Assignment 3: Pairs Trading: Simulation vs. Reality

I certify that I have acknowledged the source of all work in this assignment that is not my own original work, and that I have not copied or modified the work of others without attribution.

Name: Date:

I. Introduction

In order to make profit in stock market, investors have developed many different strategies. While some investors buy stocks based on information regarding particular companies, others utilize strategies that try to profit based on understanding of the stock market and its behaviors in general. One of such strategies in stock trades is pairs trading.

Pairs trading (see http://en.wikipedia.org/wiki/Pairs_trade) is a strategy for buying and selling stocks developed at Morgan Stanley in the 1980s. The key idea is that the movement of the ratio away from its historical average represents an opportunity to make money. For example,we should sell stock 1 and buy stock 2, when stock 1, comparing to stock 2, is doing better than it usually does. The moment we sell stock 1 and buy stock 2 is called "opening a position." When is the closing position, after which we shall buy stock 1 again and sell stock 2? When the ratio returns to its historical average,

Investors chose two highly related stocks and trade based on the performance of two companies. Since almost all the companies have ever split the stocks, in the research, I am going to use price of adjust close. Because we can't predict future, mean and standard deviation extracted from training datum apply to test data.

In the following research, we will look into pairs trading in more details, explore the modern implications of such strategy, and conduct a simulation study to explore possible relation and connection between profit and different variables associated with stock selections in pairs trading.

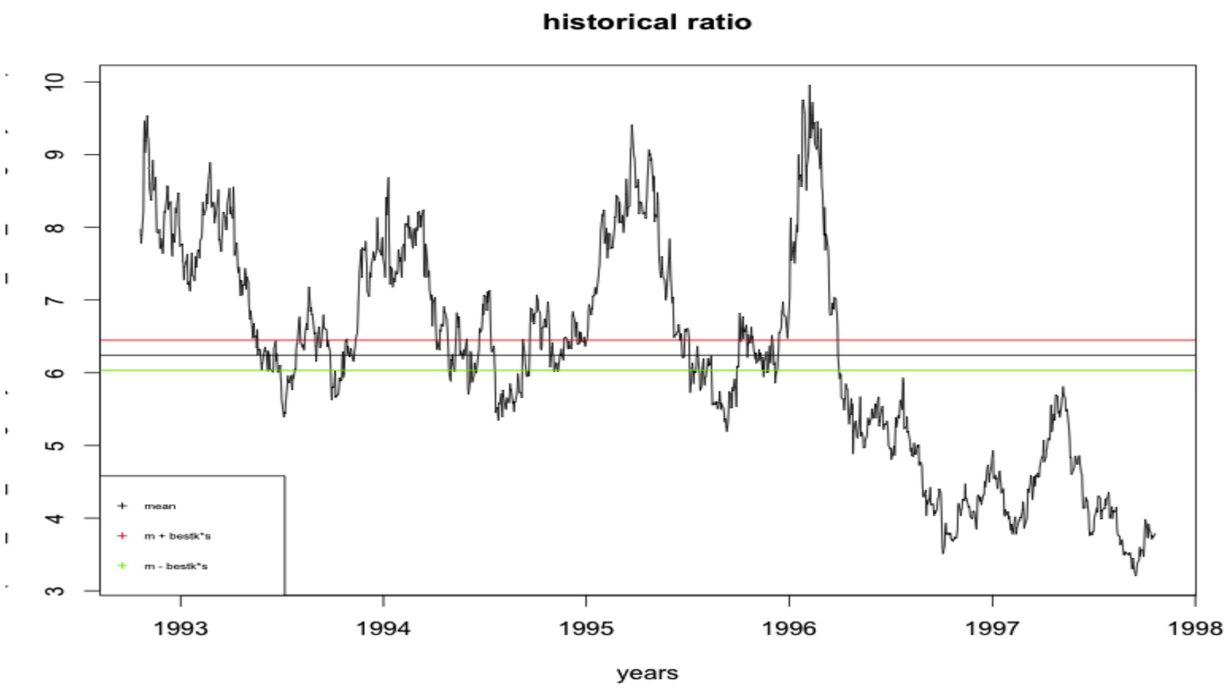
II. Example of Real Data

First of all, I choose to two real stocks, McDonald's(MCD) and Starbucks(SBUX), to calculate the profit based on pairs trading model. McDonald's and Starbucks, both providing coffee, are main competitors to each other. Both companies have their own strategies to win the market. For example, McDonald's provides a bigger variety of cheap food for customers, while customers can enjoy surfing the internet and eat higher quality food at Starbucks. The datum are downloaded from Yahoo Finance. When I looked at the adjust close price of two stocks, in the period from 1992-10-21 to 2013-10-21, the correlation is almost 85%, which could be considered as highly related stocks.

Panel(a) shows the prices change during 1992-10-21 to 1997-10-21. The historical prices is the training data, which is used to find the best k and biggest profit during these years. The x-axis represents years and the y-axis represents the adjust prices of two stocks. Black curve models the trend of MCD, and the red line is the trend of SBUX. Since the stocks prices of MCD or SBUX has a big gap, I adjust the index of SBUX.

Panel(b) shows the ratio change during 1992-10-21 to 1997-10-21. Actually, the first date of the datum is used as the first open position, meaning MCD is underperforming relative to SBUX. Therefore we buy (1/price of SBUX) units of SBUX and sell (1/price of MCD) units of MCD. Since our last position is an opening position, we don't calculate the profit starting from last open position. It means that for every $1-trade we engaged in, we would have profited 0.3224826 by the end of this time period with a transaction cost of 0.001. In the training period(1992-10-21 to 1997-10-21), the best k which returns the best total profit 0.3224826 is 0.14. (best k = 0.14 best_total_profit = 0.3224826). There is a transaction fee (p) of 0.001 every time, totally 2*0.001, we trade which has been subtracted.

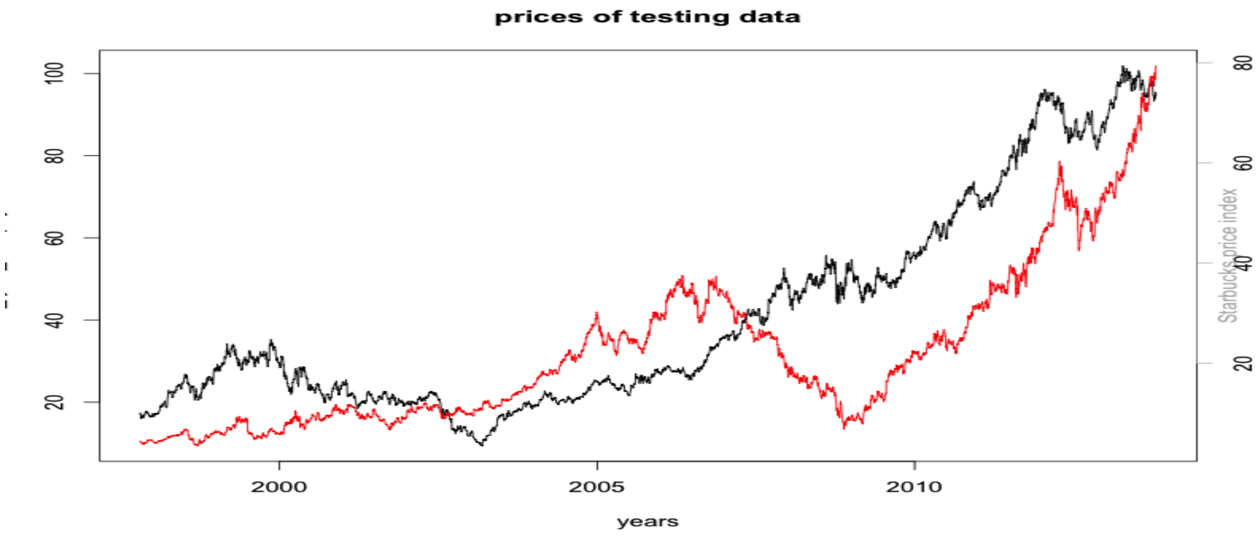
Panel(a)



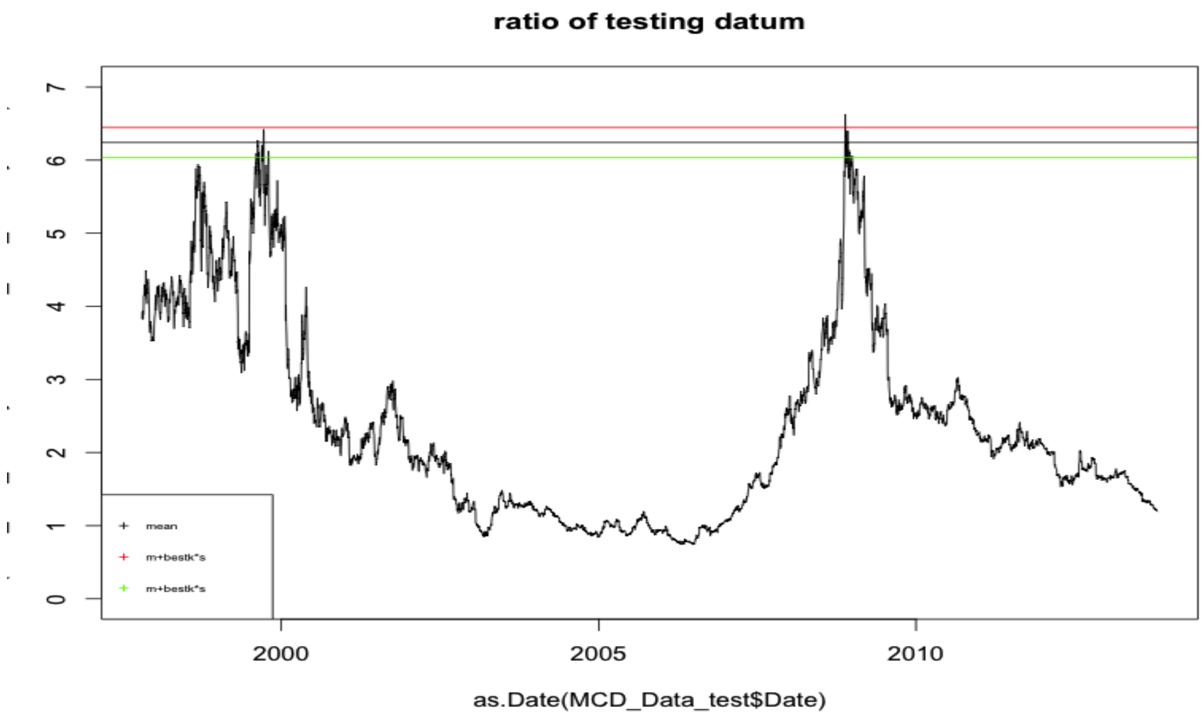
Panel(b)

Panel(c) shows the prices change during 1997-10-21 to 2013-10-21. The x-axis represents years and the y-axis represents the adjust prices of two stocks. Black curve models the trend of MCD, and the red line is the trend of SBUX. Since the stocks prices of MCD or SBUX has a big gap, I adjust the index of SBUX.

Panel(d) shows the ratio change during 1997-10-21 to 2013-10-21. Actually, the first date of the datum is used as the first open position, meaning SBUX is underperforming relative to MCD. Therefore we buy (1/price of MCD) units of MCD and sell (1/price of SBUX) units of SBUX. Since our last position is an opening position, we don't calculate the profit starting from last open position. It means that for every $1-trade we engaged in, we would have lost 0.09716028 by the end of this time period with a transaction cost of 0.001.



Panel(c)



Panel(d)

II. Simulation Study

The simulation involves generating stock data with different properties and calculating the profits using simulated data.

This model incorporates the following variables about the virtual stock picks, or sequences:

cp(correlation parameter) : a value between zero and one that controls how correlated the sequences are with each other.

Slope 1 and slope 2: a slope that is assigned to each stock price sequence; it controls the change in stock prices over time.

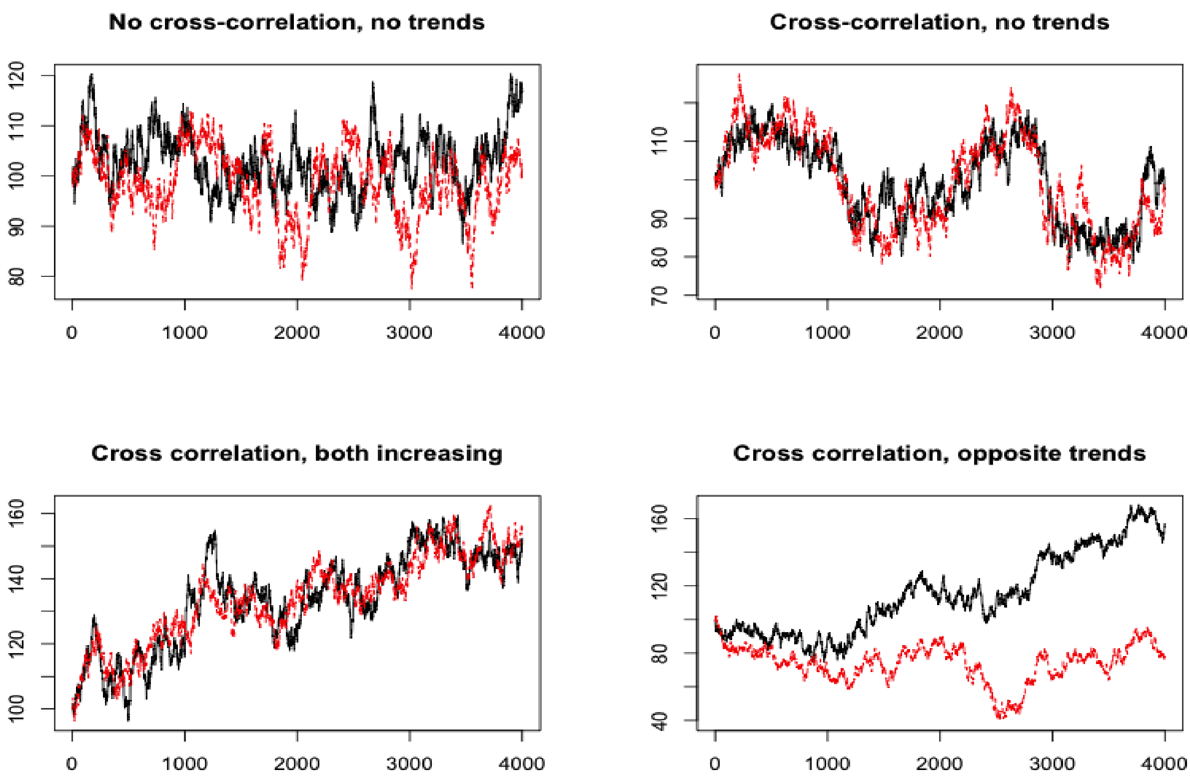
beta0: sequence ' s base value, which is set to $100 throughout the simulation.

k: choice of k.

p: trading cost, which is set to 0.001 throughout the simulation

First of all, I wanna study this four particular situations

- No cross-correlation, no trends: cp = 0, slope(1) = slope(2) = 0
- Cross-correlation, no trends: cp = 0.9, slope(1) = slope(2) = 0 11
- Cross correlation, both increasing: cp = 0.9, slope(1) = slope(2) = 0.01
- Cross correlation, opposite trends: cp = 0.9, slope(1) = 0.01, slope(2) = −0.01



As shown above, when there is no cross-correlation between the two stock picks and no trend for price change over time, the values of the two stocks when plotted together form random bands. The top right shows the price plot with the two stocks closely correlated (psi = 0.9), and no clear trend for price change over time. The bottom left is when the two stock picks are closely correlated and have increasing price over time. The bottom right is when the stock picks are positively correlated but lead opposite price-change trends over time.

Secondly, I am going to study the following factors separately :

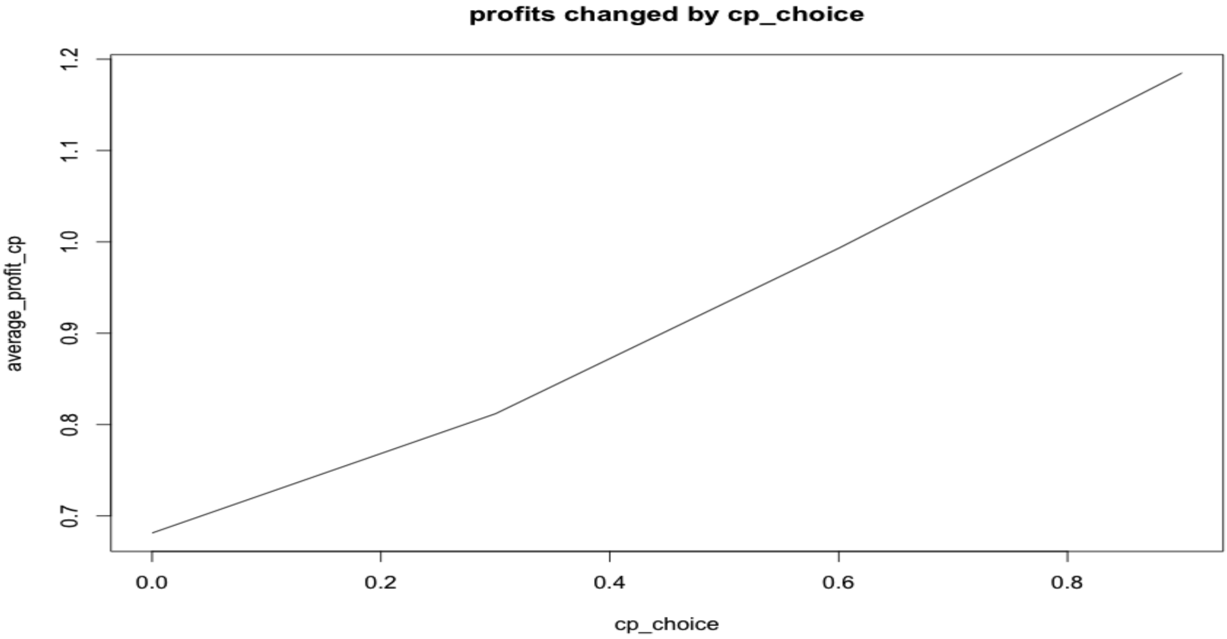
cp: correlation between the two sequences, variations of correlation parameter used: 0, 0.3, 0.6, 0.9

beta1: slopes of the two sequences in relation to time (0,0), (0.01, 0.01), (0.01, -0.01), (-0.01, -0.01)

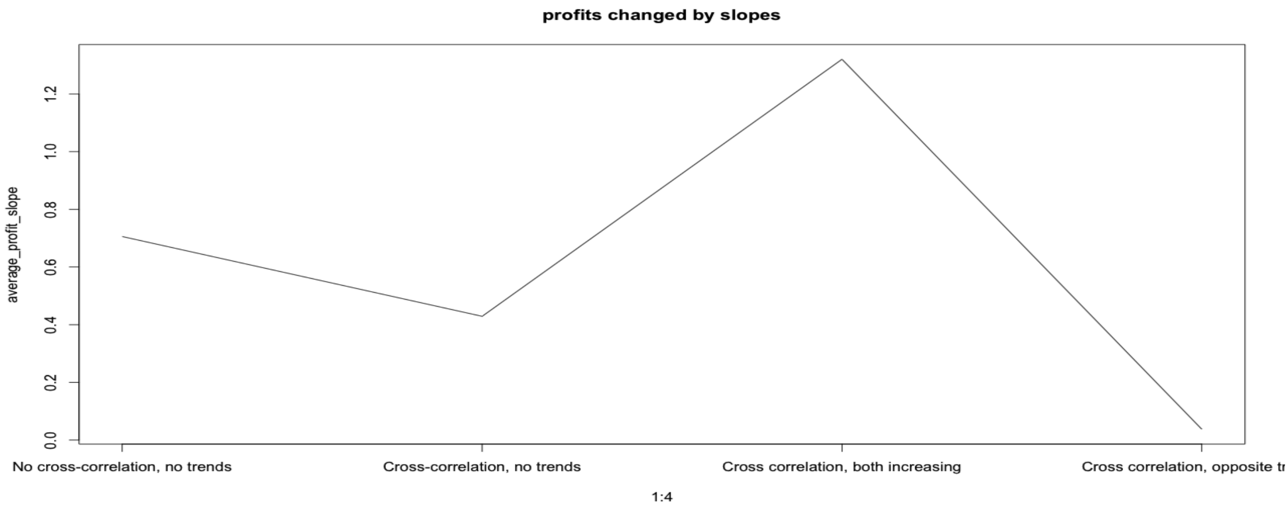
k: threshold value, variations of k used: seq(0, 4, by = 0.2)

cp:

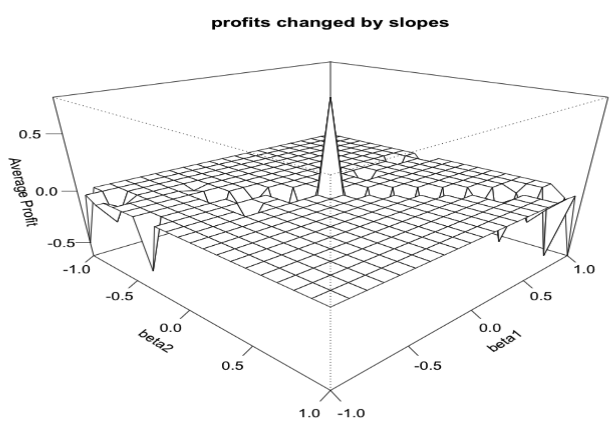
I calculate the simulation total profit of each correlation parameter, then repeat the calculation for 1000 times. Every time, the generated sequences are different, so take the mean of all observations. From the figure, we can know that, the total profit increases when correlation parameter becomes larger.



Slopes:

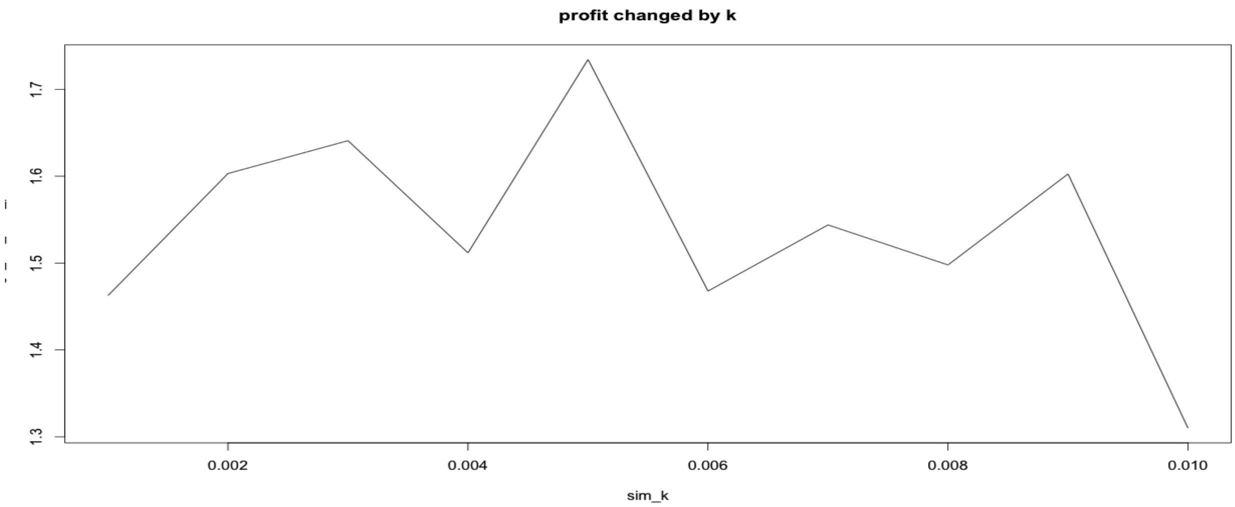


I calculate the simulation total profit of each set of slopes then repeat the calculation for 1000 times. The x-axis lists 4 possible situations of how the stocks will go. We can know that when the stocks are cross correlation and both increasing, we can get the biggest profit. However, if the stocks are cross correlation and have opposite trend, the profit is almost 0.

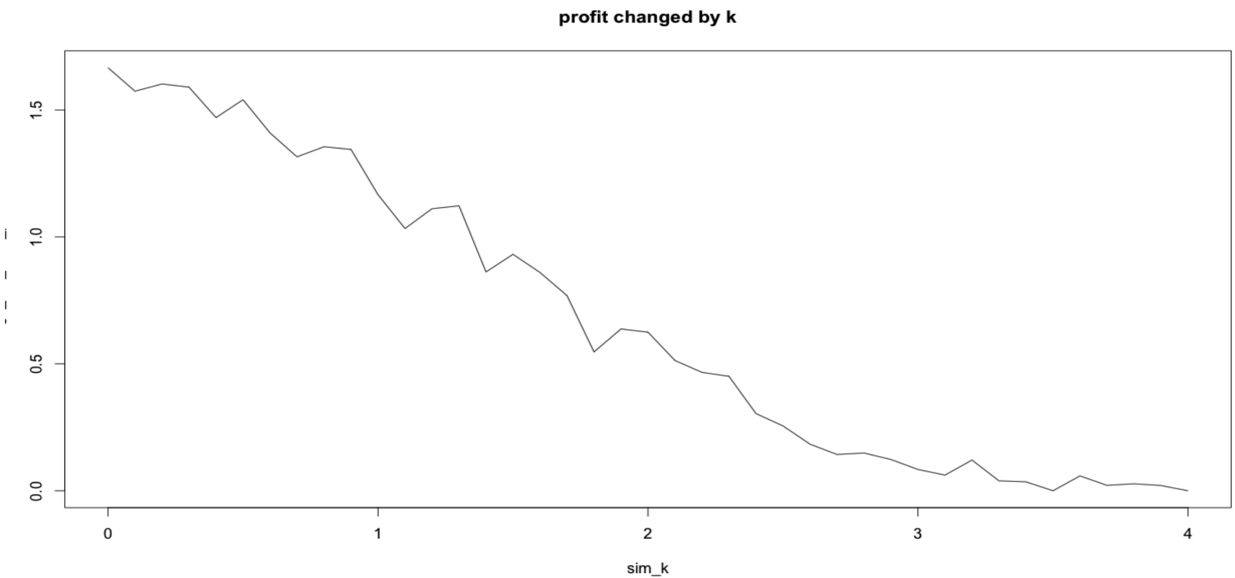


The relation also could be presented in 3D plot. The slopes are random numbers from -1 to 1. We can see that, when both slopes are 1.0, the profits meet the biggest. However, when the slopes are both -1.0, investor might lose money.

K:



The biggest profit happens between k=0.004 and 0.006, then the profit drops.



Conclusion:

From the simulation study, we can see that, correlation parameter, slopes and k are indeed the influenced factors.Formula in simulation study

$$\begin{aligned} X_t^{(1)} &= \rho X_{t-1}^{(1)} + \psi(1-\rho)X_{t-1}^{(2)} + \epsilon_t^{(1)} & \qquad Y_t^{(1)} &= \beta_0^{(1)} + \beta_1^{(1)} t + X_t^{(1)} \\ X_t^{(2)} &= \rho X_{t-1}^{(2)} + \psi(1-\rho)X_{t-1}^{(1)} + \epsilon_t^{(2)} & \qquad Y_t^{(2)} &= \beta_0^{(2)} + \beta_1^{(2)} t + X_t^{(2)}, \end{aligned}$$

From the real data, we can know that pairs trading is not a promised way to win profit. You can not totally based on the training data(historical prices and ratio) to predict how much investors are going to win in the future. In my example, MCD and SBUX, although the two stocks have strong correlation and investors get profit using pairs trading, in the period of 1992-1997, that does not guarantee the profit from 1998 to present. The result turns out that best k from training data give a negative profit later.

Appendix code

```
MCD_Data_original <- read.csv("/Users/hfj/Documents/Major/Statistics/STA 141/hw 3/MCD(1992,10,21-2013,10,21).csv")
SBUX_Data_original <- read.csv("/Users/hfj/Documents/Major/Statistics/STA 141/hw 3/SBUX(1992,10,21-2013,10,21).csv")

###The datum are listed from 2013-10-21 to 1992-10-21. First of all, we need to reverse the datum and make it start from 1992-10-21
n = nrow(MCD_Data_original)
MCD_Data = MCD_Data_original[rev(c(1:n)), ]
m = nrow(SBUX_Data_original)
SBUX_Data = SBUX_Data_original[rev(c(1:m)), ]

###The length of two stocks is different. There is a missing value in McDonald's stock. Therefore, we need to match them up to make sure they have the same length
match2stocks = match(as.Date(MCD_Data$Date),as.Date(SBUX_Data$Date))
MCD_Data <- MCD_Data[!is.na(match2stocks), ]
cor(SBUX_Data$Adj.Close, MCD_Data$Adj.Close)

###subset datum from 1992-10-21 to 1997-10-21
MCD_Data_training <- subset(MCD_Data, as.Date(MCD_Data$Date)<="1997-10-21")
SBUX_Data_training <- subset(SBUX_Data, as.Date(SBUX_Data$Date)<="1997-10-21")

###Historical Prices   (Since the stocks prices of MCD or SBUX has a big gap, I adjust the index of SBUX) the x-axis represents years and the y-axis represents the adjust prices of two stocks.
plot(MCD_Data_training$Adj.Close ~ as.Date(MCD_Data_training$Date),type = 'l',main = 'Historical Price')
par(new=T)
plot(SBUX_Data_training$Adj.Close ~ as.Date(SBUX_Data_training$Date), col = "red", type = 'l', xlab = ' ', xaxt='n', axes=F, ylab=' ')
mtext("Starbucks price index", side = 4, line = .5, col = "grey")
axis(4, col = "grey")


###Write a function to find open and position (pass coefficient of adjust close prices, mean of adjust close prices, standard deviation of adjust close prices, ratio of adjust close prices, a beginning number)
Find_trade_day = function(k,m,s,r,begin){
  r=r[-(1:begin)]
  w = c(which(r >= m + k*s), which(r <= m – k*s))
##if can't find any open position in the sequences then jump out of the function
  if(!any(w)){
    return(c())
    break
  }
##there are two situations to open the positions
  open_day <- min(w)
#sell stock 1 and buy stock 2
  if(r[open_day] >= m + k*s){
    closing = which(r[-(1:open_day)] <= m)
    n = length(closing)
```

```
    if(n == 0){
      close_day <- 9999
    }else{ close_day <- (min(closing)+open_day)}
#sell stock 2 and buy stock 1
  }else if(r[open_day] <= m - k*s){
    closing = which(r[-(1:open_day)] >= m)
    n = length(closing)
    if(n == 0){
      close_day <- 9999
    }else{close_day <- (min(closing)+open_day)}
  }
  ans = c(open_day,close_day)
  names(ans) = c("open", "close")
  ans + begin
}

###Within the function, total_profit of all the open and close position can be found. Calculate the
profit in each interval, then add up all together
trade1 = function(trade_days,m,k,s,r,stock1,stock2) {
  total_profit = 0
  for(trade_day in trade_days){
    #calculate the profits within the first open and close position
    if(r[trade_day["open"]] >= m + k*s){
      profit = (1-(1/stock1[trade_day["open"],"Adj.Close"] *
stock1[trade_day["close"],"Adj.Close"]))
      + ((1/stock2[trade_day["open"],"Adj.Close"] * stock2[trade_day["close"],"Adj.Close"])-1)-
2*0.01
      total_profit = total_profit + profit
    }
    if(r[trade_day["open"]] <= m - k*s){
      profit = (1-(1/stock2[trade_day["open"],"Adj.Close"] *
stock2[trade_day["close"],"Adj.Close"]))
      + ((1/stock1[trade_day["open"],"Adj.Close"] * stock1[trade_day["close"],"Adj.Close"])-1)-
2*0.01
      total_profit = total_profit + profit
    }
  }
  return(total_profit)
}



m <- mean(MCD_Data_training$Adj.Close/SBUX_Data_training$Adj.Close)
s <- sd(MCD_Data_training$Adj.Close/SBUX_Data_training$Adj.Close)
r <- as.numeric(MCD_Data_training$Adj.Close/SBUX_Data_training$Adj.Close)

###find k which best fits to training data(meet the max profits)
k_possible = seq(0.01, 4, by = 0.01)
bestk = Inf
best_total_profit = 0
for (k in k_possible) {
    begin <- 1
```

```
    trade_days = list()
    i = 1
    while(TRUE){
      junk = Find_trade_day(k,m,s,r,begin)
##Find_trade_day function might return null or infinite values. Get rid of them.
      if ( (sum(junk>=9999)) | is.null(junk)) {
        break
      }
      trade_days[[i]] = junk
      begin <- trade_days[[i]]["close"]
      i = i+1
    }
    training_profit = trade1(trade_days,m,k,s,r,MCD_Data_training,SBUX_Data_training)
    ##whenever find a profit is bigger, store it as the newest best total profit.
     if (training_profit > best_total_profit) {
      bestk = k
      best_total_profit = training_profit
    }
}

bestk = 0.14
best_total_profit = 0.3224826

#historical ratio
plot((MCD_Data_training$Adj.Close/SBUX_Data_training$Adj.Close) ~
as.Date(MCD_Data_training$Date),type = 'l',xlab = "years",main = "historical ratio")
abline(h = mean(MCD_Data_training$Adj.Close/SBUX_Data_training$Adj.Close))
abline(h = m + bestk*s, col = "red")
abline(h = m - bestk*s, col = "green")
legend("bottomleft", legend = c("mean","m + bestk*s","m - bestk*s"),col =
c("black","red","green"),cex = 0.5,pch=3)

###Use the datum from 1997-10-22 to 2013-10-21 as test datum to predict the profits in the
“future” (stay the mean and standard deviation same with training datum. Generate new ratio
vectors from test datum
MCD_Data_test <- subset(MCD_Data, as.Date(MCD_Data$Date)>"1997-10-21")
SBUX_Data_test <- subset(SBUX_Data, as.Date(SBUX_Data$Date)>"1997-10-21")
r_test <- as.numeric(MCD_Data_test$Adj.Close/SBUX_Data_test$Adj.Close)
begin <- 1
while(TRUE) {
  junk = Find_trade_day(bestk,m,s,r,begin)
  if ((sum(junk>=9999)) | is.null(junk)) {
    break
  }
  trade_days[[i]] = junk
  begin <- trade_days[[i]]["close"]
  i = i+1
}
test_total_profit = trade1(trade_days,m,bestk,s,r,MCD_Data_test,SBUX_Data_test)

###Plot: Prices of testing datum(1998-2013)
plot(MCD_Data_test$Adj.Close ~ as.Date(MCD_Data_test$Date), type = 'l', main = "prices of
```

```
testing data", xlab = " ")
par(new=T)
plot(SBUX_Data_test$Adj.Close ~ as.Date(SBUX_Data_test$Date), col = "red", type = 'l', xlab =
'years', xaxt='n', axes=F, ylab='adjust close price')
mtext("Starbucks price index", side = 4, line = .5, col = "grey")
axis(4, col = "grey")

###Plot: ratio of testing datum (1998-2013)
par(mfrow = c(1,1))
plot((MCD_Data_test$Adj.Close/SBUX_Data_test$Adj.Close) ~ as.Date(MCD_Data_test$Date),
type = 'l', ylim = range(c(0,7)), main = "ratio of testing datum")
abline(h = m)
abline(h = m + bestk * s, col = "red")
abline(h = m - bestk * s , col = "green")
legend("bottomleft", legend = c("mean","m+bestk*s","m+bestk*s"), col = c("black","red","green"),
cex = 0.5, pch = 3)

###Function: generate two stocks (put two sequences into a matrix)
stocksim = function(cp,slope1,slope2){
   X = matrix(0, nrow=4000, ncol=2)
   e1 = rnorm(4000,0,1)
   e2 = rnorm(4000,0,1)
   X[1,1] = e1[1]
   X[1,2] = e2[1]
   Y = matrix(0,nrow=4000,ncol=2)
   Y[1,1] = 100 + slope1*1 + X[1,1]
   Y[1,2] = 100 + slope2*1 + X[1,2]
  for(t in 2:4000){
     X[t,1] = 0.99*X[t-1,1]+cp*0.01*X[t-1,2] + e1[t]
     X[t,2] = 0.99*X[t-1,2]+cp*0.01*X[t-1,1] + e2[t]
     Y[t,1] = 100 + slope1*t + X[t,1]
     Y[t,2] = 100 + slope2*t + X[t,2]
  }
   return(Y)
}

###Observations about four different situations
par(mfrow = c(2,2))
case1 = stocksim(0,0,0)
matplot(case1,type = "l",main = "No cross-correlation, no trends")
case2 = stocksim(0.9,0,0)
matplot(case2,type = "l",main = "Cross-correlation, no trends")
case3 = stocksim(0.9,0.01,0.01)
matplot(case3,type = "l", main = "Cross correlation, both increasing")
case4 = stocksim(0.9,0.01,-0.01)
matplot(case4,type = "l", main = "Cross correlation, opposite trends")


###Function: subset training and test datum from the generated stocks; in the training data, find all
open and close positions; calculate the profit within one k; find the bestk; apply best k to test data;
get the total profit of test data
steps = function(Y){
```

```
  sim_training = Y[1:2000, ]
  sim_test = Y[2001:4000, ]
  r_sim_training <- sim_training[,1]/sim_training[,2]
  m_sim_training <- mean(r_sim_training)
  s_sim_training <- sd(r_sim_training)
  #use training data to find best k
  k_possible = seq(0.01, 4, by = 0.01)
  j = 1
  sim_training_profit = numeric()
  for (k in k_possible) {
    begin <- 1
    trade_days = list()
    i = 1
    while(TRUE){
      junk = Find_trade_day(k,m_sim_training,s_sim_training,r_sim_training,begin)
      if ( (sum(junk>=9999)) | is.null(junk)) {
        break
      }
      trade_days[[i]] = junk
      begin <- trade_days[[i]]["close"]
      i = i+1
    }
    sim_training_profit[j] =
trade2(trade_days,m_sim_training,k,s_sim_training,r_sim_training,sim_training[,1],sim_training[,2
])
    j = j+1
    #calculate the profit within one k
    sim_bestk = k_possible[which.max(sim_training_profit)]
  r_sim_test <- sim_test[,1]/sim_test[,2]
  sim_test_total_profit_sum = list()
  begin <- 1
  trade_days = list()
  i = 1
  #use the best k to find the profit in training data
  while(TRUE) {
    junk = Find_trade_day(sim_bestk,m_sim_training,s_sim_training,r_sim_test,begin)
    if ((sum(junk>=9999)) | is.null(junk)) {
      break
    }
    trade_days[[i]] = junk
    begin <- trade_days[[i]]["close"]
    i = i+1
  }
  sim_test_total_profit =
trade2(trade_days,m_sim_training,sim_bestk,s_sim_training,r_sim_test,sim_test[,1],sim_test[,2])
  return(sim_test_total_profit)
}

#Function: after getting bestk, calculate the total profit of test data
trade2 = function(trade_days,m,k,s,r,stock1,stock2) {
  total_profit = 0
  for(trade_day in trade_days){
```

```
    #calculate the profits within the first open and close position
    if(r[trade_day["open"]] >= m + k*s){
      profit = (1-(1/stock1[trade_day["open"]] * stock1[trade_day["close"]]))
      + ((1/stock2[trade_day["open"]] * stock2[trade_day["close"]])-1)-2*0.01
      total_profit = total_profit + profit
    }
    if(r[trade_day["open"]] <= m - k*s){
      profit = (1-(1/stock2[trade_day["open"]] * stock2[trade_day["close"]]))
      + ((1/stock1[trade_day["open"]] * stock1[trade_day["close"]])-1)-2*0.01
      total_profit = total_profit + profit
    }
  }
  return(total_profit)
}

###factor 1 correlation parameter (repeat B = 1000 times)
cp_choice = c(0,0.3,0.6,0.9)
average_profit_cp = numeric()
i <- 1
for(cp in cp_choice){
  average_profit_cp[i] = mean(replicate(1000,steps(stocksim(cp,0,0))))
  i = i +1
}
par(mfrow = c(1,1))
plot(cp_choice,average_profit_cp, type = "l",main = "profits changed by cp_choice")



###factor 2 slope 1 and slope 2 (repeat B times)
slope1 = c(0,0.01,-0.01,0.01)
slope2 = c(0,0.01,-0.01,-0.01)
average_profit_slope = numeric()
i <- 1
for(i in 1:4){
  average_profit_slope[i] = mean(replicate(1000,steps(stocksim(0,slope1[i],slope2[i]))))
  i = i + 1
}
plot(1:4,average_profit_slope, type = "l", ,main = "profits changed by slopes",xaxt = "n")
axis(1, at=1:4,c("No cross-correlation, no trends","Cross-correlation, no trends", "Cross correlation,
both increasing","Cross correlation, opposite trends"))

###present factor 2 in 3D plot (repeat 1000 times)
beta1 = seq(0, 1, by = 0.1)
beta2 = seq(0, 1, by = 0.1)

output = sapply(beta1, function(i) sapply(beta2, function(j)
mean(replicate(1000,steps(stocksim(0,i,j))))))
persp(beta2,beta1, output,theta = 45, phi = 25, expand = 0.7, ticktype = ‘detailed’, zlab = ‘Average
Profit’, main = "profits changed by slopes", xaxt = "n" )


#factor 3 choice of k
```

```
sim_k = seq(0.001,0.01,by = 0.001)
eachk = function(k){
  basic <- stocksim(0,0,0)
  r_k <- basic[ ,1]/basic[ ,2]
  s_k <- sd(r_k)
  m_k <- mean(r_k)
  o <- 1
  begin <- 1
  k_trade_days = list()
  while(TRUE){
    junk = Find_trade_day(k,m_k,s_k,r_k,begin)
    if ( (sum(junk>=9999)) | is.null(junk)) {
      break
    }
    k_trade_days[[o]] = junk
    begin <- k_trade_days[[o]]["close"]
    o = o+1
  }
  sim_trade_profit = trade2(k_trade_days,m_k,k,s_k,r_k,basic[,1],basic[,2])
  return(sim_trade_profit)
}
p <- 1
average_sim_trade_profit = numeric()
for(k in sim_k){
  average_sim_trade_profit[p] = mean(replicate(1000,eachk(k)))
  p = p + 1
}
plot(average_sim_trade_profit ~ sim_k , type = "l", main = "profit changed by k" )
```