atmi

Analysis of technical market indicators

Version:	1.0
Date:	2009-10-05
Author:	Waldemar Kemler, Peter Schaffner, <atmi.info@gmail.com>
Maintainer:	Waldemar Kemler, Peter Schaffner, <atmi.info@gmail.com>
Description:	Analysis and usage of the trading rules, which are based on technical market indicators as well as on the time series analysis.
Depends:	R (>= 2.9.1), TTR
License:	GPL (>=2)

Contents

1 Main functions

1.1 atmimomentum

Momentum indicator trading rules

Description

Trading rules are based on the momentum indicator. Using the momentum indicator it should be possible to describe the strength of the price change, as well as to identify the possible imminent trend reversal.

Usage

```
atmimomentum(symbol = FALSE, datamatrix = list(NA, NA),
    period = c(FALSE, FALSE), n = 10, sma = 10, rule = 1,
    ticks = 1, plot = TRUE, stopl = FALSE, interest = FALSE,
    transcost = FALSE, startcap = 1000, standard = TRUE,
    conf.level = 0.95)
```

Arguments

symbol	a unique series of letters assigned to a security for trading purposes. You can get the symbols by using the function getsymbols or from http://finance.yahoo.com/. If FALSE, a datamatrix is needed.
datamatrix	a list of two vectors. The first one is a vector of dates, the second one is a vector of prices.
period	is a vector of start and end dates, in "YYYY-MM-DD" format. If the start date is FALSE, all available data until the end date will be used. If the end date is FALSE, all available data from the start date will be used. If both are FALSE, all available data will be used.
n	length of the period for the calculation of the price difference in the computation of the indicator value.
sma	will be used only when rule = 2. This is the Number of observations for the calculation of the simple moving average (signal line).
rule	can be 1 or 2. If equal to 1, then the buy signal occurs, if the Momentum line crosses the zero line from below. The sell signal occurs, if the Momentum line crosses the zero line from above. The rule 2 is the same as 1, except that the buy/sell signals occur by crossing the signal line.

`ticks`	data frequency. For example if ticks=1, every observation will be used in the calculation. If ticks=5, every fifth observation will be used in the calculation.
`plot`	logical; if TRUE, the series with trading signals (buy=green, sell=red) and the indicator will be plotted.
`stopl`	a vector of two arguments for the stop-loss hedging strategy. The first argument can be "s" for static or "d" for dynamic stop-loss hedging. The second argument is the stop-loss limit, it can be between 0 and 1. If `stopl`=FALSE, no hedging occurs.
`interest`	interest of the non invested capital. It can be a number either greater then zero or FALSE.
`transcost`	transaction costs. It can be a number either greater then zero or FALSE.
`startcap`	initial assets. It can be a number greater than zero. Default 1000 monetary units.
`standard`	logical; If FALSE, the buy and sell signals are successively arranged, where the first trade signal is a signal to buy and the last one is a signal to sell. If TRUE, original trading signals, as they are supplied by the trading rule are returned.
`conf.level`	confidence level for the one-sided confidence interval of number of "hits". "Hits" are signal pairs (a signal to buy and a subsequent signal to sell) which have led to a positive return. The remaining signal pairs are "nonhits".

Details

To get a trading recommendation for today, the `standard` has to be TRUE. Otherwise, in case that after the last Buy signal no Sell signal (based on indicator rule) occurred, the last day will be used as a Sell signal.

Value

`signale` matrix with 2 columns of buy and sell signals (indices of data, where signals are occurred).

`numEqual` signals occurr on the intersection of two sequences or a sequence and a horizontal line. `numEqual` is a number of points, where a signal happened and on the day before the first sequence was exactly equal to the second sequence or the horizontal line.

`settings` settings, which were used for the calculation of the strategy.

`results` results of the strategy as:

- `performance` performance of the strategy for the given period.
- `num. signals` number of signal pairs of the strategy for the given period.
- `hits` number of "hits".
- `leftfront` left front of the confidence interval with the confidence level of "conf.level".
- `performance BAH` performance of the buy and hold strategy for the given period.

Results are only available, if `standard` is FALSE.

References

Otto. Loistl. (1992), Computergestütztes Wertpapiermanagement. R. Oldenbourg, München.

John J. Murphy. (1999), Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications (New York Institute of Finance), Prentice Hall Press, 254.

Author(s)

Waldemar Kemler, Peter Schaffner

See Also

atmimacd, atmirsi, atminaiv, performance

Examples

#Example 1: Momentum strategy of "AACC" with standard settings (rule=1, n=10, sma=10).

```
atmimomentum("AACC")
```

#Example 2: Momentum strategy of "AACC" with rule 2 and following settings: n=12, sma=15, startcapital=1000. With interest of invested capital (2.5%), trasaction costs (5.9 monetary units) and standard=FALSE.

```
atmomentum("AACC", rule=2, n=12, sma=15, standard=FALSE,
    interest=0.025, transcost=5.9)
```

#Example 3: Momentum strategy of "AACC" for the period from 2004-01-01 through 2005-12-31 with rule 2 and following settings: n=15, sma=15, startcapital=1000. With interest of non invested capital (2.5%), trasaction costs (5.9 monetary units), standard=FALSE and dynamic stop-loss-hedging (10%).

```
atmimomentum("AACC", period=c("2004-01-01","2005-12-31"),
    n=15, sma=10, rule=2, standard=FALSE, interest=0.025,
    transcost=5.9, stopl=c("d",0.1))
```

1.2 atmirsi

Relative Strength Index indicator trading rules

Description

Trading rules based on the Relative Strength Index (RSI). RSI is the normalized ratio between the average of the upward and downward closing prices of the last n days of a security. It was introduced in 1978 by Welles Wilder Jr. and builds on the momentum indicator, where he corrected two of the momentum inefficiencies. On the one hand, the RSI has a fixed value between 0 and 100, which makes it possible to directly compare the indicator values of two securities. On the other hand the calculation of the ratio between the average downward and upward movements leads to smooth indicators, so that it is no longer as sensitive as the momentum to respond to extreme price movements.

Usage

```
atmirsi(symbol = FALSE, datamatrix = list(NA, NA), period =
    c(FALSE, FALSE), n = 14, top = 70, bottom = 30, ticks = 1,
     plot = TRUE, stopl = FALSE, interest = FALSE, transcost =
     FALSE, startcap = 1000, standard = TRUE, conf.level =
    0.95)
```

Arguments

`symbol`	a unique series of letters assigned to a security for trading purposes. You can get the symbols by using the function getsymbols or from http://finance.yahoo.com/. If FALSE, a datamatrix is needed.
`datamatrix`	a list of two vectors. The first one is a vector of dates, the second one is a vector of prices.
`period`	is a vector of start and end dates, in "YYYY-MM-DD" format. If the start date is FALSE, all available data until the end date will be used. If the end date is FALSE, all available data from the start date will be used. If both are FALSE, all available data will be used.
`n`	number of observations for calculation of the average of UP and DOWN closes in the computation of the indicator value.
`top`	top limit for calculation of the sell signals. It can be between 0 and 100.
`bottom`	bottom limit for calculation of the buy signals. It can be between 0 and 100.
`ticks`	data frequency. For example if ticks=1, every observation will be used in the calculation. If ticks=5, every fifth observation will be used in the calculation.
`plot`	logical; if TRUE, the series with trading signals (buy=green, sell=red) and the indicator will be plotted.
`stopl`	a vector of two arguments for the stop-loss hedging strategy. The first argument can be "s" for static or "d" for dynamic stop-loss hedging. The second argument is the stop-loss limit, it can be between 0 and 1. If `stopl`=FALSE, no hedging occurs.
`interest`	interest of the non invested capital. It can be a number either greater then zero or FALSE.
`transcost`	transaction costs. It can be a number either greater then zero or FALSE.
`startcap`	initial assets. It can be a number greater than zero. Default 1000 monetary units.
`standard`	logical; If FALSE, the buy and sell signals are successively arranged, where the first trade signal is a signal to buy and the last one is a signal to sell. If TRUE, original trading signals, as they are supplied by the trading rule are returned.
`conf.level`	confidence level for the one-sided confidence interval of number of "hits". "Hits" are signal pairs (a signal to buy and a subsequent signal to sell) which have led to a positive return. The remaining signal pairs are "nonhits".

Details

To get a trading recommendation for today, the `standard` has to be TRUE. Otherwise, in case that after the last Buy signal no Sell signal (based on indicator rule) occurred, the last day will be used as a Sell signal.

Value

`signale`	matrix with 2 columns of buy and sell signals (indices of data, where signals are occurred).
`numEqual`	signals occurr on the intersection of two sequences or a sequence and a horizontal line. `numEqual` is a number of points, where a signal happened and on the day before the first sequence was exactly equal to the second sequence or the horizontal line.
`settings`	settings, which were used for the calculation of the strategy.
`results`	results of the strategy as:
`performance`	performance of the strategy for the given period.
`num. signals`	number of signal pairs of the strategy for the given period.
`hits`	number of "hits".
`leftfront`	left front of the confidence interval with the confidence level of "conf.level".
`performance BAH`	performance of the buy and hold strategy for the given period.

Results are only available, if `standard` is FALSE.

References

J. Welles. Wilder. (1978), New concepts in technical trading systems. Trend Research, Greensboro, N.C, 64-70.

Otto. Loistl. (1992), Computergestütztes Wertpapiermanagement. R. Oldenbourg, München, 125.

John J. Murphy. (1999), Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications (New York Institute of Finance), Prentice Hall Press.

Author(s)

Waldemar Kemler, Peter Schaffner

See Also

atmimacd, atmimomentum, atminaiv, performance

Examples

#Example 1: RSI strategy of "AACC" with standard settings (n=14, top=70, bottom=30).

```
atmirsi("AACC")
```

#Example 2: RSI strategy of "AACC" with following settings: n=14, top=70, bottom=30, startcapital=1000. With interest of invested capital (2.5%), trasaction costs (5.9 monetary units) and standard=FALSE.

```
atmirsi("AACC", standard=FALSE, interest=0.025,
    transcost=5.9)
```

#Example 3: RSI strategy of "AACC" for the period from 2004-01-01 through 2005-12-31 with following settings: n=7, top=80, bottom=20, startcapital=1000. With interest of non invested capital (2.5%), trasaction costs (5.9 monetary units), standard=FALSE and dynamic stop-loss-hedging (10%).

```
atmirsi("AACC", period=c("2004-01-01","2005-12-31"), n=7,
    top=80, bottom=20, standard=FALSE, interest=0.025,
    transcost=5.9, stopl=c("d",0.1))
```

1.3 atmimacd

Moving Average Convergence Divergence indicator trading rules

Description

Trading rules are based on the Moving average Convergence/Divergence (MACD), which were introduced by Gerald Appel in 1979. MACD is based on the concept of exponential moving average and today is one of the most well-known and most widely used indicators. The specific feature of this indicator is, that it can be used depending on the interpretation as a trend follower as well as an oscillator.

Usage

```
atmimacd(symbol = FALSE, datamatrix = list(NA, NA), period =
    c(FALSE, FALSE), nfast = 12, nslow = 26, trigger = 9, rule
    = 1, ticks = 1,
  plot = TRUE, stopl = FALSE, interest = FALSE, transcost =
    FALSE, startcap = 1000, standard = TRUE, conf.level =
    0.95)
```

Arguments

`symbol`	a unique series of letters assigned to a security for trading purposes. You can get the symbols by using the function getsymbols or from http://finance.yahoo.com/. If FALSE, a datamatrix is needed.
`datamatrix`	a list of two vectors. The first one is a vector of dates, the second one is a vector of prices.
`period`	is a vector of start and end dates, in "YYYY-MM-DD" format. If the start date is FALSE, all available data until the end date will be used. If the end date is FALSE, all available data from the start date will be used. If both are FALSE, all available data will be used.
`nfast`	number of dates for the fast moving average.
`nslow`	number of dates for the slow moving average.
`trigger`	number of dates for the signal moving average.
`rule`	can be 1 or 2. If equal to 1, then the buy signal occurs, if the MACD line crosses the signal line from below. The sell signal occurs, if the MACD line crosses the signal line from above. The rule 2 is the same as 1, except that the buy/sell signals occur only below/above the zero line.
`ticks`	data frequency. For example if ticks=1, every observation will be used in the calculation. If ticks=5, every fifth observation will be used in the calculation.
`plot`	logical; if TRUE, the series with trading signals (buy=green, sell=red) and the indicator will be plotted.
`stopl`	a vector of two arguments for the stop-loss hedging strategy. The first argument can be "s" for static or "d" for dynamic stop-loss hedging. The second argument is the stop-loss limit, it can be between 0 and 1. If `stopl`=FALSE, no hedging occurs.
`interest`	interest of the non invested capital. It can be a number either greater then zero or FALSE.
`transcost`	transaction costs. It can be a number either greater then zero or FALSE.
`startcap`	initial assets. It can be a number greater than zero. Default 1000 monetary units.
`standard`	logical; If FALSE, the buy and sell signals are successively arranged, where the first trade signal is a signal to buy and the last one is a signal to sell. If TRUE, original trading signals, as they are supplied by the trading rule are returned.
`conf.level`	confidence level for the one-sided confidence interval of number of "hits". "Hits" are signal pairs (a signal to buy and a subsequent signal to sell) which have led to a positive return. The remaining signal pairs are "nonhits".

Details

To get a trading recommendation for today, the `standard` has to be TRUE. Otherwise, in case that after the last Buy signal no Sell signal (based on indicator rule) occurred, the last day will be used as a Sell signal.

Value

`signale` matrix with 2 columns of buy and sell signals (indices of data, where signals are occurred).

`numEqual` signals occurr on the intersection of two sequences or a sequence and a horizontal line. `numEqual` is a number of points, where a signal happened and on the day before the first sequence was exactly equal to the second sequence or the horizontal line.

`settings` settings, which were used for the calculation of the strategy.

`results` results of the strategy as:

- `performance` performance of the strategy for the given period.
- `num. signals` number of signal pairs of the strategy for the given period.
- `hits` number of "hits".
- `leftfront` left front of the confidence interval with the confidence level of "conf.level".
- `performance BAH` performance of the buy and hold strategy for the given period.

Results are only available, if `standard` is FALSE.

References

Appel, G. (2005), Technical analysis : power tools for active investors : [new techniques for active trading in the stock market!]. Financial Times Prentice Hall books. Financial Times/Prentice Hall, Upper Saddle River, N.J, 165.

Otto. Loistl. (1992), Computergestütztes Wertpapiermanagement. R. Oldenbourg, München.

John J. Murphy. (1999), Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications (New York Institute of Finance), Prentice Hall Press, 254.

Author(s)

Waldemar Kemler, Peter Schaffner

See Also

atmimomentum, atmirsi, atminaiv, performance

Examples

#Example 1: MACD strategy of "AACC" with standard settings (rule=1, nfast=12, nslow=26, trigger=9).

```
atmimacd("AACC")
```

#Example 2: MACD strategy of "AACC" with rule 2 and following settings: nfast=12, nslow=26, trigger=9, startcapital=1000. With interest of invested capital (2.5%), trasaction costs (5.9 monetary units) and standard=FALSE.

```
atmimacd("AACC", rule=2, standard=FALSE, interest=0.025,
    transcost=5.9)
```

#Example 3: MACD strategy of "AACC" for the period from 2004-01-01 through 2005-12-31 with rule 2 and following settings: nfast=15, nslow=35, trigger=10, startcapital=1000. With interest of non invested capital (2.5%), trasaction costs (5.9 monetary units), standard=FALSE and dynamic stop-loss-hedging (10%).

```
atmimacd("AACC", period=c("2004-01-01","2005-12-31"),
    nfast=15, nslow=35, trigger=10, rule=2, standard=FALSE,
    interest=0.025, transcost=5.9, stopl=c("d",0.1))
```

1.4 atminaiv

Naiv trading rules

Description

Analysis and usage of the trading rules, which are based on the double exponential smoothing from Holt.

Usage

```
atminaiv(symbol = FALSE, trade = FALSE, datamatrix = list(NA,
    NA), period = c(FALSE, FALSE, FALSE), nAhead = 1, nFit =
    FALSE, startcap = 1000, sellfront = 0, transcost = FALSE,
    interest = FALSE, ticks = 1, plot = TRUE)
```

Arguments

`symbol`	a unique series of letters assigned to a security for trading purposes. You can get the symbols by using the function getsymbols or from http://finance.yahoo.com/. If FALSE, a datamatrix is needed.
`trade`	logical; if TRUE, on the display appears as an output a trading recommendation as well as the predicted performance for the next day. The trading recommendation can be BUY, SELL or HOLD. If FALSE, the trading strategy will be applied for the given time period based on historical data.
`datamatrix`	a list of two vectors. The first one is a vector of dates, the second one is a vector of prices.
`period`	is a vector of start, end dates and start of prediction, in "YYYY-MM-DD" format. If the start date is FALSE, all available data until the end date will be used. If the end date is FALSE, all available data from the start date will be used. If both are FALSE, all available data will be used. If start of prediction is not FALSE, the prediction will start at this point in time.
`nAhead`	forecast horizon.
`nFit`	period, which is used for fitting of the Holt model parameters alpha and beta.
`startcap`	initial assets. It can be a number greater than zero. Default 1000 monetary units.
`sellfront`	lower limit for the predicted performance. The signal sell occurs when the predicted performace under a given `sellfront` fails.
`interest`	interest of the non invested capital. It can be a number either greater then zero or FALSE.
`transcost`	transaction costs. It can be a number either greater then zero or FALSE.
`ticks`	data frequency. For example if ticks=1, every observation will be used in the calculation. If ticks=5, every fifth observation will be used in the calculation.
`plot`	logical; if TRUE, series with trading signals (buy=green, sell=red) will be plotted.

Details

For the optimal choice of smoothing factors alpha and beta, which get daily fitted, the method of minimizing the squared one-step prediction error is used.
If `trade` is TRUE, the arguments `interest` and `period` will be automaticaly set to FALSE.

Value

`buy`	a vector of indices of a data vector, where the buy signals occurred.
`sell`	a vector of indices of a data vector, where the sell signals occurred.
`fin.cap`	final capital.

Notes

The following tradig rules are used to calculate buy and sell signals.

$$\begin{aligned} \text{BUY:} \quad & \widehat{r}_{t+1} > \frac{2 * transcost}{cap_t} \\ \text{SELL:} \quad & \widehat{r}_{t+1} < sellfront \\ \text{with} \quad & \widehat{r}_{t+1} = \frac{\widehat{C}_{t+1}}{C_t} - 1 \end{aligned}$$

$\widehat{r}_{t+1}$ is the predicted performance and $\widehat{C}_{t+1}$ the with the Holt model predicted security price. A buy signal occurs, if the predicted performance is greater than the quotient of the double transaction costs and the capital. A sell signal occurs, if the predicted performance is smaller than the given `sellfront`.

References

Charles C. Holt (2007). Forecasting seasonals and trends by exponentially weighted moving averages. International Journal of Forecasting, 20(1), 7
Francis X. Diebold (2007). Elements of forecasting. Thomson Learning, London, 315, 316
Winfried Stier (2001). Methoden der Zeitreihenanalyse. Springer, Berlin [u.a.], 24

Author(s)

Waldemar Kemler, Peter Schaffner

Warning

To run the function, an active internet connection is required.

See Also

naivpredict, HoltWinters

Examples

```
#Example 1:  Atminaiv strategy with standard settings.

atminaiv(symbol="ABMC")

#Example 2:  Atminaiv strategy for "AANB". The data is available
for the period from 2007-01-01 through 2008-12-31, but the prediction
starts on 2008-01-01.

atminaiv(symbol="AANB", period=c("2007-01-01", "2008-12-31",
    "2008-01-01"), transcost=5.9, interest=0.025,
    startcap=10000)

#On the display appears as an output of the call the final capital
as well as the performance of the strategy.
#Example 3:  The call of the following command, provides a trading
recommendation for the next day based on the atminaiv strategy.


atminaiv(symbol="AANB", trade=TRUE, transcost=5.9,
    startcap=1000)
```

2 Auxiliary functions

2.1 getsymbols

get stock symbols

Description

Get a list of security symbols of a stock market index from
http://finance.yahoo.com/

Usage

```
getsymbols(indexsymbol = "GDAXI")
```

Arguments

indexsymbol	finance.yahoo.com symbol of a stock market index.

Details

Examples for indexsymbol:

DJI	Dow Jones Industrial Average
NDX	NASDAQ-100, 100 non-financial companies listed on the NASDAQ
GSPC	S&P 500, Standard & Poor 500
IXIC	NASDAQ Composite
GDAXI	German stock market index
TECDAX	30 largest German companies from the technology sector

Value

symbols	a list of security symbols of a stock market index.

Author(s)

Waldemar Kemler, Peter Schaffner

Warning

To run the function, an active internet connection is required.

See Also

getdata

Examples

```
# Getting a list of symbols for the Dow Jones Industrial Average

dji<-getsymbols("DJI")
dji
```

2.2 getdata

get stock data

Description

Get the daily finance data for the desired period from
http://finance.yahoo.com/

Usage

```
getdata(symbol = FALSE, period = c(startdate = FALSE, enddate
    = FALSE))
```

Arguments

`symbol`	a unique series of letters assigned to a security for trading purposes. You can get the symbols by using the function getsymbols or from http://finance.yahoo.com/. If FALSE, a datamatrix is needed.
`period`	is a vector of start and end dates, in "YYYY-MM-DD" format. If the start date is FALSE, all available data until the end date will be used. If the end date is FALSE, all available data from the start date will be used. If both are FALSE, all available data will be used.

Value

`datamatrix` - a matrix object with the following columns:

`Date`	trade date in "YYYY-MM-DD" format
`Open`	open price
`High`	the highest price of the day
`Low`	the lowest price of the day
`Close`	closing price
`Volume`	average volume
`Adj.Close`	adjusted closing price

Note

The last column "Adj.Close" of the returned matrix "datamatrix" provides the closing price adjusted for all applicable splits and dividend distributions.

Author(s)

Waldemar Kemler, Peter Schaffner

Warning

To run the function, an active internet connection is required.

See Also

getsymbols

Examples

```
# Returns data for "AACC" for whole available period.

datamat<-getdata("AACC")

# Returns data for "AACC" for the period from 2008-01-01 through 2008-12-31.

datamat<-getdata("AACC", period=c("2008-01-01","2008-12-31"))

# Returns all available data for "AACC" until 12/31/08.

datamat<-getdata("AACC", period=c(FALSE,"2008-12-31"))

# Returns all available data from 01/01/08. With the usage of function getsymbols.

symbol<-getsymbols("DJI")
datamat<-getdata(symbol[1], period=c("2008-01-01",FALSE))
datamat
```

2.3 intersection

intersection points

Description

Identifies intersection points of two sequences or a sequence and a horizontal line.

Usage

```
intersection(sequenceInd = NA, sequenceSig = NA, hLine = NA,
    plot = FALSE)
```

Arguments

sequenceInd	the first sequence (a data vector).
sequenceSig	the second sequence (a data vector).
hLine	the horizontal line (a number).
plot	logical; if TRUE, the intersection will be plotted.

Details

The used sequences can contain NAs only at the beginning. The function was written for the analysis of technical indicators, which are based on closing prices. Therefore an intersection point can only be identified, when the intersection has already happened. For example if an intersection occurred between day 5 and 6, the 6th day is an intersection day (see example 1).

Value

`upIntersection`	vector of indices, where the sequenceInd crossed the sequenceSig or the hLine from below.
`downIntersection`	vector of indices, where the sequenceInd crossed the sequenceSig or the hLine from above.
`numEqualUp`	number of points, where an UpIntersection happened and on the day before the sequenceInd was exactly equal to sequenceSig or the hLine.
`numEqualDown`	number of points, where an DownIntersection happened and on the day before the sequenceInd was exactly equal to sequenceSig or the hLine (see example 3).

Author(s)

Waldemar Kemler, Peter Schaffner

Examples

```
# Example 1: sequenceInd crosses sequenceSig from below between day 5 and 6.
intersection(sequenceInd=c(1:10),sequenceSig=c(10:1),plot=TRUE)

# Example 2: sequenceInd crosses a hLine.
intersection(sequenceInd=c(1:10),hLine=5,plot=TRUE)

# Example 3: sequenceInd crosses sequenceSig from above and on the day before the sequenceInd is exactly equal to sequenceSig. Furthermore sequenceInd contains NAs at the beginning.
intersection(sequenceInd=c(NA,NA,7:1),
    sequenceSig=c(1:10),plot=TRUE)

# Example 4:
intersection(sequenceInd=c(1,2,4,4,4,7,6,6,5),
    sequenceSig=c(6,5,4,4,4,4,3,2,2),plot=TRUE)
```

2.4 stoploss

stop-loss hedging

Description

Dynamic or static stop-loss hedging strategy.

Usage

```
stoploss(datavec, buy, sell, stopl=c("s",0.1))
```

Arguments

`datavec`	a vector of security prices.
`buy`	a vector of indices of a data vector, where buy signals occurred.
`sell`	a vector of indices of a data vector, where sell signals occurred.
`stopl`	a vector of two arguments for the stop-loss hedging strategy. The first argument can be "s" for static or "d" for dynamic stop-loss hedging. The second one is the stop-loss limit, it can be between 0 and 1. If `stopl`=FALSE, no hedging occurrs.

Details

In a static stop-loss, the securities are sold when the stock price falls below a certain percentage of the purchase price. In a dynamic stop-loss, the basis for the stop-loss calculation is not the last purchase price, but the highest security price after the buy signal.
It is not necessary, that the vectors of buy and sell signals are in an ordered form. They can also contain "NAs".

Value

`sell`	new sell signals, after stop-loss hedging.

References

Johannes Welcker. (1994), Technische Aktienanalyse, volume 7. Verlag Moderne Industrie, Zürich.

Author(s)

Waldemar Kemler, Peter Schaffner

See Also

atmimomentum, atmimacd, atmirsi, performance

Examples

```
datavec<-c(20,25,32,27,22,18,18,24,25,27)

#Example 1: static stop-loss hedging with 10% limit.

sell<-stoploss(datavec, buy=c(2), sell=c(8),
    stopl=c("s",0.1))

#Example 2: dynamic stop-loss hedging with 10% limit.

sell<-stoploss(datavec, buy=c(2), sell=c(8),
    stopl=c("d",0.1))

#Example 3: dynamic stop-loss hedging with 10% limit.

sell<-stoploss(datavec, buy=c(5,NA,2), sell=c(8,NA),
    stopl=c("d",0.1))
```

2.5 superfluous_filter

arrange buy and sell signals

Description

Successively arranges buy and sell signals of a trading strategy.

Usage

```
superfluous_filter(datavec, buy, sell)
```

Arguments

datavec	a vector of security prices.
buy	a vector of indices of a data vector, where the buy signals occurred.
sell	a vector of indices of a data vector, where the sell signals occurred.

Details

It is not allowed, that a buy and a sell signal occur on the same day.

Value

buy	vector of the buy signals.
sell	vector of the sell signals.

Author(s)

Waldemar Kemler, Peter Schaffner

See Also

atmimomentum, atmimacd, atmirsi

Examples

```
datavec<-c(20,25,30,27,22,18,18,24,25,27)
superfluous_filter(datavec, buy=c(NA,2,NA,6),
    sell=c(26,1,8,9,15,18,NA))
```

2.6 performance

performace of a trading strategy

Description

Calculates a performance of a trading strategy with the given signals to buy and sell. If necessary, with consideration of transaction costs and/or interest.

Usage

```
performance(datavec = NA, datevec = NA, buy = NA, sell = NA,
    startcap = 1000, interest = 0.025, transcost = 5.9)
```

Arguments

datavec	a vector of security prices.
datevec	a vector of dates.
buy	a vector of indices of data vector, where the buy signals occurred.
sell	a vector of indices of data vector, where the sell signals occurred.
startcap	initial assets. It can be a number greater than zero. Default is 1000 monetary units.
interest	annual interest rate. It can be a number either greater than zero or FALSE.
transcost	transaction costs. It can be a number either greater than zero or FALSE.

Details

The function requires successively buy and sell signals. This can be done by using the function superfluous_filter. In the calculation of the performace any divisibility of securities and a steady, deterministic interest rate is assumed. It is also assumed, that the opening price of a security is equal to the closing price of the day before and that it is always traded at the opening price. An additional assumption is that a year has 360 days.

Value

return.strategy	performance of the given strategy, which is calculated with the given buy and sell signals.
returns	single returns of the given strategy, which are calculated with the buy and sell signals.
return.buy.and.hold	performance of the buy and hold strategy for the given security and period. Usually it is used as a benchmark.

Note

The performance of the buy and hold strategy return.buy.and.hold will be calculated between the first and the last date independent from the buy and sell signals.
In consideration of transaction costs they will be checked before every purchase, whether sufficient funding is available to finance a commercial transaction (purchase and sale of a security), so that the reference account must show an account balance of more than "2 x transcost" monetary units before every purchase. A purchase fee of transcost monetary units will be deducted directly at the purchase from this amount. The remaining amount of the "2 x transcost" monetary units remains up to the sale as a security deposit on the account. It is made to ensure, that the sale fee can be paid in any case.
If the payment of interest is considered in the analysis, the non invested capital as well as the security amount (sale fee) will also lead to interest.

References

Johannes Welcker. (1994), Technische Aktienanalyse, volume 7. Verlag Moderne Industrie, Zürich.

Author(s)

Waldemar Kemler, Peter Schaffner

See Also

stoploss, superfluous_filter

Examples

```
# Initial situation:

datavec<-c(20,25,30,27,22,18,18,24,25,27)
datevec<-seq(as.Date("2009-01-01"), as.Date("2009-01-10"), by
    = "day")

# Example 1: Performance without interest and transaction costs.

per<-performance(datavec=datavec,datevec=datevec, buy=c(2,5),
    sell=c(4,9), startcap=1000, interest=FALSE,
  transcost=FALSE)

# Example 2: Performance with consideration of interest (2.5 per cent) and transaction costs
(5.9 monetary unit).

per<-performance(datavec=datavec,datevec=datevec, buy=c(2,6),
    sell=c(5,8), startcap=1000, interest=0.025,
  transcost=5.9)
```

Example 2 shows the case with the consideration of interest (2.5 per cent), transaction costs (5.9 monetary units) and initial assets `startcap` (1000 monetary units). The performance of the strategy `return.strategy` of $\sim$0.144155 is originated by the following:

startcap interest until the first buy signal	$1000 * 1.025^{\frac{2}{360}} = 1000.137$
Purchase at the first buy signal (number of securities)	$\frac{(1000.137 - 2 * 5.9)}{25} = 39.53349$
Sell at the first sell signal	$39.53349 * 22 = 869.7367$
Interest of security deposite for 3 days	$5.9 * 1.025^{\frac{3}{360}} - 5.9 = 0.001214178$ $0.001214178 + 869.7367 = 869.738$
cap + interest until the second buy signal	$869.738 * 1.025^{\frac{1}{360}} = 869.7976$
Purchase at the second buy signal (number of securities)	$\frac{(869.7976 - 5.9 * 2)}{18} = 47.66653$

Sell at the second sell signal	$47.66653 * 24 = 1143.997$
Interest of security deposite for 3 days	$5.9 * 1.025^{\frac{2}{360}} - 5.9 = 0.0008094245$
	$0.0008094245 + 1143.997 = 1143.998$
Interest until the end	$1143.998 * 1.025^{\frac{2}{360}} - 1143.998 = 0.1569458$
Endcapital	$1143.998 + 0.1569459 = 1144.155$
Performance of the strategy	$\frac{1144.155}{1000} - 1 = 0.144155$

The performance of the buy and hold strategy `return.buy.and.hold` is equal to 0.3340736 and was calculated by the following:

$\frac{1000 - 2 * 5.9}{20}$	$= 49.41$
$49.41 * 27$	$= 1334.07$
$5.9 * 1.025^{\frac{9}{360}} - 5.9$	$= 0.003643285$
$1334.07 + 0.003643285$	$= 1334.074$
$\frac{1334.074}{1000} - 1$	$= 0.3340736$

2.7 naivpredict

Time series forecasting using Holt exponential smoothing

Description

Time series forecasting using Holt exponential smoothing. In this model it is assumed that the given time series contains a trend, however, it shows no seasonal variations.

Usage

```
naivpredict(symbol = NA, datamatrix = list(NA, NA), period =
    c(FALSE, FALSE), nAhead = 5, plot = TRUE, stats = TRUE)
```

Arguments

`symbol`	a unique series of letters assigned to a security for trading purposes. You can get the symbols by using the function getsymbols or from http://finance.yahoo.com/. If FALSE, a datamatrix is needed.
`datamatrix`	a list of two vectors. The first one is a vector of dates, the second one is a vector of prices.
`period`	is a vector of start and end dates, in "YYYY-MM-DD" format. If the start date is FALSE, all available data until the end date will be used. If the end date is FALSE, all available data from the start date will be used. If both are FALSE, all available data will be used. Prediction starts from the end date.
`nAhead`	forecast horizon.
`plot`	logical; if TRUE, the series for the given period as well as prediction will be plotted.
`stats`	if stats=TRUE, statistics like `summary` in linear models as well as residual diagnostic plots will be shown.

Details

The function `naivpredict` is based on the function `HoltWinters` with `gamma=FALSE`.
For the optimal choice of smoothing factors alpha and beta, the method of minimizing the squared one-step prediction error is used.

Value

`pred`	predicted time series values.

References

Charles C. Holt (2007). Forecasting seasonals and trends by exponentially weighted moving averages. International Journal of Forecasting, 20(1), 7
Francis X. Diebold (2007). Elements of forecasting. Thomson Learning, London, 315, 316
Winfried Stier (2001). Methoden der Zeitreihenanalyse. Springer, Berlin [u.a.], 24

Author(s)

Waldemar Kemler, Peter Schaffner

Warning

To run the function, an active internet connection is required.

See Also

atminaiv, HoltWinters

Examples

```
#Prediction of price of "AACC" for the next 5 days.
naivpredict("AACC", nAhead=5)
```