Project #2: foreign exchange market interactions

Live sessions: weeks 3 and 4

Table of Contents

## Purpose, Process, Product

This group assignment provides practice in foreign exchange markets as well as R models of those markets. Specifically we will practice reading in data, exploring time series, estimating auto and cross correlations, and investigating volatility clustering in financial time series. We will summarize our experiences in debrief. We will pay special attention to the financial economics of exchange rates.

## Assignment

Submit into **Coursework > Assignments and Grading > Assignment 2: Team Project 2 > Submission** an RMD file with filename **lastname\_firstname\_Assignment2.Rmd** as well as the HTML file generated with a Knit. You will need to zip the 2 files before submission..

1. Use headers (##), r-chunks for code, and text to build a report that addresses the two parts of this project.
2. List in the text the ‘R’ skills needed to complete this project.
3. Explain each of the functions (e.g., ggplot()) used to compute and visualize results.
4. Discuss how well did the results begin to answer the business questions posed at the beginning of each part of the project.

## General format of Document to be submitted  
  
## Part 1: Exchange Rates data preparation and exploration  
  
### Problem  
(Explanatory text)  
(r chunks)  
### Question 1 - title  
(Explanatory text)  
(r chunks)  
### Question 2 - title  
(Explanatory text)  
(r chunks)  
  
## Part 2: Exchange Rates analysis  
  
### Problem  
(Explanatory text)  
(r chunks)  
### Further questions as needed  
(Explanatory text)  
(r chunks)  
  
## Conclusion  
  
### Skills and Tools  
(text here)  
### Data Insights  
(text here)  
### Business Remarks  
(text here)

## Part 1

In this set we will build and explore a data set using filters and if and diff statements. We will then answer some questions using plots and a pivot table report. We will then review a function to house our approach in case we would like to run some of the same analysis on other data sets.

### Problem

Marketing and accounts receivables managers at our company continue to note we have a significant exposure to exchange rates. Our functional currency (what we report in financial statements) is in U.S. dollars (USD).

* Our customer base is located in the United Kingdom, across the European Union, and in Japan. The exposure hits the gross revenue line of our financials.
* Cash flow is further affected by the ebb and flow of accounts receivable components of working capital in producing and selling several products. When exchange rates are volatile, so is earnings, and more importantly, our cash flow.
* Our company has also missed earnings forecasts for five straight quarters.

To get a handle on exchange rate exposures we download this data set and review some basic aspects of the exchange rates.

# Read in data  
library(zoo)  
library(xts)  
library(ggplot2)  
# Read and review a csv file from  
# FRED  
exrates <- na.omit(read.csv("../data/exrates.csv",   
 header = TRUE))  
# Check the data  
head(exrates)

## DATE USD.EUR USD.GBP USD.CNY USD.JPY  
## 1 1/29/2012 0.763678 0.638932 6.29509 77.1840  
## 2 2/5/2012 0.760684 0.633509 6.29429 76.3930  
## 3 2/12/2012 0.757491 0.632759 6.29232 77.2049  
## 4 2/19/2012 0.760889 0.634166 6.29644 78.7109  
## 5 2/26/2012 0.750301 0.632641 6.29710 80.3373  
## 6 3/4/2012 0.750474 0.629771 6.29873 81.1607

tail(exrates)

## DATE USD.EUR USD.GBP USD.CNY USD.JPY  
## 255 12/11/2016 0.938872 0.791554 6.93141 114.397  
## 256 12/18/2016 0.950478 0.796572 6.93042 116.796  
## 257 12/25/2016 0.958288 0.810481 6.94908 117.469  
## 258 1/1/2017 0.954067 0.813594 6.94929 117.100  
## 259 1/8/2017 0.951493 0.812388 6.92820 116.968  
## 260 1/15/2017 0.943352 0.820854 6.91781 115.287

str(exrates)

## 'data.frame': 260 obs. of 5 variables:  
## $ DATE : Factor w/ 260 levels "1/1/2017","1/10/2016",..: 15 103 89 94 100 126 109 114 120 130 ...  
## $ USD.EUR: num 0.764 0.761 0.757 0.761 0.75 ...  
## $ USD.GBP: num 0.639 0.634 0.633 0.634 0.633 ...  
## $ USD.CNY: num 6.3 6.29 6.29 6.3 6.3 ...  
## $ USD.JPY: num 77.2 76.4 77.2 78.7 80.3 ...

# Begin to explore the data  
summary(exrates)

## DATE USD.EUR USD.GBP USD.CNY   
## 1/1/2017 : 1 Min. :0.7199 Min. :0.5835 Min. :6.092   
## 1/10/2016: 1 1st Qu.:0.7544 1st Qu.:0.6224 1st Qu.:6.149   
## 1/11/2015: 1 Median :0.7926 Median :0.6418 Median :6.279   
## 1/12/2014: 1 Mean :0.8196 Mean :0.6561 Mean :6.310   
## 1/13/2013: 1 3rd Qu.:0.8932 3rd Qu.:0.6656 3rd Qu.:6.369   
## 1/15/2017: 1 Max. :0.9583 Max. :0.8209 Max. :6.949   
## (Other) :254   
## USD.JPY   
## Min. : 76.39   
## 1st Qu.: 96.90   
## Median :102.44   
## Mean :103.05   
## 3rd Qu.:117.19   
## Max. :124.78   
##

### Questions

1. What is the nature of exchange rates in general and in particular for this data set? We want to reflect the ups and downs of rate movements, known to managers as currency appreciation and depreciation.

* We will calculate percentage changes as log returns of currency pairs. Our interest is in the ups and downs. To look at that we use if and else statements to define a new column called direction. We will build a data frame to house this initial analysis.
* Using this data frame, interpret appreciation and depreciation in terms of the impact on the receipt of cash flow from customer’s accounts that are denominated in other than our USD functional currency.

# Compute log differences percent  
# using as.matrix to force numeric  
# type  
exrates.r <- diff(log(as.matrix(exrates[,   
 -1]))) \* 100  
head(exrates.r)

## USD.EUR USD.GBP USD.CNY USD.JPY  
## 2 -0.39282058 -0.8523826 -0.01270912 -1.0301113  
## 3 -0.42063724 -0.1184583 -0.03130311 1.0571858  
## 4 0.44758304 0.2221127 0.06545522 1.9318720  
## 5 -1.40130272 -0.2407629 0.01048156 2.0452375  
## 6 0.02305476 -0.4546859 0.02588158 1.0197119  
## 7 1.19869144 0.7988383 0.22598055 0.6384155

tail(exrates.r)

## USD.EUR USD.GBP USD.CNY USD.JPY  
## 255 -0.1234763 -0.4555311 0.602265010 0.9803397  
## 256 1.2285861 0.6319419 -0.014283828 2.0753968  
## 257 0.8183343 1.7310378 0.268885929 0.5745646  
## 258 -0.4414460 0.3833571 0.003021937 -0.3146198  
## 259 -0.2701570 -0.1483412 -0.303945688 -0.1127877  
## 260 -0.8592840 1.0367203 -0.150079365 -1.4475722

str(exrates.r)

## num [1:259, 1:4] -0.3928 -0.4206 0.4476 -1.4013 0.0231 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : chr [1:259] "2" "3" "4" "5" ...  
## ..$ : chr [1:4] "USD.EUR" "USD.GBP" "USD.CNY" "USD.JPY"

# Create size and direction  
size <- na.omit(abs(exrates.r)) # size is indicator of volatility  
head(size)

## USD.EUR USD.GBP USD.CNY USD.JPY  
## 2 0.39282058 0.8523826 0.01270912 1.0301113  
## 3 0.42063724 0.1184583 0.03130311 1.0571858  
## 4 0.44758304 0.2221127 0.06545522 1.9318720  
## 5 1.40130272 0.2407629 0.01048156 2.0452375  
## 6 0.02305476 0.4546859 0.02588158 1.0197119  
## 7 1.19869144 0.7988383 0.22598055 0.6384155

# colnames(size) <-  
# paste(colnames(size),'.size', sep =  
# '') # Teetor  
direction <- ifelse(exrates.r > 0, 1,   
 ifelse(exrates.r < 0, -1, 0)) # another indicator of volatility  
# colnames(direction) <-  
# paste(colnames(direction),'.dir',  
# sep = '')  
head(direction)

## USD.EUR USD.GBP USD.CNY USD.JPY  
## 2 -1 -1 -1 -1  
## 3 -1 -1 -1 1  
## 4 1 1 1 1  
## 5 -1 -1 1 1  
## 6 1 -1 1 1  
## 7 1 1 1 1

# Convert into a time series object:  
# 1. Split into date and rates  
dates <- as.Date(exrates$DATE[-1], "%m/%d/%Y")  
values <- cbind(exrates.r, size, direction)  
# for dplyr pivoting we need a data  
# frame  
exrates.df <- data.frame(dates = dates,   
 returns = exrates.r, size = size,   
 direction = direction)  
str(exrates.df) # notice the returns.\* and direction.\* prefixes

## 'data.frame': 259 obs. of 13 variables:  
## $ dates : Date, format: "2012-02-05" "2012-02-12" ...  
## $ returns.USD.EUR : num -0.3928 -0.4206 0.4476 -1.4013 0.0231 ...  
## $ returns.USD.GBP : num -0.852 -0.118 0.222 -0.241 -0.455 ...  
## $ returns.USD.CNY : num -0.0127 -0.0313 0.0655 0.0105 0.0259 ...  
## $ returns.USD.JPY : num -1.03 1.06 1.93 2.05 1.02 ...  
## $ size.USD.EUR : num 0.3928 0.4206 0.4476 1.4013 0.0231 ...  
## $ size.USD.GBP : num 0.852 0.118 0.222 0.241 0.455 ...  
## $ size.USD.CNY : num 0.0127 0.0313 0.0655 0.0105 0.0259 ...  
## $ size.USD.JPY : num 1.03 1.06 1.93 2.05 1.02 ...  
## $ direction.USD.EUR: num -1 -1 1 -1 1 1 1 -1 -1 1 ...  
## $ direction.USD.GBP: num -1 -1 1 -1 -1 1 1 -1 -1 1 ...  
## $ direction.USD.CNY: num -1 -1 1 1 1 1 1 -1 -1 -1 ...  
## $ direction.USD.JPY: num -1 1 1 1 1 1 1 -1 -1 -1 ...

# 2. Make an xts object with row  
# names equal to the dates  
exrates.xts <- na.omit(as.xts(values,   
 dates)) #order.by=as.Date(dates, '%d/%m/%Y')))  
str(exrates.xts)

## An 'xts' object on 2012-02-05/2017-01-15 containing:  
## Data: num [1:259, 1:12] -0.3928 -0.4206 0.4476 -1.4013 0.0231 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:12] "USD.EUR" "USD.GBP" "USD.CNY" "USD.JPY" ...  
## Indexed by objects of class: [Date] TZ: UTC  
## xts Attributes:   
## NULL

exrates.zr <- na.omit(as.zooreg(exrates.xts))  
str(exrates.zr)

## 'zooreg' series from 2012-02-05 to 2017-01-15  
## Data: num [1:259, 1:12] -0.3928 -0.4206 0.4476 -1.4013 0.0231 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:12] "USD.EUR" "USD.GBP" "USD.CNY" "USD.JPY" ...  
## Index: Date[1:259], format: "2012-02-05" "2012-02-12" "2012-02-19" "2012-02-26" "2012-03-04" ...  
## Frequency: 0.142857142857143

head(exrates.xts)

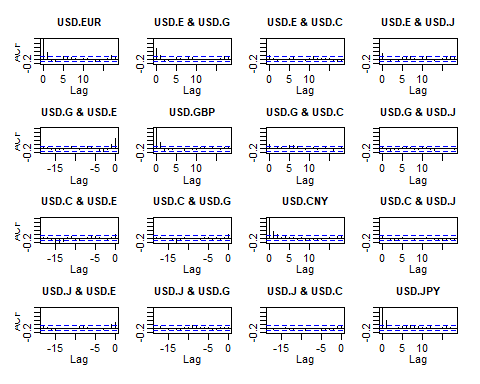
## USD.EUR USD.GBP USD.CNY USD.JPY USD.EUR  
## 2012-02-05 -0.39282058 -0.8523826 -0.01270912 -1.0301113 0.39282058  
## 2012-02-12 -0.42063724 -0.1184583 -0.03130311 1.0571858 0.42063724  
## 2012-02-19 0.44758304 0.2221127 0.06545522 1.9318720 0.44758304  
## 2012-02-26 -1.40130272 -0.2407629 0.01048156 2.0452375 1.40130272  
## 2012-03-04 0.02305476 -0.4546859 0.02588158 1.0197119 0.02305476  
## 2012-03-11 1.19869144 0.7988383 0.22598055 0.6384155 1.19869144  
## USD.GBP USD.CNY USD.JPY USD.EUR USD.GBP USD.CNY USD.JPY  
## 2012-02-05 0.8523826 0.01270912 1.0301113 -1 -1 -1 -1  
## 2012-02-12 0.1184583 0.03130311 1.0571858 -1 -1 -1 1  
## 2012-02-19 0.2221127 0.06545522 1.9318720 1 1 1 1  
## 2012-02-26 0.2407629 0.01048156 2.0452375 -1 -1 1 1  
## 2012-03-04 0.4546859 0.02588158 1.0197119 1 -1 1 1  
## 2012-03-11 0.7988383 0.22598055 0.6384155 1 1 1 1

We can plot with the ggplot2 package. In the ggplot statements we use aes, “aesthetics”, to pick x (horizontal) and y (vertical) axes. Use group =1 to ensure that all data is plotted. The added (+) geom\_line is the geometrical method that builds the line plot.

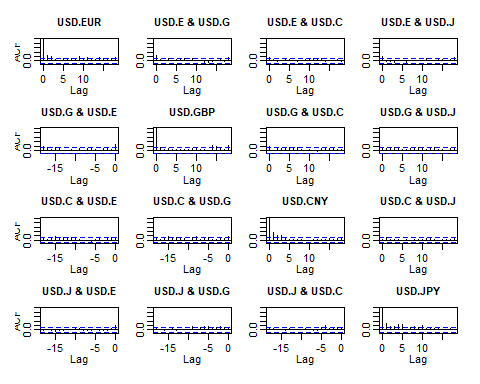
library(ggplot2)  
library(plotly)  
title.chg <- "Exchange Rate Percent Changes"  
p1 <- autoplot.zoo(exrates.xts[, 1:4]) +   
 ggtitle(title.chg) + ylim(-5, 5)  
p2 <- autoplot.zoo(exrates.xts[, 5:8]) +   
 ggtitle(title.chg) + ylim(-5, 5)  
ggplotly(p1)

1. Let’s dig deeper and compute mean, standard deviation, etc. Load the data\_moments() function. Run the function using the exrates data and write a knitr::kable() report.

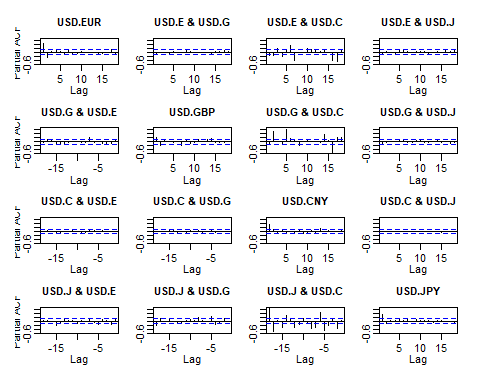
acf(coredata(exrates.xts[, 1:4])) # returns



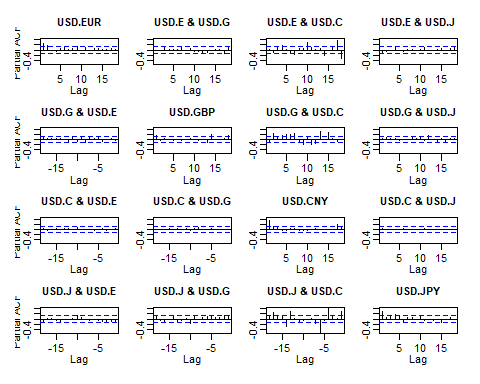
acf(coredata(exrates.xts[, 5:8])) # sizes



pacf(coredata(exrates.xts[, 1:4])) # returns



pacf(coredata(exrates.xts[, 5:8])) # sizes



# Load the data\_moments() function  
# data\_moments function INPUTS: r  
# vector OUTPUTS: list of scalars  
# (mean, sd, median, skewness,  
# kurtosis)  
data\_moments <- function(data) {  
 library(moments)  
 library(matrixStats)  
 mean.r <- colMeans(data)  
 median.r <- colMedians(data)  
 sd.r <- colSds(data)  
 IQR.r <- colIQRs(data)  
 skewness.r <- skewness(data)  
 kurtosis.r <- kurtosis(data)  
 result <- data.frame(mean = mean.r,   
 median = median.r, std\_dev = sd.r,   
 IQR = IQR.r, skewness = skewness.r,   
 kurtosis = kurtosis.r)  
 return(result)  
}  
# Run data\_moments()  
answer <- data\_moments(exrates.xts[,   
 5:8])  
# Build pretty table  
answer <- round(answer, 4)  
knitr::kable(answer)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | mean | median | std\_dev | IQR | skewness | kurtosis |
| USD.EUR | 0.7185 | 0.5895 | 0.5499 | 0.7506 | 1.3773 | 6.3808 |
| USD.GBP | 0.6884 | 0.5601 | 0.6565 | 0.6588 | 4.0555 | 34.3779 |
| USD.CNY | 0.1700 | 0.1118 | 0.2233 | 0.1536 | 4.9157 | 41.4959 |
| USD.JPY | 0.8310 | 0.6358 | 0.7371 | 0.8352 | 1.6373 | 6.3185 |

mean(exrates.xts[, 4])

## [1] 0.154916

## Part 2

We will use the data from the first part to investigate the interactions of the distribution of exchange rates.

### Problem

We want to characterize the distribution of up and down movements visually. Also we would like to repeat the analysis periodically for inclusion in management reports.

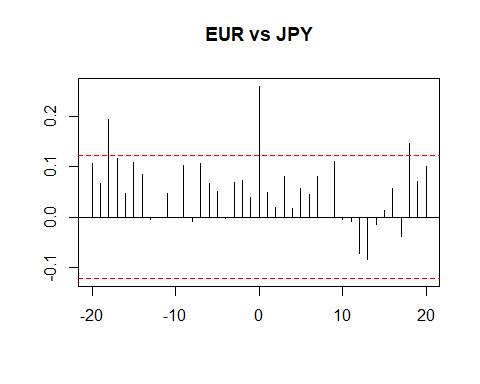
### Questions

1. How can we show the shape of our exposure to euros, especially given our tolerance for risk? Suppose corporate policy set tolerance at 95%. Let’s use the exrates.df data frame with ggplot2 and the cumulative relative frequency function stat\_ecdf.

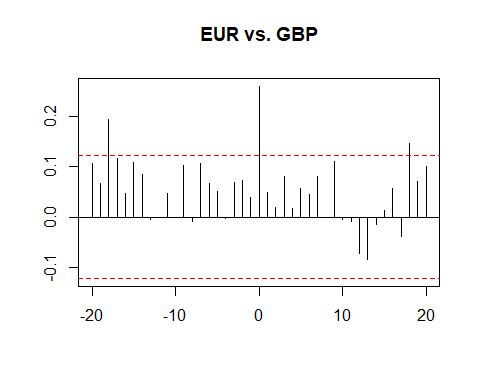
exrates.tol.pct <- 0.95  
exrates.tol <- quantile(exrates.df$returns.USD.EUR,   
 exrates.tol.pct)  
exrates.tol.label <- paste("Tolerable Rate = ",   
 round(exrates.tol, 2), "%", sep = "")  
p <- ggplot(exrates.df, aes(returns.USD.EUR,   
 fill = direction.USD.EUR)) + stat\_ecdf(colour = "blue",   
 size = 0.75, geom = "point") + geom\_vline(xintercept = exrates.tol,   
 colour = "red", size = 1.5) + annotate("text",   
 x = exrates.tol + 1, y = 0.75, label = exrates.tol.label,   
 colour = "darkred")  
ggplotly(p)

1. What is the history of correlations in the exchange rate markets? If this is a “history,” then we have to manage the risk that conducting business in one country will definitely affect business in another. Further that bad things will be followed by more bad things more often than good things. We will create a rolling correlation function, corr\_rolling, and embed this function into the rollapply() function (look this one up!).

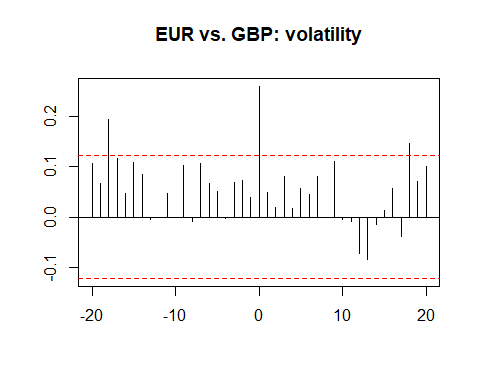
one <- ts(exrates.df$returns.USD.EUR)  
two <- ts(exrates.df$returns.USD.JPY)  
# or  
one <- ts(exrates.zr[, 1])  
two <- ts(exrates.zr[, 2])  
ccf(abs(one), abs(two), main = "EUR vs JPY",   
 lag.max = 20, xlab = "", ylab = "",   
 ci.col = "red")



# build function to repeat these  
# routines  
run\_ccf <- function(one, two, main = "one vs. two",   
 lag = 20, color = "red") {  
 # one and two are equal length series  
 # main is title lag is number of lags  
 # in cross-correlation color is color  
 # of dashed confidence interval  
 # bounds  
 stopifnot(length(one) == length(two))  
 one <- ts(one)  
 two <- ts(two)  
 main <- main  
 lag <- lag  
 color <- color  
 ccf(one, two, main = main, lag.max = lag,   
 xlab = "", ylab = "", ci.col = color)  
 # end run\_ccf  
}  
one <- ts(exrates.df$returns.USD.EUR)  
two <- ts(exrates.df$returns.USD.GBP)  
# or  
one <- exrates.zr[, 1]  
two <- exrates.zr[, 2]  
title <- "EUR vs. GBP"  
run\_ccf(abs(one), abs(two), main = title,   
 lag = 20, color = "red")



# now for volatility (sizes)  
one <- ts(abs(exrates.zr[, 1]))  
two <- ts(abs(exrates.zr[, 2]))  
title <- "EUR vs. GBP: volatility"  
run\_ccf(one, two, main = title, lag = 20,   
 color = "red")



# We see some small raw correlations  
# across time with raw returns. More  
# revealing, we see volatility of  
# correlation clustering using return  
# sizes.

One more experiment, rolling correlations and volatilities using these functions:

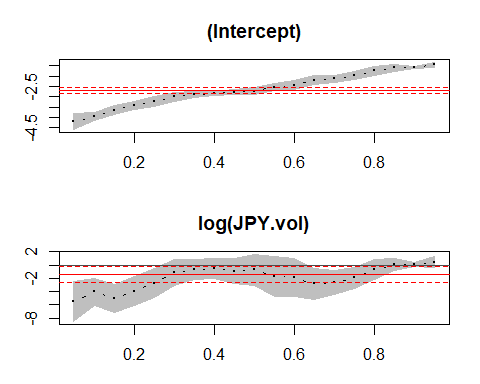
corr\_rolling <- function(x) {  
 dim <- ncol(x)  
 corr\_r <- cor(x)[lower.tri(diag(dim),   
 diag = FALSE)]  
 return(corr\_r)  
}  
vol\_rolling <- function(x) {  
 library(matrixStats)  
 vol\_r <- colSds(x)  
 return(vol\_r)  
}  
ALL.r <- exrates.xts[, 1:4]  
window <- 90 #reactive({input$window})  
corr\_r <- rollapply(ALL.r, width = window,   
 corr\_rolling, align = "right", by.column = FALSE)  
colnames(corr\_r) <- c("EUR.GBP", "EUR.CNY",   
 "EUR.JPY", "GBP.CNY", "GBP.JPY",   
 "CNY.JPY")  
vol\_r <- rollapply(ALL.r, width = window,   
 vol\_rolling, align = "right", by.column = FALSE)  
colnames(vol\_r) <- c("EUR.vol", "GBP.vol",   
 "CNY.vol", "JPY.vol")  
year <- format(index(corr\_r), "%Y")  
r\_corr\_vol <- merge(ALL.r, corr\_r, vol\_r,   
 year)

1. How related are correlations and volatilities? Put another way, do we have to be concerned that inter-market transactions (e.g., customers and vendors transacting in more than one currency) can affect transactions in a single market? Let’s model the the exrate data to understand how correlations and volatilities depend upon one another.

library(quantreg)  
taus <- seq(0.05, 0.95, 0.05) # Roger Koenker UIC Bob Hogg and Allen Craig  
fit.rq.CNY.JPY <- rq(log(CNY.JPY) ~ log(JPY.vol),   
 tau = taus, data = r\_corr\_vol)  
fit.lm.CNY.JPY <- lm(log(CNY.JPY) ~ log(JPY.vol),   
 data = r\_corr\_vol)  
# Some test statements  
CNY.JPY.summary <- summary(fit.rq.CNY.JPY,   
 se = "boot")  
CNY.JPY.summary

##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.05  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -4.21770 0.23339 -18.07170 0.00000  
## log(JPY.vol) -5.54144 1.83946 -3.01254 0.00302  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.1  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -3.96984 0.12516 -31.71711 0.00000  
## log(JPY.vol) -4.05415 1.24143 -3.26569 0.00134  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.15  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -3.65544 0.13846 -26.40133 0.00000  
## log(JPY.vol) -5.04776 1.32758 -3.80223 0.00020  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.2  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -3.41946 0.12533 -27.28267 0.00000  
## log(JPY.vol) -3.90841 1.29266 -3.02355 0.00291  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.25  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -3.24424 0.15377 -21.09852 0.00000  
## log(JPY.vol) -2.70119 1.36542 -1.97829 0.04962  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.3  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -3.00646 0.12988 -23.14804 0.00000  
## log(JPY.vol) -1.16454 1.18878 -0.97961 0.32877  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.35  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.89360 0.08175 -35.39569 0.00000  
## log(JPY.vol) -0.68706 0.89710 -0.76587 0.44489  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.4  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.85405 0.06295 -45.34000 0.00000  
## log(JPY.vol) -0.52545 0.92405 -0.56864 0.57041  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.45  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.78600 0.07884 -35.33537 0.00000  
## log(JPY.vol) -0.88196 1.11528 -0.79080 0.43024  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.5  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.72762 0.10381 -26.27525 0.00000  
## log(JPY.vol) -0.67233 1.41851 -0.47397 0.63617  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.55  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.53524 0.11252 -22.53234 0.00000  
## log(JPY.vol) -1.64576 1.83973 -0.89457 0.37237  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.6  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.43784 0.12978 -18.78487 0.00000  
## log(JPY.vol) -1.85451 1.71940 -1.07858 0.28241  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.65  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.19872 0.14297 -15.37889 0.00000  
## log(JPY.vol) -2.84742 1.41365 -2.01423 0.04567  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.7  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -2.14073 0.11280 -18.97737 0.00000  
## log(JPY.vol) -2.65416 1.08057 -2.45627 0.01511  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.75  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -1.97726 0.13096 -15.09867 0.00000  
## log(JPY.vol) -1.89488 0.94948 -1.99571 0.04767  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.8  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -1.75658 0.14610 -12.02345 0.00000  
## log(JPY.vol) -0.67468 0.92677 -0.72798 0.46769  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.85  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -1.61049 0.09091 -17.71543 0.00000  
## log(JPY.vol) 0.06474 0.56483 0.11462 0.90889  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.9  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -1.57462 0.03229 -48.76555 0.00000  
## log(JPY.vol) 0.06146 0.19074 0.32223 0.74770  
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.95  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -1.45763 0.07599 -19.18140 0.00000  
## log(JPY.vol) 0.45428 0.52841 0.85972 0.39124

plot(CNY.JPY.summary)



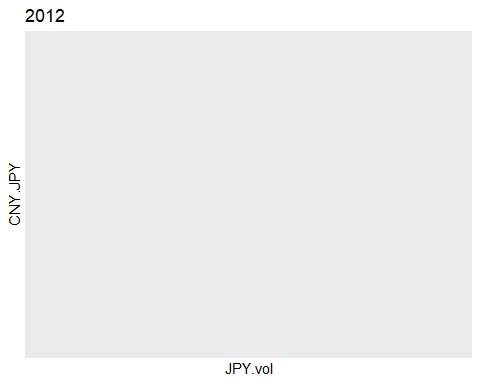
Here is the quantile regression part of the package.

1. We set taus as the quantiles of interest.
2. We run the quantile regression using the quantreg package and a call to the rq function.
3. We can overlay the quantile regression results onto the standard linear model regression.
4. We can sensitize our analysis with the range of upper and lower bounds on the parameter estimates of the relationship between correlation and volatility.
5. The log()-log() transformation allows us to interpret the regression coefficients as elasticities, which vary with the quantile. The larger the elasticity, especially if the absolute value is greater than one, the more risk dependence one market has on the other.
6. The risk relationships can also be viewed year by year. Here we see very different patterns
7. is interpreted as systematic movements in , while unsystematic movements are simply .

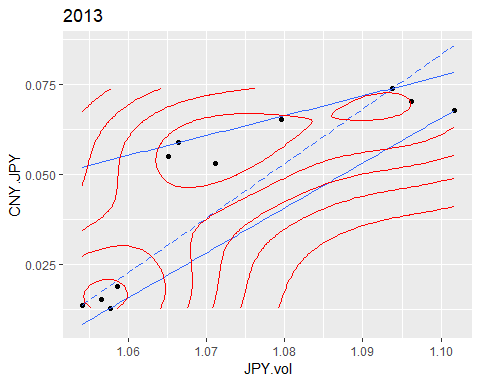
### Animation

library(quantreg)  
library(magick)  
  
datalist <- split(r\_corr\_vol, r\_corr\_vol$year)  
datalist = na.omit(datalist)  
lapply(datalist, function(data) {  
 ggplot(data, aes(JPY.vol, CNY.JPY)) +   
 geom\_point() + ggtitle(data$year) +   
 geom\_quantile(quantiles = c(0.05,   
 0.95)) + geom\_quantile(quantiles = 0.5,   
 linetype = "longdash") + geom\_density\_2d(colour = "red")  
})

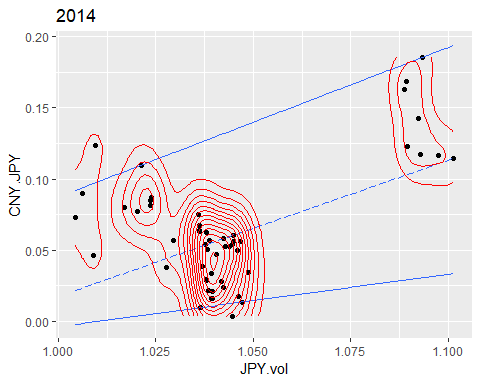
## $`2012`



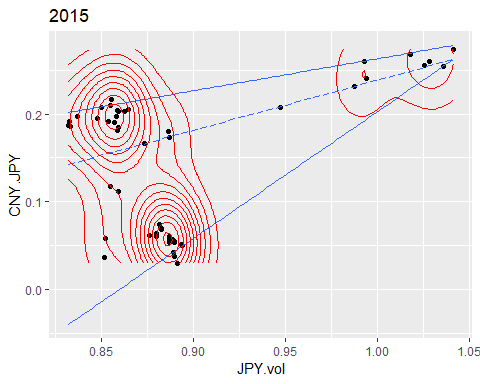
##   
## $`2013`



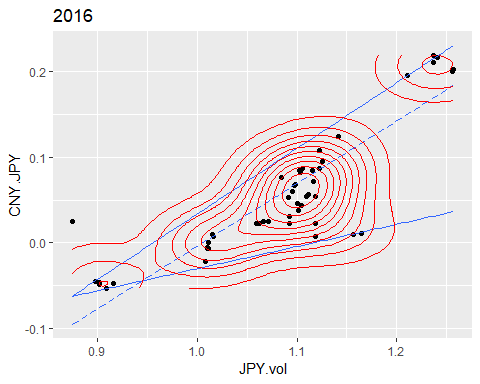
##   
## $`2014`



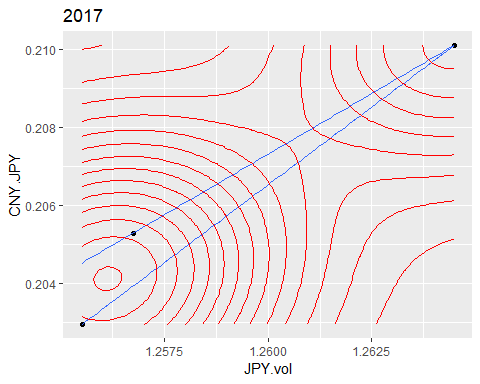
##   
## $`2015`



##   
## $`2016`

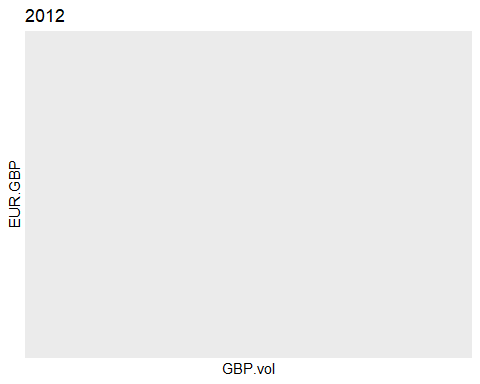


##   
## $`2017`

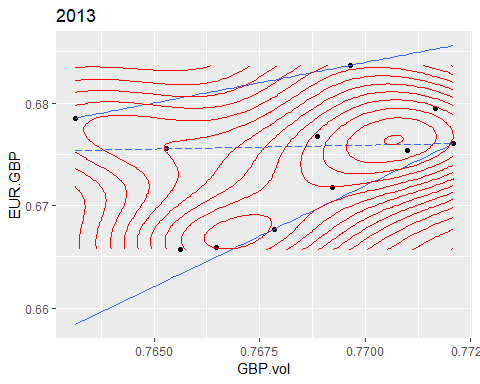


lapply(datalist, function(data) {  
 ggplot(data, aes(GBP.vol, EUR.GBP)) +   
 geom\_point() + ggtitle(data$year) +   
 geom\_quantile(quantiles = c(0.05,   
 0.95)) + geom\_quantile(quantiles = 0.5,   
 linetype = "longdash") + geom\_density\_2d(colour = "red")  
})

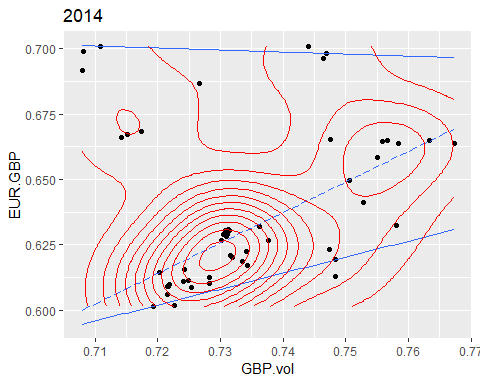
## $`2012`



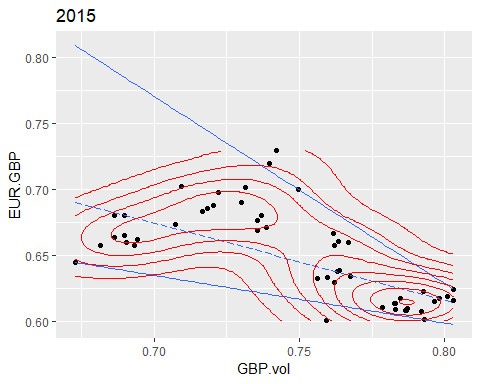
##   
## $`2013`



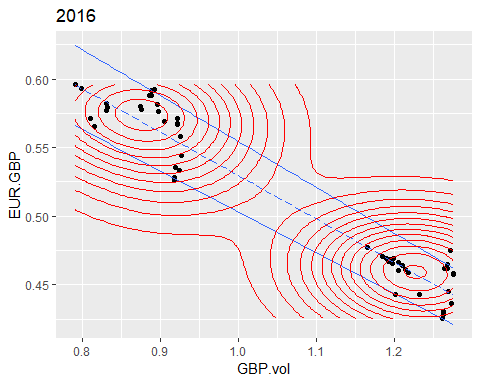
##   
## $`2014`



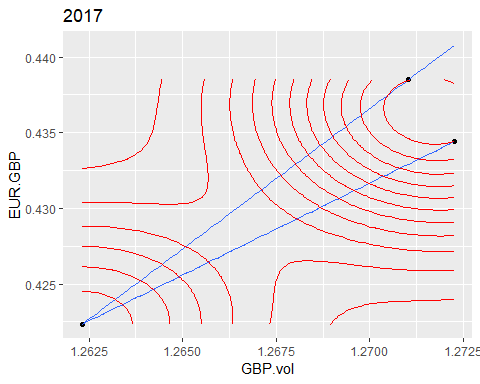
##   
## $`2015`



##   
## $`2016`

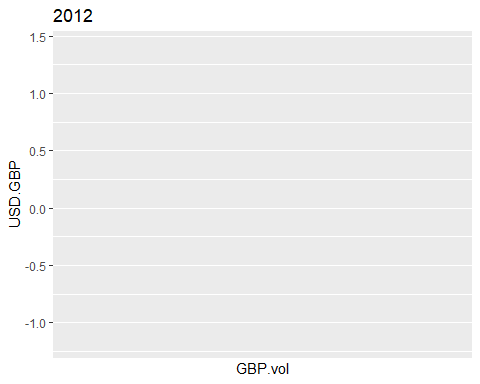


##   
## $`2017`

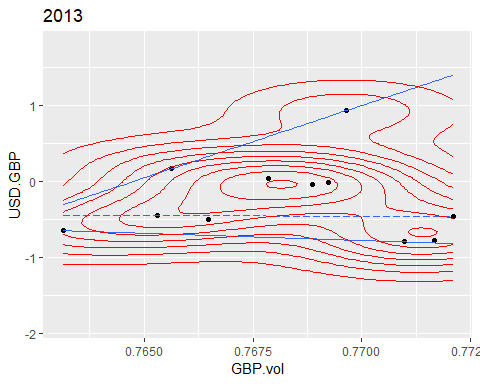


lapply(datalist, function(data) {  
 ggplot(data, aes(GBP.vol, USD.GBP)) +   
 geom\_point() + ggtitle(data$year) +   
 geom\_quantile(quantiles = c(0.05,   
 0.95)) + geom\_quantile(quantiles = 0.5,   
 linetype = "longdash") + geom\_density\_2d(colour = "red")  
})

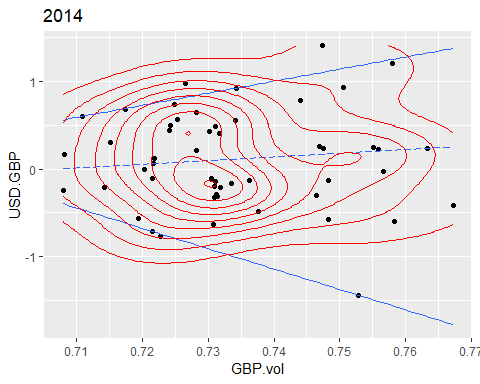
## $`2012`



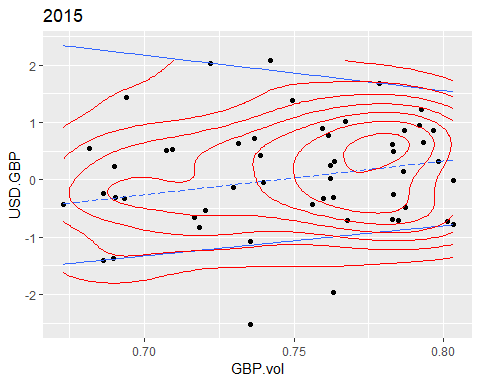
##   
## $`2013`



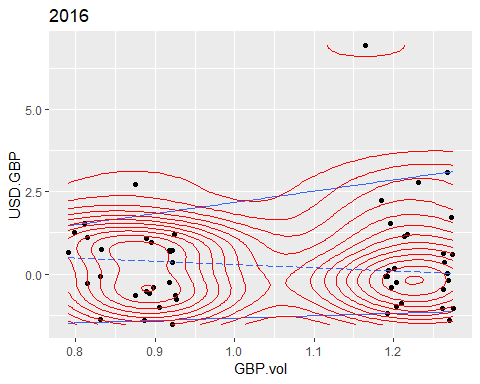
##   
## $`2014`



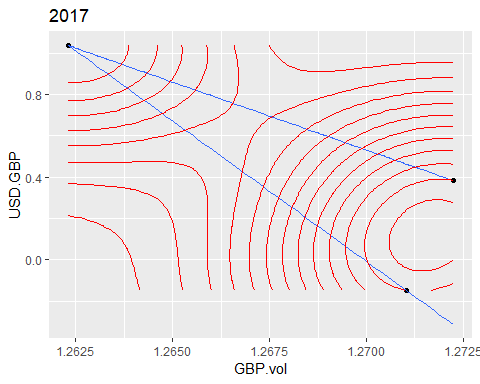
##   
## $`2015`



##   
## $`2016`

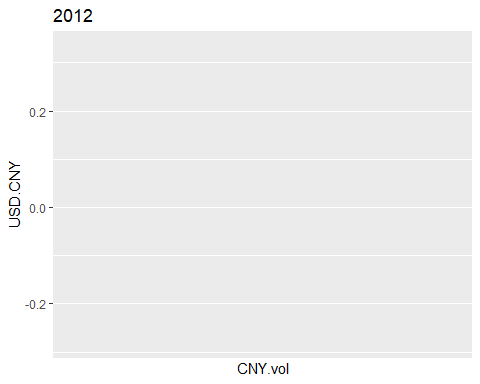


##   
## $`2017`

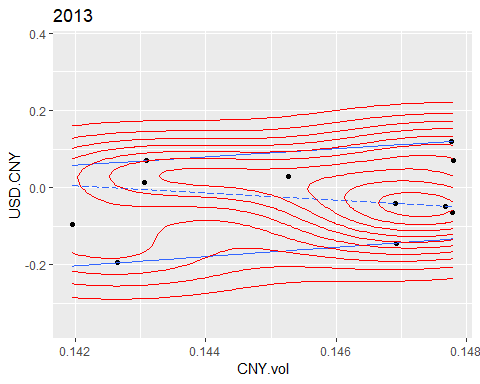


lapply(datalist, function(data) {  
 ggplot(data, aes(CNY.vol, USD.CNY)) +   
 geom\_point() + ggtitle(data$year) +   
 geom\_quantile(quantiles = c(0.05,   
 0.95)) + geom\_quantile(quantiles = 0.5,   
 linetype = "longdash") + geom\_density\_2d(colour = "red")  
})

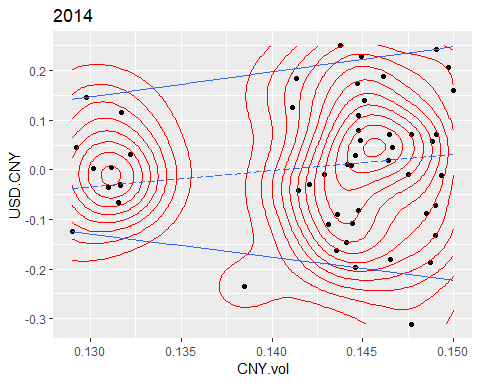
## $`2012`



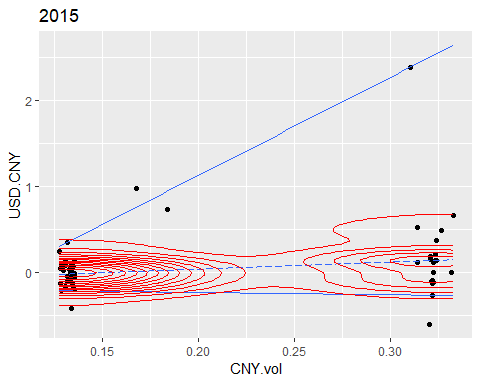
##   
## $`2013`



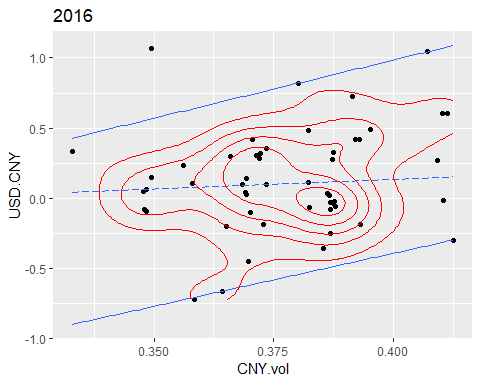
##   
## $`2014`



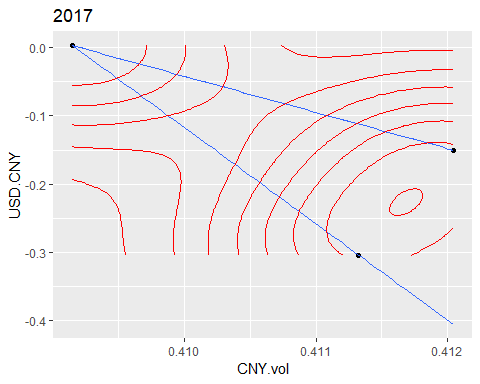
##   
## $`2015`



##   
## $`2016`



##   
## $`2017`



Attempt interpretations to help managers understand the way market interactions affect accounts receivables.

## Notes on lead and lag

In the ccf() function we get results that produce positive and negative lags. A positive lag looks back and a negative lag (a lead) looks forward in the history of a time series. Leading and lagging two different serries, then computing the moments and corelations show a definite asymmetry.

Suppose we lead the USD.EUR return by 5 days and lag the USD.GBP by 5 days. We will compare the correlation in this case with the opposite: lead the USD.GBP return by 5 days and lag the USD.EUR by 5 days. We will use the dplyr package to help us.

library(dplyr)  
x <- as.numeric(exrates.df$returns.USD.EUR) # USD.EUR  
y <- as.numeric(exrates.df$returns.USD.GBP) # USD.GBP  
xy.df <- na.omit(data.frame(date = dates,   
 ahead\_x = lead(x, 5), behind\_y = lag(y,   
 5)))  
yx.df <- na.omit(data.frame(date = dates,   
 ahead\_y = lead(y, 5), behind\_x = lag(x,   
 5)))  
answer <- data\_moments(na.omit(as.matrix(xy.df[,   
 2:3])))  
answer <- round(answer, 4)  
knitr::kable(answer)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | mean | median | std\_dev | IQR | skewness | kurtosis |
| ahead\_x | 0.0872 | 0.1078 | 0.905 | 1.1820 | 0.1671 | 3.6297 |
| behind\_y | 0.0945 | -0.0130 | 0.951 | 1.1134 | 1.7055 | 13.2014 |

answer <- data\_moments(na.omit(as.matrix(yx.df[,   
 2:3])))  
answer <- round(answer, 4)  
knitr::kable(answer)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | mean | median | std\_dev | IQR | skewness | kurtosis |
| ahead\_y | 0.1072 | -0.0047 | 0.9593 | 1.1139 | 1.6409 | 12.7057 |
| behind\_x | 0.0673 | 0.1043 | 0.8953 | 1.1624 | 0.1285 | 3.5876 |

cor(as.numeric(xy.df$ahead\_x), as.numeric(xy.df$behind\_y))

## [1] 0.0003739413

cor(as.numeric(yx.df$ahead\_y), as.numeric(yx.df$behind\_x))

## [1] -0.004339494

Leading x, lagging y will produce a negative correlation. The opposite produces an even smaller and positive correlation. Differences in means, etc. are not huge between the two cases, but when combined produce the correlational differences.

## Conclusion

### Skills and Tools

The main tools implemented for this report include RStudio, along with additional packages:

* zoo
* xts
* ggplot2
* moments
* matrixStats
* quantreg
* magick
* dplyr

The ability to interpret data moments, heteroscedasticity, autocorrelation, and partial autocorrelation are some essential components of the provided analysis.

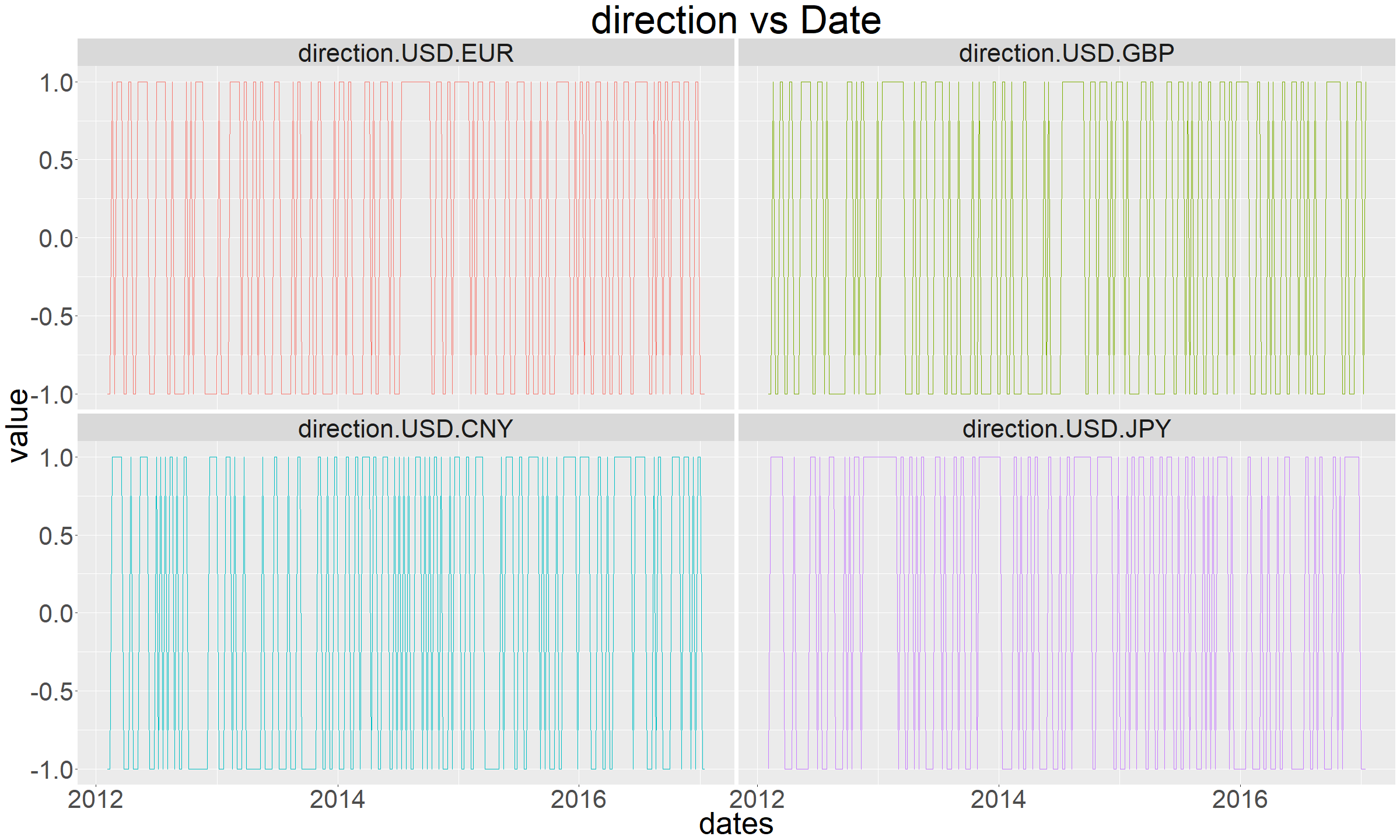
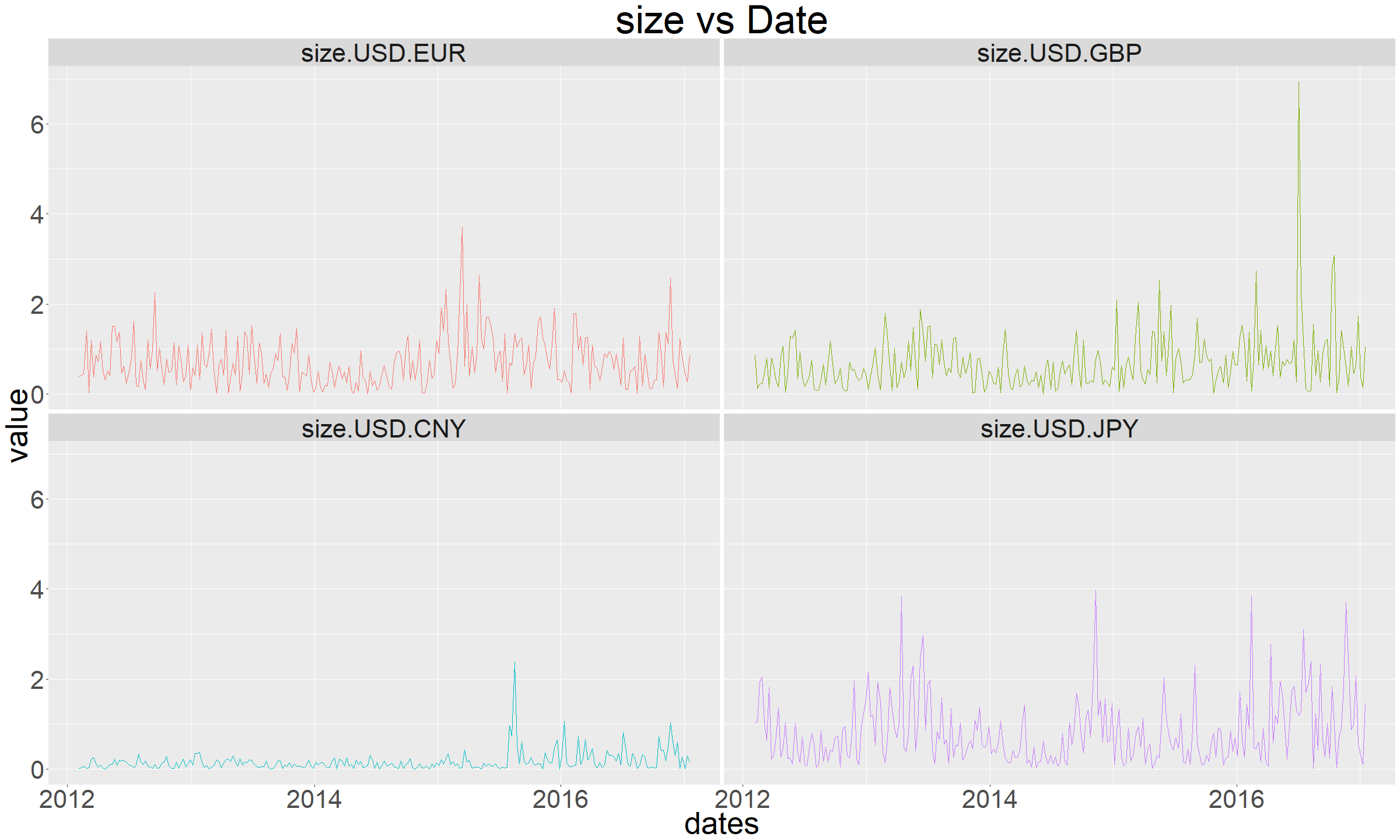
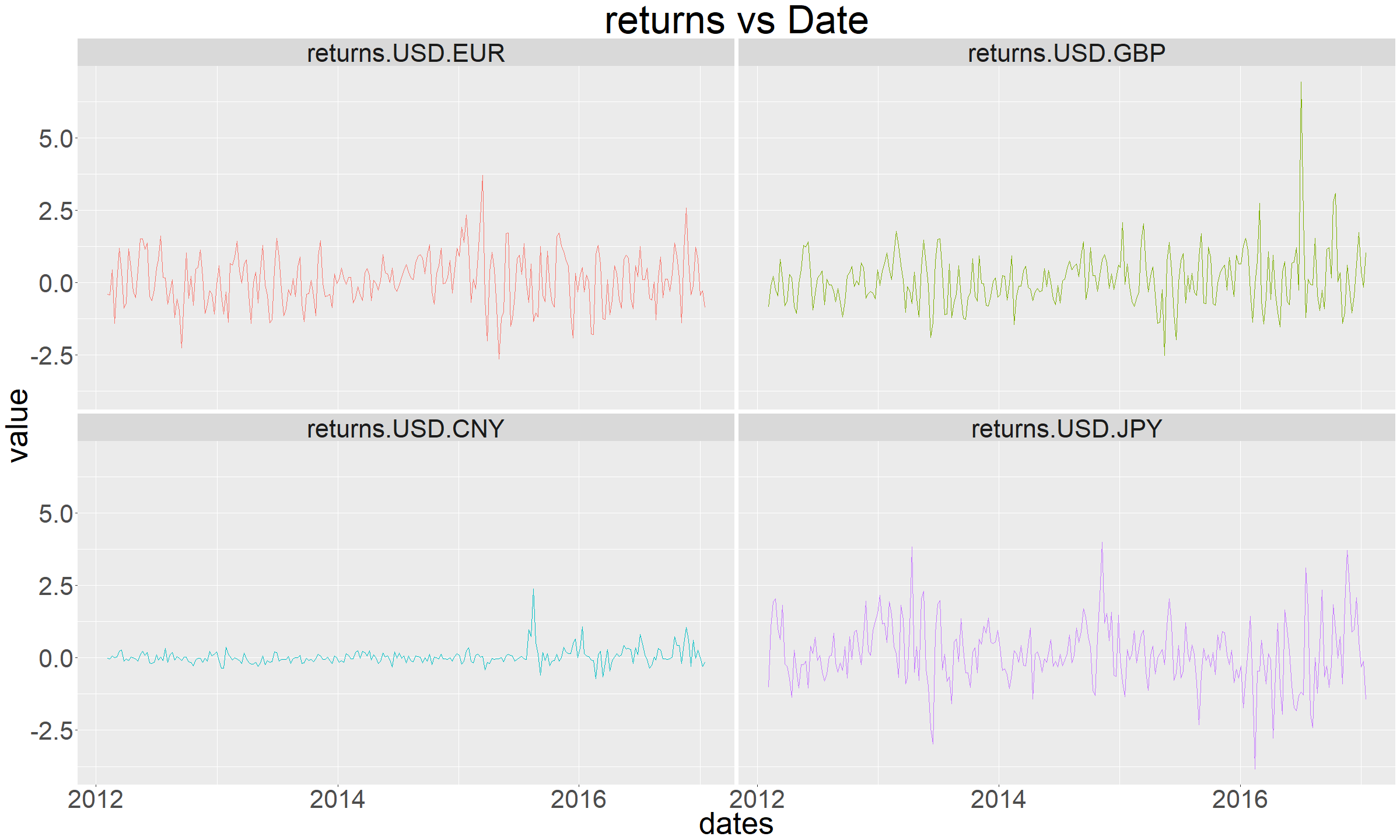
### Data Insights

The data exploration loads the exrates.csv dataset, which includes timeseries on the following exchange rates:

* USD.EUR
* USD.GBP
* USD.CNY
* USD.JPY

Corresponding timeseries plots indicate greatest return volatilility between the EUR, GBP, and JPY:

library(reshape2)  
  
types = c("returns", "size", "direction")  
for (type in types) {  
 meltdf = melt(exrates.df, id.vars = "dates",   
 measure.vars = c(paste0(type,   
 ".USD.EUR"), paste0(type,   
 ".USD.GBP"), paste0(type,   
 ".USD.CNY"), paste0(type,   
 ".USD.JPY")))  
   
 print(ggplot(meltdf, aes(x = dates,   
 y = value, colour = variable,   
 group = variable)) + geom\_line() +   
 ggtitle(paste0(type, " vs Date")) +   
 theme(plot.title = element\_text(hjust = 0.5,   
 size = 50), text = element\_text(size = 40),   
 legend.position = "none") +   
 facet\_wrap(~variable))  
}



Additionally, size serves as another measure of volatility. In the above case, JPY, along with GBP and EUR have the highest level of absolute value change. When computing return autocorrelations, multiple results are found. However, only a handful are significant:

* USD.EUR (D)
* USD.EUR + USD.GBP (D)
* USD.GBP (D)
* USD.CNY (D)
* USD.JPY (D)

Partial autocorrelations also exists:

* USD.EUR
* USD.EUR + USD.CNY
* USD.GBP + USD.CNY
* USD.JPY + USD.GBP
* USD.JPY + USD.CNY

When computing the sizes autocorrelations, multiple results are found. However, only a handful are significant:

* USD.EUR
* USD.GBP
* USD.CNY (D)
* USD.JPY

Partial autocorrelations also exists:

* USD.EUR + ESD.CNY
* USD.G + USD.CNY
* USD.CNY
* USD.JPY + USD.CNY

The above (partial|auto)correlations indicate that different timeseries distributions may have differing levels of serial correlation. Rather, than using the lag.max as blanket solution, or simply concluding that serial correlation can result in successive error terms, more attention may be required to adjust the corresponding timeseries data.

Additionally, the computed data moments provide additional measures of the data distribution:

knitr::kable(answer)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | mean | median | std\_dev | IQR | skewness | kurtosis |
| ahead\_y | 0.1072 | -0.0047 | 0.9593 | 1.1139 | 1.6409 | 12.7057 |
| behind\_x | 0.0673 | 0.1043 | 0.8953 | 1.1624 | 0.1285 | 3.5876 |

As noted earlier, more variance is present between USD.JPY, USD.EUR, and USD.GBP. Additionally, USD.CNY, and USD.GBP has the greatest kurtosis values. This means there are more frequent occuring data points near the tails of the corresponding distribution. Furthermore, USD.GBP has a high standard deviation, and high kurtosis. This indicates investment opportunities are too variable, with significant outliers accounting for high risk (i.e. kurtosis). The best conservative investment is likely USD.EUR, since it has the second smallest kurtosis (by a small margin), while the second smallest standard deviation. The best aggressive investment is likely to be USD.CNY, since it has the highest kurtosis with skewness, while having the smallest standard deviation.

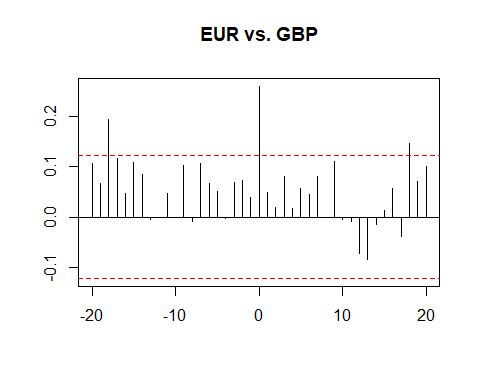
Additionally, multiple rolling correlations were conducted:

* GBP and JPY
* EUR and GBP
* EUR and GBP: volatility

Results indicate correlation largely exists between the dimensions within a 95% confidence. However, each of the plots portray three signals extending beyond the confidence bands. These instances reoccur between the different plots, suggests a potential global event, having a similar globular influence in the market. However, before investigating the nature of a globalular event, a careful determination needs to assess whether the provided timeseries dataset is properly formatted.

An earlier cumulative distribution function, indicate the cutoff tolerance for EUR:

one <- exrates.zr[, 1]  
two <- exrates.zr[, 2]  
title <- "EUR vs. GBP"  
run\_ccf(abs(one), abs(two), main = title,   
 lag = 20, color = "red")



Specifically, when defining a 95% tolerance risk, a corresponding 1.47% tolerable rate is returned. Therefore, the exposure of EUR would need to be decreased if the tolerable rate is exceeded.

Next, quantile regression was used to determine whether inter-market transactions can affect successive transactions in a given market. The tau parameter was used to define the desired quantile level. At each quantile, the coefficients were assessed using a significance measure Pr(>|t|).

Finally, the relationship between CNY.JPN and JPY.vol indicate heteroscedasticity. More specifically, the variability between the volume of JPN varies unequally with CNY.JPN. This behavior is present from 2013 through 2017. Similarly, the comparison between the volume of GBP and EUR.GBP exhibit the same pattern. It is important to note, that both cases exhibit a clumping of data points. Specifically, points with the heteroscedastic plots, generally tend aggregate closer to one another.

### Business Remarks

In general the appreciation or depreciation of the USD will impact cashflow. When a foreign customer purchases goods in a country, during a period of currency appreciation, cash flow to the sellers country will be less than the current value. This occurs since the buyers currency has depreciated against the USD. However, if the appreciation has been accounted, Chinese customers will seek goods, services or trade from another country. Higher trading costs, may decrease US exports and US demand, causing an increase in supply. Sometimes this becomes a cyclic pattern. However, this simplification does not account for the many other factors that contribute to the overall influence.

In this study, the distribution of data provides the ability to characterize exchange rates between EUR, GBP, CNY, JPN with the USD. Using comparative analysis between data moment, the best conservative investment could be argued in favor of USD.EUR. The amount of risk due to outliers is limited, while also having a relatively small standard deviation. When generating a corresponding confidence interval, more confidence can be provided for a smaller margin. Similarly, USD.CNY is likely the best aggressive investment. It is subject to a skewed distribution with a greater tendency of outliers occuring. However, the associated standard deviation is very small, while potentially being error prone due to the skew distribution. Though the smaller standard deviation, and margin is desired, when an undesirable outcome occurs, the risk can be quite high.

Additionally, multiple quantile regression between volumes with the corresponding exchange rate from differing countries, shows heteroscedastic tendency. This means as the volumes increase for a given country, the corresponding exchange rate in a different country has greater variability. In the case between China and the United States, not all years were heteroscedastic. In general, as the Chinese volume increases, the corresponding exchange rate in the United States also increased. This behavior was prevalent between 2013 through 2016. In 2017, only three data points exhibited an overall negative heteroscedastic behavior. With incomplete 2017 data, this specific year should be ignored, unless a rolling regression model is created to predict successive values within the year. Furthermore, between 2014 and 2015, the heteroscedastic data points exhibited heavy clumping. Specifically, points tended to cluster between the lower, as well as the higher Chinese volumes. In 2016, points were more dispersed, while the overall pattern exhibited no heteroscedascity. Thus, it would be interesting to see a complete dataset for the years 2017-2018, then attempt to build an ARIMA model.