Portfolio Testing

Brian Howard

March 11, 2017

Table of Contents

[Function Library 2](#_Toc508544260)

[Introduction 7](#_Toc508544261)

[Define the symbols to be used in the analysis. 7](#_Toc508544262)

[Build the data frame 7](#_Toc508544263)

[Interpolate data 7](#_Toc508544264)

[Add date column 8](#_Toc508544265)

[Normailize data 8](#_Toc508544266)

[Truncate data 8](#_Toc508544267)

[Calculate the returns 9](#_Toc508544268)

[Calculate the time based returns on annual basis 9](#_Toc508544269)

[Summarize returns 9](#_Toc508544270)

[Define the various portfolios 10](#_Toc508544271)

[Legacy USAA Portfolio 13](#_Toc508544272)

[Time series analysis 13](#_Toc508544273)

[Risk return analysis 14](#_Toc508544274)

[Time series analysis (Optimized) 15](#_Toc508544275)

[Correlation analysis 16](#_Toc508544276)

[New USAA Portfolio 17](#_Toc508544277)

[Time series analysis 17](#_Toc508544278)

[Risk return analysis 18](#_Toc508544279)

[Time series analysis (Optimized) 19](#_Toc508544280)

[Correlation analysis 20](#_Toc508544281)

[Legacy Roth Portfolio 21](#_Toc508544282)

[Time series analysis 21](#_Toc508544283)

[Risk return analysis 22](#_Toc508544284)

[Time series analysis (Optimized) 23](#_Toc508544285)

[Correlation analysis 24](#_Toc508544286)

[New Roth Portfolio 25](#_Toc508544287)

[Time series analysis 25](#_Toc508544288)

[Risk return analysis 26](#_Toc508544289)

[Time series analysis (Optimized) 27](#_Toc508544290)

[Correlation analysis 28](#_Toc508544291)

[Dividend Portfolio 29](#_Toc508544292)

[Time series analysis 29](#_Toc508544293)

[Risk return analysis 30](#_Toc508544294)

[Time series analysis (Optimized) 31](#_Toc508544295)

[Correlation analysis 32](#_Toc508544296)

[Appendix 33](#_Toc508544297)

[USSPX, USAA S&P 500 Index Member 33](#_Toc508544298)

[USSPX Year over year, USAA S&P 500 Index Member 34](#_Toc508544299)

[USISX, USAA Income Stock 35](#_Toc508544300)

[USISX Year over year, USAA Income Stock 36](#_Toc508544301)

[UFSGX, USAA First Start Growth 37](#_Toc508544302)

[UFSGX Returns, USAA First Start Growth 38](#_Toc508544303)

[USCCX, USAA Cornerstone Conservative 39](#_Toc508544304)

[USCCX Year over year, USAA Cornerstone Conservative 40](#_Toc508544305)

[VTWO, Vanguard Russell 2000 ETF 41](#_Toc508544306)

[IWM, iShares Russell 2000 42](#_Toc508544307)

[SPY, SPDR S&P 500 ETF Trust 43](#_Toc508544308)

[VFINX, Vanguard 500 Index Investor 44](#_Toc508544309)

[VYM, Vanguard High Dividend Yield ETF 46](#_Toc508544310)

# Function Library

This section contains the functions used in the code.

# ------------------------------------------------------------------------------  
# Define the generic single plane plot function  
plotSingle <- function(datadf, datax, datay, titlelabel, xlabel, ylabel, xlim, ylim, bLegend){  
   
 myplot <- ggplot()+  
 theme(plot.background = element\_rect(fill = 'white', colour = 'white')) +  
 theme(panel.background = element\_rect(fill = 'white', colour = 'grey')) +  
 theme(panel.grid.major.x = element\_blank()) +  
 theme(panel.grid.major.y = element\_line(colour="grey", size=0.5)) +  
 theme(panel.grid.minor.y = element\_line(colour="grey", size=0.25, linetype = "dotted")) +  
 geom\_line(data=datadf, aes\_string(x=datax, y=datay, colour = factor(datay)),   
 na.rm = TRUE, size = 1.0) +  
 scale\_colour\_manual(values = c("black","blue", "green")) +  
 guides(colour = guide\_legend("Series"), size = guide\_legend("Series"), shape = guide\_legend("Series")) +  
 scale\_fill\_continuous(name = "V") +  
 #geom\_smooth(method = "lm") +   
 ggtitle(titlelabel) +  
 labs(x=xlabel, y = ylabel) +  
 scale\_x\_date(limits = xlim ) +   
 scale\_y\_continuous(limits = ylim) +  
 if( bLegend){  
 theme(legend.position = "top")  
 }else{  
 theme(legend.position = "none")  
 }  
   
   
  
 return(myplot)  
}  
  
# ------------------------------------------------------------------------------  
# Define the function for calculating year over year growth.   
CalcYoY <- function (datadf, strCol, iPeriods){  
 Nrow <- nrow(datadf)  
 GrowthRateYoY <- rep(0,Nrow)  
 GrowthRateYoY[(iPeriods+1):Nrow] <- diff(as.matrix(datadf[[strCol]]), lag = iPeriods)  
 GrowthRateYoY <- (GrowthRateYoY / datadf[[strCol]])  
 return(GrowthRateYoY)  
}  
  
# Define return using log  
CalcLogRet <- function(datadf, strCol, iPeriods){  
 Nrow <- nrow(datadf)  
 GrowthRate <- rep(0,Nrow)  
 GrowthRate[(iPeriods+1):Nrow] <- diff(log(datadf[[strCol]]), lag = iPeriods)  
 return(GrowthRate)  
}  
  
  
# Small helper function to get the symbol description  
getPlotTitle <- function(datay){  
 strTitle <- paste(datay, " | ", dfSyms[grep(paste("^", datay, "$", sep=""), dfSyms$Symbol),]$Desc)  
 return(strTitle)  
}  
  
getPlotYLabel <- function(datay){  
 strY <- dfSyms[grep(paste("^", datay, "$", sep=""), dfSyms$Symbol),]$yLabel  
 return(strY)  
}  
  
plotSingleBench <- function(datay, ylim){  
   
 if( grep('.',datay) > 0 ){  
 strSym <- strSymOnly(datay)  
 strTitle <- paste(datay, " | ", dfSyms[grep(strSym, dfSyms$Symbol),]$Desc)  
 strYLabel <- dfSyms[grep(strSym, dfSyms$Symbol),]$yLabel  
 }else{  
 strTitle <- paste(datay, " | ", dfSyms[grep(datay, dfSyms$Symbol),]$Desc)  
 strYLabel <- dfSyms[grep(datay, dfSyms$Symbol),]$yLabel  
 }  
 dataBench <- "GSPC.Close\_Norm"  
 datax <- "date"  
 myPlot <- plotSingle(dfData, datax, datay,   
 strTitle, "Date",   
 strYLabel, c(as.Date(strDateStart), Sys.Date()), ylim, TRUE)  
 myPlot <- myPlot + geom\_line(data=dfData, aes\_string(x=datax, y=dataBench, colour = factor(dataBench)),   
 na.rm = TRUE, size = 0.7)  
   
   
 return(myPlot)  
   
   
}  
  
plotSingleQuick <- function(datay, ylim, strY){  
   
   
 # Create the y-axis label  
 if( missing(strY)){  
   
 # Handle those symbols with a period in their names  
 if( grep('.',datay) > 0 ){  
 strSym <- strSymOnly(datay)  
 strYLabel <- dfSyms[grep(strSym, dfSyms$Symbol),]$yLabel  
 }else{  
 strYLabel <- dfSyms[grep(datay, dfSyms$Symbol),]$yLabel  
 }  
 }else{  
 strYLabel <- strY  
 }  
   
 # Create the title for the plot  
 if( grep('.',datay) > 0 ){  
 strSym <- strSymOnly(datay)  
 strTitle <- paste(datay, " | ", dfSyms[grep(strSym, dfSyms$Symbol),]$Desc)  
 }else{  
 strTitle <- paste(datay, " | ", dfSyms[grep(datay, dfSyms$Symbol),]$Desc)  
 }  
 datax <- "date"  
 myPlot <- plotSingle(dfData, datax, datay,   
 strTitle, "Date",   
 strYLabel, c(as.Date(strDateStart ), Sys.Date()), ylim, TRUE)  
   
 return(myPlot)  
   
   
}  
  
plotRR <- function(datay, bIgnorePort){  
   
 if( missing(bIgnorePort)){  
 bIgnorePort = FALSE;  
 }  
 dfRR <- dfSyms[dfSyms[datay]>0,]  
   
 # Need a data table with just the ticker and data  
 strCols <- paste(dfRR$Symbol, ".Close\_Norm\_YoY", sep="")  
 dfPort <- data.table(dfData[,strCols])  
  
 # Range of expected returns from the porfolio  
 er\_vals <- seq(from = min(dfRR$ExpReturn), to = max(dfRR$ExpReturn), length.out = 1000)  
   
 # find an optimal portfolio for each possible possible expected return   
 # (note that the values are explicitly set between the minimum and maximum of the expected returns per asset)  
 sd\_vals <- rep(0,length(er\_vals))  
 tryCatch({  
 sd\_vals <- sapply(er\_vals, function(er) {  
 op <- portfolio.optim(as.matrix(dfPort), er)  
 return(op$ps)  
 })  
 }, error = function(e) {  
 print("Failed to find efficiency boundary")  
 })  
   
 # Collect in a table  
 plot\_dt <- data.table(sd = sd\_vals, er = er\_vals)  
   
 # find the lower and the upper frontier  
 minsd <- min(plot\_dt$sd)  
 minsd\_er <- plot\_dt[sd == minsd, er]  
 minsd\_er <- minsd\_er[1]  
 plot\_dt[, efficient := er >= minsd\_er]  
   
 # Data for the actual portfolio mix  
 dfPortRet <- dfData[datay]  
 strPfYoY <- paste(datay,"\_YoY",sep="")   
 dfPortRet[strPfYoY] <- CalcYoY(dfPortRet, datay, iRetPd)  
 Vol\_pf <- sd(dfPortRet[,strPfYoY])  
 ExpRet\_pf <- mean(dfPortRet[,strPfYoY])  
 #print(Vol\_pf)  
 #print(ExpRet\_pf)  
   
 # Data for the portfolio mix with same returns, but lower volatility  
 op\_pf <- portfolio.optim(as.matrix(dfPort), ExpRet\_pf)  
 dfRR[paste(datay,"\_Opt",sep="")] <- op\_pf$pw  
  
# Plot the data  
 myPlot <- ggplot() +   
 geom\_point(data=dfRR, aes(x = Volatility, y = ExpReturn, size=5, color = Symbol)) +  
 geom\_point(data = plot\_dt[efficient == F], aes(x = sd, y = er), size = 0.5, color = "blue") +  
 geom\_point(data = plot\_dt[efficient == T], aes(x = sd, y = er), size = 0.5, color = "red") +  
 theme\_bw() + ggtitle("Risk-Return Tradeoff (Red=Efficient), Annual Returns") +   
 xlab("Volatility") + ylab("Expected Returns") +  
 scale\_y\_continuous(label = scales::percent, limits = c(-0.05, 0.1)) +  
 scale\_x\_continuous(label = scales::percent, limits = c(-0.05, 0.25))  
  
 if( bIgnorePort==FALSE ){  
 myPlot <- myPlot + geom\_point(aes(x = Vol\_pf, y = ExpRet\_pf, size=5, color = "Actual Portfolio")) +geom\_point(aes(x = op\_pf$ps, y = ExpRet\_pf, size=5, color = "Reduced Risk Portfolio"))  
 }  
   
 return(list(dfRR, myPlot, plot\_dt))  
  
}  
  
plotCorr <- function(strPort){  
   
 lstActiveSyms <- paste(dfSyms[dfSyms[,strPort]>0,]$Symbol,'.Close\_Norm',sep="")  
  
  
 # Correlation for the entire data set  
 training.cor <- dfData[,lstActiveSyms]  
 rcorr.data <- rcorr(as.matrix(training.cor), type = "pearson")  
 #print(rcorr.data)  
  
 corrplot(cor(training.cor), type="upper", order="original",   
 tl.col="black", tl.srt=45, title ="All data")  
}

These functions help organize tables in the document

tblPort <- function(rrData, datay){  
   
 kable(rrData[[1]][c("Symbol", "Desc", datay, paste(datay, "\_Opt", sep = ""))])  
   
}

# Introduction

This is a portfolio analysis, data pulled from yahoo. I’m trying to decide what a better mix would be for this old IRA account. These next few segments load the data into the R program.

## Define the symbols to be used in the analysis.

This code pulls in the library stock ticker symbols that will be available for analysis.

## Build the data frame

This code loops through the dfSyms dataframe and pulls down the symbols. The getSymbols function returns zoo objects, this snippet converts it into a dataframe, which simplifies operations in R.

## Interpolate data

The data is interpolated to a daily basis. This is done because I also incorporate other data sampled at different intervals. When performing correlations, it is easier if all the data has the same time basis.

# Linearly interpolate the data, add normalized columns (this helps speed portfolio calcs)  
  
for (col\_name in names(dfData))  
{  
 if (is.numeric(dfData[,col\_name]))  
 {  
 # 15 Jun 2017: added this line because data from yahoo has one  
 # value that was zero. It really throws off the calcs so I  
 # replace it with NA, then interpolate it.  
 dfData[dfData == 0] <- NA  
 dfData[,col\_name] <- na.approx(dfData[,col\_name], rule=2)  
   
 }  
}

## Add date column

# This speeds some of the upcoming operations  
dfData$date = as.Date(rownames(dfData))

## Normailize data

# Linearly interpolate the data, add normalized columns (this helps speed portfolio calcs)  
  
for (col\_name in names(dfData))  
{  
 if (is.numeric(dfData[,col\_name]))  
 {  
 # Normalize data  
 strColNorm <- paste(col\_name, "\_Norm", sep="")   
 dfData[, strColNorm] <- dfData[,col\_name] / dfData[dfData$date==as.Date(strDateStart),col\_name]  
  
 }  
}

## Truncate data

A few last calculations to truncate and aggregrate the data

# Truncate by date, if needed  
dfData <- with(dfData, dfData[date>=strDateStart,])

# Calculate the returns

## Calculate the time based returns on annual basis

In this analysis all the returns are calculated on an annual basis. It means there is a year of dead data, but it eliminates seasonal variation in data (think: Christmas).

for (strName in names(dfData)){  
   
 if( is.numeric(dfData[, strName])){  
   
 strNameYoY <- paste(strName,"\_YoY",sep="")  
 dfData[strNameYoY] <- CalcYoY(dfData, strName, iRetPd)  
   
 strNameLog <- paste(strName, "\_LogRet", sep="")  
 dfData[strNameLog] <- CalcLogRet(dfData, strName, iRetPd)  
   
 strRootSym <- strName  
 strSuffix <- ""  
 if( grepl("\\.", strName)){  
 strRootSym <- substr(strName, 1, regexpr("\\.", strName)-1)  
 strSuffix <- substr(strName, regexpr("\\.", strName)+1, nchar(strName))  
 strSuffix <- paste(" (", strSuffix, ")", sep="")  
 }  
   
 strNewDesc <- dfSyms[grep(paste("^",strRootSym,"$",sep=""), dfSyms$Symbol),]$Desc  
 dfSyms <- rbind(dfSyms, data.frame(Symbol=strNameYoY, Source="Calc",   
 Desc=paste(strNewDesc, strSuffix, " Year over Year", sep=""), yLabel="Percent" ))  
   
 }  
  
}

## Summarize returns

In constructing the portfolio it is importance to understand the blend of funds. We want maximum returns for minimum volatility. That’s not possible in a perfect world so we plot the returns and volatility to find a nice balance in the portfolio.

# Linearly interpolate the data, add normalized columns (this helps speed portfolio calcs)  
  
for (col\_name in names(dfData))  
{  
 if (is.numeric(dfData[,col\_name]))  
 {  
  
 # Split the name ("USGFG.Close"" is "USGFG"" and "Close"")  
 lstSyms <- lstSymSplit(col\_name)  
   
 # Only if there is two terms  
 if( length(lstSyms) > 1){  
  
 if( lstSyms[2] == 'Close\_YoY' ){  
  
 # Return is the mean of the series  
 dfSyms[dfSyms$Symbol==lstSyms[1],'ExpReturn'] <- mean(dfData[, col\_name])  
  
 # Volatility is the standard deviation of the series.  
 dfSyms[dfSyms$Symbol==lstSyms[1],'Volatility'] <- sd(dfData[, col\_name])  
   
 }  
 }  
  
 }  
}

# Define the various portfolios

This code is evaluating two portfolios. “pf\_USAA\_Legacy” is the original portfolio and “pf\_USAA\_New” is the one I’m trying out.

dfSyms$pf\_USAA\_Legacy <- 0  
dfSyms[dfSyms$Symbol=='USSPX',]$pf\_USAA\_Legacy <- 0.4948  
dfSyms[dfSyms$Symbol=='USAWX',]$pf\_USAA\_Legacy <- 0.1429  
dfSyms[dfSyms$Symbol=='USISX',]$pf\_USAA\_Legacy <- 0.2383  
dfSyms[dfSyms$Symbol=='UFSGX',]$pf\_USAA\_Legacy <- 0.1240  
  
dfSyms$pf\_USAA\_New <- 0  
dfSyms[dfSyms$Symbol=='USPRX',]$pf\_USAA\_New <- 0.00000  
dfSyms[dfSyms$Symbol=='USSPX',]$pf\_USAA\_New <- 0.4500  
dfSyms[dfSyms$Symbol=='USAWX',]$pf\_USAA\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='USISX',]$pf\_USAA\_New <- 0.00000  
dfSyms[dfSyms$Symbol=='USNQX',]$pf\_USAA\_New <- 0.5500  
dfSyms[dfSyms$Symbol=='USCGX',]$pf\_USAA\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='UFSGX',]$pf\_USAA\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='USCCX',]$pf\_USAA\_New <- 0.0001  
dfSyms[dfSyms$Symbol=='USAAX',]$pf\_USAA\_New <- 0.0000  
  
dfSyms$pf\_Roth\_Legacy <- 0  
dfSyms[dfSyms$Symbol=='VTWO',]$pf\_Roth\_Legacy <- 0.0947  
dfSyms[dfSyms$Symbol=='VFINX',]$pf\_Roth\_Legacy <- 0.5051  
dfSyms[dfSyms$Symbol=='TMFGX',]$pf\_Roth\_Legacy <- 0.2568  
dfSyms[dfSyms$Symbol=='IWM',]$pf\_Roth\_Legacy <- 0.0001  
dfSyms[dfSyms$Symbol=='QQQ',]$pf\_Roth\_Legacy <- 0.0001  
dfSyms[dfSyms$Symbol=='HAINX',]$pf\_Roth\_Legacy <- 0.0001  
dfSyms[dfSyms$Symbol=='VEU',]$pf\_Roth\_Legacy <- 0.0001  
  
  
dfSyms$pf\_Roth\_New <- 0  
dfSyms[dfSyms$Symbol=='VTWO',]$pf\_Roth\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='VFINX',]$pf\_Roth\_New <- 0.3000  
dfSyms[dfSyms$Symbol=='TMFGX',]$pf\_Roth\_New <- 0.0500  
dfSyms[dfSyms$Symbol=='IWM',]$pf\_Roth\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='QQQ',]$pf\_Roth\_New <- 0.4000  
dfSyms[dfSyms$Symbol=='HAINX',]$pf\_Roth\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='CAMIX',]$pf\_Roth\_New <- 0.1000  
dfSyms[dfSyms$Symbol=='IVOO',]$pf\_Roth\_New <- 0.1500  
dfSyms[dfSyms$Symbol=='VEU',]$pf\_Roth\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='SPY',]$pf\_Roth\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='VIGI',]$pf\_Roth\_New <- 0.0000  
dfSyms[dfSyms$Symbol=='SCHD',]$pf\_Roth\_New <- 0.0000  
  
dfSyms$pf\_Div <- 0  
dfSyms[dfSyms$Symbol=='VFINX',]$pf\_Div <- 0.1000  
dfSyms[dfSyms$Symbol=='SCHD',]$pf\_Div <- 0.1000  
dfSyms[dfSyms$Symbol=='VYM',]$pf\_Div <- 0.1000  
dfSyms[dfSyms$Symbol=='NOBL',]$pf\_Div <- 0.1000  
  
  
# This part normalizes all the data, in case there is a typo or   
# numerical rounding that results in somthing less than 100%  
for (col\_name in names(dfSyms)){  
   
 if( length(grep('pf\_', col\_name))){  
 dfSyms[,col\_name] <- dfSyms[,col\_name] / sum(dfSyms[,col\_name])  
 }  
   
}  
  
# Update the returns  
dfData <- pfUpdateReturn("pf\_USAA\_Legacy", dfData, dfSyms)  
dfData <- pfUpdateReturn("pf\_USAA\_New", dfData, dfSyms)  
dfData <- pfUpdateReturn("pf\_Roth\_Legacy", dfData, dfSyms)  
dfData <- pfUpdateReturn("pf\_Roth\_New", dfData, dfSyms)  
dfData <- pfUpdateReturn("pf\_Div", dfData, dfSyms)  
  
kable(dfSyms[(dfSyms$pf\_USAA\_Legacy>0.0) | (dfSyms$pf\_USAA\_New>0.0)  
 | (dfSyms$pf\_Roth\_New>0.0)| (dfSyms$pf\_Div>0.0),])

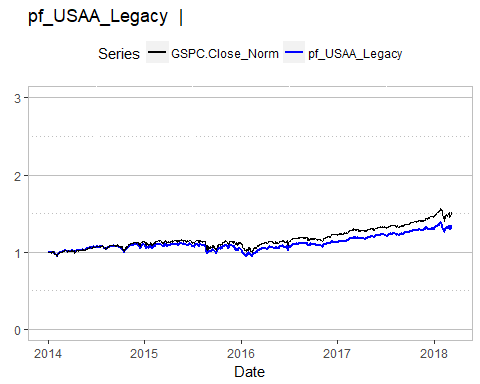
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Symbol | Source | Desc | yLabel | ExpReturn | Volatility | pf\_USAA\_Legacy | pf\_USAA\_New | pf\_Roth\_Legacy | pf\_Roth\_New | pf\_Div |
| 2 | USSPX | yahoo | USAA S&P 500 Index Member | Dollars | 0.0584645 | 0.0711347 | 0.4948 | 0.449955 | 0.0000000 | 0.00 | 0.00 |
| 3 | USAWX | yahoo | USAA World Growth | Dollars | 0.0317386 | 0.0753052 | 0.1429 | 0.000000 | 0.0000000 | 0.00 | 0.00 |
| 4 | USNQX | yahoo | USAA NASDAQ-100 Index | Dollars | 0.0930055 | 0.0979202 | 0.0000 | 0.549945 | 0.0000000 | 0.00 | 0.00 |
| 5 | USISX | yahoo | USAA Income Stock | Dollars | 0.0241307 | 0.0717094 | 0.2383 | 0.000000 | 0.0000000 | 0.00 | 0.00 |
| 6 | UFSGX | yahoo | USAA First Start Growth | Dollars | -0.0004527 | 0.0695155 | 0.1240 | 0.000000 | 0.0000000 | 0.00 | 0.00 |
| 8 | USCCX | yahoo | Cornerstone Conservative Fund | Dollars | 0.0016658 | 0.0361915 | 0.0000 | 0.000100 | 0.0000000 | 0.00 | 0.00 |
| 11 | VFINX | yahoo | Vanguard 500 Index Investor | Dollars | 0.0597201 | 0.0719599 | 0.0000 | 0.000000 | 0.5893816 | 0.30 | 0.25 |
| 13 | TMFGX | yahoo | Motley Fool Great America Investor | Dollars | 0.0512014 | 0.0949066 | 0.0000 | 0.000000 | 0.2996499 | 0.05 | 0.00 |
| 15 | QQQ | yahoo | PowerShares QQQ ETF | Dollars | 0.0965861 | 0.0939803 | 0.0000 | 0.000000 | 0.0001167 | 0.40 | 0.00 |
| 18 | IVOO | yahoo | Vanguard S&P Mid-Cap 400 ETF | Dollars | 0.0563663 | 0.0844225 | 0.0000 | 0.000000 | 0.0000000 | 0.15 | 0.00 |
| 21 | CAMIX | yahoo | Cambiar International Equity Inv | Dollars | 0.0219379 | 0.0720366 | 0.0000 | 0.000000 | 0.0000000 | 0.10 | 0.00 |
| 22 | SCHD | yahoo | Schwab U.S. Dividend Equity ETF | Dollars | 0.0501745 | 0.0670698 | 0.0000 | 0.000000 | 0.0000000 | 0.00 | 0.25 |
| 23 | VYM | yahoo | Vanguard High Dividend Yield ETF | Dollars | 0.0492435 | 0.0639490 | 0.0000 | 0.000000 | 0.0000000 | 0.00 | 0.25 |
| 24 | NOBL | yahoo | ProShares S&P 500 Dividend Aristocrats | Dollars | 0.0601265 | 0.0585058 | 0.0000 | 0.000000 | 0.0000000 | 0.00 | 0.25 |

# Legacy USAA Portfolio

## Time series analysis

This code plots out the legacy portfolio performance. It is not great, lagging behind the S&P 500. Ideally it would keep up with or beat that bench mark.

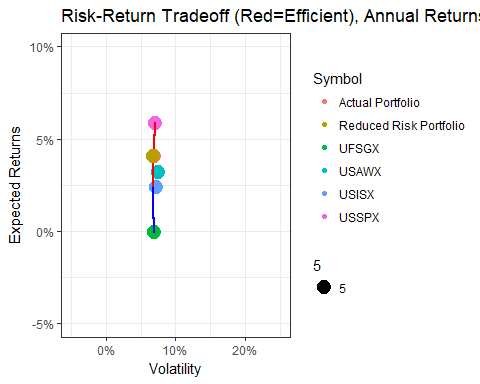
datay <- "pf\_USAA\_Legacy"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



## Risk return analysis

This section plots the funds on return and volatility axis. An ideal fund would be in the upper left corner of the plot. The USISX is in the wrong place, low returns and high risk. Uggh.

rrData <- plotRR(datay)  
rrData[[2]]



The same data as in the plot, but presented as a table.

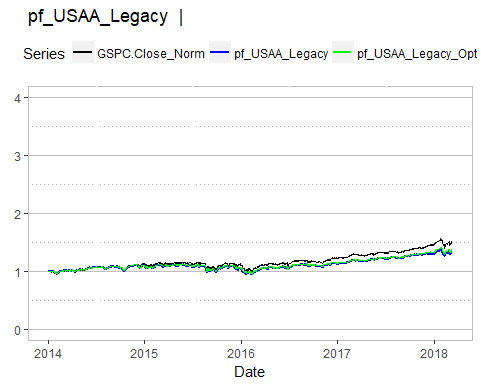
tblPort(rrData, datay)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Symbol | Desc | pf\_USAA\_Legacy | pf\_USAA\_Legacy\_Opt |
| 2 | USSPX | USAA S&P 500 Index Member | 0.4948 | 0.6926406 |
| 3 | USAWX | USAA World Growth | 0.1429 | 0.0000000 |
| 5 | USISX | USAA Income Stock | 0.2383 | 0.0105874 |
| 6 | UFSGX | USAA First Start Growth | 0.1240 | 0.2967720 |

## Time series analysis (Optimized)

As a final check, plot the optimized portfolio against the inputted portfolio and the benchmark.

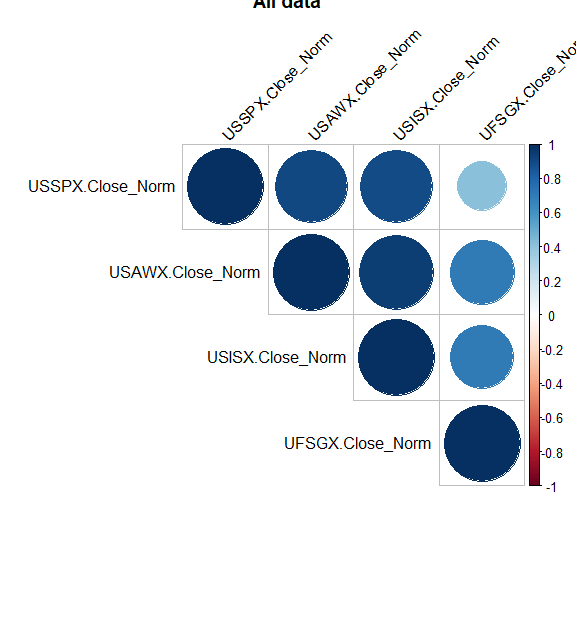
datayOpt <- paste(datay, "\_Opt", sep="")  
dfData <- pfUpdateReturn(datayOpt, dfData, rrData[[1]])  
ylim <- c(0, 4)  
myPlot <- plotSingleBench(datay, ylim)  
myPlot + geom\_line(data=dfData, aes\_string(x="date", y=datayOpt, colour=shQuote(datayOpt)), na.rm = TRUE)



## Correlation analysis

The more funds are negatively correlated, the more volatility can be reduced. This plot shows how the funds in the portfolio correlate.

plotCorr(datay)

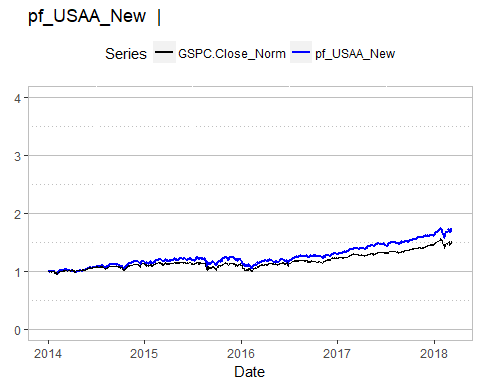


# New USAA Portfolio

## Time series analysis

This one is on a much better track, actually beating the S&P 500 index. Looking at the plots below, I think the mix should be 45% USSPX and 55% USNQX. This should have less volatility and better returns than the existing portfolio, although the models suggest that even less volatility could be had by including USCCX. I have hesitation about the bonds, with all the upcoming rate hikes by the Federal Reserve. But USCCX is highly uncorrelated…need to revisit this in the fall of 2017, after the moves by the Fed.

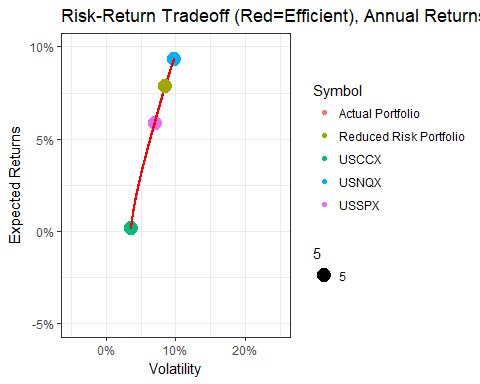
datay <- "pf\_USAA\_New"  
ylim <- c(0, 4)  
plotSingleBench(datay, ylim)



## Risk return analysis

This section plots the funds on return and volatility axis. An ideal fund would be in the upper left corner of the plot.

datay <- "pf\_USAA\_New"  
rrData <- plotRR(datay)  
rrData[[2]]



This table shows the portfolio. Column pf\_USAA\_New shows the inputted portfolio and pf\_USAA\_New\_Opt shows the portfolio with the same returns, but less risk.

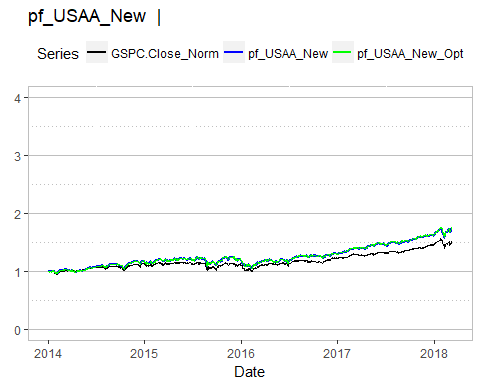
tblPort(rrData, datay)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Symbol | Desc | pf\_USAA\_New | pf\_USAA\_New\_Opt |
| 2 | USSPX | USAA S&P 500 Index Member | 0.449955 | 0.4120789 |
| 4 | USNQX | USAA NASDAQ-100 Index | 0.549945 | 0.5879211 |
| 8 | USCCX | Cornerstone Conservative Fund | 0.000100 | 0.0000000 |

## Time series analysis (Optimized)

As a final check, plot the optimized portfolio against the inputted portfolio and the benchmark.

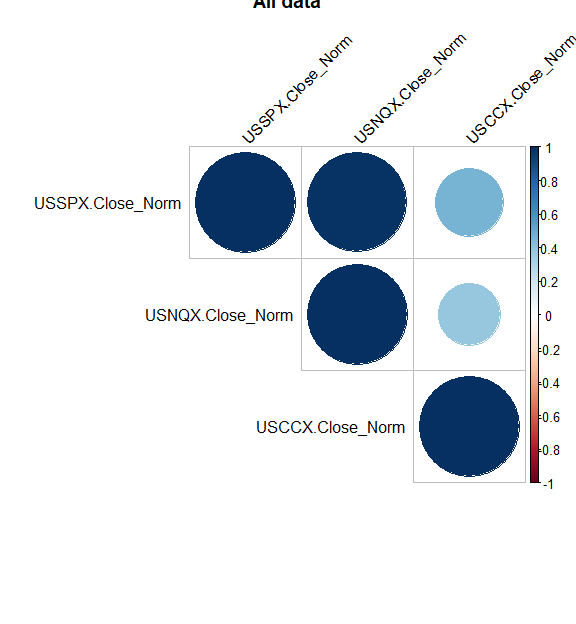
datay <- "pf\_USAA\_New"  
datayOpt <- paste(datay, "\_Opt", sep="")  
dfData <- pfUpdateReturn(datayOpt, dfData, rrData[[1]])  
ylim <- c(0, 4)  
myPlot <- plotSingleBench(datay, ylim)  
myPlot + geom\_line(data=dfData, aes\_string(x="date", y=datayOpt, colour=shQuote(datayOpt)), na.rm = TRUE)



## Correlation analysis

The more funds are negatively correlated, the more volatility can be reduced. This plot shows how the funds in the portfolio correlate.

plotCorr(datay)

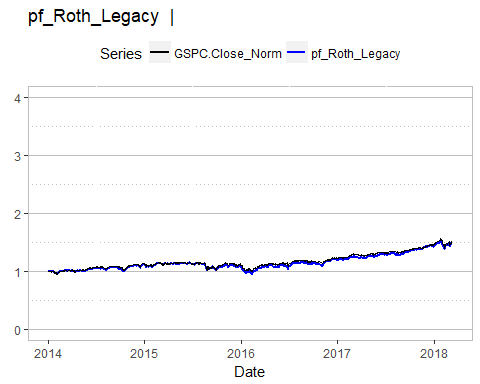


# Legacy Roth Portfolio

## Time series analysis

Uggh, just uggh. This one is performing much worse than I expected.

datay <- "pf\_Roth\_Legacy"  
ylim <- c(0, 4)  
plotSingleBench(datay, ylim)



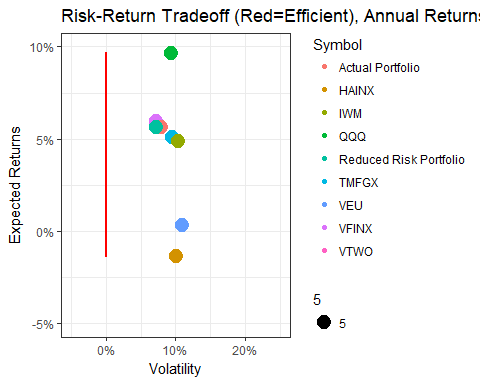
## Risk return analysis

This section plots the funds on return and volatility axis. An ideal fund would be in the upper left corner of the plot.

rrData <- plotRR(datay)

## [1] "Failed to find efficiency boundary"

rrData[[2]]



This table shows the portfolio. Column pf\_Roth\_Legacy shows the inputted portfolio and pf\_Roth\_Legacy\_Opt shows the portfolio with the same returns, but less risk.

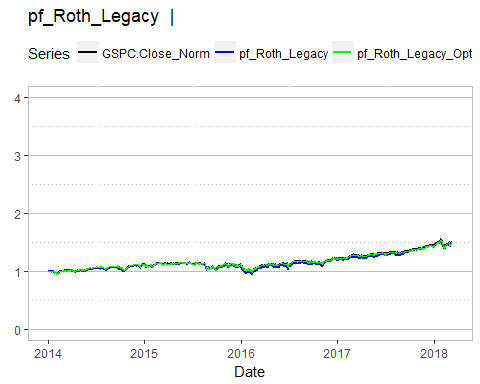
tblPort(rrData, datay)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Symbol | Desc | pf\_Roth\_Legacy | pf\_Roth\_Legacy\_Opt |
| 10 | VTWO | Vanguard Russell 2000 ETF | 0.1105018 | 0.0000000 |
| 11 | VFINX | Vanguard 500 Index Investor | 0.5893816 | 0.9541982 |
| 13 | TMFGX | Motley Fool Great America Investor | 0.2996499 | 0.0000000 |
| 14 | IWM | iShares Russell 2000 | 0.0001167 | 0.0000000 |
| 15 | QQQ | PowerShares QQQ ETF | 0.0001167 | 0.0000000 |
| 16 | HAINX | Harbor International Institutional | 0.0001167 | 0.0458018 |
| 17 | VEU | Vanguard FTSE All-Wld ex-US ETF | 0.0001167 | 0.0000000 |

## Time series analysis (Optimized)

As a final check, plot the optimized portfolio against the inputted portfolio and the benchmark.

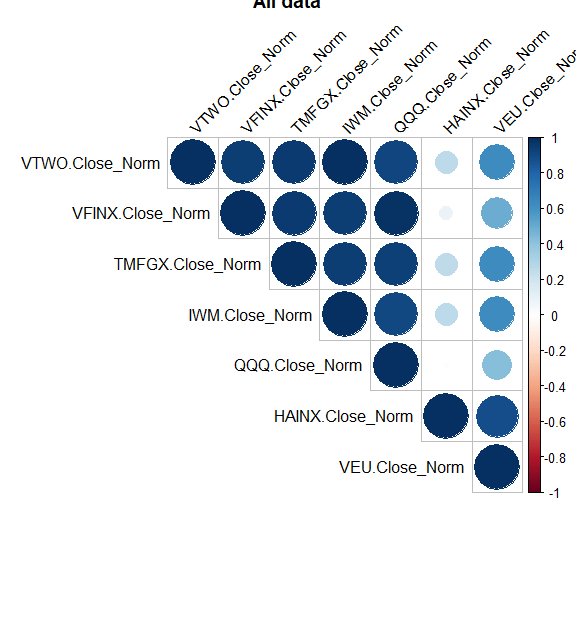
datayOpt <- paste(datay, "\_Opt", sep="")  
dfData <- pfUpdateReturn(datayOpt, dfData, rrData[[1]])  
ylim <- c(0, 4)  
myPlot <- plotSingleBench(datay, ylim)  
myPlot + geom\_line(data=dfData, aes\_string(x="date", y=datayOpt, colour=shQuote(datayOpt)), na.rm = TRUE)



## Correlation analysis

The more funds are negatively correlated, the more volatility can be reduced. This plot shows how the funds in the portfolio correlate.

plotCorr(datay)

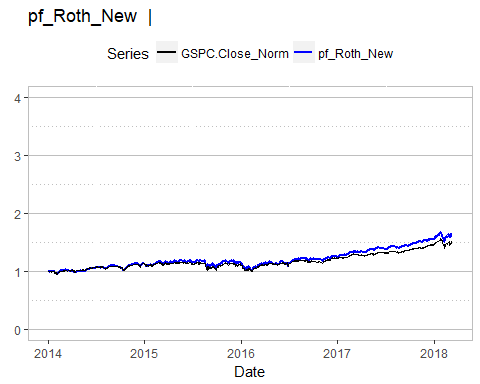


# New Roth Portfolio

## Time series analysis

CAPM portfolio theory suggests that only VTWO and QQQ are needed, but both of these track strongly with the NASDAQ 100. Looking at the correlation plot, TMFGX and CAMIX do not correlate with either one so I am going to include it as well as a hedge against volatility. The final mix will be 30% VFINX, 5% of TMFGX (very low volatility so like a bond fund), 10% of CAMIX (international exposure), 15% to IVOO (mid cap), and 40% of QQQ.

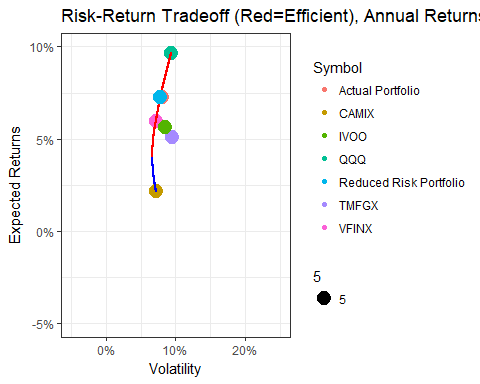
datay <- "pf\_Roth\_New"  
ylim <- c(0, 4)  
plotSingleBench(datay, ylim)



## Risk return analysis

This section plots the funds on return and volatility axis. An ideal fund would be in the upper left corner of the plot.

rrData <- plotRR(datay)  
rrData[[2]]



This table shows the portfolio. Column pf\_Roth\_New shows the inputted portfolio and pf\_Roth\_New\_Opt shows the portfolio with the same returns, but less risk.

tblPort(rrData, datay)

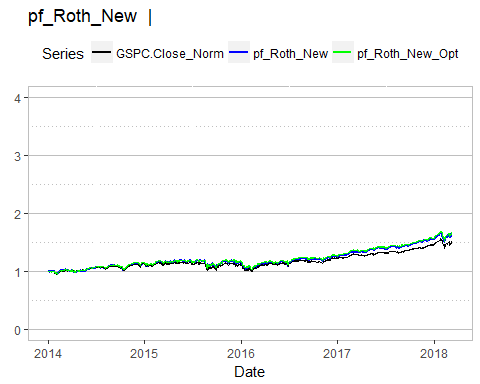
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Symbol | Desc | pf\_Roth\_New | pf\_Roth\_New\_Opt |
| 11 | VFINX | Vanguard 500 Index Investor | 0.30 | 0.656854 |
| 13 | TMFGX | Motley Fool Great America Investor | 0.05 | 0.000000 |
| 15 | QQQ | PowerShares QQQ ETF | 0.40 | 0.343146 |
| 18 | IVOO | Vanguard S&P Mid-Cap 400 ETF | 0.15 | 0.000000 |
| 21 | CAMIX | Cambiar International Equity Inv | 0.10 | 0.000000 |

## 

## Time series analysis (Optimized)

As a final check, plot the optimized portfolio against the inputted portfolio and the benchmark.

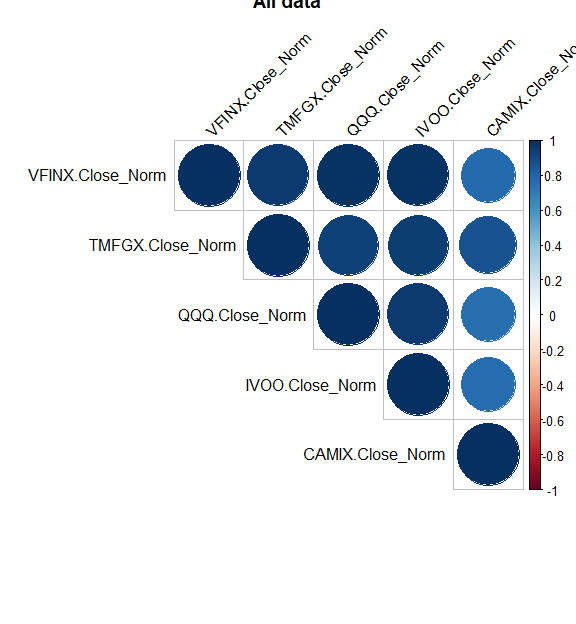
datayOpt <- paste(datay, "\_Opt", sep="")  
dfData <- pfUpdateReturn(datayOpt, dfData, rrData[[1]])  
ylim <- c(0, 4)  
myPlot <- plotSingleBench(datay, ylim)  
myPlot + geom\_line(data=dfData, aes\_string(x="date", y=datayOpt, colour=shQuote(datayOpt)), na.rm = TRUE)



## Correlation analysis

The more funds are negatively correlated, the more volatility can be reduced. This plot shows how the funds in the portfolio correlate.

plotCorr(datay)

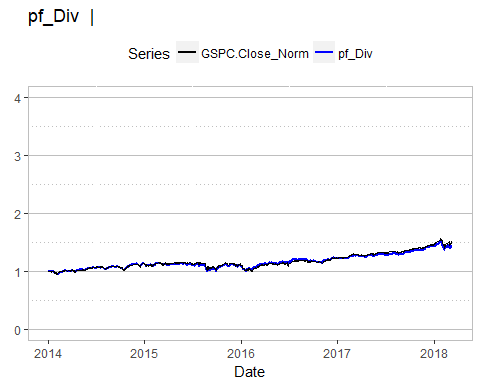


# Dividend Portfolio

## Time series analysis

This portfolio looks at a weighting favoring dividend funds..

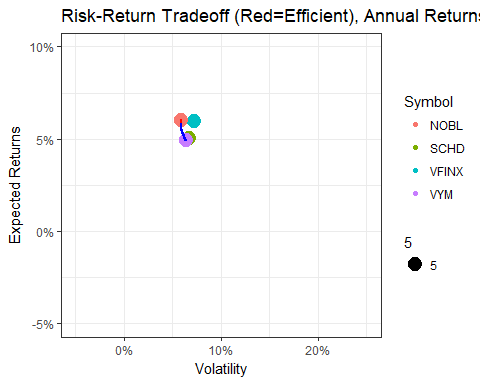
datay <- "pf\_Div"  
ylim <- c(0, 4)  
plotSingleBench(datay, ylim)



## Risk return analysis

This section plots the funds on return and volatility axis. An ideal fund would be in the upper left corner of the plot.

rrData <- plotRR(datay, TRUE)  
rrData[[2]]



This table shows the portfolio. Column pf\_Div shows the inputted portfolio and pf\_Div\_Opt shows the portfolio with the same returns, but less risk.

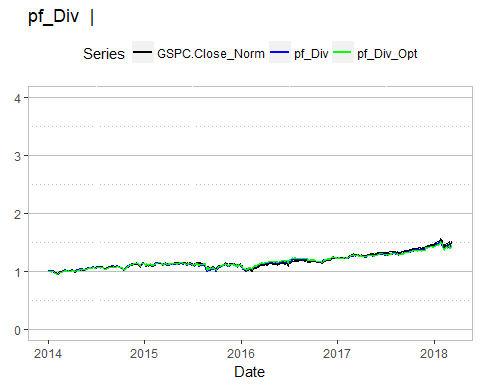
tblPort(rrData, datay)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Symbol | Desc | pf\_Div | pf\_Div\_Opt |
| 11 | VFINX | Vanguard 500 Index Investor | 0.25 | 0.000000 |
| 22 | SCHD | Schwab U.S. Dividend Equity ETF | 0.25 | 0.000000 |
| 23 | VYM | Vanguard High Dividend Yield ETF | 0.25 | 0.465448 |
| 24 | NOBL | ProShares S&P 500 Dividend Aristocrats | 0.25 | 0.534552 |

## Time series analysis (Optimized)

As a final check, plot the optimized portfolio against the inputted portfolio and the benchmark.

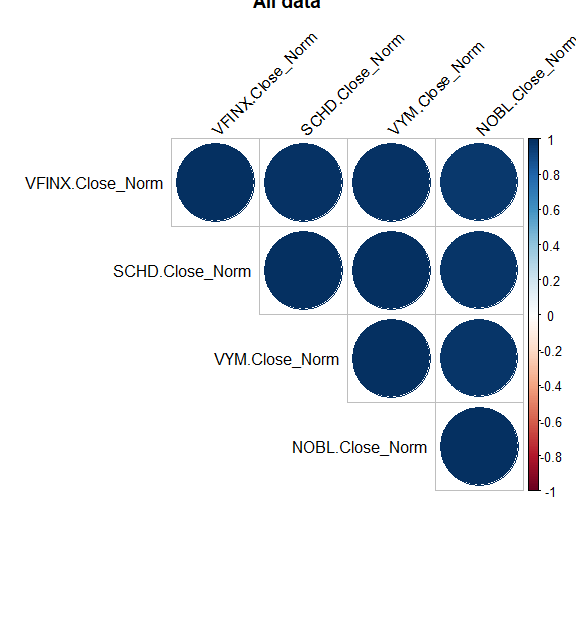
datayOpt <- paste(datay, "\_Opt", sep="")  
dfData <- pfUpdateReturn(datayOpt, dfData, rrData[[1]])  
ylim <- c(0, 4)  
myPlot <- plotSingleBench(datay, ylim)  
myPlot + geom\_line(data=dfData, aes\_string(x="date", y=datayOpt, colour=shQuote(datayOpt)), na.rm = TRUE)



## Correlation analysis

The more funds are negatively correlated, the more volatility can be reduced. This plot shows how the funds in the portfolio correlate.

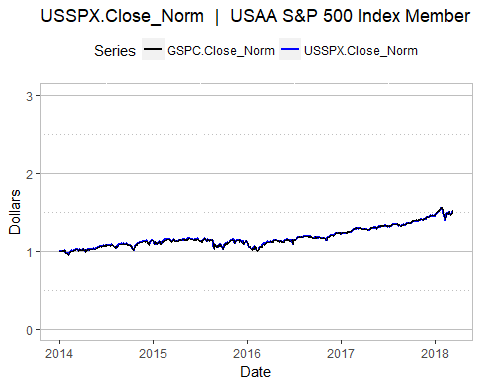
plotCorr(datay)



# Appendix

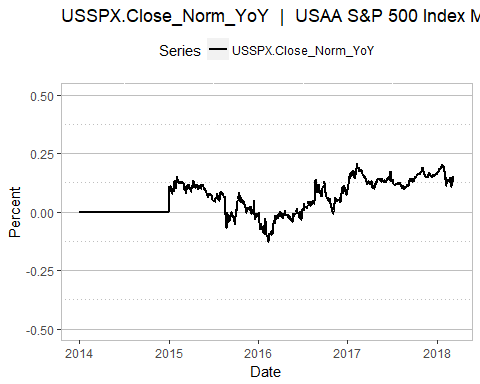
## USSPX, USAA S&P 500 Index Member

datay <- "USSPX.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



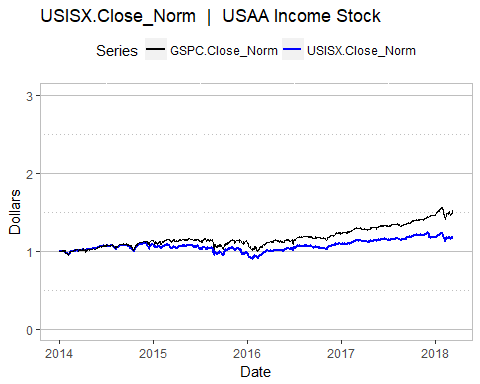
## USSPX Year over year, USAA S&P 500 Index Member

datay <- "USSPX.Close\_Norm\_YoY"  
ylim <- c(-0.5, 0.5)  
plotSingleQuick(datay, ylim, "Percent")



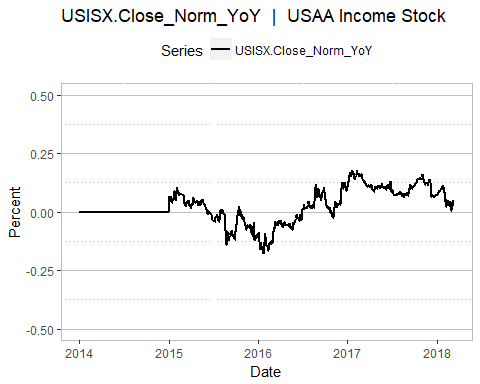
## USISX, USAA Income Stock

datay <- "USISX.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



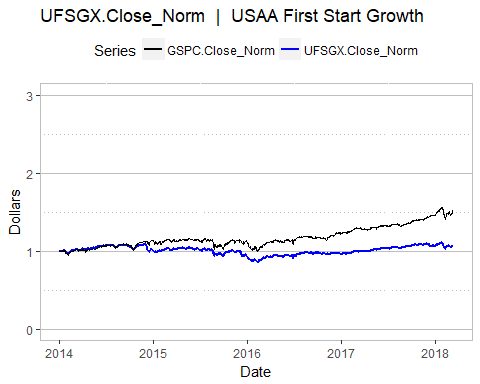
## USISX Year over year, USAA Income Stock

datay <- "USISX.Close\_Norm\_YoY"  
ylim <- c(-0.5, 0.5)  
plotSingleQuick(datay, ylim, 'Percent')



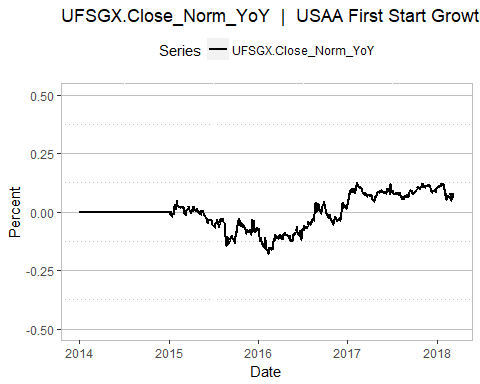
## FSGX, USAA First Start Growth

datay <- "UFSGX.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



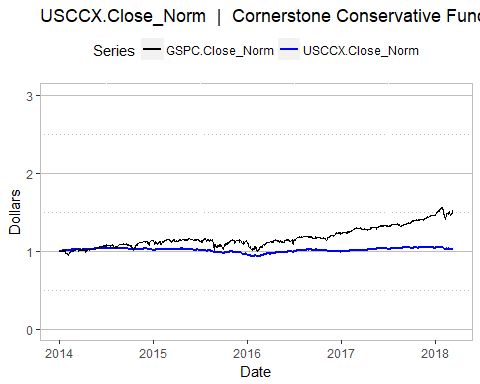
## FSGX Returns, USAA First Start Growth

datay <- "UFSGX.Close\_Norm\_YoY"  
ylim <- c(-0.5, 0.5)  
plotSingleQuick(datay, ylim, "Percent")



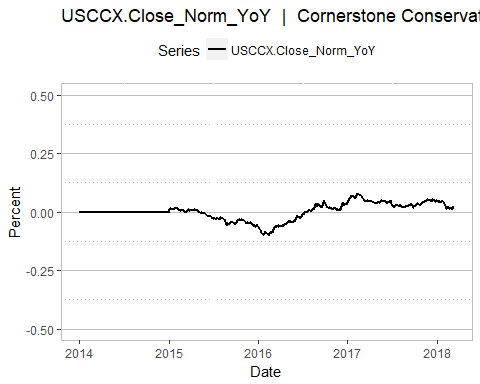
## USCCX, USAA Cornerstone Conservative

datay <- "USCCX.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



## USCCX Year over year, USAA Cornerstone Conservative

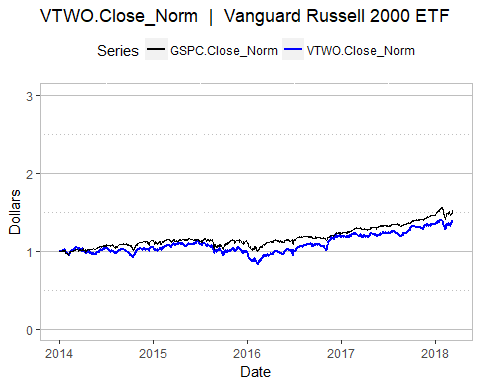
datay <- "USCCX.Close\_Norm\_YoY"  
ylim <- c(-0.5, 0.5)  
plotSingleQuick(datay, ylim, 'Percent')



## VTWO, Vanguard Russell 2000 ETF

Not as much history on this one as I would like.

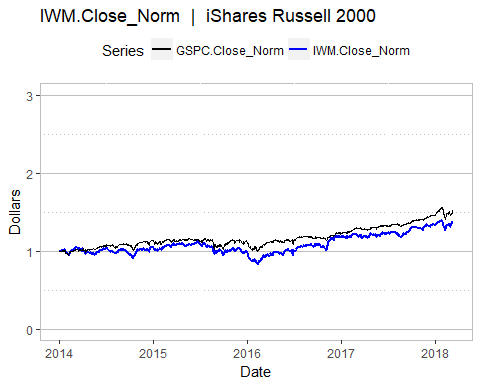
datay <- "VTWO.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



## IWM, iShares Russell 2000

Not as much history on this one as I would like.

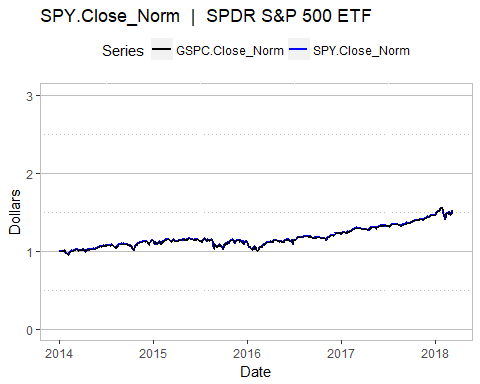
datay <- "IWM.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



## SPY, SPDR S&P 500 ETF Trust

Should be real close to the S&P 500

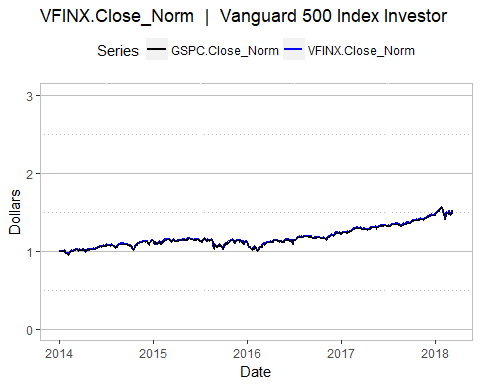
datay <- "SPY.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



## VFINX, Vanguard 500 Index Investor

Should be real close to the S&P 500

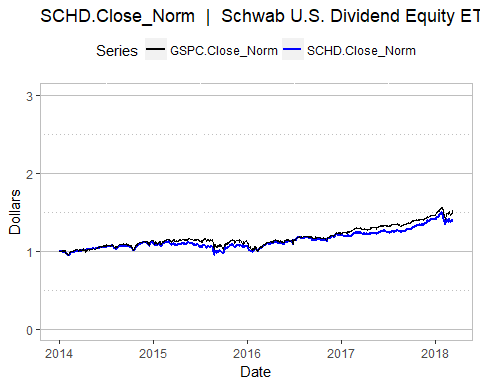
datay <- "VFINX.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



## SCHD, Schwab U.S. Dividend Equity ETF

Nice little pick of divident paying stocks on Dow Jones Dividend 100 index

datay <- "SCHD.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)



## VYM, Vanguard High Dividend Yield ETF

Another nice little pick of divident paying stocks on Dow Jones Dividend 100 index. Expense ratio: 0.08%

datay <- "VYM.Close\_Norm"  
ylim <- c(0, 3)  
plotSingleBench(datay, ylim)

