Assignment 1

Philipp Epstein

10/14/2018

Introduction

This report is written for the 1st assignment of the Financial Trading Systems course 2018 at the Bond University. During the first weeks we learned how to use the Blotter Framework in R to implement a trading strategy, backtest the strategy on a set of instruments (stock, futures, etc.) and visualize meaningful results which give insights about the strategy. Moreover, we learned how to compare the strategy to another alternative (e.g. a buy and hold strategy). The trading idea, implementation and analysis and results are summarized in the following report.

Trading Idea

Smash Day by Larry Williams

The trading idea which is used for this report is based on the Smash Day strategy introduced by Larry Williams in his book "Long Term Secrets to short term trading" 2012. The basic theoretical foundation is set by the concept of autocorrelation. There are two autocorrelation principles which can be empirically observed. The positive autocorrelation which describes the fact that if we observe an upward movement it is more likely to observe an upward movement in the future. The same is valid for a negative trend. However, the negative autocorreleation can be observed on short term time periods. We can expect a price to bounce up again after a sharp down movement and vice versa. Larry William's Smash Day pattern is based on the observation of this autocorrelation behaviour. A Smash Day pattern is initialized when we observe a close which is lower than the previous day's low. This looks like a breakout to the downside. If the very next day the price moves opposite the Smash Day and trades above the high of the Smash Day this is according to Williams a buy signal. Williams explains this buy signal with a feeling of hurt felt by the public as a result of an unfulfilled breakout. The public would want to get back on track and the price responds with a reversal. The exact opposite is true for a Sell setup. The Smash Day is set up when we observe a close above the previous day's high. The sell signal is initiated when the price reverses immidiately the next day and falls below the smash day's low.

Own variation of the Smash Day trading strategy

In addition to the Smash Day pattern, I have incorporated another rule for the buy signal and 2 additional components for the sell signal. I want to make sure that I only enter long when we are observing a longlasting upward trend. In order to do this buy orders are only performed when the current price is above the exponential moving average of the last 200 days. After that the Smash Day pattern is checked. For the sell I incorporated two "harder" conditions before checking whether we observe a Smash Day (sell). The first is that a maximum holding period can be set in order to make sure that a position is not open for too long. The second sell case is when the price falls below the 200 day exponential moving average the position is liquidated. If both is not the case a check for the smash day pattern is performed. If we observe it, we sell the current position.

These variations are incorporated to reduce the risk of falling prices.

Implementation of the trading idea

Part A: Initialization

Step 1: General Setup

We have to clear the environment and initialize the most important packages that we will use.

```
# Clear Environment
rm(list=ls())

# Loading libraries
library(blotter)
library(INFT361Course)
```

Step 2: Setting the Variables

The variables set in the next section can be adjusted to test the strategy with different parameters.

```
# Set values:
startCapital <- 1e+6
transactionCost <- -20
daterange <- '2016::2018'
emaPeriod <- 200
maxHoldingPeriod <- 7

InstrumentDirectory <- "~/Desktop/R/DownloadedData/"
instrumentlist <- c("SAP.csv","DBK.csv")
BuyHoldDirectory <- InstrumentDirectory
BuyHoldInstrument <- "DAXEX.csv"

currency("EUR")
Sys.setenv(TZ="UTC")
initdate <- '1999-12-31'
startdate <- '2000-01-01'
enddate <- '2018-12-31'
portfolioname <- "Smash Day"
accountname <- portfolioname
```

Step 3: Presetup for plotting graphs

```
# Settings for graph
myTheme <- chart_theme()
myTheme$col$up.col <- 'lightblue'
myTheme$col$dn.col <- 'brown'
myTheme$col$dn.border <- 'lightgray'
myTheme$col$up.border <- 'lightgray'

# Concatenate string for EMA with input parameter
addEMAString <- paste("add_EMA(n=",emaPeriod,")",sep = "")
```

Step 3: Initializing the portfolio

The portfolio takes the instrumentlist which includes all the stocks we defined in Step 2.

```r
# Clear portfolio and Account
suppressWarnings(rm("account.Smash Day","portfolio.Smash Day","account.buyhold","portfolio.buyhold",pos

# Initialize Portfolio and Account
initPortf(portfolioname,instrumentlist,initDate=initdate,currency="EUR")
initAcct(accountname,portfolios=portfolioname,initDate=initdate,initEq=startCapital,currency="EUR")
```

Part B: Bar by bar processing

Step 1: Go through the data bar by bar

Loading the instrument, initializing it and adding the ema to the data. After that we go through the available data bar by bar and apply the buy and sell rules which are defined in the following part.

```r
# Go through the instrumentlist and perform the activities for all instruments in that list.
for (instrument in instrumentlist) {
    LoadCourseFile(InstrumentDirectory, instrument, debugme = TRUE, dates = daterange)

    # Initialize the instrument
    stock(instrument, currency = "EUR")

    # Load the XTS file
    symbol <- get(instrument)

    # Calculate the Exponential Moving Average
    ema <- EMA(symbol$Close, n=emaPeriod)

    # Merge the xts file with the Exponential Moving Average
    symbol <- merge(symbol,ema)
    assign(instrument,symbol)
  # Starting to go bar by bar through using a "for loop"
  for (i in (emaPeriod + 1):(nrow(symbol) - 1)) {
    # Dates
    CurrentDate <- time(symbol[i])
    TomorrowDate <- time(symbol[i + 1])

    # Today's variables
    CloseToday <- as.numeric(symbol[i, "Close"])
    EMA_today <- as.numeric(symbol[i, "EMA"])
    LowToday <- as.numeric(symbol[i, "Low"])
    HighToday <- as.numeric(symbol[i, "High"])

    # Yesterday's variables
    LowYesterday <- as.numeric(symbol[i - 1, "Low"])
    HighYesterday <- as.numeric(symbol[i - 1, "High"])

    # Tomorrow's variables
    OpenTomorrow <- as.numeric(symbol[i + 1, "Open"])
    LowTomorrow <- as.numeric(symbol[i + 1, "Low"])
    HighTomorrow <- as.numeric(symbol[i + 1, "High"])

    # Config
    Equity <- getEndEq(accountname, CurrentDate)
    Position <-
```

```
    getPosQty(portfolioname, Symbol = instrument, Date = CurrentDate)

  # Check whether we have a position
  if (Position == 0) {
    # Start checking BUY rules

    # Check whether we have a Smash Day (Long).
    # Smash Day (Long) is when Todays Close is below Yesterdays Low.
    if (CloseToday < LowYesterday) {
      # Smash Day (Long)

      #Check whether todays close is above today's EMA
      if (CloseToday > EMA_today) {

        # BUY RULE: If today was a smash day place a STOP BUY order at todays high price.
        # (Buy tomorrow for 'price >= todays high')

        # Simulate STOP BUY order:

        # Option 1 to check: Check whether the open price tomorrow is above today's high
        # and add the transaction tomorrow at tomorrows open price.

        # Option 2 to check: Check whether today's high was lower than tomorrows high
        # and add the transaction tomorrow at today's high price.

        # Check Option 1
        if (OpenTomorrow > HighToday) {
          # Don't trade at the day before the last day
          if (CurrentDate != time(symbol[nrow(symbol) - 1])) {
            # Calculate the buy quantity
            BuyQuantity <- as.numeric(trunc(Equity / OpenTomorrow))
            # Add transaction
            addTxn(
              portfolioname,
              Symbol = instrument,
              TxnDate = TomorrowDate ,
              TxnPrice = OpenTomorrow,
              TxnQty = BuyQuantity,
              TxnFees = transactionCost
            )
            # Store the bar at which we placed the transaction
            BuyBar <- i
          }

        } else {
          # Check Option 2
          if (HighToday < HighTomorrow) {
            # Don't trade at the day before the last day
            if (CurrentDate != time(symbol[nrow(symbol) - 1])) {
              # Calculate the buy quantity
              BuyQuantity <- as.numeric(trunc(Equity / HighToday))
              # Add transaction
              addTxn(
```

```
                portfolioname,
                Symbol = instrument,
                TxnDate = TomorrowDate ,
                TxnPrice = HighToday,
                TxnQty = BuyQuantity,
                TxnFees = transactionCost
              )
              # Store the bar at which we placed the transaction
              BuyBar <- i
            }
          }
        }
      }
    }
  } else {
    # We already have a position

    # Check the sell rules in the following order and sell at the
    # first condition which is satisfied.

    # Sell rules:
    # Rule 1: Sell if we hold the position longer than the specified
    # maximum holding period

    # Rule 2: Sell at tomorrow's opening price if the close price
    # today falls below the EMA

    # Rule 3: Sell if we meet the Smash Day (Short) requirements.
    # Today's close must be higher than yesterday's high

    # Rule 4: If no sell rule can be applied and we reach the
    # second last day. Sell at the last day.

    # Check Rule 1:
    if ((i - BuyBar) > maxHoldingPeriod) {
      # Place the sell transaction at todays close price
      addTxn(
        portfolioname,
        Symbol = instrument,
        TxnDate = CurrentDate,
        TxnPrice = as.numeric(symbol[i, "Close"]),
        TxnQty = -Position,
        TxnFees = transactionCost
      )

    } else {
      # Check Rule 2:
      if (as.numeric(symbol[i, "Close"]) < EMA_today) {
        # Place the sell transaction at tomorrow's open price
        addTxn(
          portfolioname,
          Symbol = instrument,
          TxnDate = time(symbol[i + 1]),
```

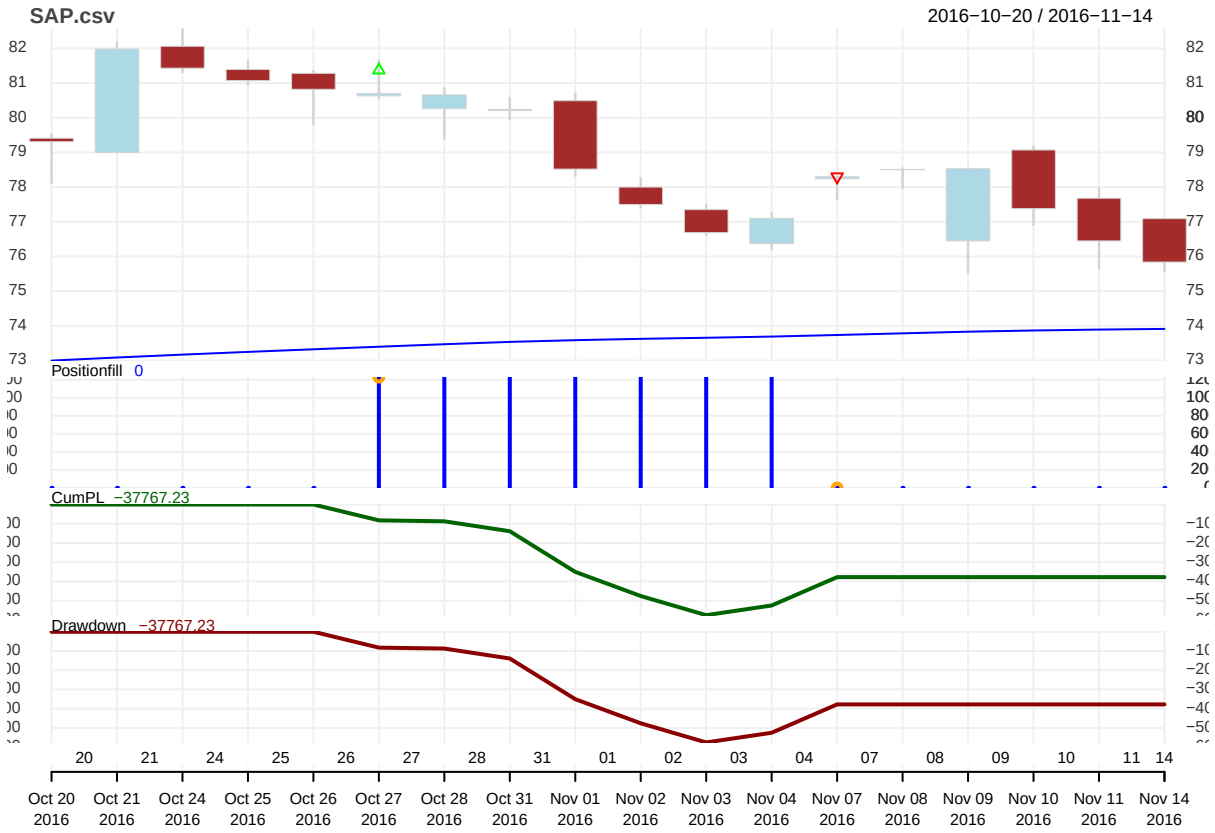
```
      TxnPrice = OpenTomorrow,
      TxnQty = -Position,
      TxnFees = transactionCost
    )

  } else {
    # Check Rule 3:

    # Sell Rule 3: If today was a smash day (short) place an order at todays
    # low price. (Buy tomorrow for 'price <= todays low')

    # Simulate this behaviour:

    # Option 1 to check: Check whether the open price tomorrow is below today's
    # low and add the transaction tomorrow at tomorrows open price.

    # Option 2 to check: Check whether today's low was larger than tomorrow's
    # low and add the transaction tomorrow at today's low price.

    # Check for Smash Day (Short)
    if (CloseToday > HighYesterday) {
      # Check for Option 1
      if (OpenTomorrow < LowToday) {
        # Add Sell transaction tomorrow at tomorrow's open price
        addTxn(
          portfolioname,
          Symbol = instrument,
          TxnDate = time(symbol[i + 1]),
          TxnPrice = OpenTomorrow,
          TxnQty = -Position,
          TxnFees = transactionCost
        )

      } else {
        # Check for Option 2
        if (LowToday > LowTomorrow) {
          # Add Sell transaction tomorrow at today's low price
          addTxn(
            portfolioname,
            Symbol = instrument,
            TxnDate = time(symbol[i + 1]),
            TxnPrice = LowToday,
            TxnQty = -Position,
            TxnFees = transactionCost
          )
        }
      }
    } else {
      # Check Rule 4
      if (i == nrow(symbol) - 1) {
        # Add Sell transaction for the last day at the close price
        addTxn(
          portfolioname,
```

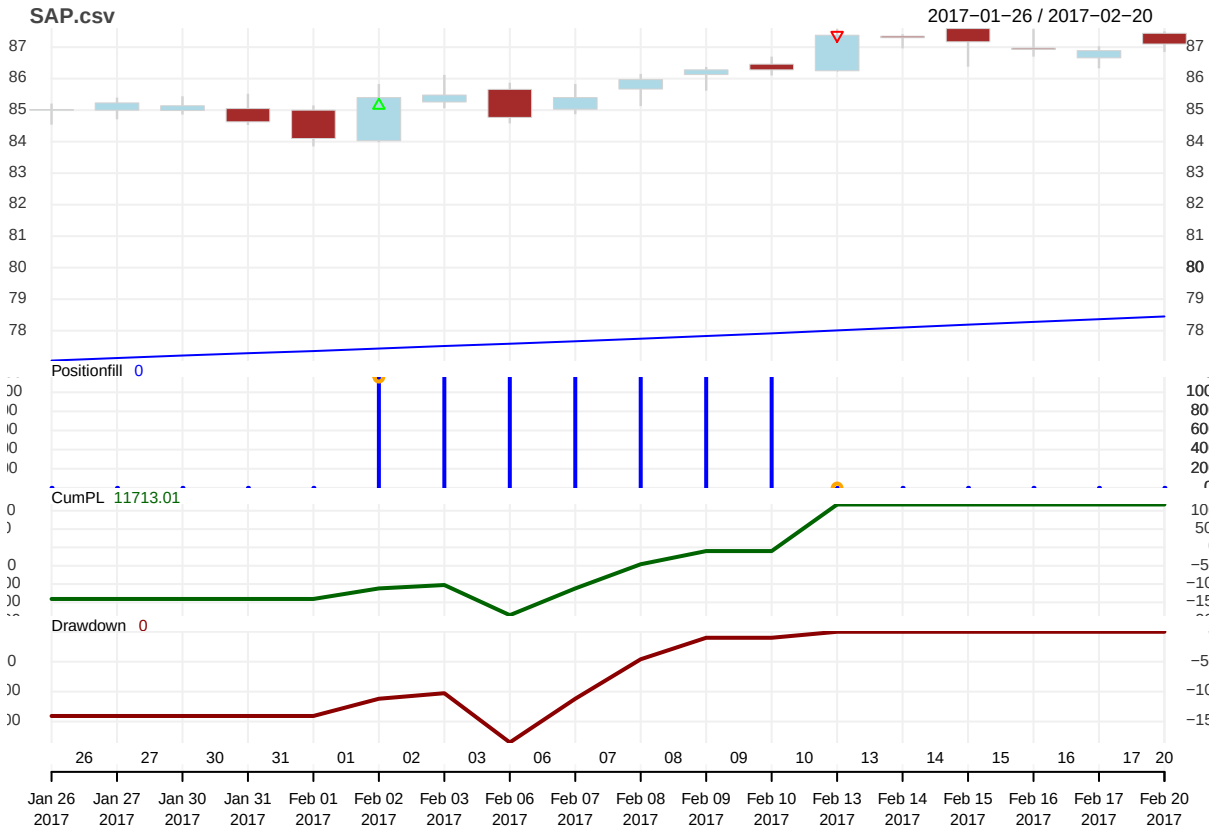
```
                Symbol = instrument,
                TxnDate = time(symbol[i + 1]),
                TxnPrice = as.numeric(symbol[i, "Close"]),
                TxnQty = -Position,
                TxnFees = transactionCost
              )
            }
          }
        }
      }
    }

    updatePortf(portfolioname, Symbols = instrument, Dates = CurrentDate)
    updateAcct(accountname, Dates = CurrentDate)
    updateEndEq(accountname, CurrentDate)

  } # End Bar-by-bar processing
} # End for loop for multiple instruments
```

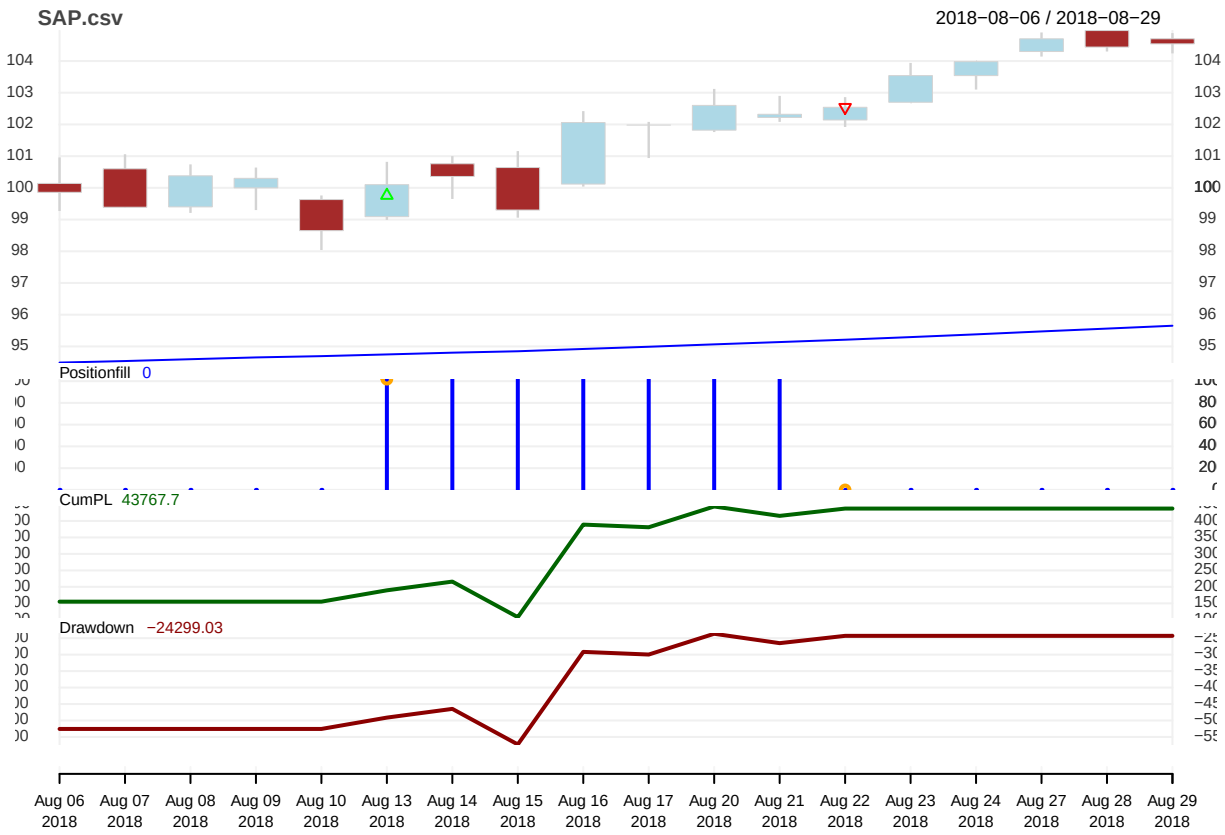
Step 2: System Check

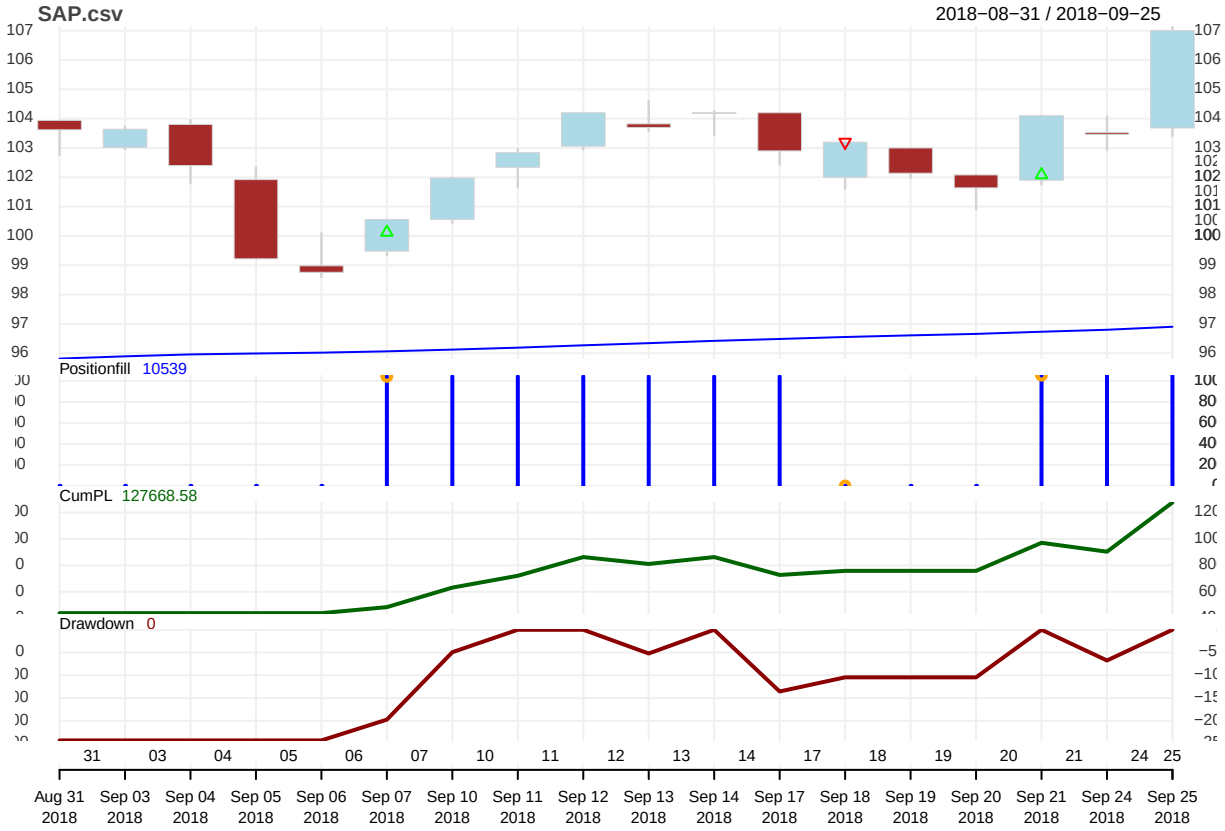
In order to make sure that the system works as designed the plots of some choosen transactions are printed in the following.

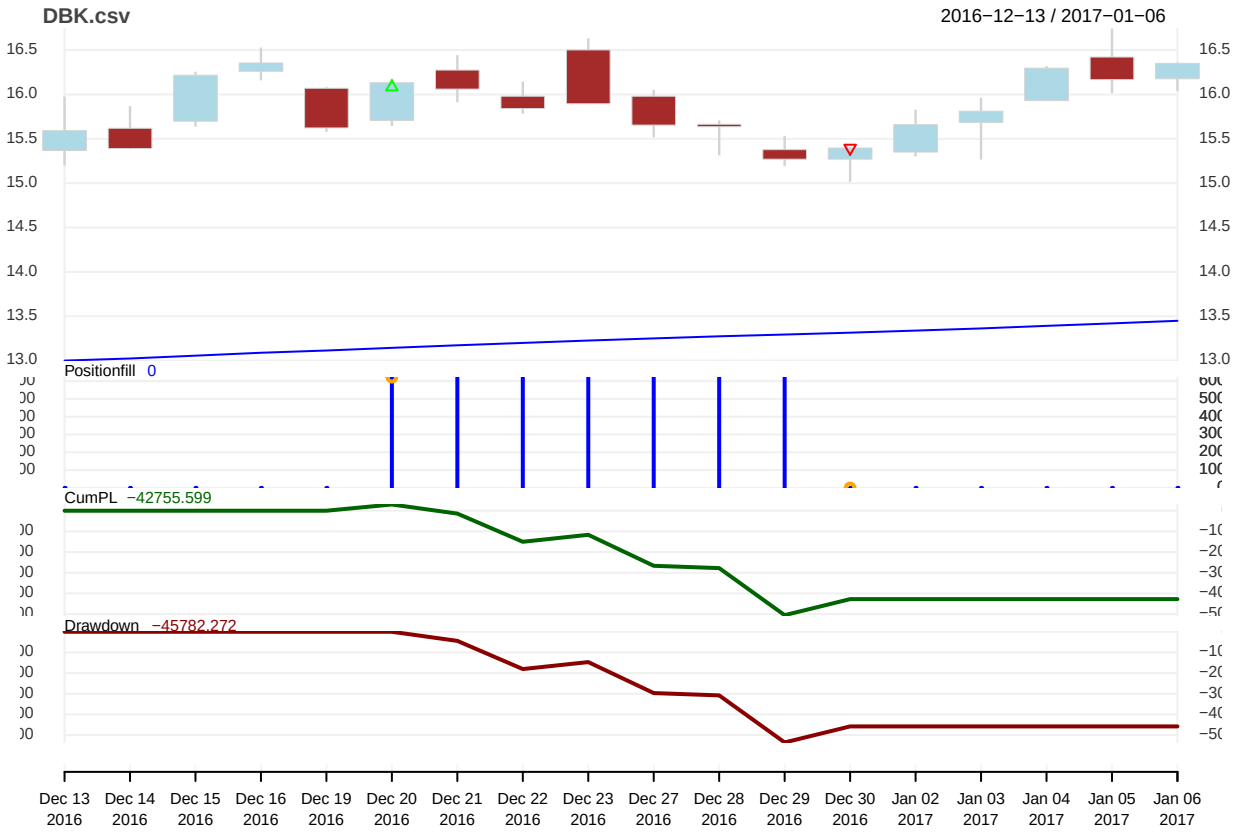
```
for (instrument in instrumentlist){
  rm(daterange_check)
  daterange_check <- c()
  transactionsInstrument <- getTxns(Portfolio=portfolioname,Symbol=instrument)
  for (i in c(2,6, (nrow(transactionsInstrument)-7), (nrow(transactionsInstrument)-5))) {
    from <- as.Date(index(transactionsInstrument[i,1]))-7
    to <- as.Date(index(transactionsInstrument[i+1,1]))+7
    daterange_check <- c(daterange_check, paste(from, "::", to, sep = ""))
  }
  for (daterange_check_i in daterange_check){
    print(chart.Posn(portfolioname,Symbol=instrument,type='candlesticks', theme=myTheme,subset=daterang
  }
}
```

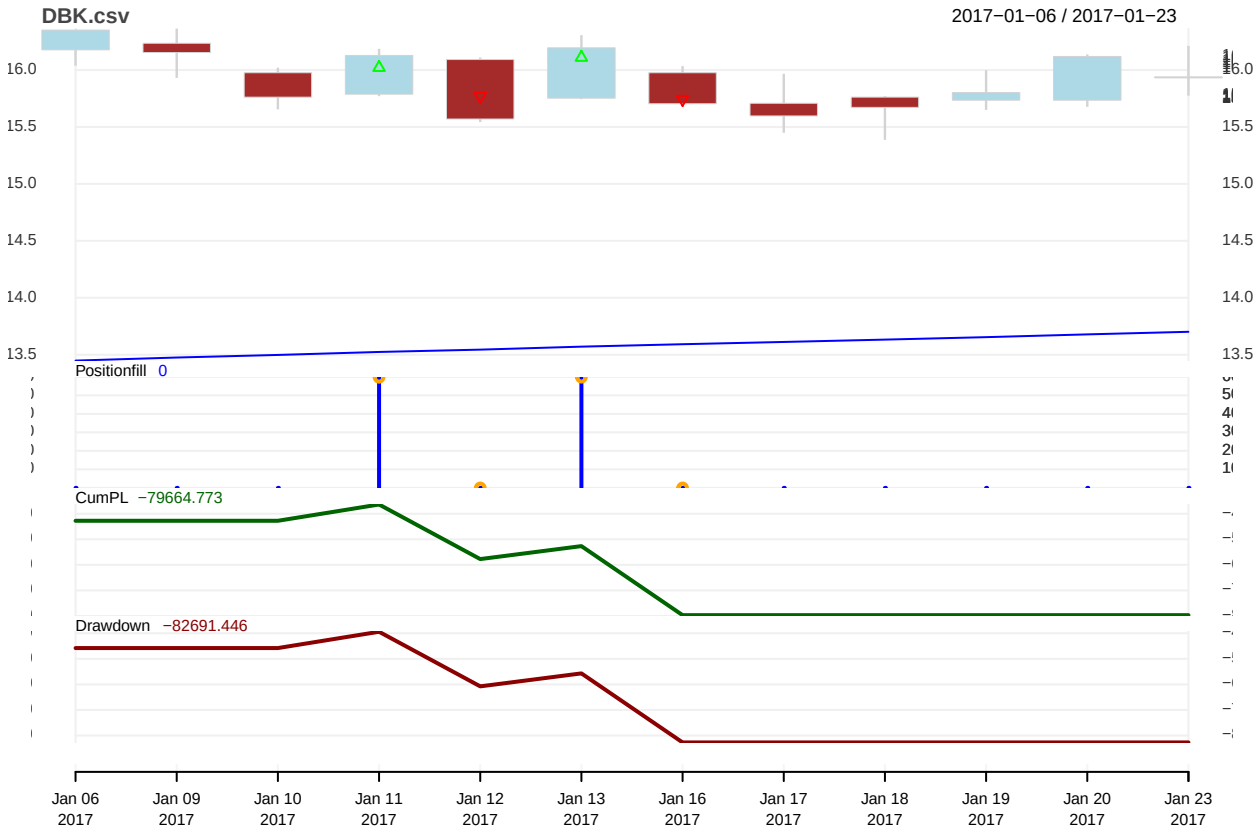

SAP.csv
2016−10−20 / 2016−11−14
Positionfill 0
CumPL −37767.23
Drawdown −37767.23
Oct 20 2016
Oct 21 2016
Oct 24 2016
Oct 25 2016
Oct 26 2016
Oct 27 2016
Oct 28 2016
Oct 31 2016
Nov 01 2016
Nov 02 2016
Nov 03 2016
Nov 04 2016
Nov 07 2016
Nov 08 2016
Nov 09 2016
Nov 10 2016
Nov 11 2016
Nov 14 2016

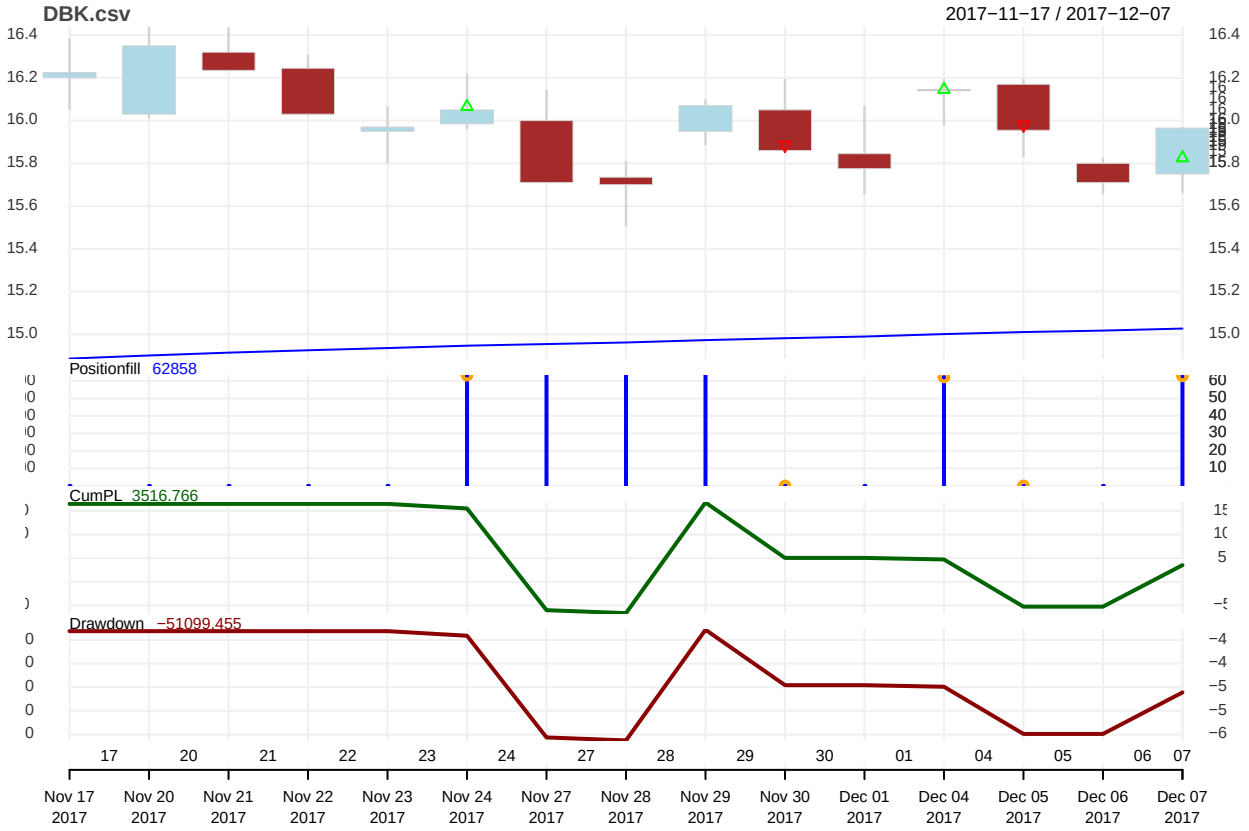

SAP.csv
2017−01−26 / 2017−02−20
Positionfill 0
CumPL 11713.01
Drawdown 0
26 27 30 31 01 02 03 06 07 08 09 10 13 14 15 16 17 20
Jan 26 2017
Jan 27 2017
Jan 30 2017
Jan 31 2017
Feb 01 2017
Feb 02 2017
Feb 03 2017
Feb 06 2017
Feb 07 2017
Feb 08 2017
Feb 09 2017
Feb 10 2017
Feb 13 2017
Feb 14 2017
Feb 15 2017
Feb 16 2017
Feb 17 2017
Feb 20 2017

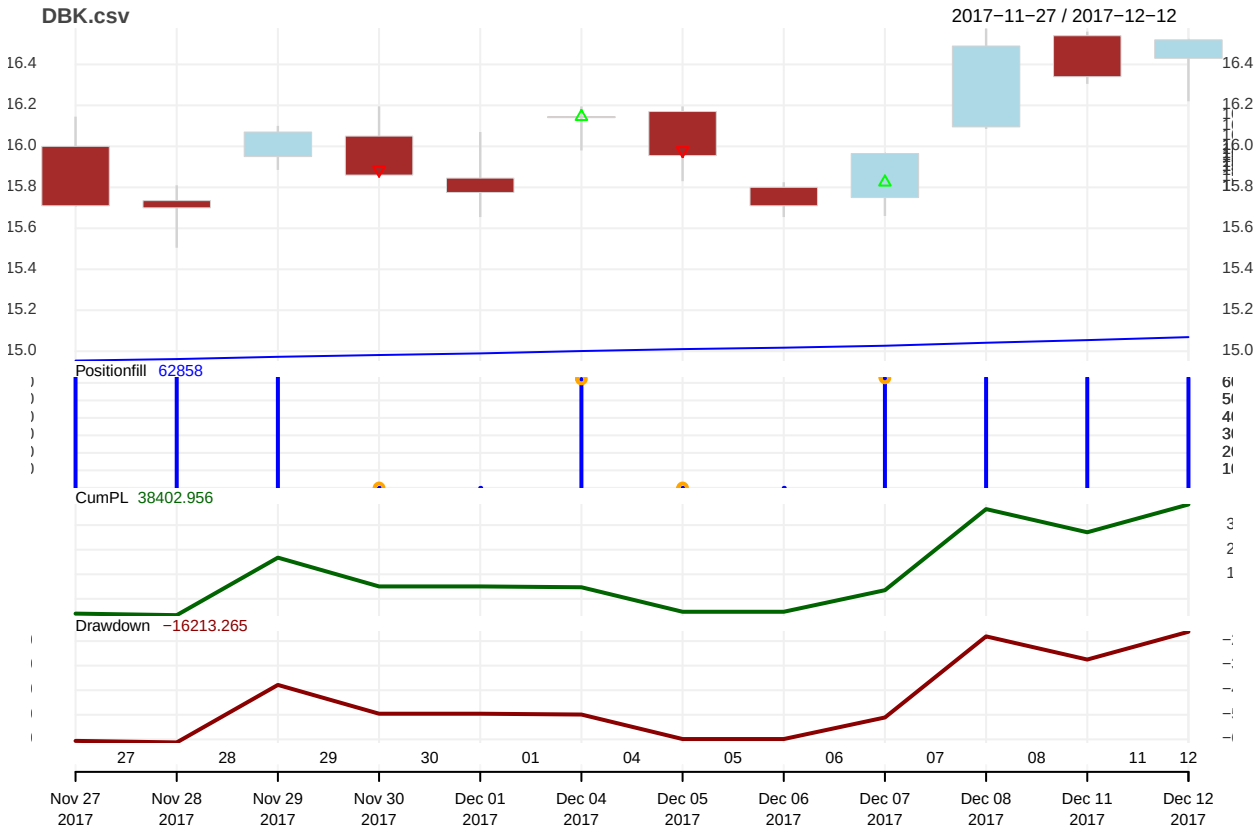

SAP.csv
2018−08−06 / 2018−08−29
Positionfill 0
CumPL 43767.7
Drawdown −24299.03
Aug 06 2018
Aug 07 2018
Aug 08 2018
Aug 09 2018
Aug 10 2018
Aug 13 2018
Aug 14 2018
Aug 15 2018
Aug 16 2018
Aug 17 2018
Aug 20 2018
Aug 21 2018
Aug 22 2018
Aug 23 2018
Aug 24 2018
Aug 27 2018
Aug 28 2018
Aug 29 2018


SAP.csv
2018−08−31 / 2018−09−25
Positionfill 10539
CumPL 127668.58
Drawdown 0
Aug 31 2018
Sep 03 2018
Sep 04 2018
Sep 05 2018
Sep 06 2018
Sep 07 2018
Sep 10 2018
Sep 11 2018
Sep 12 2018
Sep 13 2018
Sep 14 2018
Sep 17 2018
Sep 18 2018
Sep 19 2018
Sep 20 2018
Sep 21 2018
Sep 24 2018
Sep 25 2018


DBK.csv
2016−12−13 / 2017−01−06
16.5
16.0
15.5
15.0
14.5
14.0
13.5
13.0
Positionfill 0
CumPL −42755.599
Drawdown −45782.272
Dec 13 2016
Dec 14 2016
Dec 15 2016
Dec 16 2016
Dec 19 2016
Dec 20 2016
Dec 21 2016
Dec 22 2016
Dec 23 2016
Dec 27 2016
Dec 28 2016
Dec 29 2016
Dec 30 2016
Jan 02 2017
Jan 03 2017
Jan 04 2017
Jan 05 2017
Jan 06 2017


DBK.csv
2017−01−06 / 2017−01−23
16.0
15.5
15.0
14.5
14.0
13.5
Positionfill 0
CumPL −79664.773
Drawdown −82691.446
Jan 06 2017
Jan 09 2017
Jan 10 2017
Jan 11 2017
Jan 12 2017
Jan 13 2017
Jan 16 2017
Jan 17 2017
Jan 18 2017
Jan 19 2017
Jan 20 2017
Jan 23 2017


DBK.csv
2017−11−17 / 2017−12−07
16.4
16.2
16.0
15.8
15.6
15.4
15.2
15.0
Positionfill 62858
60
50
40
30
20
10
CumPL 3516.766
15
10
5
Drawdown −51099.455
−4
−4
−5
−5
−6
17
20
21
22
23
24
27
28
29
30
01
04
05
06
07
Nov 17 2017
Nov 20 2017
Nov 21 2017
Nov 22 2017
Nov 23 2017
Nov 24 2017
Nov 27 2017
Nov 28 2017
Nov 29 2017
Nov 30 2017
Dec 01 2017
Dec 04 2017
Dec 05 2017
Dec 06 2017
Dec 07 2017



Part C: Analysis and Reporting

Step 1: Visualize original data

Plot of the instrument with the EMA line which indicates the general trend of the stock exponentially smoothed for the last 200 days. Moreover, the tradevolume is added below the graph.


SAP.csv
2016−01−04 / 2018−10−11
105
100
95
90
85
80
75
70
65
Volume: 5,210,537
2016
2017
2018
Jan 04 2016
Jul 01 2016
Jan 02 2017
Jul 03 2017
Jan 02 2018
Jun 29 2018

Step 2: All transactions performed by the trading system

The following table can be used to get a better overview of the transactions performed and the exact details per transaction.

```
## [1] "SAP.csv"
##            Txn.Qty Txn.Price Txn.Fees  Txn.Value Txn.Avg.Cost
## 1999-12-31       0      0.00        0        0.0         0.00
## 2016-10-27   12289     81.37      -20   999955.9        81.37
## 2016-11-07  -12289     78.30      -20  -962228.7        78.30
## 2016-12-05   12430     77.41      -20   962206.3        77.41
## 2016-12-12  -12430     79.32      -20  -985947.6        79.32
## 2017-02-02   11578     85.15      -20   985866.7        85.15
## 2017-02-13  -11578     87.38      -20 -1011685.6        87.38
## 2017-03-28   11246     89.96      -20  1011690.2        89.96
## 2017-04-06  -11246     91.43      -20 -1028221.8        91.43
## 2017-04-12   11222     91.62      -20  1028159.6        91.62
## 2017-04-25  -11222     93.13      -20 -1045104.9        93.13
## 2017-05-12   11054     94.54      -20  1045045.2        94.54
## 2017-05-23  -11054     94.68      -20 -1046592.7        94.68
## 2017-06-16   11124     94.08      -20  1046545.9        94.08
## 2017-06-23  -11124     95.44      -20 -1061674.6        95.44
## 2017-06-29   11293     94.01      -20  1061654.9        94.01
## 2017-07-10  -11293     91.31      -20 -1031163.8        91.31
## 2017-07-12   11249     91.66      -20  1031083.3        91.66
## 2017-07-20  -11249     90.18      -20 -1014434.8        90.18
```

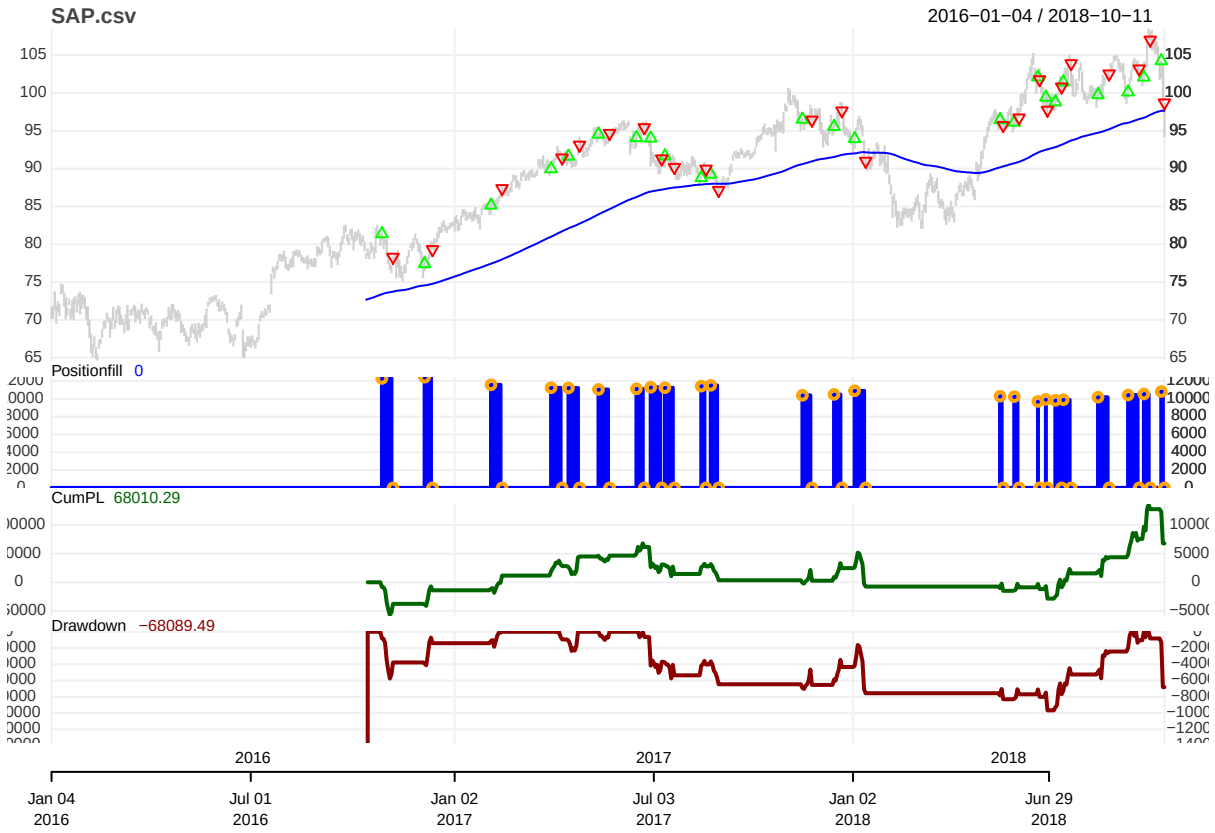
```
## 2017-08-14   11425     88.79      -20  1014425.8        88.79
## 2017-08-17  -11425     89.96      -20 -1027793.0        89.96
## 2017-08-22   11516     89.25      -20  1027803.0        89.25
## 2017-08-29  -11516     87.14      -20 -1003504.2        87.14
## 2017-11-14   10397     96.51      -20  1003414.5        96.51
## 2017-11-22  -10397     96.44      -20 -1002686.7        96.44
## 2017-12-12   10494     95.55      -20  1002701.7        95.55
## 2017-12-19  -10494     97.65      -20 -1024739.1        97.65
## 2018-01-03   10906     93.95      -20  1024618.7        93.95
## 2018-01-12  -10906     91.00      -20  -992446.0        91.00
## 2018-05-16   10291     96.44      -20   992464.0        96.44
## 2018-05-18  -10291     95.72      -20  -985054.5        95.72
## 2018-05-30   10248     96.12      -20   985037.8        96.12
## 2018-06-04  -10248     96.72      -20  -991186.6        96.72
## 2018-06-20    9707    102.10      -20   991084.7       102.10
## 2018-06-21   -9707    101.78      -20  -987978.5       101.78
## 2018-06-27    9937     99.42      -20   987936.5        99.42
## 2018-06-28   -9937     97.75      -20  -971341.8        97.75
## 2018-07-05    9831     98.80      -20   971302.8        98.80
## 2018-07-11   -9831    100.82      -20  -991161.4       100.82
## 2018-07-12    9903    101.44      -20  1004560.3       101.44
## 2018-07-19   -9903    103.90      -20 -1028921.7       103.90
## 2018-08-13   10179     99.76      -20  1015457.0        99.76
## 2018-08-22  -10179    102.54      -20 -1043754.7       102.54
## 2018-09-07   10425    100.12      -20  1043751.0       100.12
## 2018-09-18  -10425    103.20      -20 -1075860.0       103.20
## 2018-09-21   10539    102.08      -20  1075821.1       102.08
## 2018-09-27  -10539    107.02      -20 -1127883.8       107.02
## 2018-10-09   10819    104.24      -20  1127772.6       104.24
## 2018-10-11  -10819     98.71      -20 -1067943.5        98.71
##            Net.Txn.Realized.PL
## 1999-12-31                0.00
## 2016-10-27              -20.00
## 2016-11-07           -37747.23
## 2016-12-05              -20.00
## 2016-12-12            23721.30
## 2017-02-02              -20.00
## 2017-02-13            25798.94
## 2017-03-28              -20.00
## 2017-04-06            16511.62
## 2017-04-12              -20.00
## 2017-04-25            16925.22
## 2017-05-12              -20.00
## 2017-05-23             1527.56
## 2017-06-16              -20.00
## 2017-06-23            15108.64
## 2017-06-29              -20.00
## 2017-07-10           -30511.10
## 2017-07-12              -20.00
## 2017-07-20           -16668.52
## 2017-08-14              -20.00
## 2017-08-17            13347.25
## 2017-08-22              -20.00
## 2017-08-29           -24318.76
```

```
## 2017-11-14             -20.00
## 2017-11-22            -747.79
## 2017-12-12             -20.00
## 2017-12-19           22017.40
## 2018-01-03             -20.00
## 2018-01-12          -32192.70
## 2018-05-16             -20.00
## 2018-05-18           -7429.52
## 2018-05-30             -20.00
## 2018-06-04            6128.80
## 2018-06-20             -20.00
## 2018-06-21           -3126.24
## 2018-06-27             -20.00
## 2018-06-28          -16614.79
## 2018-07-05             -20.00
## 2018-07-11           19838.62
## 2018-07-12             -20.00
## 2018-07-19           24341.38
## 2018-08-13             -20.00
## 2018-08-22           28277.62
## 2018-09-07             -20.00
## 2018-09-18           32089.00
## 2018-09-21             -20.00
## 2018-09-27           52042.66
## 2018-10-09             -20.00
## 2018-10-11          -59849.07
## [1] "DBK.csv"
##            Txn.Qty Txn.Price Txn.Fees  Txn.Value Txn.Avg.Cost
## 1999-12-31       0     0.000        0        0.0        0.000
## 2016-12-20   62177    16.083      -20   999992.7       16.083
## 2016-12-30  -62177    15.396      -20  -957277.1       15.396
## 2017-01-11   59749    16.021      -20   957238.7       16.021
## 2017-01-12  -59749    15.771      -20  -942301.5       15.771
## 2017-01-13   59814    16.110      -20   963603.5       16.110
## 2017-01-16  -59814    15.744      -20  -941711.6       15.744
## 2017-01-31   54130    17.002      -20   920318.3       17.002
## 2017-02-02  -54130    16.494      -20  -892820.2       16.494
## 2017-02-03   55368    16.735      -20   926583.5       16.735
## 2017-02-14  -55368    16.440      -20  -910249.9       16.440
## 2017-03-08   55969    15.659      -20   876418.6       15.659
## 2017-03-17  -55969    15.940      -20  -892145.9       15.940
## 2017-03-21   56409    15.815      -20   892108.3       15.815
## 2017-03-30  -56409    15.830      -20  -892954.5       15.830
## 2017-04-12   57440    15.545      -20   892904.8       15.545
## 2017-04-25  -57440    16.940      -20  -973033.6       16.940
## 2017-06-07   62794    15.495      -20   972993.0       15.495
## 2017-06-12  -62794    15.530      -20  -975190.8       15.530
## 2017-07-03   61138    15.950      -20   975151.1       15.950
## 2017-07-12  -61138    16.440      -20 -1005108.7       16.440
## 2017-07-24   63532    15.820      -20  1005076.2       15.820
## 2017-07-27  -63532    16.000      -20 -1016512.0       16.000
## 2017-11-24   63272    16.065      -20  1016464.7       16.065
## 2017-11-30  -63272    15.885      -20 -1005075.7       15.885
## 2017-12-04   62251    16.145      -20  1005042.4       16.145
```

```
## 2017-12-05  -62251    15.980      -20  -994771.0       15.980
## 2017-12-07   62858    15.825      -20   994727.8       15.825
## 2017-12-18  -62858    17.100      -20 -1074871.8       17.100
## 2018-01-02   67035    16.034      -20  1074839.2       16.034
## 2018-01-05  -67035    15.884      -20 -1064783.9       15.884
##            Net.Txn.Realized.PL
## 1999-12-31               0.000
## 2016-12-20             -20.000
## 2016-12-30          -42735.599
## 2017-01-11             -20.000
## 2017-01-12          -14957.250
## 2017-01-13             -20.000
## 2017-01-16          -21911.924
## 2017-01-31             -20.000
## 2017-02-02          -27518.040
## 2017-02-03             -20.000
## 2017-02-14          -16353.560
## 2017-03-08             -20.000
## 2017-03-17           15707.289
## 2017-03-21             -20.000
## 2017-03-30             826.135
## 2017-04-12             -20.000
## 2017-04-25           80108.800
## 2017-06-07             -20.000
## 2017-06-12           2177.790
## 2017-07-03             -20.000
## 2017-07-12           29937.620
## 2017-07-24             -20.000
## 2017-07-27           11415.760
## 2017-11-24             -20.000
## 2017-11-30          -11408.960
## 2017-12-04             -20.000
## 2017-12-05          -10291.415
## 2017-12-07             -20.000
## 2017-12-18           80123.950
## 2018-01-02             -20.000
## 2018-01-05          -10075.250
```

Step 3: Graph which visualize the transactions

The following graph shows the combined view of the performance of the Smash Day trading system. It visualizes the trades (buy-transactions are visualized in green and sell-transactions are visualized in red). Moreover, the size of the blue squares indicates the size of the position (height) and the holding duration of the position (width). The green line shows the cummulative net profit curve, while the red curve indicates the drawdown on each day compared to the last reached high.


SAP.csv
2016−01−04 / 2018−10−11
Positionfill 0
CumPL 68010.29
Drawdown −68089.49
2016
2017
2018
Jan 04 2016
Jul 01 2016
Jan 02 2017
Jul 03 2017
Jan 02 2018
Jun 29 2018

Step 4: Performance Statistics

The following table summarizes some important trading statistics for all instruments.

```
library(PerformanceAnalytics) # contains lots of methods to investigate performance
# obtain the portfolio returns - with these you can compute virtually any financial metrics you wish
rets <- PortfReturns(Account=accountname)
rownames(rets) <- NULL # this step is important!

tstats <- tradeStats(Portfolio=portfolioname, Symbols=instrumentlist)
for (i in 1:nrow(tstats)) {
  trades.tab <- cbind(
    c("Trades", "Win Percent", "Loss Percent","W/L Ratio"),
    c(tstats[i,"Num.Trades"],tstats[i,"Percent.Positive"],tstats[i,"Percent.Negative"],tstats[i,"Percen
  print(row.names(tstats[i,]))
  print(trades.tab)
  }
```

```
## [1] "SAP.csv"
##      [,1]           [,2]
## [1,] "Trades"       "24"
## [2,] "Win Percent"  "58.3333333333333"
## [3,] "Loss Percent" "41.6666666666667"
## [4,] "W/L Ratio"    "1.4"
## [1] "DBK.csv"
##      [,1]           [,2]
## [1,] "Trades"       "15"
```

```
## [2,] "Win Percent"  "46.6666666666667"
## [3,] "Loss Percent" "53.3333333333333"
## [4,] "W/L Ratio"    "0.875"
```

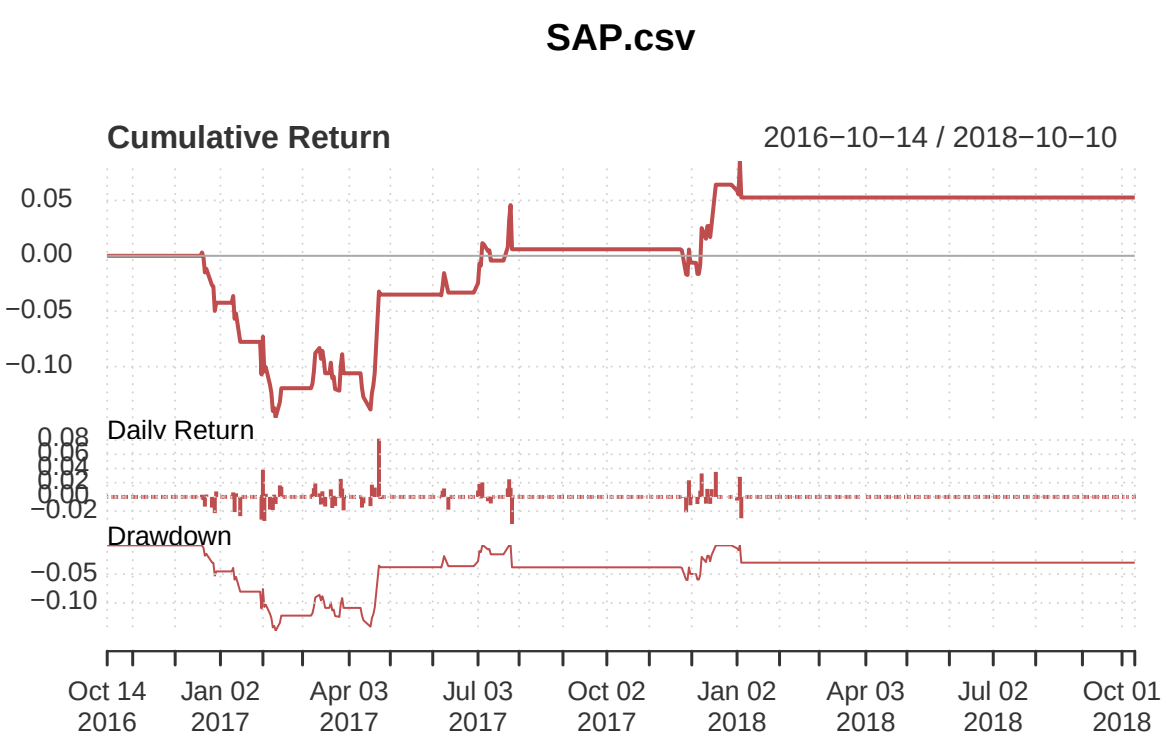
Step 5: Calculate statistics of the Portfolio and all instruments in the portfolio

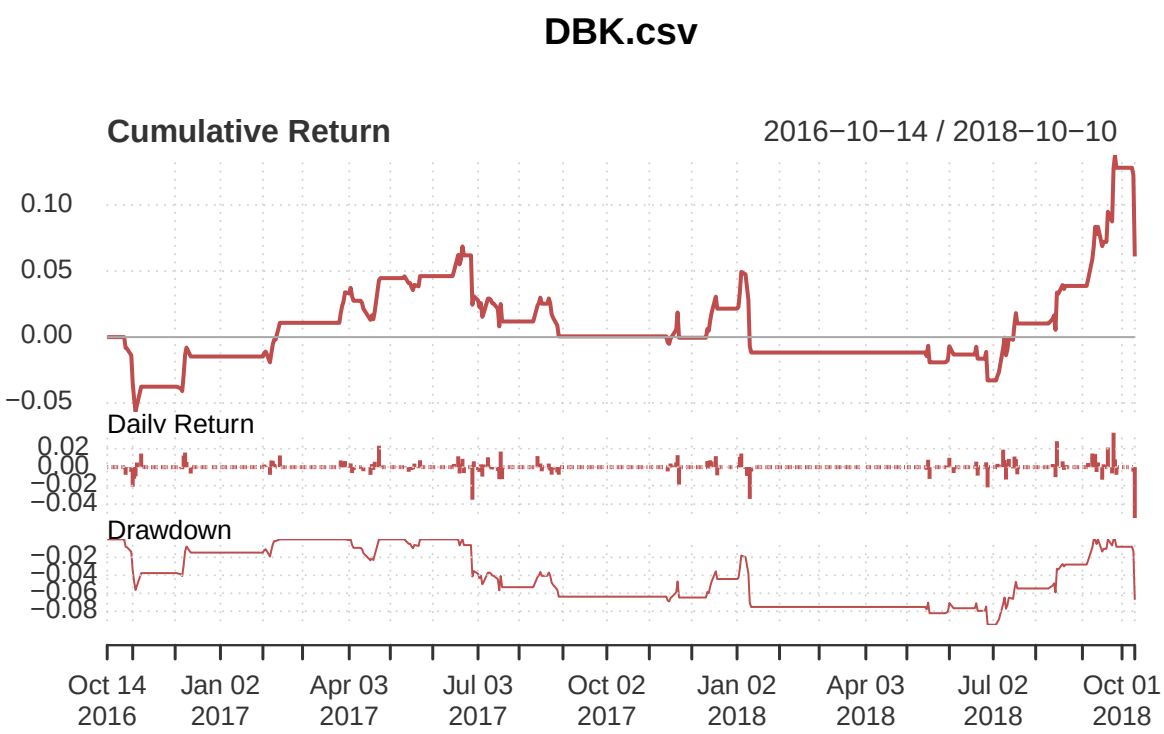
```
tab.perf <- table.Arbitrary(rets, metrics=c("Return.cumulative", "Return.annualized","SharpeRatio.annual
                            metricsNames=c("Cumulative Return", "Annualized Return", "Annualized Sharp
tab.risk <- table.Arbitrary(rets, metrics=c("StdDev.annualized", "maxDrawdown","VaR","ES"),
                            metricsNames=c("Annualized StdDev", "Max Drawdown", "Value-at-Risk","Conditi
# present the portfolio statistics
for (i in 1:ncol(tab.perf)) {
somestats <- data.frame(rownames(tab.perf),tab.perf[i,1], rownames(tab.risk),tab.risk[i,1])
colnames(rets[,i])
colnames(somestats) <- c("Performance Metric", "Performance Value", "Risk Metric", "Risk Value")
print(somestats)
}
```

```
##       Performance Metric Performance Value       Risk Metric Risk Value
## 1      Cumulative Return        0.05260045 Annualized StdDev  0.1169277
## 2      Annualized Return        0.05260045      Max Drawdown  0.1169277
## 3 Annualized Sharp Ratio        0.05260045     Value-at-Risk  0.1169277
## 4           Calmar Ratio        0.05260045   Conditional VaR  0.1169277
##       Performance Metric Performance Value       Risk Metric Risk Value
## 1      Cumulative Return        0.02596318 Annualized StdDev  0.1483677
## 2      Annualized Return        0.02596318      Max Drawdown  0.1483677
## 3 Annualized Sharp Ratio        0.02596318     Value-at-Risk  0.1483677
## 4           Calmar Ratio        0.02596318   Conditional VaR  0.1483677
```

Step 6: Visualize returns of the trading strategy for every instrument

```
for (i in 1:length(instrumentlist)){
  charts.PerformanceSummary(rets[,i],colorset=rainbow12equal,main=instrumentlist[i])
}
```


SAP.csv
Cumulative Return
2016−10−14 / 2018−10−10
0.05
0.00
−0.05
−0.10
Dailv Return
0.08
0.06
0.04
0.02
0.00
−0.02
Drawdown
−0.05
−0.10
Oct 14
2016
Jan 02
2017
Apr 03
2017
Jul 03
2017
Oct 02
2017
Jan 02
2018
Apr 03
2018
Jul 02
2018
Oct 01
2018



Compare with Buy and Hold Strategy

In order to compare the trading strategy properly we need to define a benchmark against which we can measure the results. In this case a simple buy and hold strategy is used. At the first date of the trading period we place a buy order and sell our position at the last day of the selected period. In order to do this we create a new Portfolio and a new Account.

Step 1: Perform the Buy and Hold Strategy

```
# We remove any objects, in case there was a buyhold portfolio initialized before
suppressWarnings(try(rm(list=c("account.buyhold","portfolio.buyhold"),pos=.blotter)))

# The Buy and hold symbol is loaded
LoadCourseFile(BuyHoldDirectory,BuyHoldInstrument,debugme=TRUE,dates=daterange)
# The Buy and hold instrument is initialized
stock(BuyHoldInstrument,currency="EUR")

BuyHoldSymbol<-get(BuyHoldInstrument)

# The portfolio and account "buyhold" is initialized
initPortf("buyhold",BuyHoldInstrument,initDate=initdate,currency="EUR")
initAcct("buyhold",portfolios="buyhold",initDate=initdate,initEq=startCapital,currency="EUR")

# The first date of the defined daterange is selected
currentdate <- first(time(BuyHoldSymbol))
```

```
# The close price at this date is selected
closeprice <- as.numeric(Cl(BuyHoldSymbol[currentdate,]))

# Calculate the unitsize we can buy with our startingcapital
unitsize <- as.numeric(trunc(startCapital/closeprice))

# Place the transaction for the instrument at the first date
addTxn("buyhold",Symbol=BuyHoldInstrument,TxnDate=currentdate,TxnPrice=closeprice,TxnQty=unitsize,Txn

# Select the last date of the daterange period
lastdate <-last(time(BuyHoldSymbol))

# Select the price at the last date
lastprice <- as.numeric(Cl(BuyHoldSymbol[lastdate,]))

# Sell the position at the last date of the daterange
addTxn("buyhold",Symbol=BuyHoldInstrument,TxnDate=lastdate,TxnPrice=lastprice,TxnQty=-unitsize,TxnFee

# update portfolio and account
updatePortf(Portfolio="buyhold")
updateAcct(name="buyhold")
updateEndEq(Account="buyhold")
```

Step 2: Visualize the Buy and Hold strategy

We can see that we hold the position from the first until the last date. The cumulative profits are visualized by the green line.

```
chart.Posn("buyhold",Symbol=BuyHoldInstrument, theme=myTheme)
```

Step 3: Compare the returns of the trading strategy with the buy and hold strategy

In order to compare the results of both strategies we calculat the returns for the buy and hold strategy and combine them with the returns of the trading strategy which were calculatet before.

```
rets.bh <- PortfReturns(Account='buyhold')
returns <- cbind(rets,rets.bh)
#rulecol <- paste(portfolioname,instrument,sep="-")
#colnames(returns) <- c(rulecol,"Buy-and-hold")
```

We compare the two strategies by showing some statistical metrics of the returns and plot the returns in one chart to directly compare the performance of the strategies.

```
table.Stats(returns)
```

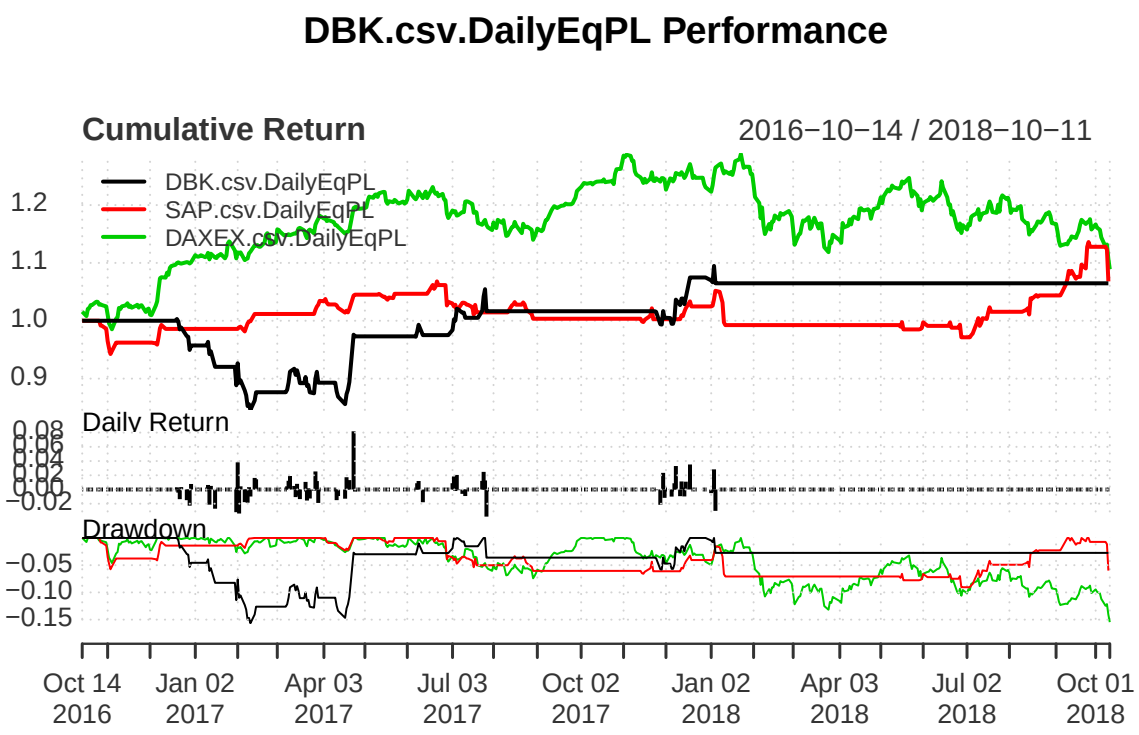
```
##                  DBK.csv.DailyEqPL SAP.csv.DailyEqPL DAXEX.csv.DailyEqPL
## Observations            504.0000         504.0000           705.0000
## NAs                     201.0000         201.0000             0.0000
## Minimum                  -0.0381          -0.0551            -0.0664
## Quartile 1                0.0000           0.0000            -0.0053
## Median                    0.0000           0.0000             0.0007
## Arithmetic Mean           0.0001           0.0001             0.0001
## Geometric Mean            0.0001           0.0001             0.0001
## Quartile 3                0.0000           0.0000             0.0062
## Maximum                   0.0819           0.0373             0.0384
## SE Mean                   0.0003           0.0003             0.0004
## LCL Mean (0.95)          -0.0005          -0.0004            -0.0007
## UCL Mean (0.95)           0.0008           0.0006             0.0009
```

```
## Variance                   0.0001           0.0000           0.0001
## Stdev                      0.0074           0.0059           0.0106
## Skewness                   2.6623          -1.5893          -0.4098
## Kurtosis                  37.1553          25.3966           2.4233
```

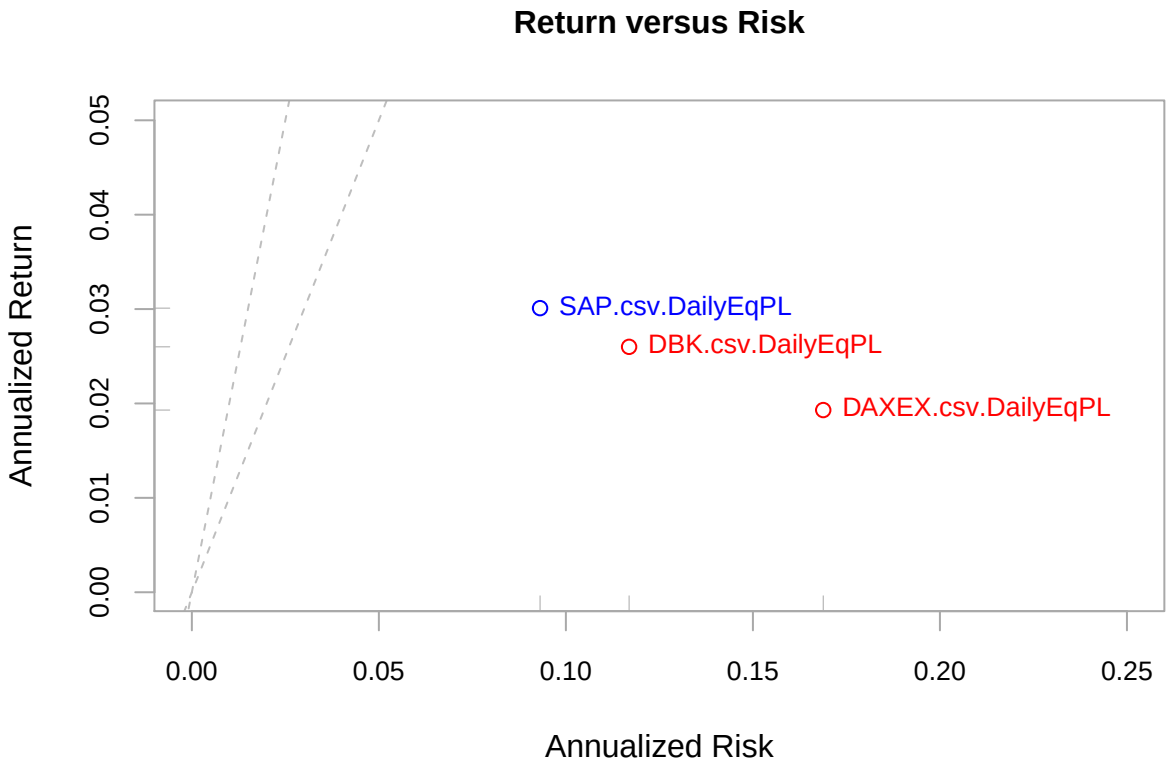
```
table.AnnualizedReturns(returns)
```

```
##                           DBK.csv.DailyEqPL SAP.csv.DailyEqPL
## Annualized Return                    0.0260            0.0301
## Annualized Std Dev                   0.1169            0.0931
## Annualized Sharpe (Rf=0%)            0.2220            0.3234
##                           DAXEX.csv.DailyEqPL
## Annualized Return                      0.0193
## Annualized Std Dev                     0.1688
## Annualized Sharpe (Rf=0%)              0.1145
```

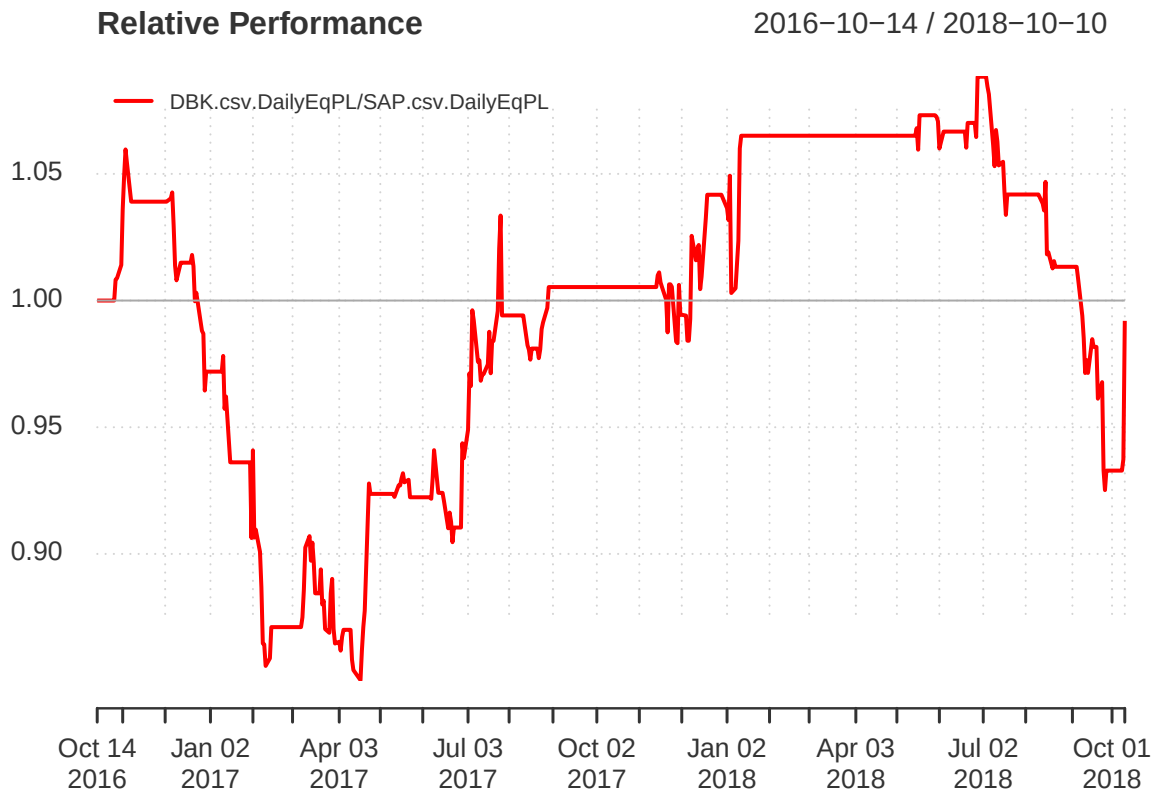
```
charts.PerformanceSummary(returns,geometric=FALSE,wealth.index=TRUE)
```




```
chart.RiskReturnScatter(returns,Rf=0,add.sharpe=c(1,2),xlim=c(0,0.25),main="Return versus Risk",colorse
```

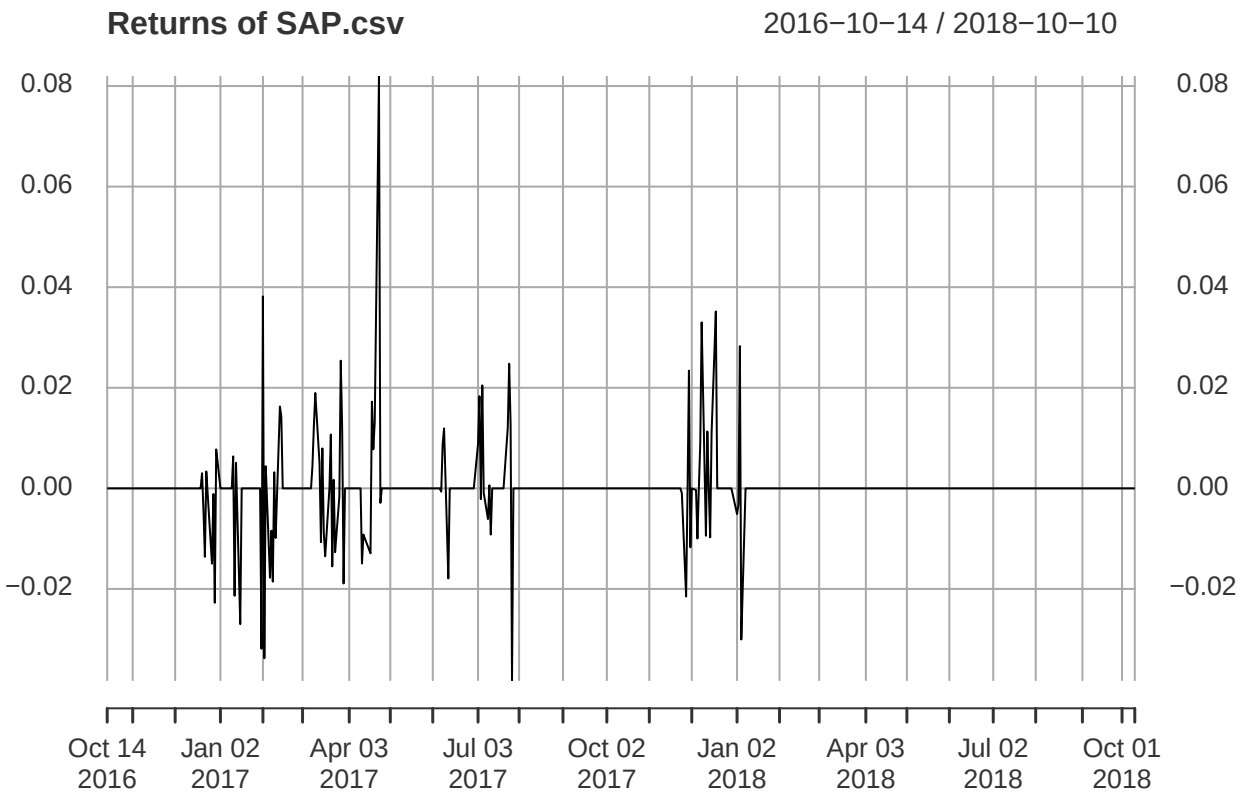


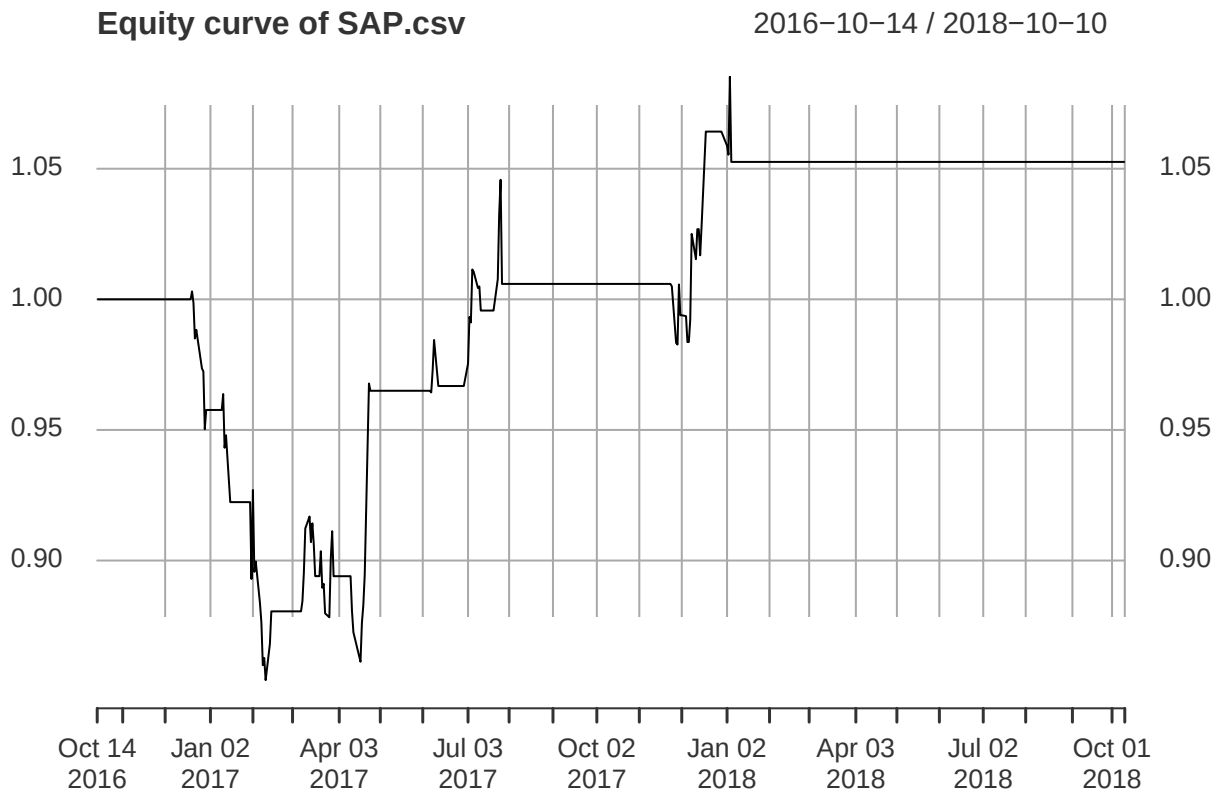

```
chart.RelativePerformance(returns[,1],returns[,2],colorset=c("red","blue"),lwd=2,legend.loc="topleft")
```

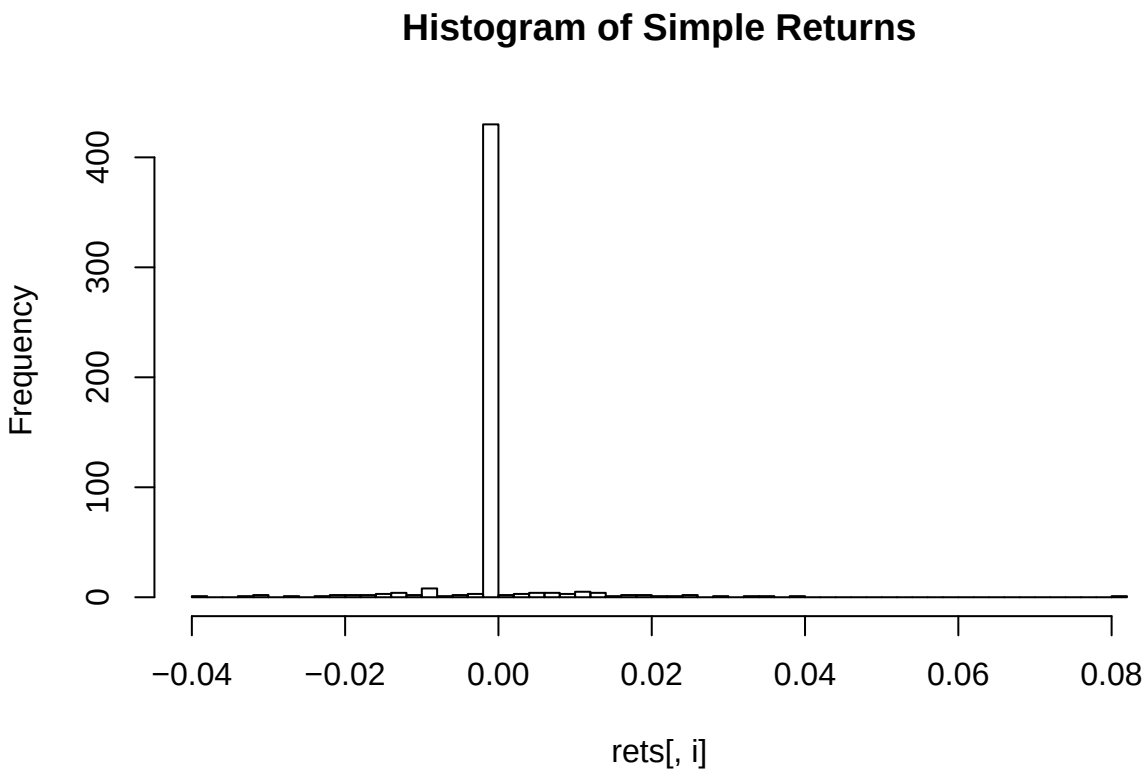


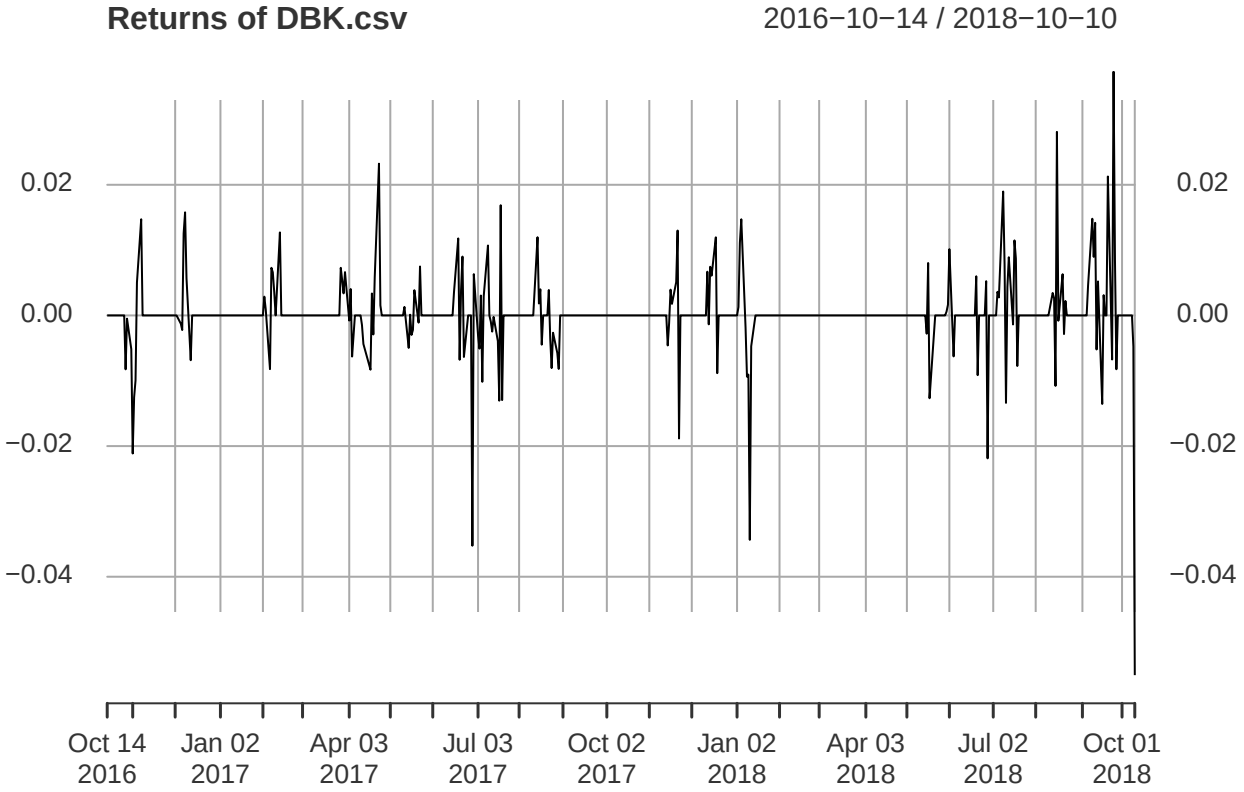
Calculations and visualizations based on returns

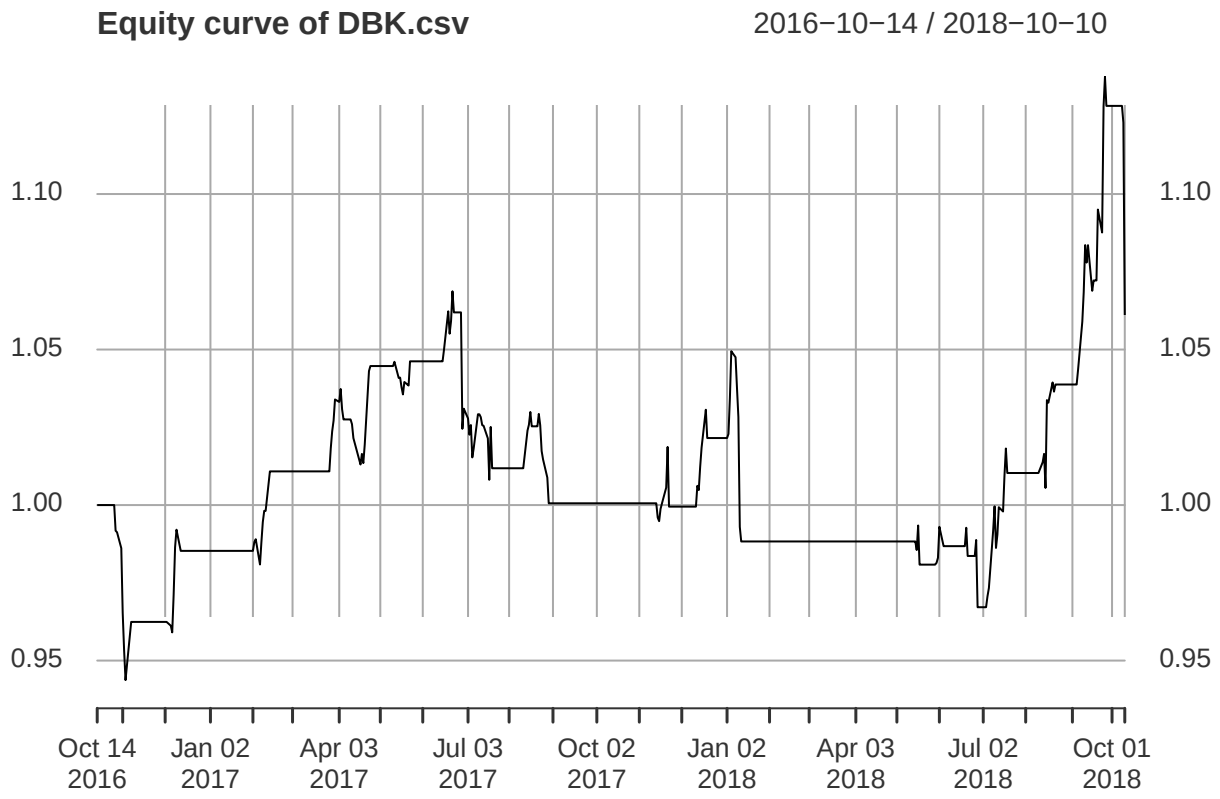
The visualization of returns, equity curve, value at risk and some other measure which can be calculated based on the returns of the trading strategy, can be observed in the following graphs. ### Returns ### Equity Curve ### Value at Risk

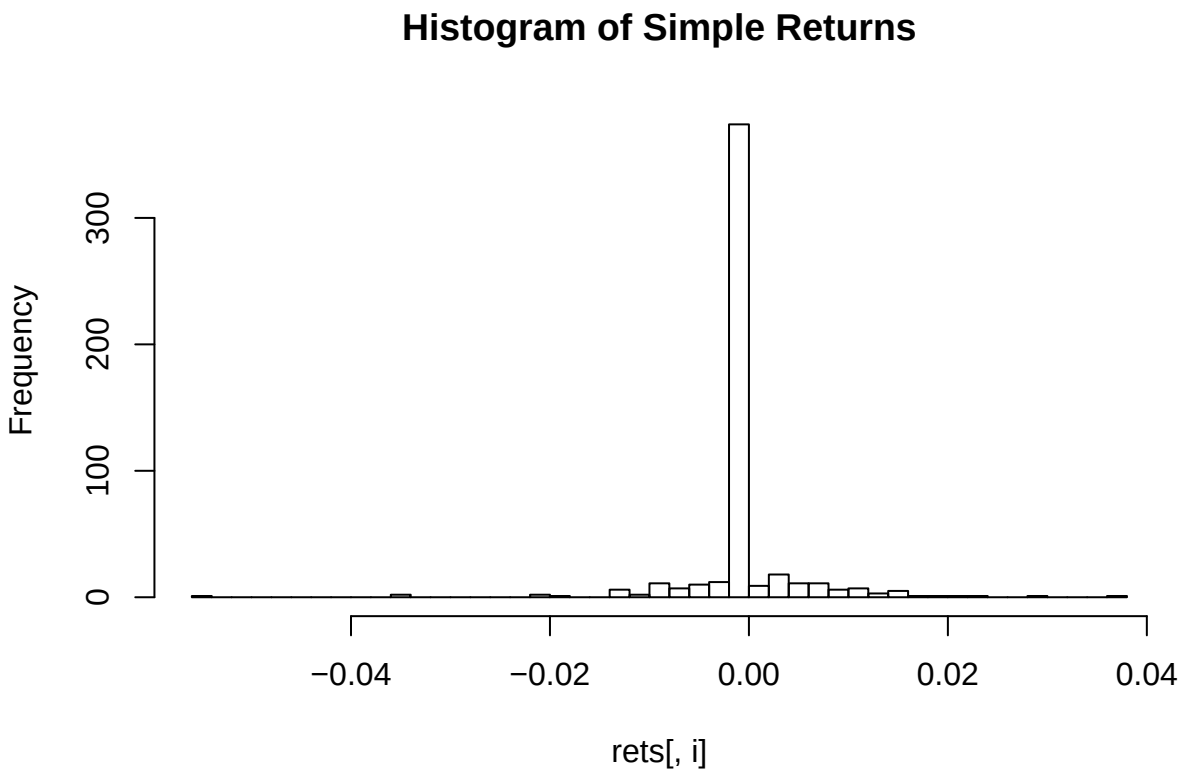

Returns of SAP.csv
2016−10−14 / 2018−10−10
0.08
0.06
0.04
0.02
0.00
−0.02
Oct 14 2016
Jan 02 2017
Apr 03 2017
Jul 03 2017
Oct 02 2017
Jan 02 2018
Apr 03 2018
Jul 02 2018
Oct 01 2018


Equity curve of SAP.csv
2016−10−14 / 2018−10−10
1.05
1.00
0.95
0.90
Oct 14 2016
Jan 02 2017
Apr 03 2017
Jul 03 2017
Oct 02 2017
Jan 02 2018
Apr 03 2018
Jul 02 2018
Oct 01 2018


Histogram of Simple Returns
Frequency
0
100
200
300
400
−0.04
−0.02
0.00
0.02
0.04
0.06
0.08
rets[, i]


Returns of DBK.csv
2016−10−14 / 2018−10−10
0.02
0.00
−0.02
−0.04
Oct 14
2016
Jan 02
2017
Apr 03
2017
Jul 03
2017
Oct 02
2017
Jan 02
2018
Apr 03
2018
Jul 02
2018
Oct 01
2018


Equity curve of DBK.csv
2016−10−14 / 2018−10−10
1.10
1.05
1.00
0.95
Oct 14 2016
Jan 02 2017
Apr 03 2017
Jul 03 2017
Oct 02 2017
Jan 02 2018
Apr 03 2018
Jul 02 2018
Oct 01 2018



Value at Risk

```
## [1] "A 10000 Euro investment in SAP.csv could lose with a probability of"
##      1%
## -227.19
## [1] "A 10000 Euro investment in DBK.csv could lose with a probability of"
##     1%
## -186.8
```

Conclusion and Suggestions