Applied Financial Economics

Jay Wei

2023-07-24

Contents

1 Fundamental Analysis

1.1 Dividend Discount Model

Fundamental value = discounted sum of future dividends

$$V_t = \frac{1}{1+r} D_{t+1} + \frac{1}{(1+r)^2} D_{t+2} + \cdots$$

where r is required rate of return and it is estimated by various methods:

- CAPM: risk-free rate + risk premium

$$r = r_f + \beta(r_M - r_f)$$

- WACC: weighted average cost of capital

$$r = \frac{E}{D+E} r_e + \frac{D}{D+E} r_d$$

1.1.1 Two special cases

- No growth of dividend ($D_{t+1} = D, \forall t$)

$$V_t = \frac{\frac{1}{1+r} D}{1 - \frac{1}{1+r}} = \frac{D}{r}$$

- Dividends grow at rate g ($D_{t+1} = (1+g)D_t$)

$$V_t = \frac{\frac{1}{1+r} D_{t+1}}{1 - \frac{1+g}{1+r}} = \frac{D_{t+1}}{r-g}, \quad r > g$$

Trading signals:

- Buy: $P_t < V_t$
- Sell: $P_t > V_t$

1.2 PE Ratio

P/E ratio: price per share divided by earning per share

$$PE = \frac{P_t}{EPS_t}$$

Trading signals:

- Buy: $PE < \delta_{\text{buy}}$
- Sell: $PE > \delta_{\text{sell}}$

Usually δ_{buy} is around 15 and δ_{sell} is 25 but this varies across industries and economy situation.

1.3 Financial Statements

1. Balance Sheet: things owns and owes at a fixed point in time

Assets = Liability + shareholders' equity

- Assets: something the company owns that has value
  - physical property: plants, equipment, inventories
  - intangible: trademark, patents
  - financial: cash, investment, account receivable
- Liability: what the company owes to others
  - obligation: loan, unpaid rent/wages/taxes
- Shareholders' equity: capital or net worth
  - money left if company is sold and liabilities are paid

Liquidity of Asset: how quickly it can be converted into cash

- Current asset: convert to cash within one year
- Non-current asset: requires more than one year to sell

Due date of Liability: when it need to be repaid

- Current liablity: to pay within one year
- Long term liablity: more than one year

2. Income Statement: money made and spent over a period of time

3. Statements of Shareholder's equity: changes in the interests of shareholders over time

Ending Retained Earnings = Beginning Retained Earnings + Net Income - Dividend

4. Cash Flow Statement: cash transactions over a period of time

1.4 Raio Analysis

1. Performance

- Dividend Yield = Dividend/Price of stock
- Dividend Payout Ratio = Dividend/Net Income
- Return on Assets (ROA) = Net Income/Total Equity
- Return on Equity (ROE) = Net Income/Total Equity
- Earnings Per Share (EPS) = (Net Income - Dividends on Preferred Stock)/Average Outstanding Shares
- Gross Profit Margin = (Revenue - Cost of Goods Sold)/Revenue

2. Liquidity

- Current Ratio = Current Asset/Current Liability
- Interest Coverage Ratio. = Earnings before Interest and Taxes/Interest Expense
- Working Capital = Current Assets - Current Liabilities

3. Activity

- Asset Turnover = Total Revenue/ Total Asset
- Inventory Turnover = Sales/ Average Inventory
- Average Collection Period = (Average Balance of Accounts Receivable x No. of Days)/Toatal Net Credit Sales

4. Financing

- Debt Ratio = Total Liability/ Total Asset
- Debt/Equity Ratio = Total Liability/ Total Equity

Signs of a good company:

- High Current Ratio: pay off current debts easily
- Appropriate Debt Ratio: not too aggressive and too conservative
- High ROA/ROE: good earning strength
- High Asset and Inventory Turnover: efficient asset and inventory management

2 Technical Analysis

```
suppressMessages(library(quantmod))
suppressMessages(library(TTR))
```

```
# Fetch U.S. symbols from the internet
# nyseSymbols <- stockSymbols(c("NASDAQ", "NYSE"), sort.by = c("Exchange", "Symbol"))
```

```
getSymbols("AAPL")
```

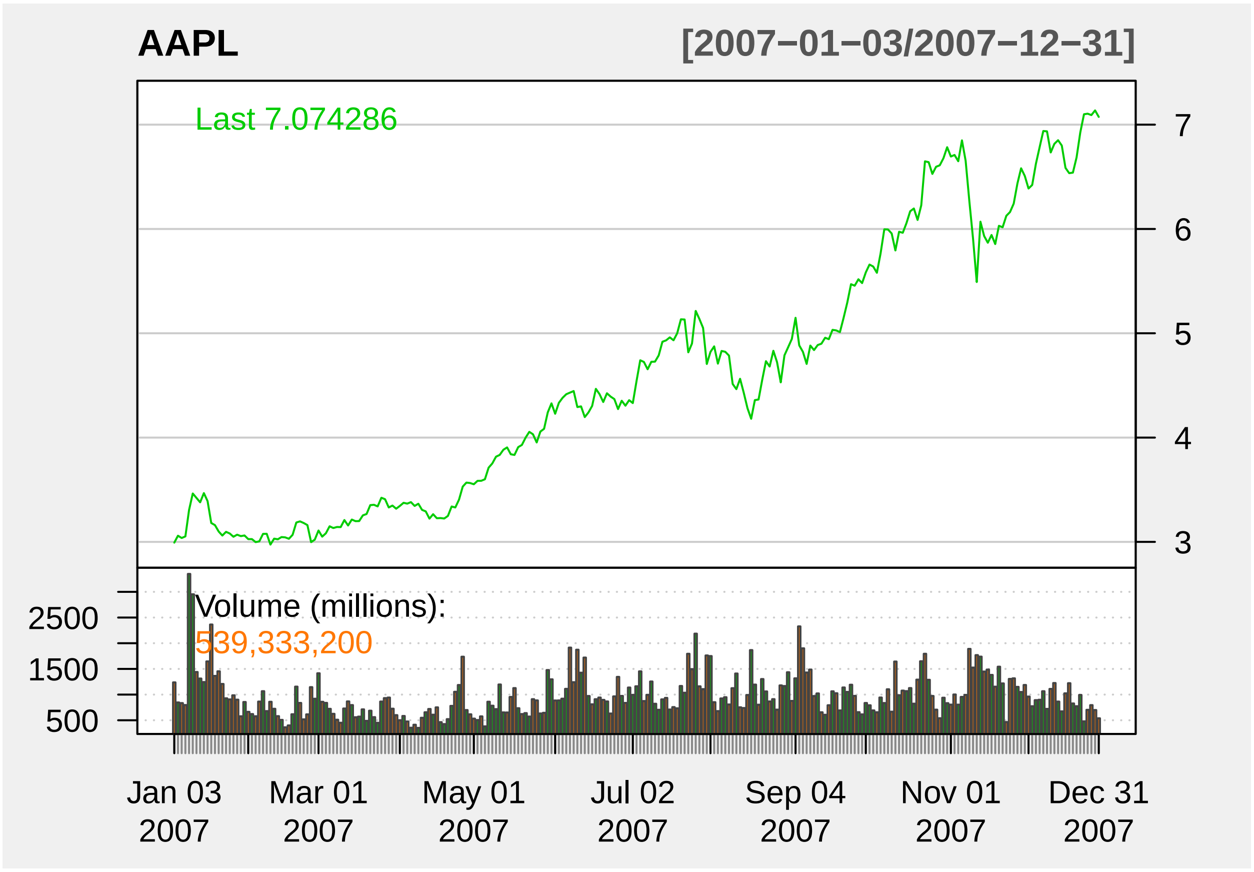
```
## [1] "AAPL"
```

```
head(AAPL, n=3)
```

```
##            AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume AAPL.Adjusted
## 2007-01-03  3.081786  3.092143 2.925000   2.992857  1238319600      2.543757
## 2007-01-04  3.001786  3.069643 2.993571   3.059286   847260400      2.600218
## 2007-01-05  3.063214  3.078571 3.014286   3.037500   834741600      2.581701
```

This is callsed OHLCVA price data. We have opening, high, low, open, close, volume and adjusted closing prices. High and low are the highest and lowest prices of the trading day. Open and close are the opening and the closing prices of the trading day. Volume is the number of shares transacted on the trading day. Adjusted closing price is the closing price that adjusts for event after market closes such as stock splits and dividend.

```
chartSeries(AAPL,
            type="line",
            subset='2007',
            theme=chartTheme('white'))
```



2.1 Trending Indicator

2.1.1 Simple Filter Rule

Simple filter rule (naive trading rule) at time t

- Buy: $r_t > \delta$
- Sell: $r_t < -\delta$

where P_t is the clsing price at time t,

$$r_t = \frac{P_t - P_{t-1}}{P_{t-1}}$$

is rate of change in price, and $\delta > 0$ is a threshold depend on risk preference and stock characteristics.

Used as a benchmark for comparative testing of other trading strategies (another benchmark is buy and hold strategy)

2.1.2 Simple Moving Average (SMA) Rules

Simple moving average over n lagged periods at time t:

$$sma_t(P, n) = \frac{P_t + P_{t-1} + ... + P_{t-n+1}}{n}$$

Trading signals:

- Buy: $r_t > \delta$
- Sell: $r_t < -\delta$

where $r_t = \frac{sma_t(P,S) - sma_t(P,L)}{sma_t(P,L)}$

and short-term (fast) $sma_t(P, S)$ and long-term (slow) $sma_t(P, L)$

Often $S \leq 5$ and $L \geq 50$

```
mySMA <- function (price,n){
  sma <- c()
  sma[1:(n-1)] <- NA
  for (i in n:length(price)){
    sma[i]<-mean(price[(i-n+1):i])
  }
  sma <- reclass(sma,price)
  return(sma)
}
```

```
mysma <- mySMA(Cl(AAPL), n=20)
tail(mysma, n=3)
```

```
##                 [,1]
## 2023-07-26 191.7770
## 2023-07-27 191.9755
## 2023-07-28 192.2875
```

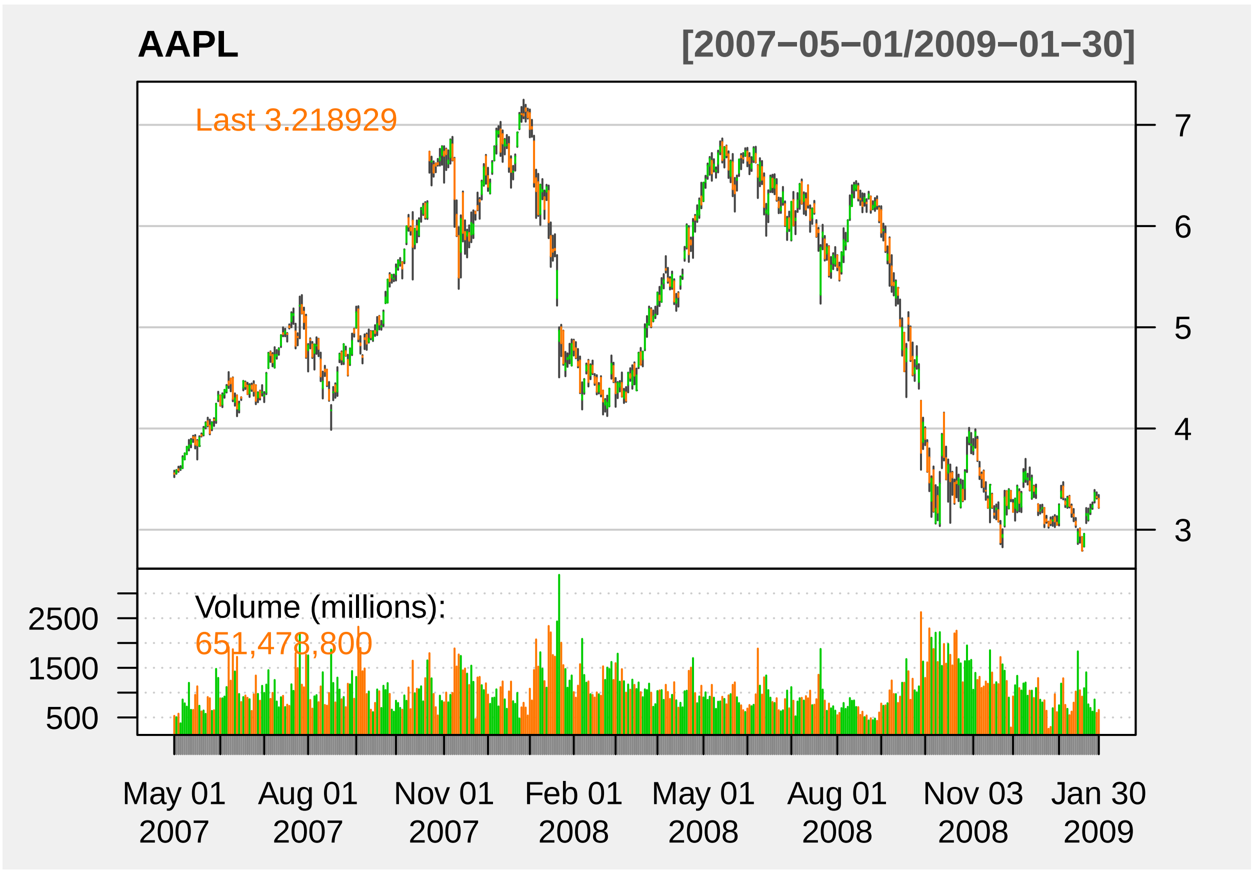
```
# use SMA() in TTR package
sma <- SMA(Cl(AAPL), n=20)
tail(sma, n=3)
```

```
##                  SMA
## 2023-07-26 191.7770
## 2023-07-27 191.9755
## 2023-07-28 192.2875
```

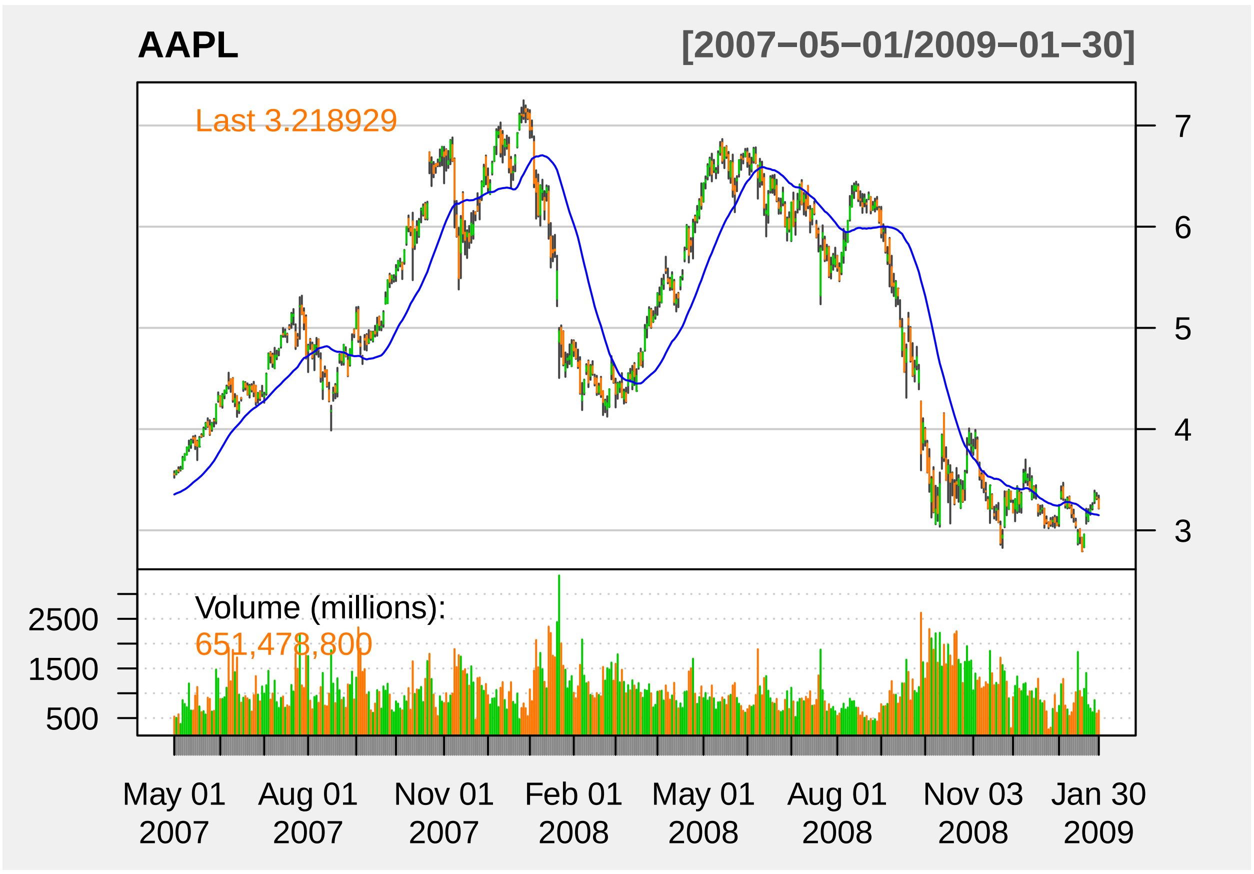
Buy signal arises when a short-run SMA crosses from below to above a long-run SMA.

Sell signal arrises when a short-run SMA crosses from above to above a long-run SMA.

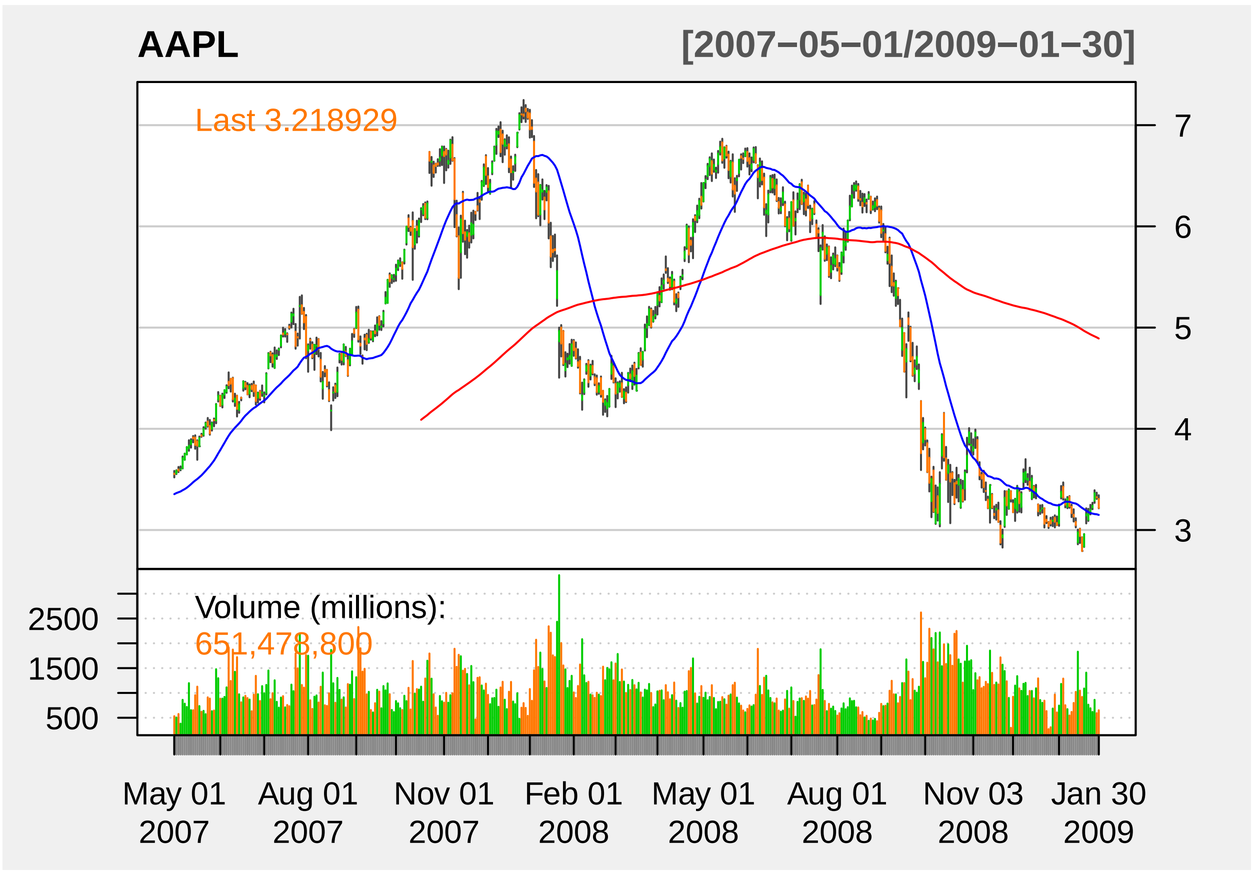
```
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white')
            )
```



```
addSMA(n=30,on=1,col = "blue")
```



```
addSMA(n=200,on=1,col = "red")
```



2.1.3 Exponential Moving Average Rule

EMA with n lagged period at time t:

$$
\begin{aligned}
ema_t(P,n) &= \beta P_t + \beta(1-\beta)P_{t-1} + \beta(1-\beta)^2 P_{t-2} + \cdots && (1)\\
&= \beta P_t + (1-\beta) ema_{t-1}(P,n) && (2)
\end{aligned}
$$

where the smoothing coefficient β is usually $\beta = \frac{2}{n+1}$

First EMA: use SMA

Subsequent EMAs: update formula

Trading signals:

- Buy: $r_t > \delta$
- Sell: $r_t < -\delta$

where $r_t = \frac{ema_t(P,S) - ema_t(P,L)}{ema_t(P,L)}$

```r
myEMA <- function (price,n){
  ema <- c()
  ema[1:(n-1)] <- NA
  ema[n]<- mean(price[1:n]) # use SMA for first EMA
  beta <- 2/(n+1)
  for (i in (n+1):length(price)){
    ema[i]<-beta * price[i] +
      (1-beta) * ema[i-1]
  }
  ema <- reclass(ema,price)
  return(ema)
}
```

```r
ema <-myEMA(Cl(AAPL),n=20)
tail(ema,n=3)
```

```
##                [,1]
## 2023-07-26 191.0900
## 2023-07-27 191.2929
## 2023-07-28 191.7250
```

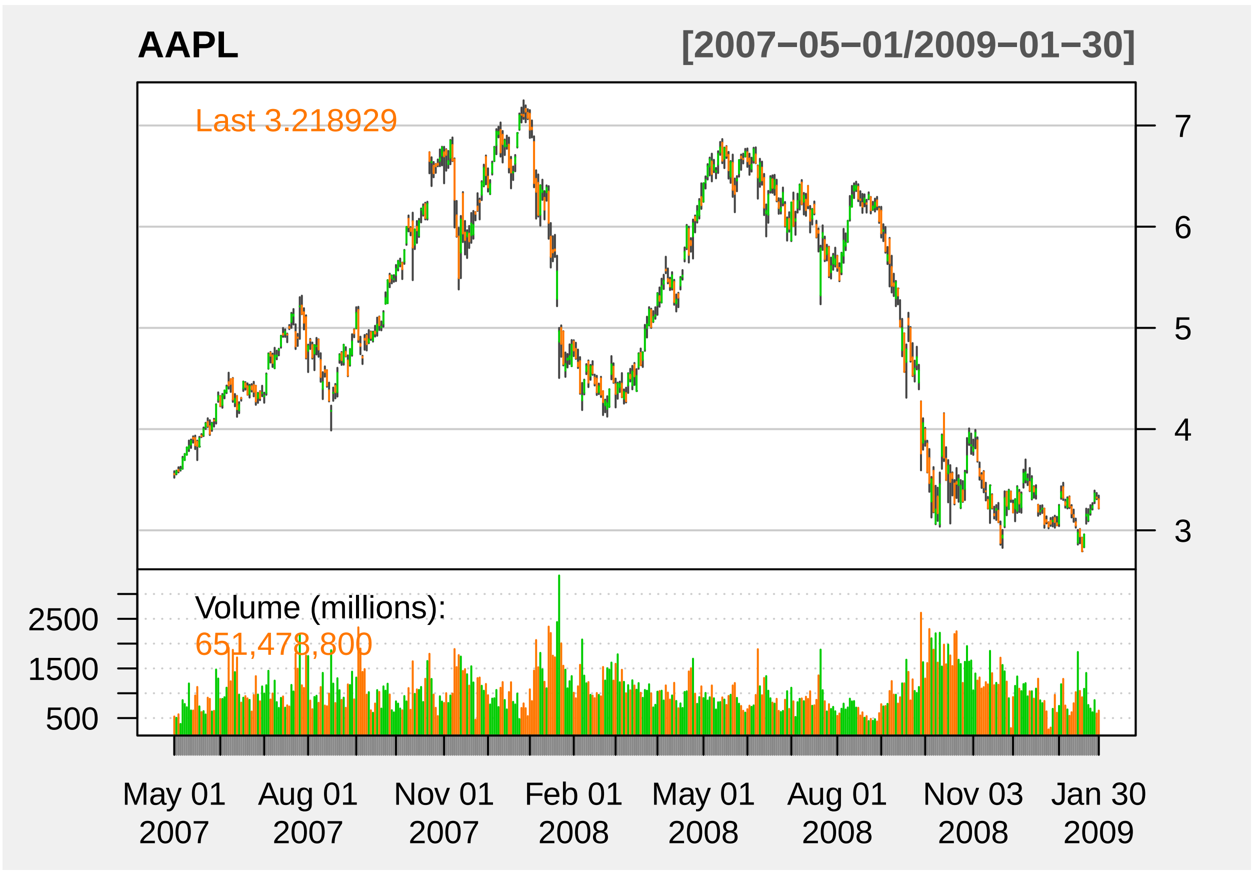
```r
# use EMA() in TTR package
ema <-EMA(Cl(AAPL),n=20)
tail(ema,n=3)
```

```
##                 EMA
## 2023-07-26 191.0900
## 2023-07-27 191.2929
## 2023-07-28 191.7250
```

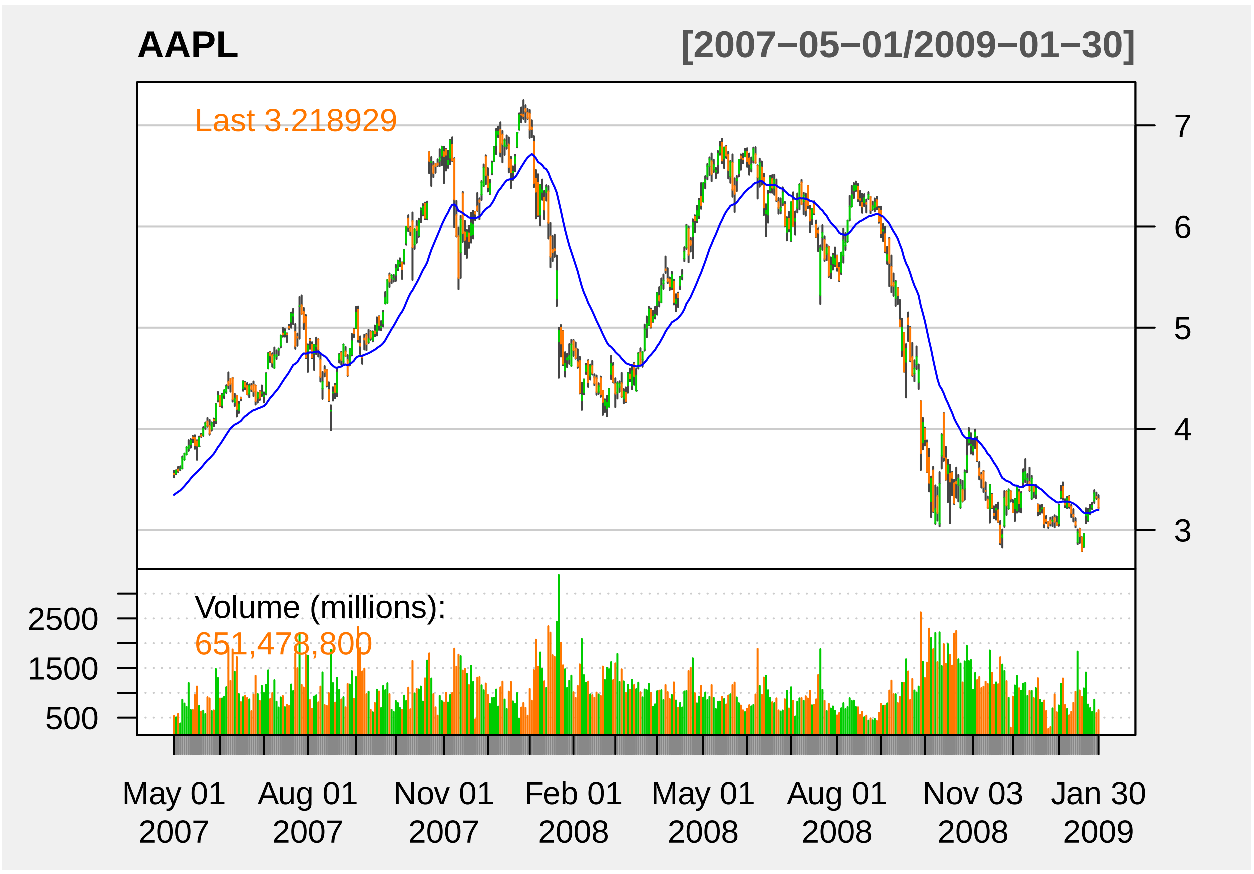
Buy signal arises when a short-run EMA crosses from below to above a long-run EMA.

Sell signal arrises when a short-run EMA crosses from above to above a long-run EMA.

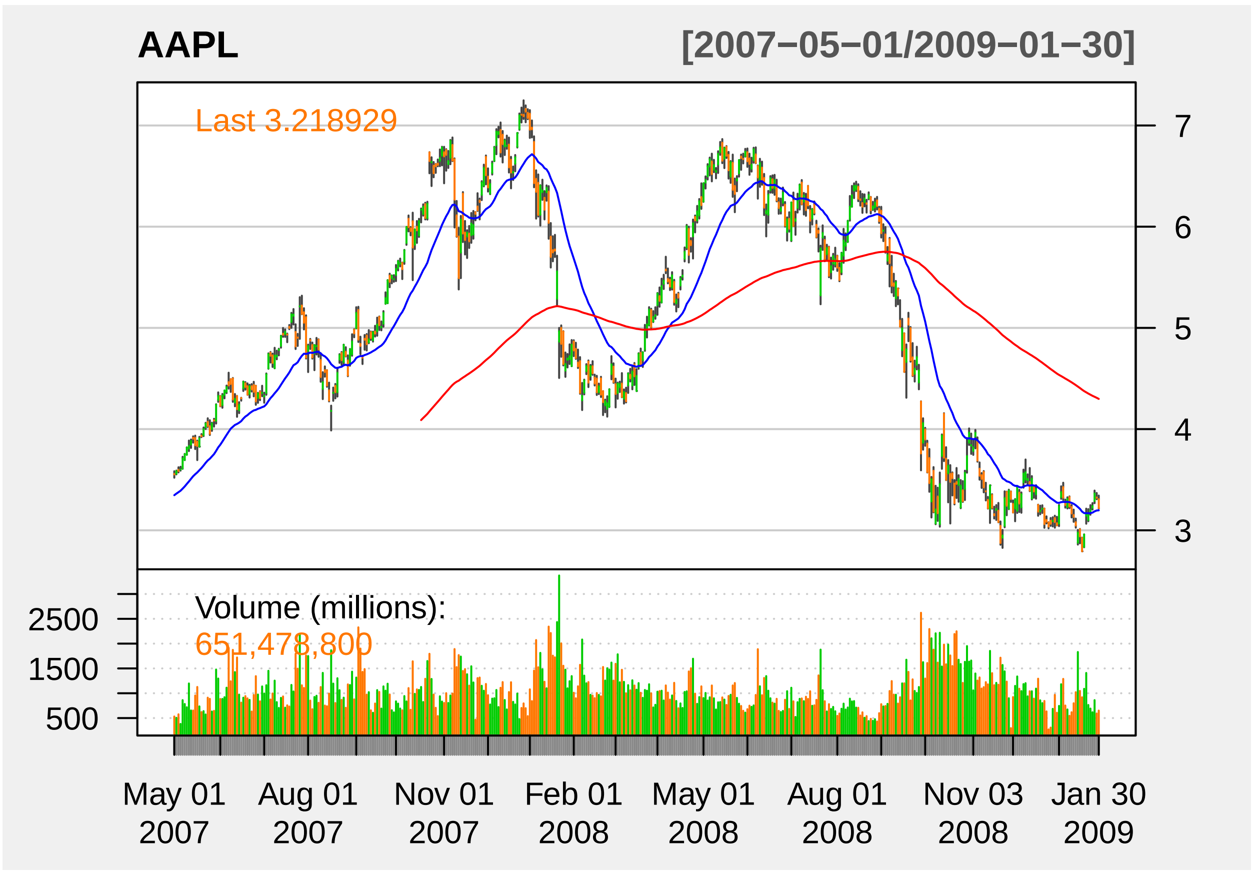
```r
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white'))
```



```r
addEMA(n=30,on=1,col = "blue")
```



```r
addEMA(n=200,on=1,col = "red")
```



2.1.4 Moving Average Convergence/Divergence (MACD)

MACD is the difference between a fast EMA and a slow EMA:

$$MACD_t(S,L) = ema_t(P,S) - ema_t(P,L)$$

Its exponential smoothing is called the signal line:

$$sig_t(S,L,K) = ema_t(MACD(S,L),K)$$

Usually, $K = 9$ days, $S = 12$ days and $L = 26$ days. Note that MACD sometimes appear as the percetnage format

$$MACD_t(S,L) = \frac{ema_t(P,S) - ema_t(P,L)}{ema_t(P,L)}$$

Trading signals:

- Buy: $MACD_t > sig_t$ and $MACD_{t-k} < sig_{t-k}, \quad k = 1, ..., K$
  - or MACD crosses from below to above the signal line
- Sell: $MACD_t < sig_t$ and $MACD_{t-k} > sig_{t-k}, \quad k = 1, ..., K$
  - or MACD crosses from above to below the signal line

```
myMACD <- function (price,S,L,K){
  MACD <- EMA(price,S) - EMA(price,L)
  signal <- EMA(MACD,K)
  output <- cbind(MACD,signal)
  colnames(output) <- c("MACD","signal")
  return(output)
}
```

```
macd <- myMACD(Cl(AAPL), 12, 26,9)
tail(macd,n=5)
```

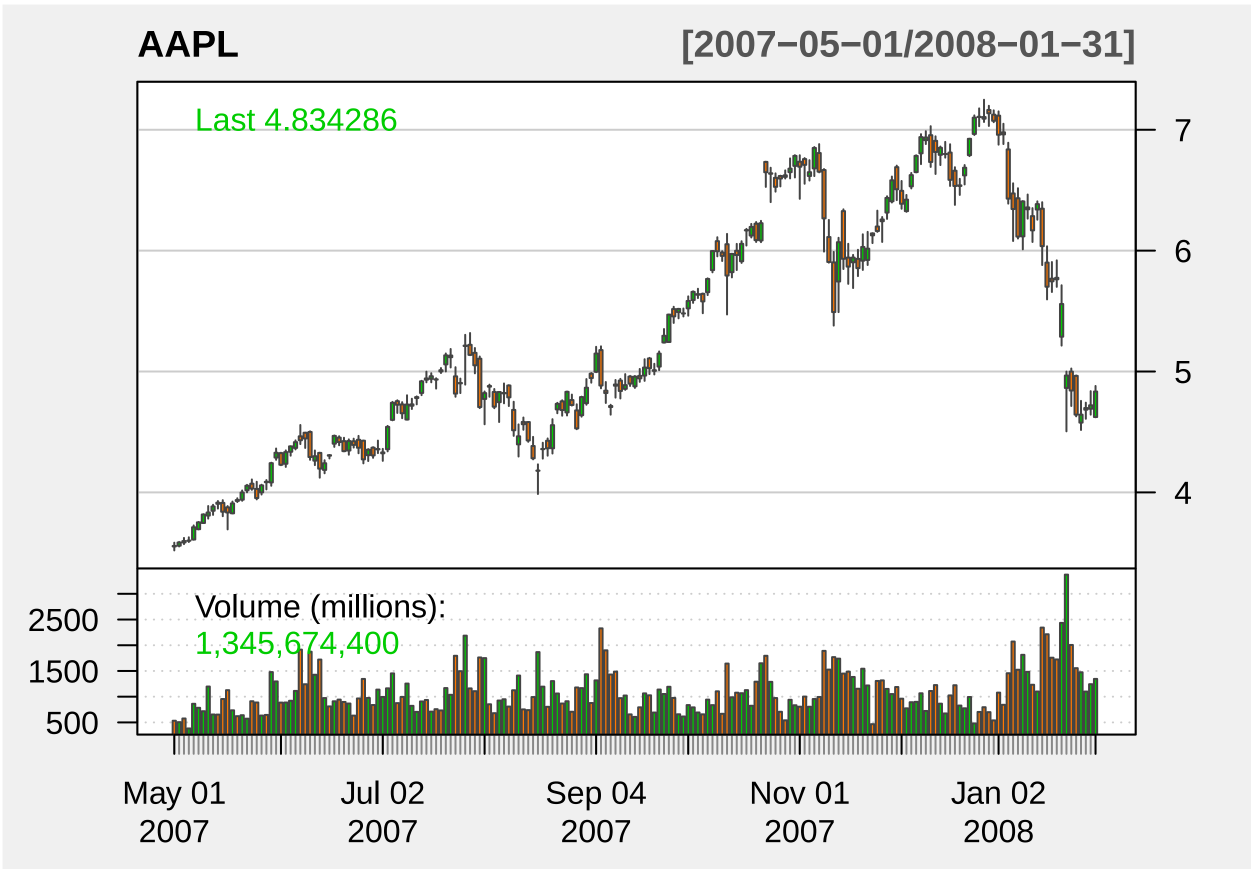
```
##                  MACD   signal
## 2023-07-24 2.721609 3.035871
## 2023-07-25 2.657851 2.960267
## 2023-07-26 2.647808 2.897775
## 2023-07-27 2.507658 2.819752
## 2023-07-28 2.577481 2.771298
```

```
macd <- MACD(Cl(AAPL), nFast=12, nSlow=26,
             nSig=9, percent=FALSE)
tail(macd,n=5)
```

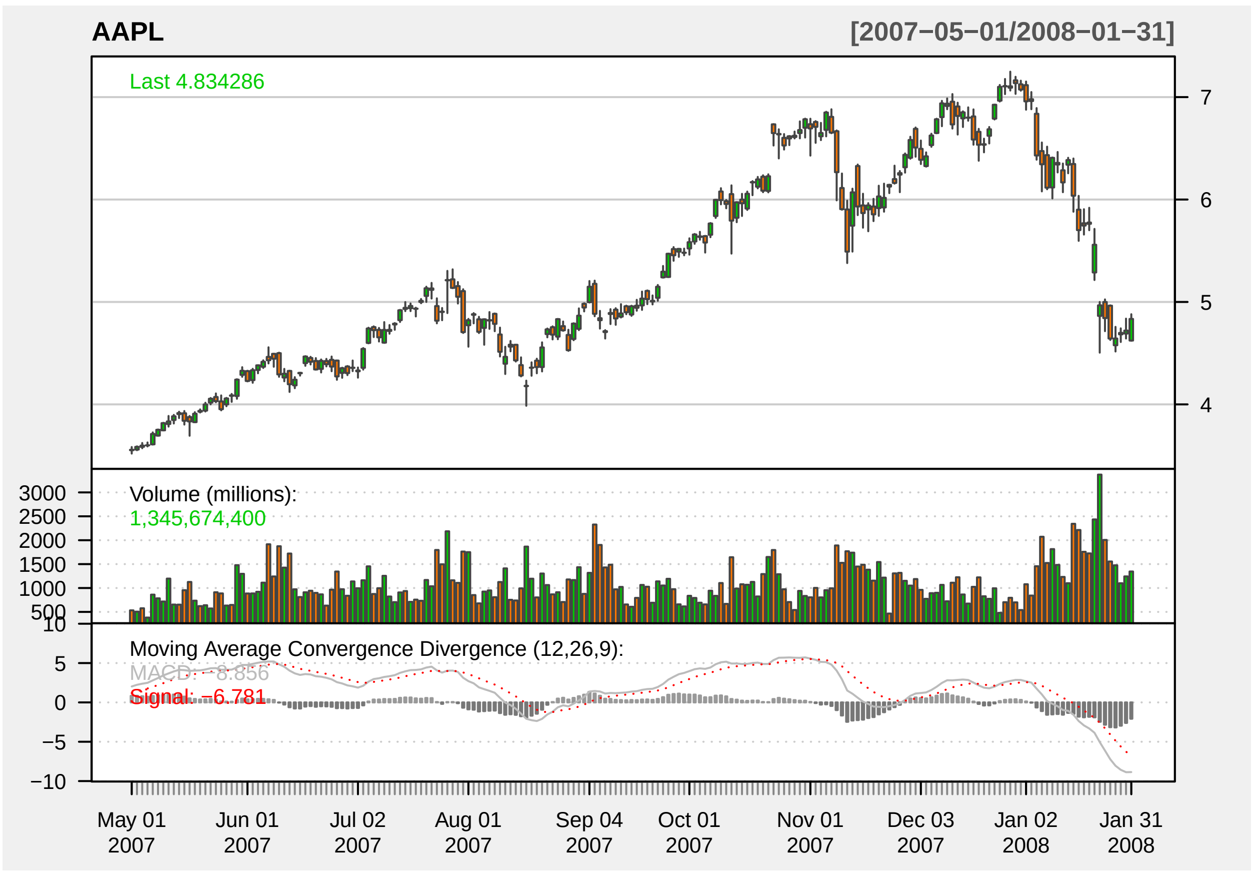
```
##                  macd   signal
## 2023-07-24 2.721609 3.035871
## 2023-07-25 2.657851 2.960267
## 2023-07-26 2.647808 2.897775
## 2023-07-27 2.507658 2.819752
## 2023-07-28 2.577481 2.771298
```

where `nFast`, `nSlow`, and `nSig` are S, L, and K in our formula. The `percent` is whether MACD is in percentage form or in difference form.

```
chartSeries(AAPL,
            subset='2007-05::2008-01',
            theme=chartTheme('white'))
```



```
addMACD(fast=12,slow=26,signal=9,type="EMA")
```



2.2 Momentum Indicator

n-day momentum at day t equals $M_t(n) = P_t - P_{t-n}$

Trading signals:

- Buy: $M_t(n) > 0$ and $M_{t-1}(n)$, ..., $M_{t-K}(n) < 0$
- Sell: $M_t(n) < 0$ and $M_{t-1}(n)$, ..., $M_{t-K}(n) > 0$

where K is a sensitivity parameter.

```
mymom <- function (price,n){
  mom <- rep(0,n)
  N <- nrow(price)
  Lprice <- Lag(price,n)
  for (i in (n+1):N){
    mom[i]<-price[i]-Lprice[i]
  }
  mom <- reclass(mom,price)
  return(mom)
}
```

We calculate 2-day momentum based on closing price of AAPL.

```
M <- mymom(Cl(AAPL), n=2)
head (M,n=5)
```

```
##                  [,1]
## 2007-01-03   0.000000
## 2007-01-04   0.000000
## 2007-01-05  0.044643
## 2007-01-08 -0.006786
## 2007-01-09  0.268571
```

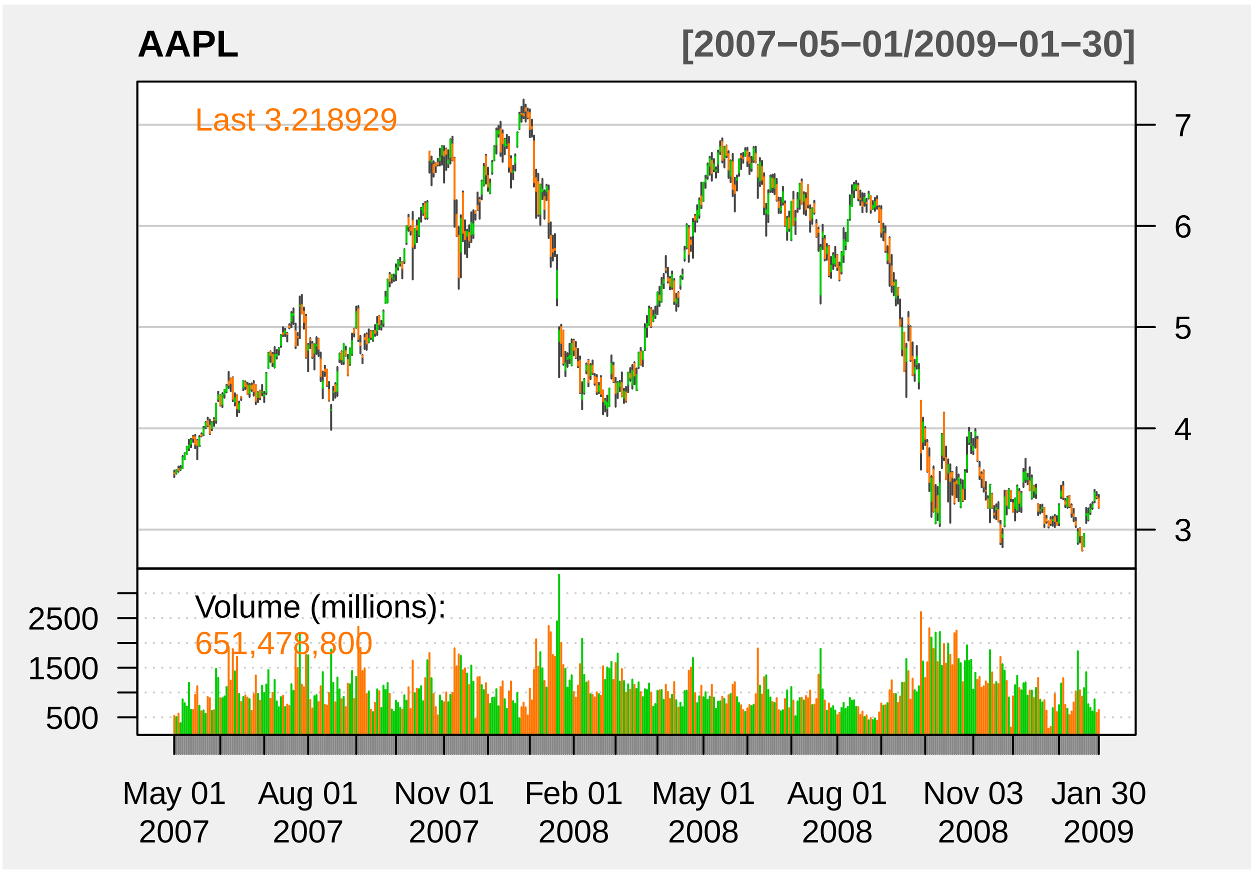
```
# using TTR package
M <- momentum(Cl(AAPL), n=2)
head (M,n=5)
```

```
##            AAPL.Close
## 2007-01-03         NA
## 2007-01-04         NA
## 2007-01-05   0.044643
## 2007-01-08  -0.006786
## 2007-01-09   0.268571
```

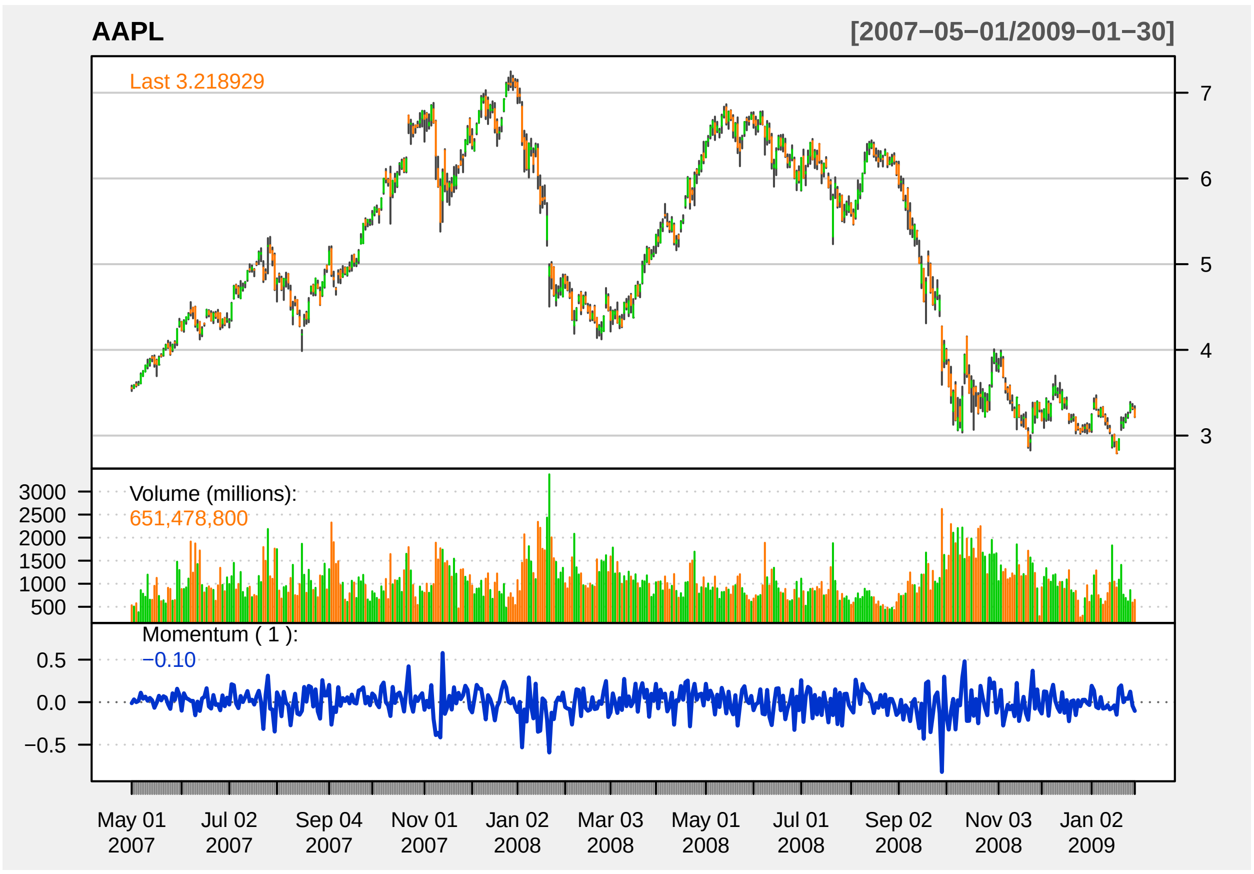
Buy signal arises when momentum changes from negative to positive.

Sell signal arises when momentum changes from positive to negative.

```
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white'))
```



```
addMomentum(n=1)
```



2.2.1 Rate of Change (ROC)

Rate of Change (ROC) with K-day momentum at day t equals to return of K days.

For discrete type, we have

$$ROC_t(K) = \frac{P_t - P_{t-k}}{P_{t-k}}$$

For continuous type, we have

$$ROC_t(K) = \log(P_t) - \log(P_{t-k})$$

```r
myROC <- function (price,n){
  roc <- rep(0,n)
  N <- nrow(price)
  Lprice <- Lag(price,n)
  for (i in (n+1):N){
    roc[i]<-(price[i]-Lprice[i])/Lprice[i]
  }
  roc <- reclass(roc,price)
  return(roc)
}
```

```r
roc <- myROC(Cl(AAPL),n=2)
tail(roc,n=3)
```

```
##                    [,1]
## 2023-07-26  0.009079118
## 2023-07-27 -0.002065871
## 2023-07-28  0.006838057
```

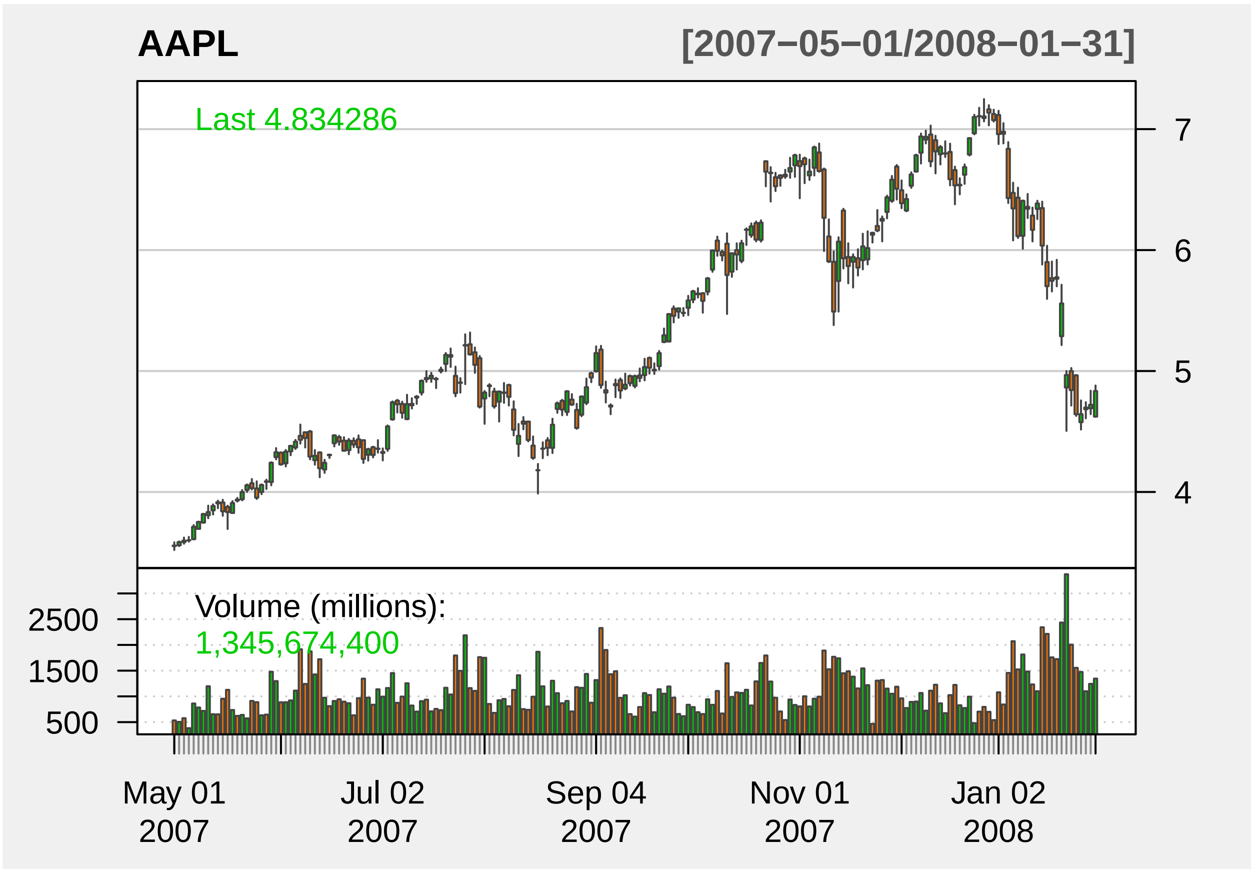
```r
roc <- ROC(Cl(AAPL),type="discrete",n=2)
tail(roc,n=3)
```

```
##               AAPL.Close
## 2023-07-26  0.009079118
## 2023-07-27 -0.002065871
## 2023-07-28  0.006838057
```

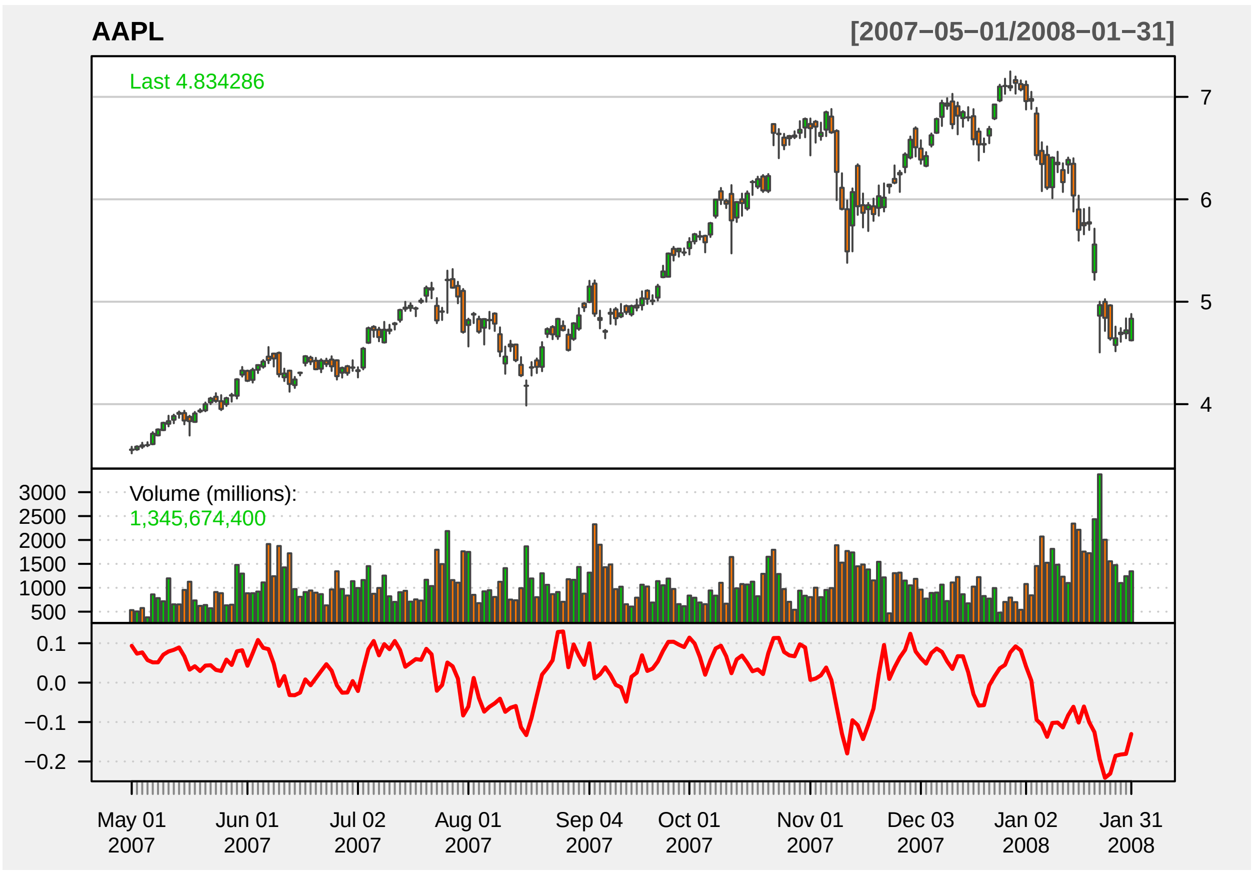
Buy signal arises when ROC changes from negative to positive.

Sell signal arises when ROC changes from positive to negative.

```r
chartSeries(AAPL,
            subset='2007-05::2008-01',
            theme=chartTheme('white'))
```



```r
addROC(n=7)
```



2.2.2 Simple Relative Strength Index (RSI)

The calculation of RSI requires several steps:

1. Whether price has gone up or down each day
2. Relative strength (RS): the ratio of the (simple or exponential) average numbers of up day to the average of down day
3. Relative strength index (RSI): normalize RS to the scale from 0 to 100.

Upward and downward indicators, U_t and D_t, are

$$U_t = \begin{cases} 1, & P_t > P_{t-1} \\ 0, & P_t \le P_{t-1} \end{cases}, \quad D_t = \begin{cases} 0, & P_t \ge P_{t-1} \\ 1, & P_t < P_{t-1} \end{cases}$$

$up_t(N)$ and $down_t(N)$ are average numbers of upward moves and downward moves of closing price of past n days.

$$up_t(n) = \frac{U_t + U_{t-1} + ... + U_{t-n+1}}{n} \tag{3}$$

$$down_t(n) = \frac{D_t + D_{t-1} + ... + D_{t-n+1}}{n} \tag{4}$$

For a given period n (= 14 days), RS is ratio of average up days to down days.

$$RS_t(n) = \frac{up_t(n)}{down_t(n)}$$

Finally, RSI normalizes RS between 0 to 100.

$$RSI_t(n) = 100\frac{RS_t(n)}{1+RS_t(n)}$$

Trading signals:

- Buy: $RSI_t(n) < 30$
- Sell: $RSI_t(n) > 70$

In calculation, we usually consider directly:

$$RSI_t(n) = 100\frac{up_t(n)}{up_t(n)+down_t(n)}$$

```r
myRSI <- function (price,n){
  N <- length(price)
  U <- rep(0,N)
  D <- rep(0,N)
  rsi <- rep(NA,N)
  Lprice <- Lag(price,1)
  for (i in 2:N){
    if (price[i]>=Lprice[i]){
      U[i] <- 1
    } else {
      D[i] <- 1
    }
    if (i>n){
      AvgUp <- mean(U[(i-n+1):i])
      AvgDn <- mean(D[(i-n+1):i])
      rsi[i] <- AvgUp/(AvgUp+AvgDn)*100
      }
    }
  rsi <- reclass(rsi, price)
  return(rsi)
}
```

```r
rsi <- myRSI(Cl(AAPL), n=14)
tail(rsi,n=3)
```

```
##                  [,1]
## 2023-07-26 57.14286
## 2023-07-27 57.14286
## 2023-07-28 64.28571
```

```r
rsi <- RSI(Cl(AAPL), SMA, n=14)
tail(rsi,n=3)
```

```
##                   rsi
## 2023-07-26 57.91642
## 2023-07-27 57.40959
## 2023-07-28 70.41853
```

Note that the third input is a function. You may use SMA or EMA.

Further notice that this is different from what we have with `myRSI`. It is because their upward and downward indicators use numerical values instead:

$$U_t = \begin{cases} P_t - P_{t-1}, & P_t > P_{t-1} \\ 0, & P_t \leq P_{t-1} \end{cases}$$

and

$$D_t = \begin{cases} 0, & P_t \geq P_{t-1} \\ P_{t-1} - P_t, & P_t < P_{t-1} \end{cases}$$

Hence, we need to modify our code:

```r
myRSI.mod <- function (price,n){
  N <- length(price)
  U <- rep(0,N)
  D <- rep(0,N)
  rsi <- rep(NA,N)
  Lprice <- Lag(price,1)
  for (i in 2:N){
    if (price[i]>=Lprice[i]){
      U[i] <- price[i]- Lprice[i]
    } else{
      D[i] <- Lprice[i]- price[i]
    }
    if (i>n){
      AvgUp <- mean(U[(i-n+1):i])
      AvgDn <- mean(D[(i-n+1):i])
      rsi[i] <- AvgUp/(AvgUp+AvgDn)*100
      }
    }
  rsi <- reclass(rsi, price)
  return(rsi)
}
```

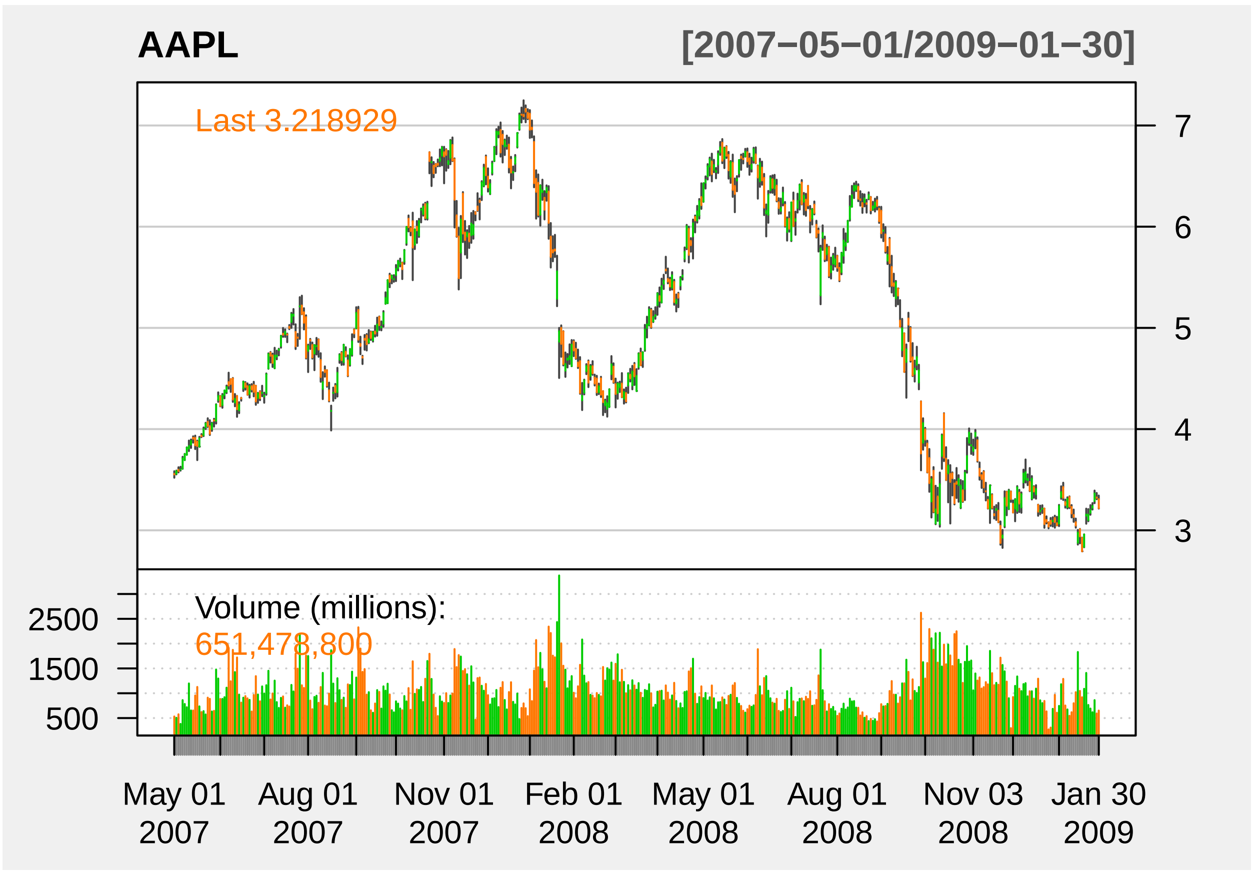
```r
rsi <- myRSI.mod(Cl(AAPL), n=14)
tail(rsi,n=3)
```

```
##                [,1]
## 2023-07-26 57.91642
## 2023-07-27 57.40959
## 2023-07-28 70.41853
```

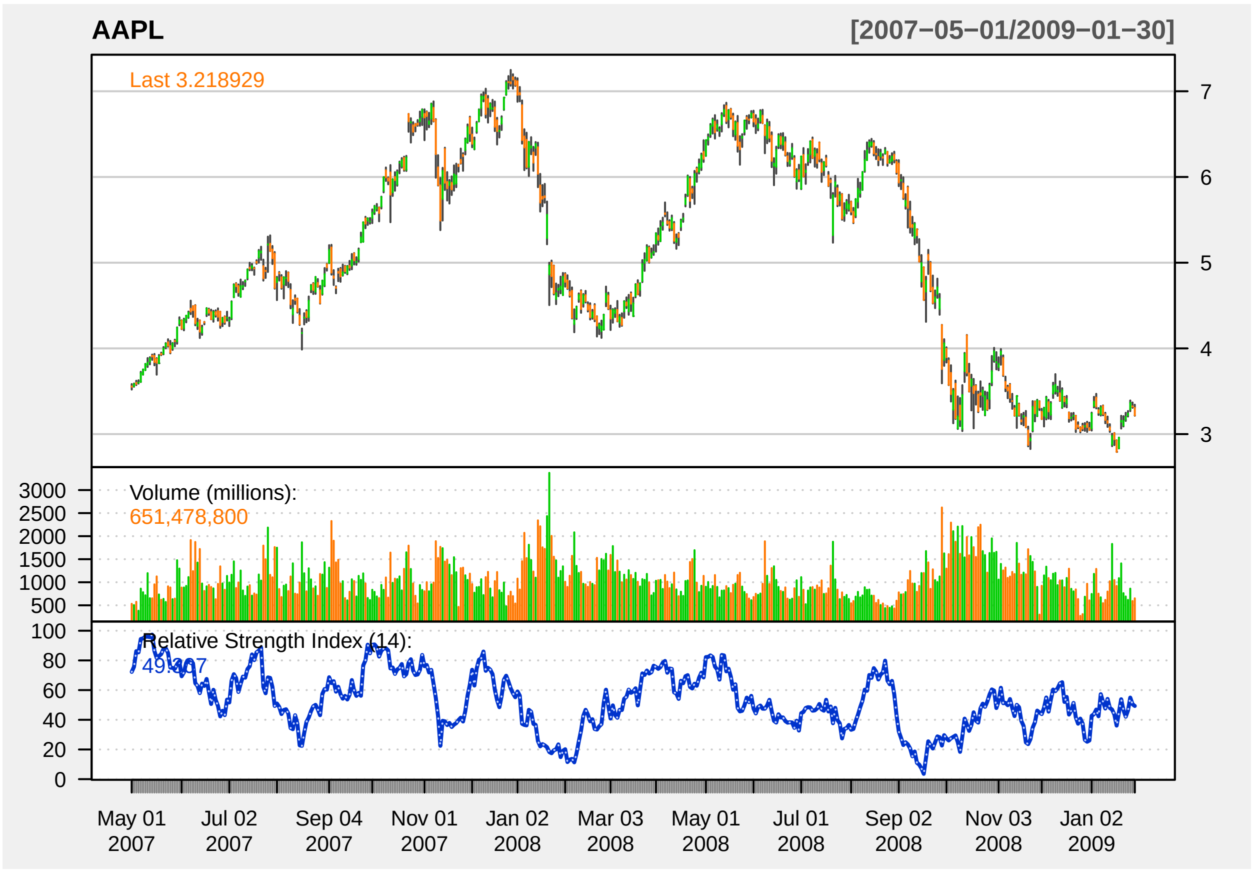
Buy signal arises when RSI is less than 30.

Sell signal arrises when RSI is higher than 30.

```r
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white'))
```



```r
addRSI(n=14,maType="SMA")
```



2.2.3 Smoothed Relative Strength Index (RSI)

Both $up_t(n)$ and $down_t(n)$ are exponentially smoothed:

$$up_t(n) = (1-\beta)up_{t-1}(n) + \beta U_t \tag{5}$$
$$down_t(n) = (1-\beta)down_{t-1}(n) + \beta D_t \tag{6}$$

where $\beta = \frac{2}{n+1}$

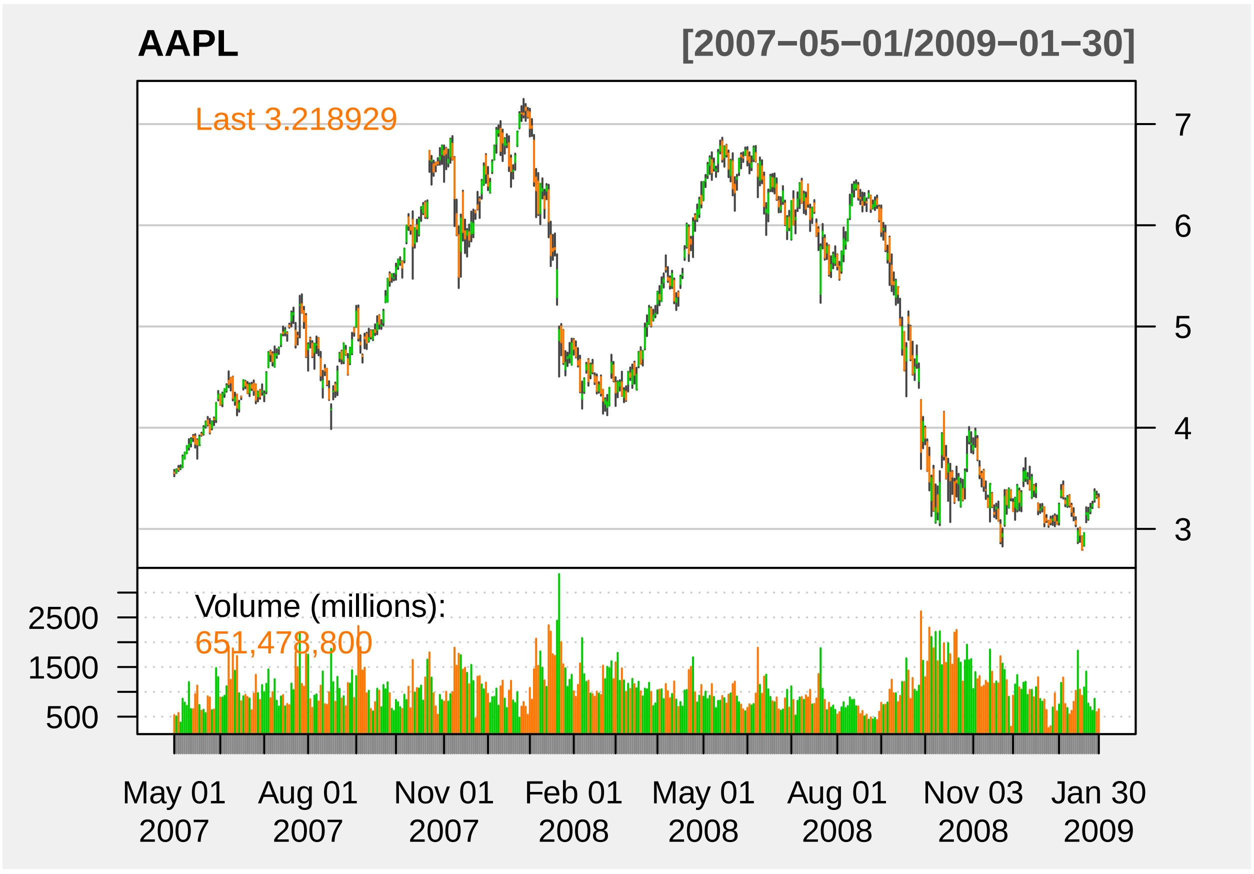
First up/down: average of U_t and D_t

Subsequent up/down: Follow the formula

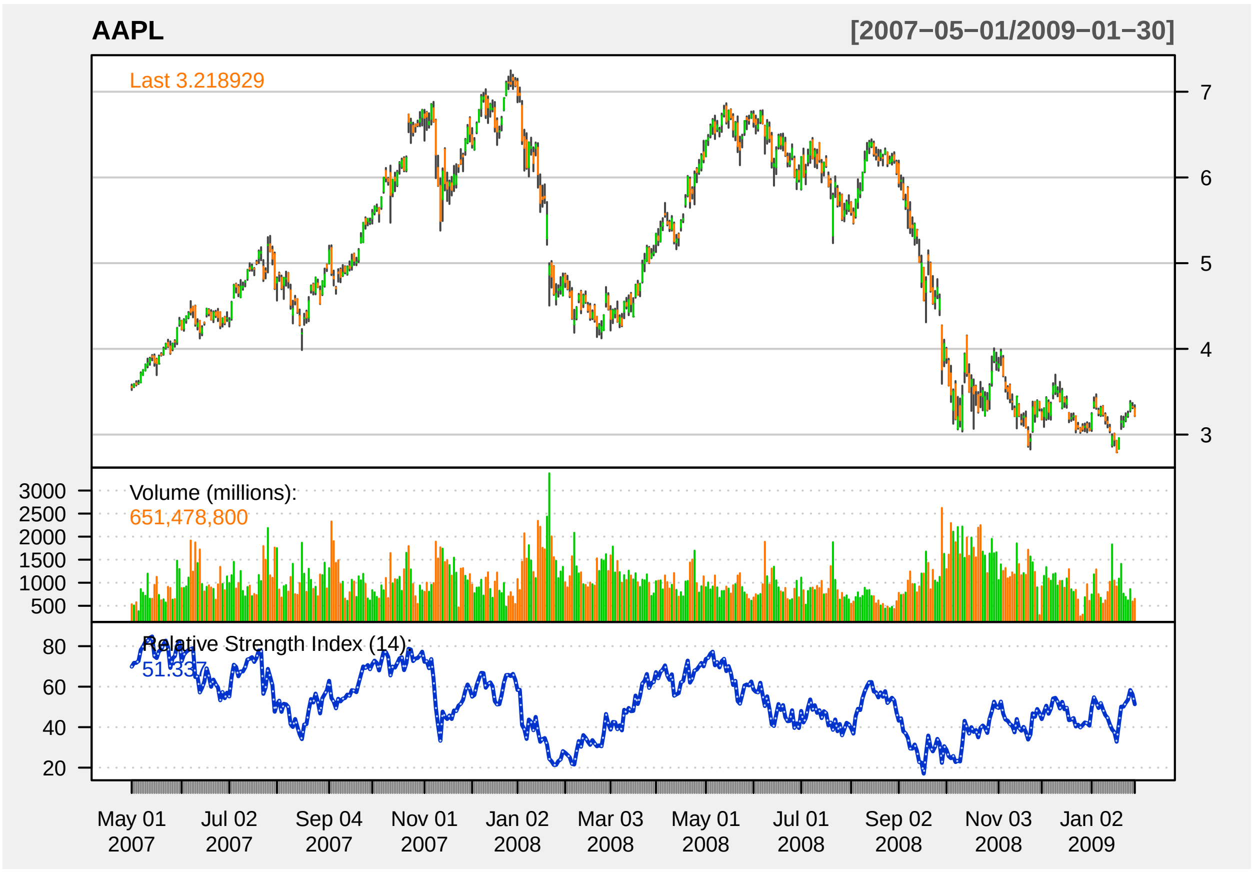
Trading signals:

- Buy: $RSI_t(n) < 30$
- Sell: $RSI_t(n) > 70$

```r
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white'))
```



```r
addRSI(n=14,maType="EMA")
```



2.3 Volatility Indicator

2.3.1 Channel Breakouts

A channel (or band) is an area that surrounds a trend within which price movement does not indicate formation of a new trend.

Trading signal from a n-day Channel Breakout is:

- Buy: $P_t > (1+B)M_{t-1}(n)$
- Sell: $P_t < (1-B)m_{t-1}(n)$

where $m_t(n) = \min(P_t, ..., P_{t-n})$, $M_t(n) = \max(P_t, ..., P_{t-n})$, and B is the channel bandwidth where $0 < B < 1$

2.3.2 Bollinger band

For Bollinger band, channel is drawn over a trend line plus/minus 2 standard deviation (SD).

- Trend line: SMA or EMA of price (price can be closing price or average of high, low, and close)
- Bandwidth: determined by asset volatility

$$B_t(n) = k \times stdev(P_t, n)$$

where usually we have $k = 2$ and $n = 20$

The lower and upper bands are $up_t = P_t + B_t(n)$ and $down_t = P_t - B_t(n)$

To denote the location of the price, we use

$$\%B = \frac{P_t - down_t}{up_t - down_t}$$

where it is greater than 1 if it is above the upper band, and less than 0 when it is below the lower band.

Trading Signal based on SMA:

- Buy: $P_t > sma_t(P,n) + B_t(n)$
  - Buy signal arises when price is above the band.
- Sell: $P_t < sma_t(P,n) - B_t(n)$
  - Sell signal arises when price is below the band.

Similar formula based on EMA.

```
myBBands <- function (price,n,sd){
  mavg <- SMA(price,n)
  sdev <- rep(0,n)
  N <- nrow(price)
  for (i in (n+1):N){
    sdev[i]<- sd(price[(i-n+1):i])
  }
  up <- mavg + sd*sdev
  dn <- mavg - sd*sdev
  pctB <- (price - dn)/(up - dn)
  output <- cbind(dn, mavg, up, pctB)
  colnames(output) <- c("dn", "mavg", "up",
        "pctB")
  return(output)
}
```

```
bb <-myBBands(Cl(AAPL),n=20,sd=2)
tail(bb,n=5)
```

```
##                  dn     mavg       up      pctB
## 2023-07-24 186.0944 191.0375 195.9806 0.6732201
## 2023-07-25 187.2004 191.4550 195.7096 0.7544292
## 2023-07-26 187.6308 191.7770 195.9232 0.8283762
## 2023-07-27 187.9607 191.9755 195.9903 0.6549886
## 2023-07-28 188.0877 192.2875 196.4873 0.9217424
```

```
# use TTR package
bb <-BBands(Cl(AAPL),n=20, sd=2)
tail(bb,n=5)
```

```
##                  dn     mavg       up      pctB
## 2023-07-24 186.2195 191.0375 195.8555 0.6777201
## 2023-07-25 187.3081 191.4550 195.6019 0.7610388
## 2023-07-26 187.7358 191.7770 195.8182 0.8369069
## 2023-07-27 188.0623 191.9755 195.8887 0.6590149
## 2023-07-28 188.1940 192.2875 196.3810 0.9326986
```

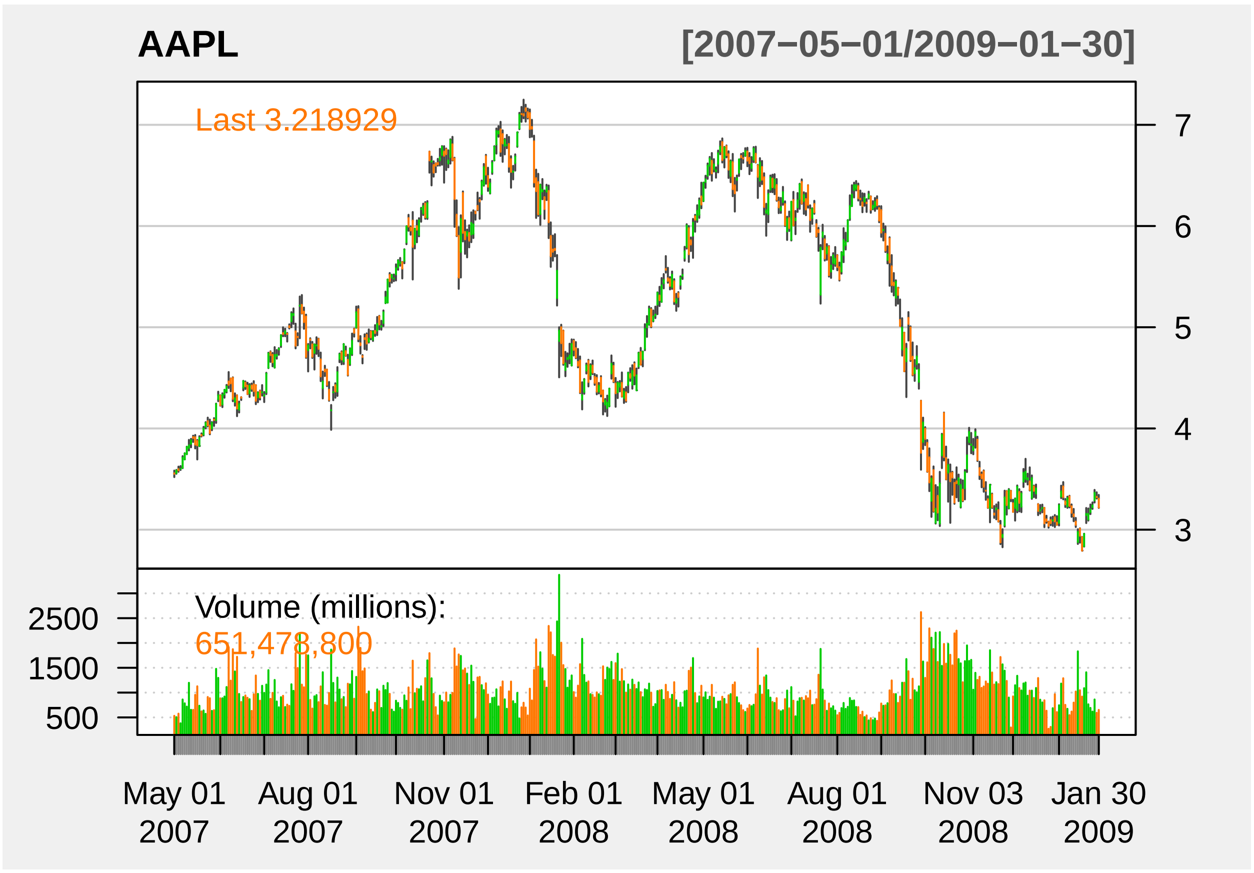
Note that it is different from above. It turns out that the standard deviation used is population instead of sample one. We only need to correct it by the factor of $\sqrt{\frac{n-1}{n}}$

```
myBBands.mod <- function (price,n,sd){
  mavg <- SMA(price,n)
  sdev <- rep(0,n)
  N <- nrow(price)
  for (i in (n+1):N){
    sdev[i]<- sd(price[(i-n+1):i])
  }
  sdev <- sqrt((n-1)/n)*sdev
  up <- mavg + sd*sdev
  dn <- mavg - sd*sdev
  pctB <- (price - dn)/(up - dn)
  output <- cbind(dn, mavg, up, pctB)
  colnames(output) <- c("dn", "mavg", "up",
        "pctB")
  return(output)
}
```

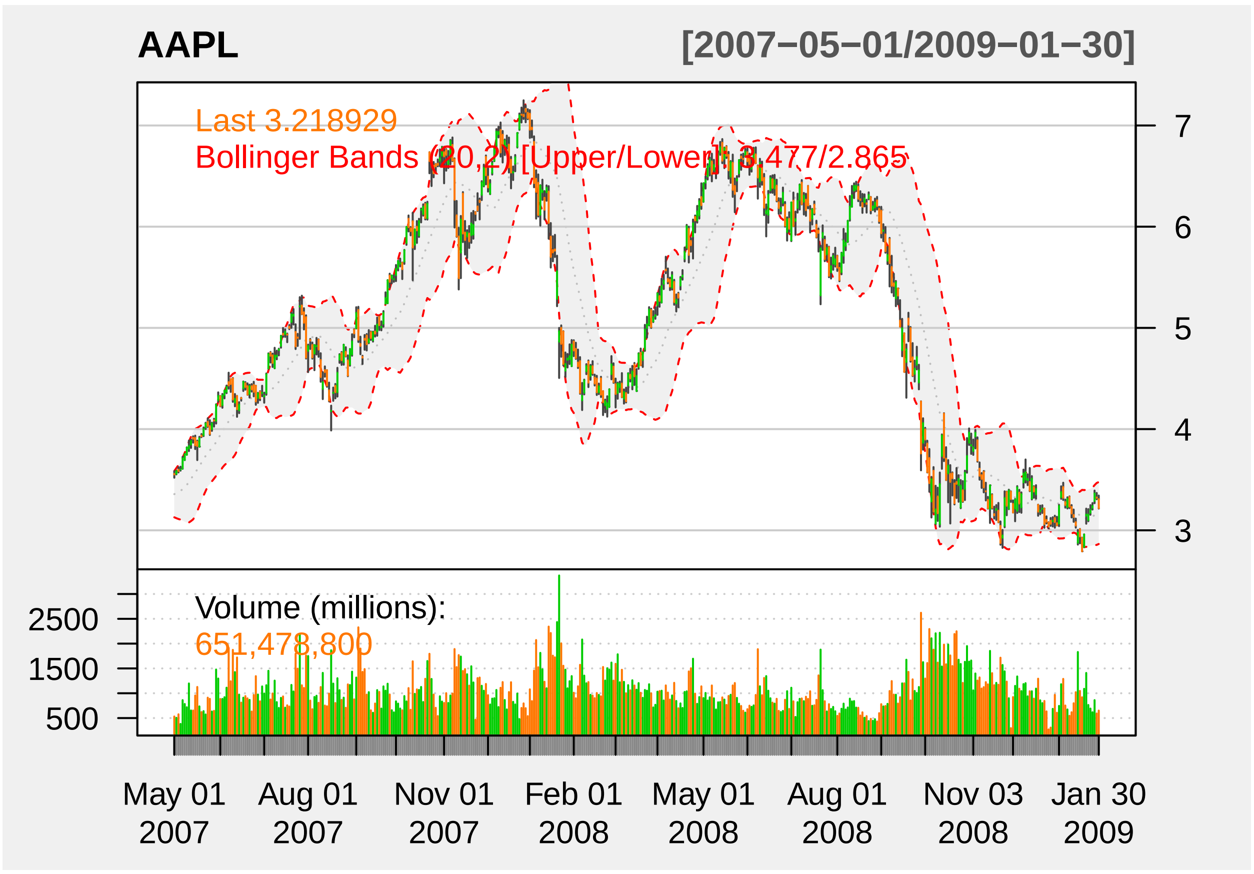
```
bb <-myBBands.mod(Cl(AAPL),n=20,sd=2)
tail(bb,n=5)
```

```
##                  dn     mavg       up      pctB
## 2023-07-24 186.2195 191.0375 195.8555 0.6777201
## 2023-07-25 187.3081 191.4550 195.6019 0.7610388
## 2023-07-26 187.7358 191.7770 195.8182 0.8369069
## 2023-07-27 188.0623 191.9755 195.8887 0.6590149
## 2023-07-28 188.1940 192.2875 196.3810 0.9326986
```

```
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white'))
```



```
addBBands(n=20,sd=2)
```



Remark 1.

Fundamental Analysis can be used to limit the set of stocks, and Techniqcal Analysis can be used to decide when to buy/sell.

2.4 Examples

Directional Movement Index

The +DI is the percentage of the true range that is up. The -DI is the percentage of the true range that is down. A buy signal is generated when the +DI crosses up over the -DI. A sell signal is generated when the -DI crosses up over the +DI. You should wait to enter a trade until the extreme point is reached. That is, you should wait to enter a long trade until the price reaches the high of the bar on which the +DI crossed over the -DI, and wait to enter a short trade until the price reaches the low of the bar on which the -DI crossed over the +DI.

The DI was developed by J. Welles Wilder.

$$\Delta High_t = high_t - high_{t-1}$$

$$\Delta low_t = low_t - low_{t-1}$$

$$+DM_t = \begin{cases} \Delta High_t & \text{if } \Delta High_t > \Delta Low_t, \text{ and } \Delta High_t > 0 \\ 0, & \text{o.w.} \end{cases}$$

$$-DM_t = \begin{cases} \Delta Low_t & \text{if } \Delta Low_t > \Delta High_t, \text{ and } \Delta Low_t > 0 \\ 0, & \text{o.w.} \end{cases}$$

After selecting the number of periods (Wilder used 14 days originally), +DI and -DI are:

- +DI = 100 times the smoothed moving average of (+DM) divided by average true range
- -DI = 100 times the smoothed moving average of (-DM) divided by average true range

where the true range (TR) and the average true range (ATR) are defined to be:

$$TR_t = \max(high_t, close_{t-1}) - \min(low_t, close_{t-1})$$

The ATR at the moment of time t is calculated using the following formula: (one form of an exponential moving average)

$$ATR_t = \frac{(n-1)ATR_{t-1} + TR_t}{n}$$

The first ATR value is calculated using the arithmetic mean formula:

$$ATR = \frac{1}{n}\sum_{i=1}^{n} TR_i$$

The DX is usually smoothed with a moving average (i.e. the ADX). The values range from 0 to 100, but rarely get above 60. To interpret the DX, consider a high number to be a strong trend, and a low number, a weak trend.

$$DX = |\frac{+DI - (-DI)}{(+DI) + (-DI)}|$$

The ADX is a Welles Wilder style moving average of the Directional Movement Index (DX). The values range from 0 to 100, but rarely get above 60. To interpret the ADX, consider a high number to be a strong trend, and a low number, a weak trend.

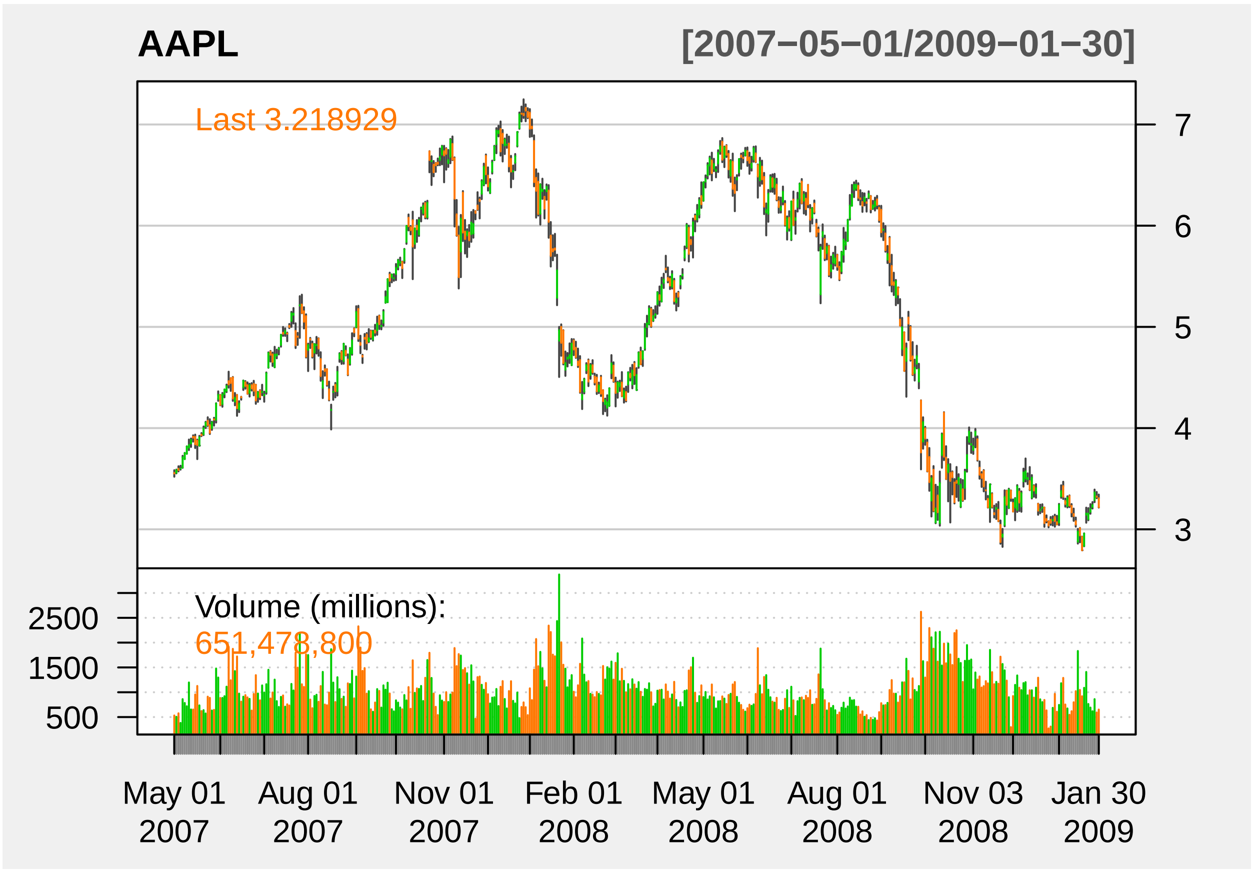
$$ADX_t = \frac{(n-1)ADX_{t-1} + DX}{n}$$

```r
# Directional Movement Index
adx <- ADX(AAPL[,c("AAPL.High", "AAPL.Low", "AAPL.Close")], n=14)
tail(adx, n=3)
```

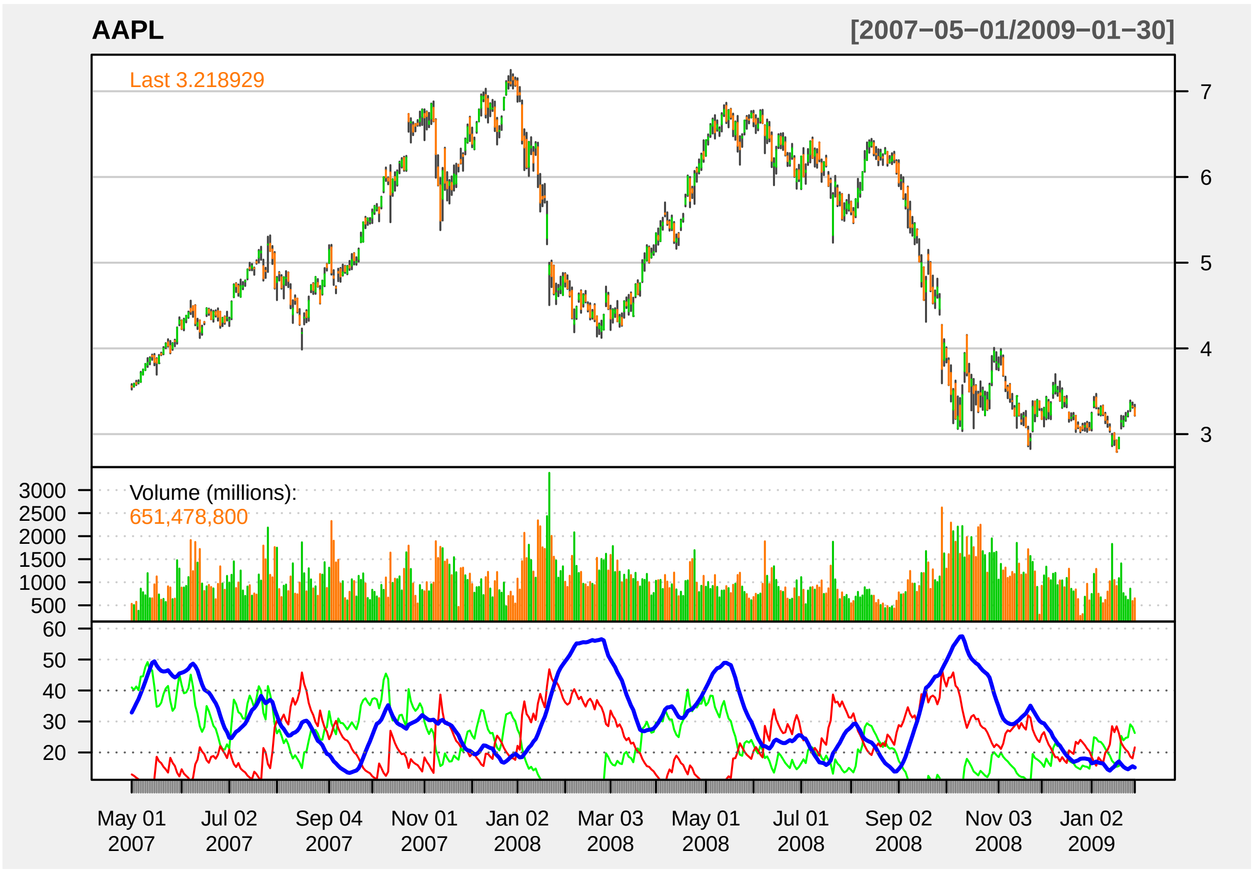
```
##                   DIp       DIn       DX      ADX
## 2023-07-26 28.42331 10.947999 44.38590 42.52201
## 2023-07-27 28.99188  9.733438 49.73088 43.03693
## 2023-07-28 26.65642  8.949353 49.73088 43.51507
```

A buy/sell signal is generated when the +/-DI crosses up over the -/+DI, when the DX/ADX signals a strong trend. A high/low DX signals a strong/weak trend. DX is usually smoothed with a moving average (i.e. the ADX).

```r
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white'))
```



```
addADX(n=14, maType = "EMA", wilder = TRUE)
```



```
# the red line is DIn
# the green line is DIp
# the blue line is ADX
```

Stochastic Oscillator/Stochastic Momentum Index

The Stochastic Oscillator measures where the close is in relation to the recent trading range. The values range from zero to 100. %D values over 75 indicate an overbought condition; values under 25 indicate an oversold condition. When the Fast %D crosses above the Slow %D, it is a buy signal; when it crosses below, it is a sell signal. The Raw %K is generally considered too erratic to use for crossover signals.

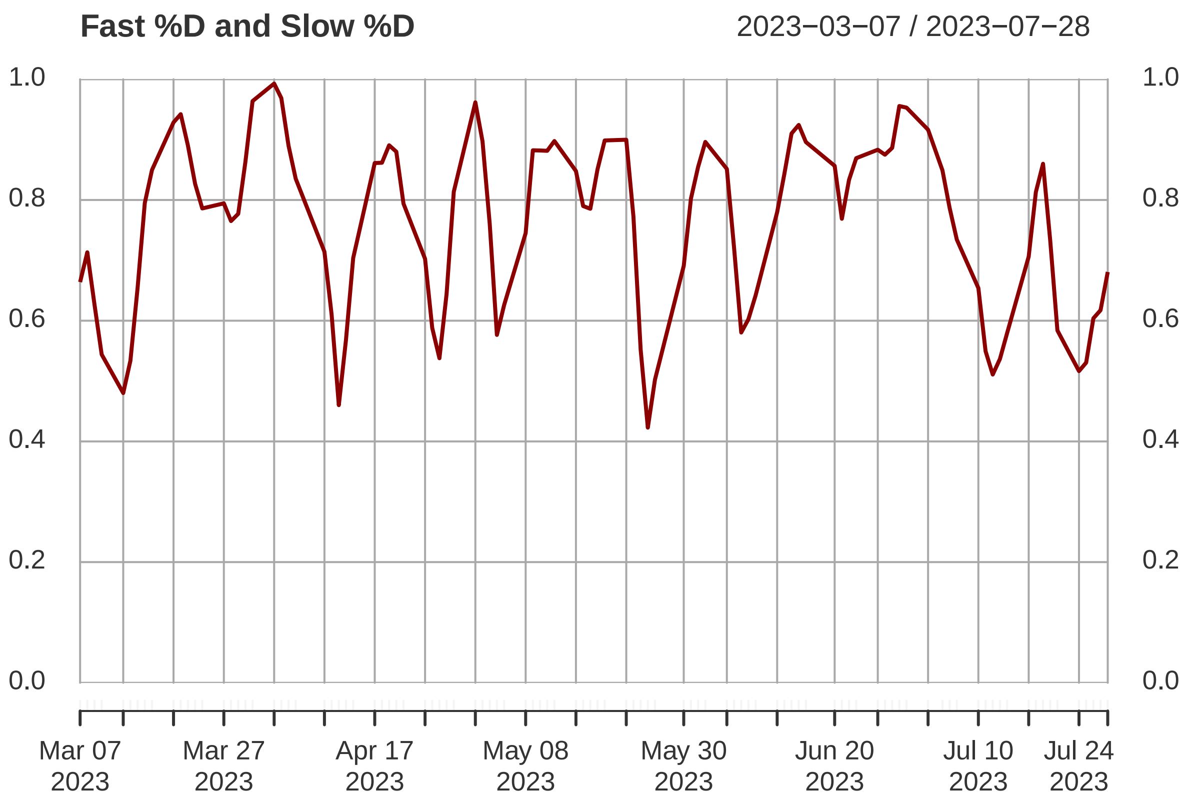
$$\%K = 100\frac{Close - LowestLow_{\text{last n periods}}}{HighestHigh_{\text{last n periods}} - LowestLow_{\text{last n periods}}} \tag{7}$$

$$\%D = MovingAverage(\%K) \tag{8}$$

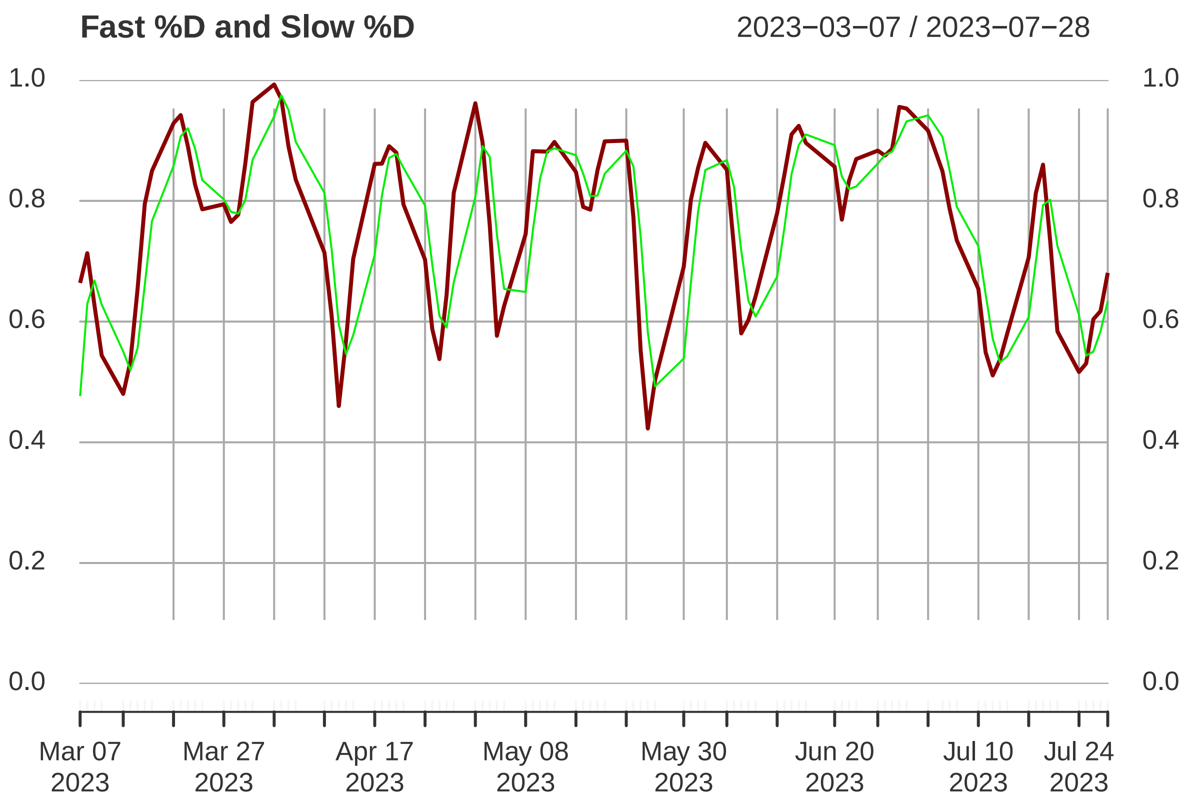
Terminology:

- Fast Stochastic: Refers to both %K and %D where %K is un-smoothed
- Slow Stochastic: Refers to both %K and %D where %K is smoothed
- Raw %K: Un-smoothed %K
- Fast %K: Un-smoothed %K
- Slow %K: Smoothed %K
- Fast %D: Moving average of an un-smoothed %K
- Slow %D: Moving average of a smoothed %K, in effect: a double smoothed %K
- %D Always refers to a smoothed %K (whether or not the %K itself is smoothed)

```r
# Stochastic
stochOsc <- stoch(AAPL[,c("AAPL.High", "AAPL.Low", "AAPL.Close")])
plot(tail(stochOsc[,"fastD"], 100), type="l",
    main="Fast %D and Slow %D", ylab="",
    ylim=range(stochOsc, na.rm=TRUE), col="darkred" )
```



```r
lines(tail(stochOsc[, "slowD"], 100), col="green2")
```



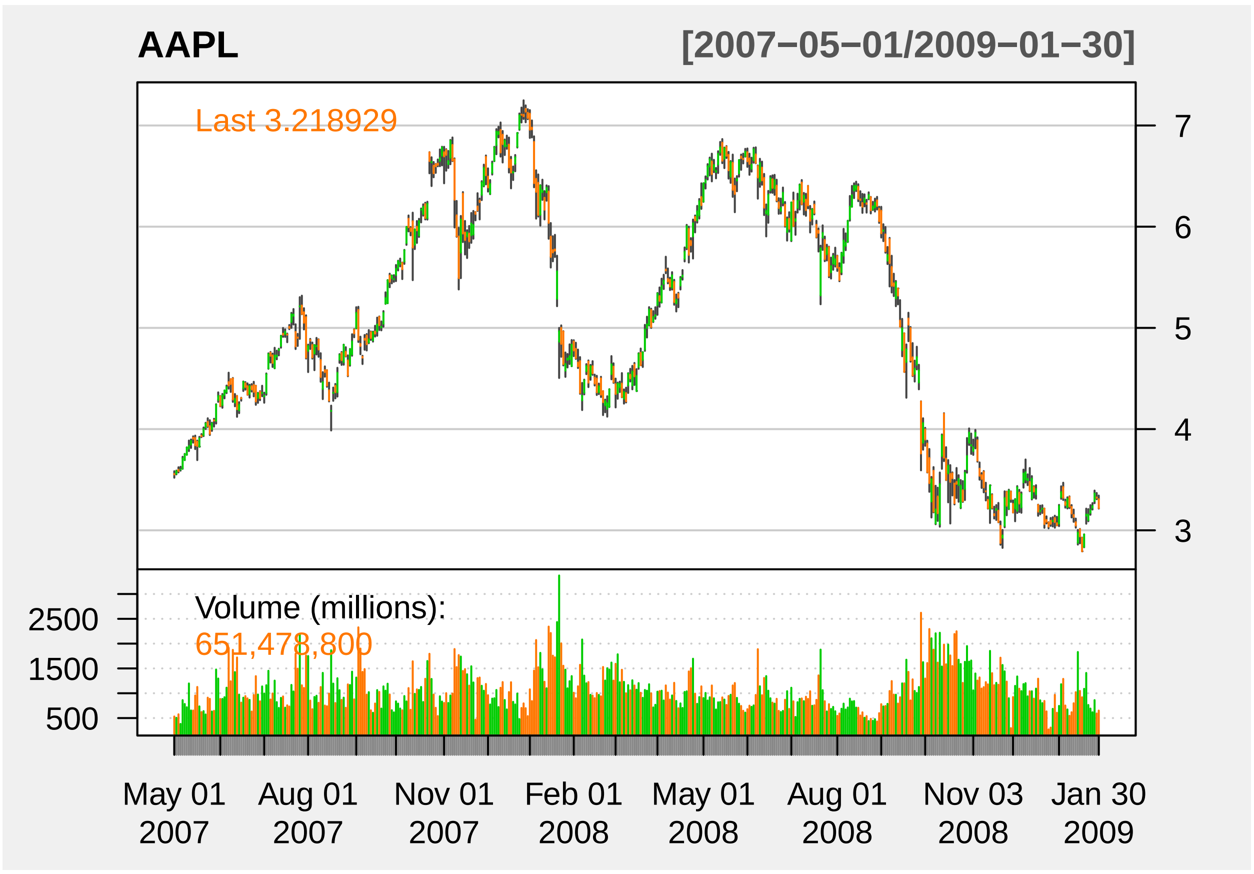
The Stochastic Momentum Index (SMI) is based on the Stochastic Oscillator. The difference is that the Stochastic Oscillator calculates where the close is relative to the high/low range, while the SMI calculates where the close is relative to the midpoint of the high/low range. The values of the SMI range from +100 to -100. When the close is greater than the midpoint, the SMI is above zero, when the close is less than than the midpoint, the SMI is below zero.

The SMI is interpreted the same way as the Stochastic Oscillator. Extreme high/low SMI values indicate overbought/oversold conditions. A buy signal is generated when the SMI rises above -50, or when it crosses above the signal line. A sell signal is generated when the SMI falls below +50, or when it crosses below the signal line. Also look for divergence with the price to signal the end of a trend or indicate a false trend.

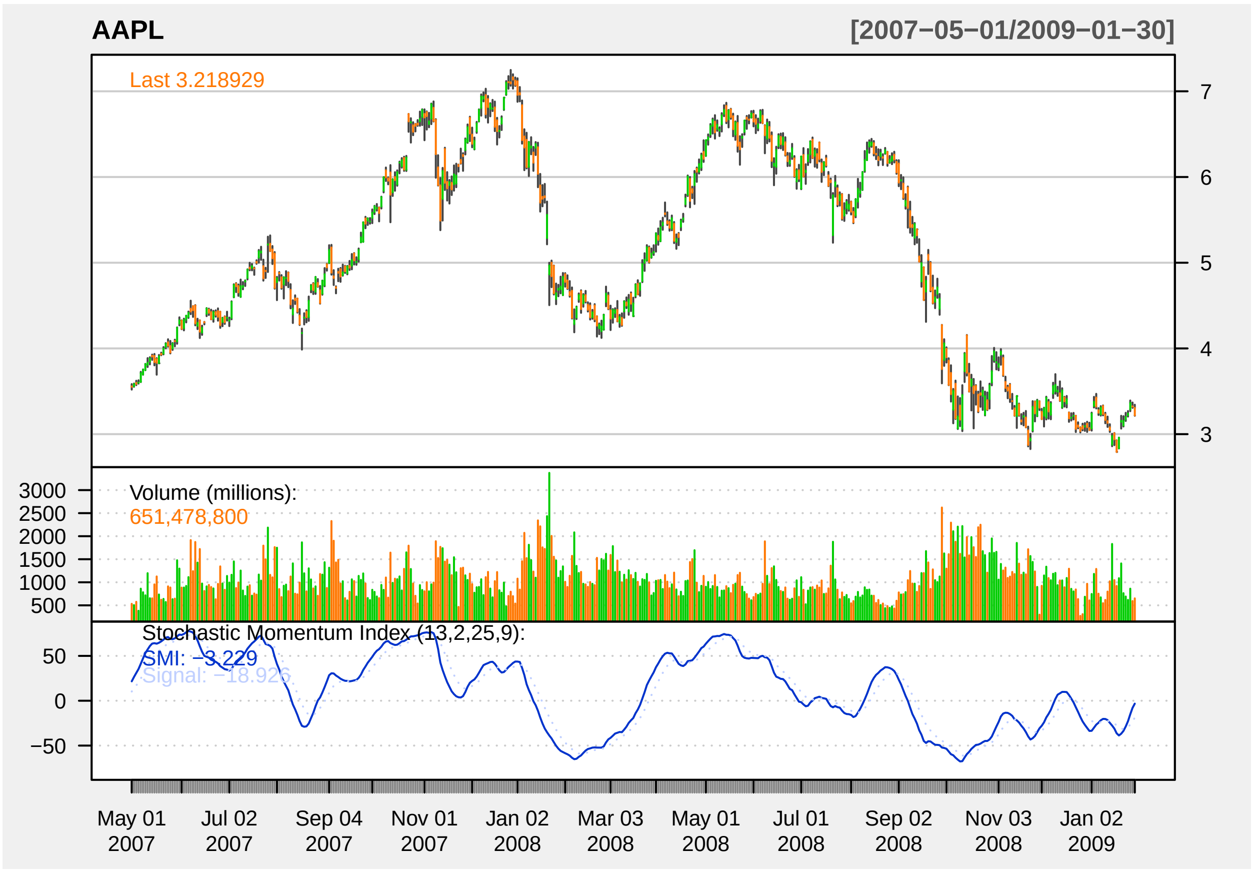
```
smi <- SMI(AAPL[,c("AAPL.High", "AAPL.Low", "AAPL.Close")], n = 13, nFast = 2, nSlow = 25,
tail(smi, n=3)
```

```
##                  SMI   signal
## 2023-07-26 36.81267 43.32849
## 2023-07-27 35.33637 41.73006
## 2023-07-28 35.60010 40.50407
```

```
chartSeries(AAPL,
            subset='2007-05::2009-01',
            theme=chartTheme('white'))
```



```
addSMI(n=13,slow=25,fast=2,signal=9,ma.type="EMA")
```



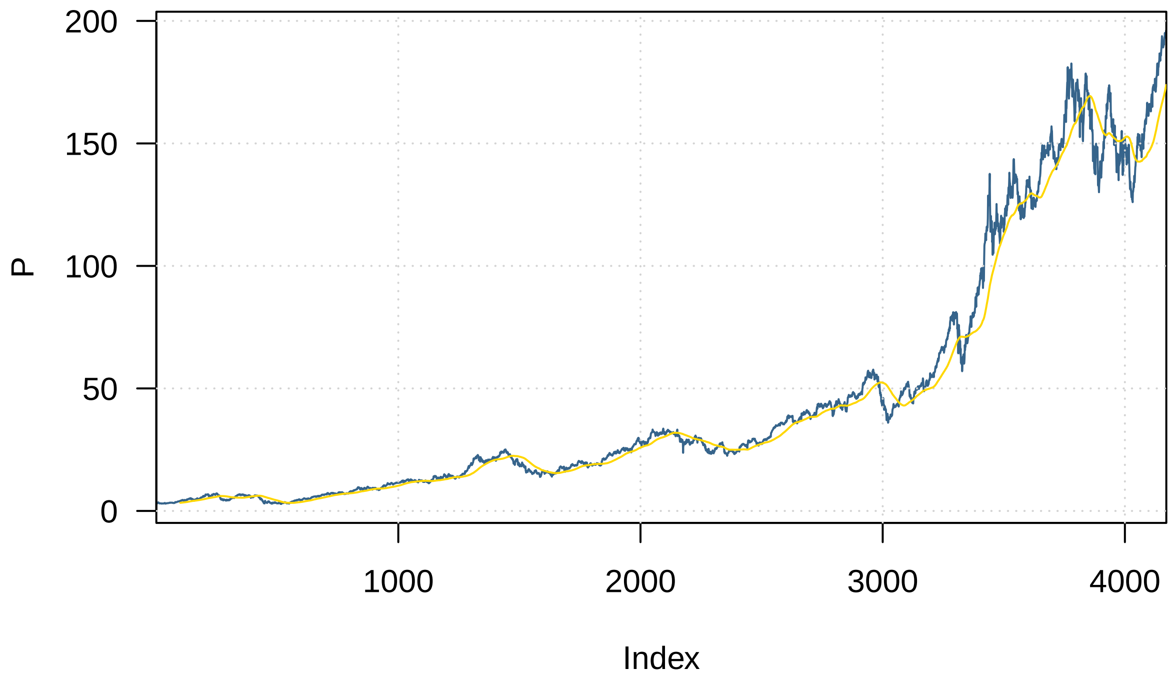
If a High-Low-Close series is provided, the indicator is calculated using the high/low values. If a vector is provided, the calculation only uses that series. This allows stochastics to be calculated for: (1) series that have no HLC definition (e.g. foreign exchange), and (2) stochastic indicators (e.g. stochastic RSI).

The stochastic oscillator and the stochastic momentum index are interpreted similarly. Readings below 20 (above 80) are considered oversold (overbought). However, readings below 20 (above 80) are not necessarily bearish (bullish). Lane believed some of the best sell (buy) signals occurred when the oscillator moved from overbought (oversold) back below 80 (above 20).

For the stochastic oscillator, buy (sell) signals can also be given when %K crosses above (below) %D. Crossover signals are quite frequent however, which may result in whipsaws.

Weighted Moving Average

```r
### Using QUANTMOD
P = as.numeric(AAPL[,1]) # Open price
sma1 = SMA(P,100)
plot(P,type="l",xaxs="i",las=1,col="steelblue4")
grid()
lines(sma1,col="gold")
```



The Weighted Moving Average calculates a weight for each value in the series. The more recent values are assigned greater weights. The Weighted Moving Average is similar to a Simple Moving average in that it is not cumulative, that is, it only includes values in the time period (unlike an Exponential Moving Average). The Weighted Moving Average is similar to an Exponential Moving Average in that more recent data has a greater contribution to the average.

$$WMA_t(P,n) = \frac{nP_t + (n-1)P_{t-1} + (n-2)P_{t-2} + ... + P_{t-n+1}}{n(n+1)/2}$$

`WMA(x, n = 10, wts = 1:n, ...)`

WMA is similar to an EMA, but with linear weighting if the length of `wts` is equal to `n`. If the length of `wts` is equal to the length of `x`, the WMA will use the values of `wts` as weights.

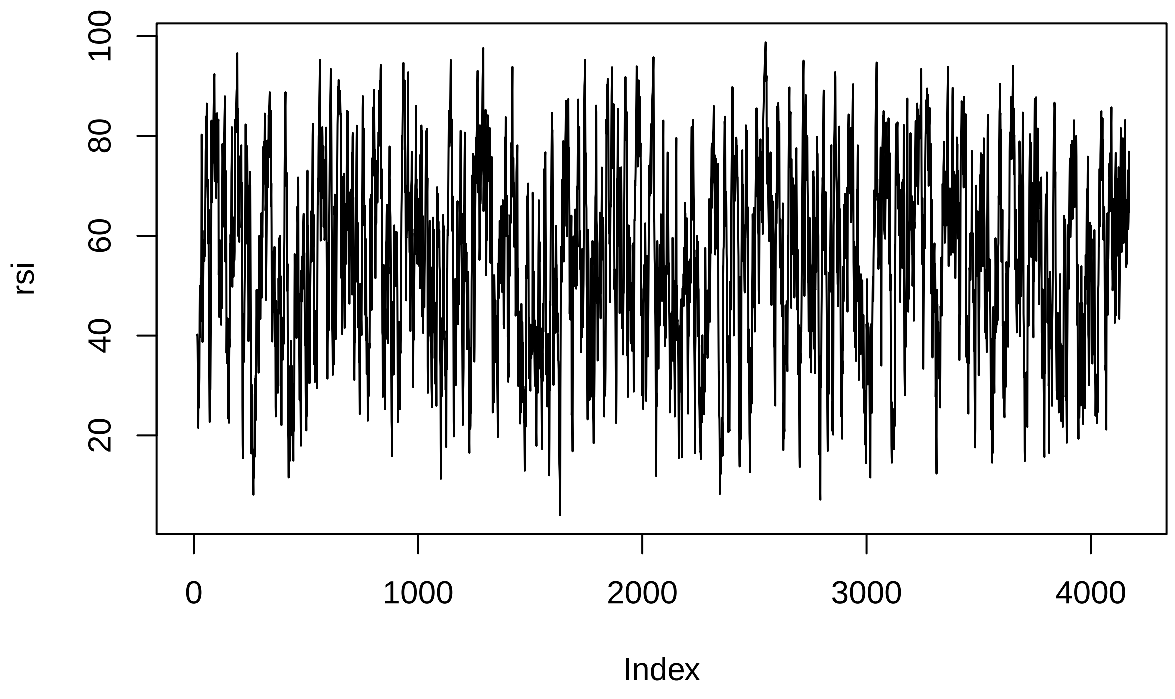
`wts`

Vector of weights. Length of wts vector must equal the length of x, or n (the default).

```
wma <- WMA(Cl(AAPL), n=20)
tail(wma, n=3)
```

```
##                  [,1]
## 2023-07-26 192.3710
## 2023-07-27 192.5084
## 2023-07-28 192.8755
```

```
rsi = RSI(P, n=14, maType="WMA")
plot(rsi,type="l")
```



```
rm(list = ls())
```

3 Factor Model

3.1 Single Factor Model

Return of asset i is proportional to a factor f

$$E(R_i) = \alpha_i + \beta_i E(f)$$

Hence, we have

$$R_{i,t} = \alpha_i + \beta_i f_t + \epsilon_{i,t}, \forall t = 1, ..., T$$

3.1.1 Sharpe Single Factor Model (CAPM)

β is estimated by time-series regression of excess return on market excess return

$$R_{i,t} - r_f = \alpha_i^* + \beta_i [R_{M,t} - r_f] + \epsilon_{i,t}$$

where $\alpha_i^* = \alpha_i - r_f(1 - \beta_i)$

Alternatively, we have

$$\beta_i = \frac{cov(R_{i,t} - r_f, R_{M,t} - r_f)}{Var(R_{M,t} - r_f)} = \frac{cov(R_{i,t}, R_{M,t})}{Var(R_{M,t})}$$

3.1.2 Security Market Line

$$E[R_{i,t}] = r_f + \beta_i (E[R_{M,t}] - r_f)$$
$$\beta_i = \frac{cov(R_{i,t}, R_{M,t})}{var(R_{M,t})}$$

or $\mu_i = r_f + \beta_i(\mu_M - r_f)$

Since $E[R_{M,t}] - r_f = (\mu_M - r_f) > 0$, high (low) $\beta_i \implies$ high (low) μ_i

CAPM SML relationship implies that $\alpha_i^* = 0$ for every asset i.

3.1.3 Regression Test of the CAPM

Use linear regression to estimate the excess returns SI model

$$R_{i,t} - r_f = \alpha_i^* + \beta_i (R_{M,t} - r_f) + \epsilon_{i,t}$$

for i = 1,…, N assets

```r
# 1. requires data in file berndt.csv

# read prices from csv file
berndt.df = read.csv("https://faculty.washington.edu/ezivot/econ424/berndt.csv", stringsAs
colnames(berndt.df)
```

```
##  [1] "Date"   "CITCRP" "CONED"  "CONTIL" "DATGEN" "DEC"    "DELTA"  "GENMIL"
##  [9] "GERBER" "IBM"    "MARKET" "MOBIL"  "PANAM"  "PSNH"   "TANDY"  "TEXACO"
## [17] "WEYER"  "RKFREE"
```

```r
suppressMessages(library(zoo))

# create zooreg object - regularly spaced zoo object
# zooreg object is basically zoo object with frequency attribute
berndt.z = zooreg(berndt.df[,-1], start=c(1978, 1), end=c(1987,12),
                  frequency=12)  # freq=12 >> monthly data

index(berndt.z) = as.yearmon(index(berndt.z)) # yearmon is a class for representing month
```

```r
start(berndt.z)
```

```
## [1] "Jan 1978"
```

```r
end(berndt.z)
```

```
## [1] "Dec 1987"
```

```r
nrow(berndt.z)
```

```
## [1] 120
```

```r
# create excess returns by subtracting off risk free rate
# note: coredata() function extracts data from zoo object
returns.mat = as.matrix(coredata(berndt.z))
excessReturns.mat = returns.mat - returns.mat[,"RKFREE"]
excessReturns.df = as.data.frame(excessReturns.mat)
```

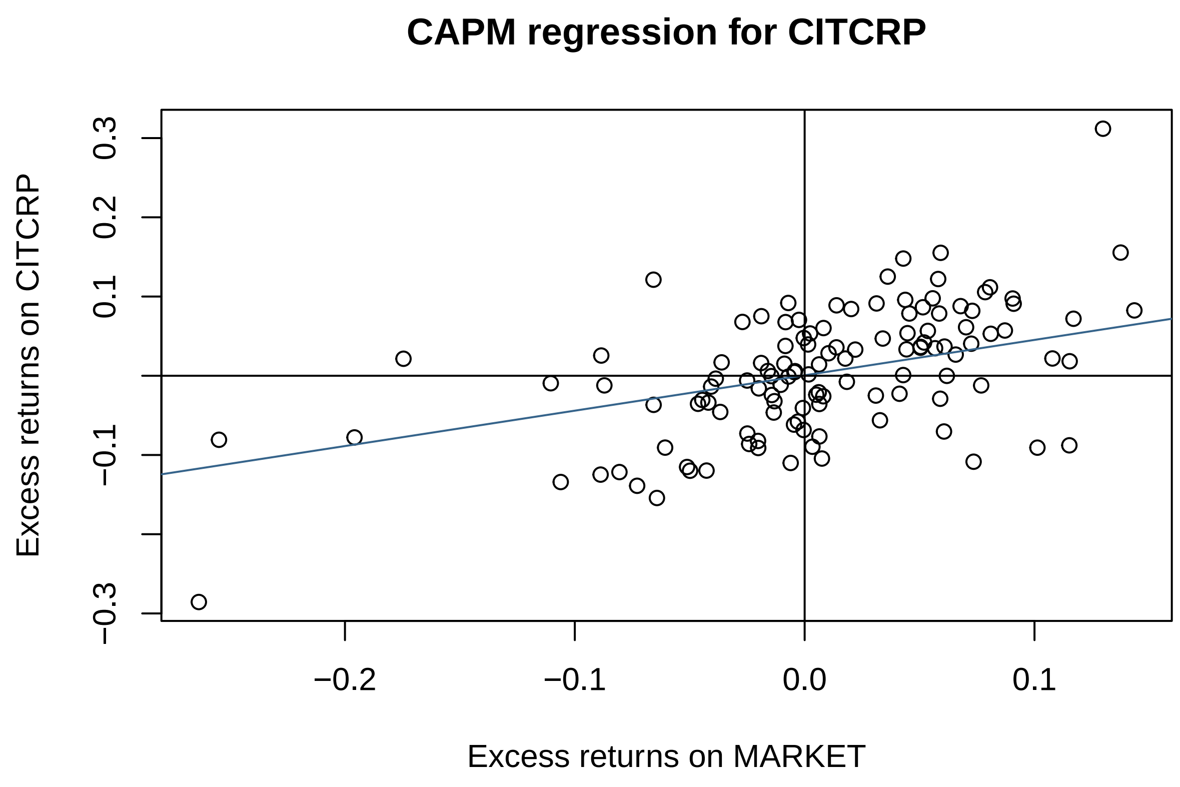
```r
# CAPM regression for CITCRP (citicorp) using 1st 5 years of data
capm.fit = lm(CITCRP~MARKET,data=excessReturns.df,subset=1:60)

suppressMessages(library(stargazer))
stargazer(capm.fit , title = "CAPM Regression Table for CITCRP", header = FALSE)
```

Table 1: CAPM Regression Table for CITCRP

	Dependent variable:
	CITCRP
MARKET	0.447^{***}
	(0.119)
Constant	0.001
	(0.009)
Observations	60
R^2	0.195
Adjusted R^2	0.181
Residual Std. Error	0.070 (df = 58)
F Statistic	14.036^{***} (df = 1; 58)
Note:	$^{*}p<0.1$; $^{**}p<0.05$; $^{***}p<0.01$

```r
# plot data and regression line
plot(excessReturns.df$MARKET,excessReturns.df$CITCRP,
     main="CAPM regression for CITCRP",
     ylab="Excess returns on CITCRP",
     xlab="Excess returns on MARKET")
abline(capm.fit, col="steelblue4")                        # plot regression line
abline(h=0,v=0)                            # plot horizontal and vertical lines at 0
```

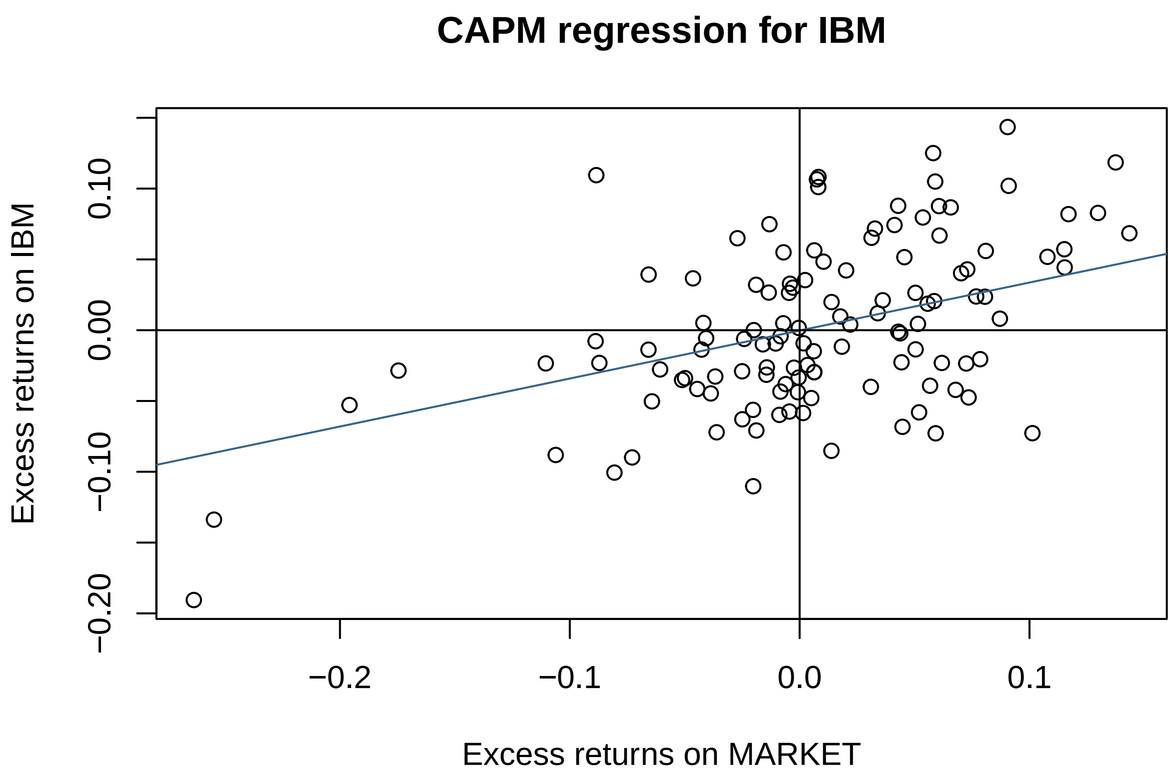


```
# CAPM regression for IBM using 1st 5 years of data
capm.fit = lm(IBM~MARKET,data=excessReturns.df,subset=1:60)
stargazer(capm.fit , title = "CAPM Regression Table for IBM", header = FALSE)
```

Table 2: CAPM Regression Table for IBM

	Dependent variable:
	IBM
MARKET	0.339^{***}
	(0.089)
Constant	−0.0002
	(0.007)
Observations	60
R^2	0.201
Adjusted R^2	0.187
Residual Std. Error	0.052 (df = 58)
F Statistic	14.575^{***} (df = 1; 58)
Note:	*p<0.1; **p<0.05; ***p<0.01

```
# plot data and regression line
plot(excessReturns.df$MARKET,excessReturns.df$IBM,
     main="CAPM regression for IBM",
     ylab="Excess returns on IBM",
     xlab="Excess returns on MARKET")
abline(capm.fit, col="steelblue4")                    # plot regression line
abline(h=0,v=0)                       # plot horizontal and vertical lines at 0
```



Test the hypothesis

$$H_0 : \alpha_i^* = 0(\text{CAPM holds})$$
$$H_1 : \alpha_i^* \neq 0(\text{CAPM does not hold})$$

for all assets i = 1,…,N assets

```
# estimate CAPM and test alpha=0 for all assets using 1st 5 years of data
# trick use S-PLUS function apply to do this all at once

capm.tstats = function(r,market) {
  capm.fit = lm(r~market)                         # fit capm regression
  capm.summary = summary(capm.fit)        # extract summary info
  t.stat = coef(capm.summary)[1,3]        # t-stat on intercept
  t.stat
}
```

```
# test function using CITCRP data
tmp = capm.tstats(excessReturns.mat[1:60,1],
                  excessReturns.mat[1:60,"MARKET"])
tmp # same as what we expected
```

```
## [1] 0.06501986
```

```r
# using apply fcn

colnames(excessReturns.mat[,-c(10,17)])
```

```
##  [1] "CITCRP" "CONED"  "CONTIL" "DATGEN" "DEC"    "DELTA"  "GENMIL" "GERBER"
##  [9] "IBM"    "MOBIL"  "PANAM"  "PSNH"   "TANDY"  "TEXACO" "WEYER"
```

```r
tstats = apply(excessReturns.mat[1:60,-c(10,17)],2,
               FUN=capm.tstats,
               market=excessReturns.mat[1:60,"MARKET"])
tstats
```

```
##      CITCRP       CONED      CONTIL      DATGEN         DEC       DELTA
##  0.06501986  1.21411728 -0.67030349 -1.04296670  0.03092165  0.62136933
##      GENMIL      GERBER         IBM       MOBIL       PANAM        PSNH
##  0.54110492 -0.06884828 -0.03630053  0.08595583 -0.89396008 -0.27454911
##       TANDY      TEXACO       WEYER
##  1.99703362 -0.40444639 -0.51999713
```

```r
# test H0: alpha = 0 using 5% test
abs(tstats) > 2
```

```
## CITCRP  CONED CONTIL DATGEN    DEC  DELTA GENMIL GERBER    IBM  MOBIL  PANAM
##  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE
##   PSNH  TANDY TEXACO  WEYER
##  FALSE  FALSE  FALSE  FALSE
```

```r
any(abs(tstats) > 2)
```

```
## [1] FALSE
```

Question 1. *What if you reject $H_0 : \alpha_i^* = 0$?*

Suppose $\alpha_i^* > 0$ (positive ”alpha”). Then

$$\alpha_i^* = (E[R_{i,t}] - r_f) - \beta_i(E[R_{M,t}] - r_f) > 0$$

so that expected excess return on asset i is greater than what CAPM predicts.

1. Asset is underpriced relative to CAPM (expected return too high→current price too low)

2. If CAPM is true, then expected return should fall soon which implies that current price should rise soon.

$\alpha_i^* > 0 \implies$ *buy asset today;* $\alpha_i^* < 0 \implies$ *sell asset today.*

3.1.4 Prediction Test of CAPM

Security Market Line (SML) says

$$\mu_i - r_f = \beta_i(\mu_{M,t} - r_f)$$
$$\mu_{M,t} - r_f > 0$$

Implication:

- High (low) β stocks should have high (low) average returns μ_i

```
#
# plot average return against beta
#

# compute average (excess) returns over 1st 5 years
mu.hat = colMeans(excessReturns.mat[1:60,-c(10,17)]) # exclude market and riskfree
mu.hat
```

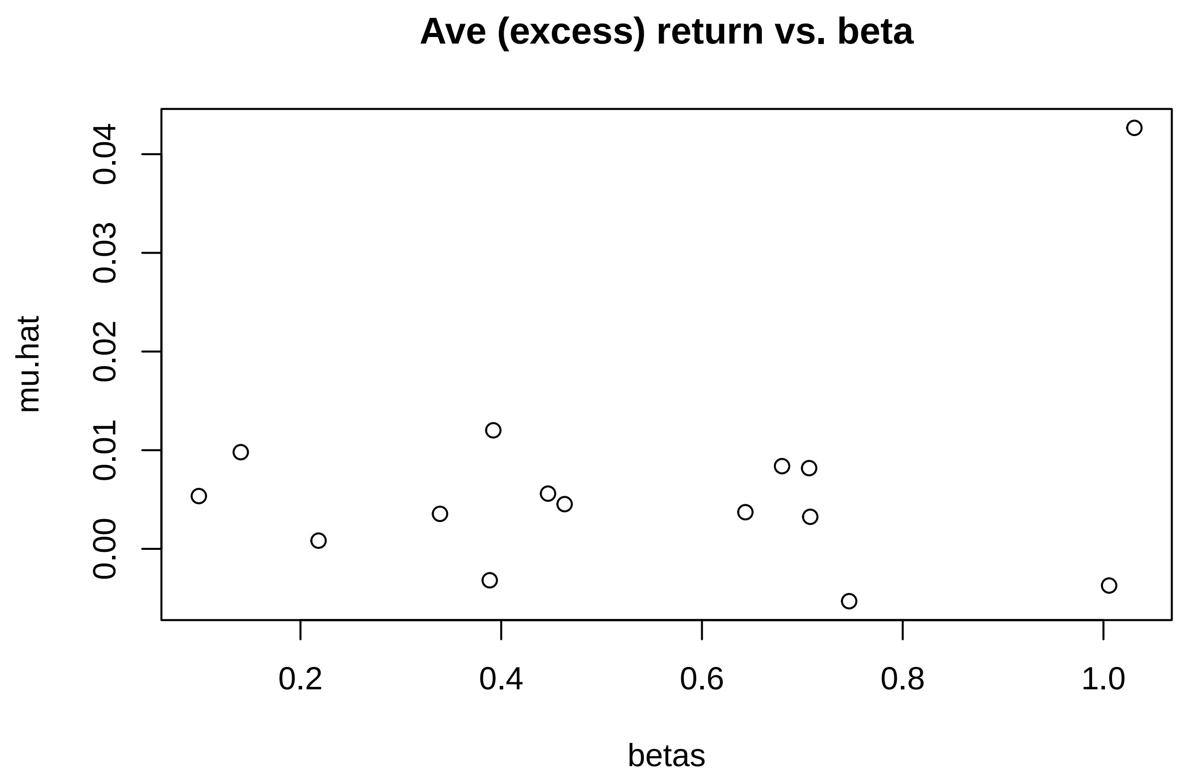
```
##       CITCRP        CONED       CONTIL       DATGEN          DEC        DELTA
##  0.005598167  0.009798167 -0.003185167 -0.003718500  0.008181500  0.012014833
##       GENMIL       GERBER          IBM        MOBIL        PANAM         PSNH
##  0.005348167  0.004531500  0.003548167  0.008381500 -0.005301833  0.000831500
##        TANDY       TEXACO        WEYER
##  0.042664833  0.003714833  0.003248167
```

```
# compute beta over 1st 5 years
capm.betas = function(r,market) {
  capm.fit = lm(r~market)                    # fit capm regression
  capm.beta = coef(capm.fit)[2]              # extract coefficients
  capm.beta
}
```

```
betas = apply(excessReturns.mat[1:60,-c(10,17)],2,
              FUN=capm.betas,
              market=excessReturns.mat[1:60,"MARKET"])
betas
```

```
##     CITCRP      CONED     CONTIL     DATGEN        DEC      DELTA     GENMIL
## 0.44663080 0.14049874 0.38854870 1.00561709 0.70677295 0.39209800 0.09873775
##     GERBER        IBM      MOBIL      PANAM       PSNH      TANDY     TEXACO
## 0.46316113 0.33901221 0.67977729 0.74664275 0.21801661 1.03081022 0.64326107
##      WEYER
## 0.70788726
```

```
# plot average returns against betas
plot(betas,mu.hat,main="Ave (excess) return vs. beta")
```



Estimate SML using regression

$$\hat{\mu}_i - r_f = \gamma_0 + \gamma_i \hat{\beta}_i + error_i$$

```
# estimate regression of ave return on beta
sml.fit = lm(mu.hat~betas)
sml.fit
```

```
##
## Call:
## lm(formula = mu.hat ~ betas)
##
## Coefficients:
## (Intercept)        betas
##   0.0004277    0.0111446
```

```
summary(sml.fit)
```

```
##
## Call:
## lm(formula = mu.hat ~ betas)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0153534 -0.0044752 -0.0006577  0.0020990  0.0307492
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0004277  0.0062655   0.068    0.947
## betas       0.0111446  0.0104246   1.069    0.304
```

```
##
## Residual standard error: 0.01115 on 13 degrees of freedom
## Multiple R-squared:  0.08081,    Adjusted R-squared:  0.0101
## F-statistic: 1.143 on 1 and 13 DF,  p-value: 0.3045
```

and test hypotheses

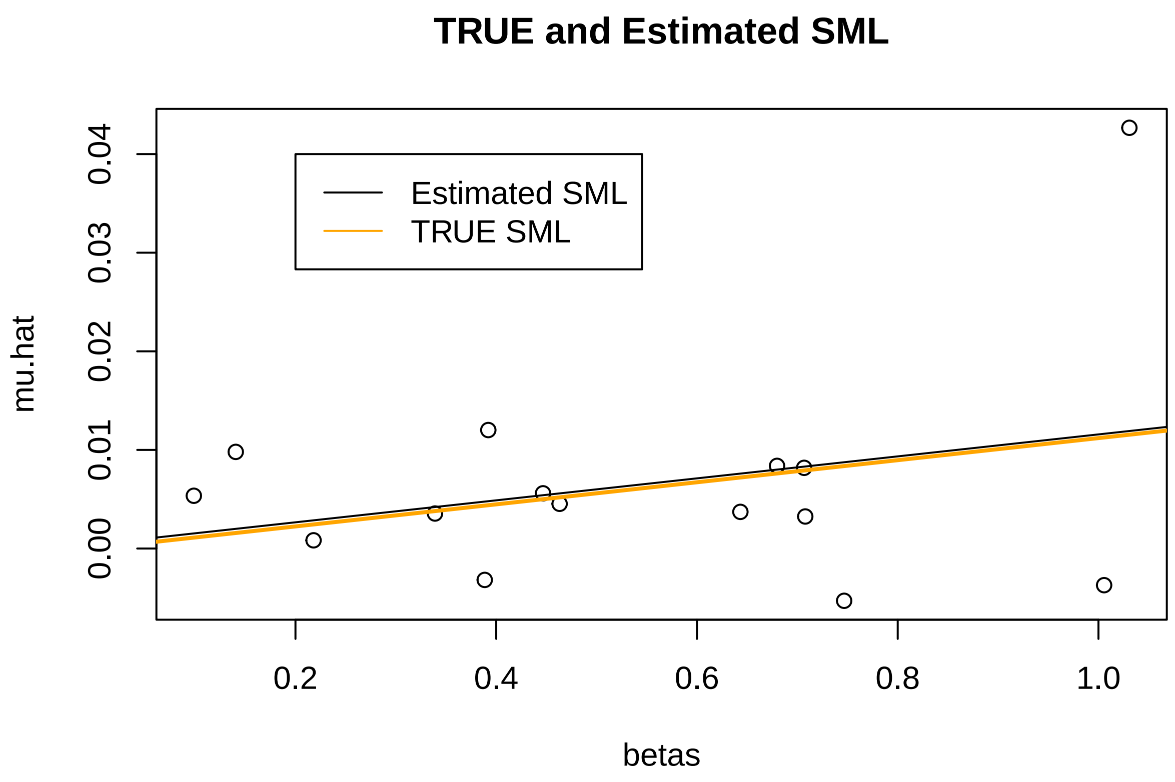
$$H_0 : \gamma_0 = 0 \text{ and } \gamma_1 = \mu_{M,t} - r_f$$
$$H_1 : \gamma_0 \neq 0 \text{ and } \gamma_1 \neq \mu_{M,t} - r_f$$

```
# intercept should be zero and slope should be excess return on market

mean(excessReturns.mat[1:60,"MARKET"]) # gamma_1
```

```
## [1] 0.01119817
```

```
plot(betas,mu.hat,main="TRUE and Estimated SML")
abline(sml.fit) # estimated SML
abline(a=0,b=mean(excessReturns.mat[1:60,"MARKET"]),lty=1, col="orange", lwd=2) # true SM
legend(0.2, 0.04, legend=c("Estimated SML","TRUE SML"),
       lty=c(1,1), col=c("black","orange"))
```



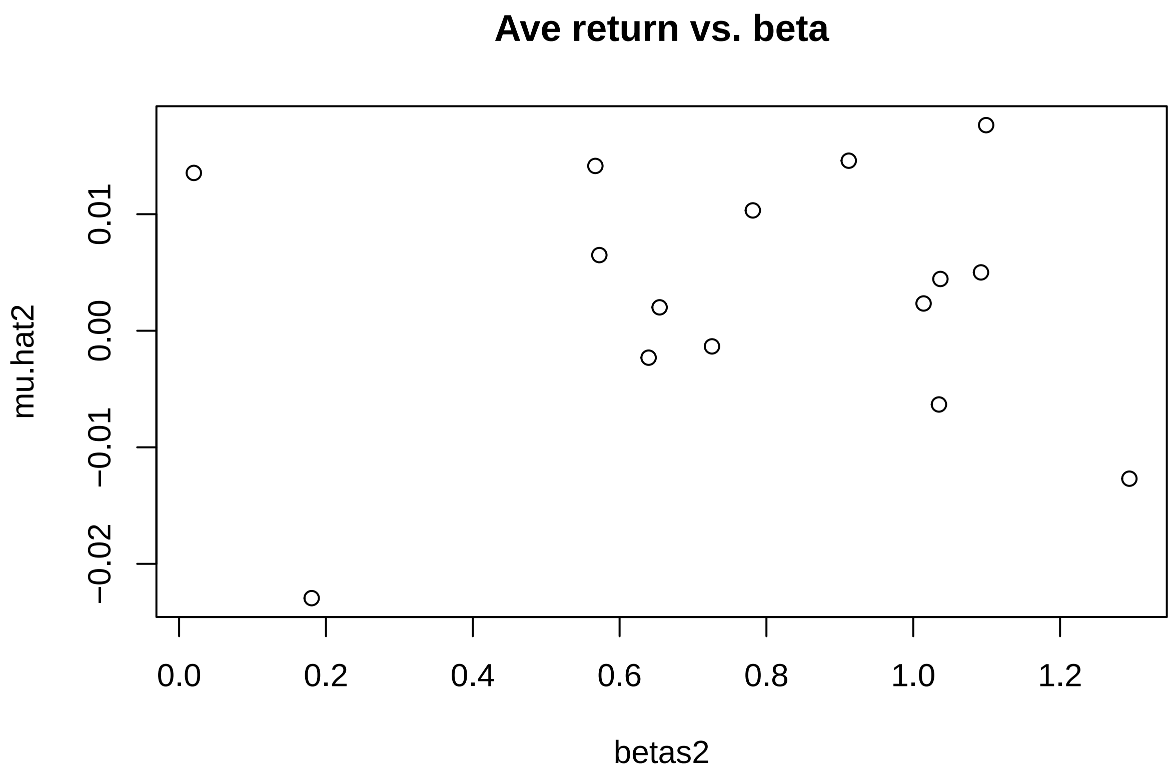
```
# compute average returns over 2nd 5 years
mu.hat2 = colMeans(excessReturns.mat[61:120,-c(10,17)])
mu.hat2
```

```
##      CITCRP        CONED       CONTIL       DATGEN          DEC        DELTA
##  0.004441333  0.013541333 -0.012692000  0.005008000  0.017641333 -0.002308667
##      GENMIL       GERBER          IBM        MOBIL        PANAM         PSNH
##  0.014141333  0.014591333  0.002008000  0.010324667 -0.001342000 -0.022942000
##        TANDY       TEXACO        WEYER
## -0.006325333  0.006491333  0.002341333
```

```
betas2 = apply(excessReturns.mat[61:120,-c(10,17)],2,
               FUN=capm.betas,
               market=excessReturns.mat[61:120,"MARKET"])
betas2
```

```
##     CITCRP      CONED     CONTIL     DATGEN        DEC      DELTA     GENMIL
## 1.03701118 0.02001884 1.29443411 1.09229391 1.09928950 0.63952500 0.56697377
##     GERBER        IBM      MOBIL      PANAM       PSNH      TANDY     TEXACO
## 0.91216187 0.65448596 0.78146833 0.72584473 0.18050799 1.03501175 0.57241684
##      WEYER
## 1.01411692
```

```
# plot average returns against betas
plot(betas2,mu.hat2,main="Ave return vs. beta")
```



```
# estimate regression of ave return on beta
sml.fit2 = lm(mu.hat2~betas2)
sml.fit2
```

```
##
## Call:
## lm(formula = mu.hat2 ~ betas2)
##
## Coefficients:
## (Intercept)       betas2
##    0.002028     0.001247
```

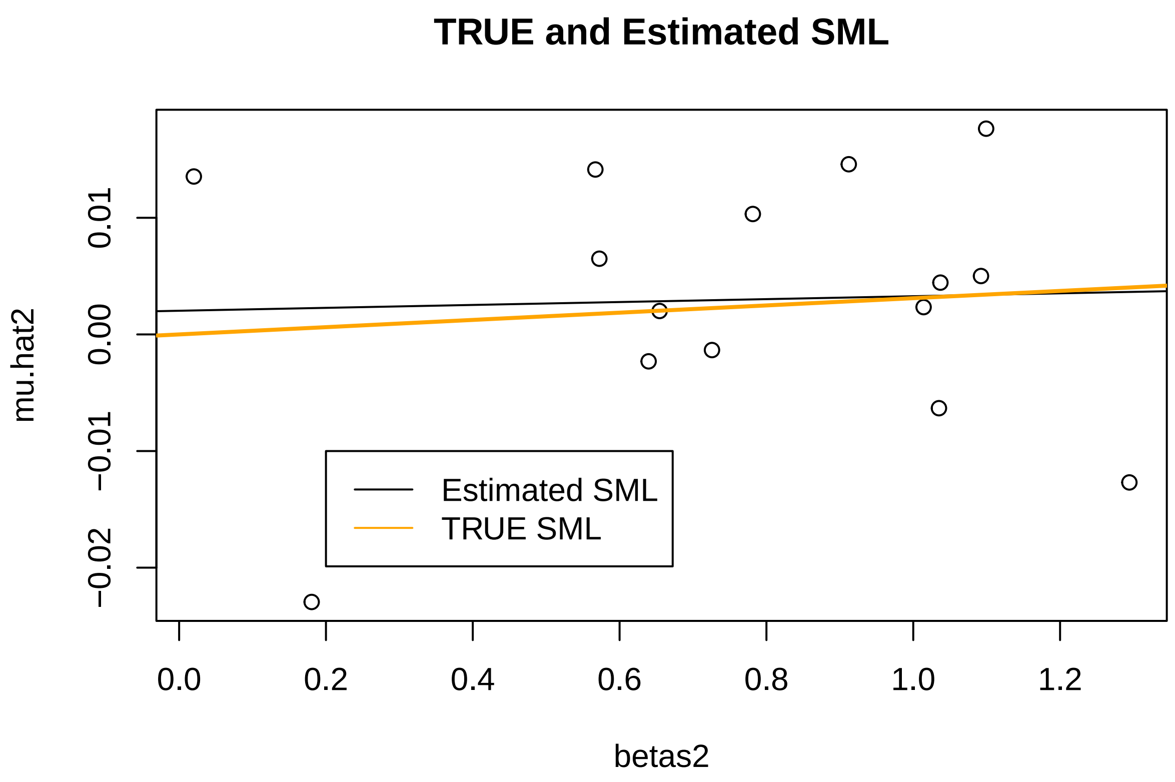
```
summary(sml.fit2)
```

```
##
## Call:
## lm(formula = mu.hat2 ~ betas2)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.025195 -0.004705  0.001120  0.009364  0.014242
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.002028   0.007372   0.275    0.788
## betas2      0.001247   0.008712   0.143    0.888
##
## Residual standard error: 0.01146 on 13 degrees of freedom
## Multiple R-squared:  0.001573,   Adjusted R-squared:  -0.07523
## F-statistic: 0.02048 on 1 and 13 DF,  p-value: 0.8884
```

```r
# intercept should be zero and slope should be excess return on market
mean(excessReturns.mat[61:120,"MARKET"])
```

```
## [1] 0.003108
```

```r
plot(betas2,mu.hat2,main="TRUE and Estimated SML")
abline(sml.fit2)
abline(a=0,b=mean(excessReturns.mat[61:120,"MARKET"]),lwd=2, col="orange")
legend(0.2, -0.01, legend=c("Estimated SML","TRUE SML"),
       lty=c(1,1), col=c("black","orange"))
```



3.1.5 Prediction Test II

A true prediction test would use β's estimated during one period to predict average returns in another period. For example, one can:

1. Split sample into 2 non-overlapping 5 year sub-samples
2. Estimate β_i over first 5 years
3. Estimate $\mu_i - r_f$ over second 5 years
4. Perform prediction test as described above

```
#
# prediction test II of CAPM
# estimate beta using 1st 5 years of data and
# compute average returns using 2nd 5 years of data
#


# estimate regression of 2nd period ave return on
# 1st period beta
sml.fit12 = lm(mu.hat2~betas)
sml.fit12
```

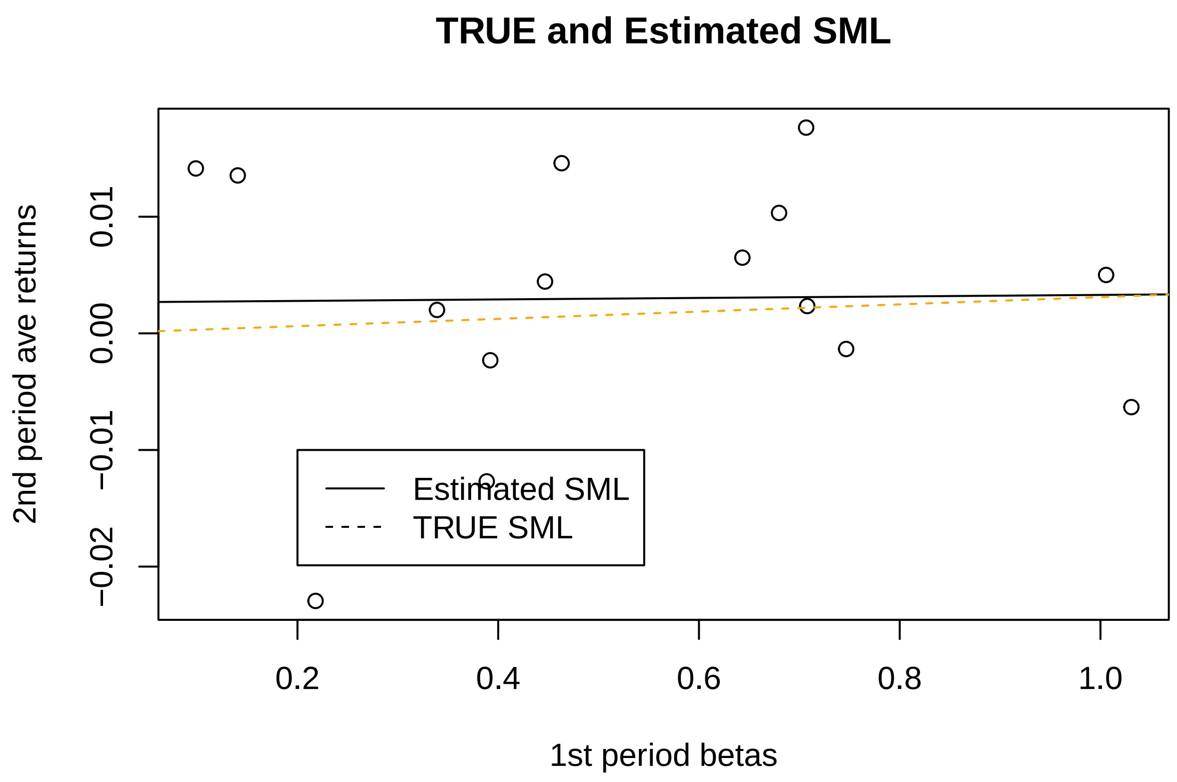
```
##
## Call:
## lm(formula = mu.hat2 ~ betas)
##
## Coefficients:
## (Intercept)        betas
##   0.0026507    0.0006443
```

```
summary(sml.fit12)
```

```
##
## Call:
## lm(formula = mu.hat2 ~ betas)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.025733 -0.004843  0.001503  0.009018  0.014535
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0026507  0.0064454   0.411    0.688
## betas       0.0006443  0.0107240   0.060    0.953
##
## Residual standard error: 0.01147 on 13 degrees of freedom
## Multiple R-squared:  0.0002776,  Adjusted R-squared:  -0.07662
## F-statistic: 0.00361 on 1 and 13 DF,  p-value: 0.953
```

```
plot(betas,mu.hat2,main="TRUE and Estimated SML",
     xlab="1st period betas",ylab="2nd period ave returns")
abline(sml.fit12) # estimated SML
abline(a=0,b=mean(excessReturns.mat[61:120,"MARKET"]),lty=2,col="orange") # true SML
```

```
legend(0.2, -0.01, legend=c("Estimated SML","TRUE SML"),
       lty=c(1,2))
```



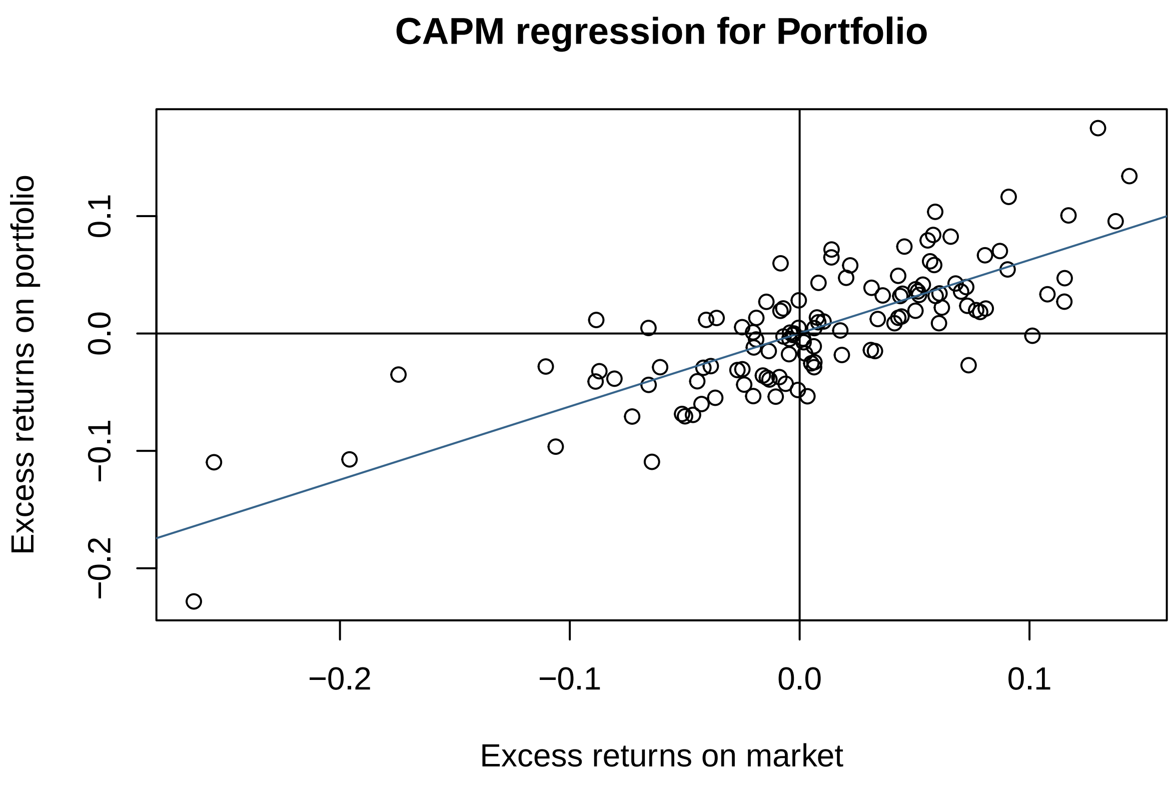
3.1.6 CAPM for portfolio

```
# estimate CAPM for portfolio
#
port = rowMeans(excessReturns.mat[,-c(10,17)])
new.df = data.frame(cbind(port,excessReturns.mat[,"MARKET"]))
colnames(new.df) = c("port","market")
port.fit = lm(port~market,data=new.df)
summary(port.fit)
```

```
##
## Call:
## lm(formula = port ~ market, data = new.df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.073109 -0.028372 -0.001034  0.021696  0.093728
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0002231  0.0030401   0.073    0.942
## market      0.6238986  0.0442571  14.097   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.03312 on 118 degrees of freedom
## Multiple R-squared:  0.6274, Adjusted R-squared:  0.6243
## F-statistic: 198.7 on 1 and 118 DF,  p-value: < 2.2e-16
```

```
plot(new.df$market,new.df$port,
     main="CAPM regression for Portfolio",
     ylab="Excess returns on portfolio",
     xlab="Excess returns on market")
abline(port.fit, col="steelblue4")                        # plot regression line
abline(h=0,v=0)                       # plot horizontal and vertical lines at 0
```



3.2 Three Factor Model

3.2.1 Fama-French Three Factor Model

3-factor model capture the anomalies of B/M (Book Equity to Market Equity ratio) and size.

- Value effect: HML
  - **H**igh book-to-market **M**inus **L**ow book-to-market
  - HML captures book-to-market ratio, which looks at stocks that my be undervalued
- Size effect: SMB
  - **S**mall capitalization **M**inus **B**ig capitalization
  - SMB is based on empirical observations that the size of the firm matters in stock returns

3-factor model:

$$R_{i,t} - r_{f,t} = \alpha_i + \beta_i[R_{M,t} - r_{f,t}] + \beta_{i,\text{SMB}} SMB_t + \beta_{i,\text{HML}} HML_t + \epsilon_{i,t}$$

3.2.2 Factor Model: Six Portfolios

	Small size	Median Market Equity (ME)	Big size
low B/M (30th B/M percentile)	SG: Small Growth		BG: Big Growth
Medium B/M	SN: Small Neutral		BN: Big Neutral
High B/M (70th B/M percentile)	SV: Small Value		BV: Big Value

SMB (small minus big): average return of three minus three big

$$SMB = \frac{1}{3}(SV + SN + SG) - \frac{1}{3}(BV + BN + BG)$$

HML (high minus low): average return of two value minus two growth

$$HML = \frac{1}{2}(SV + BV) - \frac{1}{2}(SG + BG)$$

Fortunately, these factors are available from Ken French's website

The numbers are reported as whole number percentages. For example, 1.25 for 1.25%. Need to adjust by dividing 100 to get the decimal value.

```
# download.file("http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/F-F_Research
# download.file("http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/Portfolios_F
# unzip("F-F_Research_Data_Factors.zip")
# unzip('Portfolios_Formed_on_ME.zip')
```

```
fffactors=read.delim('F-F_Research_Data_Factors.txt',
                     col.names=c('t','mkt.rf','smb','hml','rf'),
                     sep="",skip=4,nrows=1067,header=FALSE,stringsAsFactors=FALSE)

head(fffactors)
```

```
##        t mkt.rf   smb   hml   rf
## 1 192607   2.96 -2.56 -2.43 0.22
## 2 192608   2.64 -1.17  3.82 0.25
## 3 192609   0.36 -1.40  0.13 0.23
## 4 192610  -3.24 -0.09  0.70 0.32
## 5 192611   2.53 -0.10 -0.51 0.31
## 6 192612   2.62 -0.03 -0.05 0.28
```

```
tail(fffactors)
```

```
##           t mkt.rf   smb   hml rf
## 1062 201412  -0.06  2.49  2.27  0
## 1063 201501  -3.11 -0.55 -3.58  0
## 1064 201502   6.13  0.61 -1.86  0
## 1065 201503  -1.12  3.04 -0.37  0
## 1066 201504   0.59 -3.03  1.82  0
## 1067 201505   1.36  0.92 -1.14  0
```

```
fffactors=fffactors[,-1] # exclude time
```

```r
portfolio=read.delim('Portfolios_Formed_on_ME.txt',
                     col.names=c("t","smaller.0","Lo.30","Med.40","Hi.30","Lo.20","Qnt.2"
                     sep="",nrows=1067,header=FALSE,skip=13,stringsAsFactors=FALSE)
knitr::kable(head(portfolio), "simple")
```

t	smaller.0	Lo.30	Med.40	Hi.30	Lo.20	Qnt.2	Qnt.3	Qnt.4	Hi.20	Lo.10	Dec.2	Dec.3	D
192607	-99.99	0.14	1.54	3.42	0.37	0.48	1.68	1.41	3.67	-0.12	0.52	-0.05	
192608	-99.99	3.19	2.73	2.91	2.21	3.58	3.70	1.50	3.07	1.13	2.55	4.00	
192609	-99.99	-1.73	-0.88	0.80	-1.39	-1.25	0.07	-0.23	0.81	0.59	-2.00	-2.01	-
192610	-99.99	-2.94	-3.26	-2.79	-2.56	-3.99	-2.65	-3.36	-2.74	-4.29	-2.01	-3.25	-
192611	-99.99	-0.38	3.73	2.74	-0.95	3.03	3.50	3.25	2.71	-3.28	-0.23	0.08	
192612	-99.99	4.15	1.66	3.04	2.45	3.47	1.37	2.81	3.00	-2.49	3.93	5.60	

```r
knitr::kable(tail(portfolio), "simple")
```

	t	smaller.0	Lo.30	Med.40	Hi.30	Lo.20	Qnt.2	Qnt.3	Qnt.4	Hi.20	Lo.10	Dec.2	De
1062	201412	-99.99	4.51	0.70	-0.35	5.38	2.37	0.91	0.48	-0.44	4.78	5.94	3
1063	201501	-99.99	-4.35	-2.89	-3.06	-4.65	-4.06	-2.72	-1.75	-3.22	-3.71	-5.50	-4
1064	201502	-99.99	6.54	6.99	6.02	5.62	6.67	7.37	7.29	5.89	4.68	6.50	7
1065	201503	-99.99	1.64	1.12	-1.60	1.45	2.03	1.04	0.63	-1.78	1.46	1.45	1
1066	201504	-99.99	-2.37	-1.08	1.00	-2.41	-2.25	-1.65	-0.82	1.21	-1.96	-2.84	-2
1067	201505	-99.99	2.50	1.98	1.21	2.33	2.71	2.62	1.82	1.10	1.65	2.96	2

```r
portfolio=portfolio[,-1] # exclude time
```

Each record contains returns for:

Negative (not used) 30% 40% 30% 5 Quintiles 10 Deciles

Missing data are indicated by -99.99 or -999.

```r
portfolio.rf=portfolio-fffactors$rf # Excess returns
sample=cbind(portfolio.rf,fffactors) # Combine samples

dates=seq(as.Date("1926-07-01"),as.Date("2015-05-01"),by="month") # Generate time stamp
sample=cbind(dates,sample) # Combine time values with the sample
```

```r
lm.Dec.2=lm(Dec.2 ~ mkt.rf + smb + hml, data=sample)
summary(lm.Dec.2)
```

```
##
## Call:
## lm(formula = Dec.2 ~ mkt.rf + smb + hml, data = sample)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.0915 -0.8671 -0.0573  0.8017 14.4405
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.17106    0.05391  -3.173  0.00155 **
## mkt.rf       1.06769    0.01070  99.823  < 2e-16 ***
## smb          1.27648    0.01755  72.754  < 2e-16 ***
## hml          0.49410    0.01561  31.652  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.742 on 1063 degrees of freedom
## Multiple R-squared:  0.9611, Adjusted R-squared:  0.961
## F-statistic:  8753 on 3 and 1063 DF,  p-value: < 2.2e-16
```

```
n=19 # Number of portfolios (but the first one should not be used)
coeffs=as.data.frame(matrix(nrow=2*4+2,ncol=n)) # Create the empty table
names(coeffs)=names(sample)[2:(n+1)] # Specify the names of the table's columns
attributes(coeffs)$row.names=c("Intercept","t.Intercept","mkt.rf","t.mkt.rf","smb","t.smb'
```

```
for (i in 1:n){
  lm=lm(sample[,(i+1)] ~ mkt.rf + smb + hml, data=sample) # Estimate the model
  coeffs[c(1,3,5,7),i]=summary(lm)$coefficients[,1] # Paste the coefficients into the rig
  coeffs[c(2,4,6,8),i]=summary(lm)$coefficients[,3] # Paste the t-statistics into the rig
  coeffs[9,i]=length(lm$model[,1]) # Paste the number of used observations into the right
  coeffs[10,i]=summary(lm)$r.squared # Paste the R-squared into the right row
}
coeffs=round(coeffs,digits=3) # Round the results for better readability
```

```
# 30% 40% 30%
knitr::kable(coeffs[, 2:4], "simple")
```

	Lo.30	Med.40	Hi.30
Intercept	-0.127	-0.004	0.015
t.Intercept	-3.430	-0.149	2.108
mkt.rf	1.049	1.056	0.991
t.mkt.rf	142.635	194.659	721.442
smb	1.190	0.559	-0.132
t.smb	98.635	62.836	-58.394
hml	0.468	0.190	-0.004
t.hml	43.553	23.978	-1.824
Obs	1067.000	1067.000	1067.000
R-squared	0.980	0.983	0.998

```
# 5 Quintiles
knitr::kable(coeffs[,5:9], "simple")
```

	Lo.20	Qnt.2	Qnt.3	Qnt.4	Hi.20
Intercept	-0.165	-0.060	-0.005	0.015	0.016
t.Intercept	-2.767	-2.027	-0.144	0.449	1.631
mkt.rf	1.035	1.055	1.070	1.051	0.986
t.mkt.rf	87.674	180.109	171.262	162.557	520.708
smb	1.372	0.932	0.590	0.295	-0.166
t.smb	70.840	97.007	57.573	27.870	-53.482
hml	0.585	0.344	0.215	0.126	-0.012
t.hml	33.964	40.195	23.598	13.400	-4.258

	Lo.20	Qnt.2	Qnt.3	Qnt.4	Hi.20
Obs	1067.000	1067.000	1067.000	1067.000	1067.000
R-squared	0.955	0.985	0.979	0.972	0.996

```r
# 10 Deciles
knitr::kable(coeffs[,10:19], "simple")
```

	Lo.10	Dec.2	Dec.3	Dec.4	Dec.5	Dec.6	Dec.7	Dec.8	Dec.
Intercept	-0.167	-0.171	-0.085	-0.045	-0.030	0.013	0.006	0.017	-0.00
t.Intercept	-1.655	-3.173	-2.259	-1.222	-0.946	0.328	0.160	0.469	-0.11
mkt.rf	0.997	1.068	1.074	1.044	1.058	1.077	1.050	1.051	1.03
t.mkt.rf	49.891	99.823	143.723	142.146	166.134	136.501	138.721	142.380	166.17
smb	1.559	1.276	1.009	0.880	0.723	0.495	0.410	0.231	0.06
t.smb	47.553	72.754	82.330	72.997	69.245	38.237	32.985	19.057	6.40
hml	0.798	0.494	0.381	0.323	0.196	0.229	0.137	0.122	0.11
t.hml	27.353	31.652	34.954	30.107	21.134	19.901	12.409	11.372	12.48
Obs	1067.000	1067.000	1067.000	1067.000	1067.000	1067.000	1067.000	1067.000	1067.00
R-squared	0.895	0.961	0.977	0.975	0.979	0.965	0.964	0.962	0.97

3.3 More Factor

3.3.1 Carhart Four Factor Model

Momentum: UMD

- **U**p **M**inus **D**own (1-year momentum)
- UMD = Top 30% minus Bottom 30%

Fama-French with Momentum:

$$R_{i,t} - r_{f,t} = \alpha_i + \beta_i[R_{M,t} - r_{f,t}] + \beta_{i,\text{SMB}} SMB_t + \beta_{i,\text{HML}} HML_t + \beta_{i,\text{UMD}} UMD_t + \epsilon_{i,t}$$

3.3.2 Fama-French Five Factor Model

No momentum Factor:

- Not technical analysis
- Fundamental analysis: quality factors - profitability and investment

Profitability factor (RMW):

- **R**obust **M**inus **W**eak operating profitability
- Operating profitability = operating profit - interest

Investment factor (CMA):

- **C**onservative **M**inus **A**ggressive
- Investment = growth of total asset
- managers are destroying values as empire building

```r
rm(list = ls())
```

4 Event Study

To measure the effects of events on stock market.

Assume the one-day event day is t = 0 and one-year estimation window is from t = -256 to t = -6.

4.1 Hypothesis Testing

4.1.1 Normal Return

Firm i's stock return at time t is $R_{i,t}$

Two types of normal return:

- Constant return model

$$R_{i,t} = \mu_i + \epsilon_{i,t}$$

where $E(\epsilon_{i,t}) = 0$ and $Var(\epsilon_{i,t}) = \sigma^2_\epsilon$

- Market model

$$R_{i,t} = \alpha_i + \beta_i R_{M,t} + \epsilon_{i,t}$$

where $E(\epsilon_{i,t}) = 0$ and $Var(\epsilon_{i,t}) = \sigma^2_\epsilon$

4.1.2 Abnormal Return

Abnormal return $AR_{i,t}$

- Under constant return model

$$AR_{i,t} = R_{i,t} - \hat{R}_{i,t} = R_{i,t} - \bar{R}_i$$

where $\bar{R}_i$ is average return during estimation period.

- Under market model

$$AR_{i,t} = R_{i,t} - \hat{R}_{i,t} = R_{i,t} - \hat{\alpha}_i - \hat{\beta}_i R_{M,t}$$

where $\hat{\alpha}_i$ and $\hat{\beta}_i$ are estimates using OLS of security i return on market return using data from estimation period.

Suppose estimation windows is from -256 to -6. Length of estimation windows is L = 250.

Estimated variance of abnormal return $\hat{\sigma}^2_{\epsilon_i}$:

- Under constant return model

$$Var(\widehat{AR_{i,t=0}}) = \hat{\sigma}^2_{\epsilon_i} = \frac{1}{L-1} \sum_{t=-256}^{-6} (R_{i,t} - \hat{R}_{i,t})^2$$

- Under market model

$$\widehat{Var(AR_{i,t=0})} = \hat{\sigma}^2_{\epsilon_i} = \frac{1}{L-2}\sum_{t=-256}^{-6}(R_{i,t} - \hat{R}_{i,t})^2$$

$$H_0 : AR_{i,t} = 0 \text{ abnormal return = 0}$$
$$H_1 : AR_{i,t} \neq 0$$

If the length of estimation window L is long enough, we have

$$\frac{AR_{i,t=0}}{\sqrt{Var(AR_{i,t=0})}} \sim N(0,1)$$

where $Var(AR_{i,t=0}) = \hat{\sigma}^2_{\epsilon_i}$

4.1.3 Cross Sectional Analysis

Consider there is a single event (macro or political event) that affects many firms or even the whole stock market. Say N firms.

Event study is to conduct a hypothesis testing on the **mean** of abnormal return (AR_t)

$$AR_t = \frac{1}{N}\sum_{i=1}^{N} AR_{i,t}$$

Null hypothesis: $AR_{t=0} = 0$

When the estimation window is long enough, we have

$$\frac{AR_{t=0}}{\sqrt{Var(AR_{t=0})}} \sim N(0,1)$$

where $Var\widehat{(AR_{t=0})} = \frac{1}{N^2}\hat{\sigma}^2_{\epsilon_i}$

4.1.4 Event More than One Day

Event window may be more than one day. For example, it may take a few days or month for the effect of the effect to materialize. Then event windows may be like -1 to +1 or -5 to +5, or even -10 to +10.

In this case, we need to aggregate abnormal return during the event period, i.e., cumulative abnormal return (CAR)

$$CAR_i(t_1, t_2) = \sum_{t=t_1}^{t_2} AR_{i,t}$$

$$CAR(t_1, t_2) = \frac{1}{N}\sum_{i=1}^{N} CAR_i(t_1, t_2)$$

When the estimation window is long enough, we have

$$\frac{CAR(t_1,t_2)}{\sqrt{Var(CAR(t_1,t_2))}} \sim N(0,1)$$

where $Var(\widehat{CAR}(t_1,t_2)) = \frac{1}{N^2}\sum_{i=1}^{N}(t_2-t_1+1)\hat{\sigma}^2_{\epsilon_i}$

4.2 Regression Approach

Consider the following return equation for firm i:

$$R_{i,t} = \alpha_i + \beta_i R_{M,t} + \gamma_i D_t + \epsilon_{i,t}$$

where D_t is dummy variable for event windows.

Then estimate γ_i and conduct hypothesis testing.

4.2.1 More than One Event

Assume there are $A \geq 2$ events.

We have two methods:

- One single dummy: dummy = 1 for the event

$$R_{i,t} = \alpha_i + \beta_i R_{M,t} + \gamma_i D_t + \epsilon_{i,t}$$

where $D_t = 1$ for time with an event.

- Multiple dummies: one dummy for each event

$$R_{i,t} = \alpha_i + \beta_i R_{M,t} + \sum_{a=1}^{A}\gamma_{i,a} D_{a,t} + \epsilon_{i,t}$$

Then estimate $\gamma_{i,a}$ and conduct hypothesis testing for each coefficient or their joint effects.

4.2.2 A Group of Firms

For more than one firm, consider a portfolio of firms.

For $A \geq 2$ events, we have

$$R_{p,t} = \alpha_p + \beta_p R_{M,t} + \sum_{a=1}^{A}\gamma_{p,a} D_{a,t} + \epsilon_{p,t}$$

where $R_{p,t}$ is the portfolio return, $D_{a,t}$ is a dummy for event a.

The key is to estimate $\gamma_{p,a}$ and do hypothesis testing.

4.2.3 When Firms Are different in Nature

Grouping will not be useful: cancel out.

Consider a system of return equation for N firms and each experiencing A events.

$$\begin{aligned}
R_{1,t} &= \alpha_1 + \beta_1 R_{M,t} + \sum_{a=1}^{A} \gamma_{1,a} D_{a,t} + \epsilon_{1,t} \\
R_{2,t} &= \alpha_2 + \beta_2 R_{M,t} + \sum_{a=1}^{A} \gamma_{2,a} D_{a,t} + \epsilon_{2,t} \\
&\vdots \\
R_{N,t} &= \alpha_N + \beta_N R_{M,t} + \sum_{a=1}^{A} \gamma_{N,a} D_{a,t} + \epsilon_{N,t}
\end{aligned}$$