Project #4

Escapades in Market Risk

Purpose, Process, Product

Various `R` features and finance topics will be reprised in this chapter. 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.

Assignment

This assignment will span Weeks 6 & 7 (two weeks). The project (4) is due before Live Session 7. Submit into **Coursework > Assignments and Grading > Project 4 > Submission** an `RMD` file with filename **lastname-firstname__Project4.Rmd**. If you have difficulties submitting a `.Rmd` file, then submit a `.txt` file.

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. List skills and functions (e.g., `fit.GPD()`) 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.

5. Organize all the code & documentation in the flex_dashboard.

6. Knit to flex_dashboard - this will produce html output.

Assignment

We begin to expand our range of production capabilities. we started with an `R` script in project 1, then moved onto a `R Markdown` PDF file in projects 2 and 3. Now we introduce ourselves to the `flexdashboard` package with a little bit of `shiny` to get us used to building a web application directly.

1. Install the `flexdashboard` and `shiny` packages. In Rstudop console type 'library(c("flexdashboard", "shiny")) on the command line to attach these packages to the workspace.
2. Go to the RStudio `flexdashboard` site to learn about the basics of building a web application.
3. Go to `File` and click on `New File` where you will choose `R Markdown`. Because you already loaded `flexdashboard` this next step will work. In the dialog box choose `From Template`. You should see a `Flex Dashboard` template choice in the list box, which you will select and then click `OK`. A new `Rmd` file will then appear in the scripts pane of RStudio.
4. `Knit` this new file. A dialog box will appear for you to name the file.
5. Start to modify this template to document your own data analysis journey from here out. Each live session will contain a short segment on doing exactly that.
6. Optionally, use the ExtremeFinance application to test your installation and guide your work.

7. Join any Office hour prior to Live Session 8 to discuss installation & testing issues.

Problem

A freight forwarder with a fleet of bulk carriers wants to optimize their portfolio in the metals markets with entry into the nickel business and use of the tramp trade. Tramp ships are the company's "swing" option without any fixed charter or other constraint. They allow the company flexibility in managing several aspects of freight uncertainty. They have allocated $250 million to purchase metals. The company wants us to:

1. Retrieve and begin to analyze data about potential commodities to diversify into
2. Compare potential commodities with existing commodities in conventional metals spot markets
3. Begin to generate economic scenarios based on events that may, or may not, materialize in the commodities
4. The company wants to mitigate their risk by diversifying their cargo loads

Identify the optimal combination of Nickel, Copper, and Aluminium to trade

1. Product: Metals commodities and freight charters
2. Metal, Company, and Geography:
    a. Nickel: MMC Norilisk, Russia
    b. Copper: Codelco, Chile and MMC Norilisk, Russia
    c. Aluminium: Vale, Brasil and Rio Tinto Alcan, Australia
3. Customers: Ship Owners, manufacturers, traders
4. All metals traded on the London Metal Exchange

Key business questions

1. How would the performance of these commodities affect the size and timing of shipping arrangements?

2. How would the value of new shipping arrangements affect the value of our business with our current customers?

3. How would we manage the allocation of existing resources given we have just landed in this new market? ### Getting to a reponse: more detailed questions

4. What is the decision the freight-forwarder must make? List key business questions and data needed to help answer these questions and support the freight-forwarder's decision.

5. Develop the stylized facts of the markets the freight-forwarder faces. Include level, returns, size times series plots. Calculate and display in a table the summary statistics, including quantiles, of each of these series. Use autocorrelation, partial autocorrelation, and cross correlation functions to understand some of the persistence of returns including leverage and volatility clustering effects. Use quantile regressions to develop the distribution of sensitivity of each market to spill-over effects from other markets. Interpret these stylized "facts" in terms of the business decision the freight-forwarder makes.

6. How much capital would the freight-forwarder need? Determine various measures of risk in the tail of each metal's distribution. Then figure out a loss function to develop the portfolio of risk, and the determination of risk capital the freight-forwarder might need. Confidence intervals might be used to create a risk management plan with varying tail experience thresholds.

7. More importantly, begin to import your data into this model. You will have to modify some of the column subsets and all of the titles.

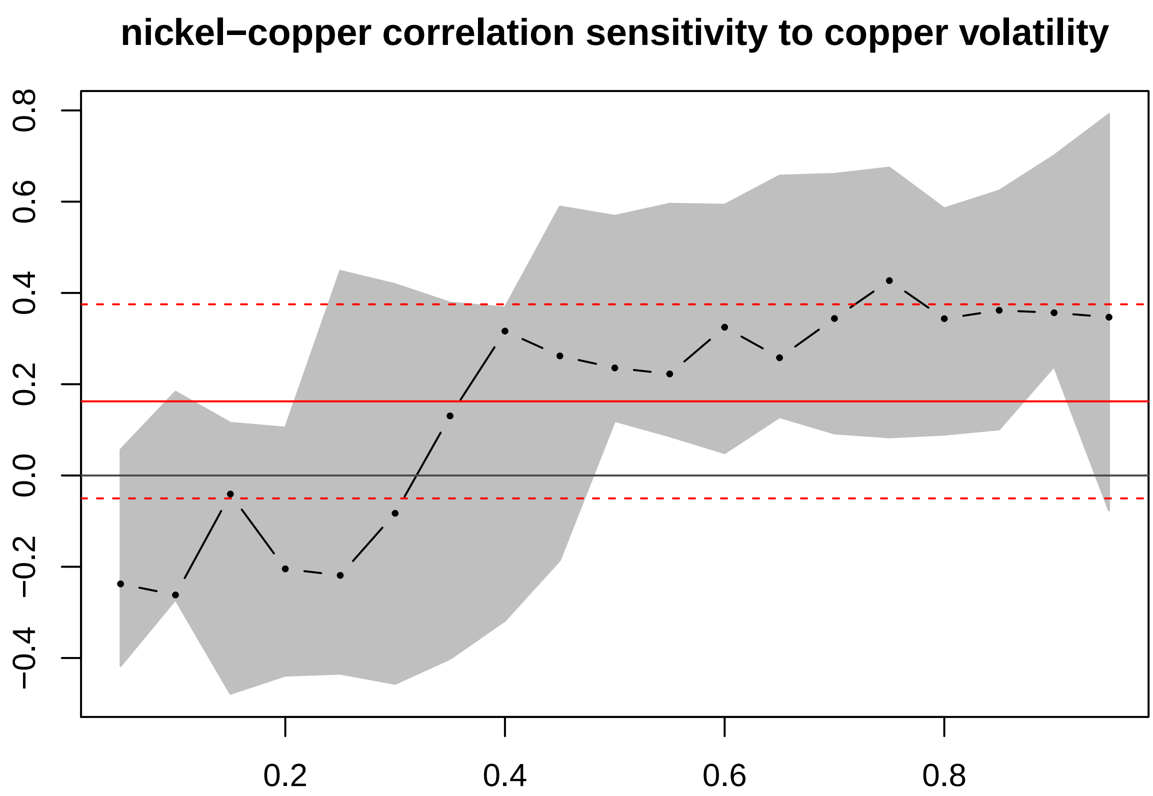
```r
require(ggplot2)
require(flexdashboard)
require(shiny)
require(QRM)
require(qrmdata)
require(xts)
require(zoo)
```

```
require(psych)

rm(list = ls())
# Exploratory Analysis
data <- na.omit(read.csv("data/metaldata.csv",
    header = TRUE))
# Compute log differences percent
# using as.matrix to force numeric
# type
data.r <- diff(log(as.matrix(data[, -1]))) *
    100
# Create size and direction
size <- na.omit(abs(data.r))  # size is indicator of volatility
# head(size)
colnames(size) <- paste(colnames(size),
    ".size", sep = "")  # Teetor
direction <- ifelse(data.r > 0, 1, ifelse(data.r <
    0, -1, 0))  # another indicator of volatility
colnames(direction) <- paste(colnames(direction),
    ".dir", sep = "")
# Convert into a time series object:
# 1. Split into date and rates
dates <- as.Date(data$DATE[-1], "%m/%d/%Y")
dates.chr <- as.character(data$DATE[-1])
str(dates.chr)
```

```
##  chr [1:1297] "3/15/2017" "3/14/2017" "3/13/2017" "3/10/2017" ...
```

```
values <- cbind(data.r, size, direction)
# for dplyr pivoting and ggplot2 need
# a data frame also known as 'tidy
# data'
data.df <- data.frame(dates = dates,
    returns = data.r, size = size, direction = direction)
data.df.nd <- data.frame(dates = dates.chr,
    returns = data.r, size = size, direction = direction,
    stringsAsFactors = FALSE)
# non-coerced dates for subsetting on
# non-date columns 2. Make an xts
# object with row names equal to the
# dates
data.xts <- na.omit(as.xts(values, dates))  #order.by=as.Date(dates, '%d/%m/%Y')))
# str(data.xts)
data.zr <- as.zooreg(data.xts)
returns <- data.xts

# Market analysis of the stylized
# facts and market risk preliminaries
corr.rolling <- function(x) {
    dim <- ncol(x)
    corr.r <- cor(x)[lower.tri(diag(dim),
        diag = FALSE)]
    return(corr.r)
}
```

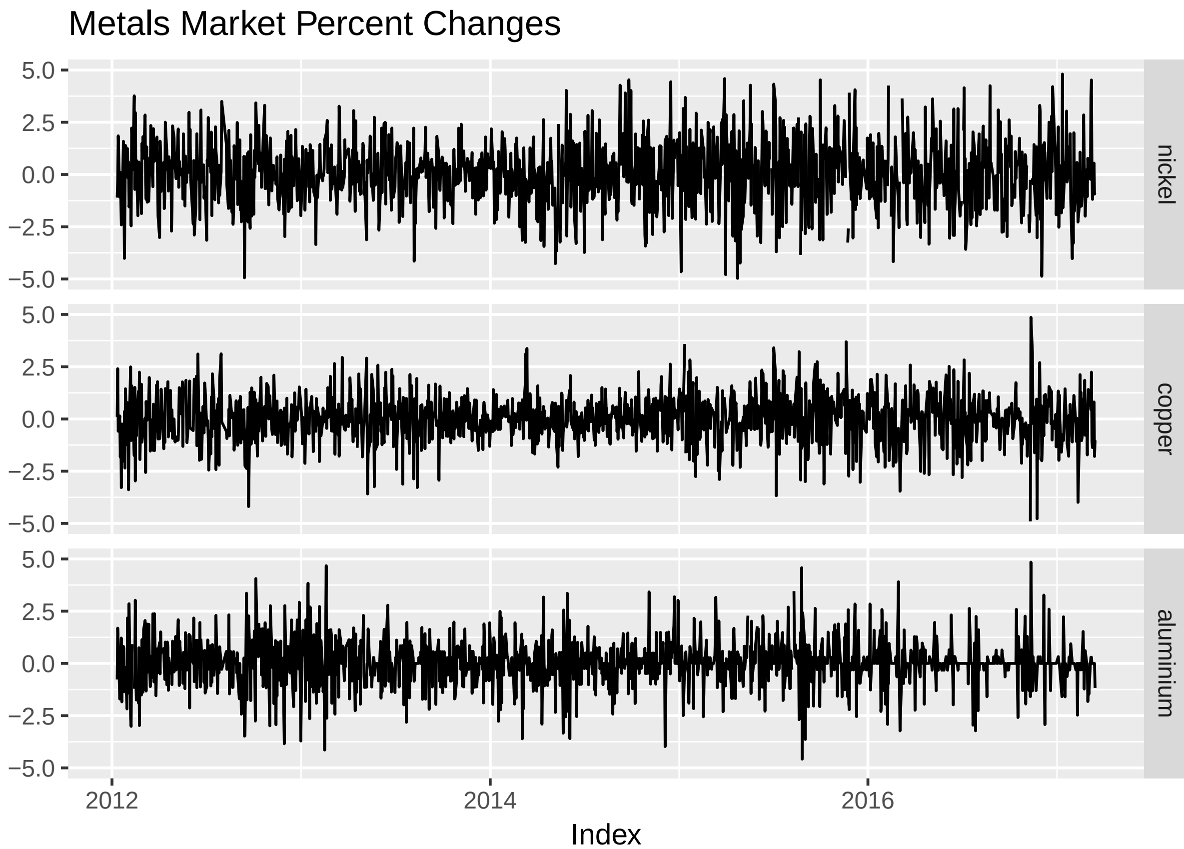
```r
ALL.r <- data.xts[, 1:3]  # Only three series here
window <- 90  #reactive({input$window})
corr.returns <- rollapply(ALL.r, width = window,
    corr.rolling, align = "right", by.column = FALSE)
# colnames(corr.returns) <- c('nickel
# & copper', 'nickel & aluminium',
# 'copper & aluminium')
corr.returns.df <- data.frame(Date = index(corr.returns),
    nickel.copper = corr.returns[, 1],
    nickel.aluminium = corr.returns[,
        2], copper.aluminium = corr.returns[,
        3])

# Market dependencies
require(matrixStats)
R.corr <- apply.monthly(as.xts(ALL.r),
    FUN = cor)
R.vols <- apply.monthly(ALL.r, FUN = colSds)  # from MatrixStats\t
# Form correlation matrix for one
# month
R.corr.1 <- matrix(R.corr[20, ], nrow = 3,
    ncol = 3, byrow = FALSE)
rownames(R.corr.1) <- colnames(ALL.r[,
    1:3])
colnames(R.corr.1) <- rownames(R.corr.1)
R.corr <- R.corr[, c(2, 3, 6)]
colnames(R.corr) <- colnames(corr.returns)
colnames(R.vols) <- c("nickel.vols",
    "copper.vols", "aluminium.vols")
R.corr.vols <- na.omit(merge(R.corr,
    R.vols))
nickel.vols <- as.numeric(R.corr.vols[,
    "nickel.vols"])
copper.vols <- as.numeric(R.corr.vols[,
    "copper.vols"])
aluminium.vols <- as.numeric(R.corr.vols[,
    "aluminium.vols"])
require(quantreg)
# hist(rho.fisher[, 1])
nickel.corrs <- R.corr.vols[, 1]
# hist(nickel.corrs)
taus <- seq(0.05, 0.95, 0.05)  # Roger Koenker UI Bob Hogg and Allen Craig
fit.rq.nickel.copper <- rq(nickel.corrs ~
    copper.vols, tau = taus)
# fit.lm.nickel.copper <-
# lm(nickel.corrs ~ copper.vols)
plot(summary(fit.rq.nickel.copper), parm = "copper.vols",
    main = "nickel-copper correlation sensitivity to copper volatility")
```



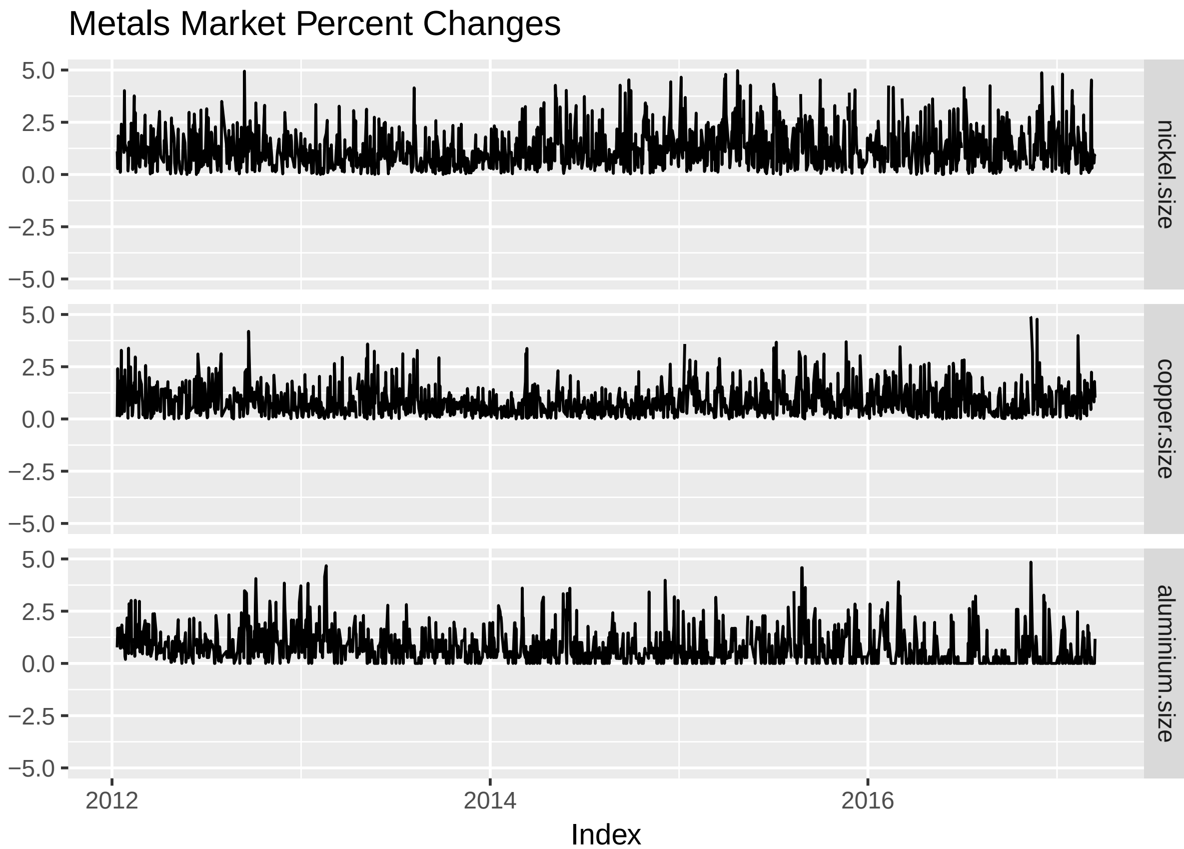

```
#' Some test statements\t
# summary(fit.rq.nickel.copper, se =
# 'boot')
#'
# summary(fit.lm.nickel.copper, se =
# 'boot')
# plot(summary(fit.rq.nickel.copper),
# parm = 'copper.vols', main =
# 'nickel-copper correlation
# sensitivity to copper volatility')
# #, ylim = c(-0.1 , 0.1))
```

** TRY the other combinations**

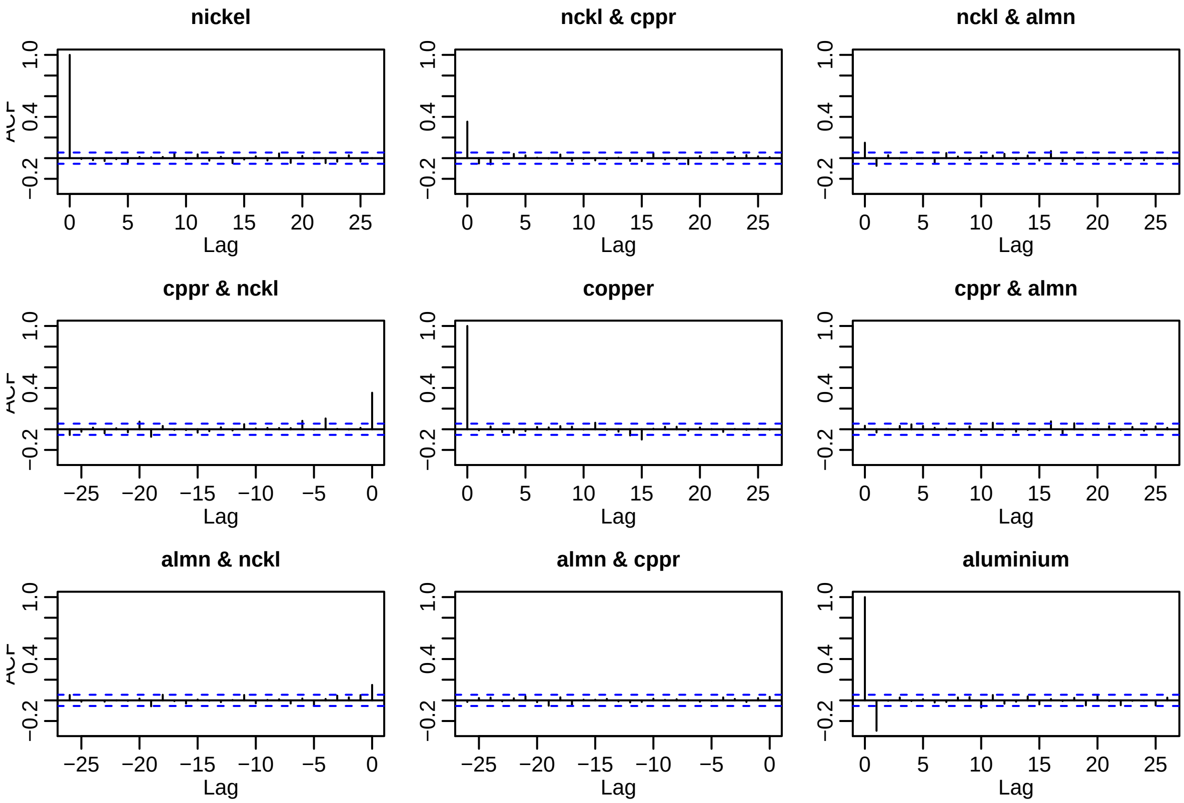
```
##
title.chg <- "Metals Market Percent Changes"
autoplot.zoo(data.xts[, 1:3]) + ggtitle(title.chg) +
    ylim(-5, 5)
```



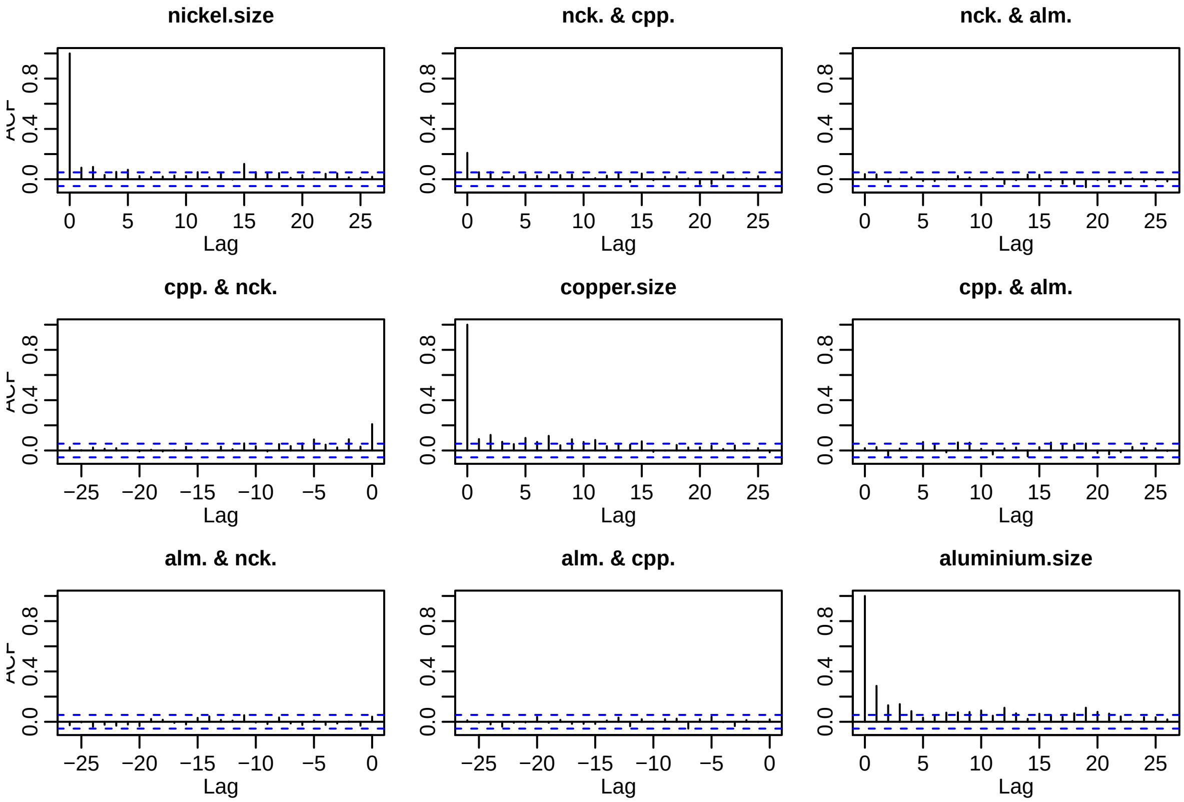

```
autoplot.zoo(data.xts[, 4:6]) + ggtitle(title.chg) +
    ylim(-5, 5)
```



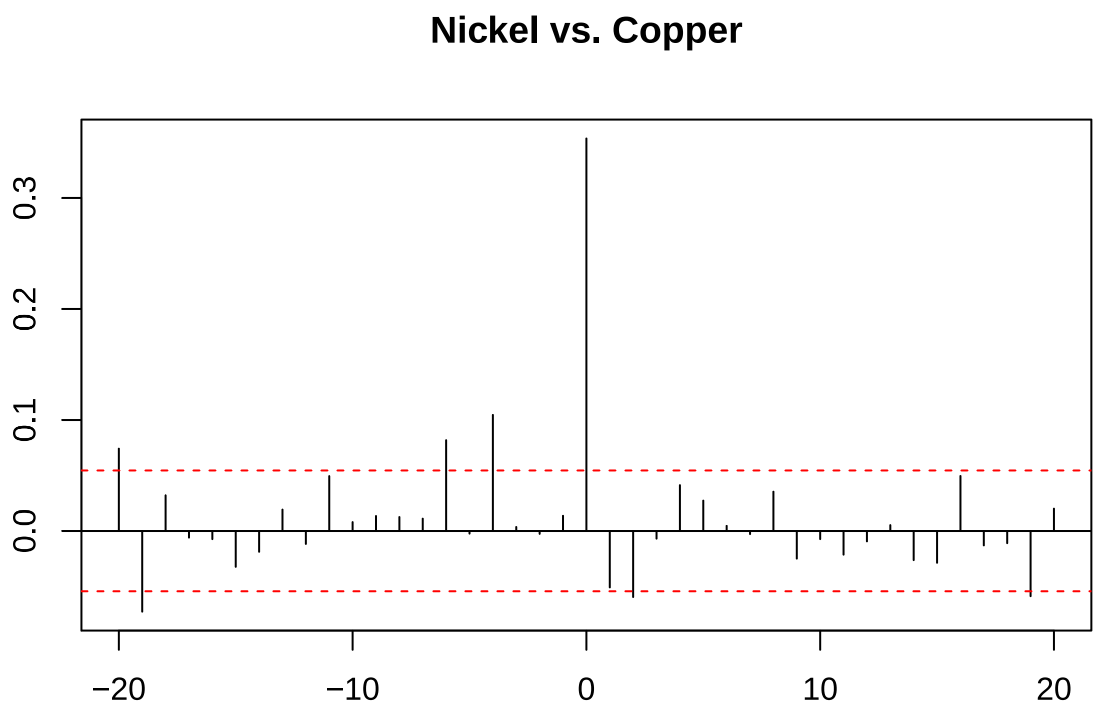

```
acf(coredata(data.xts[, 1:3]))  # returns
```



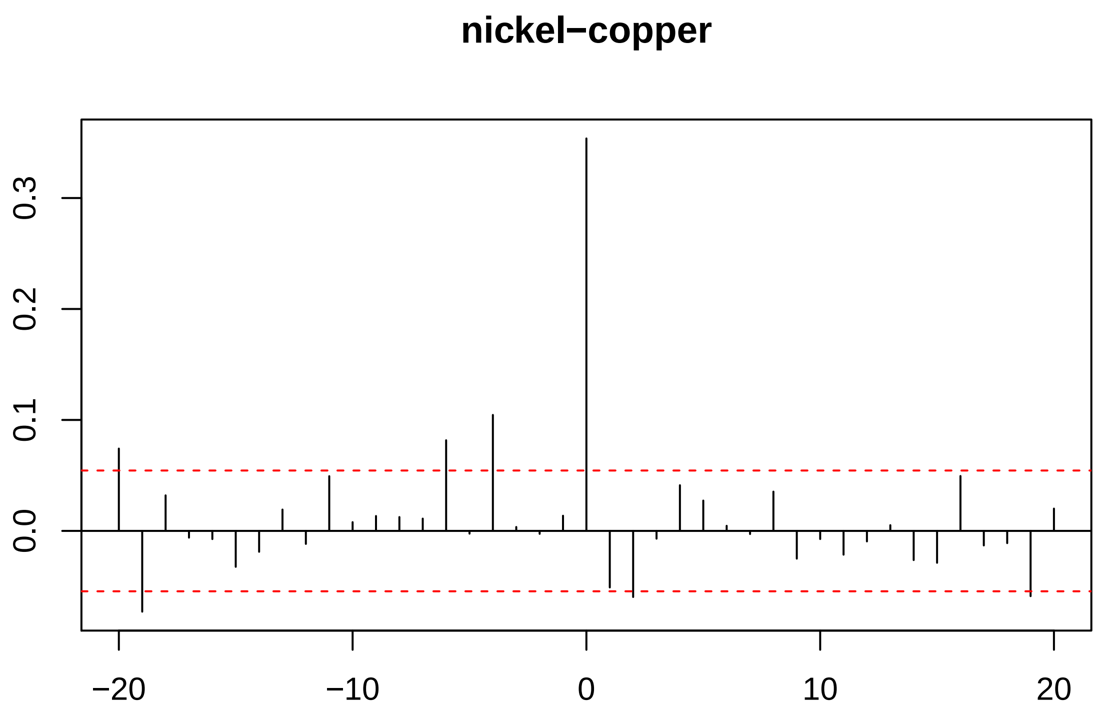

```
acf(coredata(data.xts[, 4:6]))  # sizes
```



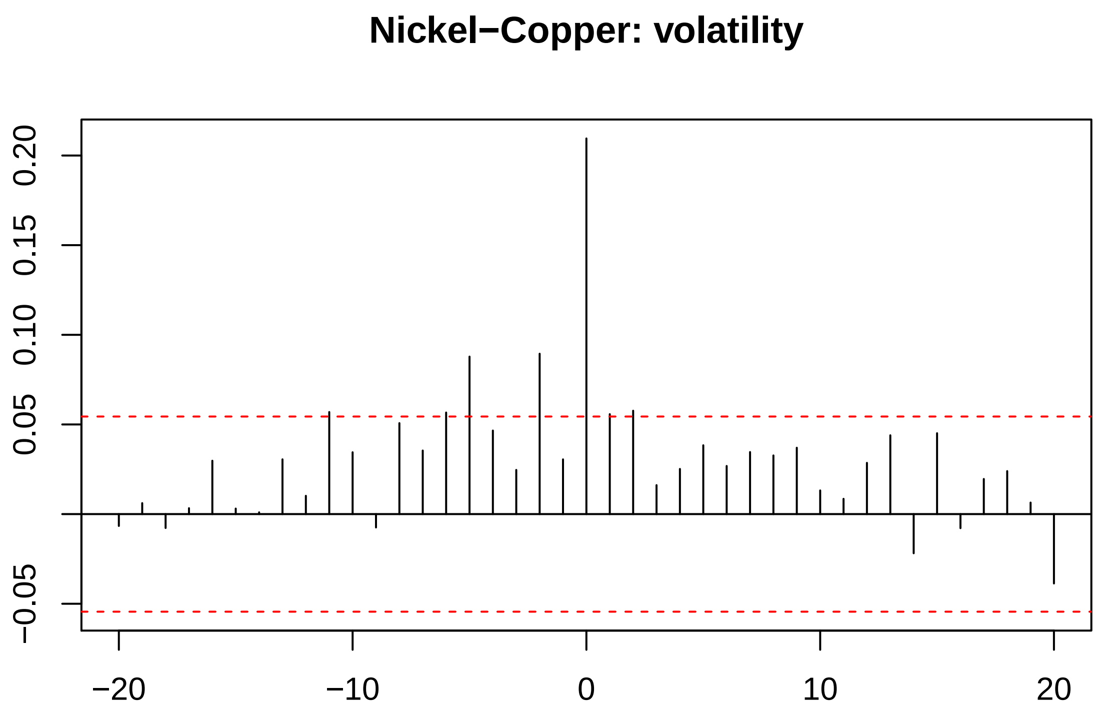

```r
# pacf here
one <- ts(data.df$returns.nickel)
two <- ts(data.df$returns.copper)
# or
one <- ts(data.zr[, 1])
two <- ts(data.zr[, 2])
title.chg <- "Nickel vs. Copper"
ccf(one, two, main = title.chg, lag.max = 20,
    xlab = "", ylab = "", ci.col = "red")
```




```
# build function to repeat these
# routines
run_ccf <- function(one, two, main = title.chg,
    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
}
title <- "nickel-copper"
run_ccf(one, two, main = title, lag = 20,
    color = "red")
```




```r
# now for volatility (sizes)
one <- abs(data.zr[, 1])
two <- abs(data.zr[, 2])
title <- "Nickel-Copper: volatility"
run_ccf(one, two, main = title, lag = 20,
    color = "red")
```



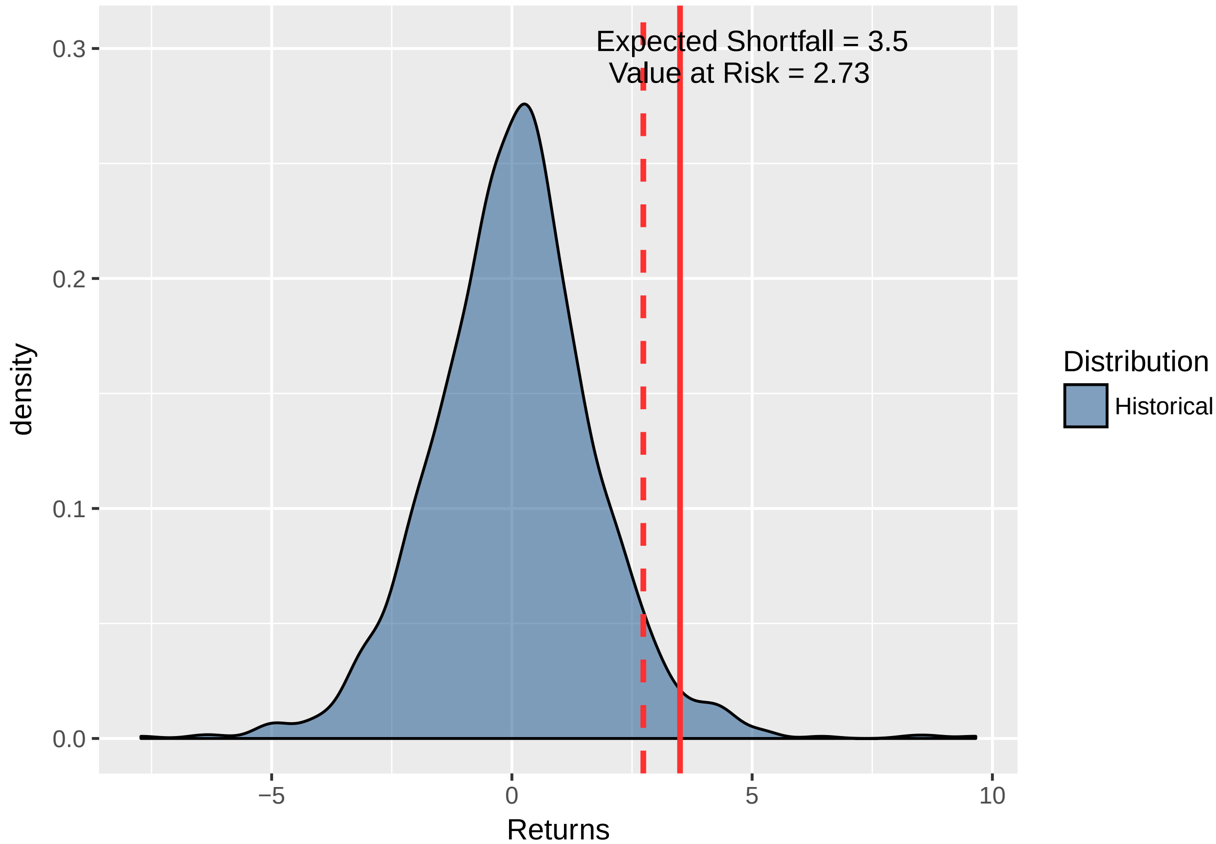

```r
## Load the data_moments() function
## data_moments function INPUTS: r
## vector OUTPUTS: list of scalars
## (mean, sd, median, skewness,
## kurtosis)
data_moments <- function(data) {
    require(moments)
    require(matrixStats)
    mean.r <- colMeans(data)
    median.r <- colMedians(data)
    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(data.xts[, 5:8])
# Build pretty table
answer <- round(answer, 4)
knitr::kable(answer)
```

	mean	median	std_dev	IQR	skewness	kurtosis
copper.size	0.8830	0.6823	0.7849	0.9217	1.7713	7.8654
aluminium.size	0.8072	0.5510	0.8986	1.0136	1.8899	8.4006
nickel.dir	0.0447	1.0000	0.9959	2.0000	-0.0895	1.0150
copper.dir	0.0540	1.0000	0.9931	2.0000	-0.1081	1.0235

```
mean(data.xts[, 4])
```

```
## [1] 1.282969
```

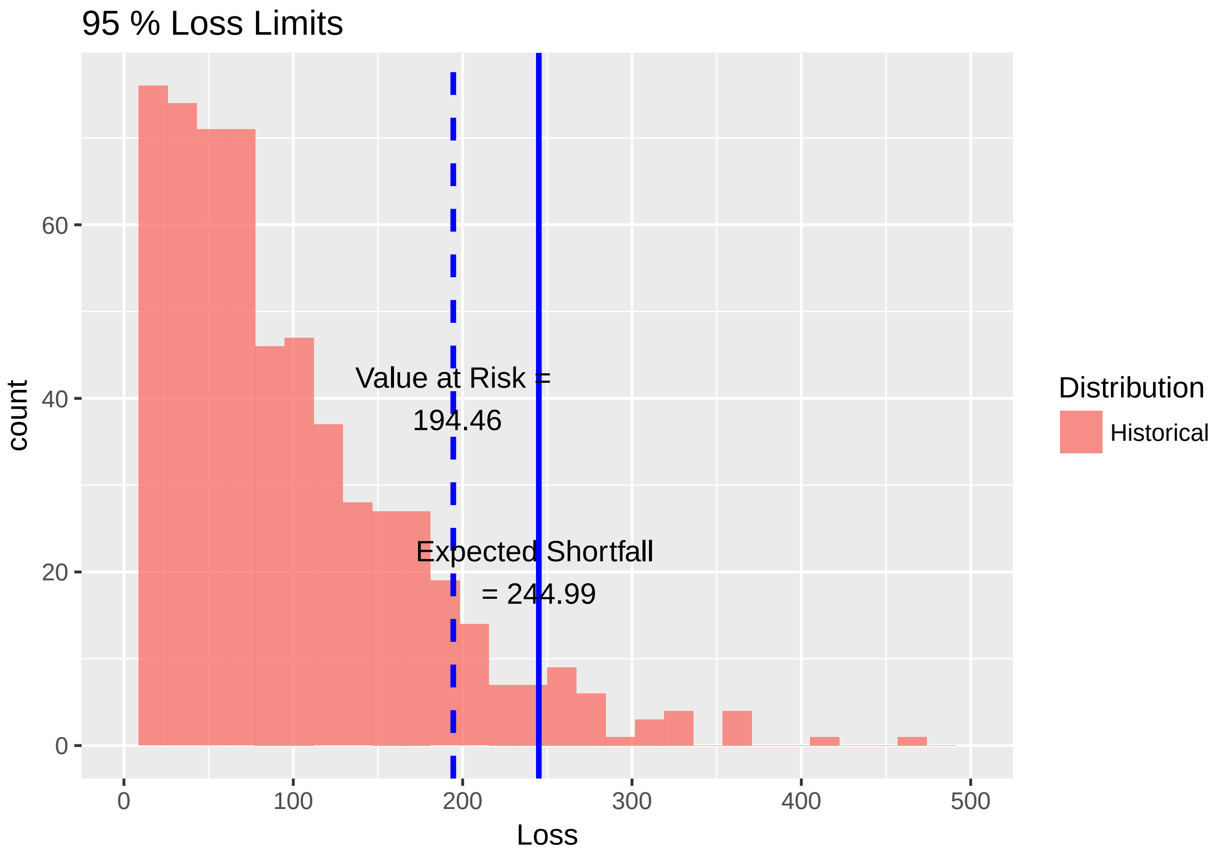
```
##
returns1 <- returns[, 1]
colnames(returns1) <- "Returns"  #kluge to coerce column name for df
returns1.df <- data.frame(Returns = returns1[,
    1], Distribution = rep("Historical",
    each = length(returns1)))

alpha <- 0.95  # reactive({ifelse(input$alpha.q>1,0.99,ifelse(input$alpha.q<0,0.001,input$alpha.q))})

# Value at Risk
VaR.hist <- quantile(returns1, alpha)
VaR.text <- paste("Value at Risk =",
    round(VaR.hist, 2))

# Determine the max y value of the
# desity plot.  This will be used to
# place the text above the plot
VaR.y <- max(density(returns1.df$Returns)$y)

# Expected Shortfall
ES.hist <- median(returns1[returns1 >
    VaR.hist])
ES.text <- paste("Expected Shortfall =",
    round(ES.hist, 2))

p <- ggplot(returns1.df, aes(x = Returns,
    fill = Distribution)) + geom_density(alpha = 0.5) +
    geom_vline(aes(xintercept = VaR.hist),
        linetype = "dashed", size = 1,
        color = "firebrick1") + geom_vline(aes(xintercept = ES.hist),
    size = 1, color = "firebrick1") +
    annotate("text", x = 2 + VaR.hist,
        y = VaR.y * 1.05, label = VaR.text) +
    annotate("text", x = 1.5 + ES.hist,
        y = VaR.y * 1.1, label = ES.text) +
    scale_fill_manual(values = "dodgerblue4")
p
```



** DO the same for returns 2 aand 3**

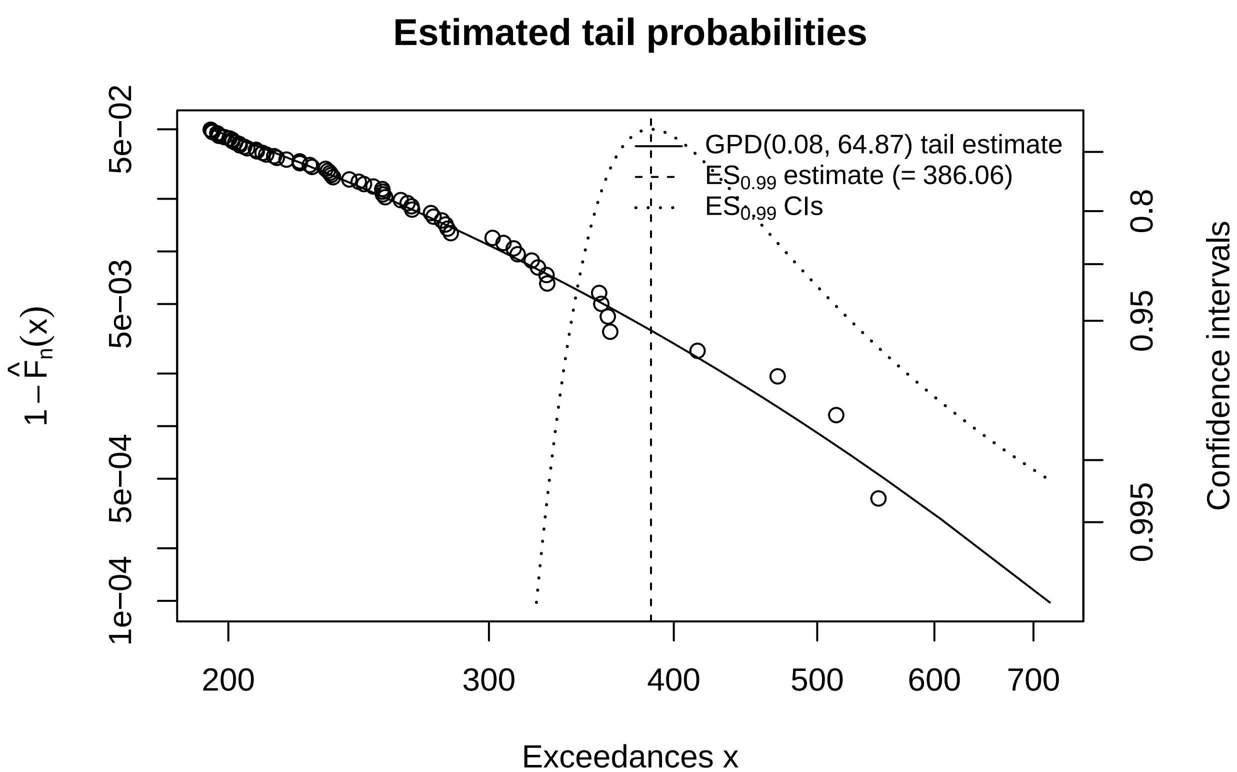
```
## Now for Loss Analysis Get last
## prices
price.last <- as.numeric(tail(data[,
    -1], n = 1))
# Specify the positions
position.rf <- c(1/3, 1/3, 1/3)
# And compute the position weights
w <- position.rf * price.last
# Fan these the length and breadth of
# the risk factor series
weights.rf <- matrix(w, nrow = nrow(data.r),
    ncol = ncol(data.r), byrow = TRUE)
# head(rowSums((exp(data.r/100)-1)*weights.rf),
# n=3) We need to compute exp(x) - 1
# for very small x: expm1
# accomplishes this
# head(rowSums((exp(data.r/100)-1)*weights.rf),
# n=4)
loss.rf <- -rowSums(expm1(data.r/100) *
    weights.rf)
loss.rf.df <- data.frame(Loss = loss.rf,
    Distribution = rep("Historical",
        each = length(loss.rf)))
## Simple Value at Risk and Expected
## Shortfall
```

```
alpha.tolerance <- 0.95
VaR.hist <- quantile(loss.rf, probs = alpha.tolerance,
    names = FALSE)
## Just as simple Expected shortfall
ES.hist <- median(loss.rf[loss.rf > VaR.hist])
VaR.text <- paste("Value at Risk =\n",
    round(VaR.hist, 2))  # ='VaR'&c12
ES.text <- paste("Expected Shortfall \n=",
    round(ES.hist, 2))
title.text <- paste(round(alpha.tolerance *
    100, 0), "% Loss Limits")
# using histogram bars instead of the
# smooth density
p <- ggplot(loss.rf.df, aes(x = Loss,
    fill = Distribution)) + geom_histogram(alpha = 0.8) +
    geom_vline(aes(xintercept = VaR.hist),
        linetype = "dashed", size = 1,
        color = "blue") + geom_vline(aes(xintercept = ES.hist),
    size = 1, color = "blue") + annotate("text",
    x = VaR.hist, y = 40, label = VaR.text) +
    annotate("text", x = ES.hist, y = 20,
        label = ES.text) + xlim(0, 500) +
    ggtitle(title.text)
p
```



```
# mean excess plot to determine
# thresholds for extreme event
# management
data <- as.vector(loss.rf)  # data is purely numeric
umin <- min(data)  # threshold u min
umax <- max(data) - 0.1  # threshold u max
nint <- 100  # grid length to generate mean excess plot
grid.0 <- numeric(nint)  # grid store
e <- grid.0  # store mean exceedances e
upper <- grid.0  # store upper confidence interval
lower <- grid.0  # store lower confidence interval
u <- seq(umin, umax, length = nint)  # threshold u grid
alpha <- 0.95  # confidence level
for (i in 1:nint) {
    data <- data[data > u[i]]  # subset data above thresholds
    e[i] <- mean(data - u[i])  # calculate mean excess of threshold
    sdev <- sqrt(var(data))  # standard deviation
    n <- length(data)  # sample size of subsetted data above thresholds
    upper[i] <- e[i] + (qnorm((1 + alpha)/2) *
        sdev)/sqrt(n)  # upper confidence interval
    lower[i] <- e[i] - (qnorm((1 + alpha)/2) *
        sdev)/sqrt(n)  # lower confidence interval
}
mep.df <- data.frame(threshold = u, threshold.exceedances = e,
    lower = lower, upper = upper)
loss.excess <- loss.rf[loss.rf > u]
# Voila the plot => you may need to
# tweak these limits!
p <- ggplot(mep.df, aes(x = threshold,
    y = threshold.exceedances)) + geom_line() +
    geom_line(aes(x = threshold, y = lower),
        colour = "red") + geom_line(aes(x = threshold,
    y = upper), colour = "red") + annotate("text",
    x = 400, y = 200, label = "upper 95%") +
    annotate("text", x = 200, y = 0,
        label = "lower 5%")
## GPD to describe and analyze the
## extremes require(QRM)
alpha.tolerance <- 0.95
u <- quantile(loss.rf, alpha.tolerance,
    names = FALSE)
fit <- fit.GPD(loss.rf, threshold = u)  # Fit GPD to the excesses
xi.hat <- fit$par.ests[["xi"]]  # fitted xi
beta.hat <- fit$par.ests[["beta"]]  # fitted beta
data <- loss.rf
n.relative.excess <- length(loss.excess)/length(loss.rf)  # = N_u/n
VaR.gpd <- u + (beta.hat/xi.hat) * (((1 -
    alpha.tolerance)/n.relative.excess)^(-xi.hat) -
    1)
ES.gpd <- (VaR.gpd + beta.hat - xi.hat *
    u)/(1 - xi.hat)
n.relative.excess <- length(loss.excess)/length(loss.rf)  # = N_u/n
VaR.gpd <- u + (beta.hat/xi.hat) * (((1 -
```

```
    alpha.tolerance)/n.relative.excess)^(-xi.hat) -
    1)
ES.gpd <- (VaR.gpd + beta.hat - xi.hat *
    u)/(1 - xi.hat)
# Plot away
VaRgpd.text <- paste("GPD: Value at Risk =",
    round(VaR.gpd, 2))
ESgpd.text <- paste("Expected Shortfall =",
    round(ES.gpd, 2))
title.text <- paste(VaRgpd.text, ESgpd.text,
    sep = " ")
loss.plot <- ggplot(loss.rf.df, aes(x = Loss,
    fill = Distribution)) + geom_density(alpha = 0.2)
loss.plot <- loss.plot + geom_vline(aes(xintercept = VaR.gpd),
    colour = "blue", linetype = "dashed",
    size = 0.8)
loss.plot <- loss.plot + geom_vline(aes(xintercept = ES.gpd),
    colour = "blue", size = 0.8)
#+ annotate('text', x = 300, y = 0.0075, label = VaRgpd.text, colour = 'blue') + annotate('text', x = 3
loss.plot <- loss.plot + xlim(0, 500) +
    ggtitle(title.text)
# Confidence in GPD
showRM(fit, alpha = 0.99, RM = "ES",
    method = "BFGS")
showRM(fit, alpha = 0.99, RM = "ES",
    method = "BFGS")  ##
```



```
# Generate overlay of historical and
# GPD; could also use Gaussian or t
# as well from the asynchronous
# material
```

** Conclusion (organize within flex_dashboard)**

Skills used

: (INSERT text here) :

Data Insights

: (INSERT text here) :

Business Summary

: (INSERT text here) :