Introduction to change point detection

Computer lab worksheet

Euan McGonigle

Instructions

- Answers are available to you at: if you feel you're spending too much time on any questions particularly Question 1, you can copy-paste the code to move onto questions 2 – 4.
- Bonus questions: Q 1(e) is a bonus, as is Question 5. Feel free to skip these if you want.
- For any functions you're not sure how to use, use `?function_name` in R to get help (or ask me!).
- In the time we have, I don't expect you to finish all questions. If you can, try to finish at least Q2.

Question 1: Creating a test statistic to estimate change points

(a) Without using pre-existing change point detection libraries, write a function to calculate **either**:

- the CUSUM statistic vector $\{T_{0,k,n}\}_{k=1}^{n-1}$ for a change point in the mean,
- the MOSUM statistic vector $\{T_G(k)\}_{k=G}^{n-G}$ for a change point in the mean, for a given bandwidth G.

Solution:

```
CUSUM.calc <- function(x){

  n <- length(x)
  I.plus <- I.minus <- I.prod <- rep(0, n - 1)
  I.plus[1] <- sqrt(1 - 1/n) * x[1]
  I.minus[1] <- 1/sqrt(n^2 - n) * sum(x[2:n])
```

```
  for (k in 1:(n - 2)) {
      factor <- sqrt((n - k - 1) * k/(k + 1)/(n - k))
      I.plus[k + 1] <- I.plus[k] * factor + x[k + 1] * sqrt(1/(k + 1) - 1/n)
      I.minus[k + 1] <- I.minus[k]/factor - x[k + 1]/sqrt(n^2/(k + 1) - n)
  }
  x.CUSUM <- I.plus - I.minus
  return(abs(x.CUSUM))

}
```

```
MOSUM.calc <- function(x, G){

  n <- length(x)

  sums <- rep(NA, n)
  currentSum <- sum(x[1:G])

  sums[1] <- currentSum
  for (k in 2:(n-G)) {
    currentSum <- currentSum + x[k + G - 1]
    currentSum <- currentSum - x[k - 1]
    sums[k] <- currentSum
  }

  x.MOSUM  <- c(rep(NA, G - 1), sums[(G + 1):n] - sums[1:(n - G)], NA)/sqrt(2*G)

  return(abs(x.MOSUM))

}
```

(b) To set a threshold for declaring changes, we need to know the noise level σ. In reality, this is unknown; how could we estimate it?

Hint: the sample standard deviation would be positively biased by the change points. Medians are more robust than means: how can we transform the data to remove most of the effect of the change points, before using a median-like analogue of the standard deviation? If in doubt, Google is your friend.

Solution: We want a robust estimator of σ that is unaffected by change points. One option is using the median absolute deviation: the median of the absolute deviation from the median. To do this in the presence of change points, we can take the (scaled) first differences of change points, so that at a change point, the data looks like an outlier. Then, the MAD of this series gives a robust estimator.

This is one of many options. You could:

- use the standard deviation, but only using a small portion of data at the beginning.
- use other robust methods such as influence functions.
- use local estimators (see the mosum R package) like $\hat{\sigma}^2(k) = (\hat{\sigma}^2_{(k-G+1):(k)} + \hat{\sigma}^2_{(k+1):(k+G)})/2$, where $\hat{\sigma}^2_{s:e}$ denotes the sample variance calculated on the data from start s to end e.

(c) Implement your chosen method of estimating σ, and add the functionality to your CUSUM or MOSUM function from part (a) so that the test statistic calculation is scaled by your estimate $\hat{\sigma}$.

Solution:

```
sigma.est <- function(x){

  x.diff <- diff(x)/sqrt(2) #scaled so that var(x) = var(transformed x)

  sigma.hat <- mad(x.diff)

  return(sigma.hat)

}

#answer for the MOSUM method:

MOSUM.calc2 <- function(x, G){

  x.MOSUM <- MOSUM.calc(x, G)
  sigma.hat <- sigma.est(x)

  return(x.MOSUM/sigma.hat)
}


CUSUM.calc2 <- function(x){

  x.CUSUM <- CUSUM.calc(x)
  sigma.hat <- sigma.est(x)

  return(x.CUSUM/sigma.hat)

}
```

(d) What is the computational complexity of the code you wrote in part (a)? How fast could it be?

Solution: By using sequential updates, both methods can be computed on $O(n)$ computational compelxity.

(e) (Bonus) For simultaneous estimation of multiple change points with the MOSUM approach, we can calculate all $\hat{\theta}$ that satisfy:

$$T_G(\hat{\theta}) > D \quad \text{and} \quad \hat{\theta} = \text{argmax}_{k:\,|k-\hat{\theta}|\leq \eta G} T_G(k).$$

for some window fraction $\eta \in (0,1)$ and a threshold D. That is, $\hat{\theta}$ is declared a change point if it is a local maximiser of $T_G(k)$ over a sufficiently large interval of radius ηG, at which the threshold D is exceeded.

Extend the MOSUM function to return as output the change point estimators satisfying this criterion.

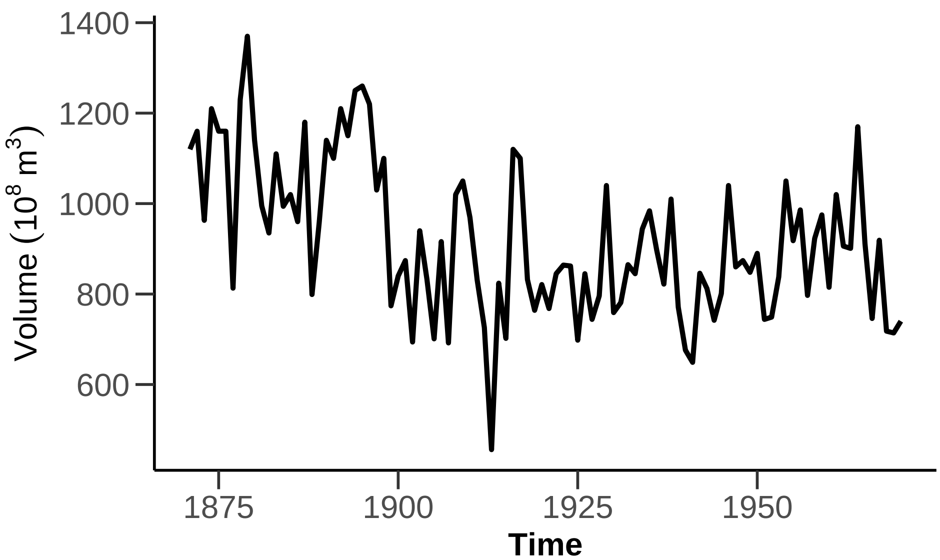
Solution:

```
MOSUM.calc3 <- function(x, G, eta, D){

  x.MOSUM <- MOSUM.calc2(x, G)

  n <- length(x)
  cpt.ests <- numeric(0)
  window_length <- floor(eta*G)

  exceedings <- (x.MOSUM > D)

  localMaxima <- (c((diff.default(x.MOSUM) < 0), NA) & c(NA, diff.default(x.MOSUM) > 0))
  candidates <- which(exceedings & localMaxima)

  for (j in seq_len(length(candidates))){

    k_star <- candidates[j]
    m_star <- x.MOSUM[k_star]
    left_thresh <- max(G, k_star - window_length)
    right_thresh <- min(n-G, k_star + window_length)
    largest <- TRUE
    for (l in left_thresh:right_thresh) {
      if (x.MOSUM[l] > m_star) {
        largest <- FALSE
        break
```

```
      }
    }
    if (largest) {
      cpt.ests <- c(cpt.ests, k_star)
    }
  }

  return(cpt.ests)
}
```

Question 2: Nile annual river flow

In this question you will use the code you've written so far to analyse data collected on the river Nile. Data in the file `nile_volume.txt` records measurements of the annual volume (in units $10^8 m^3$) of discharge from the Nile River at Aswan for the years 1871 to 1970. The measurements are of meteorological importance as evidence of a possible abrupt change in the rainfall levels around the turn of the 20th century.

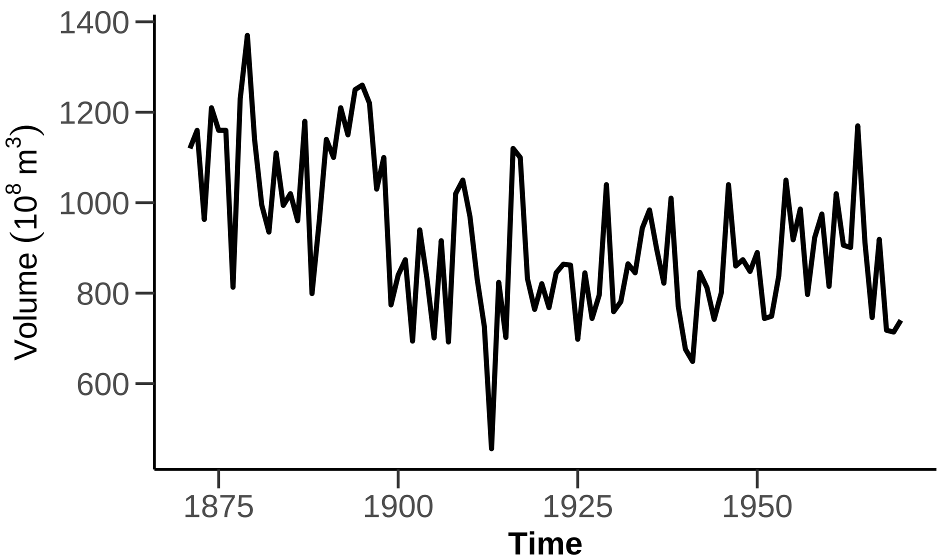


For whichever method (CUSUM or MOSUM) you did not code up in Q1, you can use toe code in the answers for this question.

(a) Load the data into R from the `nile_volume.txt` file, and plot it. By eye, where does it look like there could be a change in mean?

Solution:

```
library(ggplot2)

nile.data <- read.table("/Users/euanmcgonigle/Documents/Southampton-Work/Software/Rough Code/
                        header = TRUE)

p1 <- ggplot(data = nile.data, aes(x=year,y=volume))+
  geom_line(data=nile.data,color="black",linewidth=0.9)+
  theme_classic()+
  labs(x="Time",y=expression(Volume~(10^8~m^3))) +
  theme(axis.text=element_text(size=12),axis.ticks.length=unit(.25, "cm"),
        axis.title=element_text(size=12),title = element_text(size=18,face="bold"))

p1
```



Around 1900 it looks like there could be a drop in the volume of river flow.

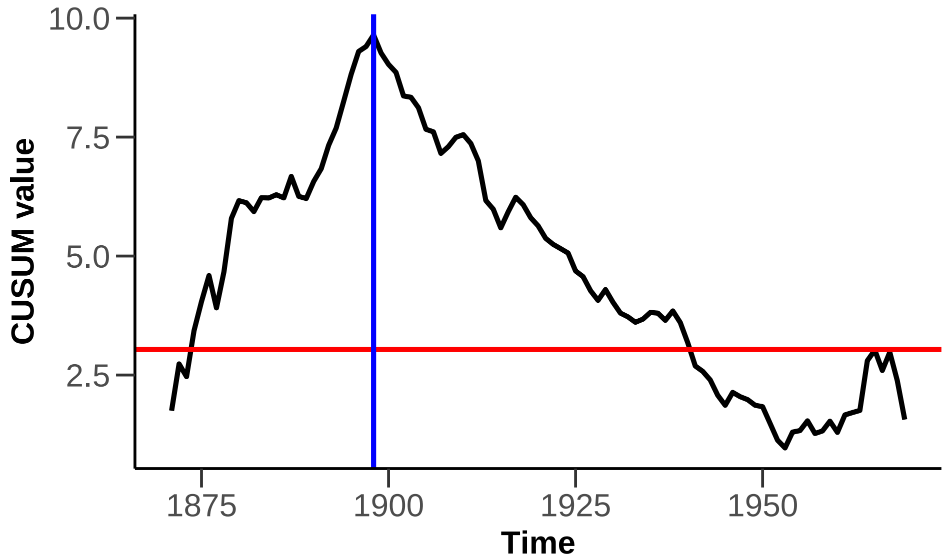
(b) Calculate the CUSUM statistic for the Nile volume data. Using the threshold $D = \sqrt{2\log(n)}$, perform a test to decide if there is a change point. If there is one, where is it?

Solution:

```r
nile.CUSUM <- CUSUM.calc2(nile.data$volume)

max(nile.CUSUM) > sqrt(2 * log(nrow(nile.data))) # check if above threshold
```

```
[1] TRUE
```

```r
nile.cpt <- which.max(nile.CUSUM)

CUSUM.df <- data.frame(year = nile.data$year[1:99], CUSUM = nile.CUSUM)

CUSUM.plot <- ggplot(data = CUSUM.df, aes(x=year,y=CUSUM))+
  geom_line(data=CUSUM.df,color="black",linewidth=0.9)+
  theme_classic()+
  labs(x="Time",y="CUSUM value") +
  theme(axis.text=element_text(size=12),axis.ticks.length=unit(.25, "cm"),
        axis.title=element_text(size=12),title = element_text(size=18,face="bold"))+
    geom_hline(yintercept = sqrt(2*log(100)), color = "red", linewidth =0.9)+
    geom_vline(xintercept = CUSUM.df$year[nile.cpt], color = "blue", linewidth =0.9)


CUSUM.plot
```



The change point is located at year 1898.

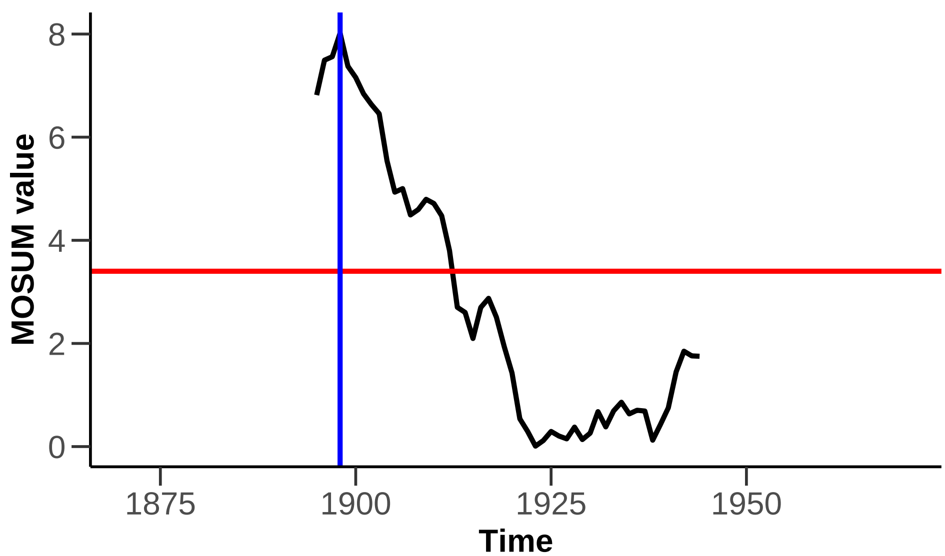
(c) Calculate the MOSUM statistic for the Nile volume data using a bandwidth $G = 25$. Using the threshold $D = 3.4$, compute the change point estimators:

- either by eye by plotting the MOSUM test statistic, or
- using the code from Q1 (e) with $\eta = 0.4$.

Does this agree with your answer from part (b)?

Solution:

```
nile.MOSUM <- MOSUM.calc2(nile.data$volume, G = 25)

nile.cpt2 <- which.max(nile.MOSUM)

MOSUM.df <- data.frame(year = nile.data$year, MOSUM = nile.MOSUM)

MOSUM.plot <- ggplot(data = MOSUM.df, aes(x=year,y=MOSUM))+
  geom_line(data=MOSUM.df,color="black",linewidth=0.9)+
  theme_classic()+
  labs(x="Time",y="MOSUM value") +
  theme(axis.text=element_text(size=12),axis.ticks.length=unit(.25, "cm"),
        axis.title=element_text(size=12),title = element_text(size=18,face="bold"))+
    geom_hline(yintercept = 3.4, color = "red", linewidth =0.9)+
    geom_vline(xintercept = MOSUM.df$year[nile.cpt2], color = "blue", linewidth =0.9)


MOSUM.plot
```

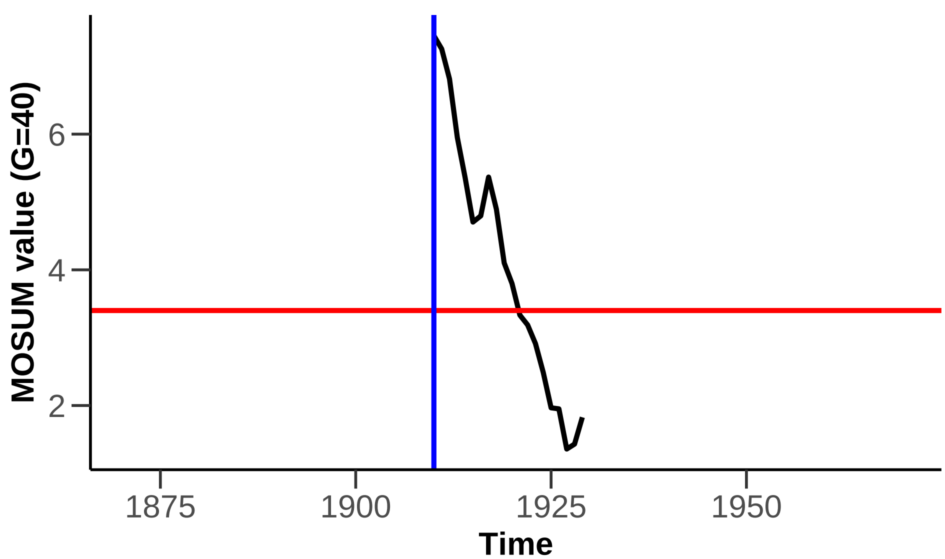


The MOSUM change point is estimated at year 1898, which agrees with the CUSUM estimate.

(d) Now use the MOSUM method with bandwidth $G = 40$. How does your answer differ from part (c)?

Solution:

```
nile.MOSUM <- MOSUM.calc2(nile.data$volume, G = 40)

nile.cpt <- which.max(nile.MOSUM)

MOSUM.df <- data.frame(year = nile.data$year, MOSUM = nile.MOSUM)

MOSUM.plot <- ggplot(data = MOSUM.df, aes(x=year,y=MOSUM))+
  geom_line(data=MOSUM.df,color="black",linewidth=0.9)+
  theme_classic()+
  labs(x="Time",y="MOSUM value (G=40)") +
  theme(axis.text=element_text(size=12),axis.ticks.length=unit(.25, "cm"),
        axis.title=element_text(size=12),title = element_text(size=18,face="bold"))+
    geom_hline(yintercept = 3.4, color = "red", linewidth =0.9)+
    geom_vline(xintercept = MOSUM.df$year[nile.cpt], color = "blue", linewidth =0.9)


MOSUM.plot
```



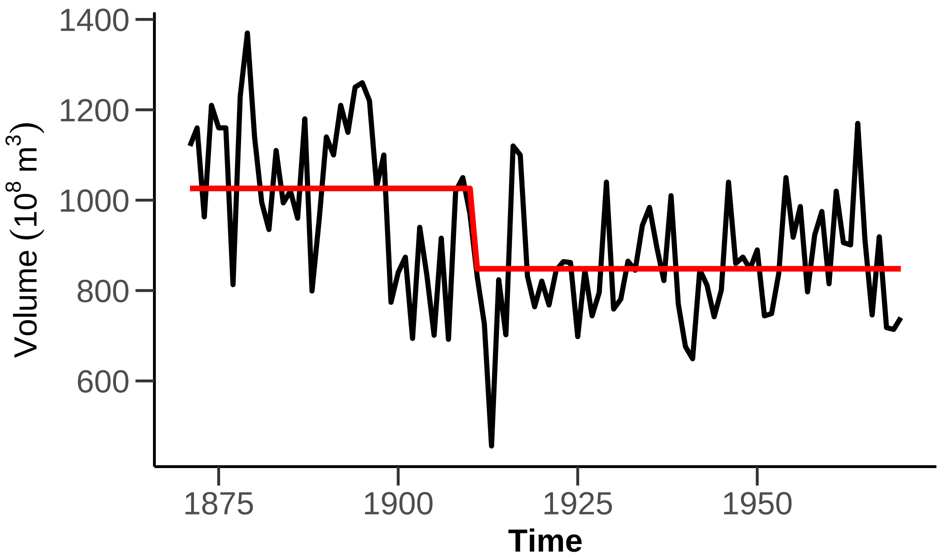
The change point is now estimated as 1910. Without making any modifications to the way the the test statistic is calculated, the first time point we can find the change is at year 1910, which is later than the estimate from part (c). This highlights the need to pick a good bandwidth!

(e) The estimated mean signal $\hat{f}_t$ can be calculated using sample means of the segments defined by the estimated change points. For the CUSUM method, add the estimated $\hat{f}_t$ to your plot from part (a).

Solution

```
calc.mean.func <- function(x, change.loc){

  no.cpts <- length(change.loc)
  n <- length(x)
  change.loc <- c(change.loc,n)
  fitted.mean <- rep(0,n)

  if(no.cpts==0){
    fitted.mean[1:n] <- mean(x)
  }
  else if(no.cpts==1){
    fitted.mean[1:change.loc[1]] <- mean(x[1:change.loc[1]])
    fitted.mean[(change.loc[1]+1):n] <- mean(x[(change.loc[1]+1):n])
  }
  else if(no.cpts>1){
    fitted.mean[1:change.loc[1]] <- mean(x[1:change.loc[1]])
    for (segs in 1:no.cpts){
      fitted.mean[(change.loc[segs]+1):change.loc[segs+1]] <- mean(x[(change.loc[segs]+1):cha
    }
  }

  return(fitted.mean)
}

nile.mean <- calc.mean.func(nile.data$volume, change.loc = nile.cpt)

mean.df <- data.frame(year = nile.data$year, f_hat = nile.mean)

p1 <- ggplot(data = nile.data, aes(x=year,y=volume))+
  geom_line(data=nile.data,color="black",linewidth=0.9)+
  geom_line(data=mean.df,color="red",linewidth=1, aes(x=year,y=f_hat))+
  theme_classic()+
  labs(x="Time",y=expression(Volume~(10^8~m^3))) +
```

```r
  theme(axis.text=element_text(size=12),axis.ticks.length=unit(.25, "cm"),
        axis.title=element_text(size=12),title = element_text(size=18,face="bold"))

p1
```



Question 3: Using the mosum and changepoint packages

(a) Load the `changepoint` and `mosum` R packages into your R workspace. Using the help files, get acquainted with the `cpt.mean()` and `mosum()` functions.

```r
library(changepoint)
library(mosum)
```

(b) Setting the argument `method = "AMOC"` (at most one change), use the `cpt.mean()` function on the Nile data set. Do the results agree with your answer from question 1(a)?

Solution:

```r
nile.cpts.cusum <- cpt.mean(nile.data$volume, method = "AMOC")
nile.data$year[nile.cpts.cusum@cpts]
```

```
[1] 1898 1970
```

Yes: same answer as before (`cpt.mean()` also returns the last time index).

(c) Use the `cpt.mean()` function to apply the PELT algorithm on the Nile data set. Is the answer the same as part (b)? **Note**: you will need to standardise the data first (subtract the mean and divide by the standard deviation).

Solution:

```
#need to standardise the data first!
nile.cpts.pelt <- cpt.mean((nile.data$volume - mean(nile.data$volume))/sd(nile.data$volume))
nile.data$year[nile.cpts.pelt@cpts]
```

```
[1] 1898 1970
```

Yes: same answer as before.

(d) By investigating the output returned by the `cpt.mean()` function, what was the value of the penalty function used?

Solution:

```
nile.cpts.pelt@pen.value
```

```
[1] 13.81551
```

(d) Using the bandwidth $G = 25$, use the `mosum()` function on the Nile data set. Do the results agree with your answer from question 1(b)?

Solution:

```
nile.cpts.mosum <- mosum(as.numeric(nile.data$volume), G = 25)
nile.data$year[nile.cpts.mosum$cpts]
```

```
[1] 1898
```

Yes: same answer as before.

(e) By investigating the output returned by the `mosum()` function, what are the estimated jump sizes and p-value(s) of the detected change points?

Solution:

```r
nile.cpts.mosum$cpts.info
```

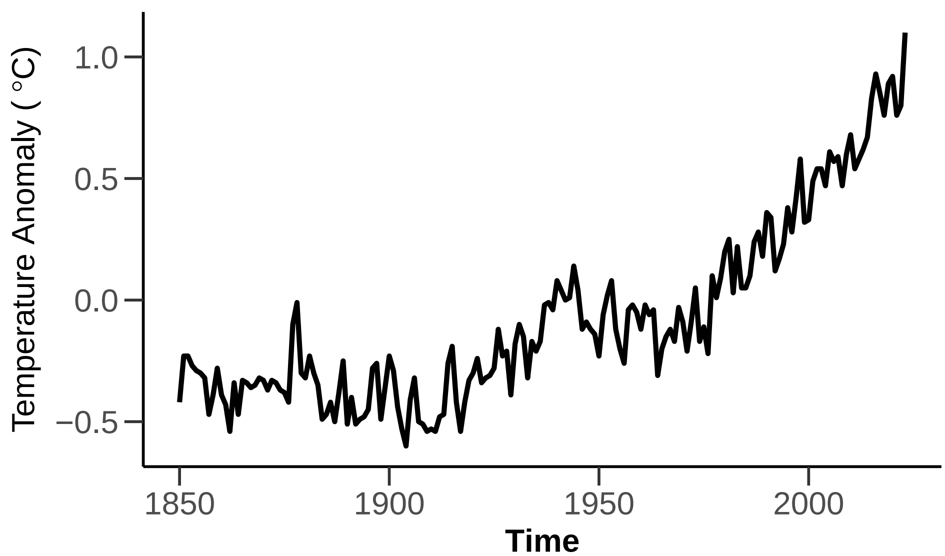
```
  cpts G.left G.right    p.value    jump
1   28     25      25 0.00054662 1.86403
```

The p-value associated to the change is 0.000546620028168743 (this is calculated using the asymptotic null distribution of the test statistic). The jump size is 1.86402973639102.

Python users: you can use the `ruptures` function `Pelt()` and the `mosum` function `mosum()` for this question. See https://centre-borelli.github.io/ruptures-docs/ and https://pypi.org/project/mosum/ for help.

Question 4: Global yearly mean sea temperature anomalies

Global mean surface temperature series help enable the monitoring of global warming. The warming can be quantified by, for example, a change from a base period used as reference. The Hadley Centre/Climatic Research Unit (HadCRUT) surface temperature data set provides the annual global temperature anomalies from 1850-2023, where anomalies are calculated relative to the 1961–1990 period. The data are shown below, which can be seen to exhibit a gradual upward trend.



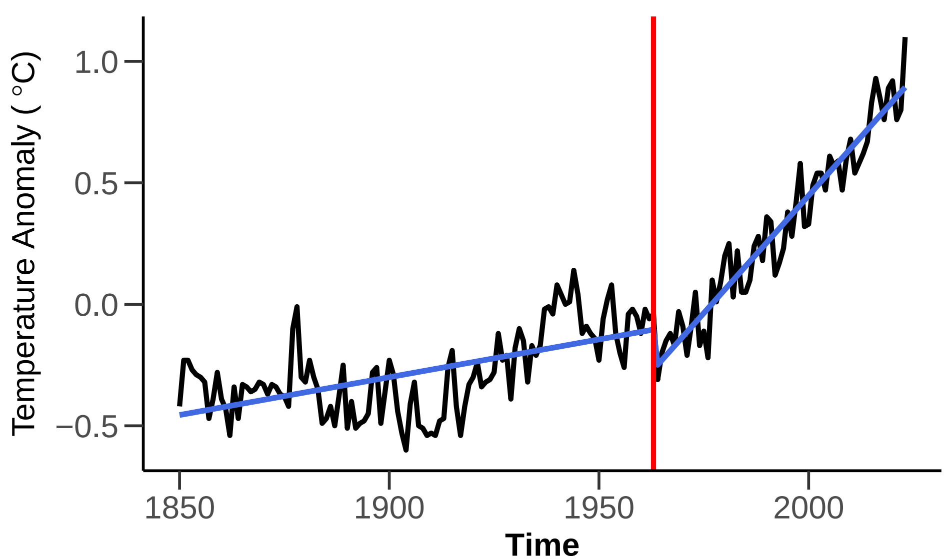
(a) Load the data into `R` from the `temp_anomalies.txt` file, and plot it. Using e.g. the `lm()` function, fit two linear trends to the data: one from 1850 to 1963, and one from 1964 to 2023, and add these lines to the plot, as well as a vertical line through year 1963.

Solution:

```r
which(temp.data$year == 1963)
```

```
[1] 114
```

```r
data1 <- temp.data[1:114,]
data2 <- temp.data[115:174, ]
fitted.data <- c(lm(anomaly ~ year, data = data1)$fitted.values,
                 lm(anomaly ~ year, data = data2)$fitted.values)

est.data <- data.frame(year = temp.data$year, anomaly = fitted.data)

trend.plot <- ggplot(data = temp.data, aes(x=year,y=anomaly))+
  geom_line(data=temp.data,color="black",linewidth=0.9)+
  geom_line(data=est.data,color="royalblue",linewidth=1)+
  theme_classic()+
  labs(x="Time",y=expression('Temperature Anomaly ('*~degree*C*')'))+
  theme(axis.text=element_text(size=12),axis.ticks.length=unit(.25, "cm"),
        axis.title=element_text(size=12),title =
          element_text(size=18,face="bold")) +
  geom_vline(xintercept = temp.data$year[114], color = "red", linewidth =0.9)

trend.plot
```



(b) Using your own functions, and functions from the `changepoint` and `mosum` packages, fit 4 models to the data set;

1. a constant mean model with no change points,
2. a mean change point model,
3. a linear trend model with no change points.
4. a linear trend change point model.

Looking at the data, are there any other models that might be appropriate?

Hint: use `the cpt.reg()` function in `changepoint` for linear trend changes. If you are struggling, the `EnvCpt` R package has everything you need.

Python users: you can use the `Pelt` function and change the `model` parameter.

Solution:

```
#Fitting the 4 different models:
m1 <- mean(temp.data$anomaly)
m2 <- cpt.mean((temp.data$anomaly - mean(temp.data$anomaly))/
                 sd(temp.data$anomaly))
m3 <- lm(temp.data$anomaly ~ I(1:174))

T.d <- cbind(temp.data$anomaly, 1, 1:174)
m4 <- cpt.reg(T.d)
```

It might be the case that the data has autocorrelation (nearby observations are correlated). If this is the case, we could consider models that account for this, such as an autoregressive (AR) model with change points in the mean/trend.

We can also do all model fits at once in `EnvCpt` package (note however the mean change now includes a possible change in variance too):

```
library(EnvCpt)

temp.cpt.models <- envcpt(temp.data$anomaly, models = c("mean", "meancpt",
                                   "trend", "trendcpt"), verbose = FALSE)
```

(c) Pick the "best" change point model from your candidates computed in part (b), and justify your choice. Add the estimated mean/trend function onto a plot of the data.

Hint: the Akiake information criterion (AIC) is given by $\text{AIC} = 2k - 2\log(\hat{L})$, where k is the number of parameters of the model, and $\hat{L}$ is the maximised value of the likelihood function for the model. The smaller the AIC, the "better" the model. For normally distributed data, you can use `dnorm()` in R to compute the likelihood.

Solution:

```
m1.lik <- sum(dnorm(temp.data$anomaly, mean = m1, sd = sd(temp.data$anomaly),
                    log = TRUE))
m1.aic <- 2*2 - 2*m1.lik

mean.m2 <- calc.mean.func(temp.data$anomaly, m2@cpts[1:2])
m2.lik <- sum(dnorm(temp.data$anomaly, mean = mean.m2,
                    sd = sd(temp.data$anomaly - mean.m2), log = TRUE))
m2.aic <- 2*3 - 2*m2.lik

mean.m3 <- m3$fitted.values
m3.lik <- sum(dnorm(temp.data$anomaly, mean = mean.m3,
                    sd = sd(temp.data$anomaly - mean.m3), log = TRUE))
m3.aic <- 2*3 - 2*m3.lik

mean.m4 <- numeric(0)
cpts <- c(0, m4@cpts)
betas <- param.est(m4)$beta
for(i in 1:nseg(m4)){
  mean.m4 <- c(mean.m4, betas[i,]%*%t(data.set(m4)[(cpts[i]+1):cpts[i+1],-1]))
}

m4.lik <- sum(dnorm(temp.data$anomaly, mean = mean.m4,
                    sd = sd(temp.data$anomaly - mean.m4), log = TRUE))
m4.aic <- 2 * 9 - 2 * m4.lik
aic.values <- c(m1.aic, m2.aic, m3.aic, m4.aic)
aic.values
```

```
[1]  159.38100 -168.53453  -55.83466 -309.96109
```

You can also just directly use `AIC()` and `BIC()` functions in R to compare models fitted using the `EnvCpt` package (note the slight difference in values compared to above, as simultaneous variance changes are also allowed in `EnvCpt`):

```
AIC(temp.cpt.models)
```

```
      mean    meancpt    meanar1    meanar2 meanar1cpt meanar2cpt
 159.37812 -254.80551         NA         NA         NA         NA
     trend   trendcpt   trendar1   trendar2 trendar1cpt trendar2cpt
 -55.83755 -300.72621         NA         NA         NA         NA
```

```r
which.min(AIC(temp.cpt.models))
```

```
trendcpt
       8
```

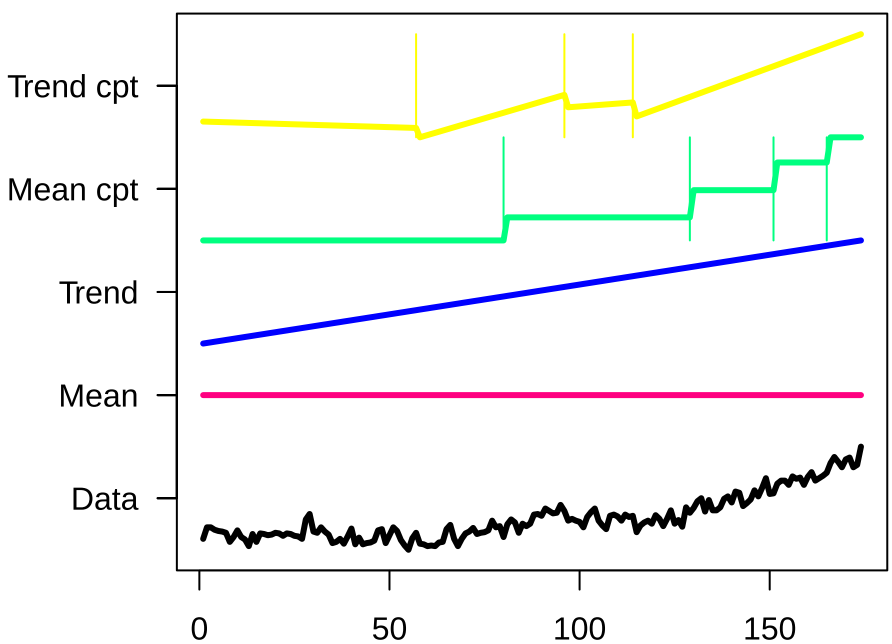
```r
BIC(temp.cpt.models)
```

```
       mean      meancpt      meanar1      meanar2  meanar1cpt  meanar2cpt
  165.69623   -210.57873           NA           NA          NA          NA
      trend     trendcpt     trendar1     trendar2 trendar1cpt trendar2cpt
  -46.36038   -253.34038           NA           NA          NA          NA
```

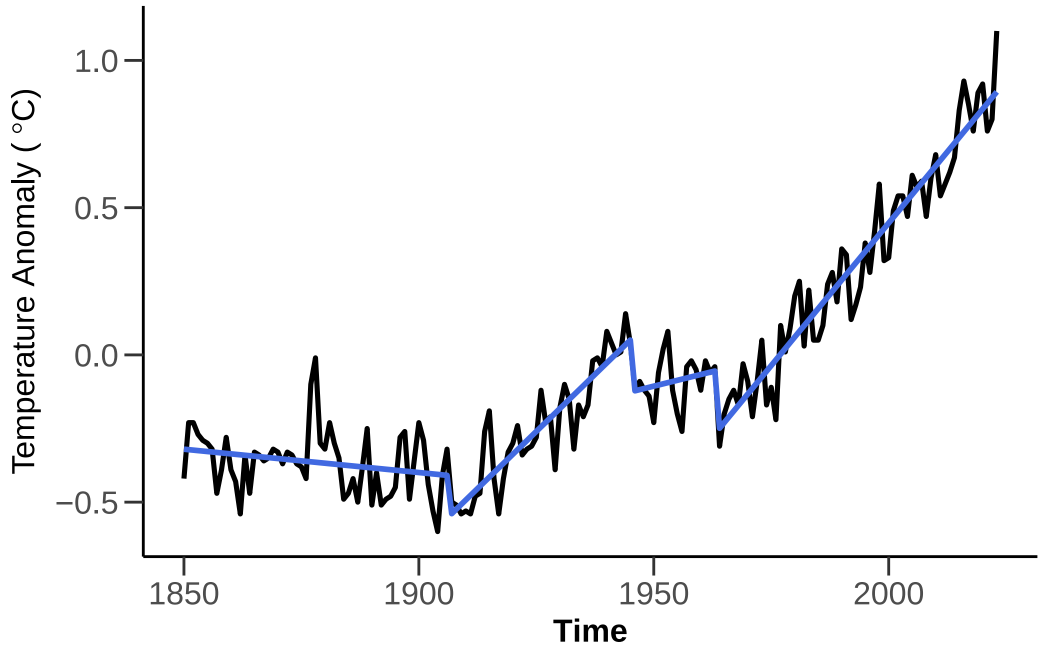
```r
which.min(BIC(temp.cpt.models))
```

```
trendcpt
       8
```

```r
plot(temp.cpt.models, type = 'fit')
```



```
est.data <- data.frame(year = temp.data$year, anomaly = mean.m4)
trend.plot <- ggplot(data = temp.data, aes(x=year,y=anomaly))+
  geom_line(data = temp.data, color = "black", linewidth = 0.9) +
  geom_line(data = est.data, color = "royalblue", linewidth = 1) +
  theme_classic() +
  labs(x = "Time",y = expression('Temperature Anomaly ('*~degree*C*')')) +
  theme(axis.text = element_text(size=12), axis.ticks.length=unit(.25, "cm"),
        axis.title = element_text(size=12), title =
          element_text(size=18,face="bold"))
trend.plot
```



Question 5 (Bonus): Multivariate data

(a) Suppose we want to find mean change points in a time series $\{X_t\}_{t=1}^n$, where $X_t = (X_{1t}, \ldots, X_{pt})^\top$ is a p-dimensional vector. One approach is to calculate a test statistic for each variable, and then combine the results across variables to give a single test statistic. For example, for the i-th variable of $\{X_t\}_{t=1}^n$, denoted $\{X_{it}\}_{t=1}^n$, the MOSUM test statistic is given by

$$T_G(k,i) = \frac{1}{\sqrt{2G}} \left(\sum_{t=k+1}^{k+G} X_{it} - \sum_{t=k-G+1}^{k} X_{it} \right).$$

Then, the MOSUM statistic $ T_{G}(k)$ for a change point in $\{X_t\}_{t=1}^n$ can be computed by aggregating the $\{T_G(k,i)\}_{i=1}^p$ using some aggregating function f, so that

$$T_G(k) = f\left(T_G(k,1),\dots,T_G(k,p)\right).$$

What possibilities could we use for the aggregating function f?

Solution:

We could use any appropriate norm, such as the L_1 or L_2 norm:

- $f(x_1,\dots,x_p) = (\sum_{i=1}^p x_i^2)^{1/2}$ (the L_2 norm)
- $f(x_1,\dots,x_p) = \max_{i=1,\dots,p}|x_i|$ (the L_∞ norm)

(b) Using your aggregating function f, write a function to compute the MOSUM or CUSUM statistic for a change in the mean vector of a multivariate time series.

Solution:

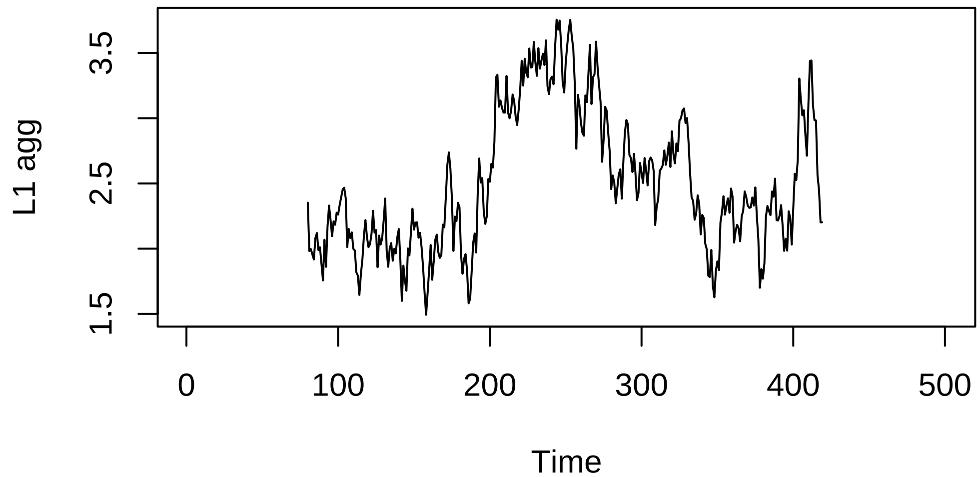
```
multi.MOSUM <- function(x, G, agg = c("L1", "L2")[1]){

  matrix.MOSUM <- abs(apply(x, 2, MOSUM.calc2, G = G))

  if(agg == "L2"){
    x.MOSUM <- sqrt(rowSums(matrix.MOSUM^2))
  }else{
    x.MOSUM <- apply(matrix.MOSUM, 1, max)
  }

  return(x.MOSUM)

}
```

(c) Simulate a data set of length $n = 500$ and dimension $p = 40$ as follows. Generate $\{X_t\}_{t=1}^{250}$ from the standard p-dimensional normal distribution, and generate $\{X_t\}_{t=251}^{500}$ from the p-dimensional normal distribution with identity covariance matrix, and mean vector given by $\mu = 1.4\times(1/\sqrt{p},\dots,1/\sqrt{p})^\top$. Use the method from part (b), with $G = 80$, to compute the test statistic for a change in mean. Is the change point easy to see?

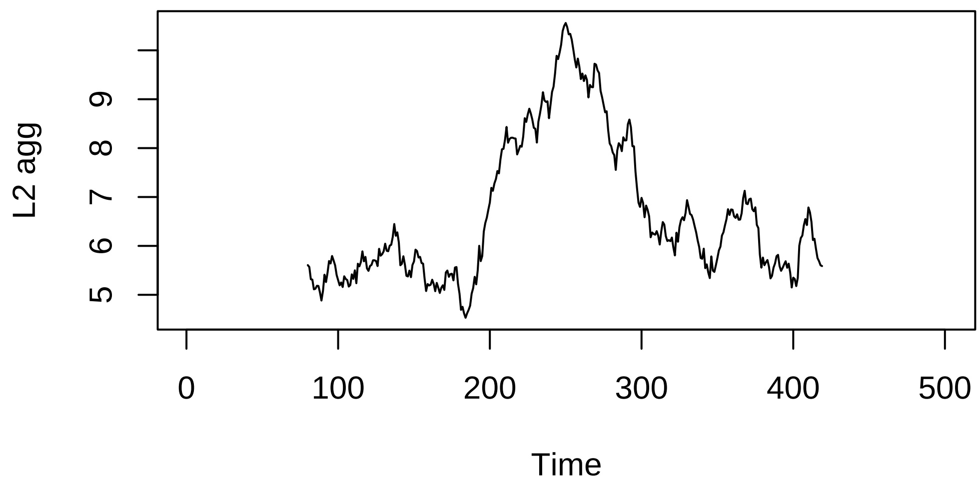
Solution:

```
set.seed(1)
x <- matrix(rnorm(40*500), nrow = 500, ncol = 40)

x[251:500,] <- x[251:500,] + 1.4/sqrt(40)
```

```
plot.ts(multi.MOSUM(x, G = 80, agg = "L1"), ylab = "L1 agg")
```




```
plot.ts(multi.MOSUM(x, G = 80, agg = "L2"), ylab = "L2 agg")
```

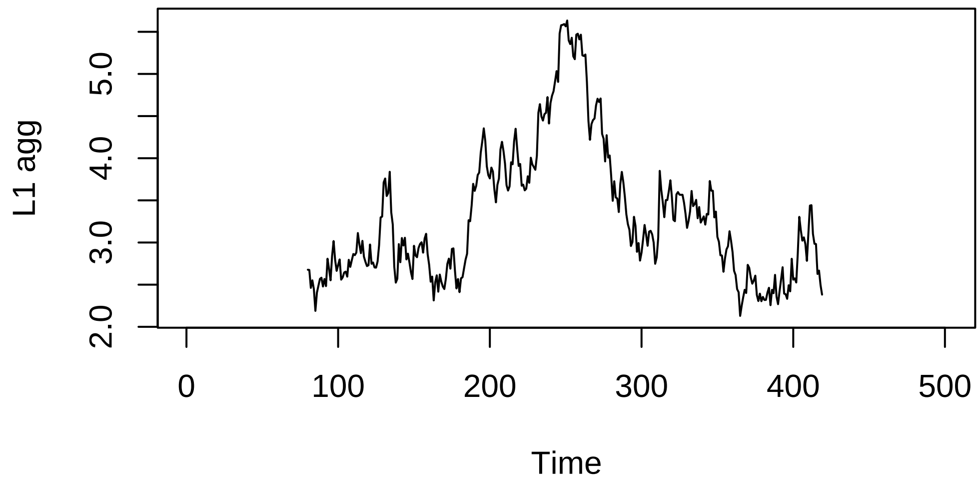


(d) Re-do part (c), but generate $\{X_t\}_{t=251}^{500}$ from the $p = 200$-dimensional normal distribution with identity covariance matrix, and mean vector given μ with first 2 entries equal to 0.8, and last $p - 2$ entries equal to 0. Is the change point easy to see?

Solution:

```
set.seed(1)
x <- matrix(rnorm(200*500), nrow = 500, ncol = 200)
```

```
x[251:500,1:2] <- x[251:500,1:2] + 0.8

plot.ts(multi.MOSUM(x, G = 80, agg = "L1"), ylab = "L1 agg")
```




```
plot.ts(multi.MOSUM(x, G = 80, agg = "L2"), ylab = "L2 agg")
```