# Project 2: Foreign Exchange Markets

#### *Alex Hyman, Matt LaFlair, Robin Kim, Sasha Singh*

#### *8/5/2018*

# Introduction

The marketing and accounts receivable managers at our company notify us that we have significant exposure to exchange rates. Our functional currency is U.S Dollars (USD), but we operate in the United Kingdom, European Union, and Japan. The exposure to exchange rates hits the gross revenue of our financial lines.

Our cash flow is also affected by the ebb and flow of accounts receivable components of working capital in producing and selling several products. When the exchange rates are volatile, so are our earnings. The goal of this project is to explore the relationships between the different markets to get a better understanding of how our earnings are affected by the exchange markets. This is especially important as we have missed our earnings forecasts for five consecutive quarters.

# Part 1

## Importing the Data

First, we will load in time series data of the exchange rates for the European, United Kingdom, Chinese, and Japanese markets. We will obtain the csv file from the turing.manhattan.edu website and view the structure and a sample of the exchange rates file.

**library**(zoo) *#For creating time series objects*  
**library**(xts) *#For time series analysis*  
**library**(ggplot2) *#For creating graphics*  
**library**(plotly)  
  
*#The URL for the exchange data data*  
URL <- "https://turing.manhattan.edu/~wfoote01/finalytics/data/exrates.csv"  
  
*#Reading in the exchange rates and omitting the missing data from the*   
*#url provided by turing.manhattan.edu and keeping the dates as characters*  
exrates <- na.omit(read.csv(url(URL), stringsAsFactors = F))  
*#Converting the string dates to actual dates*  
exrates$DATE <- as.Date(exrates$DATE, "%m/%d/%Y")  
  
*#Five columns (date, eur2usd, gbp2usd, cny2usd, jpy2usd)*  
*#the data is daily exchange rates*  
head(exrates) *#Looking at the data*

## DATE USD.EUR USD.GBP USD.CNY USD.JPY  
## 1 2013-01-28 1.3459 1.5686 6.2240 90.73  
## 2 2013-01-29 1.3484 1.5751 6.2259 90.65  
## 3 2013-01-30 1.3564 1.5793 6.2204 91.05  
## 4 2013-01-31 1.3584 1.5856 6.2186 91.28  
## 5 2013-02-01 1.3692 1.5744 6.2265 92.54  
## 6 2013-02-04 1.3527 1.5737 6.2326 92.57

tail(exrates) *#Looking at the end of the data*

## DATE USD.EUR USD.GBP USD.CNY USD.JPY  
## 1248 2018-01-19 1.2238 1.3857 6.3990 110.56  
## 1249 2018-01-22 1.2230 1.3944 6.4035 111.15  
## 1250 2018-01-23 1.2277 1.3968 6.4000 110.46  
## 1251 2018-01-24 1.2390 1.4198 6.3650 109.15  
## 1252 2018-01-25 1.2488 1.4264 6.3189 108.70  
## 1253 2018-01-26 1.2422 1.4179 6.3199 108.38

str(exrates) *#Viewing the structure of the data*

## 'data.frame': 1253 obs. of 5 variables:  
## $ DATE : Date, format: "2013-01-28" "2013-01-29" ...  
## $ USD.EUR: num 1.35 1.35 1.36 1.36 1.37 ...  
## $ USD.GBP: num 1.57 1.58 1.58 1.59 1.57 ...  
## $ USD.CNY: num 6.22 6.23 6.22 6.22 6.23 ...  
## $ USD.JPY: num 90.7 90.7 91 91.3 92.5 ...

*#1253 different instances of exchange rates*  
summary(exrates) *#From 28 Jan 2013 to 26 Jan 2018*

## DATE USD.EUR USD.GBP USD.CNY   
## Min. :2013-01-28 Min. :1.038 Min. :1.212 Min. :6.040   
## 1st Qu.:2014-04-25 1st Qu.:1.107 1st Qu.:1.324 1st Qu.:6.178   
## Median :2015-07-27 Median :1.158 Median :1.514 Median :6.261   
## Mean :2015-07-26 Mean :1.199 Mean :1.474 Mean :6.401   
## 3rd Qu.:2016-10-24 3rd Qu.:1.314 3rd Qu.:1.573 3rd Qu.:6.627   
## Max. :2018-01-26 Max. :1.393 Max. :1.716 Max. :6.958   
## USD.JPY   
## Min. : 90.65   
## 1st Qu.:102.14   
## Median :109.88   
## Mean :109.33   
## 3rd Qu.:116.76   
## Max. :125.58

*# USD to CNY appears to be the most steady*

## Question 1: Nature of Exchange Rates

**My comment and changes will be in blue - Matthew LaFlair**

Exchange rates are simply the value, or purchasing power of one nation's currency in another’s. The exchange rates within this project are viewed in terms in USD, as it is our main business location. Because we are interested in how each exchange rate changes over time we will want to look at the percent change in the exchange rate on a daily basis. To calculate the percent change in the exchange rate over time, we will use the difference of logarithms for the sequential data. These calculated numbers will be in the units of percent change.

*#Using the properties of the natural log to determine*  
*#the percent change in exchange rates*  
exrates.r <- diff(log(as.matrix(exrates[,-1]))) \* 100  
head(exrates.r) *#first 6 days of percent change*

## USD.EUR USD.GBP USD.CNY USD.JPY  
## 2 0.1855770 0.41352605 0.03052233 -0.08821260  
## 3 0.5915427 0.26629486 -0.08837968 0.44028690  
## 4 0.1473405 0.39811737 -0.02894123 0.25228994  
## 5 0.7919091 -0.70886373 0.12695761 1.37092779  
## 6 -1.2124033 -0.04447127 0.09792040 0.03241316  
## 7 0.3100091 -0.54159233 -0.05456676 0.82836254

tail(exrates.r) *#last 6 days of percent change*

## USD.EUR USD.GBP USD.CNY USD.JPY  
## 1248 0.00000000 -0.2306640 -0.28869056 -0.2890175  
## 1249 -0.06539153 0.6258788 0.07029877 0.5322280  
## 1250 0.38356435 0.1719691 -0.05467255 -0.6227176  
## 1251 0.91621024 1.6332111 -0.54837584 -1.1930381  
## 1252 0.78784876 0.4637771 -0.72690896 -0.4131289  
## 1253 -0.52990891 -0.5976884 0.01582429 -0.2948224

str(exrates.r) *#Shows there are 4 columns with 1252 instances*

## num [1:1252, 1:4] 0.186 0.592 0.147 0.792 -1.212 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : chr [1:1252] "2" "3" "4" "5" ...  
## ..$ : chr [1:4] "USD.EUR" "USD.GBP" "USD.CNY" "USD.JPY"

*#creating a matrix for the volatility of the exchange markets*  
size <- na.omit(abs(exrates.r))  
head(size) *#viewing the first 6 rows of the size matrix*

## USD.EUR USD.GBP USD.CNY USD.JPY  
## 2 0.1855770 0.41352605 0.03052233 0.08821260  
## 3 0.5915427 0.26629486 0.08837968 0.44028690  
## 4 0.1473405 0.39811737 0.02894123 0.25228994  
## 5 0.7919091 0.70886373 0.12695761 1.37092779  
## 6 1.2124033 0.04447127 0.09792040 0.03241316  
## 7 0.3100091 0.54159233 0.05456676 0.82836254

*#Creating a vector of names for the matrix that has the same name as*  
*#the exrates matrix, but with ".size" appended at the end*  
colnames(size) <- paste(colnames(size), ".size", sep = "")  
*#viewing the first six rows exchange rates size matrix*  
head(size)

## USD.EUR.size USD.GBP.size USD.CNY.size USD.JPY.size  
## 2 0.1855770 0.41352605 0.03052233 0.08821260  
## 3 0.5915427 0.26629486 0.08837968 0.44028690  
## 4 0.1473405 0.39811737 0.02894123 0.25228994  
## 5 0.7919091 0.70886373 0.12695761 1.37092779  
## 6 1.2124033 0.04447127 0.09792040 0.03241316  
## 7 0.3100091 0.54159233 0.05456676 0.82836254

*#Creating an empty matrix with the dimensions of exrates.r*  
direction <- exrates.r  
direction[exrates.r > 0] <- 1 *#setting the matrix > 0 = 1*  
direction[exrates.r < 0] <- -1 *#setting the matrix > 0 = -1*  
direction[exrates.r == 0] <- 0 *#setting the matrix == 0 = 0*  
*#setting the column names to the exchange market with ".dir" appended*  
colnames(direction) <- paste(colnames(exrates.r), ".dir", sep = "")  
  
*#Converting into a time series object*  
*#Vector of only dates*  
*#removing the first index of the dates because of the diff function*  
dates <- exrates$DATE[-1]  
*#Creating a matrix that has has the percent change, absolute value*  
*#of the percent change, and the whether the foreign exchange*   
*#appreciated, depreciated, or stayed the same*  
values <- cbind(exrates.r, size, direction)  
*#Creating the data frame with the dates, returns, size, and direction*  
exrates.df = data.frame(dates = dates, returns = exrates.r,  
 size = size, direction = direction)  
*#Viewing structure of the data to ensure all is looking normal*  
str(exrates.df)

## 'data.frame': 1252 obs. of 13 variables:  
## $ dates : Date, format: "2013-01-29" "2013-01-30" ...  
## $ returns.USD.EUR : num 0.186 0.592 0.147 0.792 -1.212 ...  
## $ returns.USD.GBP : num 0.4135 0.2663 0.3981 -0.7089 -0.0445 ...  
## $ returns.USD.CNY : num 0.0305 -0.0884 -0.0289 0.127 0.0979 ...  
## $ returns.USD.JPY : num -0.0882 0.4403 0.2523 1.3709 0.0324 ...  
## $ size.USD.EUR.size : num 0.186 0.592 0.147 0.792 1.212 ...  
## $ size.USD.GBP.size : num 0.4135 0.2663 0.3981 0.7089 0.0445 ...  
## $ size.USD.CNY.size : num 0.0305 0.0884 0.0289 0.127 0.0979 ...  
## $ size.USD.JPY.size : num 0.0882 0.4403 0.2523 1.3709 0.0324 ...  
## $ direction.USD.EUR.dir: num 1 1 1 1 -1 1 -1 -1 -1 1 ...  
## $ direction.USD.GBP.dir: num 1 1 1 -1 -1 -1 1 1 1 -1 ...  
## $ direction.USD.CNY.dir: num 1 -1 -1 1 1 -1 1 1 0 0 ...  
## $ direction.USD.JPY.dir: num -1 1 1 1 1 1 1 -1 -1 1 ...

*#Converts the matrix into a time series object*  
exrates.xts <- na.omit(as.xts(values, dates))  
*#Viewing the structure of the time series to make sure nothing is wrong*  
str(exrates.xts)

## An 'xts' object on 2013-01-29/2018-01-26 containing:  
## Data: num [1:1252, 1:12] 0.186 0.592 0.147 0.792 -1.212 ...  
## - 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

*#converting from the xts object to a zooreg object*  
exrates.zr <- na.omit(as.zooreg(exrates.xts))  
*#Looking at the structure of the of the zooreg object*  
str(exrates.zr)

## 'zooreg' series from 2013-01-29 to 2018-01-26  
## Data: num [1:1252, 1:12] 0.186 0.592 0.147 0.792 -1.212 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:12] "USD.EUR" "USD.GBP" "USD.CNY" "USD.JPY" ...  
## Index: Date[1:1252], format: "2013-01-29" "2013-01-30" "2013-01-31" "2013-02-01" "2013-02-04" ...  
## Frequency: 1

The result of the above code converts the daily foreign exchange rates to the percent change in the foreign exchange rates, creates a matrix of the absolute value of the percent change, and creates another matrix that indicates whether the exchange rate appreciated, depreciated, or remained the same. Finally, the above block combines all the data created and converts them in to a zoo time series object using the date vector.

Next, we will create an interactive visual plot of the percent daily change for the four exchange markets ranging from 2013 to 2018 using the ggplot2 and plotly libraries.

*#Creates the title for the the plot*  
title.chg <- "Exchange Rates Percent Change"  
*#Creating the ggplot object for the percent change with a y-limit of -5% to 5%*  
*#for each of the four foreign exchange market*  
p1 <- autoplot.zoo(exrates.zr[,1:4]) + ggtitle(title.chg) + ylim(-5,5)  
*#Displays an interactive plot for the ggplot object created above*  
ggplotly(p1)

-5.0-2.50.02.55.0-5.0-2.50.02.55.0-5.0-2.50.02.55.0201420162018-5.0-2.50.02.55.0

Exchange Rates Percent Change IndexUSD.EURUSD.GBPUSD.CNYUSD.JPY

*#Sets a title for the size plot*  
title.size <- "Absolute Value of Percent Change"  
*#Creates a ggplot object for the size of the the change with the y limit*   
*#from 0 to 5*  
p2 <- autoplot.zoo(exrates.zr[,5:8]) + ggtitle(title.size) + ylim(0,5)

The above interactive plots show the daily percent change in the foreign exchange markets relative to the U.S Dollar. These plots also show that the foreign exchange markets for the Japanese yen consistently displays the most volatile behavior, while the exchange market for the Chinese Yuan appears to be the most steady. The foreign exchange markets for the british pound and the euro have a fairly similar volatility, somewhere between the Yen and the Yuan. This makes intuitive sense as until recently, the UK was a part of the European Union and had their markets fairly entwined. There is some excess volatility in the GBP exchange market around June 2016 as the United Kingdom voted to leave the European Union, allowing for political speculation of the future of the UK.

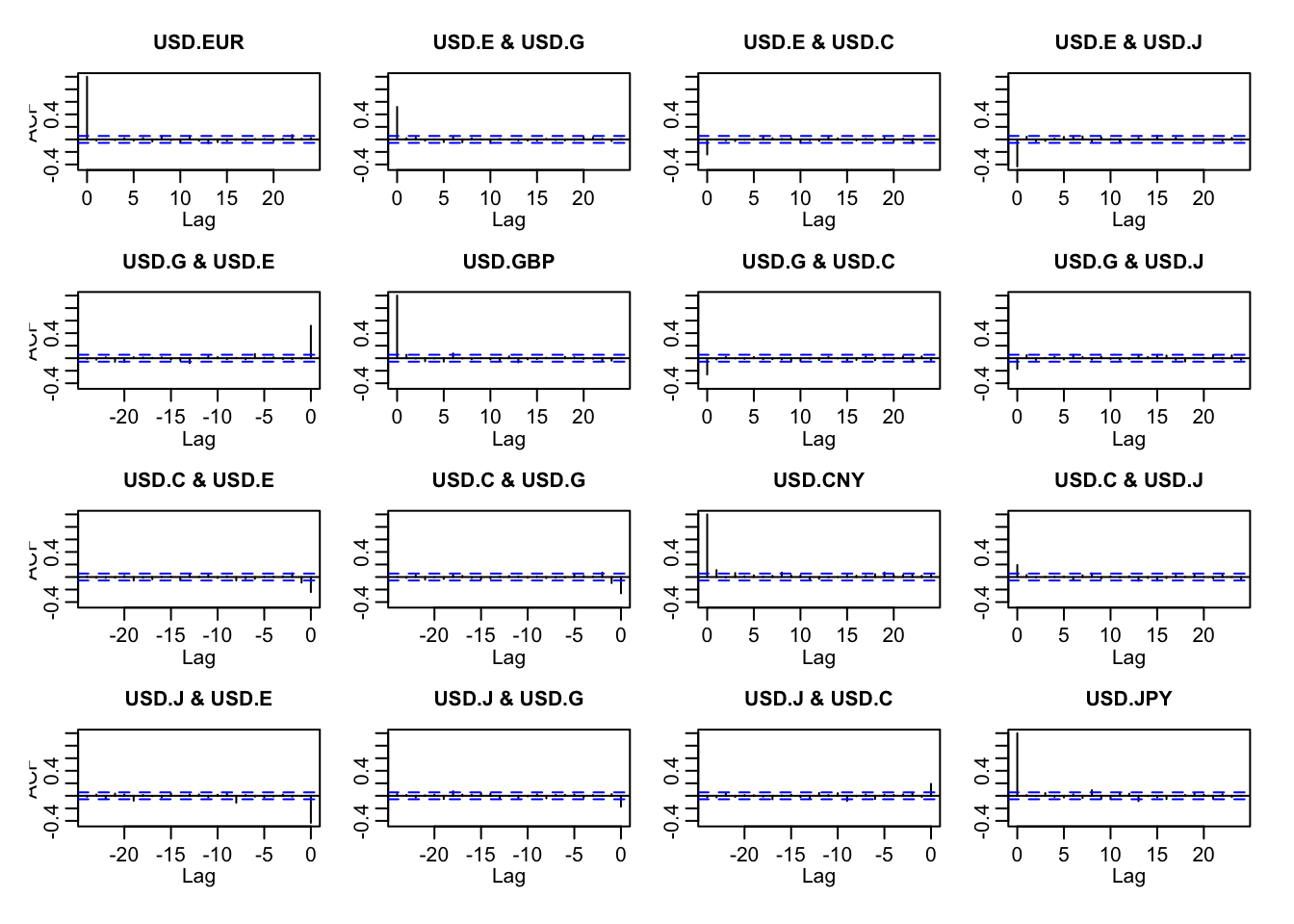
The relatively flat line for the percent change in the exchange rate from USD to CNY shows that the value of the yuan is generally fixed against the price of the dollar, meaning it you could buy approximately the same amount of yuan with the U.S Dollar during any time in that timespan. This fixed price against the dollar is significant, because it keeps the Yuan cheap, even as the Chinese economy continues to grow. Although, it is worth noting that the yuan exchange rate has shown some an increase in volatility, in particular its magnitudes since late 2015, and more recently in 2017. This may be a result of a key yuan index changing its calculation method.

Using this plot, we can see that the volatility in our earnings due to the foreign exchange market is primarily caused by our business in Japan. The volatility in GBP and Euro markets is not as significant as the Yen, but could cause significant issues to our cash flow, especially depending on the volume of revenue. If we were looking at potentially accessing the Chinese market to stabilize some of the volatility in our cash flow, we would want to do more research on the barrier of entry to the Chinese market as the economy of China is dominated by State owned enterprises, and thus the political risk may outweigh the financial gains. It would also be beneficial to see if our prices would be competitive in the Chinese market as the Chinese economy has been growing at a higher rate than the United States, while valuing the currency at approximately the same price, seemingly devaluing the Yuan.

## Question 2: Foreign Exchange Market Relationships and Descriptive Statistics

Next, we will be interested in seeing how the foreign exchange markets interact, and whether events in the past have an effect on the current market. This will be viewed in terms of the autocorrelations in percent change and autocorrelations in the size of the change. Using the partial autocorrelation, we will be able to see how long of a memory the markets have and which days are significant in determining today’s market. First we will create an autocorrelation matrix for the percent change in the markets.

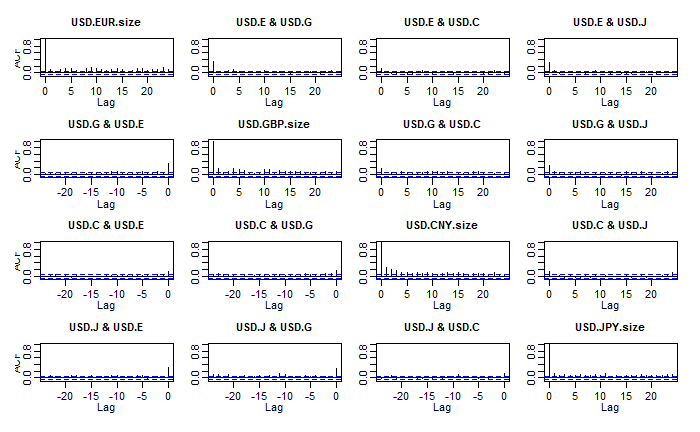
*#Creating an autocorrelation matrix for each market*  
*#The diagonal is shows the memory in the past for the same market*  
*#The x-axis is the amount of lag*   
*#If x-axis is negative, the exchange market that is listed first*  
*#in the title is the exchange market that is being lagged*  
acf(coredata(exrates.xts[,1:4]))

 The vertical lines that appear along the x-axis show the autocorrelation coefficient, and vertical lines that appear above the dashed line have a significant autocorrelation coefficient. The autocorrelation matrix shows that there is some correlation in the percent change between the markets on the same day (zero lag), but not really any memory in the markets to create accurate forecasts with this data alone.

The most significant same day correlations are between the Euro/GBP interaction and between the Euro/JPY interaction. Same day positive movement, either by the GBP or EUR yields positive movement in the other. With the correlation coefficient of ~0.5, 25% of the moment in either the EUR or GBP can be explained by the other . The interactions of the Chinese Yuan and every other market appears to be less significant than the all other interactions. There also appears to be little interaction between the change in the GBP exchange rate and the change in the JPY exchange rate.

Next, we will create an autocorrelation matrix for the sizes in the percent changes. Interactions that are significant will mean that a large magnitude of returns breeds more large magnitude of returns (independent of direction), and that a small magnitude of returns breeds more smaller magnitude of returns.

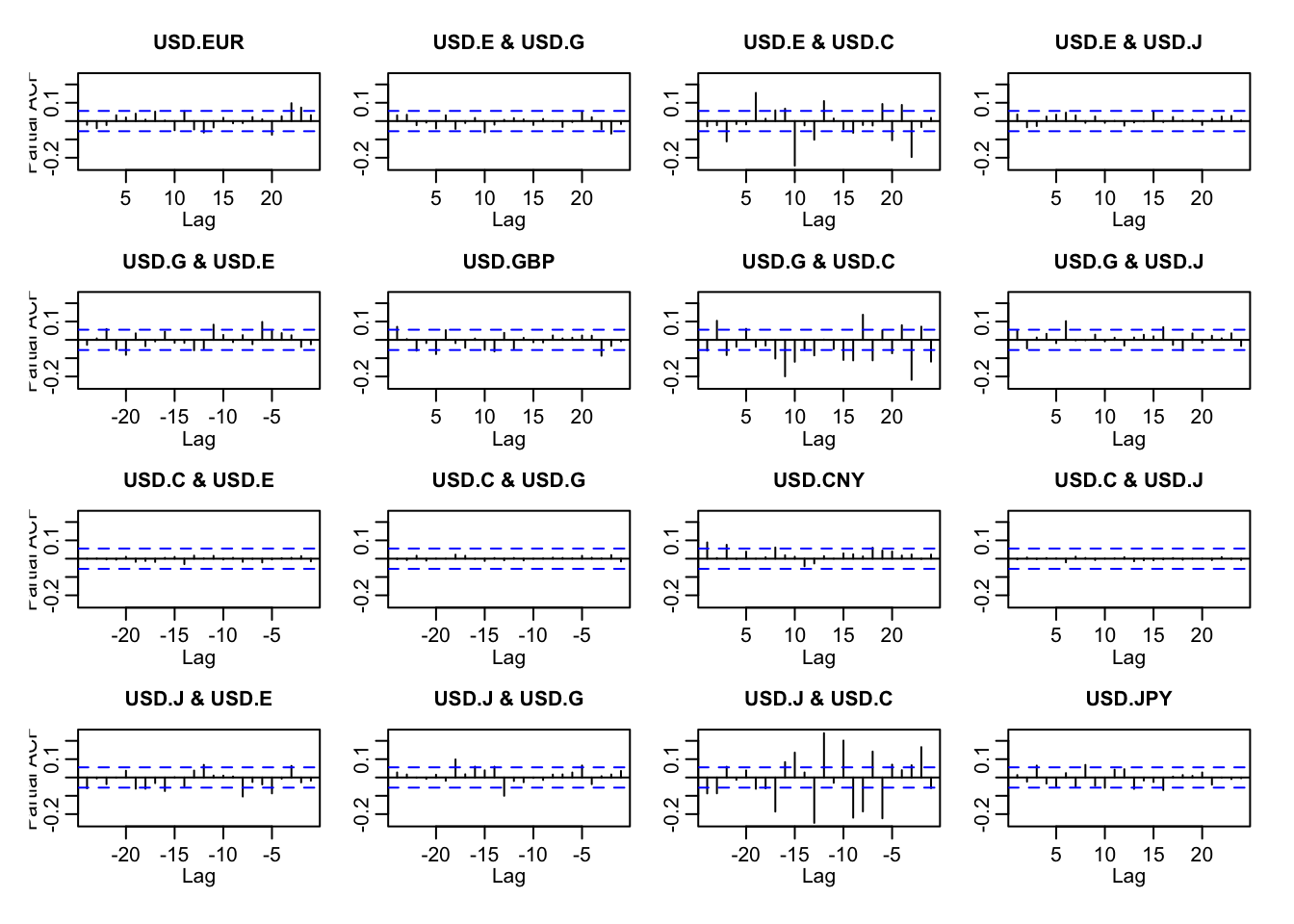
*#Creating an autocorrelation matrix for the foreign*  
*#exchange markets for columns 5-7 (Size of the*   
*#percent change). Not including the japanese market*  
*#to have larger plots*  
acf(coredata(exrates.xts[,5:7]))

**

Using the autocorrelation matrix for the magnitude of percent change in the foreign exchange markets, we can see that there is some memory within the same market. This can be determined by looking along the diagonal of the matrix and seeing a number of vertical lines outside the range of the confidence interval. The memory within the same market is also more relevant in the short-term (lags 1-5) as the frequency of autocorrelation intervals above the confidence interval is greater than the frequency of significant autocorrelation terms with a greater lag; but, there seems to be some persistence in the market as there are some coefficients outside the confidence interval around a lag of 20. This graphic shows that it is likely that a certain magnitude of returns today will bring a like magnitude of returns in the future. Whether the exchange rate will appreciate or depreciate is not within the scope of this graphic as we are only looking at the size of the percent change.

Next we will use the partial autocorrelation between the foreign exchange markets to see if there is any correlation in the interactions between markets in the past, independent of what has happened between the lag date and the present.

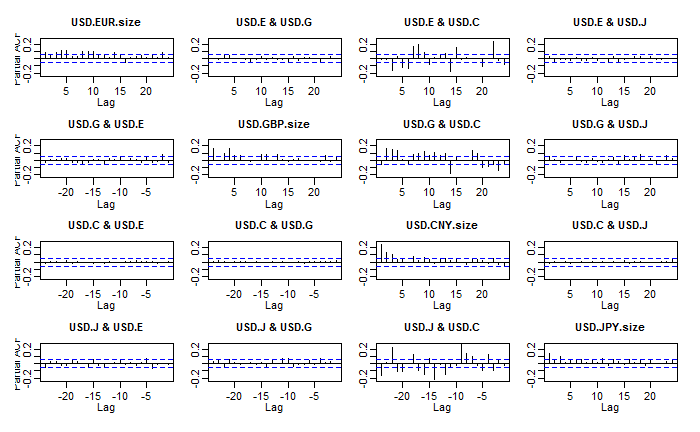
*#Creating a partial*   
pacf(coredata(exrates.xts[,1:4]))



The partial autocorrelation matrix for the foreign exchange markets shows that there is a significant amount of memory in the GBP and EUR exchange markets in terms of the CNY exchange rate. This is particularly apparent around the 10 lag mark. The partial autocorrelation matrix also shows that there is a significant amount of memory in the JPY exchange in terms of past events in the CNY exchange. While it is interesting to note that the partial autocorrelation of the CNY exchange is significant with all the other exchanges, it could also be worth noticing that there is not a lot of memory in the CNY exchange when looking at last event in its own exchange, which is typical for the other exchanges, but the CNY exchange does not appear to behave in the same manner as the other exchanges. This could be the result of the CNY exchange essentially mimicking the daily percent changes of the USD.

Next we will create a partial autocorrelation plot that will show whether the size of the percent change in the exchange rate in the past has an effect on the percent change of today’s rate, independent of what has happened between the lag and the present.

pacf(coredata(exrates.xts[,5:7]))



The partial autocorrelation for value of the percent change in the market indicate that the EUR exchange rate maintains significant memory up to 15 days prior, and even some correlation to around 25 days ago. The GBP shows similar artifacts up to 14 days prior, surprisingly there is little relationship between the EUR and GBP in terms of volatility. The most important information of these plots however is how the Chinese yuan effects significantly affects the volatility in all other markets, while it is show non volatility relationship of others on its own. The yuan market maintains a memory of day lags of 1, 2, 3, 7.

data\_moments <- **function**(data) {  
 **library**(moments) *#Package for skewness and kurtosis*  
 **library**(matrixStats) *#Package for statistics on matrix columns*  
 mean.r <- colMeans(data) *#Calculates the mean for each column*  
 median.r <- colMedians(data) *#Calculates the median for each column*  
 sd.r <- colSds(data) *#Standard deviation for each column*  
 IQR.r <- colIQRs(data) *#Difference between Q1 and Q3*  
 skewness.r <- skewness(data) *#Skewness for each column*  
 kurtosis.r <- kurtosis(data) *#kurtosis for each column*  
 *#Creates a data frame with the statistics for each column*  
 result <- data.frame(mean = mean.r,  
 median = median.r, std\_dev = sd.r,   
 IQR = IQR.r, skewness = skewness.r,   
 kurtosis = kurtosis.r)  
 **return**(result)  
}  
answer <- data\_moments(exrates.xts[,5:8])  
knitr::kable(answer, digits = 4)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **mean** | **median** | **std\_dev** | **IQR** | **skewness** | **kurtosis** |
| USD.EUR.size | 0.4003 | 0.2935 | 0.3695 | 0.4313 | 1.7944 | 8.0424 |
| USD.GBP.size | 0.4008 | 0.2995 | 0.4266 | 0.4173 | 6.0881 | 93.5604 |
| USD.CNY.size | 0.1027 | 0.0601 | 0.1375 | 0.1154 | 3.9004 | 31.0222 |
| USD.JPY.size | 0.4533 | 0.3250 | 0.4455 | 0.4684 | 2.2201 | 10.4898 |

The data moments table above shows the calculated statistics for each of the four foreign exchange markets. In support of our initial analysis, the mean and median volatility for the Euro and the Pound are relatively similar. The standard deviation for the pound is slightly greater, most likely from the increased volatility due to Brexit in 2016. This increased volatility in the GBP is also an explanation for the huge kurtosis number. This high kurtosis means that there are more extreme events in the tails of the distribution. The skewness of the USD to GBP is positive, but this does not necessarily mean the dollar is appreciating in these extreme situation, as this is the absolute value of change, not indicating a particular direction. Further analysis

Calculate the mean for returns; comment on japanese yen being greater than the other returns

colMeans(exrates.xts[, 1:4])

## USD.EUR USD.GBP USD.CNY USD.JPY   
## -0.006404068 -0.008067620 0.001221294 0.014197724

# Part 2

### Introduction

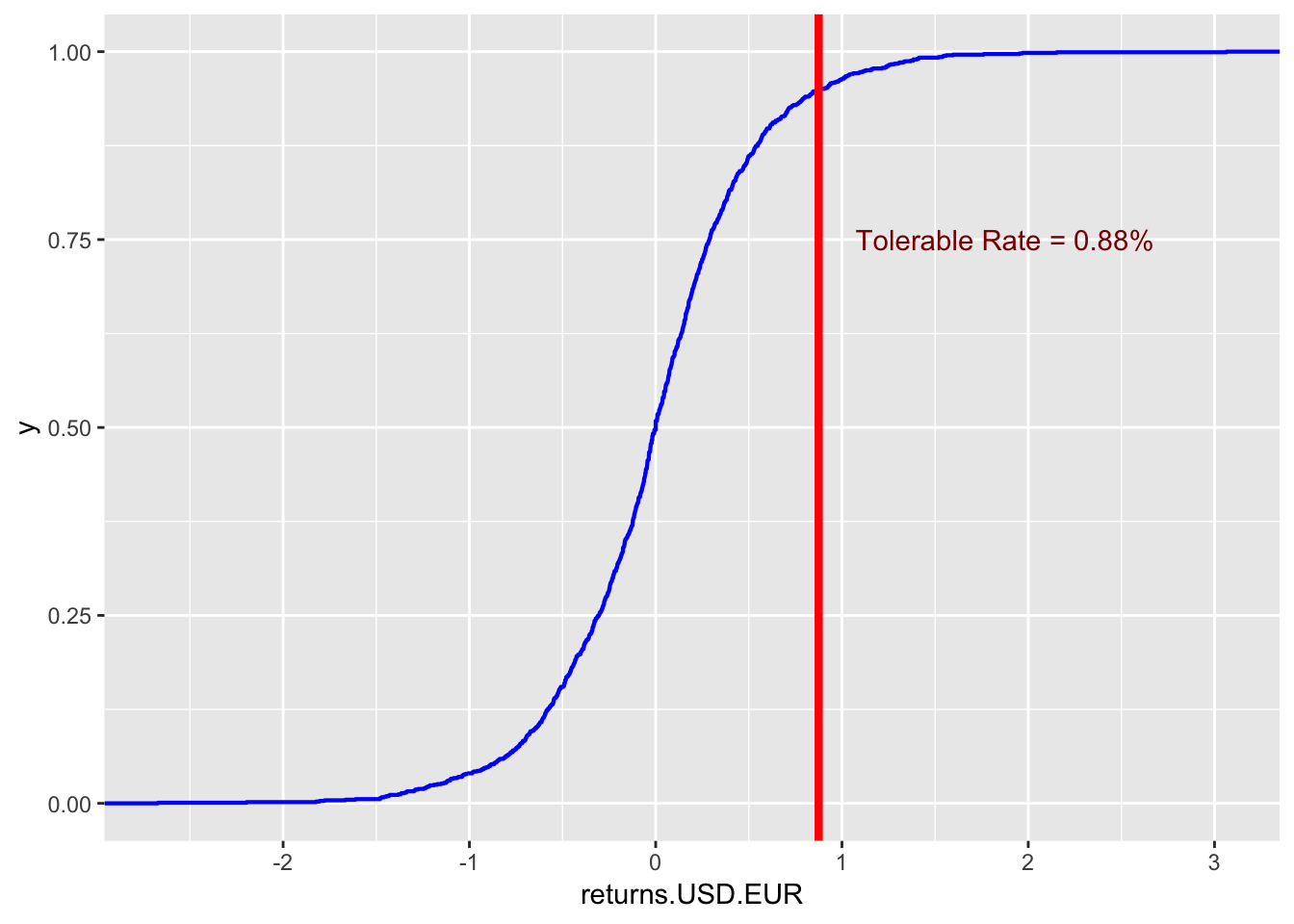
We want to characterize the distribution of up and down movements visually. Also we would like to repeat the analysis periodically for inclusion in management reports.

## Question 1: Distribution of returns

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.

To better understand the daily behavior of exchange rates, creating an estimated cumulative density function to model the distribution of returns will allow us to see how likely it is for the daily change in the exchange rate to be below a specified value. In creating a investing plan that has a 5% risk allowance, we can use this estimated cumulative density function to see at what percentage U.S Dollar depreciation against the Euro is past our tolerance for risk. The following Block of R code will set a tolerable risk percentage to 0.95 and find out at what rate has the dollar depreciated outside of our risk tolerance.

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.dir)) +   
 stat\_ecdf(colour = "blue", size = 0.75) +   
 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")  
p



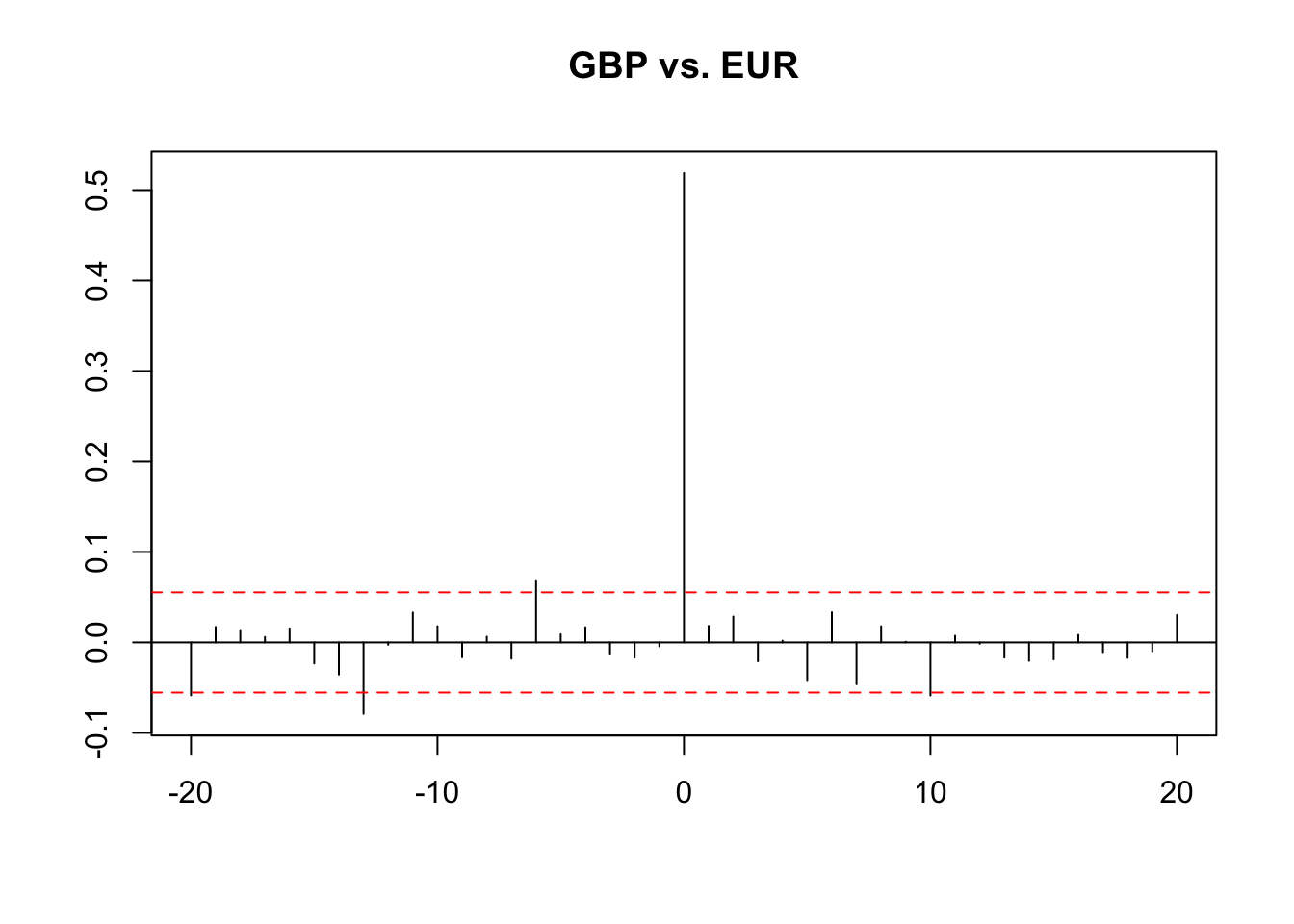
The estimated cumulative density function shows that on a given day there is a 95% percent chance that the percent change in the foreign exchange between the dollar and the euro will be below 0.88%. This also means that there is only a 5% chance that the dollar will depreciate against the pound by more than 0.88%. If the dollar has depreciated by more than 0.88% in a recent date, we should look to conduct further analysis of the exchange rate before buying the Euro.

The distribution of percent changes in the exchange between the dollar and the Euro also appears fairly symmetrical around the y-intercept, with approximately 50% of the time the dollar will appreciate and 50% of the time the dollar will appreciate.

## Question 2:

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!).

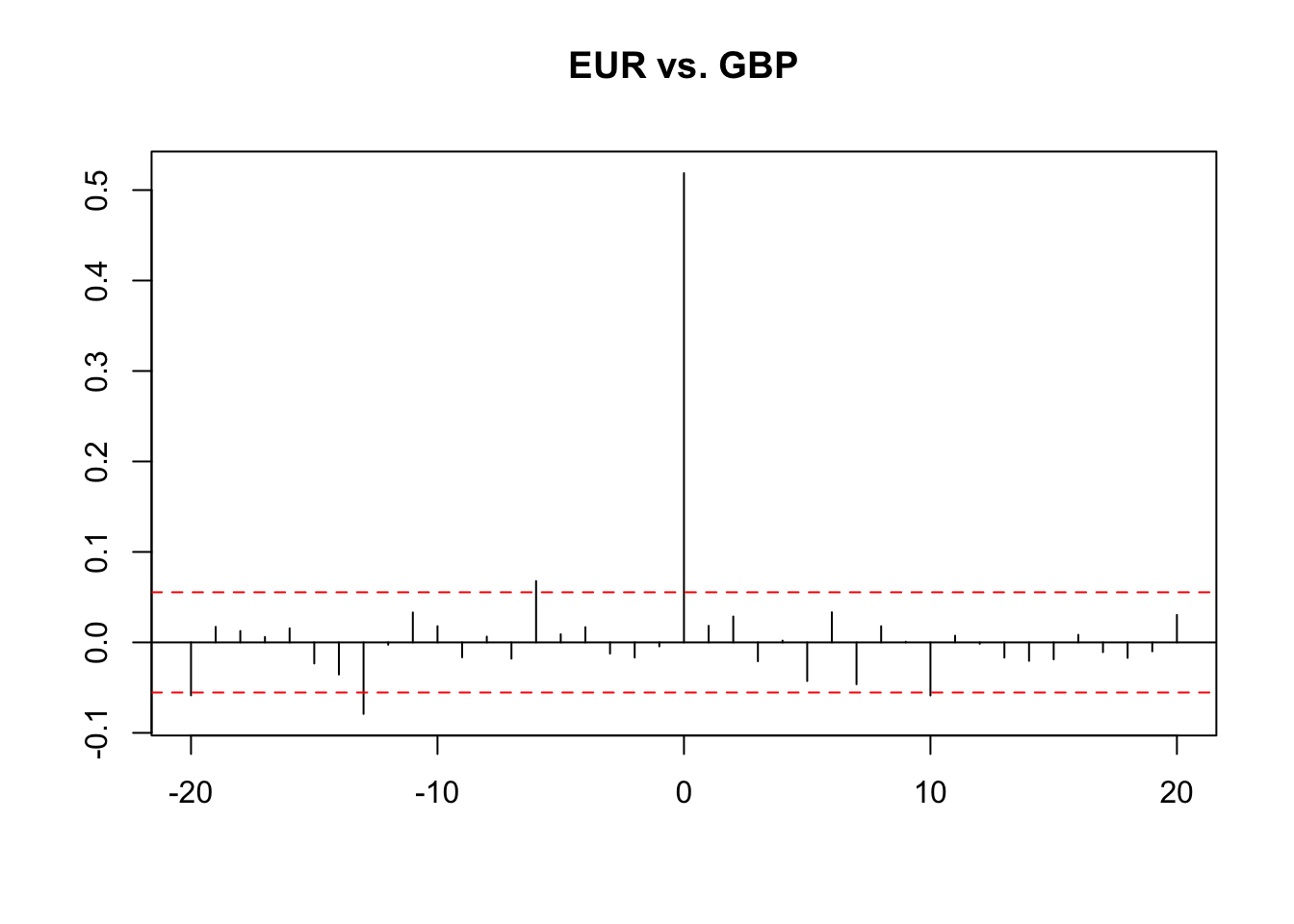
one <- ts(exrates.df$returns.USD.EUR)  
two <- ts(exrates.df$returns.USD.GBP)  
*# or*  
one <- ts(exrates.zr[, 1])  
two <- ts(exrates.zr[, 2])  
ccf(one, two, main = "GBP vs. EUR", lag.max = 20,   
 xlab = "", ylab = "", ci.col = "red")



**Sasha:**

There seems to be some small raw correlations across time with raw returns. Moreover, we see volatility of correlation clustering in using return sizes. This means that the same risk will occur in both countries.

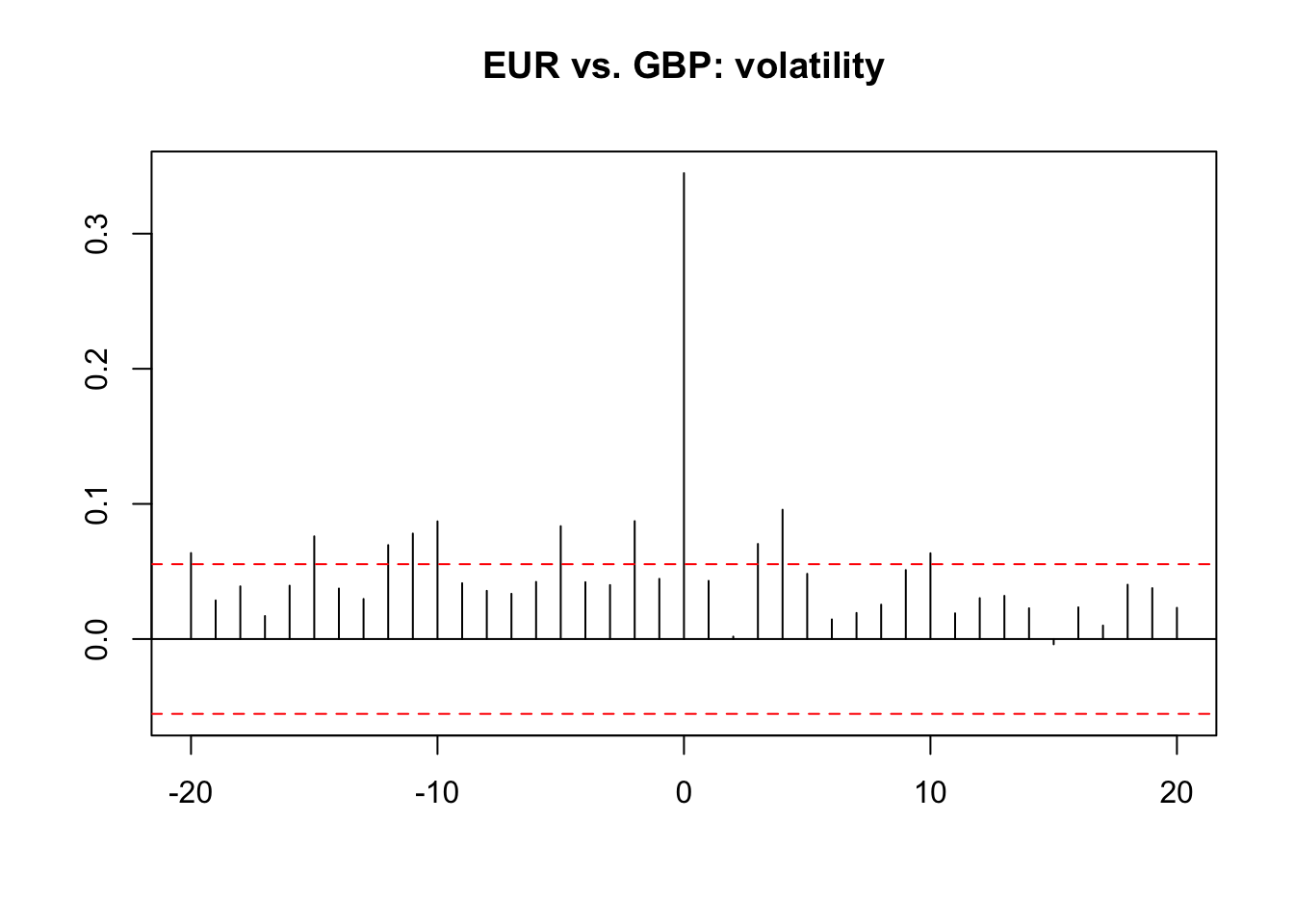
*# build function to repeat these*  
*# routines*  
run\_ccf <- **function**(one, two, main = "one vs. two",   
 lag = 20, color = "red") {  
 *# one and two are equal length series*  
 *# main is title lag is number of lags*  
 *# in cross-correlation color is color*  
 *# of dashed confidence interval*  
 *# 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(one, two, main = title, lag = 20,   
 color = "red")



**Sasha:**

There seems to be some small raw correlations across time with raw returns. Moreover, we see volatility of correlation clustering in using return sizes.

*# 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.*

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)

**Sasha:**

To further understand that bad things will be followed by more bad things more often than good things, we conducted a rolling correlation function; and in doing so, there seems to still be volatility clustering of the absolute sizes of returns.

## Question 4:

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)

## Loading required package: SparseM

##   
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':  
##   
## backsolve

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)

## Warning in log(CNY.JPY): NaNs produced

fit.lm.CNY.JPY <- lm(log(CNY.JPY) ~ log(JPY.vol),   
 data = r\_corr\_vol)

## Warning in log(CNY.JPY): NaNs produced

*# Some test statements*  
CNY.JPY.summary <- summary(fit.rq.CNY.JPY,   
 se = "boot")  
CNY.JPY.summary

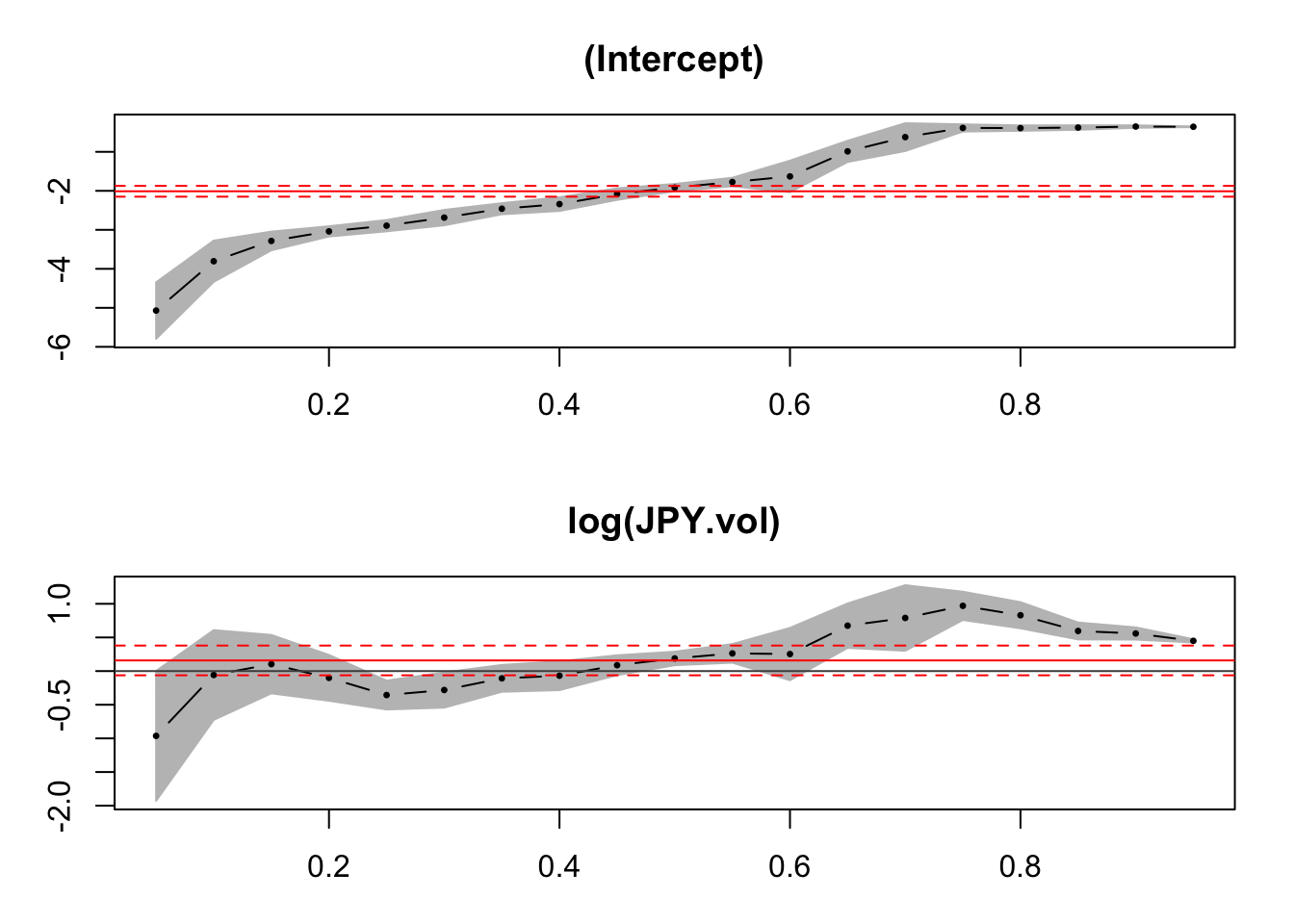
##   
## Call: rq(formula = log(CNY.JPY) ~ log(JPY.vol), tau = taus, data = r\_corr\_vol)  
##   
## tau: [1] 0.05  
##   
## Coefficients:  
## Value Std. Error t value Pr(>|t|)   
## (Intercept) -5.07014 0.44026 -11.51627 0.00000  
## log(JPY.vol) -0.96318 0.58629 -1.64284 0.10075  
##   
## Call: 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) -3.80541 0.32149 -11.83695 0.00000  
## log(JPY.vol) -0.05799 0.40536 -0.14306 0.88628  
##   
## 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) -3.28519 0.14570 -22.54789 0.00000  
## log(JPY.vol) 0.10332 0.26383 0.39163 0.69542  
##   
## 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) -3.03994 0.07989 -38.04943 0.00000  
## log(JPY.vol) -0.10061 0.20698 -0.48610 0.62701  
##   
## 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) -2.89330 0.08798 -32.88422 0.00000  
## log(JPY.vol) -0.35602 0.13038 -2.73064 0.00644  
##   
## 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) -2.68656 0.11992 -22.40310 0.00000  
## log(JPY.vol) -0.28125 0.15891 -1.76985 0.07708  
##   
## 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) -2.45886 0.08654 -28.41150 0.00000  
## log(JPY.vol) -0.10791 0.12119 -0.89040 0.37348  
##   
## 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) -2.34030 0.10674 -21.92459 0.00000  
## log(JPY.vol) -0.06848 0.13008 -0.52645 0.59870  
##   
## 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) -2.08348 0.09005 -23.13658 0.00000  
## log(JPY.vol) 0.08756 0.08941 0.97930 0.32768  
##   
## 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) -1.91646 0.05663 -33.84302 0.00000  
## log(JPY.vol) 0.18794 0.06103 3.07920 0.00214  
##   
## 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) -1.77384 0.06613 -26.82450 0.00000  
## log(JPY.vol) 0.26318 0.08392 3.13618 0.00177  
##   
## 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) -1.63052 0.24561 -6.63857 0.00000  
## log(JPY.vol) 0.25338 0.23610 1.07319 0.28346  
##   
## 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) -0.98780 0.16515 -5.98127 0.00000  
## log(JPY.vol) 0.67504 0.20063 3.36455 0.00080  
##   
## 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) -0.62254 0.21684 -2.87095 0.00418  
## log(JPY.vol) 0.78950 0.29601 2.66719 0.00778  
##   
## 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) -0.38695 0.05637 -6.86431 0.00000  
## log(JPY.vol) 0.96979 0.12782 7.58732 0.00000  
##   
## 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) -0.39138 0.04220 -9.27545 0.00000  
## log(JPY.vol) 0.83000 0.11808 7.02921 0.00000  
##   
## 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) -0.37793 0.03458 -10.93030 0.00000  
## log(JPY.vol) 0.59616 0.07641 7.80222 0.00000  
##   
## 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) -0.35178 0.01926 -18.26474 0.00000  
## log(JPY.vol) 0.55875 0.05522 10.11908 0.00000  
##   
## 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) -0.35746 0.00542 -65.95636 0.00000  
## log(JPY.vol) 0.44931 0.01451 30.95894 0.00000

**Sasha:**

Relationships between high-stress periods and correlation are abundant; normal buying behaviors are reflected in these markets.

plot(CNY.JPY.summary)

## Warning in log(CNY.JPY): NaNs produced



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 and scenarios.

Expand

Comment what we are doing below:

Will place my verbiage here tomorrow. -Matt

**library**(quantreg)  
**library**(magick)

## Linking to ImageMagick 6.9.9.39  
## Enabled features: cairo, fontconfig, freetype, lcms, pango, rsvg, webp  
## Disabled features: fftw, ghostscript, x11

img <- image\_graph(res = 96)  
datalist <- split(r\_corr\_vol, r\_corr\_vol$year)  
out <- lapply(datalist, **function**(data) {  
 p <- ggplot(data, aes(JPY.vol, CNY.JPY)) +   
 geom\_point() + ggtitle(data$year) +   
 geom\_quantile(quantiles = c(0.05,   
 0.95)) + geom\_quantile(quantiles = 0.5,   
 linetype = "longdash") + geom\_density\_2d(colour = "red")  
 print(p)  
})

## Warning: Removed 89 rows containing non-finite values (stat\_quantile).

## Smoothing formula not specified. Using: y ~ x

## Warning: Removed 89 rows containing non-finite values (stat\_quantile).

## Smoothing formula not specified. Using: y ~ x

## Warning: Removed 89 rows containing non-finite values (stat\_density2d).

## Warning: Removed 89 rows containing missing values (geom\_point).

## Smoothing formula not specified. Using: y ~ x

## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x  
## Smoothing formula not specified. Using: y ~ x

**while** (!is.null(dev.list())) dev.off()  
*# img <-*  
*# image\_background(image\_trim(img),*  
*# 'white')*  
animation <- image\_animate(img, fps = 0.5)  
animation