Introduction to change point detection: lab worksheet answers

Euan McGonigle

2024-09-12

Question 1: Creating a test statistic to estimate change points

(a) Without using pre-existing libraries, write a function to calculate the CUSUM statistic vector $\{\mathcal{T}_{0,k,n}\}_{k=1}^{n-1}$ for a change point in the mean.

```
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))

}
```

(b) Without using pre-existing libraries, write a function to calculate the MOSUM statistic vector $\{\mathcal{T}_G(k)\}_{k=G}^{n-G}$ for a change point in the mean, for a given bandwidth G.

```
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))
```

```
}
```

(c) 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 changes. 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.

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 $\widehat{\sigma}^2(k) = (\widehat{\sigma}^2_{(k-G+1):(k)} + \widehat{\sigma}^2_{(k+1):(k+G)})/2$, where $\widehat{\sigma}^2_{s:e}$ denotes the sample variance calculated on the data from start s to end e.

(d) Implement your chosen method of estimating σ, and add the functionality to your CUSUM and MOSUM functions so that the test statistic calculations in parts (a) and (b) are scaled by your estimate $\widehat{\sigma}$.

```
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)

}
```

(e) (Harder) To find multiple change points with the MOSUM approach, we can calculate all $\widehat{\theta}$ that satisfy:

$$\mathcal{T}_\ell(G, \widehat{\theta}) > D \quad \text{and} \quad \widehat{\theta} = \text{argmax}_{k:\,|k-\widehat{\theta}|\le \eta G} \mathcal{T}_G(k).$$

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

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

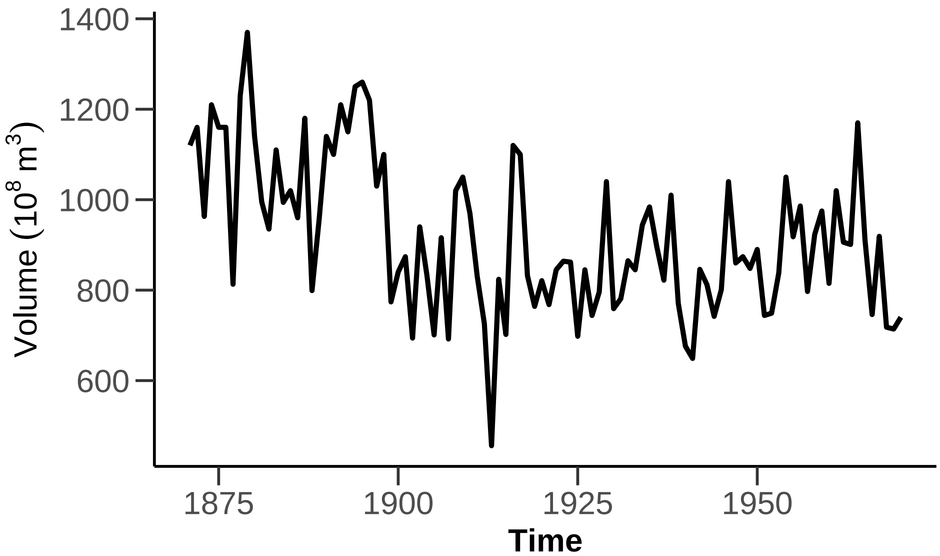
```
MOSUM.calc3 <- function(x, G, eta, D){

  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)
}
```

(f) What is the computational complexity of the methods you wrote in parts (a) and (b)? How fast could it be?

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.

```r
library(ggplot2)

nile.data <- read.table("/Users/euanmcgonigle/Documents/Southampton-Work/Software/Rough Code/nile_volume
                        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
```

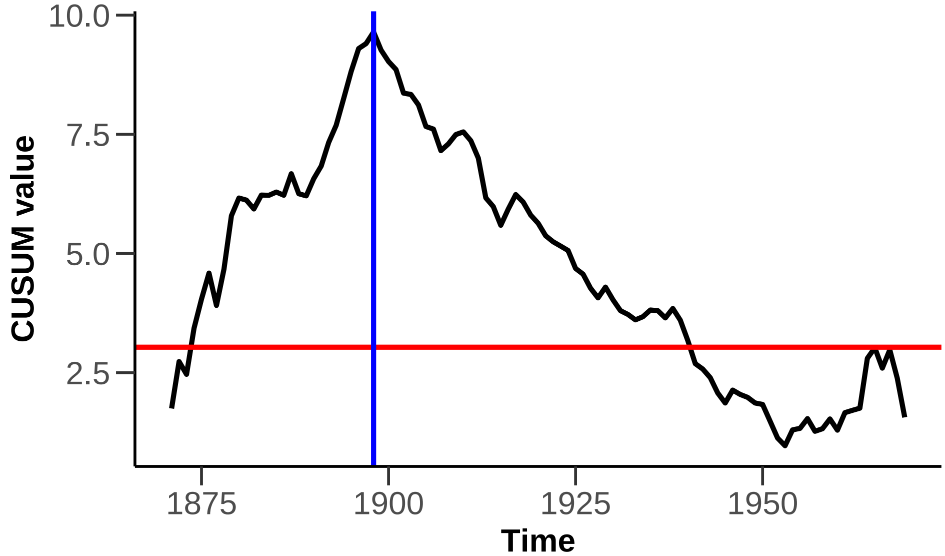


(a) Load the data into R from the `nile_volume.txt` file, and plot it. By eye, where does it look like there could be a change in mean?

(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?

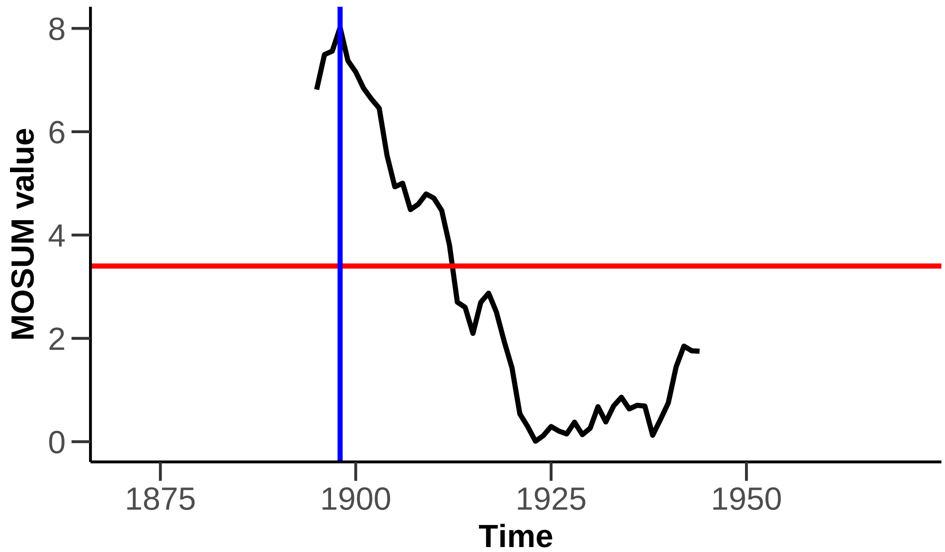
```r
nile.CUSUM <- CUSUM.calc2(nile.data$volume)

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
```



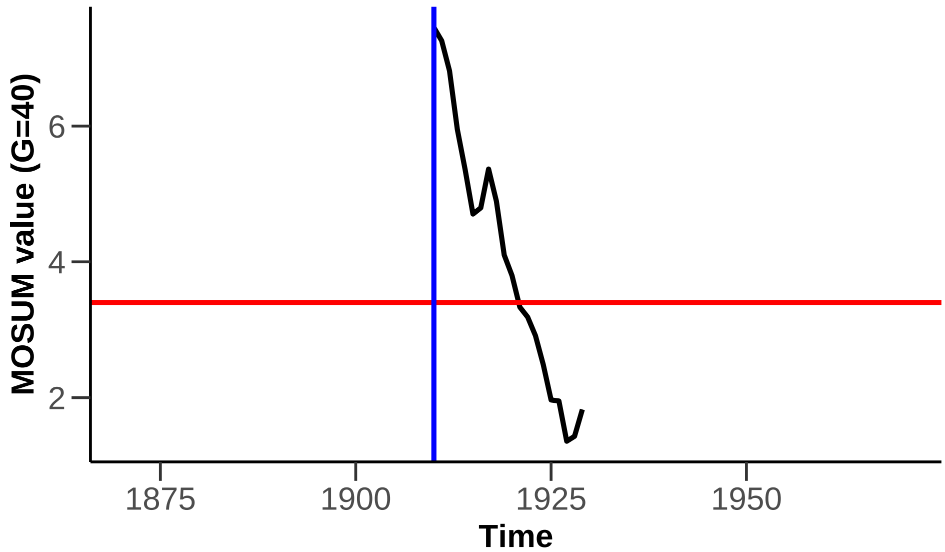
(c) Calculate the MOSUM statistic for the Nile volume data using a bandwidth $G = 25$. Using the threshold $D = 3.4$, compute the change point estimators as in q1 (e). Does this agree with your answer from part (b)?

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

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") +
  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
```



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

```
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
```



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

```
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):change.loc[seg
    }
  }

  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))) +
  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 aquainted with the `cpt.mean` and `mosum` functions.

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

(b) Using the default settings, use the `cpt.mean` function on the Nile data set. Do the results agree with your answer from question 1(a)?

```
nile.cpts.cusum <- cpt.mean(nile.data$volume)
```

(c) By altering the input parameters, apply the PELT algorithm using the `cpt.mean` function on the Nile data set. Is the answer the same as part (b)?

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

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

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

(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)?

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

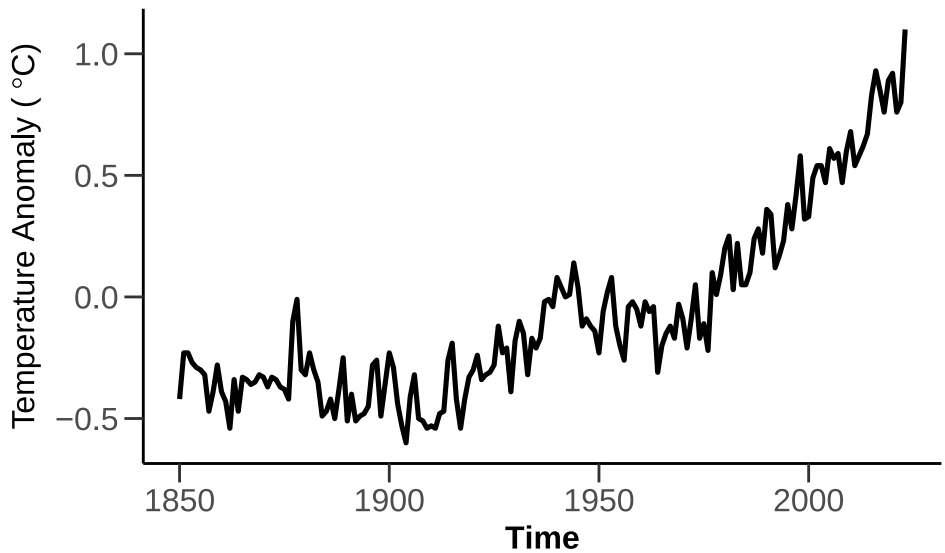
(e) By investigating the output returned by the `mosum` function, what are the p-value(s) of the detected change points?

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

```
*Python* users: you can use the \texttt{ruptures} function \texttt{Pelt} and the \texttt{mosum} functio
```

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.

```
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: 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.

```
#I've cheated here using the EnvCpt package. You could manually search
#for change points using the least-squares approach.

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.

```
#  I've cheated again - you can manually calculate the AIC for the 4 models to compare.
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
```

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

```
## trendcpt
##        8
```

```
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
```

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

```
## trendcpt
##        8
```

```
trendcpt <- numeric(0)
cpts <- c(0, temp.cpt.models$trendcpt@cpts)
betas <- param.est(temp.cpt.models$trendcpt)$beta
for(i in 1:nseg(temp.cpt.models$trendcpt)){
  trendcpt <- c(trendcpt,betas[i,]%*%t(data.set(temp.cpt.models$trendcpt)[(cpts[i]+1):cpts[i+1],-1]))
}
trendcpt <- (trendcpt-min(trendcpt))/diff(range(trendcpt))
```

Question 5: Multivariate data

(a) Suppose we want to find mean change points in a time series $\{X_t\}_{t=1}^n$, where $X_t = (X_{1t}, \dots, X_{pt})^\intercal$ 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

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

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

$$\mathcal{T}_G(k) = f\left(\mathcal{T}_G(k,1), \dots, \mathcal{T}_G(k,p)\right).$$

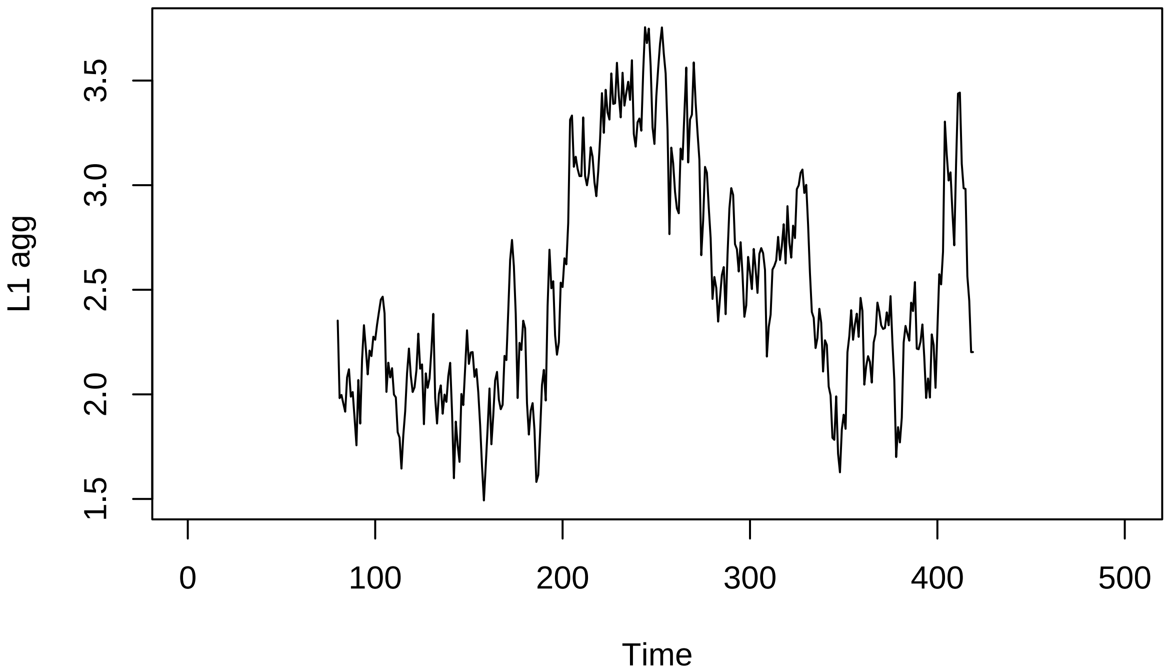
What possibilities could we use for the aggregating function f?

(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.

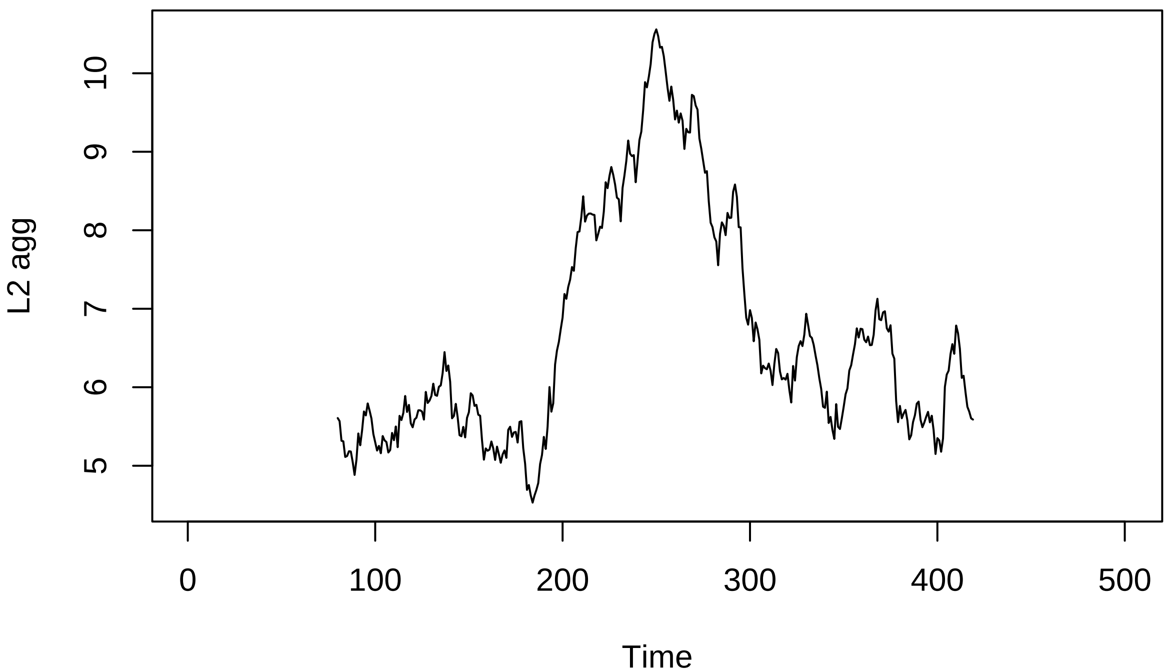
```r
multi.MOSUM <- function(x, G, agg = c("L1", "L2")[1]){

  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 $\mu = 1.4 \times (1/\sqrt{p}, \dots, 1/\sqrt{p})^\intercal$. 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?

```r
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")
```



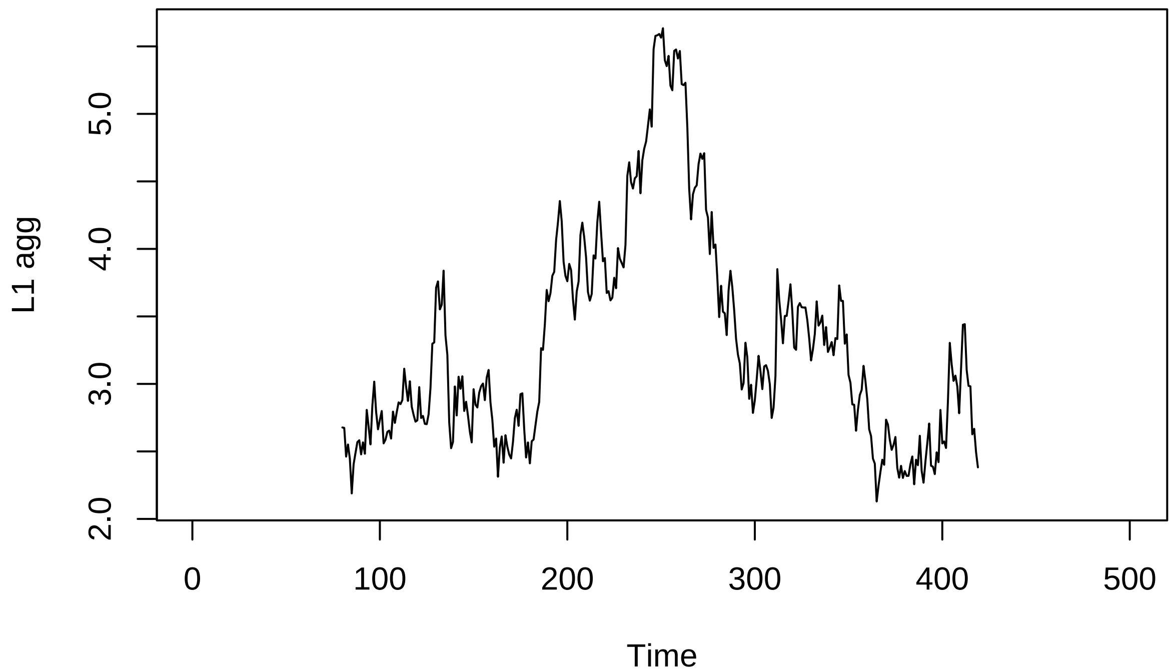

```r
plot.ts(multi.MOSUM(x, G = 80, agg = "L2"), ylab = "L2 agg")
```



(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?

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

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

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




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