in the frequency domain, for instance by the Fourier transform or wavelet transform.



3.4 Change in pattern

Change in pattern is more difficult to handle than the preceding. Such changes can occur, for instance, in ECG signals. One way to detect them is to use Wasserstein distances between the empirical distributions. At this point, we can see that change point detection is closely related to anomaly detection.



4 Single change point detection

We now introduce the general likelihood ratio based approach to test this hypothesis. The potential for using a likelihood based approach to detect changepoints was first proposed by Hinkley (1970) who derives the asymptotic distribution of the likelihood ratio test statistic for a change in the mean within normally distributed observations.

The detection of a single changepoint can be posed as a hypothesis test. The null hypothesis, H_0, corresponds to no changepoint (m = 0) and the alternative hypothesis, H_1, is a single changepoint (m = 1).

A test statistic can be constructed which we will use to decide whether a change has occurred. The likelihood ratio method requires the calculation of the maximum log-likelihood under both null and alternative hypotheses. For the null hypothesis the maximum log-likelihood is

$$logp(y_{1:n}|\hat{\theta})$$

where $p(\cdot)$ is the probability density function associated with the distribution of the data and $\hat{\theta}$ is the maximum likelihood estimate of the parameters.

Under the alternative hypothesis, consider a model with a changepoint at τ_1, with $\tau_1 \in \{1, 2, ..., n-1\}$. Then the maximum log likelihood for a given τ_1 is

$$ML(\tau_1) = logp(y_{1:\tau_1}|\hat{\theta}_1) + logp(y_{\tau_1+1:n}|\hat{\theta}_2)$$

Given the discrete nature of the changepoint location, the maximum log-likelihood value under the alternative is simply $\max_{\tau_1} ML(\tau_1)$ where the maximum is taken over all possible changepoint locations. The test statistic is thus

$$\lambda = 2\Big[\max_{\tau_1} ML(\tau_1) - logp(y_{1:n}|\hat{\theta})\Big].$$

The test involves choosing a threshold, c, such that we reject the null hypothesis if $\lambda > c$. If we reject the null hypothesis, i.e., detect a changepoint, then we estimate its position as $\hat{\tau}_1$ the value of τ_1 that maximizes $ML(\tau_1)$.

5 Multiple change points detection

With increased collection of time series and signal streams there is a growing need to be able to efficiently and accurately estimate the location of multiple changepoints.

It is clear that the likelihood test statistic can be extended to multiple changes simply by summing the likelihood for each of the m segments. The problem becomes one of identifying the maximum of $ML(\tau_{1:m})$ where $\tau_{1:m} = (\tau_1, \cdots, \tau_m)$ over all possible combinations of $\tau_{1:m}$.

Arguably the most common approach to identify multiple changepoints in the literature is to minimize

$$\sum_{i=1}^{m+1} C(y_{\tau_{i-1}+1:\tau_i}) + \beta f(m) \tag{5.1}$$

where C is a cost function for a segment e.g., negative log-likelihood and f(m) is a penalty to guard against over fitting (a multiple changepoint version of the threshold c). A brute force approach to solve this minimization considers 2^{n-1} solutions reducing to $\binom{n-1}{m}$ if m is know. The changepoint package implements three multiple changepoint algorithms that minimize (5.1); binary segmentation (Edwards and Cavalli-Sforza 1965), segment neighborhoods (Auger and Lawrence 1989) and the pruned exact linear time (PELT).

Binary segmentation is arguably the most widely used multiple changepoint search method. Briefly, binary segmentation first applies a single changepoint test statistic to the entire data, if a changepoint is identified the data is split into two at the changepoint location. The single changepoint procedure is repeated on the two new data sets, before and after the change. If changepoints are identified in either of the new data sets, they are split further. This process continues until no changepoints are found in any parts of the data. This procedure is an approximate minimization of (5.1) with f(m) = m as any changepoint locations are conditional on changepoints identified previously. Binary segmentation is thus an approximate algorithm but is computationally fast as it only considers a subset of the 2^{n-1} possible solutions. The computational complexity of the algorithm is $O(nlogn)$.

The PELT algorithm proposed is similar to that of the segment neighborhood algorithm in that it provides an exact segmentation. However, due to the construction of the PELT algorithm, it can be shown to be more computationally efficient, due to its use of dynamic programming and pruning which can result in an $O(n)$ search algorithm.

6 Changes in mean: The cpt.mean function

Early work on changepoint problems focused on identifying changes in mean and includes the work of Page (1954) and Hinkley (1970) who created the likelihood ratio and cumulative sum (CUSUM) test statistics respectively.

Within the changepoint package all change in mean methods are accessed using the cpt.mean function. The function is structured as follows:

```
cpt.mean(data, penalty = "SIC", pen.value = 0, method = "AMOC", Q = 5,
test.stat = "Normal", class = TRUE, param.estimates = TRUE)
```

We begin by demonstrating the various possible stages within a change in mean analysis. To this end we simulate a dataset (m.data) of length 400 with multiple changepoints at 100, 200, 300. The sequence has four segments and the means for each segment are 0, 1, 0, 0.2.

```
library(changepoint)
> set.seed(10)
> mean.data <- c(rnorm(100, 0, 1), rnorm(100, 1, 1), rnorm(100, 0, 1),
+ rnorm(100, 0.3, 1))
> ts.plot(mean.data, xlab = "Index")
```

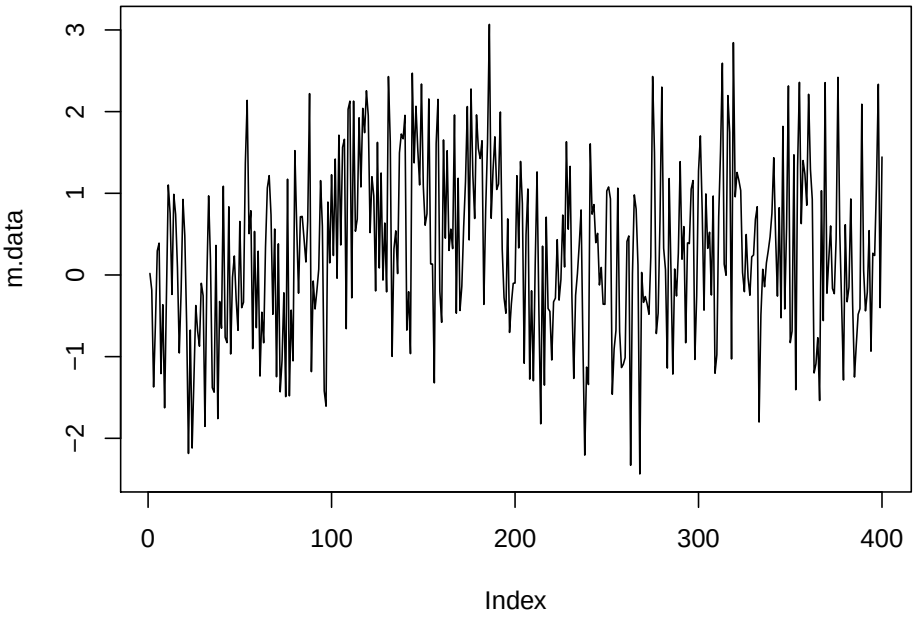


Figure 2: mean.data

The first question we aim to answer is "Is there a change within the data?". From a visual inspection of the data in Figure 1, we suspect multiple changes in mean may exist. The challenge in multiple changepoint detection is identifying the optimal number and location of changepoints as the number of solutions increases rapidly with the size of the data. In this example where n = 400, we have 399 possible solutions for a single changepoint, for two changes there are 79401 possible solutions and this is not taking into account that we do not know how many changes there are! As such it is clearly desirable to use an efficient method for searching the large solution space.

```
> m.pelt <- cpt.mean(mean.data, method = "PELT")
> plot(m.pelt, type = "l", cpt.col = "blue", xlab = "Index",cpt.width = 4)
> cpts(m.pelt)
```

```
>97,192

> m.binseg <- cpt.mean(mean.data, method = "BinSeg")
> plot(m.binseg, type = "l", xlab = "Index", cpt.width = 4)
> cpts(m.binseg)
> 79 192

m.pm <- cpt.mean(mean.data, penalty = "Manual", pen.value = "1.5 * log(n)",
 method = "PELT")
> plot(m.pm, type = "l", cpt.col = "blue", xlab = "Index", cpt.width = 4)
> cpts(m.pm)
>97 192 273
```

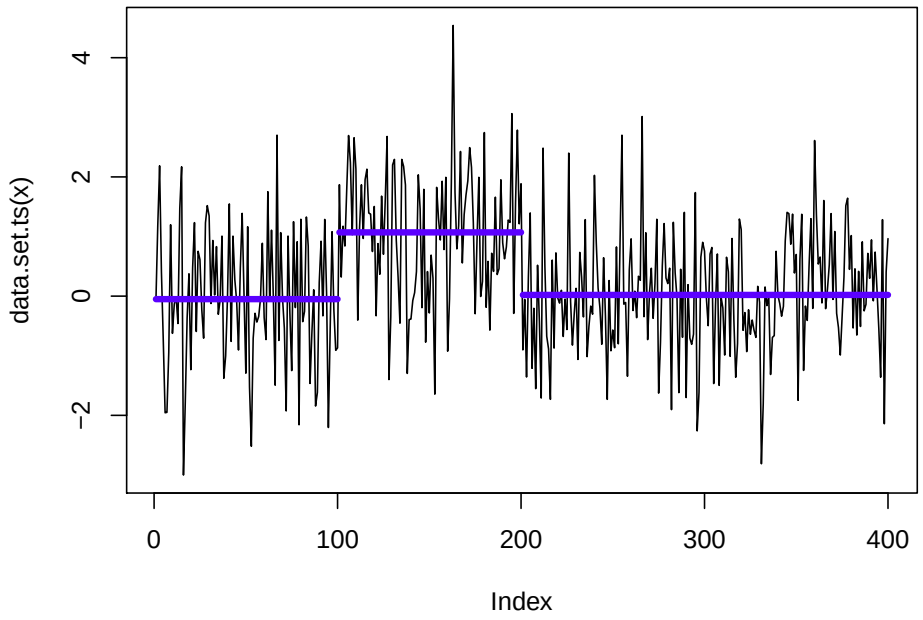


Figure 3: Binary segmentation changepoints with default penalt

Note that there are two changes towards the end of the dataset which have very small segments. These are plausibly artifacts of the data rather than true changes in the underlying process. In an effort to remove these seemingly spurious changepoints we can increase the penalty to 1.5 * log(n) rather than log(n) (SIC). This change is achieved by changing the penalty type to "Manual" and set-

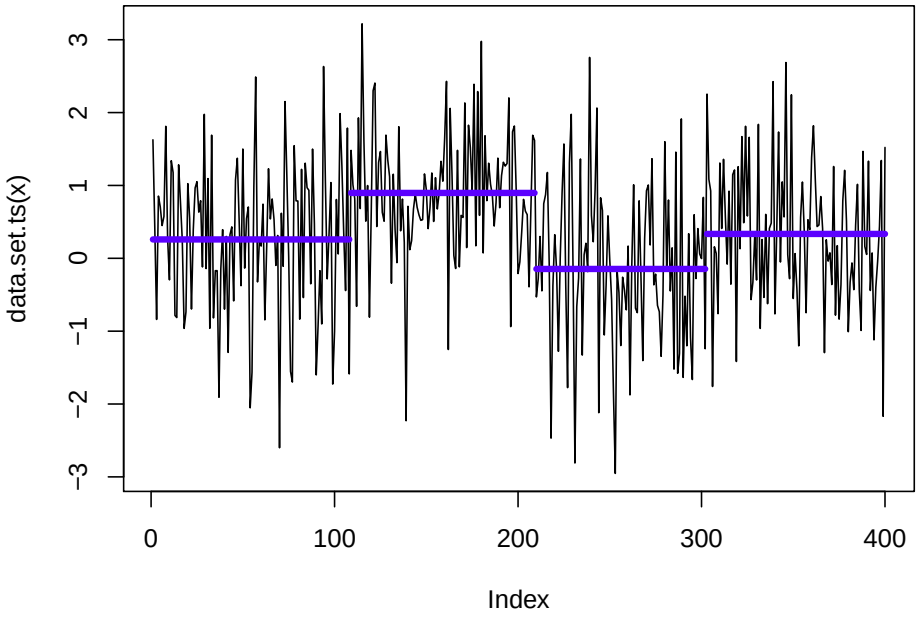


Figure 4: PELT changepoints with manual penalty

ting the value argument to "1.5 * log(n)". Figure 1(d) shows the result which seem more plausible.

Consider segmenting array comparative genomic hybridization (aCGH) data from Glioblastoma multiforme (GBM), a type of brain tumor. These arrays were developed to identify DNA copy number alteration corresponding to chromosomal aberrations. High-throughput aCGH data are intensity ratios of diseased vs. control samples indexed by the location on the genome. Values greater than 1 indicate diseased samples have additional chromosomes and values less than 1 indicate fewer chromosomes. Detection of these aberrations can aid future screening and treatments of diseases.

```
> data("Lai2005fig4", package = "changepoint")
> Lai.default <- cpt.mean(Lai2005fig4[, 5], method = "PELT")
> plot(Lai.default, pch = 20, col = "grey", cpt.col = "black", type = "p",
 xlab = "Index")
> cpts(Lai.default)
>81  85  89  96 123 133
coef(Lai.default)
$mean
[1] 0.2468910 4.6699210 0.4495538 4.5902489 0.2079891 4.2913844 0.2291286
```

7 Changes in variance: The cpt.var function

Consider an example of Irish wind speeds. With the increase of wind based renewables in the power grid, there has become great interest in forecasting wind speeds. Often modelers assume a constant dependence structure when modeling the existing data before producing a forecast. Here we conduct a naive changepoint analysis of wind speed data which are available in the R package gstat. The data provided are daily wind speeds from 12 meteorological stations in the Republic of Ireland.

Here we consider a single site, Claremorris depicted in Figure .

```
> data("wind", package = "gstat")
> ts.plot(wind[, 11], xlab = "Index")
```

The variability of the data appears smaller in some sections and larger in others, this motivates a search for changes in variability. Wind speeds are by nature diurnal and thus have a periodic mean. The change in variance approaches within the cpt.var function require the data to have a fixed value mean over time and thus this periodic mean must be removed prior to analysis. Whilst there are a range of options for removing this mean, we choose to take first differences as this does not require any modeling assumptions. Following this we assume that the differences follow a Normal distribution with changing variance and thus use the cpt.var function.

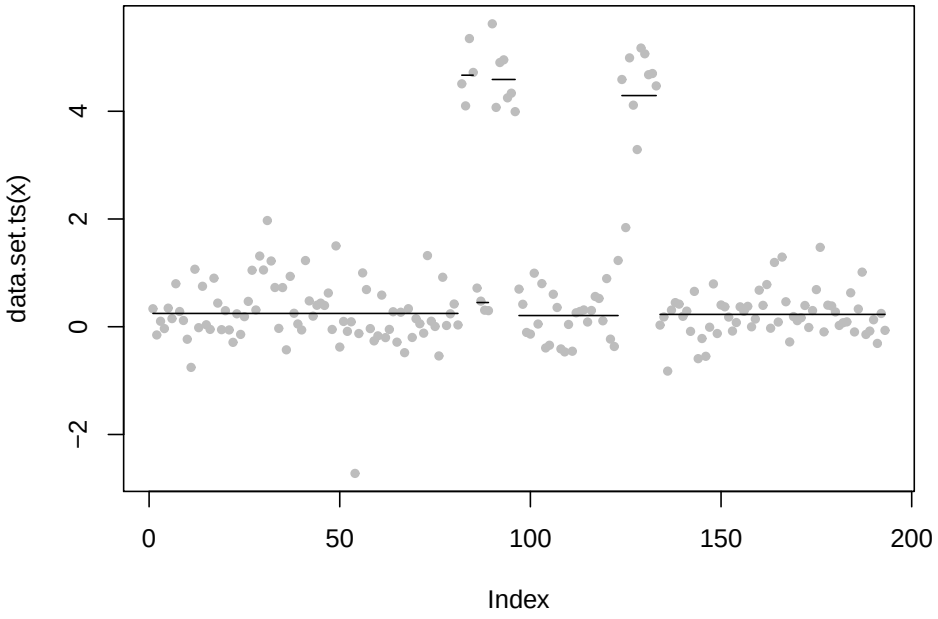


Figure 5: Case study

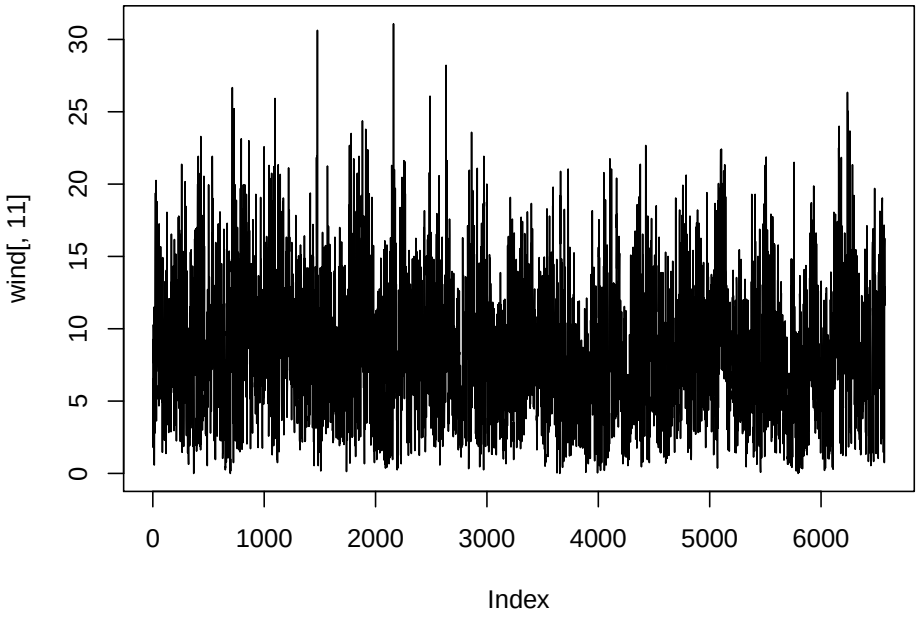


Figure 6: Irish

```
>wind.pelt <- cpt.var(diff(wind[, 11]), method = "PELT")
> plot(wind.pelt, xlab = "Index")
>2
> logLik(wind.pelt)
>wind.bs <- cpt.var(diff(wind[, 11]), method = "BinSeg")
>ncpts(wind.bs)
>1
```

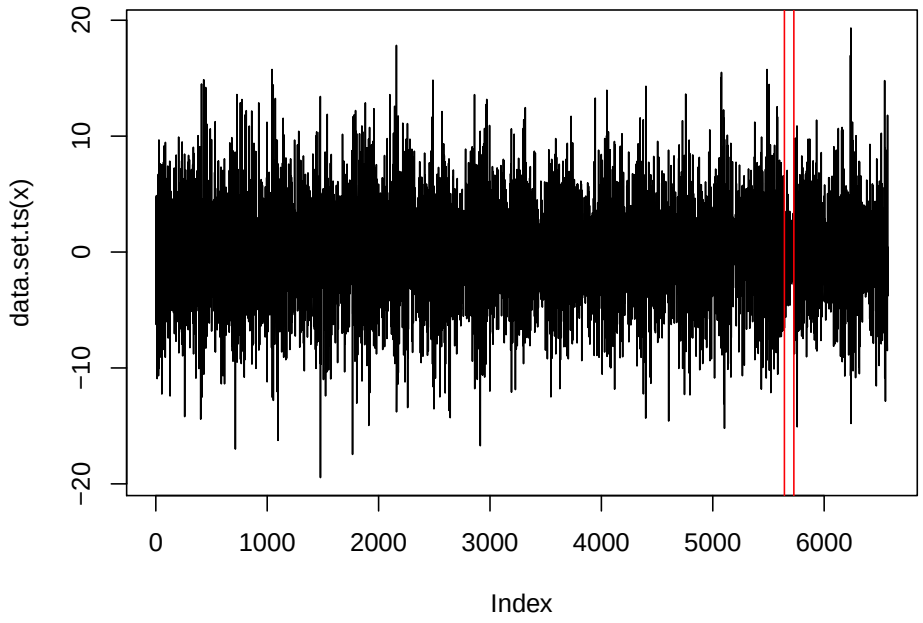


Figure 7: Variance change point

Example of multiple changes in variance at 50,100,150 in simulated data

```
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
cpt.var(x,penalty="Manual",pen.value="log(2*log(n))",method="BinSeg",test.stat="CSS",Q=5
class=FALSE) # returns optimal number of changepoints is 4, locations are 50,53,99,150.
```

Changes in mean and variance is to use the cpt.meanvar function.