MScFE 610 Econometrics (C19-S4)

Group Work Assignment Submission 3 M7

Kwadwo Amo-Addai (kwadwoamoad@gmail.com)

David Sabater Dinter (david.sabater@gmail.com)

Ruben Ohayon (rubensacha.ohayon@gmail.com)

Andrea Chello (chelloandrea@gmail.com)

Pasin Marupanthorn (oporkabbb@hotmail.com)

3.3.1. Algorithmic Trading

Design your own algorithmic trading strategy in R.
Number of assets in the strategy: one or more assets
Type of asset: you select it (stock, commodity, FX, crypto etc)
Timeframe: you select it
Coding language: R; you can also use Excel for basic calculations and testing
Model: regression, ARMA, GARCH, VAR, VEC or any other quantitative model you know. You can combine models with technical analysis indicators (MA, MACD, Bollinger bands etc) as in Module 7 examples. You can also use machine learning algorithms (though it is not compulsory).
You can use Module 7 examples or models from previous modules.

1. Explain the algorithm step by step
2. Provide R code and/or Excel calculations
3. Provide charts
4. Calculate returns, cumulative returns, standard deviation and forecasts
5. Indicate research papers or books on this topic

3.3.2. Improve the Strategy

1. Indicate ways for improving the previous algorithmic trading strategy
2. Indicate research papers or books on the topic

3.3.3. Report Writing

Write up all the results from the analyses required in this project into a well-structured formal report with introduction, comments, code, and conclusion sections.

The group work project should contain:

- R or Python code (or both coding languages)
- Excel (not compulsory)
- docx, pdf, xlsx or txt file with comments, charts, results, and conclusion. (You can also use Open Office if you do not have Microsoft Office.)

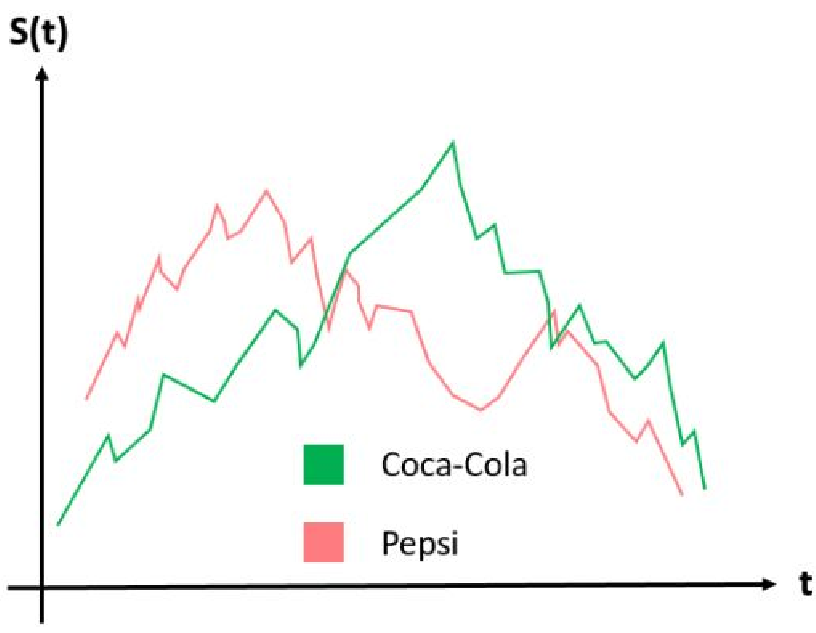
Report - Pairs Trading Strategy

Introduction

A standard pairs trading strategy involves a long-short pair of equities (such as stocks). Two companies in the same sector are likely to be exposed to similar market factors. Occasionally their relative stock prices will diverge due to certain events, but will revert to the long-running mean.

This strategy was pioneered by Garry Bamberger and Nunzio Tartaglia at Morgan Stanley around the 1980s. Most of the hedge funds rely on this strategy today as well.

So statistical arbitrage such as pairs trading is a market-neutral strategy. We make an assumption how stock prices should move relative to each other. Let’s consider two companies within the same industry: Pepsi (PEP) and Coca-Cola (KO). We can make an assumption that if these two stocks diverge, they should eventually re-converge. Why? Because they are very similar and they are in the same industry.



Other examples may include commodities such as gold and the stock of gold mines. Another example is oil and stocks of oil producer companies. Of course there must be some correlation between these commodities and assets.

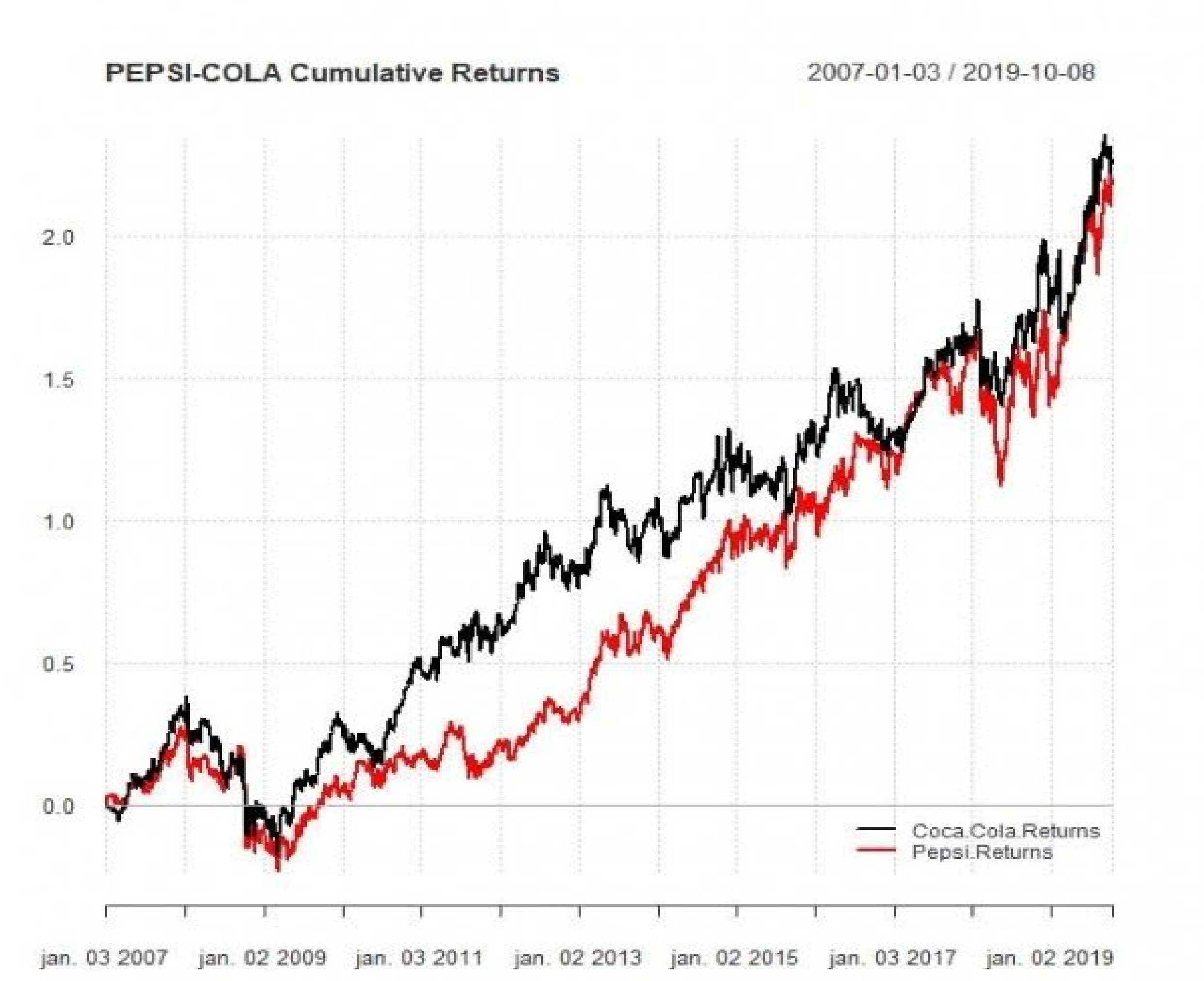
Correlation in R

```
# install.packages("...")
library('tseries'); library('quantmod'); library('PerformanceAnalytics'); library('urca'); library('roll');
##  CORRELATION BETWEEN A PAIR OF ASSETS
# Pepsi and Coca-Cola stocks
my_portfolio <- c("PEP", "KO")
stocks <- lapply(my_portfolio,getSymbols,auto.assign=FALSE)
names(stocks) <- my_portfolio
# get the adjusted closing prices
pep <- stocks[[my_portfolio[1]]][,6]
ko <- stocks[[my_portfolio[2]]][,6]
# get the daily returns
return_pep = dailyReturn(pep,type='arithmetic')
return_ko = dailyReturn(ko,type='arithmetic')
colnames(return_pep) <- 'Pepsi Returns'
colnames(return_ko) <- 'Coca-Cola Returns'
# calculating the covariance
cor(return_ko,return_pep)
# plotting the daily returns
chart.CumReturns(cbind(return_ko,return_pep),main="PEPSI-COLA Cumulative
Returns",legend.loc="bottomright")
```

We are going to deal with Pepsi (PEP) and Coca-Cola (KO) stocks. The aim is to calculate the correlation between these assets.

We are after the adjusted closing prices – this is why we want to get the 6-th column. Then we calculate the daily returns, the covariance and finally plot the results.

According to the upcoming plot below, there must be some correlation between the two assets. The exact value is 0.6816855. Because it is a positive number (approximately close to 1) we can say that there is a positive correlation between Pepsi and Coca-Cola stocks.



Cointegration in R

COINTEGRATION

So we have to use the Pepsi (xt) adjusted closing prices and the Coca-Cola (yt) adjusted closing prices in the regression.

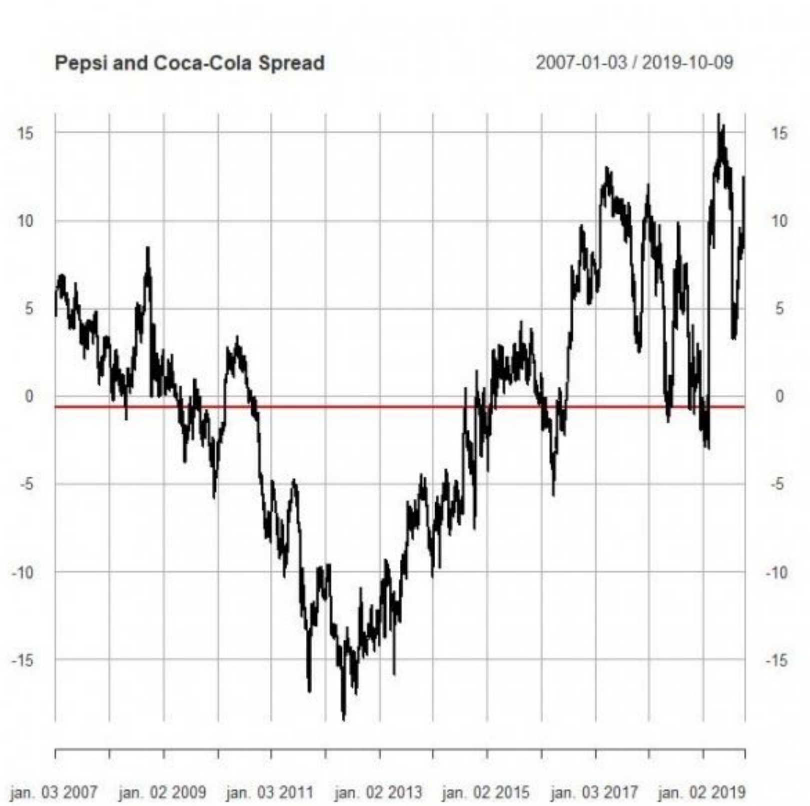
Because there is a high positive correlation factor (~0.7) of course there is some linear relationship between the prices.

The β value is the result of the regression. After that we can calculate the s spread based on the formula above.

If we consider the spread it performs periodical oscillations around some mean value.

Obviously, such a pair is easy and convenient to trade with since it is known that the spread will return to its average value with high probability.

```
# spread can be calculated based on linear regression
spread <- pep - lm(pep~0+ko)$coefficients[1]*ko
# there is some oscillation around the mean of the spread
spread_mean <- xts(rep(mean(spread),length(spread)),order.by=as.Date(index(spread)))
# let's plot the spread and it's mean
plot(cbind(spread,spread_mean),main="Pepsi and Coca-Cola Spread")
```



Now let’s use the Engle-Granger cointegration test. It consists of testing whether two paired assets (stocks in this case) prices linear regression residuals (so the spread itself) are stationary.

The single stock prices (Pepsi and Coca-Cola stock prices) are not stationary. In order to use pairs trading strategy first we have to make sure that the spread itself so the assets’ prices differences are stationary.

If we knew β (and we know it from the linear regression) we could just test it for stationarity with for example Dickey-Fuller test or Phillips-Perron test.

```
# Augmented Dickey-Fuller test
adf.test(spread)
# Phillips-Perron test
pp.test(spread, lshort=F)
```

We have to check the p-values and if these values < 0.05 then we can say with 95% confidence level that the process is stationary.

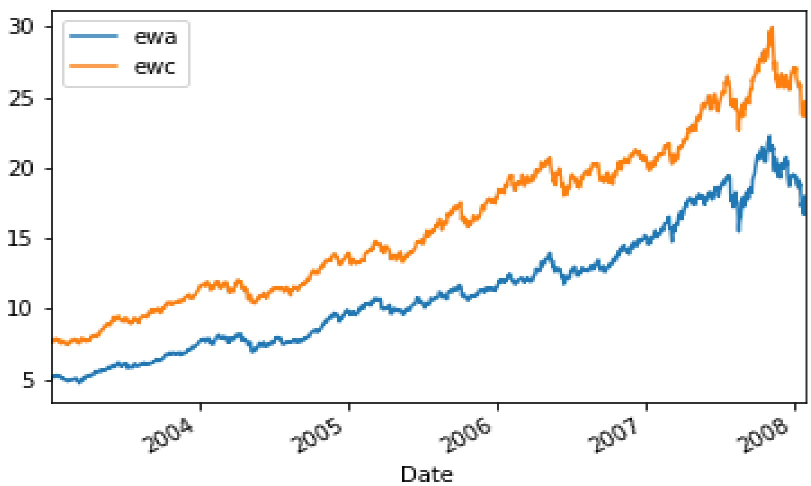
In this case the values are greater than 0.05 which means that this pair (Pepsi and Coca-Cola) is not good for statistical arbitrage and pairs trading.

Testing Correlation and Cointegration of ETF pairs (in Python)

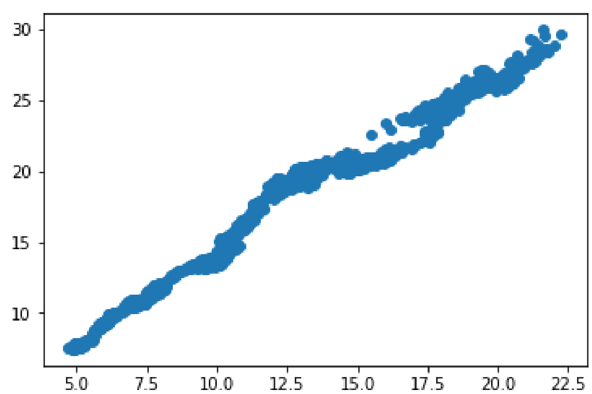
Let us take another example. There is an iShares MSCI Australia and an iShares MSCI Canada. These are two ETFs which track the economy of Australia, respectively Canada. Their ticker symbols are EWA and EWC. We will test their cointegration. First, we define the timeframe and download the data from yahoo finance.

```
import pandas_datareader.data as web
import matplotlib.pyplot as plt
from scipy import stats
import statsmodels.tsa.stattools as ts
import datetime
"""
There is an iShares MSCI Australia and an iShares MSCI Canada. These are two ETFs which
track the economy of Australia, respectively Canada.
Their ticker symbols are EWA and EWC. We will test their cointegration. First, we define the
timeframe and download the data from yahoo finance.
"""
start = datetime.datetime(2003, 1, 1)
end = datetime.datetime(2008, 1, 27)
ewa_prices = web.DataReader("EWA", 'yahoo', start, end)
ewc_prices = web.DataReader("EWC", 'yahoo', start, end)
#
```

```
ewa_close=ewa_prices['Adj Close']
ewc_close=ewc_prices['Adj Close']
#
ewa_close.plot(label='ewa', legend=True)
ewc_close.plot(label='ewc', legend=True)
plt.show()
#
```

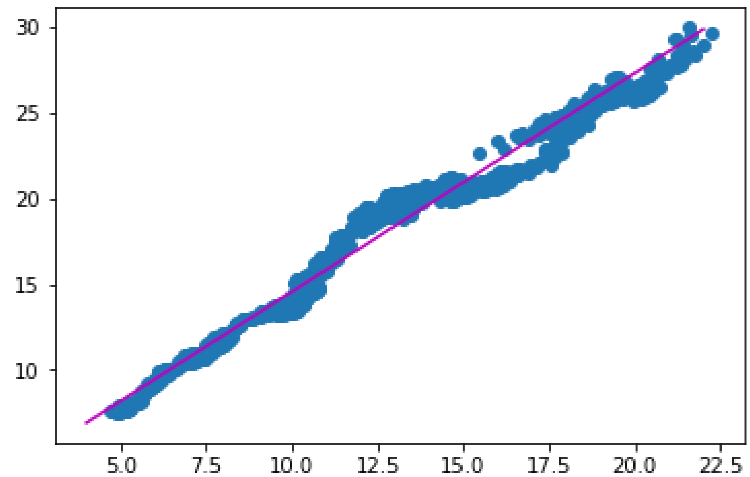



```
#
plt.scatter(ewa_close, ewc_close)
plt.show()
```

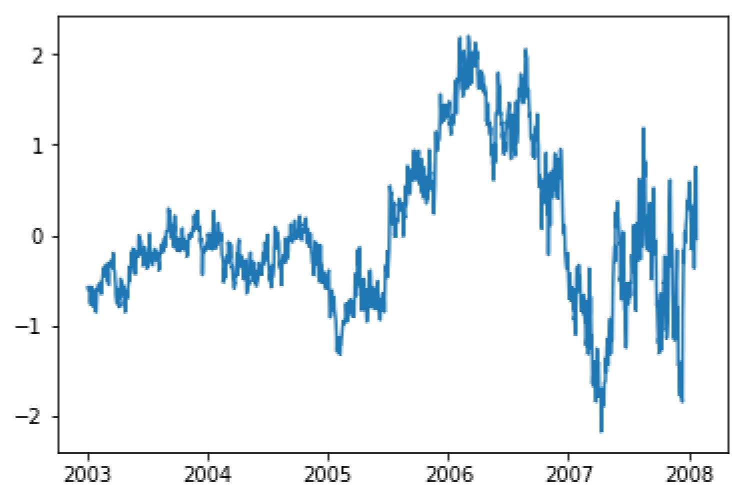



```
#

# The graph produced fairly looks like a line. Perfect. So we fit a line.
slope, intercept, r_value, p_value, std_err = stats.linregress(ewa_close, ewc_close)
print("slope: " + str(slope) +
```

```
    "\nintercept: " + str(intercept) +
    "\nr-value: " + str(r_value) +
    "\np-value: " + str(p_value) +
    "\nstd-err: " + str(std_err))
#
plt.scatter(ewa_close, ewc_close)
x=[4,22]
plt.plot(x,[slope*x_i+intercept for x_i in x], color='m')
plt.show()
```




```
# Now we compute the error and test if it is mean-reverting.
error = ewc_close - slope * ewa_close - intercept
plt.plot(error)
plt.show()
```



```
# Execute ADF Test
ts.adfuller(error,1)
```

Output:

```
(-3.3459094549609389,
 0.012946959495104054,
 1,
 1273,
 {'1%': -3.4354973175106842,
  '10%': -2.5679802172809003,
  '5%': -2.8638130956084464},
 -845.44796472685812)
```

We got a p-value of **0.0129**. That means we are statistically significant. So we could go on to build a trading strategy by trading on the error.

The interesting part is that it is only significant, since I chose the timeframe appropriately.

For other timeframes the cointegration relationship interestingly does not hold.

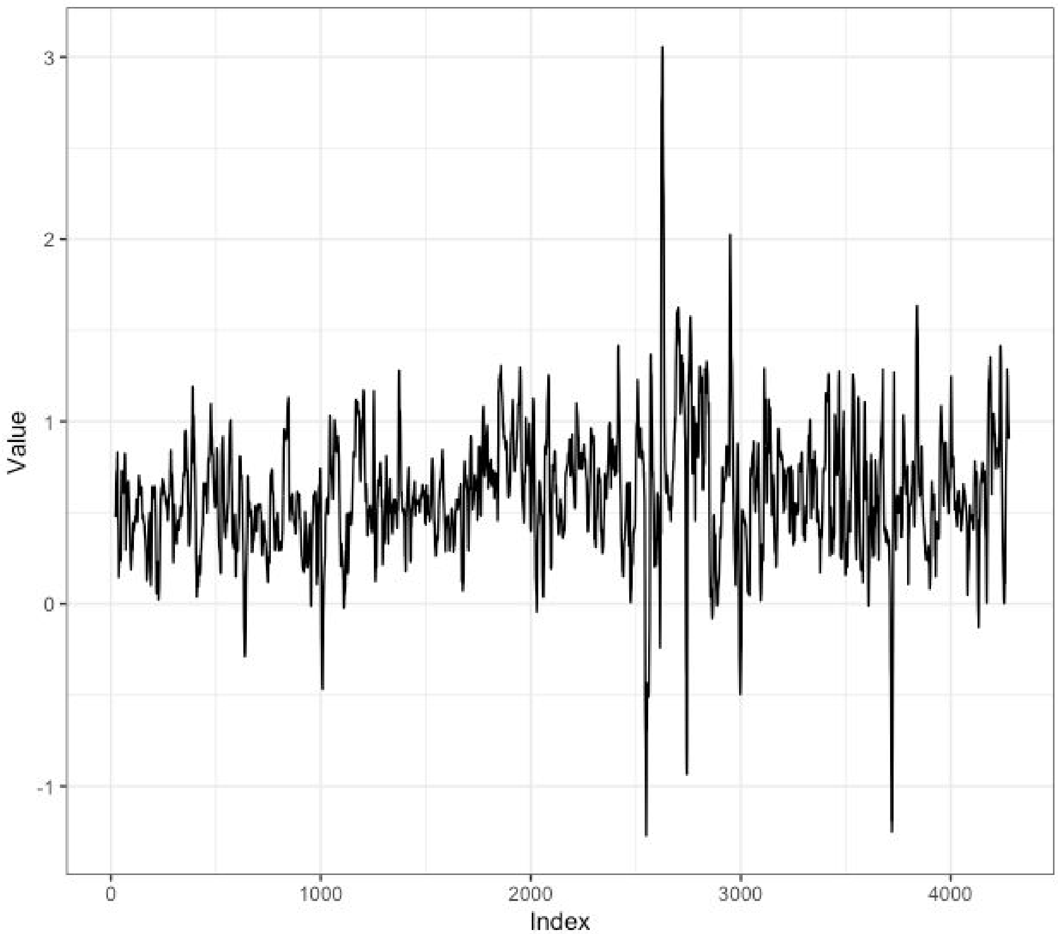
Implementing the Trading Strategy (for Cointegrated ETFs - EWA & EWC) in R

We finally selected EWA and EWC ETFs based on their cointegration and also easy access for retail traders, we used the code from [Cointegrated ETF Pairs Part II], fixing a few issues with vector sizes based on rolling linear regression of 21 days, we also extended the backtest from 2003-01-01 until 2019-12-31. The implementation also includes performance metrics from the backtest.
We use Bollinger Bands as the indicator to derive the trading signals and rules, based on a rolling z-score and using the spread of the two ETFs approximated by a rolling linear regression model.

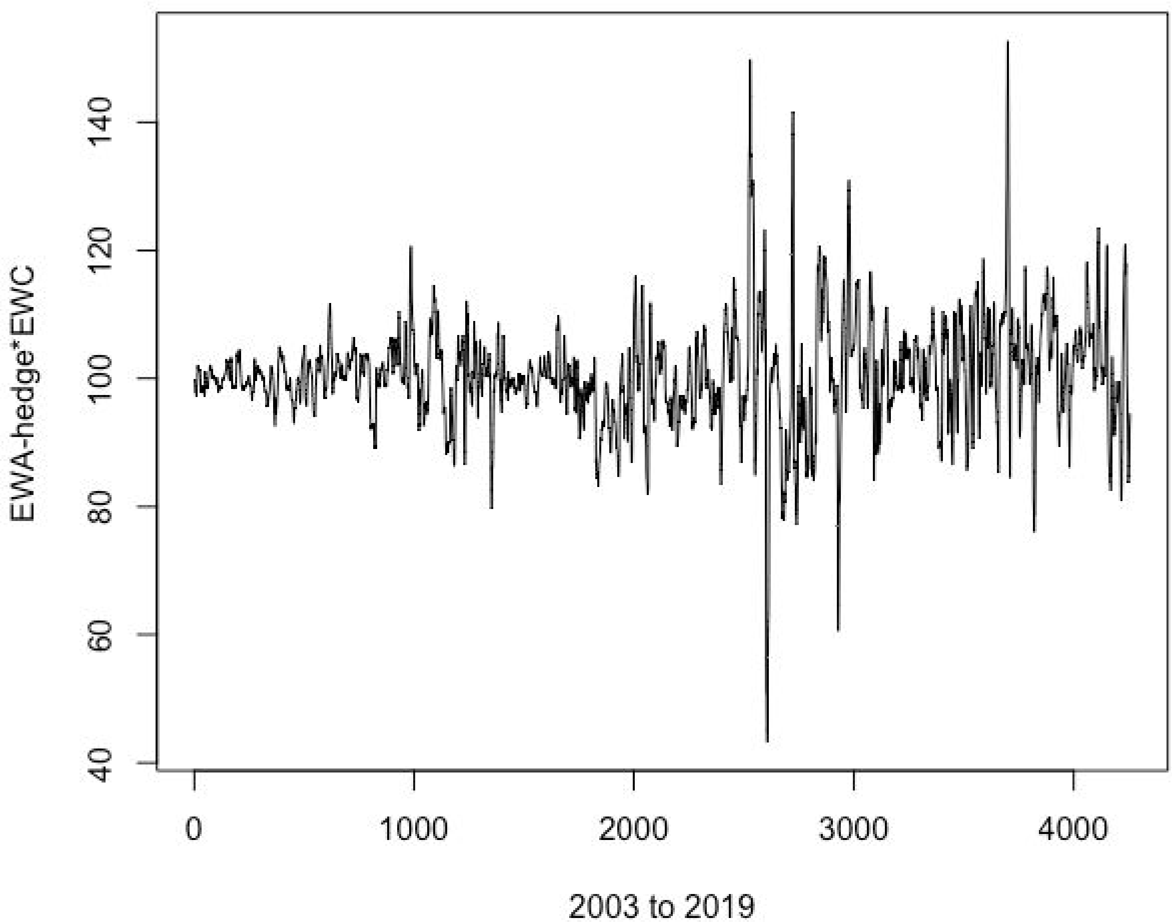
###########CODE##############

```
install.packages("devtools") # if not installed
install.packages("FinancialInstrument") #if not installed
install.packages("PerformanceAnalytics") #if not installed
install.packages("knitr") #if not installed
install.packages("tseries") #if not installed
install.packages("roll") #if not installed

# next install blotter from GitHub
devtools::install_github("braverock/blotter")
# Leverage quantstrat from GitHub as backtesting framework to evaluate EWA-EWC pair
trading strategy
devtools::install_github("braverock/quantstrat")
# Install quantstart plugin IKTrading including investing functions
devtools::install_github("IlyaKipnis/IKTrading")
# Install quantstart plugin DSTrading including investing signals
devtools::install_github("IlyaKipnis/DSTrading")

require(quantstrat)
require(IKTrading)
require(DSTrading)
require(knitr)
require(PerformanceAnalytics)
require(tseries)
require(roll)
require(ggplot2)
```

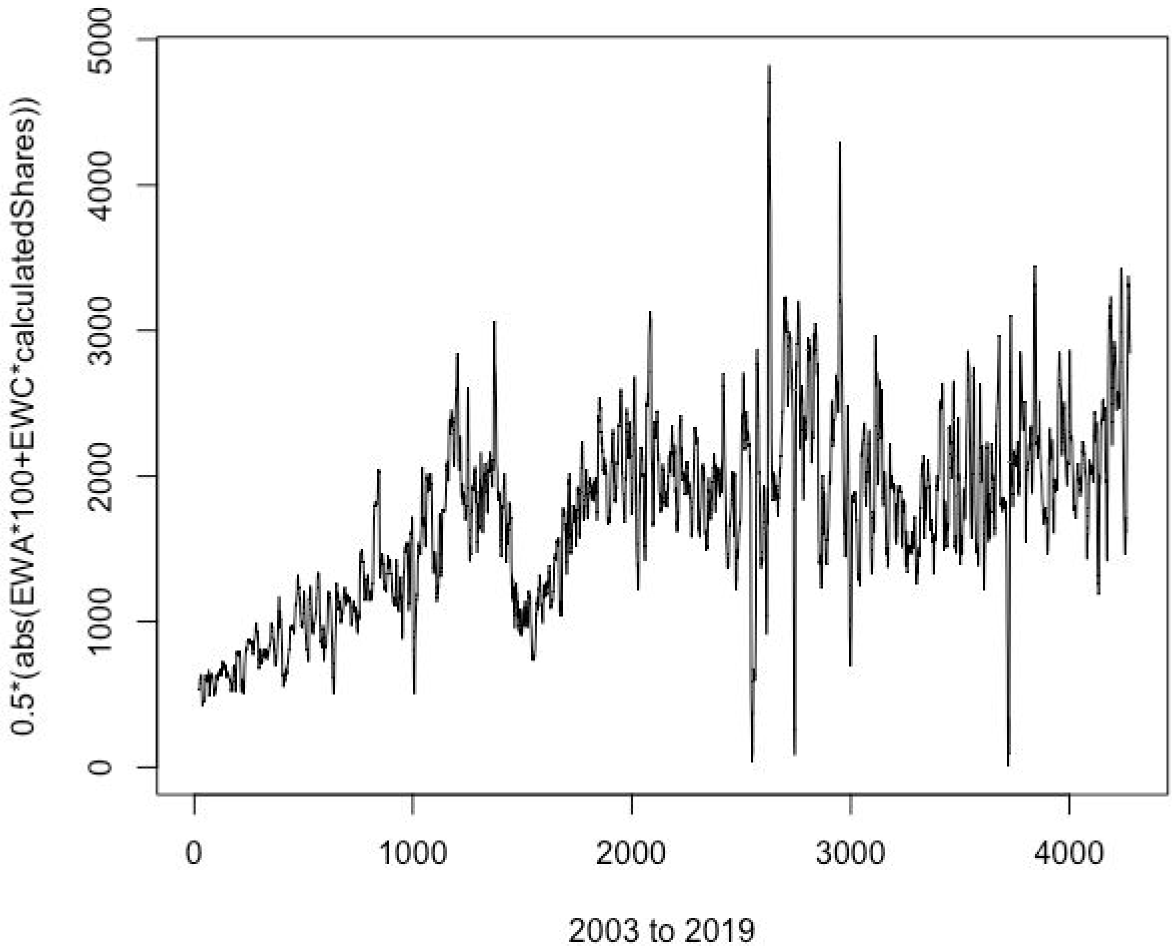
```
# Full test
initDate="2003-01-01"
from="2003-01-01"
to="2019-12-31"

## Create "symbols" for Quanstrat
## adj1 = EWA (Australia), adj2 = EWC (Canada)

## Get ETF historical prices
getSymbols("EWA", from=from, to=to)
getSymbols("EWC", from=from, to=to)
dates = index(EWA)

adj1 = unclass(EWA$EWA.Adjusted)
adj2 = unclass(EWC$EWC.Adjusted)

## Ratio (EWC/EWA)
ratio = adj2/adj1

## Rolling regression
window = 21
lm = roll_lm(adj2,adj1,window)

## Plot beta
rollingbeta <- fortify.zoo(lm$coefficients[,2],melt=TRUE)
ggplot(rollingbeta, ylab="beta", xlab="time") + geom_line(aes(x=Index,y=Value)) + theme_bw()
```




```
## Calculate the spread
sprd <- vector(length=4278-21)
for (i in 21:4278) {
  sprd[i-21] = (adj1[i]-rollingbeta[i,3]*adj2[i]) + 98.86608 ## Make the mean 100
}
plot(sprd, type="l", xlab="2003 to 2019", ylab="EWA-hedge*EWC")
```




```
## Find minimum capital assuming 50% margin from broker
hedgeRatio = ratio*rollingbeta$Value*100
spreadPrice = 0.5*abs(adj2*100+adj1*hedgeRatio)
plot(spreadPrice, type="l", xlab="2003 to 2019",
ylab="0.5*(abs(EWA*100+EWC*calculatedShares))")
```




```
## Combine columns and turn into xts (time series), remove unnecessary columns
close = sprd
date = as.data.frame(dates[22:4278])
data = cbind(date, close)
dfdata = as.data.frame(data)
xtsData = xts(dfdata, order.by=as.Date(dfdata$date))
xtsData$close = as.numeric(xtsData$close)
xtsData$dum = vector(length = 4257)
xtsData$dum = NULL
xtsData$dates.22.4278. = NULL

## Add SMA, moving stdev, and z-score
rollz<-function(x,n){
  avg=rollapply(x, n, mean)
  std=rollapply(x, n, sd)
```

```
  z=(x-avg)/std
  return(z)
}

## Varying the lookback has a large affect on the data
xtsData$zScore = rollz(xtsData,50)
symbols = 'xtsData'

## Backtest
currency('USD')
Sys.setenv(TZ="UTC")
stock(symbols, currency="USD", multiplier=1)

#trade sizing and initial equity settings
tradeSize <- 10000
initEq <- tradeSize

strategy.st <- portfolio.st <- account.st <- "EWA_EWC"
rm.strat(portfolio.st)
rm.strat(strategy.st)
initPortf(portfolio.st, symbols=symbols, initDate=initDate, currency='USD')
initAcct(account.st, portfolios=portfolio.st, initDate=initDate, currency='USD',initEq=initEq)
initOrders(portfolio.st, initDate=initDate)
strategy(strategy.st, store=TRUE)

#Signals to enter and exit positions
strategy.st <- add.signal(strategy = strategy.st,
                    name="sigFormula",
                    arguments = list(label = "enterLong",
                                formula = "zScore < -1",
                                cross = TRUE),
                    label = "enterLong")

strategy.st <- add.signal(strategy = strategy.st,
                    name="sigFormula",
                    arguments = list(label = "exitLong",
                                formula = "zScore > 1",
                                cross = TRUE),
                    label = "exitLong")

strategy.st <- add.signal(strategy = strategy.st,
                    name="sigFormula",
                    arguments = list(label = "enterShort",
```

```
                    formula = "zScore > 1",
                    cross = TRUE),
               label = "enterShort")

strategy.st <- add.signal(strategy = strategy.st,
               name="sigFormula",
               arguments = list(label = "exitShort",
                    formula = "zScore < -1",
                    cross = TRUE),
               label = "exitShort")

#Trading rules based on signals
strategy.st <- add.rule(strategy = strategy.st,
     name = "ruleSignal",
     arguments = list(sigcol = "enterLong",
               sigval = TRUE,
               orderqty = 15,
               ordertype = "market",
               orderside = "long",
               replace = FALSE,
               threshold = NULL),
     type = "enter")

strategy.st <- add.rule(strategy = strategy.st,
     name = "ruleSignal",
     arguments = list(sigcol = "exitLong",
               sigval = TRUE,
               orderqty = "all",
               ordertype = "market",
               orderside = "long",
               replace = FALSE,
               threshold = NULL),
     type = "exit")

strategy.st <- add.rule(strategy = strategy.st,
     name = "ruleSignal",
     arguments = list(sigcol = "enterShort",
               sigval = TRUE,
               orderqty = -15,
               ordertype = "market",
               orderside = "short",
               replace = FALSE,
               threshold = NULL),
```

```
        type = "enter")

strategy.st <- add.rule(strategy = strategy.st,
        name = "ruleSignal",
        arguments = list(sigcol = "exitShort",
                    sigval = TRUE,
                    orderqty = "all",
                    ordertype = "market",
                    orderside = "short",
                    replace = FALSE,
                    threshold = NULL),
        type = "exit")

#apply strategy
t1 <- Sys.time()
out <- applyStrategy(strategy=strategy.st,portfolios=portfolio.st)
t2 <- Sys.time()
print(t2-t1)

# Time difference of 5.136077 secs from last execution in Macbook Pro 2019

#set up analytics
updatePortf(portfolio.st)
dateRange <- time(getPortfolio(portfolio.st)$summary)[-1]
updateAcct(portfolio.st,dateRange)
updateEndEq(account.st)

#Stats
tStats <- tradeStats(Portfolios = portfolio.st, use="trades", inclZeroDays=FALSE)
tStats[,4:ncol(tStats)] <- round(tStats[,4:ncol(tStats)], 2)
print(data.frame(t(tStats[,-c(1,2)])))

######### Backtest performance results ############
#Num.Txns           344.00
#Num.Trades         134.00
#Net.Trading.PL     31144.19
#Avg.Trade.PL       232.42
#Med.Trade.PL       194.50
#Largest.Winner     913.82
#Largest.Loser      -86.01
#Gross.Profits      31266.42
#Gross.Losses       -122.23
#Std.Dev.Trade.PL   181.57
```

#Std.Err.Trade.PL 15.69
#Percent.Positive 97.76
#Percent.Negative 2.24
#Profit.Factor 255.81
#Avg.Win.Trade 238.67
#Med.Win.Trade 201.01
#Avg.Losing.Trade -40.74
#Med.Losing.Trade -21.17
#Avg.Daily.PL 232.95
#Med.Daily.PL 195.85
#Std.Dev.Daily.PL 182.15
#Std.Err.Daily.PL 15.79
#Ann.Sharpe 20.30
#Max.Drawdown -1269.75
#Profit.To.Max.Draw 24.53
#Avg.WinLoss.Ratio 5.86
#Med.WinLoss.Ratio 9.49
#Max.Equity 31144.69
#Min.Equity -13.74
#End.Equity 31144.19

#Averages
(aggPF <- sum(tStats$Gross.Profits)/-sum(tStats$Gross.Losses))
#Average profits 255.7999
(aggCorrect <- mean(tStats$Percent.Positive))
#Average positive trades 97.76
(numTrades <- sum(tStats$Num.Trades))
#Number of trades 134
(meanAvgWLR <- mean(tStats$Avg.WinLoss.Ratio))
#Average winLoss ratio 5.86

#portfolio cash PL
portPL <- .blotter$portfolio.EWA_EWC$summary$Net.Trading.PL

Sharpe Ratio
(SharpeRatio.annualized(portPL, geometric=FALSE))
Net.Trading.PL
#####Annualized Sharpe Ratio (Rf=0%) 3.070985

Chart Performance strategy vs. SPY
instRets <- PortfReturns(account.st)

```
portfRets <- xts(rowMeans(instRets)*ncol(instRets), order.by=index(instRets))

cumPortfRets <- cumprod(1+portfRets)
firstNonZeroDay <- index(portfRets)[min(which(portfRets!=0))]
getSymbols("SPY", from=firstNonZeroDay, to="2019-12-31")
SPYrets <- diff(log(Cl(SPY)))[-1]
cumSPYrets <- cumprod(1+SPYrets)
comparison <- cbind(cumPortfRets, cumSPYrets)
colnames(comparison) <- c("strategy", "SPY")
chart.TimeSeries(comparison, legend.loc = "topleft", colorset = c("green","red"))
```

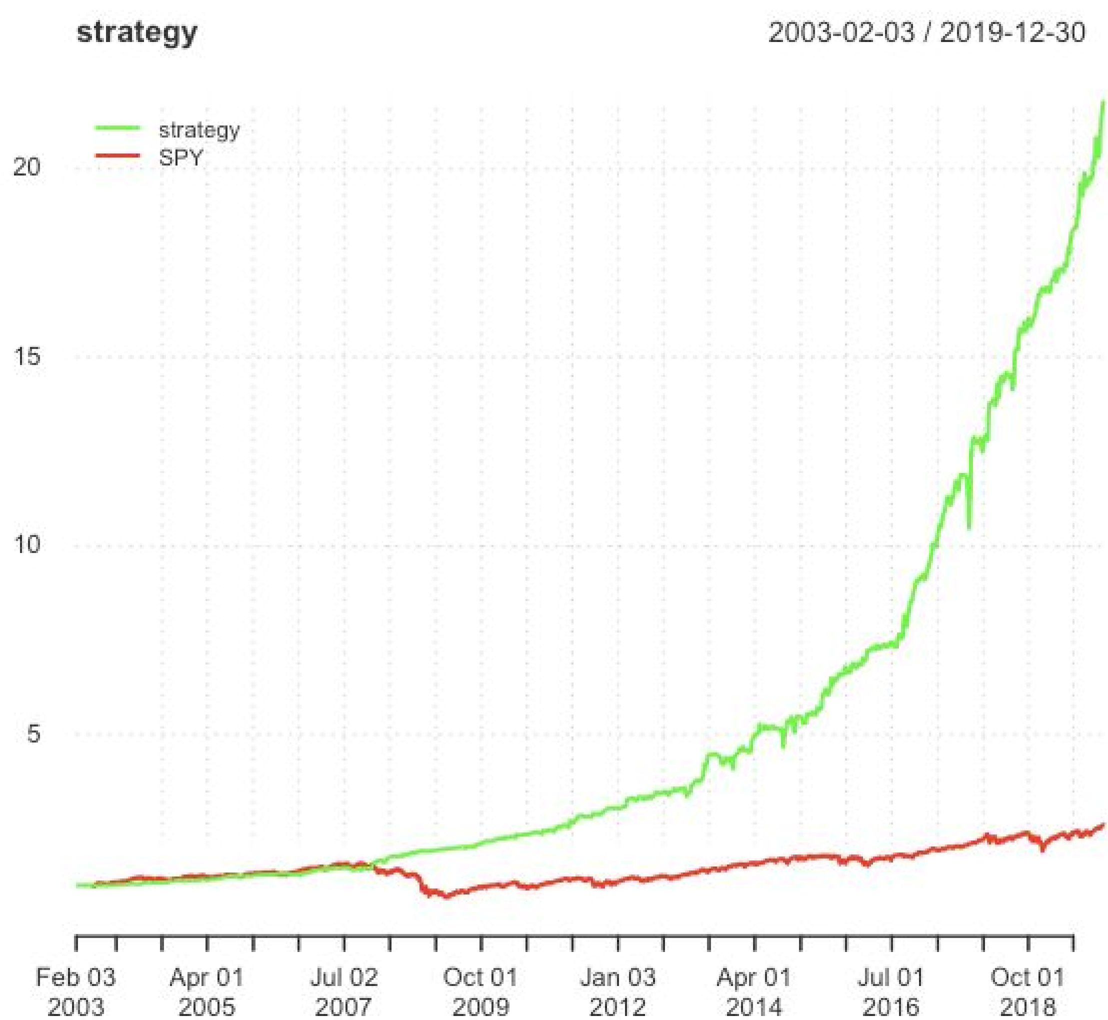
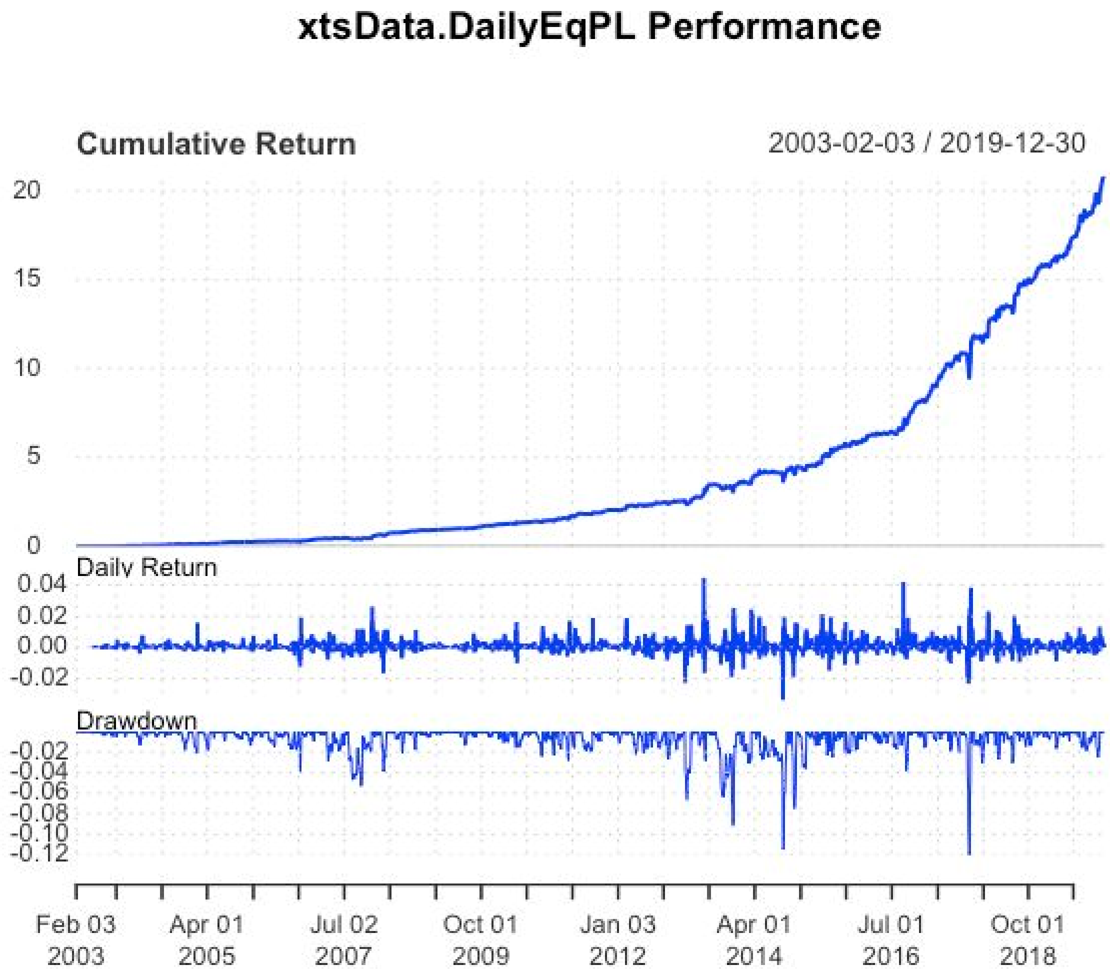


Chart Daily Positions

```
rets <- PortfReturns(Account = account.st)
rownames(rets) <- NULL
charts.PerformanceSummary(rets, colorset = bluefocus)
```


xtsData.DailyEqPL Performance
Cumulative Return
2003-02-03 / 2019-12-30
20
15
10
5
0
Daily Return
0.04
0.02
0.00
-0.02
Drawdown
-0.02
-0.04
-0.06
-0.08
-0.10
-0.12
Feb 03 2003
Apr 01 2005
Jul 02 2007
Oct 01 2009
Jan 03 2012
Apr 01 2014
Jul 01 2016
Oct 01 2018

Strategy Improvements - Accounting for Transaction Costs

To make our algorithmic trading process more realistic, one of the most important aspects to consider is the consideration of how transaction costs would affect the strategies. As much of the research shows, the profitability of trading strategies can be linked to the correct inclusion of transaction costs. This comes out of the consideration that, when dealing with such strategies, they involve far more trades than a simple traditional long-only approach. Therefore, there might be the risk that transaction costs could eliminate any excess returns forecasted by the trading strategies.

We could then improve our trading strategies by looking at a case outlined by Ernest Chan in book, “Quantitative Trading: How to Build Your Own Algorithmic Trading Business”, whereby he looks at two cases within a simple mean-reverting model taken from a paper by Amir Khandani and Andrew Lo at MIT. Their strategy was built on going long on those stocks which had the worst previous one-day returns and shorting the ones who had the best previous one-day returns. One thing we can note here, is that their strategy worked quite well in the presence of the assumption of “no transaction costs”. Therefore, we can impose the condition of subtracting about 5 basis points per trade and see how the outcome fares with regards to the situation with no transaction costs.

Here, we are using the file input from the S&P 500 stock universe.

```
%% MATLAB Code

Clear;

inputFile='Export.txt';
outputFile='SPX 20071123';

[mysym, mytday, myop, myhi, mylo, mycl, myvol]=...
textread(inputFile, '%s %u %f %f %f %f %u', …
'delimiter', ',');

% Since the single file consists of many symbols,
% we need to find the unique set of symbols. stocks=unique(mysym);
% Since the single file consists of many repeating set % of dates for different symbols, we need
% to find the unique set of dates.

tday=unique(mytday);

op=NaN(length(tday), length(stocks));
hi=NaN(length(tday), length(stocks));
```

```
lo=NaN(length(tday), length(stocks));
cl=NaN(length(tday), length(stocks));
vol=NaN(length(tday), length(stocks));

for s=1:length(stocks)
        stk=stocks{s};

% find the locations (indices) of the data with
% the current symbol.
idxA=strmatch(stk, mysym, 'exact');

% find the locations (indices) of the data with
 % the current set of dates.
[foo, idxtA, idxtB]=intersect(mytday(idxA), tday);

% Extract the set of prices for the current symbol
% from the downloaded data.

op(idxtB, s)=myop(idxA(idxtA));
hi(idxtB, s)=myhi(idxA(idxtA));
 lo(idxtB, s)=mylo(idxA(idxtA));
cl(idxtB, s)=mycl(idxA(idxtA));
 vol(idxtB, s)=myvol(idxA(idxtA));
end

save(outputFile, 'tday', 'stocks', 'op', 'hi', ... 'lo', 'cl', 'vol');
```

Case 1. Backtest the strategy in the absence of transaction costs:

```
clear;

startDate=20060101;
endDate=20061231;

load('SPX 20071123', 'tday', 'stocks', 'cl');

% daily returns
dailyret=(cl-lag1(cl))./lag1(cl);


% equal weighted market index return
marketDailyret=smartmean(dailyret, 2);
```

```
% weight of a stock is proportional to the negative
% distance to the market index.
weights=...
-(dailyret-repmat(marketDailyret,[1 size(dailyret,2)]))./ repmat(smartsum(isfinite(cl), 2), ... [1
size(dailyret, 2)]);

% those stocks that do not have valid prices or
% daily returns are excluded.

weights(~isfinite(cl) | ~isfinite(lag1(cl)))=0;
dailypnl=smartsum(lag1(weights).*dailyret, 2);

% remove pnl outside of our dates of interest
dailypnl(tday < startDate | tday > endDate) = [];

% Sharpe ratio should be about 0.25
sharpe=... sqrt(252)*smartmean(dailypnl, 1)/smartstd(dailypnl, 1)
```

Results case 1:

The original paper posted by Amir Khandani and Andrew Lo lead to obtain a Sharpe ratio of 4.47, however, following the implementation done by Ernest Chan of the strategy we can see that the Sharpe ratio has now declined at 0.25 for the period in question (2006). This is because the backtest has been performed on a universe of large market capitalization stocks on the S&P 500, instead of small and microcap stocks used in the original paper.

```
function y = smartsum(x, dim)

%y = smartsum(x, dim)
%Sum along dimension dim, ignoring NaN.

hasData=isfinite(x);
x(~hasData)=0;
y=sum(x,dim);
y(all(~hasData, dim))=NaN;

"smartmean.m"
function y = smartmean(x, dim)

% y = smartmean(x, dim)
% Mean value along dimension dim, ignoring NaN.

hasData=isfinite(x);
```

```
x(~hasData)=0;
y=sum(x,dim)./sum(hasData, dim);
y(all(~hasData, dim))=NaN; % set y to NaN if all entries are NaN's.

"smartstd.m"
function y = smartstd(x, dim)

%y = smartstd(x, dim)
% std along dimension dim, ignoring NaN and Inf

hasData=isfinite(x);
x(~hasData)=0;
y=std(x);
y(all(~hasData, dim))=NaN;
```

Case 2. Considering Transaction Costs: Let's deduct a 5-basis point transaction cost per trade

```
% daily pnl with transaction costs deducted
onewaytcost=0.0005; % assume 5 basis points

% remove weights outside of our dates of interest
weights(tday < startDate | tday > endDate, :) = [];

% transaction costs are only incurred when
% the weights change
dailypnlminustcost=...
dailypnl - smartsum(abs(weights-lag1(weights)), 2).* onewaytcost;

% Sharpe ratio should be about -3.19
sharpeminustcost=... sqrt(252)*smartmean(dailypnlminustcost, 1)/...
smartstd(dailypnlminustcost, 1)
```

Results Case 2: As we can see, if we adjust for the existence of transaction costs the Sharpe ratio plummets to -3.19 indicating that the strategy now is largely unprofitable.

Conclusion

Pairs trading strategy based on the divergence and convergence movements of stochastic trends in a pair of stocks' prices which usually belong to the same industrial group. In practice, we will choose two stocks or other financial instruments that are highly positively correlated (Spearman's Rho) and cointegrated. The idea of this strategy is that when one stock goes up, the other will go down, they will re-converge to the same stochastics trend in the end. To take profit, we will open a long position on the stock that they believe will go up and a short position on the stock that they believe will go down. Usually, the long stock will be underperforming at the time the position is opened and the short stock will be overperforming.

In this work, R and Python were used to create the system trading based on the pair trading strategy. In the beginning, we chose to study the pair of Pepsi and Coca-Cola stock prices using R. The data was downloaded from Yahoo finance. The test for the correlation yielded a highly positive correlation at 0.6816855. This refers to the same linear stochastics trends of both stock prices. However, the p value from the tests of cointegration (Engle-Granger, Dickey-Fuller test or Phillips-Perron) yielded a large p rejecting the null hypothesis of cointegration (Non stationary spread). Although they are highly positive collelated, they are not suitable to put onto the pair trading they are not cointegrated.

The second pair we studied is a pair of ETFs: iShares MSCI Australia and iShares MSCI Canada (EWA & EWC). Again, we used the data from Yahoo finance, but this time we implement coding by Python. We also tested correlation and cointegration similar to the first pair. The results indicated that all tests were valid for positive correlation and cointegration. Therefore, we chose to put the pair EWA and EWC into the investigating algorithm.

We created algorithmic trading based on pair trading between EWA and EWC by R. A prediction based on rolling linear regression of 21 days. The z-score was used as an indicator for making decisions short or long order. It was calculated by the formula, z = (x -mu)/sigma, where x is spread between stock's prices, mu is the average spread over 21 days and and sigma is the standard deviation of spread. We short the outperforming stock and to long the underperforming one when $z < -1$ and do it in the opposite way when $z > 1$. We performed the backtesting from 2003-01-01 until 2019-12-31. The performance of the proposed strategy was compared to the SPY. The result shows that the proposed strategy gives more cumulative profit than SPY.

We also proposed the ways of improvement algorithm by making it more realistic. A transaction costs were considered as it would affect the strategies in the real application. The S&P 500 were chosen as a case study. In this section, we used Matlab for coding and calculation. To compare the effect on strategy by the transition costs, we illustrated two cases of the simulation. One is the case of absence of transaction costs and another is including transaction costs. The backtesting shows that, in the first case, after we weighted the backtest with large capitalization stocks, instead of small and microcap stocks, in the S&P 500 the Sharpe ratio initially obtained in the paper the analysis is based on was way too high as the results obtained indicate a Sharpe of 0.25. Then, by additionally constraining the trading process by including a 5-basis point per trade deduction, the Sharpe ratio falls to negative territory indicating that a trading strategy is only as profitable if you take into account the full picture of the markets and market microstructures.

Furthermore, theoretically, we can also improve the strategy using more robust statistical measures for correlation and more robust regression. The reason we should improve it this way is that Spearman Rho's is symmetric, non-robust and linear statistical measure. It is difficult to detect the correlation between the data in the real world.

References

- Ernest Chan - “Algorithmic Trading: Winning Strategies and Their Rationale”
- Kanamura, Takashi; Rachev, Svetlozar; Fabozzi, FranK (5 July 2008). "The Application of Pairs Trading to Energy Futures Markets" (PDF). Karlsruhe Institute of Technology. Retrieved 20 January 2015.
- Rad, Hossein; Low, Rand Kwong Yew; Faff, Robert (2016-04-27). "The profitability of pairs trading strategies: distance, cointegration and copula methods". Quantitative Finance. 0 (10): 1541–1558.
- Lowenstein, Roger (2000). When genius failed : the rise and fall of Long-Term Capital Management (1 ed.). New York: Random House. ISBN 978-0-375-50317-7.
- "Co-integration Trading Strategy". Bullmen Binary. Retrieved 20 January 2015.
- Mahdavi Damghani, Babak (2013). "The Non-Misleading Value of Inferred Correlation: An Introduction to the Cointelation Model". Wilmott. 2013 (1): 50–61.
- S. Mudchanatongsuk, J. A. Primbs and W. Wong: "Optimal Pairs Trading: A Stochastic Control Approach". Proceedings of the American Control Conference, 2008.
- "A New Approach to Modeling and Estimation for Pairs Trading". Monash University, Working Paper.
- C. Alexander: "Market Models: A Guide to Financial Data Analysis". Wiley, 2001.
- Bookstaber, Richard. A Demon Of Our Own Design, p. 186. Wiley, 2006.
- A. D. Schmidt: "Pairs Trading - A Cointegration Approach". University of Sydney, 2008.
- G. Vidyamurthy: "Pairs trading: quantitative methods and analysis". Wiley, 2004.
- O'Hara, Maureen; Lopez De Prado, Marcos; Easley, David (2011), "Easley, D., M. López de Prado, M. O'Hara: The Microstructure of the 'Flash Crash': Flow Toxicity, Liquidity Crashes and the Probability of Informed Trading", The Journal of Portfolio Management, Vol. 37, No. 2, pp. 118–128, Winter,
- Lauricella, Tom, and McKay, Peter A. "Dow Takes a Harrowing 1,010.14-Point Trip," Online Wall Street Journal, May 7, 2010. Retrieved May 9, 2010
- TECHNICAL COMMITTEE OF THE INTERNATIONAL ORGANIZATION OF SECURITIES COMMISSIONS (July 2011), "Regulatory Issues Raised by the Impact of Technological Changes on Market Integrity and Efficiency" (PDF), IOSCO Technical Committee, retrieved July 12, 2011
- Hendershott, Terrence, Charles M. Jones, and Albert J. Menkveld. (2010), "Does Algorithmic Trading Improve Liquidity?", Journal of Finance
- Brown, Brian (2010). Chasing the Same Signals: How Black-Box Trading Influences Stock Markets from Wall Street to Shanghai. Singapore: John Wiley & Sons.
- "Agent-Human Interactions in the Continuous Double Auction" (PDF), IBM T.J.Watson Research Center, August 2001
- "Algorithmic trading, Ahead of the tape", The Economist, 383 (June 23, 2007), p. 85, June 21, 2007

- "High-Frequency Firms Tripled Trades in Stock Rout, Wedbush Says". Bloomberg/Financial Advisor. August 12, 2011. Retrieved March 26, 2013.
- Farmer, J. Done (November 1999). "Physicists attempt to scale the ivory towers of finance". Computing in Science & Engineering.
- Bigiotti, Alessandro; Navarra, Alfredo (October 19, 2018), "Optimizing Automated Trading Systems", Advances in Intelligent Systems and Computing, Springer International Publishing, pp. 254–261
- Amery, Paul (November 11, 2010). "Know Your Enemy". IndexUniverse.eu. Retrieved March 26, 2013.
- Rob Curren, Watch Out for Sharks in Dark Pools, The Wall Street Journal, August 19, 2008, p. c5. Available at WSJ Blogs retrieved August 19, 2008
- Brown, Brian (2010). Chasing the Same Signals: How Black-Box Trading Influences Stock Markets from Wall Street to Shanghai. Singapore: John Wiley & Sons.
- Gjerstad, Steven; Dickhaut, John (January 1998), "Price Formation in Double Auctions, Games and Economic Behavior, 22(1):1–29", S. Gjerstad and J. Dickhaut
- Rekenthaler, John (February–March 2011). "The Weighting Game, and Other Puzzles of Indexing" (PDF). Morningstar Advisor. pp. 52–56 [56]. Archived from the original (PDF) on July 29, 2013. Retrieved March 26, 2013.
- The Associated Press, July 2, 2007 Citigroup to expand electronic trading capabilities by buying Automated Trading Desk, accessed July 4, 2007
- Farmer, J. Done (November 1999). "Physicists attempt to scale the ivory towers of finance". Computing in Science & Engineering.
- Hendershott, Terrence, Charles M. Jones, and Albert J. Menkveld. (2010), "Does Algorithmic Trading Improve Liquidity?", Journal of Finance
- Cointegrated ETF Pairs Part II blog post https://quantoisseur.com/2017/01/20/cointegrated-etf-pairs-part-ii/
- Bowen, D. High Frequency Equity Pairs Trading: Transaction Costs, Speed of Execution and Patterns in Returns, March 2010
- Chan, E. Quantitative Trading: How to Build Your Own Algorithmic Trading Business, 2009