Uploading the libraries

knitr::opts\_chunk$set(echo = TRUE)  
library(mosaic)

## Loading required package: dplyr

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

## Loading required package: lattice

## Loading required package: ggformula

## Loading required package: ggplot2

## Loading required package: ggstance

##   
## Attaching package: 'ggstance'

## The following objects are masked from 'package:ggplot2':  
##   
## geom\_errorbarh, GeomErrorbarh

##   
## New to ggformula? Try the tutorials:   
## learnr::run\_tutorial("introduction", package = "ggformula")  
## learnr::run\_tutorial("refining", package = "ggformula")

## Loading required package: mosaicData

## Loading required package: Matrix

## Registered S3 method overwritten by 'mosaic':  
## method from   
## fortify.SpatialPolygonsDataFrame ggplot2

##   
## The 'mosaic' package masks several functions from core packages in order to add   
## additional features. The original behavior of these functions should not be affected by this.  
##   
## Note: If you use the Matrix package, be sure to load it BEFORE loading mosaic.

##   
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':  
##   
## mean

## The following object is masked from 'package:ggplot2':  
##   
## stat

## The following objects are masked from 'package:dplyr':  
##   
## count, do, tally

## The following objects are masked from 'package:stats':  
##   
## binom.test, cor, cor.test, cov, fivenum, IQR, median,  
## prop.test, quantile, sd, t.test, var

## The following objects are masked from 'package:base':  
##   
## max, mean, min, prod, range, sample, sum

library(quantmod)

## Loading required package: xts

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

##   
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':  
##   
## first, last

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## Version 0.4-0 included new data defaults. See ?getSymbols.

library(foreach)

Importing the stocks that we want to use

port1 <- c("EDEN", "GXC", "CXSE", "QEMM", "IEMG")  
  
port2 <- c("FSZ", "JPMV", "FCA", "TUR")  
  
port3 <- c("EWJ", "EWL", "EWN", "ASHR", "KFYP", "GREK", "ERUS")  
  
port1.data <- getSymbols(port1, from = "2014-01-01")

## 'getSymbols' currently uses auto.assign=TRUE by default, but will  
## use auto.assign=FALSE in 0.5-0. You will still be able to use  
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")  
## and getOption("getSymbols.auto.assign") will still be checked for  
## alternate defaults.  
##   
## This message is shown once per session and may be disabled by setting   
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.

port2.data <- getSymbols(port2, from = "2014-01-01")  
port3.data <- getSymbols(port3, from = "2014-01-01")

## pausing 1 second between requests for more than 5 symbols

## pausing 1 second between requests for more than 5 symbols  
## pausing 1 second between requests for more than 5 symbols

Adjusting for splits and dividends

EDENa <- adjustOHLC(EDEN)  
GXCa <- adjustOHLC(GXC)  
CXSEa <- adjustOHLC(CXSE)  
QEMMa <- adjustOHLC(QEMM)  
IEMGa <- adjustOHLC(IEMG)  
  
FSZb <- adjustOHLC(FSZ)  
JPMVb <- adjustOHLC(JPMV)  
FCAb <- adjustOHLC(FCA)  
TURb <- adjustOHLC(TUR)  
  
EWJc <- adjustOHLC(EWJ)  
EWLc <- adjustOHLC(EWL)  
EWNc <- adjustOHLC(EWN)  
ASHRc <- adjustOHLC(ASHR)  
KFYPc <- adjustOHLC(KFYP)  
GREKc <- adjustOHLC(GREK)  
ERUSc <- adjustOHLC(ERUS)

Combining close to close changes in a single matrix

all\_returns\_a <- cbind(ClCl(EDENa),ClCl(GXCa), ClCl(CXSEa),ClCl(QEMMa),ClCl(IEMGa))  
all\_returns\_b <- cbind(ClCl(FSZb),ClCl(JPMVb), ClCl(FCAb),ClCl(TURb))  
all\_returns\_c <- cbind(ClCl(EWJc),ClCl(EWLc), ClCl(EWNc),ClCl(ASHRc),ClCl(KFYPc),ClCl(GREKc),ClCl(ERUSc))  
  
all\_returns\_a = as.matrix(na.omit(all\_returns\_a))  
all\_returns\_b = as.matrix(na.omit(all\_returns\_b))  
all\_returns\_c = as.matrix(na.omit(all\_returns\_c))  
  
print("Portfolio a: ")

## [1] "Portfolio a: "

head(all\_returns\_a)

## ClCl.EDENa ClCl.GXCa ClCl.CXSEa ClCl.QEMMa  
## 2014-06-06 -0.0022484355 0.0002689567 -0.0005862615 0.009933741  
## 2014-06-09 0.0015023850 0.0084677014 0.0031286665 -0.002131164  
## 2014-06-10 0.0007499906 0.0058643477 0.0077973101 0.000000000  
## 2014-06-11 -0.0106801576 -0.0051676030 -0.0069632687 0.000000000  
## 2014-06-12 0.0077651517 -0.0006659963 -0.0001947409 0.000000000  
## 2014-06-13 -0.0052621501 0.0106624418 0.0126631205 0.018728422  
## ClCl.IEMGa  
## 2014-06-06 0.0098951880  
## 2014-06-09 0.0032661288  
## 2014-06-10 0.0045959019  
## 2014-06-11 -0.0032405262  
## 2014-06-12 -0.0036336392  
## 2014-06-13 0.0001919962

print("")

## [1] ""

print("Portfolio b: ")

## [1] "Portfolio b: "

head(all\_returns\_b)

## ClCl.FSZb ClCl.JPMVb ClCl.FCAb ClCl.TURb  
## 2014-06-06 -0.0004515918 0.004694875 -0.0045641716 0.011635866  
## 2014-06-09 0.0038400497 -0.007009365 0.0000000000 0.001860639  
## 2014-06-10 -0.0045003826 -0.005098000 0.0050436041 0.013000169  
## 2014-06-11 -0.0099458178 0.002759145 -0.0018248631 -0.051999983  
## 2014-06-12 0.0002283562 0.007665075 0.0009141225 0.004043600  
## 2014-06-13 -0.0034238986 -0.001755413 0.0000000000 -0.006303642

print("")

## [1] ""

print("Portfolio c: ")

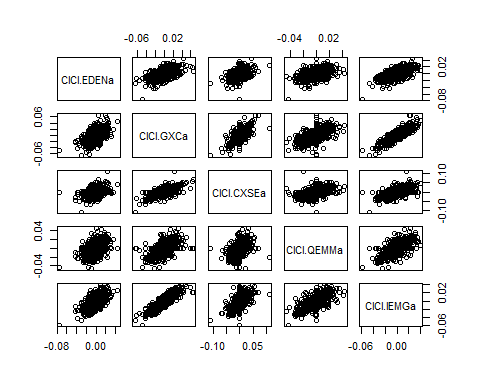
## [1] "Portfolio c: "

head(all\_returns\_c)

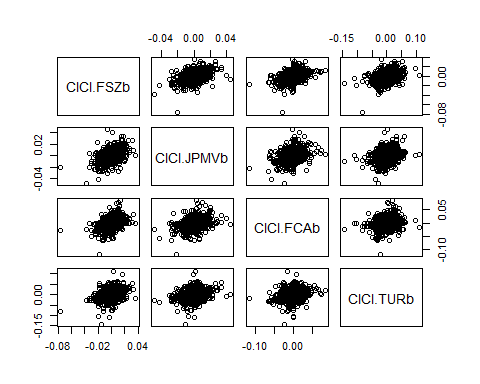
## ClCl.EWJc ClCl.EWLc ClCl.EWNc ClCl.ASHRc  
## 2014-01-03 0.005862710 0.0099348032 -0.0015673981 -0.0045454957  
## 2014-01-06 -0.003330558 -0.0003073778 -0.0007849686 -0.0249066002  
## 2014-01-07 0.004177130 0.0009224785 0.0062844464 0.0021286079  
## 2014-01-08 0.001663852 -0.0012289094 0.0003902420 0.0004247239  
## 2014-01-09 -0.003322259 0.0055367890 0.0011705424 -0.0148619115  
## 2014-01-10 0.006666667 0.0104007345 0.0074045207 0.0073275859  
## ClCl.KFYPc ClCl.GREKc ClCl.ERUSc  
## 2014-01-03 -0.010716442 -0.010118786 -0.005673712  
## 2014-01-06 -0.001547601 0.004888933 -0.019495934  
## 2014-01-07 0.002014972 0.026536841 0.004364597  
## 2014-01-08 0.014385150 0.026712668 -0.002897127  
## 2014-01-09 0.000000000 0.005874906 -0.005326804  
## 2014-01-10 0.000000000 0.014184398 0.018500437

Computing the returns from the closing prices

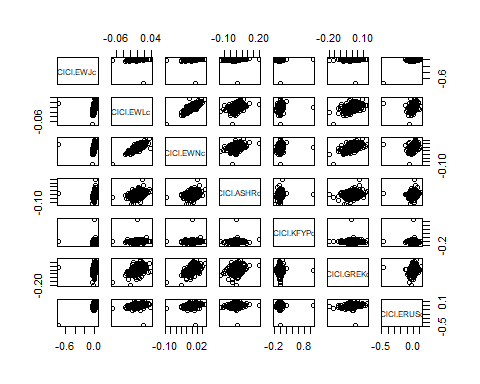
pairs(all\_returns\_a)



pairs(all\_returns\_b)



pairs(all\_returns\_c)

 Sampling a random return from the empirical joint distribution

set.seed(45)  
return.today\_a <- resample(all\_returns\_a, 1, orig.ids = FALSE)  
return.today\_b <- resample(all\_returns\_b, 1, orig.ids = FALSE)  
return.today\_c <- resample(all\_returns\_c, 1, orig.ids = FALSE)

Update the value of my holdings, starting with an equal distribution to each asset

set.seed(45)  
total\_wealth = 100000  
my\_weights\_a <- c(0.2, 0.2, 0.2, 0.2, 0.2)  
my\_weights\_b <- c(0.25, 0.25, 0.25, 0.25)  
my\_weights\_c <- c(0.14, 0.14, 0.15, 0.14, 0.14, 0.14, 0.15)  
  
holdings\_a <- total\_wealth\*my\_weights\_a  
holdings\_b <- total\_wealth\*my\_weights\_b  
holdings\_c <- total\_wealth\*my\_weights\_c  
  
holdings\_a <- holdings\_a\*(1+return.today\_a)  
holdings\_b <- holdings\_b\*(1+return.today\_b)  
holdings\_c <- holdings\_c\*(1+return.today\_c)  
  
total\_wealth\_a <- sum(holdings\_a)  
total\_wealth\_b <- sum(holdings\_b)  
total\_weatlh\_c <- sum(holdings\_c)  
  
total\_wealth\_a

## [1] 100952.9

total\_wealth\_b

## [1] 98416.81

total\_weatlh\_c

## [1] 100240.9

Loop over 4 trading weeks

set.seed(45)  
total\_wealth = 100000  
my\_weights\_a <- c(0.2, 0.2, 0.2, 0.2, 0.2)  
my\_weights\_b <- c(0.25, 0.25, 0.25, 0.25)  
my\_weights\_c <- c(0.14, 0.14, 0.15, 0.14, 0.14, 0.14, 0.15)  
  
holdings\_a <- total\_wealth\*my\_weights\_a  
holdings\_b <- total\_wealth\*my\_weights\_b  
holdings\_c <- total\_wealth\*my\_weights\_c  
  
n\_days = 20  
wealthtracker\_a = rep(0, n\_days)  
wealthtracker\_b = rep(0, n\_days)  
wealthtracker\_c = rep(0, n\_days)  
for(today in 1:n\_days){  
 return.today\_a <- resample(all\_returns\_a, 1, orig.ids = FALSE)  
 return.today\_b <- resample(all\_returns\_b, 1, orig.ids = FALSE)  
 return.today\_c <- resample(all\_returns\_c, 1, orig.ids = FALSE)   
   
 holdings\_a <- holdings\_a\*(1+return.today\_a)  
 holdings\_b <- holdings\_b\*(1+return.today\_b)  
 holdings\_c <- holdings\_c\*(1+return.today\_c)  
   
 total\_wealth\_a <- sum(holdings\_a)  
 total\_wealth\_b <- sum(holdings\_b)  
 total\_wealth\_c <- sum(holdings\_c)  
   
 wealthtracker\_a[today] = total\_wealth\_a  
 wealthtracker\_b[today] = total\_wealth\_b  
 wealthtracker\_c[today] = total\_wealth\_c  
   
}  
  
total\_wealth\_a

## [1] 100104.7

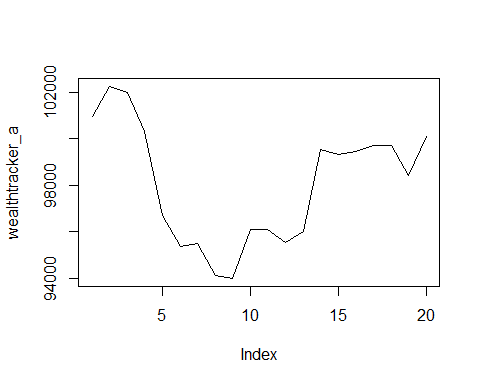
total\_wealth\_b

## [1] 102691

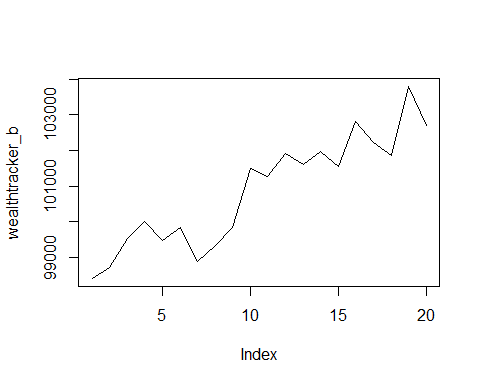
total\_wealth\_c

## [1] 94916.98

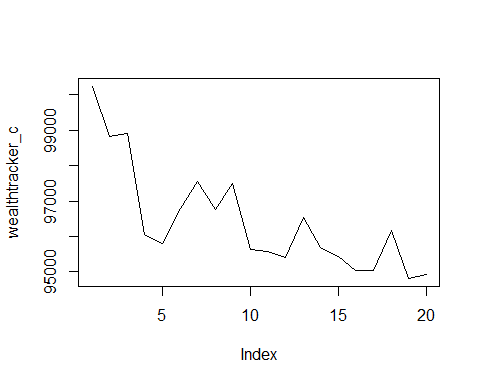
plot(wealthtracker\_a, type = 'l')



plot(wealthtracker\_b, type = 'l')



plot(wealthtracker\_c, type = 'l')

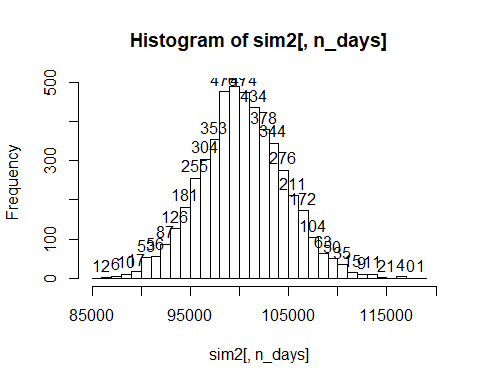


Second Portfolio

knitr::opts\_chunk$set(echo = TRUE)  
initial\_wealth = 100000  
sim2 = foreach(i = 1:5000, .combine = "rbind")%do%{  
 total\_wealth\_b = initial\_wealth  
   
 my\_weights\_b <- c(0.25, 0.25, 0.25, 0.25)  
   
 holdings\_b <- my\_weights\_b\*total\_wealth\_b  
   
 n\_days = 20  
   
 wealthtracker\_b = rep(0, n\_days)  
   
 for(today in 1:n\_days){  
 return.today\_b <- resample(all\_returns\_b, 1, orig.ids = FALSE)  
   
 holdings\_b <- holdings\_b\*(1+return.today\_b)  
   
 total\_wealth\_b <- sum(holdings\_b)  
   
 wealthtracker\_b[today] = total\_wealth\_b  
 }  
 wealthtracker\_b  
  
   
}  
head(sim2)

## [,1] [,2] [,3] [,4] [,5] [,6]  
## result.1 100069.09 99504.49 100239.77 101742.54 102306.67 102292.39  
## result.2 100212.53 100428.40 102200.01 104151.79 103875.69 103657.38  
## result.3 100853.32 101869.97 102674.87 102639.20 103702.01 105245.96  
## result.4 100871.72 101898.85 101842.91 102619.58 102834.31 102624.07  
## result.5 99857.27 97376.31 98003.09 97887.97 97793.52 97413.71  
## result.6 100527.75 99963.53 99971.77 101271.77 101545.28 101542.11  
## [,7] [,8] [,9] [,10] [,11] [,12]  
## result.1 102000.15 100125.32 99595.93 100079.51 99793.38 99690.28  
## result.2 102396.45 102949.80 103368.60 103144.50 103589.20 103389.24  
## result.3 105299.76 105808.16 106400.91 106658.02 107398.08 107251.09  
## result.4 103254.17 102544.38 103836.22 104190.83 104714.52 103194.47  
## result.5 97309.82 96628.53 97257.68 97443.53 96391.40 96322.86  
## result.6 101347.86 99398.50 99657.77 99041.67 99408.89 99741.64  
## [,13] [,14] [,15] [,16] [,17] [,18]  
## result.1 99520.97 100455.89 100249.48 100049.60 101991.5 103088.5  
## result.2 103184.53 104923.90 104668.27 104829.71 104458.0 104404.1  
## result.3 106588.64 106738.72 107097.80 107051.86 107031.5 107344.6  
## result.4 103500.55 103138.94 104110.34 104732.24 105984.9 106578.2  
## result.5 96334.62 99057.13 99248.74 98873.59 100508.0 100180.5  
## result.6 100194.32 100245.25 100638.97 102129.36 102503.9 102835.9  
## [,19] [,20]  
## result.1 103282.5 103049.8  
## result.2 105836.4 105456.5  
## result.3 105377.3 104223.6  
## result.4 105441.5 107499.8  
## result.5 102197.1 100493.1  
## result.6 102194.3 101777.8

hist(sim2[,n\_days], labels = TRUE,25)



VAR = quantile(sim2[,n\_days], .05)  
VAR

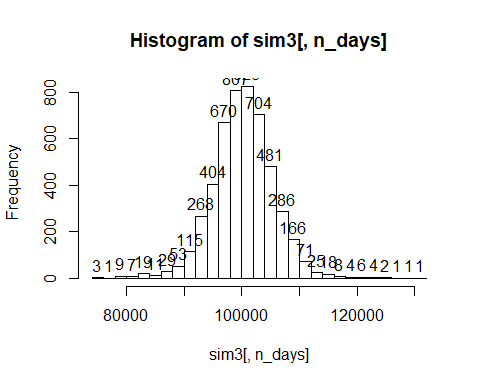
## 5%   
## 93234.85

Portfolio C

knitr::opts\_chunk$set(echo = TRUE)  
initial\_wealth = 100000  
sim3 = foreach(i = 1:5000, .combine = "rbind")%do%{  
 total\_wealth\_c = initial\_wealth  
   
 my\_weights\_c <- c(0.14, 0.14, 0.15, 0.14, 0.14, 0.14, 0.15)  
   
 holdings\_c <- my\_weights\_c\*total\_wealth\_c  
   
 n\_days = 20  
   
 wealthtracker\_c = rep(0, n\_days)  
   
 for(today in 1:n\_days){  
 return.today\_c <- resample(all\_returns\_c, 1, orig.ids = FALSE)  
   
 holdings\_c <- holdings\_c\*(1+return.today\_c)  
   
 total\_wealth\_c <- sum(holdings\_c)  
   
 wealthtracker\_c[today] = total\_wealth\_c  
 }  
 wealthtracker\_c  
  
   
}  
head(sim3)

## [,1] [,2] [,3] [,4] [,5] [,6]  
## result.1 99842.84 100729.01 101138.92 100325.41 101124.20 102330.98  
## result.2 101007.11 102304.86 104051.95 103549.75 102710.24 104394.14  
## result.3 100201.08 100019.15 100282.81 100239.82 99771.65 100446.96  
## result.4 99873.81 100154.12 100784.25 100679.43 101759.51 102122.56  
## result.5 99026.10 98309.49 98402.94 98606.19 98723.05 98719.79  
## result.6 99636.22 99993.29 100806.06 101094.28 100260.40 101122.38  
## [,7] [,8] [,9] [,10] [,11] [,12]  
## result.1 102570.79 102591.26 103943.39 99933.63 100376.19 102625.27  
## result.2 105053.62 103970.13 104533.71 104930.00 105824.21 105838.19  
## result.3 101159.16 102791.33 102825.78 103342.50 104319.07 103965.51  
## result.4 101531.89 99416.44 100128.82 99554.44 99296.60 98838.27  
## result.5 98190.96 95893.17 96722.86 98347.42 97685.76 96694.08  
## result.6 100309.67 99290.68 98837.29 99065.03 99438.31 99733.73  
## [,13] [,14] [,15] [,16] [,17] [,18]  
## result.1 99717.27 98401.74 97314.15 96578.42 96056.96 95911.96  
## result.2 105522.67 105864.06 105485.29 104840.33 105994.52 106543.87  
## result.3 104245.80 106125.72 105430.60 104846.94 106302.47 107860.42  
## result.4 97472.03 98428.83 97706.93 96901.01 95735.28 94260.28  
## result.5 97078.66 96098.15 95387.28 95535.45 95745.94 95841.02  
## result.6 100486.77 103565.86 103685.16 104106.71 105451.08 105620.23  
## [,19] [,20]  
## result.1 95723.14 93308.63  
## result.2 109226.27 106359.30  
## result.3 106852.67 105159.78  
## result.4 91934.46 89580.77  
## result.5 95689.65 96766.28  
## result.6 105146.26 104784.34

hist(sim3[,n\_days], labels = TRUE, 25)



VAR = quantile(sim3[,n\_days], .05)  
VAR

## 5%   
## 92038.78

I selected the portfolios with the intention of giving each one a different aim, despite all of the ETF’s being chosen from either the China ETF category, the Japan ETF category, Euro ETF category, or the Emerging Markets category. The first portfolio was chosen by being comprised of ETF’s which have the highest previous day’s closing cost. It contained 5 ETF’s (EDEN, GXC, CXSE, QEMM, and IEMG), with closing costs ranging from $47.99 to $87.82. The second portfolio contained 4 ETF’s (FSZ, JPMV, FCA, and TUR), and those ETF’s were chosen with the aim of minimizing the percent change from the previous day (in order to minimize short-term volatility). The third portfolio contained 7 ETF’s (EWJ, EWL, EWN, ASHR, KFYP, GREK, and ERUS), and were chosen with the intention of maximizing YTD, in order to maximize long-term growth. For the first portfolio the VaR was equal to 93091.06. That means that there is if there were no trading over the course of a day, there is a 5% chance that the stock would decrease in value by $93,091.06. For the 2nd portfolio, the VaR was equal to 93291.91, and the third portfolio had a VaR of 91864.87. These both translate to roughly the same thing as did portfolio 1. Looking at the histograms, portofolio 1 had a maximum value between 100000 and 102000, implying that at the very least, you would make back what you’d invested a majority of the time. Portfolio 2 had an absolute maximum value between 100000 and 101000, implying that you’d also make back your money by investing in that stock a majority of the time. Histogram 3 however had a majority of values between 98000 and 100000, implying that a majority of days, you would actually be losing money. This narrows us down to Portfolio 1 and Portfolio 3. Looking back at VaR, we see that portfolio 1 has a fatter tail than portfolio 3, therefore we choose Portfolio 3 for the purpose of our experiment.