



Introduction to Changepoint Analysis

Rebecca Killick(r.killick@lancs.ac.uk)
NHS Workshop 2020



- What are changepoints?
- Notation
- Likelihood based changepoints
 - Change in mean
 - Change in variance
 - Change in mean & variance
- How many changes?
- Non-parametric changepoints
- Checking assumptions (if time allows)

There will be tasks throughout the sections.

What are Changepoints?



Changepoints are also known as:

- breakpoints
- segmentation
- structural breaks
- regime switching
- detecting disorder

and can be found in a wide range of literature including

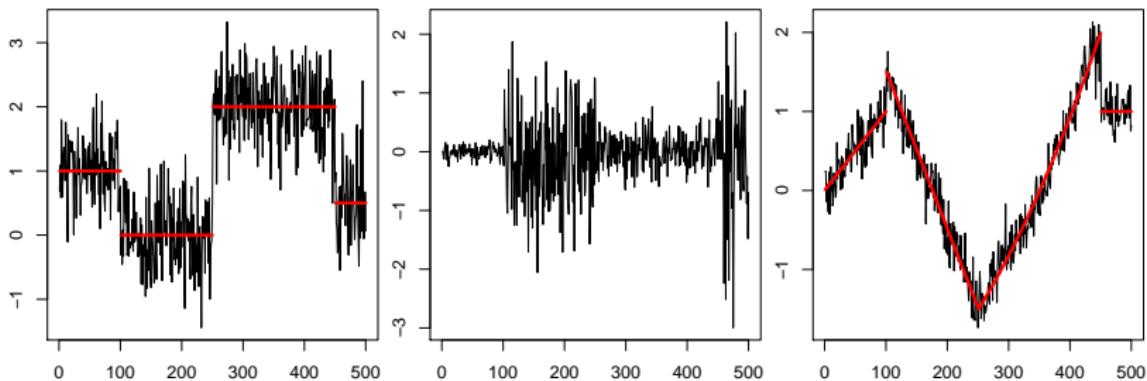
- quality control
- economics
- medicine
- environment
- linguistics
- ...

What are changepoints?



For data y_1, \dots, y_n , if a changepoint exists at τ , then y_1, \dots, y_τ differ from $y_{\tau+1}, \dots, y_n$ in some way.

There are many different types of change.

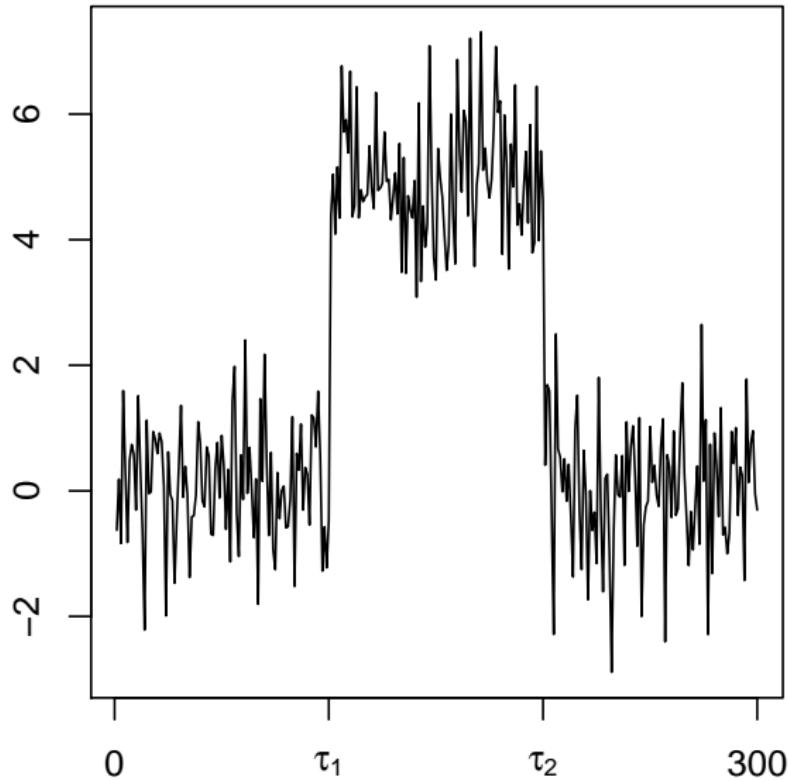


What is the goal?



- Has a change occurred?
- If yes, where is the change?
- What is the difference between the pre and post change data?
 - Maybe this is the type of change
 - Maybe it is the parameter values before and after the change
- What is the probability that a change has occurred?
- How certain are we of the changepoint location?
- How many changes have occurred (+ all the above for each change)?
- Why has there been a change?

Notation and Concepts

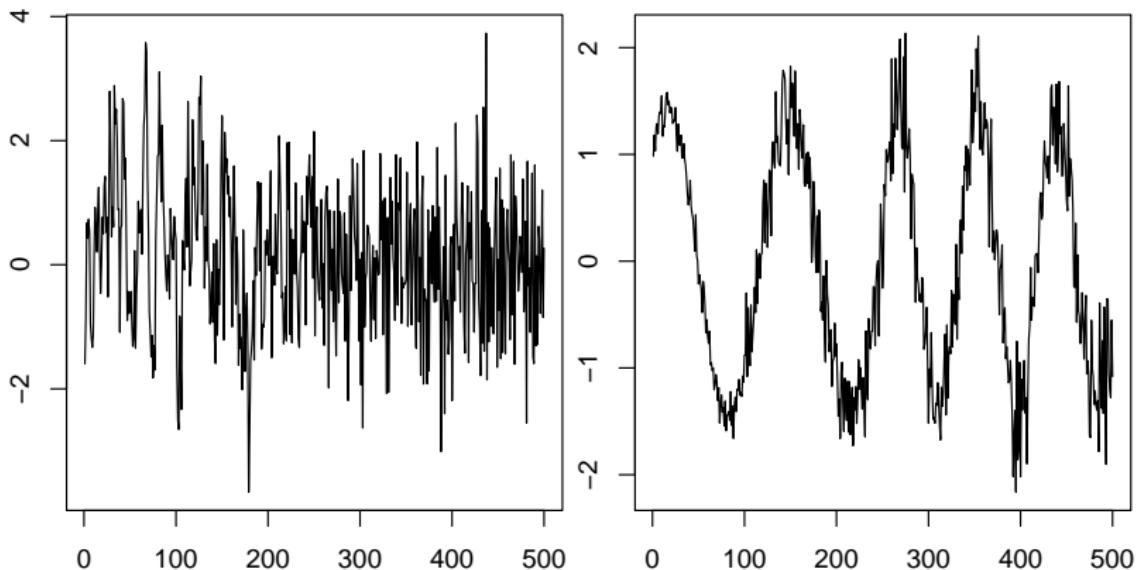




Thus a changepoint model for a change in mean has the following formulation:

$$y_t = \begin{cases} \mu_1 & \text{if } 1 \leq t \leq \tau_1 \\ \mu_2 & \text{if } \tau_1 < t \leq \tau_2 \\ \vdots & \vdots \\ \mu_{m+1} & \text{if } \tau_m < t \leq \tau_{m+1} = n \end{cases}$$

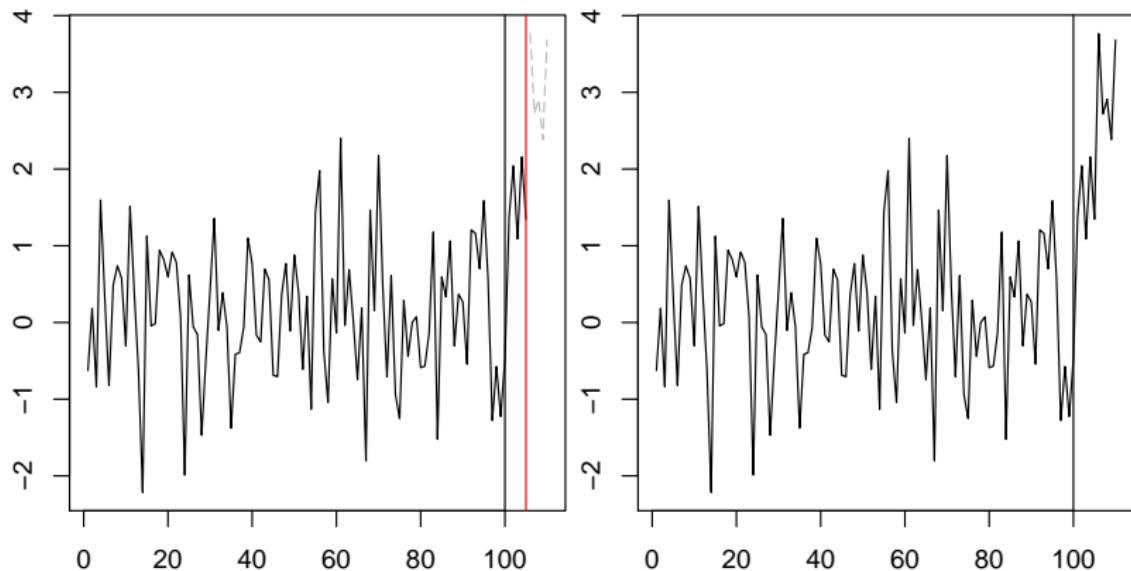
More complicated changes





- Online
 - Processes data as it arrives or in batches
 - Goal is quickest detection of a change
 - Often used in processing control, intrusion detection
- Offline
 - Processes all the data in one go
 - Goal is accurate detection of a change
 - Often used in genome analysis, audiology

Online vs Offline





Today we will use the

```
library(changepoint)
```

```
library(changepoint.np)
```

packages.

Other notable R packages are available for changepoint analysis including

- strucchange - for changes in regression
- bcp - if you want to be Bayesian
- cpm - for online changes (changepoint.online coming soon)
- EnvCpt - for testing between changes in mean, trend and/or AR(1) structure

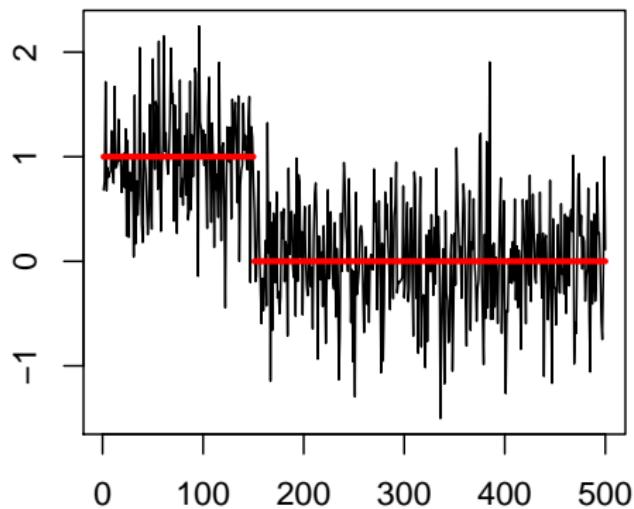
Single Changepoint



Assume we have time-series data where

$$y_t | \theta_t \sim N(\theta_t, 1),$$

but where the means, θ_t , are piecewise constant through time with a single change.

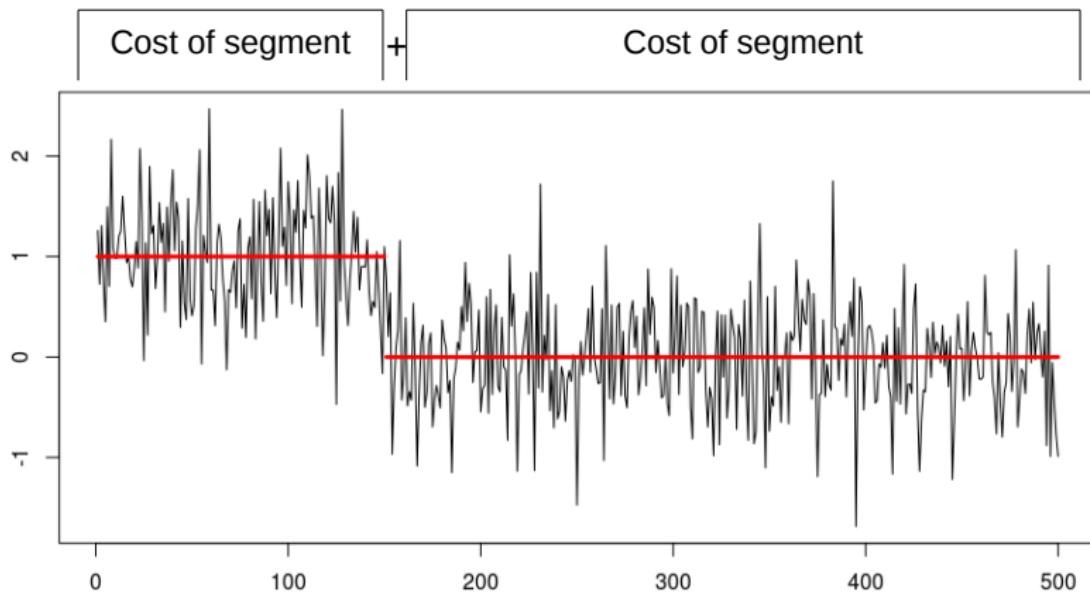


Single Changepoint



Fit for whole data

=



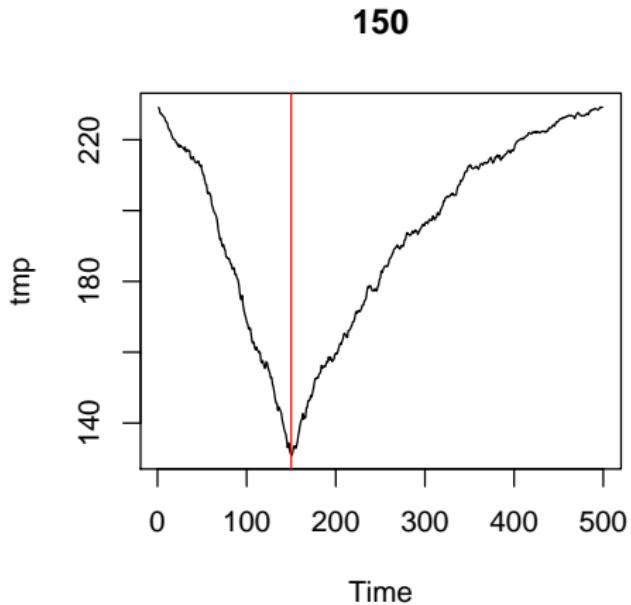
Finding a single change

Mathematics
& Statistics

Lancaster
University



Finding a single change



Finding a single change



How do we know if the changepoint found is “significant” or not?

- We also calculate the cost of the whole data with no change
- If the difference is **large enough** then we say there is a change

In practice **large enough** is hard to define as it is application dependent.

The default in the `changepoint` package is MBIC - a Modified Bayesian Information Criterion.

This will not be appropriate for all data sets! More later . . .



The changepoint R package contains 3 wrapper functions:

- cpt.mean - mean only changes
- cpt.var - variance only changes
- cpt.meanvar - mean and variance changes

The package also contains:

- functions/methods for the cpt S4 class
- 5 data sets
- other R functions that are made available for those who know what they are doing and might want to extend/modify the package.

The cpt class



- S4 class
- Slots store all the information from the analysis
 - e.g. `data.set`, `cpts`, `param.est`, `pen.value`, `ncpts.max`
- Slots are accessed via their names e.g. `cpts(x)`
- Standard methods are available for the class e.g. `plot`, `summary`
- Additional generic functions are available e.g. `seg.len`, `ncpts`
- Each core function outputs a `cpt` object



```
cpt.mean(data, penalty="MBIC", pen.value=0,  
method="AMOC", Q=5, test.stat="Normal", class=TRUE,  
param.estimates=TRUE,minseglen=1)
```

- data - vector or ts object
- penalty - cut-off point, MBIC, SIC, BIC, AIC, Hannan-Quinn, Asymptotic, Manual.
- pen.value - Type I error for Asymptotic, number or character for manual.
- method - AMOC, PELT, SegNeigh, BinSeg.
- Q - max number of changes for SegNeigh or BinSeg.
- test.stat - Test statistic, Normal or CUSUM.
- class - return a cpt object or not.
- param.estimates - return parameter estimates or not.
- minseglen - minimum number of data points between changes.

Single Change in Mean



IMPORTANT: The `cpt.mean` function assumes that the variance of a time series is 1. If this is not the case then you need to scale the data prior to analysis.

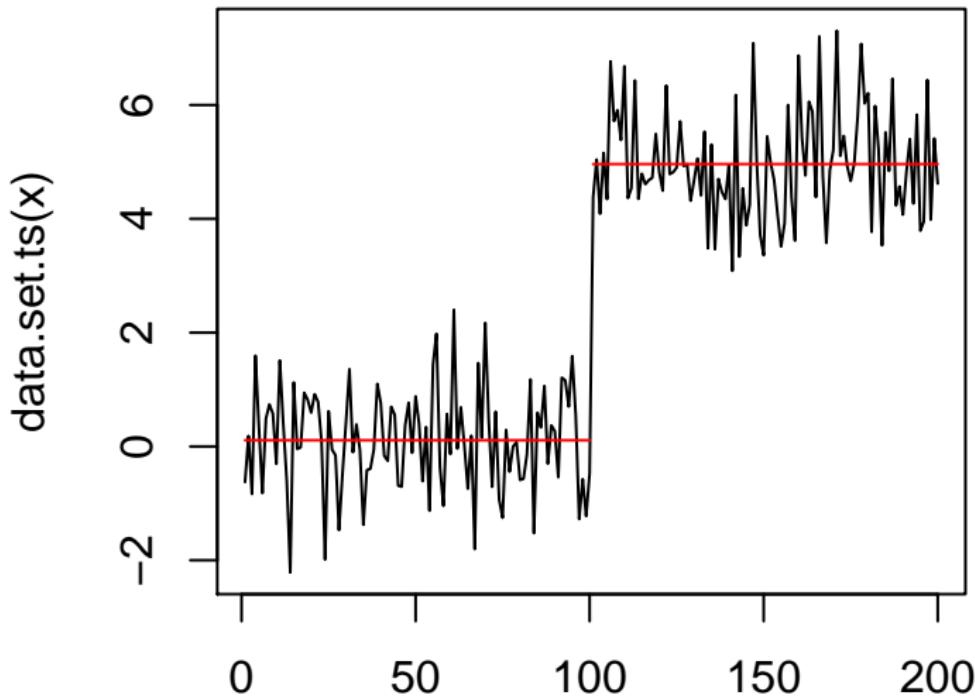
```
set.seed(1)
m1=c(rnorm(100,0,1),rnorm(100,5,1))
m1.amoc=cpt.mean(m1)
cpts(m1.amoc)
```

```
## [1] 100
```

Single Change in Mean



```
plot(m1.amoc)
```

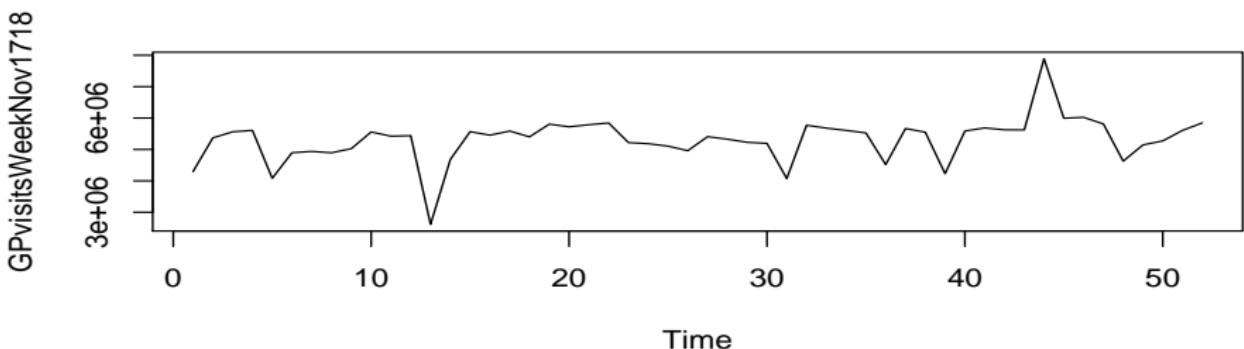


Task: GP Weekly Appts.



Data from NHS Digital, weekly number of appointments (for all types) between Nov 2017 and Oct 2018.

```
load('GPvisitsWeekNov1718.Rdata')  
ts.plot(GPvisitsWeekNov1718)
```



Is there a change?

Task: GP Weekly Appts.



Use the cpt.mean function to see if there is evidence for a change in mean in the weekly GP appointment data.

```
load('GPvisitsWeekNov1718.Rdata')
```

If you identify a change, where is it and what are the pre and post change means?

Don't forget to

```
library(changepoint)
```

before you start

Task: GP Weekly Appts.



```
GP.default=cpt.mean(GPvisitsWeekNov1718)
cpts(GP.default)
```

```
## [1] 14
```

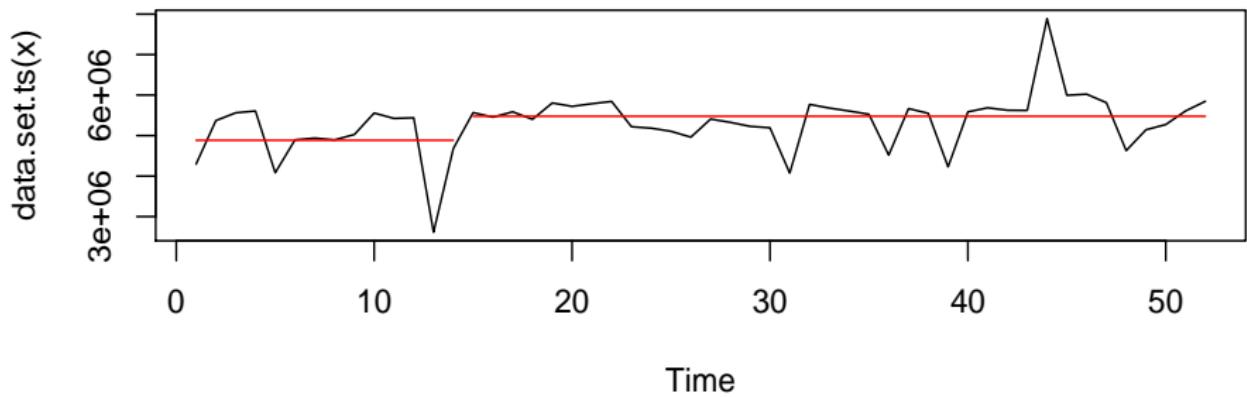
```
param.est(GP.default)
```

```
## $mean
## [1] 4884362 5477487
```

Task: GP Weekly Appts.



```
plot(GP.default)
```



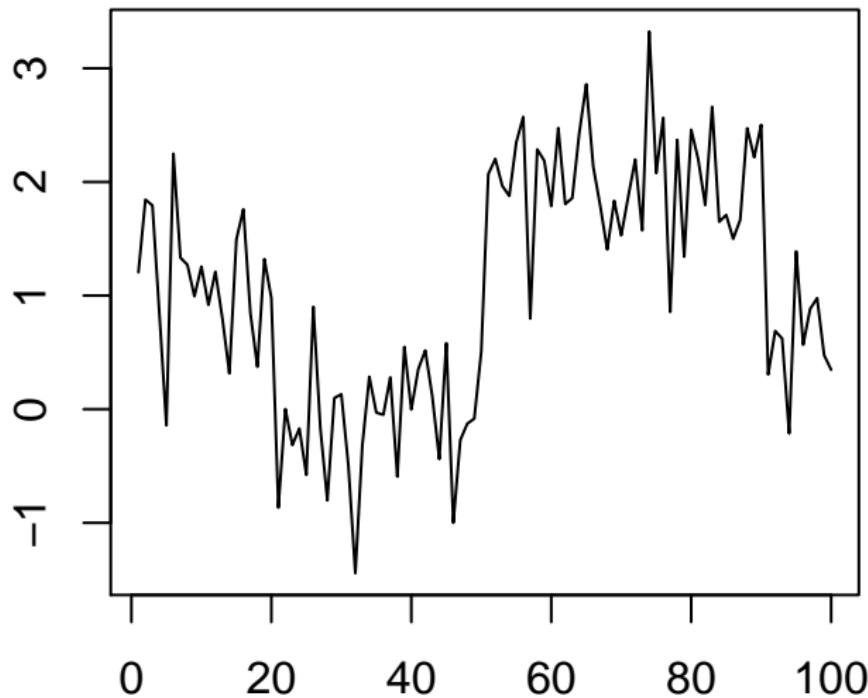
Task: GP Weekly Appts.



```
GP.scale=cpt.mean(as.vector(scale(GPvisitsWeekNov1718)))
cpts(GP.scale)

## numeric(0)
```

Multiple Changepoints



Multiple Changepoints

Mathematics
& Statistics

Lancaster
University



The Challenge



- What are the values of τ_1, \dots, τ_m ?
- What is m ?
- For n data points there are 2^{n-1} possible solutions
- If m is known there are still $\binom{n-1}{m}$ solutions
- If $n = 1000$ and $m = 10$, 2.607755×10^{23} solutions
- How do we search the solution space efficiently?



- At Most One Change (AMOC)

Approximate but computationally fast:

- Binary Segmentation (BinSeg) (Scott and Knott (1974)) which is $\mathcal{O}(n \log n)$ in CPU time.

Slower but exact:

- Segment Neighbourhood (SegNeigh) (Auger and Lawrence (1989)) is $\mathcal{O}(Qn^2)$.

Fast and exact:

- Pruned Exact Linear Time (PELT) (Killick et al. (2012)) At worst $\mathcal{O}(n^2)$. For linear penalties , scaling changes, $\mathcal{O}(n)$.



```
cpt.var(data, penalty, pen.value, know.mean=FALSE, mu=NA,  
method, Q, test.stat="Normal", class, param.estimates,  
minseglen=2)
```

Majority of arguments are the same as for cpt.mean

- know.mean - if known we don't count it as an estimated parameter when calculating penalties.
- mu - Mean if known.
- test.stat - Normal or CSS (cumulative sums of squares)
- minseglen - Default is 2

Changes in Variance



```
set.seed(1)
v1=c(rnorm(100,0,1),rnorm(100,0,2),rnorm(100,0,10),
     rnorm(100,0,9))
v1.man=cpt.var(v1,method='PELT',penalty='Manual',
                pen.value='2*log(n)')
cpts(v1.man)
param.est(v1.man)
```

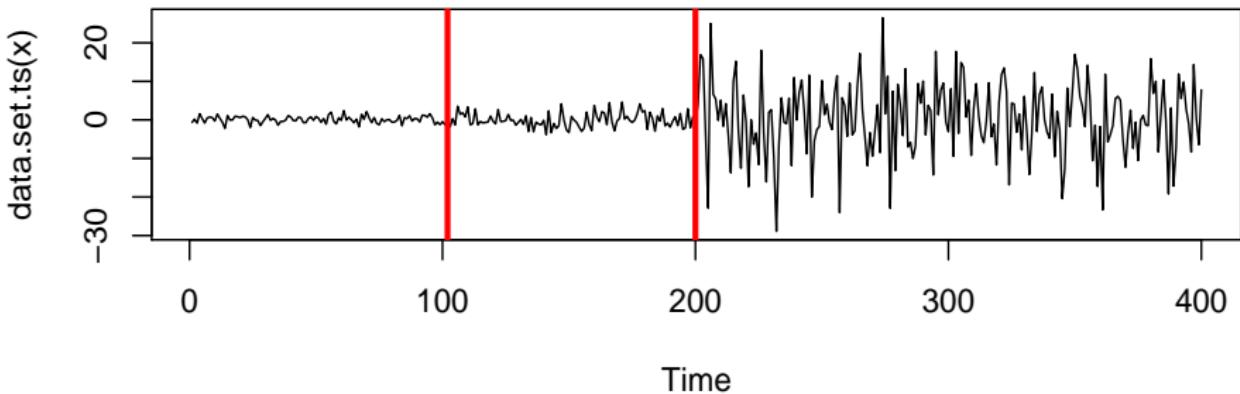
```
## [1] 102 200
## $variance
## [1] 0.8007158 3.6933616 92.3876410
##
## $mean
## [1] 0.1986058
```

Changes in Variance



Ratios of true variances (4, 25, 0.81)

```
plot(v1.man, cpt.width=3)
```





```
cpt.meanvar(data, penalty, pen.value, method, Q,  
test.stat="Normal", class, param.estimates,  
shape=1,minseglen=2)
```

Again the same underlying structure as cpt.mean.

- test.stat - choice of Normal, Gamma, Exponential, Poisson.
- shape - assumed shape parameter for Gamma.
- minseglen - minimum segment length of 2

Mean & Variance



```
set.seed(1)
mv1=c(rexp(50,rate=1),rexp(50,5),rexp(50,2),rexp(50,7))
mv1.pelt=cpt.meanvar(mv1,test.stat='Exponential',
                      method='BinSeg',Q=10,penalty="SIC")
cpts(mv1.pelt)

## [1] 50 100 150

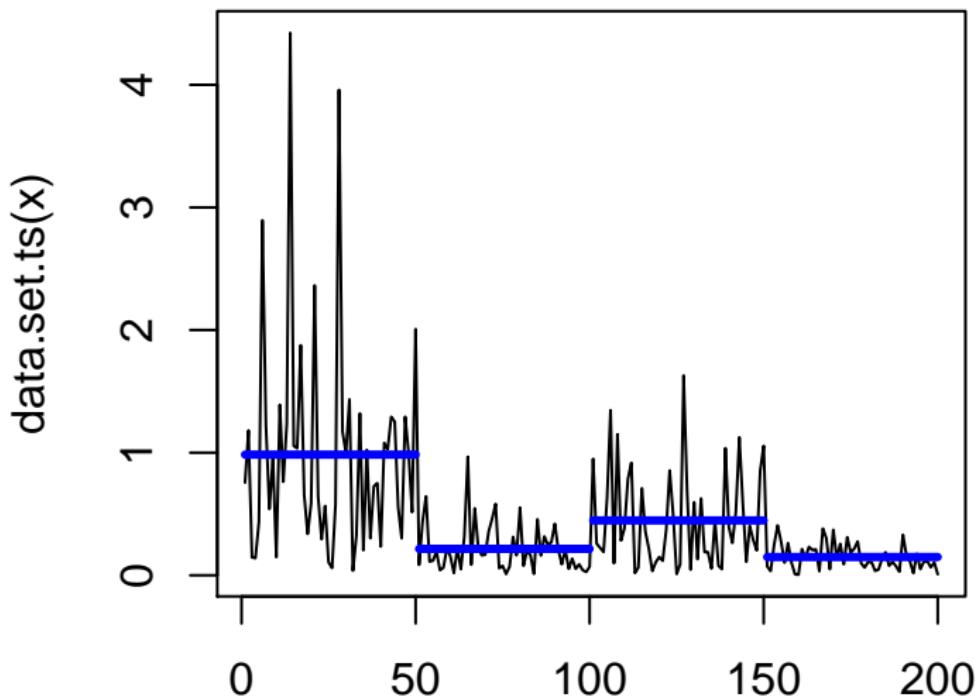
param.est(mv1.pelt)

## $rate
## [1] 1.016217 4.641184 2.235431 6.705612
```

Mean & Variance



```
plot(mv1.pelt,cpt.width=3,cpt.col='blue')
```



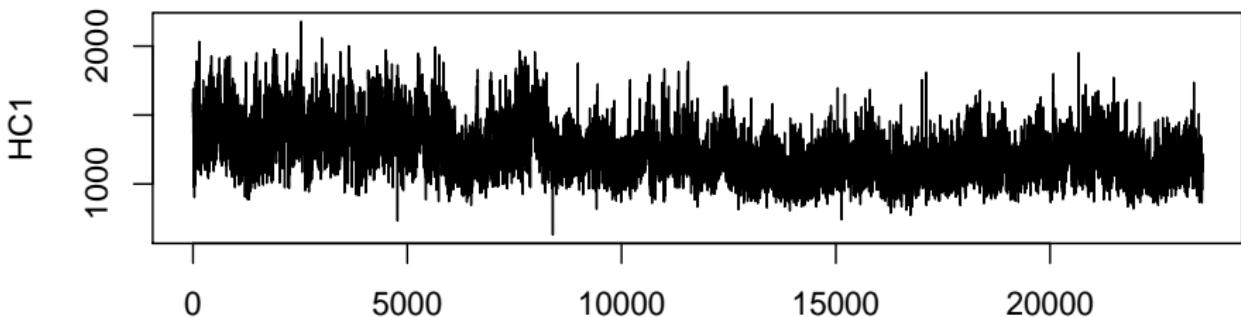
Task HC1



G+C content within part of Human Chromosome 1, data from NCBI.
3kb windows along the Human Chromosome from 10Mb to 33Mb.

Use the cpt.meanvar function to identify regions with different C+G content.

```
data(HC1)  
ts.plot(HC1)
```



A solution: HC1



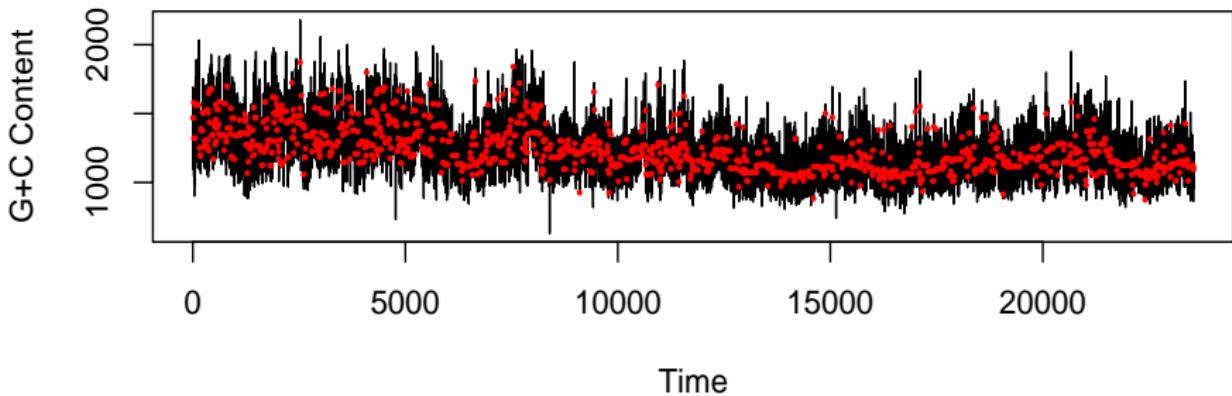
```
data(HC1)
hc1.pelt=cpt.meanvar(HC1,method='PELT',penalty='Manual',
                      pen.value=14)
ncpts(hc1.pelt)

## [1] 805
```

A solution: HC1



```
plot(hc1.pelt, ylab='G+C Content', cpt.width=3)
```

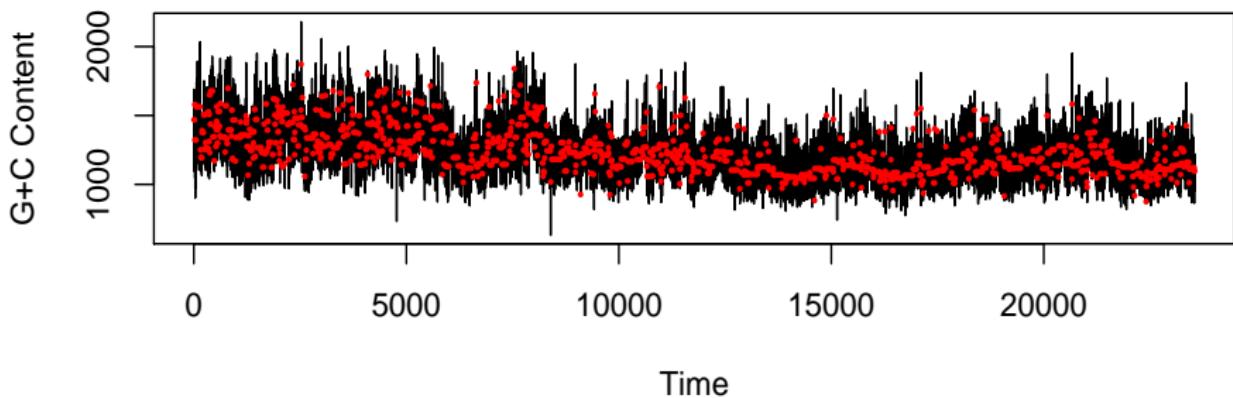


Number of changes?



Does the number of changes appear reasonable?

```
plot(hc1.pelt, ylab='G+C Content', cpt.width=3)
```





Changepoints for a range of penalties

Use `penalty='CROPS'` with `method='PELT'` to get all segmentations for a range of penalty values.

```
v1.crops=cpt.var(v1,method="PELT",penalty="CROPS",  
pen.value=c(5,500))
```

```
## [1] "Maximum number of runs of algorithm = 10"  
## [1] "Completed runs = 2"  
## [1] "Completed runs = 3"  
## [1] "Completed runs = 4"  
## [1] "Completed runs = 5"  
## [1] "Completed runs = 6"  
## [1] "Completed runs = 8"  
## [1] "Completed runs = 9"
```

```
cpts.full(v1.crops)
```

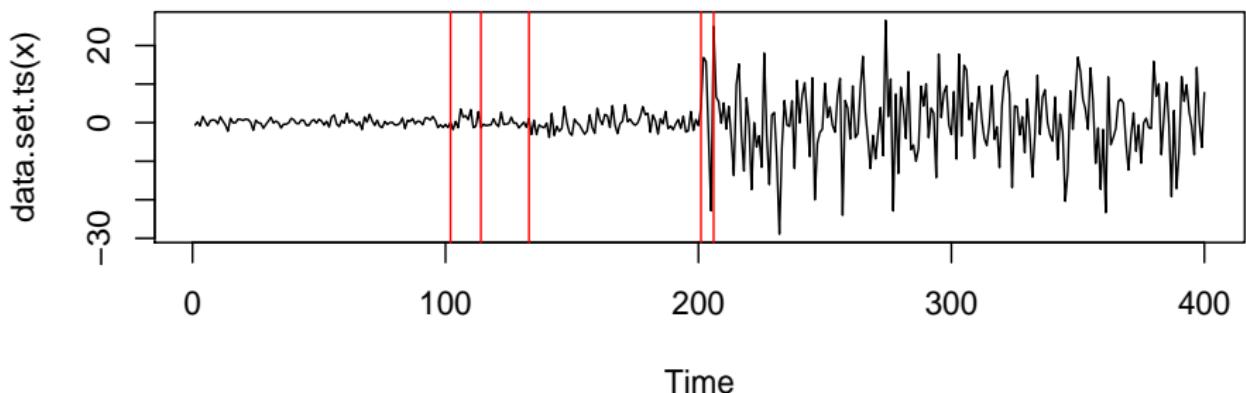
```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,] 102  114  133  201  206  213  375  379
## [2,] 102  114  133  201  206  375  379   NA
## [3,] 102  114  133  201  206   NA   NA   NA
## [4,]  96  133  201  206   NA   NA   NA   NA
## [5,] 102  201  206   NA   NA   NA   NA   NA
## [6,] 102  200   NA   NA   NA   NA   NA   NA
## [7,] 200   NA   NA   NA   NA   NA   NA   NA
## [8,]   NA   NA   NA   NA   NA   NA   NA   NA
```



```
pen.value.full(v1.crops)
```

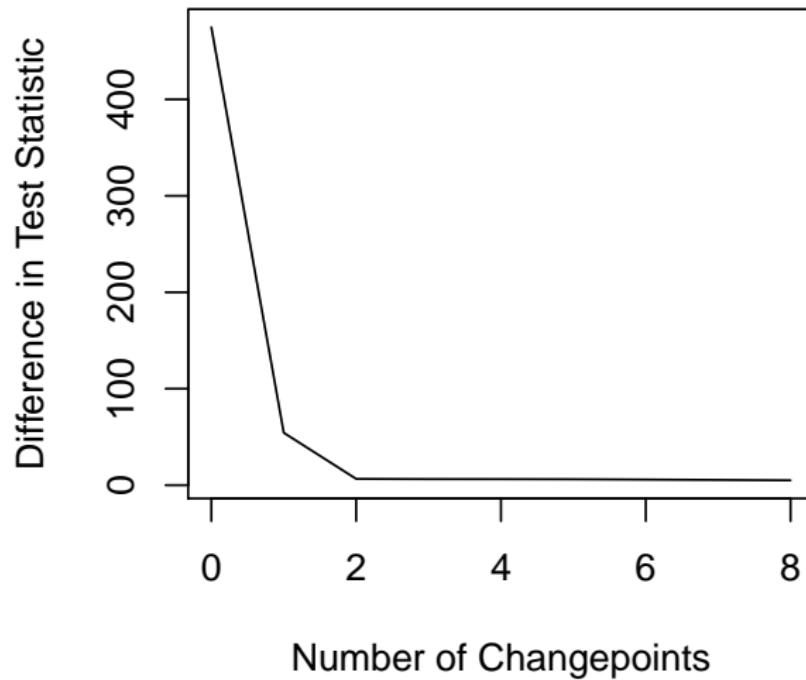
```
## [1] 5.000000 5.431360 6.151053 6.270164 6.314013
## [8] 474.797364
```

```
plot(v1.crops,ncpts=5)
```





```
plot(v1.crops, diagnostic=TRUE)
```





```
cpt.np(data, penalty, pen.value, method,  
test.stat="empirical_distribution", class, minseglen=1,  
nquantiles=10)
```

Again the same underlying structure as cpt.mean.

- test.stat - choice of empirical_distribution
- minseglen - minimum segment length of 1
- nquantiles - number of quantiles to use

Example



```
set.seed(12)
J <- function(x){(1+sign(x))/2}
n <- 1000
tau <- c(0.1,0.13,0.15,0.23,0.25,0.4,0.44,0.65,0.76,0.78,
        0.81)*n
h <- c(2.01, -2.51, 1.51, -2.01, 2.51, -2.11, 1.05, 2.16,
       -1.56, 2.56, -2.11)
sigma <- 0.5
t <- seq(0,1,length.out = n)
data <- array()
for (i in 1:n){
  data[i] <- sum(h*J(n*t[i] - tau)) + (sigma * rnorm(1))
}
```

Example

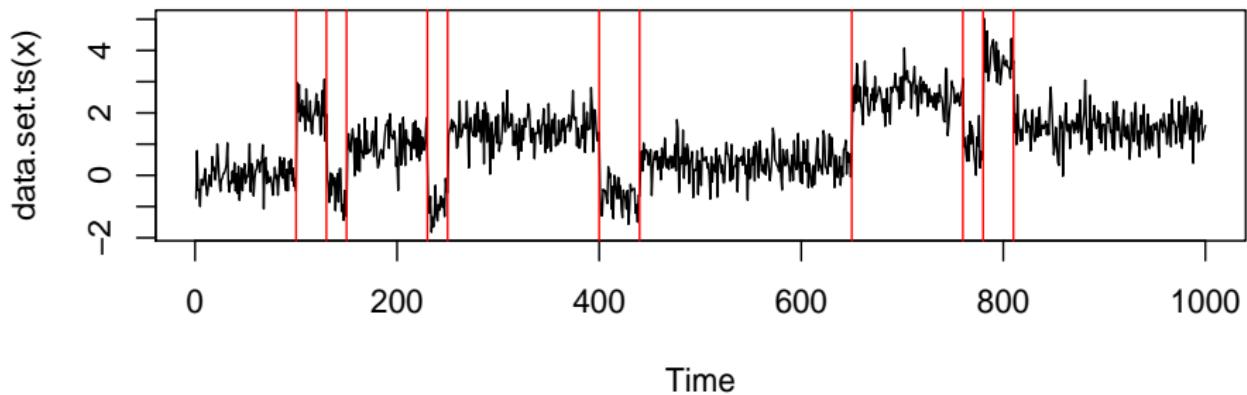


```
out <- cpt.np(data, method="PELT", minseglen=2,  
                nquantiles =4*log(length(data)))  
cpts(out)  
  
## [1] 100 130 150 230 250 400 440 650 760 780 810
```

Example



```
plot(out)
```





Look at the HeartRate data from the `changepoint.np` package. Use one of the non-parametric functions to see if there is evidence for changes in heart rate.

```
data(HeartRate)
```

A solution: HeartRate



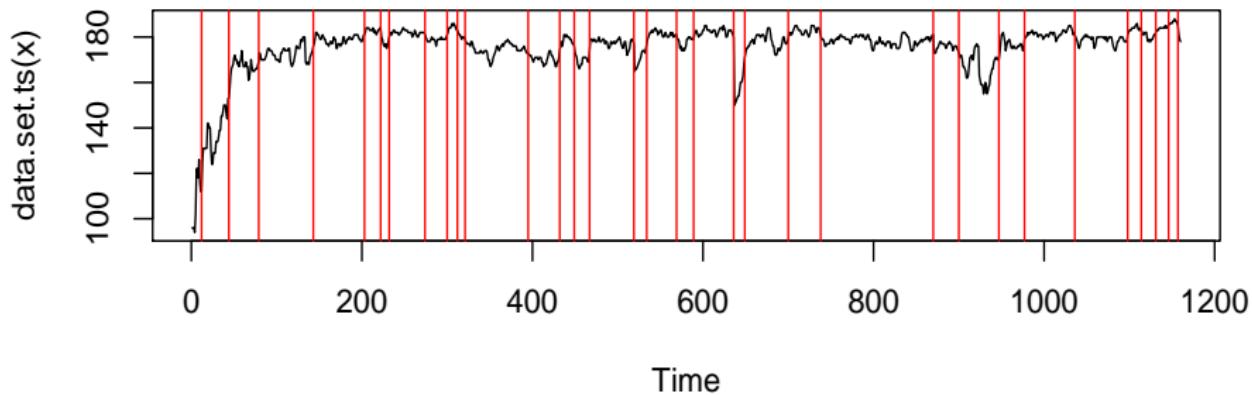
```
HR.pelt=cpt.np(HeartRate,method='PELT',
                 nquantiles=4*log(length(HeartRate)))
ncpts(HR.pelt)

## [1] 33
```

A solution: HeartRate



```
plot(HR.pelt)
```



A solution: HeartRate



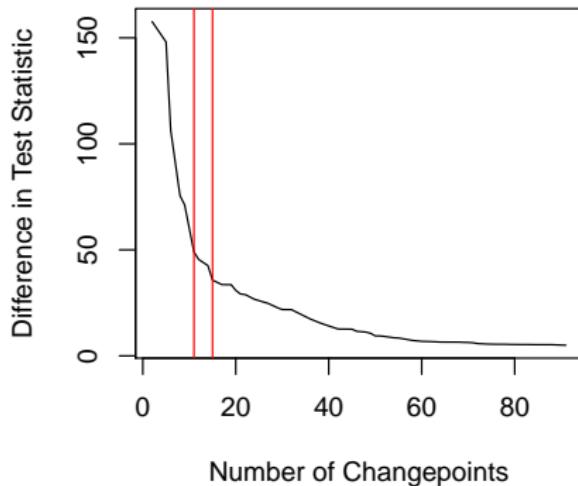
```
HR.crops=cpt.np(HeartRate, penalty = "CROPS",
                 pen.value = c(5,200), method="PELT",minseglen=2,
                 nquantiles =4*log(length(HeartRate)))
```

```
## [1] "Maximum number of runs of algorithm = 91"
## [1] "Completed runs = 2"
## [1] "Completed runs = 3"
## [1] "Completed runs = 5"
## [1] "Completed runs = 9"
## [1] "Completed runs = 17"
## [1] "Completed runs = 32"
## [1] "Completed runs = 52"
## [1] "Completed runs = 75"
## [1] "Completed runs = 84"
```

A solution: HeartRate



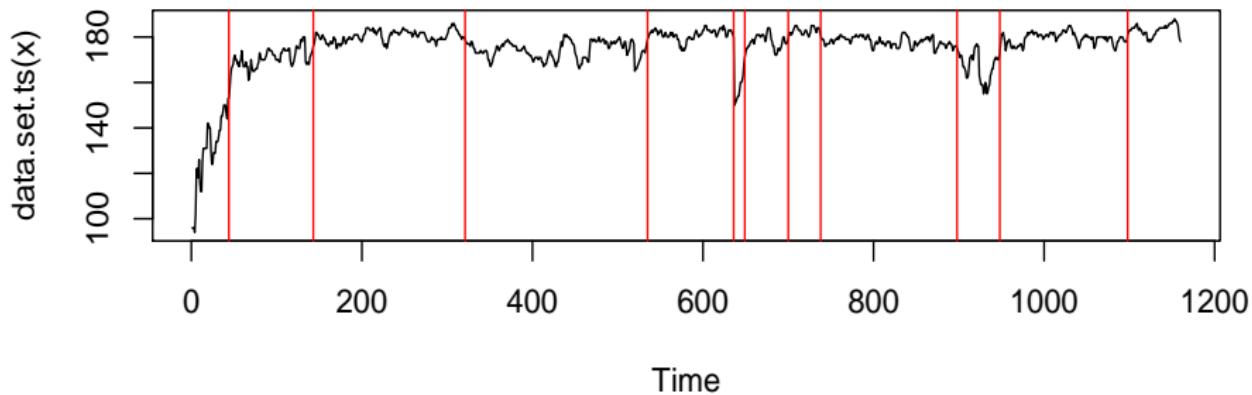
```
plot(HR.crops, diagnostic = TRUE)  
abline(v=11,col='red')  
abline(v=15,col='red')
```



A solution: HeartRate



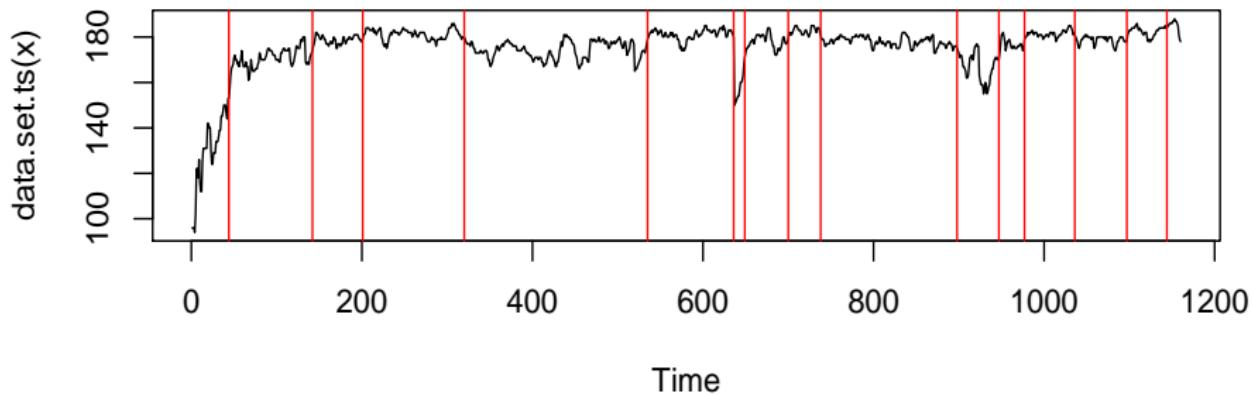
```
plot(HR.crops, ncpts = 11)
```



A solution: HeartRate



```
plot(HR.crops, ncpts = 15)
```





The main assumptions for a Normal likelihood ratio test for a change in mean are:

- Independent data points;
- Normal distributed points pre and post change;
- Constant variance across the data.

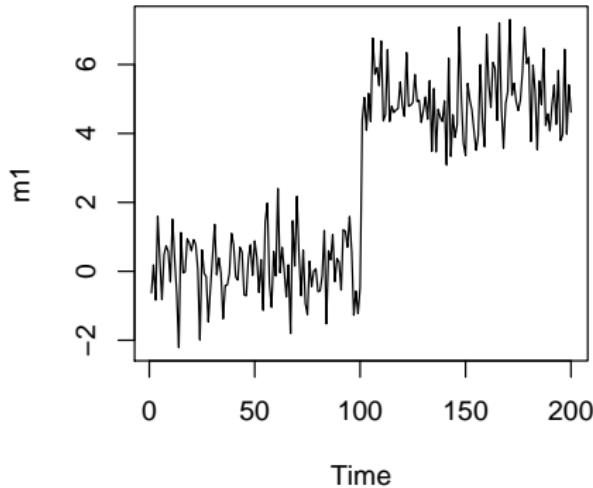
How can we check these?

Checking Assumptions



In reality we can't check assumptions prior to analysis.

```
ts.plot(m1)
```

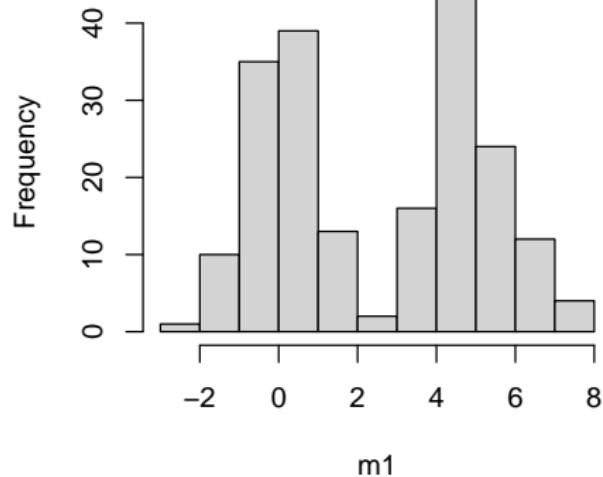


Checking Assumptions



```
hist(m1)
```

Histogram of m1



Checking Assumptions



```
shapiro.test(m1)
```

```
##  
##  Shapiro-Wilk normality test  
##  
## data: m1  
## W = 0.91086, p-value = 1.31e-09
```

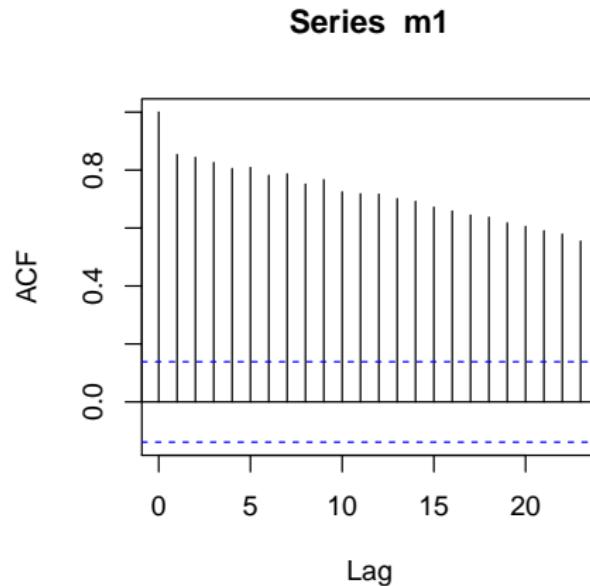
```
ks.test(m1,pnorm,mean=mean(m1),sd=sd(m1))
```

```
##  
##  One-sample Kolmogorov-Smirnov test  
##  
## data: m1  
## D = 0.15491, p-value = 0.0001355  
## alternative hypothesis: two-sided
```

Checking Assumptions



`acf(m1)`



How to check



- Check the residuals

```
means=param.est(m1.amoc)$mean
m1.resid=m1$rep(means, seg.len(m1.amoc))
shapiro.test(m1.resid)
```

```
##
##  Shapiro-Wilk normality test
##
## data: m1.resid
## W = 0.99228, p-value = 0.3721
```

Residual Check



```
ks.test(m1.resid, pnorm, mean=mean(m1.resid), sd=sd(m1.resid))

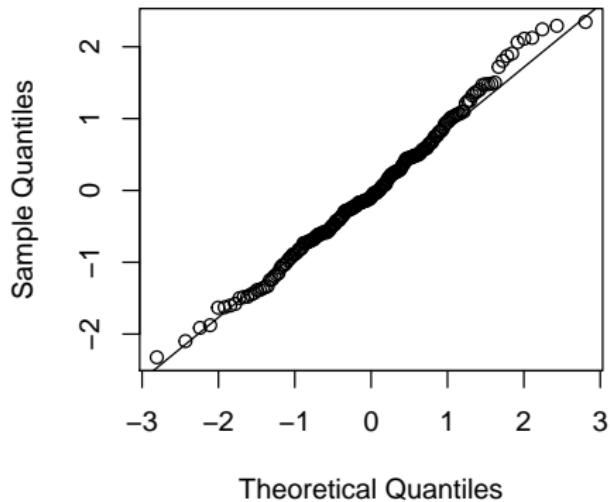
##
##  One-sample Kolmogorov-Smirnov test
##
## data: m1.resid
## D = 0.045812, p-value = 0.7953
## alternative hypothesis: two-sided
```

Residual Check



```
qqnorm(m1.resid)  
qqline(m1.resid)
```

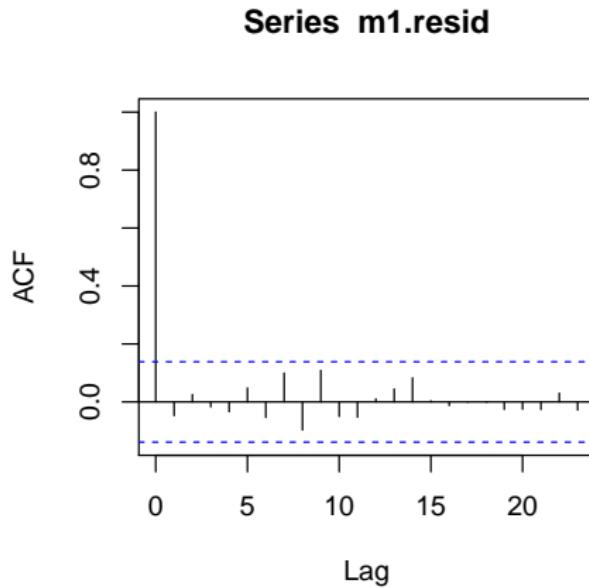
Normal Q-Q Plot



Residual Check



```
acf(m1.resid)
```



Task



Check the assumptions you have made on the simulated, , GP Appointments, HC1 and HeartRate data using the residual check.

What effect might any invalid assumptions have on the inference?

Consolidating Task



Download the ratings for the following TV shows from the IMDB and analyze the series using some of the techniques you have learnt from today. For each series, do you identify any changes? Are the assumptions you are making valid? What effect might any invalid assumptions have on the inference?

- Doctor Who
- Grey's Anatomy
- Mistresses
- The Simpsons
- Top Gear

(Understandably IMBD does not allow screen scraping nor downloads of information for redistribution so you will have to copy and paste the table into Excel, or equivalent, yourself in order to get the ratings data into R.)



Just from looking at the data, can you predict which shows have been canceled?



JSS: Killick, Eckley (2014)

PELT: Killick, Fearnhead, Eckley (2012)

CROPS: Kaynes, Eckley, Fearnhead (2015)

cpt.np: Haynes, Fearnhead, Eckley (2016)