```
## Step 1: Estimating minimum spanning trees (MSTs).

# Load igraph package.
library(igraph)

# Load a data file with 73 variables where the first column contains a date vector while the rest of the columns
contain FX spot prices.
load("Data.RData")

# Create a date variable.
Date <- format(Data[,1],format="%d/%m/%Y")

# Create a data variable excluding the date column.
Data <- Data[,-c(1)]

# Convert spot prices into log prices as a matrix.
Log_data <- as.matrix(log(Data))

# Calculate price changes.
Rets <- diff(Log_data)
Date <- Date[-c(1)]

# Count the number of rows.
Row_number <- dim(Rets)[1]

# Set the length of the rolling window.
first_run <- 1+251

# Start a for-loop for rolling estimation.
for(i in first_run:Row_number)

    # Assign the values of rolling returns and the corresponding date to variables.
    {Rolling_rets <- Rets[(i-251):i,]
    Rolling_date <- (Date[i])

    # Estimate rolling correlation coefficients from log returns.
    Corr_mat <- cor(Rolling_rets)

    # Label rows and columns.
    colnames(Corr_mat) <-
    c("EURGBP","EURAUD","EURNZD","EURUSD","EURCAD","EURCHF","EURJPY","EURCNH",
    "GBPEUR","GBPAUD","GBPNZD","GBPUSD","GBPCAD","GBPCHF","GBPJPY","GBPCNH",
    "AUDEUR","AUDGBP","AUDNZD","AUDUSD","AUDCAD","AUDCHF","AUDJPY","AUDCNH",
    "NZDEUR","NZDGBP","NZDAUD","NZDUSD","NZDCAD","NZDCHF","NZDJPY","NZDCNH",
    "USDEUR","USDGBP","USDAUD","USDNZD","USDCAD","USDCHF","USDJPY","USDCNH",
    "CADEUR","CADGBP","CADAUD","CADNZD","CADUSD","CADCHF","CADJPY","CADCNH",
    "CHFEUR","CHFGBP","CHFAUD","CHFNZD","CHFUSD","CHFCAD","CHFJPY","CHFCNH",
    "JPYEUR","JPYGBP","JPYAUD","JPYNZD","JPYUSD","JPYCAD","JPYCHF","JPYCNH",
    "CNHEUR","CNHGBP","CNHAUD","CNHNZD","CNHUSD","CNHCAD","CNHCHF","CNHJPY")

    rownames(Corr_mat) <-
    c("EURGBP","EURAUD","EURNZD","EURUSD","EURCAD","EURCHF","EURJPY","EURCNH",
    "GBPEUR","GBPAUD","GBPNZD","GBPUSD","GBPCAD","GBPCHF","GBPJPY","GBPCNH",
    "AUDEUR","AUDGBP","AUDNZD","AUDUSD","AUDCAD","AUDCHF","AUDJPY","AUDCNH",
    "NZDEUR","NZDGBP","NZDAUD","NZDUSD","NZDCAD","NZDCHF","NZDJPY","NZDCNH",
    "USDEUR","USDGBP","USDAUD","USDNZD","USDCAD","USDCHF","USDJPY","USDCNH",
    "CADEUR","CADGBP","CADAUD","CADNZD","CADUSD","CADCHF","CADJPY","CADCNH",
```

```
"CHFEUR","CHFGBP","CHFAUD","CHFNZD","CHFUSD","CHFCAD","CHFJPY","CHFCNH",
"JPYEUR","JPYGBP","JPYAUD","JPYNZD","JPYUSD","JPYCAD","JPYCHF","JPYCNH",
"CNHEUR","CNHGBP","CNHAUD","CNHNZD","CNHUSD","CNHCAD","CNHCHF","CNHJPY")

# Convert correlation matrix into Euclidean distance matrix.
Dis_mat <- as.matrix(sqrt(2-2*Corr_mat))

# Convert the distance matrix as an adjacency matrix.
G <- graph.adjacency(Dis_mat,mode=c("undirected"),weighted=TRUE)

# Estimate minimum spanning tree (MST) using Prim's algorithm.
MST <- minimum.spanning.tree(G,algorithm="prim")

## Step 2: Visualising results.

# Set layout algorithm - the layout of the previous network is used to calculate the layout of the new
network such that changes in the layout of the MST is minimised.
if (i == first_run){
    lay <- layout.kamada.kawai(MST)
    layout_store <<- lay
    V(MST)$x <- lay[,1]
    V(MST)$y <- lay[,2]
} else {
    lay<-layout.kamada.kawai(MST,coords = layout_store)
    layout_store <<- lay
    V(MST)$x <- lay[,1]
    V(MST)$y <- lay[,2]
}

# Set node colours based on base currency.
# EUR nodes.
V(MST)$color[1:8] <- rgb(0,128,0,255,max=255)
# GBP nodes.
V(MST)$color[9:16] <- rgb(15,0,128,255,max=255)
# AUD, NZD nodes.
V(MST)$color[17:32] <- rgb(192,192,192,255,max=255)
#USD nodes.
V(MST)$color[33:40] <- rgb(255,1,255,255,max=255)
# CAD nodes.
V(MST)$color[41:48] <- rgb(192,192,192,255,max=255)
#CHF, JPY nodes.
V(MST)$color[49:64] <- rgb(255,255,0,255,max=255)
#CNH nodes.
V(MST)$color[65:72] <- rgb(255,1,255,255,max=255)

# Set png file names.
Plot_file_name=paste(paste('Dynamic_MST_plot_',i-251,sep=''),'.png',sep='')

# Load the png device.
png(Plot_file_name,res=1300,height=3600,width=5600,pointsize=10)

# Set the margins of the png file.
par(mar=c(1,1,3.5,1))

# Produce the MST plot.
plot(MST,layout=lay,vertex.size=4,vertex.shape="circle", asp=FALSE,
vertex.label.cex=0.3, vertex.label.dist=0.4, vertex.label.degree=-pi/2,
```

```
        vertex.label.color="black",vertex.color=V(MST)$color,
        edge.width=0.5,edge.color=rgb(0,0,0,255,max=255))
        title(Rolling_date,cex.main=0.8)

        # Close the png device.
        dev.off()

# Close the for-loop.
}

## Step 3: Recording an animation.

# A series of png plots can then be recorded as an animation in Windows Movie Maker.
```