Package 'TTR'

September 1, 2020

Type Package

Title Technical Trading Rules

Version 0.24.2

Author Joshua Ulrich

Maintainer Joshua Ulrich <josh.m.ulrich@gmail.com>

Imports xts (>= 0.10-0), zoo, curl

LinkingTo xts

Enhances quantmod

Suggests RUnit

Description A collection of over 50 technical indicators for creating technical trading rules. The package also provides fast implementations of common rolling-window functions, and several volatility calculations.

License GPL (>= 2)

URL https://github.com/joshuaulrich/TTR

BugReports https://github.com/joshuaulrich/TTR/issues

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-09-01 09:50:31 UTC

R topics documented:

`adjRatios` *Split and dividend adjustment ratios*

Description

Create split and dividend adjustment ratio vectors.

Usage

```
adjRatios(splits, dividends, close)
```

Arguments

splits	Split series that is coercible to xts.
dividends	Dividend series that is coercible to xts.
close	Close price series that is coercible to xts.

Details

- If only splits is provided, the resulting object will only have as many observations as splits.
- If splits and close are provided, the resulting object will have as many observations as max(NROW(splits),NROW(close)).
- close is required if dividends is provided.

Value

A xts object containing the columns:

Split The split adjustment ratio.

Div The dividend adjustment ratio.

Author(s)

Joshua Ulrich

ADX *Welles Wilder's Directional Movement Index*

Description

Directional Movement Index; developed by J. Welles Wilder.

Usage

```
ADX(HLC, n = 14, maType, ...)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices.
n	Number of periods to use for DX calculation (not ADX calculation).
maType	A function or a string naming the function to be called.
...	Other arguments to be passed to the maType function.

Details

The DIp/DIn (positive/negative) is the percentage of the true range that is up/down.

Value

A object of the same class as HLC or a matrix (if try.xts fails) containing the columns:

DIp The positive Direction Index.

DIn The negative Direction Index.

DX The Direction Index.

ADX The Average Direction Index (trend strength).

Note

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).

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/DI.htm
https://www.fmlabs.com/reference/DX.htm
https://www.fmlabs.com/reference/ADX.htm
https://www.fmlabs.com/reference/ADXR.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=49
https://www.linnsoft.com/techind/directional-indicator-diplus-diminus
https://www.linnsoft.com/techind/adx-avg-directional-movement
https://www.linnsoft.com/techind/adxr-avg-directional-movement-rating
https://school.stockcharts.com/doku.php?id=technical_indicators:average_directional_index_adx

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. The DX calculation uses ATR. See aroon, CCI, TDI, VHF, GMMA for other indicators that measure trend direction/strength.

Examples

```
data(ttrc)
dmi.adx <- ADX(ttrc[,c("High","Low","Close")])
```

aroon *Aroon*

Description

The Aroon indicator attempts to identify starting trends. The indicator consists of up and down lines, which measure how long it has been since the highest high/lowest low has occurred in the last n periods. Developed by Tushar Chande in 1995.

Usage

```
aroon(HL, n = 20)
```

Arguments

HL	Object that is coercible to xts or matrix and contains either a High-Low price series, or a Close price series.
n	Number of periods to use in the calculation.

Details

Aroon up (down) is the elapsed time, expressed as a percentage, between today and the highest (lowest) price in the last n periods. If today's price is a new high (low) Aroon up (down) will be 100. Each subsequent period without another new high (low) causes Aroon up (down) to decrease by (1 / n) x 100.

Value

A object of the same class as HL or a matrix (if try.xts fails) containing the columns:

aroonUp The Aroon up indicator.

aroonDn The Aroon down indicator.

oscillator The Aroon oscillator (aroonUp -aroonDn).

Note

If High-Low prices are given, the function calculates the max/min using the high/low prices. Otherwise the function calculates the max/min of the single series.

Up (down) trends are indicated when the aroonUp(Dn) is between 70 and 100. Strong trends are indicated when when the aroonUp(Dn) is above 70 while the aroonDn(Up) is below 30. Also, crossovers may be useful.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/Aroon.htm
https://www.fmlabs.com/reference/AroonOscillator.htm
https://www.linnsoft.com/techind/aroon-arn
https://school.stockcharts.com/doku.php?id=technical_indicators:aroon

See Also

See CCI, ADX, TDI, VHF, GMMA for other indicators that measure trend direction/strength.

Examples

```
## Get Data and Indicator ##
data(ttrc)
trend <- aroon( ttrc[,c("High", "Low")], n=20 )
```

ATR *True Range / Average True Range*

Description

True range (TR) is a measure of volatility of a High-Low-Close series; average true range (ATR) is a Welles Wilder's style moving average of the TR. Developed by J. Welles Wilder in 1978.

Usage

```
ATR(HLC, n = 14, maType, ...)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices.
n	Number of periods for moving average.
maType	A function or a string naming the function to be called.
...	Other arguments to be passed to the maType function.

Details

TR incorporates yesterday's close in the calculation (high minus low). E.g. if yesterday's close was higher than today's high, then the TR would equal yesterday's close minus today's low.

The ATR is a component of the Welles Wilder Directional Movement Index (DX, ADX).

Value

A object of the same class as HLC or a matrix (if try.xts fails) containing the columns:

tr The true range of the series.

atr The average (as specified by ma) true range of the series.

trueHigh The true high of the series.

trueLow The true low of the series.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/TR.htm
https://www.fmlabs.com/reference/ATR.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=35
https://www.linnsoft.com/techind/true-range-tr
https://school.stockcharts.com/doku.php?id=technical_indicators:average_true_range_atr

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. See DX, which uses true range. See chaikinVolatility for another volatility measure.

Examples

```
data(ttrc)
atr <- ATR(ttrc[,c("High","Low","Close")], n=14)
```

BBands *Bollinger Bands*

Description

Bollinger Bands are a way to compare a security's volatility and price levels over a period of time. Developed by John Bollinger.

Usage

```
BBands(HLC, n = 20, maType, sd = 2, ...)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices. If only a univariate series is given, it will be used. See details.
n	Number of periods for moving average.
maType	A function or a string naming the function to be called.
sd	The number of standard deviations to use.
...	Other arguments to be passed to the maType function.

Details

Bollinger Bands consist of three lines:

The middle band is generally a 20-period SMA of the typical price ([high + low + close]/3). The upper and lower bands are sd standard deviations (generally 2) above and below the MA.

The middle band is usually calculated using the typical price, but if a univariate series (e.g. Close, Weighted Close, Median Price, etc.) is provided, it will be used instead.

Value

A object of the same class as HLC or a matrix (if try.xts fails) containing the columns:

dn The lower Bollinger Band.

mavg The middle Moving Average (see notes).

up The upper Bollinger Band.

pctB The %B calculation.

Note

Using any moving average other than SMA will result in inconsistencies between the moving average calculation and the standard deviation calculation. Since, by definition, a rolling standard deviation uses a simple moving average.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/Bollinger.htm
https://www.fmlabs.com/reference/BollingerWidth.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=36
https://www.linnsoft.com/techind/bollinger-bands
https://school.stockcharts.com/doku.php?id=technical_indicators:bollinger_bands
https://school.stockcharts.com/doku.php?id=technical_indicators:bollinger_band_width

See Also

See EMA, SMA, etc. for moving average options; and note Warning section.

Examples

```
## The examples below show the differences between using a
## High-Low-Close series, and just a close series when
## calculating Bollinger Bands.
data(ttrc)
bbands.HLC <- BBands( ttrc[,c("High","Low","Close")] )
bbands.close <- BBands( ttrc[,"Close"] )
```

CCI *Commodity Channel Index*

Description

The Commodity Channel Index (CCI) attempts to identify starting and ending trends.

Usage

```
CCI(HLC, n = 20, maType, c = 0.015, ...)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices. If only a univariate series is given, it will be used. See details.
n	Number of periods for moving average.
maType	A function or a string naming the function to be called.
c	Constant to apply to the mean deviation.
...	Other arguments to be passed to the maType function.

Details

CCI relates the current price and the average of price over n periods. The CCI usually falls in a channel of -100 to 100. A basic CCI trading system is: Buy (sell) if CCI rises above 100 (falls below -100) and sell (buy) when it falls below 100 (rises above -100).

CCI is usually calculated using the typical price, but if a univariate series (e.g. Close, Weighted Close, Median Price, etc.) is provided, it will be used instead.

Value

A object of the same class as HLC or a vector (if try.xts fails) containing the CCI values.

Note

If HLC is a High-Low-Close matrix, then typical price will be calculated. If HLC is a vector, then those values will be used instead of the typical price.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/CCI.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=42
https://www.linnsoft.com/techind/cci-commodity-channel-index
https://school.stockcharts.com/doku.php?id=technical_indicators:commodity_channel_index_cci

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. See aroon, ADX, TDI, VHF, GMMA for other indicators that measure trend direction/strength.

Examples

```
data(ttrc)
cci <- CCI(ttrc[,c("High","Low","Close")])
```

---

chaikinAD *Chaikin Accumulation / Distribution*

Description

The Chaikin Accumulation / Distribution (AD) line is a measure of the money flowing into or out of a security. It is similar to On Balance Volume (OBV). Developed by Marc Chaikin.

Usage

```
chaikinAD(HLC, volume)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices.
volume	Vector or matrix of volume observations corresponding to the HLC object.

Details

The AD line is similar to OBV; the difference is that OBV sums volume multiplied by +/- 1 if the close is higher/lower than the previous close, while the AD line multiplies volume by the close location value (CLV).

Value

A object of the same class as `HLC` and `volume` or a vector (if `try.xts` fails) containing the accumulation / distribution values.

Note

The Accumulation/Distribution Line is interpreted by looking for a divergence in the direction of the indicator relative to price.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/AccumDist.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=27
https://www.linnsoft.com/techind/accumulation-distribution
https://school.stockcharts.com/doku.php?id=technical_indicators:accumulation_distribution_line

See Also

See `OBV`, and `CLV`.

Examples

```
data(ttrc)
ad <- chaikinAD(ttrc[,c("High","Low","Close")], ttrc[,"Volume"])
```

`chaikinVolatility` *Chaikin Volatility*

Description

Chaikin Volatility measures the rate of change of the security's trading range. Developed by Marc Chaikin.

Usage

```
chaikinVolatility(HL, n = 10, maType, ...)
```

Arguments

HL	Object that is coercible to xts or matrix and contains High-Low prices.
n	Number of periods for moving average.
maType	A function or a string naming the function to be called.
...	Other arguments to be passed to the maType function.

Details

The Chaikin Volatility indicator defines volatility as an increase in the difference between the high and low.

Value

A object of the same class as HL or a vector (if try.xts fails) containing the Chaikin Volatility values.

Note

A rapid increase in Chaikin Volatility indicates an approaching bottom. A slow decrease in Chaikin Volatility indicates an approaching top.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/ChaikinVolatility.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=120

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. See TR for another volatility measure.

Examples

```
data(ttrc)
volatility <- chaikinVolatility(ttrc[,c("High","Low")])
```

CLV *Close Location Value*

Description

The Close Location Value (CLV) relates the day's close to its trading range.

Usage

```
CLV(HLC)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices.

Details

The CLV will fall in a range of -1 to +1. If the CLV is +/-1, the close is at the high/low; if the CLV is 0, the close is directly between the high and low.

Value

A object of the same class as HLC or a vector (if try.xts fails) containing the Close Location Values of a High-Low-Close price series.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://school.stockcharts.com/doku.php?id=technical_indicators:accumulation_distribution_line

See Also

See chaikinAD, which uses CLV.

Examples

```
data(ttrc)
clv <- CLV(ttrc[,c("High","Low","Close")])
```

CMF *Chaikin Money Flow*

Description

Chaikin Money Flow compares total volume over the last n time periods to total volume times the Close Location Value (CLV) over the last n time periods. Developed by Marc Chaikin.

Usage

```
CMF(HLC, volume, n = 20)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices.
volume	Vector or matrix of volume observations corresponding to the HLC object.
n	Number of periods to use.

Details

Chaikin Money Flow is calculated by taking dividing the sum of the Chaikin Accumulation / Distribution line over the past n periods by the sum of volume over the past n periods.

Value

A object of the same class as HLC and volume or a vector (if try.xts fails) containing the Chaikin Money Flow values.

Note

When Chaikin Money Flow is above/below +/- 0.25 it is a bullish/bearish signal. If Chaikin Money Flow remains below zero while the price is rising, it indicates a probable reversal.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/ChaikinMoneyFlow.htm
https://www.linnsoft.com/techind/chaikin-money-flow-cmf
https://school.stockcharts.com/doku.php?id=technical_indicators:chaikin_money_flow_cmf

See Also

See CLV, and chaikinAD.

Examples

```
data(ttrc)
cmf <- CMF(ttrc[,c("High","Low","Close")], ttrc[,"Volume"])
```

CMO *Chande Momentum Oscillator*

Description

The Chande Momentum Oscillator (CMO) is a modified RSI. Developed by Tushar S. Chande.

Usage

```
CMO(x, n = 14)
```

Arguments

x	Price, volume, etc. series that is coercible to xts or matrix.
n	Number of periods to use.

Details

The CMO divides the total movement by the net movement ([up - down] / [up + down]), where RSI divides the upward movement by the net movement (up / [up + down]).

Value

A object of the same class as x or a vector (if try.xts fails) containing Chande Momentum Oscillator values.

Note

There are several ways to interpret the CMO:

1. Values over/under +/- 50 indicate overbought/oversold conditions.
2. High CMO values indicate strong trends.
3. When the CMO crosses above/below a moving average of the CMO, it is a buy/sell signal.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/CMO.htm

See Also

See RSI.

Examples

```
data(ttrc)
cmo <- CMO(ttrc[,"Close"])
```

CTI *Ehler's Correlation Trend Indicator*

Description

Ehler's Correlation Trend Indicator (CTI) measures the Spearman correlation of the price with the ideal trend line: a straight line with increasing slope.

Usage

```
CTI(price, n = 20, slope = 1)
```

Arguments

price	Price series that is coercible to xts or matrix.
n	Number of periods to use.
slope	Slope of desired trend.

Details

The CTI measures the Spearman correlation between the price and the ideal trend line with slope of slope, over the past n days.

See URL in references section for further details.

Value

A object of the same class as price or a matrix (if try.xts fails) with the column:

cti The Correlation Trend Indicator.

Note

Positive/negative CTI values signal positive/negative correlation with the desired trend line slope. A simple strategy could be long when the CTI is positive and, short when it is negative.

Author(s)

Ethan Smith, Joshua Ulrich

References

John Ehlers, Correlation Trend Indicator, Stocks & Commodities May-2020 The following site(s) were used to code/document this indicator:
https://financial-hacker.com/petra-on-programming-a-unique-trend-indicator/

See Also

See aroon, CCI, ADX, VHF, GMMA, TDI for other indicators that measure trend direction/strength.

Examples

```
data(ttrc)
cti <- CTI(ttrc[,"Close"], n = 20)
```

DonchianChannel *Donchian Channel*

Description

Donchian Channels were created by Richard Donchian and were used to generate buy and sell signals for the Turtle Trading system.

Usage

```
DonchianChannel(HL, n = 10, include.lag = FALSE)
```

Arguments

HL	Object that is coercible to xts or matrix and contains High-Low prices.
n	Number of periods for moving average.
include.lag	Should values be lagged so that today's prices are not included in the calculation? See Note.

Details

Donchian Channels consist of two (sometimes three) lines:

The top line is the highest high of the past `n` periods. The bottom line is the lowest low of the past `n` periods. The middle line is the average of the top and bottom lines.

Value

A object of the same class as `HL` or a matrix (if `try.xts` fails) containing the columns:

high The highest high series.

mid The average of `high` and `low`.

low The lowest low series.

Note

The default of `include.lag=FALSE` makes `DonchainChannel` consistent with other **TTR** functions, in that it includes the current period in the calculation.

The default is different than the original calculation, which would calculate the indicator using periods t-1 through t-n. Setting `include.lag=TRUE` will return the result of the original calculation.

The default of this argument may change in the future.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.linnsoft.com/techind/donchian-channels

See Also

See `BBands`.

Examples

```
data(ttrc)
dc <- DonchianChannel( ttrc[,c("High","Low")] )
```

DPO *De-Trended Price Oscillator*

Description

The Detrended Price Oscillator (DPO) removes the trend in prices - or other series - by subtracting a moving average of the price from the price.

Usage

```
DPO(x, n = 10, maType, shift = n/2 + 1, percent = FALSE, ...)
```

Arguments

x	Price, volume, etc. series that is coercible to xts or matrix.
n	Number of periods for moving average.
maType	A function or a string naming the function to be called.
shift	The number of periods to shift the moving average.
percent	logical; if TRUE, the percentage difference between the slow and fast moving averages is returned, otherwise the difference between the respective averages is returned.
...	Other arguments to be passed to the maType function.

Details

The Detrended Price shows cycles and overbought / oversold conditions.

Value

A object of the same class as x or a vector (if try.xts fails) containing the DPO values.

Warning

The detrended price oscillator removes the trend in the series by centering the moving average. Centering the moving average causes it to include future data. Therefore, even though this indicator looks like a classic oscillator, it should not be used for trading rule signals.

Note

DPO does not extend to the last date because it is based on a displaced moving average. The calculation shifts the results shift periods, so the last shift periods will be zero.
As stated above, the DPO can be used on any univariate series, not just price.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://school.stockcharts.com/doku.php?id=technical_indicators:detrended_price_osci

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. See MACD for a general oscillator.

Examples

```
data(ttrc)
priceDPO <- DPO(ttrc[,"Close"])
volumeDPO <- DPO(ttrc[,"Volume"])
```

DVI *DV Intermediate Oscillator*

Description

The DV Intermediate oscillator (DVI) is a very smooth momentum oscillator that can also be used as a trend indicator. Created by David Varadi.

Usage

```
DVI(
  price,
  n = 252,
  wts = c(0.8, 0.2),
  smooth = 3,
  magnitude = c(5, 100, 5),
  stretch = c(10, 100, 2),
  exact.multiplier = 1
)
```

Arguments

price	Price series that is coercible to xts or matrix.
n	Number of periods for the percent rank.
wts	The weight given to the smoothed returns (magnitude) component and the up/down days (stretch) component, respectively.
smooth	The number of periods to smooth price.

`magnitude` A set of 3 periods used to smooth magnitude.

`stretch` A set of 3 periods used to smooth stretch.

`exact.multiplier`
The weight applied to identical values in the window. See `runPercentRank`.

Details

The DVI combines smoothed returns over different time windows and the relative number of up versus down days (stretch) over different time windows.

Value

A object of the same class as `price` or a vector (if `try.xts` fails) containing the DVI values.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://cssanalytics.wordpress.com/2009/12/13/what-is-the-dvi/
https://marketsci.wordpress.com/2010/07/27/css-analytics%E2%80%99-dvi-indicator-revealed/

Examples

```
data(ttrc)
dvi <- DVI(ttrc[,"Close"])
```

EMV *Arms' Ease of Movement Value*

Description

Arms' Ease of Movement Value (EMV) emphasizes days where the security moves easily and minimizes days where the security does not move easily. Developed by Richard W. Arms, Jr.

Usage

```
EMV(HL, volume, n = 9, maType, vol.divisor = 10000, ...)
```

Arguments

HL	Object that is coercible to xts or matrix and contains High-Low prices.
volume	Vector or matrix of volume observations corresponding to the HL object.
n	Number of periods for moving average.
maType	A function or a string naming the function to be called.
vol.divisor	An increment to make the results larger and easier to work with.
...	Other arguments to be passed to the maType function.

Details

The EMV is calculated by dividing the midpoint ([high + low]/2) move by the 'Box Ratio' (volume divided by the high minus low).

Value

A object of the same class as HL and volume or a matrix (if try.xts fails) containing the columns:

emv The ease of movement values.

maEMV The smoothed (as specified by ma) ease of movement values.

Note

A buy/sell signal is generated when the EMV crosses above/below zero. When the EMV hovers around zero, there are small price movements and/or high volume, and the price is not moving easily.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/ArmsEMV.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=51
https://www.linnsoft.com/techind/arms-ease-movement

See Also

See EMA, SMA, etc. for moving average options; and note Warning section.

Examples

```
data(ttrc)
emv <- EMV(ttrc[,c("High","Low")], ttrc[,"Volume"])
```

GMMA *Guppy Multiple Moving Averages*

Description

Calculate the Guppy Multiple Moving Average of a series.

Usage

```
GMMA(
  x,
  short = c(3, 5, 8, 10, 12, 15),
  long = c(30, 35, 40, 45, 50, 60),
  maType
)
```

Arguments

x	Price, volume, etc. series that is coercible to xts or matrix.
short	Vector of short-term periods.
long	Vector of long-term periods.
maType	Either: 1. A function or a string naming the function to be called. 2. A *list* with the first component like (1) above, and additional parameters specified as *named* components. See Examples.

Details

The Guppy Multiple Moving Average signals a changing trend when the `short` and `long` groups of moving averages intersect. An up/down trend exists when the short/long-term moving averages are greater than the long/short-term averages.

Value

A object of the same class as `x` or `price` or a vector (if `try.xts` fails) containing the Guppy Multiple Moving Average.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.investopedia.com/terms/g/guppy-multiple-moving-average.asp

See Also

See aroon, CCI, ADX, VHF, TDI for other indicators that measure trend direction/strength.

Examples

```
data(ttrc)
gmma <- GMMA(ttrc[,"Close"])
```

KST *Know Sure Thing*

Description

The Know Sure Thing (KST) is a smooth, summed, rate of change indicator. Developed by Martin Pring.

Usage

```
KST(
  price,
  n = c(10, 10, 10, 15),
  nROC = c(10, 15, 20, 30),
  nSig = 9,
  maType,
  wts = 1:NROW(n),
  ...
)
```

Arguments

price	Price series that is coercible to xts or matrix.
n	A vector of the number of periods to use in the MA calculations.
nROC	A vector of the number of periods to use in the ROC calculations.
nSig	The number of periods to use for the KST signal line.
maType	Either: 1. A function or a string naming the function to be called. 2. A *list* with the first component like (1) above, and additional parameters specified as *named* components. See Examples.
wts	A vector the same length as n, of the weight for each period (need not sum to one).
...	Other arguments to be passed to the maType function in case (1) above.

Details

For each day (week, month, etc.), the KST calculates the ROC over several periods. Those ROCs are smoothed using the given moving averages, then multiplied by their respective weighting values. The resulting values are summed for each day (month, week, etc.).

Value

A object of the same class as `price` or a vector (if `try.xts` fails) containing the Know Sure Thing values.

Note

The KST indicates bullish/bearish momentum as it crosses above/below its moving average. Because the KST tends to lead price action, look for trend confirmation in the price.

The default arguments are for the daily KST. There is also the Long-Term KST, with arguments: `n=c(9,12,18,24)` - where the periods are months, not days - and the moving average periods are 6, 6, 6, and 9 months, respectively.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://web.archive.org/web/20110715112957/http://www.pring.com/movieweb/daily_kst.htm
https://web.archive.org/web/20100101162707/http://www.pring.com/movieweb/KST_MCM.htm

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. See ROC for the rate-of-change function. See MACD for a generic oscillator.

Examples

```
data(ttrc)
kst <- KST(ttrc[,"Close"])

kst4MA <- KST(ttrc[,"Close"],
  maType=list(list(SMA),list(EMA),list(DEMA),list(WMA)))
```

lags *Miscellaneous Tools*

Description

Various functions that may be useful in designing technical trading rules.

Usage

```
lags(x, n = 1)

growth(price, signals, ...)

naCheck(x, n = 0)
```

Arguments

x	Object that is coercible to xts or matrix.
n	Number of periods to use.
price	Price series that is coercible to xts or matrix.
signals	Signals to use (defaults to vector of ones). Use '0' for no position, '1' for long position, and '-1' for short position.
...	Further arguments to be passed from or to other methods.

Details

growth calculates the growth of an investment using given prices and signals.

lags calculates the lags of a given series.

Value

growth returns a vector of the growth of the investment.

lags returns a matrix of lagged values of the original vector.

Note

In growth you can specify the number of periods and type of compounding to use when calculating returns of the price series via the '...' argument.

Author(s)

Joshua Ulrich

MACD *MACD Oscillator*

Description

The MACD was developed by Gerald Appel and is probably the most popular price oscillator. The MACD function documented in this page compares a fast moving average (MA) of a series with a slow MA of the same series. It can be used as a generic oscillator for any univariate series, not only price.

Usage

```
MACD(x, nFast = 12, nSlow = 26, nSig = 9, maType, percent = TRUE, ...)
```

Arguments

x	Object that is coercible to xts or matrix; usually price, but can be volume, etc.
nFast	Number of periods for fast moving average.
nSlow	Number of periods for slow moving average.
nSig	Number of periods for signal moving average.
maType	Either: 1. A function or a string naming the function to be called. 2. A *list* with the first component like (1) above, and additional parameters specified as *named* components. See Examples.
percent	logical; if TRUE, the percentage difference between the fast and slow moving averages is returned, otherwise the difference between the respective averages is returned.
...	Other arguments to be passed to the maType function in case (1) above.

Details

The MACD function either subtracts the fast MA from the slow MA, or finds the rate of change between the fast MA and the slow MA.

Value

A object of the same class as x or a matrix (if try.xts fails) containing the columns:

macd The price (volume, etc.) oscillator.

signal The oscillator signal line (a moving average of the oscillator).

Note

The MACD is a special case of the general oscillator applied to price. The MACD can be used as a general oscillator applied to any series. Time periods for the MACD are often given as 26 and 12, but the original formula used exponential constants of 0.075 and 0.15, which are closer to 25.6667 and 12.3333 periods.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
Moving Average Convergence/Divergence (MACD):
https://www.fmlabs.com/reference/MACD.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=66
https://www.linnsoft.com/techind/macd
https://school.stockcharts.com/doku.php?id=technical_indicators:moving_average_convergence_divergence_macd

Price Oscillator:
https://www.fmlabs.com/reference/PriceOscillator.htm
https://www.fmlabs.com/reference/PriceOscillatorPct.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=94
https://school.stockcharts.com/doku.php?id=technical_indicators:price_oscillators_ppo

Volume Oscillator:
https://www.fmlabs.com/reference/PVO.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=122

See Also

See EMA, SMA, etc. for moving average options; and note Warning section.

Examples

```
data(ttrc)

macd  <- MACD( ttrc[,"Close"], 12, 26, 9, maType="EMA" )
macd2 <- MACD( ttrc[,"Close"], 12, 26, 9,
         maType=list(list(SMA), list(EMA, wilder=TRUE), list(SMA)) )
```

MFI *Money Flow Index*

Description

The MFI is a ratio of positive and negative money flow over time.

Usage

```
MFI(HLC, volume, n = 14)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices. If only a univariate series is given, it will be used. See details.
volume	Vector or matrix of volume observations corresponding to HLC object.
n	Number of periods to use.

Details

Money Flow (MF) is the product of price and volume. Positive/negative MF occur when today's price is higher/lower than yesterday's price. The MFI is calculated by dividing positive MF by negative MF for the past n periods. It is then scaled between 0 and 100.

MFI is usually calculated using the typical price, but if a univariate series (e.g. Close, Weighted Close, Median Price, etc.) is provided, it will be used instead.

Value

A object of the same class as HLC and volume or a vector (if try.xts fails) containing the MFI values.

Note

Divergence between MFI and price can be indicative of a reversal. In addition, values above/below 80/20 indicate market tops/bottoms.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/default.htm?url=MoneyFlowIndex.htm
https://www.linnsoft.com/techind/money-flow-index-mfi
https://school.stockcharts.com/doku.php?id=technical_indicators:money_flow_index_mfi

See Also

See OBV and CMF.

Examples

```
data(ttrc)
mfi <- MFI(ttrc[,c("High","Low","Close")], ttrc[,"Volume"])
```

OBV *On Balance Volume (OBV)*

Description

On Balance Volume (OBV) is a measure of the money flowing into or out of a security. It is similar to Chaikin Accumulation / Distribution.

Usage

```
OBV(price, volume)
```

Arguments

price	Price series that is coercible to xts or matrix.
volume	Volume series that is coercible to xts or matrix, that corresponds to price object.

Details

OBV is calculated by adding (subtracting) each day's volume to a running cumulative total when the security's price closes higher (lower).

Value

A object of the same class as price and volume or a vector (if try.xts fails) containing the OBV values.

Note

OBV is usually compared with the price chart of the underlying security to look for divergences/confirmation.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/OBV.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=82
https://www.linnsoft.com/techind/balance-open-interest
https://school.stockcharts.com/doku.php?id=technical_indicators:on_balance_volume_obv

See Also

See chaikinAD.

Examples

```
data(ttrc)
obv <- OBV(ttrc[,"Close"], ttrc[,"Volume"])
```

PBands *Construct (optionally further smoothed and centered) volatility bands around prices*

Description

John Bollinger's famous adaptive volatility bands most often use the typical price of an HLC series, or may be calculated on a univariate price series (see BBands).

Usage

```
PBands(
  prices,
  n = 20,
  maType = "SMA",
  sd = 2,
  ...,
  fastn = 2,
  centered = FALSE,
  lavg = FALSE
)
```

Arguments

prices	A univariate series of prices.
n	Number of periods to average over.
maType	A function or a string naming the function to be called.
sd	The number of standard deviations to use.
...	any other pass-thru parameters, usually for function named by maType.
fastn	Number of periods to use for smoothing higher-frequency 'noise'.
centered	Whether to center the bands around a series adjusted for high frequency noise, default FALSE.
lavg	Whether to use a longer (n*2) smoothing period for centering, default FALSE.

Details

This function applies a second moving average denoted by `fastn` to filter out higher-frequency noise, making the bands somewhat more stable to temporary fluctuations and spikes.

If `centered` is `TRUE`, the function also further smoothes and centers the bands around a centerline adjusted to remove this higher frequency noise. If `lavg` is also `TRUE`, the smoothing applied for the middle band (but not the volatility bands) is doubled to further smooth the price-response function.

If you have multiple different price series in `prices`, and want to use this function, call this functions using `lapply(prices,PBands,...)`.

Value

A object of the same class as `prices` or a matrix (if `try.xts` fails) containing the columns:

dn The lower price volatility Band.

center The smoothed centerline (see details).

up The upper price volatility Band.

Author(s)

Brian G. Peterson

See Also

BBands

Examples

```
data(ttrc)
pbands.close <- PBands( ttrc[,"Close"] )
```

ROC *Rate of Change / Momentum*

Description

Calculate the (rate of) change of a series over `n` periods.

Usage

```
ROC(x, n = 1, type = c("continuous", "discrete"), na.pad = TRUE)

momentum(x, n = 1, na.pad = TRUE)
```

Arguments

x	Price, volume, etc. series that is coercible to xts or matrix.
n	Number of periods to use.
type	Compounding type; either "continuous" (the default) or "discrete".
na.pad	Should periods prior to n be appended? Default is TRUE.

Details

The ROC indicator provides the percentage difference of a series over two observations, while the momentum indicator simply provides the difference.

Value

A object of the same class as x or a vector (if try.xts fails) containing the rate-of-change (or return) values for ROC or a vector containing the differenced price series for momentum.

Author(s)

Joshua Ulrich

Examples

```
data(ttrc)
roc <- ROC(ttrc[,"Close"])
mom <- momentum(ttrc[,"Close"])
```

---

rollSFM *Analysis of Running/Rolling/Moving Windows*

Description

Various functions to analyze data over a moving window of periods.

Usage

```
rollSFM(Ra, Rb, n = 60)
```

Arguments

Ra	Object coercible to xts or matrix, containing the excess return for an individual security
Rb	Object coercible to xts or matrix, containing the market / benchmark return
n	Number of periods to use in the window

Value

A object of the same class as Ra (and Rb?) or a vector (if try.xts fails).

rollSFM returns single-factor model parameters and R-squared over a n-period moving window.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator: https://en.wikipedia.org/wiki/Simple_linear_regression

---

RSI *Relative Strength Index*

Description

The Relative Strength Index (RSI) calculates a ratio of the recent upward price movements to the absolute price movement. Developed by J. Welles Wilder.

Usage

```
RSI(price, n = 14, maType, ...)
```

Arguments

price	Price series that is coercible to xts or matrix.
n	Number of periods for moving averages.
maType	Either: 1. A function or a string naming the function to be called. 2. A *list* with the first component like (1) above, and additional parameters specified as *named* components. See Examples.
...	Other arguments to be passed to the maType function in case (1) above.

Details

The RSI calculation is RSI = 100 -100 / (1 + RS), where RS is the smoothed ratio of 'average' gains over 'average' losses. The 'averages' aren't true averages, since they're divided by the value of n and not the number of periods in which there are gains/losses.

Value

A object of the same class as price or a vector (if try.xts fails) containing the RSI values.

Note

The RSI is usually interpreted as an overbought/oversold (over 70 / below 30) indicator. Divergence with price may also be useful. For example, if price is making new highs/lows, but RSI is not, it could indicate a reversal.

You can calculate a stochastic RSI by using the function stoch on RSI values.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
Relative Strength Index:
https://www.fmlabs.com/reference/RSI.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=100
https://www.linnsoft.com/techind/relative-strength-index-rsi
https://school.stockcharts.com/doku.php?id=technical_indicators:relative_strength_index_rsi

Stochastic RSI:
https://www.fmlabs.com/reference/StochRSI.htm
https://school.stockcharts.com/doku.php?id=technical_indicators:stochrsi

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. See CMO for a variation on RSI.

Examples

```
data(ttrc)
price <- ttrc[,"Close"]

# Default case
rsi <- RSI(price)

# Case of one 'maType' for both MAs
rsiMA1 <- RSI(price, n=14, maType="WMA", wts=ttrc[,"Volume"])

# Case of two different 'maType's for both MAs
rsiMA2 <- RSI(price, n=14, maType=list(maUp=list(EMA),maDown=list(WMA)))
```

`runPercentRank` *Percent Rank over a Moving Window*

Description

This function computes a running/rolling percentage rank.

Usage

```
runPercentRank(x, n = 260, cumulative = FALSE, exact.multiplier = 0.5)
```

Arguments

`x`	Object coercible to xts or matrix.
`n`	Number of periods to use in the window or, if `cumulative=TRUE`, the number of observations to use before the first result is returned. Must be between 1 and `nrow(x)`, inclusive.
`cumulative`	Logical, use from-inception calculation?
`exact.multiplier`	The weight applied to identical values in the window. Must be between 0 and 1, inclusive. See details.

Details

The computation for a percentage rank can vary depending on the weight given to values in the window identical to the value being ranked. This weight can be set using the `exact.multiplier` argument which defaults to 0.5.

Value

A object of percent ranks over a n-period moving window of the same class as `x` and `y` or a vector (if `try.xts` fails).

Note

It may be important to note that this computation is different from the one used in Microsoft Excel's PERCENTRANK formula. Excel's computation is rather strange and gives inconsistent results as it uses interpolation to rank values that are not found within the lookback window.

Author(s)

Charlie Friedemann

References

The following site(s) were used to code/document this indicator:
https://en.wikipedia.org/wiki/Percentile_rank

runSum *Analysis of Running/Rolling/Moving Windows*

Description

Various functions to analyze data over a moving window of periods.

Usage

```
runSum(x, n = 10, cumulative = FALSE)

runMin(x, n = 10, cumulative = FALSE)

runMax(x, n = 10, cumulative = FALSE)

runMean(x, n = 10, cumulative = FALSE)

runMedian(x, n = 10, non.unique = "mean", cumulative = FALSE)

runCov(x, y, n = 10, use = "all.obs", sample = TRUE, cumulative = FALSE)

runCor(x, y, n = 10, use = "all.obs", sample = TRUE, cumulative = FALSE)

runVar(x, y = NULL, n = 10, sample = TRUE, cumulative = FALSE)

runSD(x, n = 10, sample = TRUE, cumulative = FALSE)

runMAD(
  x,
  n = 10,
  center = NULL,
  stat = "median",
  constant = 1.4826,
  non.unique = "mean",
  cumulative = FALSE
)

wilderSum(x, n = 10)
```

Arguments

x	Object coercible to xts or matrix.
n	Number of periods to use in the window or, if `cumulative=TRUE`, the number of observations to use before the first result is returned. Must be between 1 and `nrow(x)`, inclusive.
cumulative	Logical, use from-inception calculation?

non.unique	One of 'mean', 'max', or 'min'; which compute their respective statistics for the two middle values of even-sized samples.
y	Object coercible to xts or matrix.
use	Only "all.obs" currently implemented.
sample	Logical, sample covariance if TRUE (denominator of n-1)
center	The values to use as the measure of central tendency, around which to calculate deviations. The default (NULL) uses the median.
stat	Statistic to calculate, one of 'median' or 'mean' (e.g. median absolute deviation or mean absolute deviation, respectively.)
constant	Scale factor applied to approximate the standard deviation.

Value

A object of the same class as x and y or a vector (if try.xts fails).

runSum returns sums over a n-period moving window.

runMin returns minimums over a n-period moving window.

runMax returns maximums over a n-period moving window.

runMean returns means over a n-period moving window.

runMedian returns medians over a n-period moving window.

runCov returns covariances over a n-period moving window.

runCor returns correlations over a n-period moving window.

runVar returns variances over a n-period moving window.

runSD returns standard deviations over a n-period moving window.

runMAD returns median/mean absolute deviations over a n-period moving window.

wilderSum retuns a Welles Wilder style weighted sum over a n-period moving window.

Author(s)

Joshua Ulrich

SAR *Parabolic Stop-and-Reverse*

Description

The Parabolic Stop-and-Reverse calculates a trailing stop. Developed by J. Welles Wilder.

Usage

```
SAR(HL, accel = c(0.02, 0.2))
```

Arguments

HL	Object that is coercible to xts or matrix and contains High-Low prices.
accel	accel[1]: Acceleration factor. accel[2]: Maximum acceleration factor.

Details

The calculation for the SAR is quite complex. See the URLs in the references section for calculation notes.

The SAR assumes that you are always in the market, and calculates the Stop And Reverse point when you would close a long position and open a short position or vice versa.

Value

A object of the same class as HL or a vector (if try.xts fails) containing the Parabolic Stop and Reverse values.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.linnsoft.com/techind/parabolic-sar-sar
https://www.fmlabs.com/reference/SAR.htm
https://school.stockcharts.com/doku.php?id=technical_indicators:parabolic_sar
https://www.metastock.com/Customer/Resources/TAAZ/?p=87

See Also

See ATR and ADX, which were also developed by Welles Wilder.

Examples

```
data(ttrc)
sar <- SAR(ttrc[,c("High","Low")])
```

SMA *Moving Averages*

Description

Calculate various moving averages (MA) of a series.

Usage

```
SMA(x, n = 10, ...)

EMA(x, n = 10, wilder = FALSE, ratio = NULL, ...)

DEMA(x, n = 10, v = 1, wilder = FALSE, ratio = NULL)

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

EVWMA(price, volume, n = 10, ...)

ZLEMA(x, n = 10, ratio = NULL, ...)

VWAP(price, volume, n = 10, ...)

VMA(x, w, ratio = 1, ...)

HMA(x, n = 20, ...)

ALMA(x, n = 9, offset = 0.85, sigma = 6, ...)
```

Arguments

x	Price, volume, etc. series that is coercible to xts or matrix.
n	Number of periods to average over. Must be between 1 and nrow(x), inclusive.
...	any other passthrough parameters
wilder	logical; if TRUE, a Welles Wilder type EMA will be calculated; see notes.
ratio	A smoothing/decay ratio. ratio overrides wilder in EMA, and provides additional smoothing in VMA.
v	The 'volume factor' (a number in [0,1]). See Notes.
wts	Vector of weights. Length of wts vector must equal the length of x, or n (the default).
price	Price series that is coercible to xts or matrix.
volume	Volume series that is coercible to xts or matrix, that corresponds to price series, or a constant. See Notes.
w	Vector of weights (in [0,1]) the same length as x.
offset	Percentile at which the center of the distribution should occur.
sigma	Standard deviation of the distribution.

Details

SMA calculates the arithmetic mean of the series over the past n observations.

EMA calculates an exponentially-weighted mean, giving more weight to recent observations. See Warning section below.

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.

DEMA is calculated as: DEMA = (1 + v) * EMA(x,n) -EMA(EMA(x,n),n) * v (with the corresponding wilder and ratio arguments).

EVWMA uses volume to define the period of the MA.

ZLEMA is similar to an EMA, as it gives more weight to recent observations, but attempts to remove lag by subtracting data prior to (n-1)/2 periods (default) to minimize the cumulative effect.

VWMA and VWAP calculate the volume-weighted moving average price.

VMA calculate a variable-length moving average based on the absolute value of w. Higher (lower) values of w will cause VMA to react faster (slower).

HMA a WMA of the difference of two other WMAs, making it very reponsive.

ALMA inspired by Gaussian filters. Tends to put less weight on most recent observations, reducing tendency to overshoot.

Value

A object of the same class as x or price or a vector (if try.xts fails) containing the columns:

SMA Simple moving average.

EMA Exponential moving average.

WMA Weighted moving average.

DEMA Double-exponential moving average.

EVWMA Elastic, volume-weighted moving average.

ZLEMA Zero lag exponential moving average.

VWMA Volume-weighed moving average (same as VWAP).

VWAP Volume-weighed average price (same as VWMA).

VWA Variable-length moving average.

HMA Hull moving average.

ALMA Arnaud Legoux moving average.

Warning

Some indicators (e.g. EMA, DEMA, EVWMA, etc.) are calculated using the indicators' own previous values, and are therefore unstable in the short-term. As the indicator receives more data, its output becomes more stable. See example below.

Note

For EMA, wilder=FALSE (the default) uses an exponential smoothing ratio of 2/(n+1), while wilder=TRUE uses Welles Wilder's exponential smoothing ratio of 1/n. The EMA result is initialized with the n-period sample average at period n. The exponential decay is applied from that point forward.

Since WMA can accept a weight vector of length equal to the length of x or of length n, it can be used as a regular weighted moving average (in the case wts=1:n) or as a moving average weighted by volume, another indicator, etc.

Since DEMA allows adjusting v, it is technically Tim Tillson's generalized DEMA (GD). When v=1 (the default), the result is the standard DEMA. When v=0, the result is a regular EMA. All other values of v return the GD result. This function can be used to calculate Tillson's T3 indicator (see example below). Thanks to John Gavin for suggesting the generalization.

For EVWMA, if volume is a series, n should be chosen so the sum of the volume for n periods approximates the total number of outstanding shares for the security being averaged. If volume is a constant, it should represent the total number of outstanding shares for the security being averaged.

Author(s)

Joshua Ulrich, Ivan Popivanov (HMA, ALMA)

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/ExpMA.htm
https://www.fmlabs.com/reference/WeightedMA.htm
https://www.fmlabs.com/reference/DEMA.htm
https://www.fmlabs.com/reference/T3.htm
https://www.linnsoft.com/techind/evwma-elastic-volume-weighted-moving-average
https://www.fmlabs.com/reference/ZeroLagExpMA.htm
https://www.fmlabs.com/reference/VIDYA.htm
https://www.traderslog.com/hullmovingaverage
https://web.archive.org/web/20180222085959/http://arnaudlegoux.com/

See Also

See wilderSum, which is used in calculating a Welles Wilder type MA.

Examples

```
data(ttrc)
ema.20 <-   EMA(ttrc[,"Close"], 20)
sma.20 <-   SMA(ttrc[,"Close"], 20)
dema.20 <-  DEMA(ttrc[,"Close"], 20)
evwma.20 <- EVWMA(ttrc[,"Close"], ttrc[,"Volume"], 20)
zlema.20 <- ZLEMA(ttrc[,"Close"], 20)
alma <- ALMA(ttrc[,"Close"])
hma <- HMA(ttrc[,"Close"])
```

```
## Example of Tim Tillson's T3 indicator
T3 <- function(x, n=10, v=1) DEMA(DEMA(DEMA(x,n,v),n,v),n,v)
t3 <- T3(ttrc[,"Close"])

## Example of short-term instability of EMA
## (and other indicators mentioned above)
x <- rnorm(100)
tail( EMA(x[90:100],10), 1 )
tail( EMA(x[70:100],10), 1 )
tail( EMA(x[50:100],10), 1 )
tail( EMA(x[30:100],10), 1 )
tail( EMA(x[10:100],10), 1 )
tail( EMA(x[ 1:100],10), 1 )
```

SNR *Signal to Noise Ratio*

Description

The n-day SNR for a given market is calculated by taking the absolute price change over an n-day period and dividing it by the average n-day volatility.

Usage

```
SNR(HLC, n, ...)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices.
n	Number of periods for moving average.
...	Other arguments to be passed to ATR.

Details

$$SNR_n = \frac{|C_t - C_{t-n}|}{ATR_n}$$

Using average true range as the volatility measure captures more of the intraday and overnight volatility in a way that a measurement of Close-to-Close price change does not.

The interpretation is then relatively intuitive: an SNR value of five indicates that the market has moved five times the volatility (average true range) over the given look-back period.

Value

A object of the same class as HLC or a matrix (if try.xts fails) containing the signal to noise ratio.

Author(s)

Peter Carl

References

Skeggs, James and Hill, Alex (2015). Back in Black Part 2: The Opportunity Set for Trend Following.

`stoch` *Stochastic Oscillator / Stochastic Momentum Index*

Description

The stochastic oscillator is a momentum indicator that relates the location of each day's close relative to the high/low range over the past n periods. Developed by George C. Lane in the late 1950s. The SMI relates the close to the midpoint of the high/low range. Developed by William Blau in 1993.

Usage

```
stoch(
  HLC,
  nFastK = 14,
  nFastD = 3,
  nSlowD = 3,
  maType,
  bounded = TRUE,
  smooth = 1,
  ...
)

SMI(HLC, n = 13, nFast = 2, nSlow = 25, nSig = 9, maType, bounded = TRUE, ...)
```

Arguments

`HLC`	Object that is coercible to xts or matrix and contains High-Low-Close prices. If only a univariate series is given, it will be used. See details.
`nFastK`	Number of periods for fast %K (i.e. the number of past periods to use).
`nFastD`	Number of periods for fast %D (i.e. the number smoothing periods to apply to fast %K).
`nSlowD`	Number of periods for slow %D (i.e. the number smoothing periods to apply to fast %D).
`maType`	Either: 1. A function or a string naming the function to be called. 2. A *list* with the first component like (1) above, and additional parameters specified as *named* components. See Examples.

bounded	Logical, should current period's values be used in the calculation?
smooth	Number of internal smoothing periods to be applied before calculating FastK. See Details.
...	Other arguments to be passed to the maType function in case (1) above.
n	Number of periods to use.
nFast	Number of periods for initial smoothing.
nSlow	Number of periods for double smoothing.
nSig	Number of periods for signal line.

Details

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 - see examples).

The smooth argument is the number of periods of internal smoothing to apply to the differences in the high-low-close range before calculating Fast K. Thanks to Stanley Neo for the suggestion.

Value

A object of the same class as HLC or a matrix (if try.xts fails) containing the columns:

fastK Stochastic Fast %K

fastD Stochastic Fast %D

slowD Stochastic Slow %D

SMI Stochastic Momentum Index

signal Stochastic Momentum Index signal line

Note

The calculation for William's %R is similar to that of stochastics' fast %K.

The value for fast %K will be 0.5 whenever the highest high and lowest low are the same over the last n periods.

The stochastic oscillator and SMI calculate relative value of the close versus the high/low range and the midpoint of the high/low range, respectively.

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.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document these indicators:
Stochastic Oscillator:
https://www.fmlabs.com/reference/StochasticOscillator.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=106
https://www.linnsoft.com/techind/stochastics
https://school.stockcharts.com/doku.php?id=technical_indicators:stochastic_oscillator_fast_slow_and_full

SMI:
https://www.fmlabs.com/reference/default.htm?url=SMI.htm

See Also

See EMA, SMA, etc. for moving average options; and note Warning section. See WPR to compare it's results to fast %K.

Examples

```
data(ttrc)
stochOSC <- stoch(ttrc[,c("High","Low","Close")])
stochWPR <- WPR(ttrc[,c("High","Low","Close")])

plot(tail(stochOSC[,"fastK"], 100), type="l",
    main="Fast %K and Williams %R", ylab="",
    ylim=range(cbind(stochOSC, stochWPR), na.rm=TRUE) )
lines(tail(stochWPR, 100), col="blue")
lines(tail(1-stochWPR, 100), col="red", lty="dashed")

stoch2MA <- stoch( ttrc[,c("High","Low","Close")],
    maType=list(list(SMA), list(EMA, wilder=TRUE), list(SMA)) )

SMI3MA <- SMI(ttrc[,c("High","Low","Close")],
    maType=list(list(SMA), list(EMA, wilder=TRUE), list(SMA)) )

stochRSI <- stoch( RSI(ttrc[,"Close"]) )
```

`stockSymbols` *Fetch Internet Data*

Description

Get investment data from the internet.

Usage

```
stockSymbols(
  exchange = c("AMEX", "NASDAQ", "NYSE", "ARCA", "BATS", "IEX"),
  sort.by = c("Exchange", "Symbol"),
  quiet = FALSE
)

getYahooData(
  symbol,
  start,
  end,
  freq = "daily",
  type = "price",
  adjust = TRUE,
  quiet = FALSE
)
```

Arguments

exchange	Character vector of exchange names on which desired instrument symbols are traded.
sort.by	Character vector of columns by which returned data will be sorted. Must be one or more of "Name", "Symbol", "Market.Cap", or "Exchange".
quiet	Logical; if TRUE, status messages will be printed to the console.
symbol	Yahoo! Finance instrument symbol.
start	Numeric; first date of desired data, in YYYYMMDD format. Default is first date of series.
end	Numeric; last date of desired data, in YYYYMMDD format. Default is last date of series.
freq	Desired data frequency. One of "daily", "weekly", "monthly".
type	Type of data to return. One of "price", or "split". type="split" will return both split and dividend data.
adjust	Logical; if TRUE, the Open, High, Low, and Close prices will be adjusted for dividends and splits, and Volume will be adjusted for dividends.

Details

getYahooData fetches individual stock data from the Yahoo! Finance website. It also adjusts price for splits and dividends, and volume for splits. See the Warning section, and note that it is deprecated in favor of getSymbols in the quantmod package.

stockSymbols fetches instrument symbols from the nasdaq.com website, and adjusts the symbols to be compatible with the Yahoo! Finance website.

Value

`getYahooData` returns an xts object containing the columns:

`stockSymbols` returns a character vector containing all the listed symbols for the given exchanges.

Date Trade date, in CCYYMMDD format.

Open Open price.

High High price.

Low Low price.

Close Close price.

Volume Volume.

Warning

As of TTR 0.23-2, `getYahooData` has been patched to work with changes to Yahoo Finance, which also included the following changes to the raw data:

- The adjusted close column appears to no longer include dividend adjustments
- The open, high, and low columns are adjusted for splits, and
- The raw data may contain missing values.
- The raw data may contain errors.

Note

The symbols returned by `stockSymbols` may not be in the format necessary to retrieve data using `getYahooData`.

`getYahooData` has only been tested on daily data. It isn't known if the function correctly adjusts data for any other frequency.

Author(s)

Joshua Ulrich

References

- Quant StackExchange: Download list of all stock symbols?
- CQS symbol convention
- Yahoo Finance symbol conventions

Examples

```
### Note: you must have a working internet
### connection for these examples to work!
if (interactive()) {
  ge <- getYahooData("GE", 19990404, 20050607, adjust = FALSE)

  nyse.symbols <- stockSymbols("NYSE")
```

```
}
```

TDI *Trend Detection Index*

Description

The Trend Detection Index (TDI) attempts to identify starting and ending trends. Developed by M. H. Pee.

Usage

```
TDI(price, n = 20, multiple = 2)
```

Arguments

price	Price series that is coercible to xts or matrix.
n	Number of periods to use.
multiple	Multiple used to calculate (2).

Details

The TDI is the (1) absolute value of the n-day sum of the n-day momentum, minus the quantity of (2) multiple*n-day sum of the absolute value of the n-day momentum, minus (3) n-day sum of the absolute value of the n-day momentum.

I.e. TDI = (1) - [(2) - (3)]

The direction indicator is the sum of the n-day momentum over the last n days.

See URL in references section for further details.

Value

A object of the same class as price or a matrix (if try.xts fails) containing the columns:

tdi The Trend Detection Index.

di The Direction Indicator.

Note

Positive/negative TDI values signal a trend/consolidation. A positive/ negative direction indicator signals a up/down trend. I.e. buy if the TDI and the direction indicator are positive, and sell if the TDI is positive while the direction indicator is negative.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.linnsoft.com/techind/trend-detection-index-tdi

See Also

See aroon, CCI, ADX, VHF, GMMA for other indicators that measure trend direction/strength.

Examples

```
data(ttrc)
tdi <- TDI(ttrc[,"Close"], n=30)
```

TRIX *Triple Smoothed Exponential Oscillator*

Description

The TRIX indicator calculates the rate of change of a triple exponential moving average. Developed by Jack K. Hutson.

Usage

```
TRIX(price, n = 20, nSig = 9, maType, percent = TRUE, ...)
```

Arguments

price	Price series that is coercible to xts or matrix.
n	Number of periods for moving average.
nSig	Number of periods for signal line moving average.
maType	Either: 1. A function or a string naming the function to be called. 2. A *list* with the first component like (1) above, and additional parameters specified as *named* components. See Examples.
percent	logical; if TRUE, the rate of change is calculated using the ROC function, otherwise the momentum function is used.
...	Other arguments to be passed to the maType function in case (1) above.

Details

The TRIX is calculated as follows:
3MA = MA(MA(MA(price)))
trix = 100 * [3MA(t) / 3MA(t-1) - 1]

Value

A object of the same class as `price` or a vector (if `try.xts` fails) containing the TRIX values.

Note

Buy/sell signals are generated when the TRIX crosses above/below zero. A nine-period EMA of the TRIX is used as a default signal line. Buy/sell signals are generated when the TRIX crosses above/below the signal line and is also above/below zero.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/default.htm?url=TRIX.htm
https://www.metastock.com/Customer/Resources/TAAZ/?p=114
https://www.linnsoft.com/techind/trix-triple-smoothed-exponential-oscillator
https://school.stockcharts.com/doku.php?id=technical_indicators:trix

See Also

See EMA, SMA, etc. for moving average options; and note Warning section.

Examples

```
data(ttrc)
trix  <- TRIX(ttrc[,"Close"])
trix4 <- TRIX(ttrc[,"Close"],
maType=list(list(SMA), list(EMA, wilder=TRUE), list(SMA), list(DEMA)))
```

TTR *Functions to create Technical Trading Rules (TTR)*

Description

This package contains many of the most popular technical analysis functions, as well as functions to retrieve U.S. stock symbols, and data from Yahoo Finance.

Details

Users will probably be most interested in the following functions:
ADX
BBands
changes
MovingAverages
MACD
RSI
runFun
stoch
VWAP
WebData

Author(s)

Joshua Ulrich

Maintainer: Joshua Ulrich

References

The following sites were used to code/document this package:
https://www.fmlabs.com/reference/default.htm
https://www.metastock.com/Customer/Resources/TAAZ/
https://www.linnsoft.com/indicators
https://school.stockcharts.com/doku.php?id=technical_indicators

Examples

```
data(ttrc)

# Bollinger Bands
bbands <- BBands( ttrc[,c("High","Low","Close")] )

# Directional Movement Index
adx <- ADX(ttrc[,c("High","Low","Close")])

# Moving Averages
ema <- EMA(ttrc[,"Close"], n=20)
sma <- SMA(ttrc[,"Close"], n=20)

# MACD
macd <- MACD( ttrc[,"Close"] )

# RSI
rsi <- RSI(ttrc[,"Close"])

# Stochastics
```

```
stochOsc <- stoch(ttrc[,c("High","Low","Close")])

### Note: you must have a working internet connection
### for the examples below to work!
if (interactive()) {
  # Fetch U.S. symbols from the internet
  nyseSymbols <- stockSymbols("NYSE")

  # Fetch Yahoo! Finance data from the internet
  ge <- getYahooData("GE", 19990404, 20050607, adjust = FALSE)
}
```

---

ttrc *Technical Trading Rule Composite data*

Description

Historical Open, High, Low, Close, and Volume data for the periods January 2, 1985 to December 31, 2006. Randomly generated.

Format

The format is:

Date:	Class 'Date'	5480 5481 5482 5485 5486 ...
Open:	num	3.18 3.09 3.11 3.09 3.10 ...
High:	num	3.18 3.15 3.12 3.12 3.12 ...
Low:	num	3.08 3.09 3.08 3.07 3.08 ...
Close:	num	3.08 3.11 3.09 3.10 3.11 ...
Volume:	num	1870906 3099506 2274157 2086758 2166348 ...

Details

These data do not represent an actual security. They are provided so examples do not necessitate an internet connection.

Source

Randomly generated.

Examples

```
data(ttrc)
plot(tail(ttrc[,"Close"],100), type="l")
```

`ultimateOscillator` *The Ultimate Oscillator*

Description

The Ultimate Oscillator is a momentum oscillator designed to capture momentum across three different time frames.

Usage

```
ultimateOscillator(HLC, n = c(7, 14, 28), wts = c(4, 2, 1))
```

Arguments

`HLC`	Object that is coercible to xts or matrix and contains High-Low-Close prices.
`n`	A vector of the number of periods to use for each average calculation.
`wts`	The weights applied to each average.

Details

Created by Larry Williams in 1976.

Author(s)

Ivan Popivanov

References

The following site(s) were used to code/document this indicator:
https://school.stockcharts.com/doku.php?id=technical_indicators:ultimate_oscillator

Examples

```
data(ttrc)
ult.osc <- ultimateOscillator(ttrc[,c("High","Low","Close")])
```

VHF *Vertical Horizontal Filter*

Description

The Vertical Horizontal Filter (VHF) attempts to identify starting and ending trends. Developed by Adam White.

Usage

```
VHF(price, n = 28)
```

Arguments

price	Object that is coercible to xts or matrix and contains a Close price series, or a High-Low-Close price series.
n	Number of periods to use.

Details

The VHF is calculated by subtracting the n-period lowest low from the n-period highest high and dividing that result by the n-period rolling sum of the close price changes.

Value

A object of the same class as price or a vector (if try.xts fails) containing the VHF values.

Note

If Close prices are given, the function calculates the max/min using only those prices (the default). If HLC prices are given, the function calculates the max/min using the high/low prices (added for flexibility).

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.metastock.com/Customer/Resources/TAAZ/#119

See Also

See aroon, CCI, ADX, TDI, GMMA for other indicators that measure trend direction/strength.

Examples

```
data(ttrc)
vhf.close <- VHF(ttrc[,"Close"])
vhf.hilow <- VHF(ttrc[,c("High","Low","Close")])
```

`volatility` *Volatility*

Description

Selected volatility estimators/indicators; various authors.

Usage

```
volatility(OHLC, n = 10, calc = "close", N = 260, mean0 = FALSE, ...)
```

Arguments

`OHLC`	Object that is coercible to xts or matrix and contains Open-High-Low-Close prices (or only Close prices, if `calc="close"`).
`n`	Number of periods for the volatility estimate.
`calc`	The calculation (type) of estimator to use.
`N`	Number of periods per year.
`mean0`	Use a mean of 0 rather than the sample mean.
`...`	Arguments to be passed to/from other methods.

Details

- Close-to-Close Volatility (`calc="close"`)

$$\sigma_{cl} = \sqrt{\frac{N}{n-2}\sum_{i=1}^{n-1}(r_i - \bar{r})^2}$$

$$where \;\; r_i = \log\left(\frac{C_i}{C_{i-1}}\right)$$

$$and \;\; \bar{r} = \frac{r_1 + r_2 + \ldots + r_{n-1}}{n-1}$$

- OHLC Volatility: Garman and Klass (`calc="garman.klass"`)
  The Garman and Klass estimator for estimating historical volatility assumes Brownian motion with zero drift and no opening jumps (i.e. the opening = close of the previous period). This estimator is 7.4 times more efficient than the close-to-close estimator.

$$\sigma = \sqrt{\frac{N}{n}\sum\left[\frac{1}{2}\left(\log\frac{H_i}{L_i}\right)^2 - (2\log 2 - 1)\left(\log\frac{C_i}{O_i}\right)^2\right]}$$

- High-Low Volatility: Parkinson (`calc="parkinson"`)
  The Parkinson formula for estimating the historical volatility of an underlying based on high and low prices.

$$\sigma = \sqrt{\frac{N}{4n \times \log 2} \sum_{i=1}^{n} \left(\log \frac{H_i}{L_i}\right)^2}$$

- OHLC Volatility: Rogers and Satchell (`calc="rogers.satchell"`)
  The Roger and Satchell historical volatility estimator allows for non-zero drift, but assumed no opening jump.

$$\sigma = \sqrt{\frac{N}{n} \sum \left[\log \frac{H_i}{C_i} \times \log \frac{H_i}{O_i} + \log \frac{L_i}{C_i} \times \log \frac{L_i}{O_i}\right]}$$

- OHLC Volatility: Garman and Klass - Yang and Zhang (`calc="gk.yz"`)
  This estimator is a modified version of the Garman and Klass estimator that allows for opening gaps.

$$\sigma = \sqrt{\frac{N}{n} \sum \left[\left(\log \frac{O_i}{C_{i-1}}\right)^2 + \frac{1}{2}\left(\log \frac{H_i}{L_i}\right)^2 - (2 \times \log 2 - 1)\left(\log \frac{C_i}{O_i}\right)^2\right]}$$

- OHLC Volatility: Yang and Zhang (`calc="yang.zhang"`)
  The Yang and Zhang historical volatility estimator has minimum estimation error, and is independent of drift and opening gaps. It can be interpreted as a weighted average of the Rogers and Satchell estimator, the close-open volatility, and the open-close volatility.

  Users may override the default values of α (1.34 by default) or k used in the calculation by specifying `alpha` or `k` in `...`, respectively. Specifying `k` will cause `alpha` to be ignored, if both are provided.

$$\sigma^2 = \sigma_o^2 + k\sigma_c^2 + (1-k)\sigma_{rs}^2$$

$$\sigma_o^2 = \frac{N}{n-1} \sum \left(\log \frac{O_i}{C_{i-1}} - \mu_o\right)^2$$

$$\mu_o = \frac{1}{n} \sum \log \frac{O_i}{C_{i-1}}$$

$$\sigma_c^2 = \frac{N}{n-1} \sum \left(\log \frac{C_i}{O_i} - \mu_c\right)^2$$

$$\mu_c = \frac{1}{n} \sum \log \frac{C_i}{O_i}$$

$$\sigma_{rs}^2 = \frac{N}{n} \sum \left(\log \frac{H_i}{C_i} \times \log \frac{H_i}{O_i} + \log \frac{L_i}{C_i} \times \log \frac{L_i}{O_i}\right)$$

$$k = \frac{\alpha - 1}{alpha + \frac{n+1}{n-1}}$$

Value

A object of the same class as `OHLC` or a vector (if `try.xts` fails) containing the chosen volatility estimator values.

Author(s)

Joshua Ulrich

References

The following sites were used to code/document these indicators. All were created by Thijs van den Berg under the GNU Free Documentation License and were retrieved on 2008-04-20. The original links are dead, but can be accessed via internet archives.

Close-to-Close Volatility (`calc="close"`):
https://web.archive.org/web/20100421083157/http://www.sitmo.com/eq/172

OHLC Volatility: Garman Klass (`calc="garman.klass"`):
https://web.archive.org/web/20100326172550/http://www.sitmo.com/eq/402

High-Low Volatility: Parkinson (`calc="parkinson"`):
https://web.archive.org/web/20100328195855/http://www.sitmo.com/eq/173

OHLC Volatility: Rogers Satchell (`calc="rogers.satchell"`):
https://web.archive.org/web/20091002233833/http://www.sitmo.com/eq/414

OHLC Volatility: Garman Klass - Yang Zhang (`calc="gk.yz"`):
https://web.archive.org/web/20100326215050/http://www.sitmo.com/eq/409

OHLC Volatility: Yang Zhang (`calc="yang.zhang"`):
https://web.archive.org/web/20100326215050/http://www.sitmo.com/eq/409

See Also

See `TR` and `chaikinVolatility` for other volatility measures.

Examples

```
data(ttrc)
ohlc <- ttrc[,c("Open","High","Low","Close")]
vClose <- volatility(ohlc, calc="close")
vClose0 <- volatility(ohlc, calc="close", mean0=TRUE)
vGK <- volatility(ohlc, calc="garman")
vParkinson <- volatility(ohlc, calc="parkinson")
vRS <- volatility(ohlc, calc="rogers")
```

`williamsAD` *Williams Accumulation / Distribution*

Description

The Williams Accumulation / Distribution (AD) line is a measure of market momentum. Developed by Larry Williams.

Usage

```
williamsAD(HLC)
```

Arguments

`HLC`	Object that is coercible to xts or matrix and contains High-Low-Close prices.

Details

The Williams AD line differs from OBV and chaikinAD in that it doesn't take volume into account.

Value

A object of the same class as `HLC` or a vector (if `try.xts` fails) containing the accumulation / distribution values.

Note

The Accumulation/Distribution Line is interpreted by looking for a divergence in the direction of the indicator relative to price.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/WilliamsAD.htm
https://www.metastock.com/Customer/Resources/TAAZ/#125

See Also

See `OBV`, `chaikinAD`, and `ATR`.

Examples

```
data(ttrc)
ad <- williamsAD(ttrc[,c("High","Low","Close")])
```

WPR *William's %R*

Description

William's % R.

Usage

```
WPR(HLC, n = 14)
```

Arguments

HLC	Object that is coercible to xts or matrix and contains High-Low-Close prices. If only a univariate series is given, it will be used. See details.
n	Number of periods to use.

Details

If an 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.

Value

A object of the same class as HLC or a vector (if try.xts fails) containing the William's %R values.

Note

The William's %R calculation is similar to stochastics' fast %K.

The value for William's %R will be 0.5 whenever the highest high and lowest low are the same over the last n periods.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/WilliamsR.htm
https://www.metastock.com/Customer/Resources/TAAZ/#126
https://www.linnsoft.com/techind/williams-r-wpr
https://school.stockcharts.com/doku.php?id=technical_indicators:williams_r

See Also

See stoch.

Examples

```
data(ttrc)
stochOsc <- stoch(ttrc[,c("High","Low","Close")])
stochWPR<- WPR(ttrc[,c("High","Low","Close")])

plot(tail(stochOsc[,"fastK"], 100), type="l",
    main="Fast %K and Williams %R", ylab="",
    ylim=range(cbind(stochOsc, stochWPR), na.rm=TRUE) )
lines(tail(stochWPR, 100), col="blue")
lines(tail(1-stochWPR, 100), col="red", lty="dashed")
```

ZigZag *Zig Zag*

Description

Zig Zag higlights trends by removing price changes smaller than change and interpolating lines between the extreme points.

Usage

```
ZigZag(HL, change = 10, percent = TRUE, retrace = FALSE, lastExtreme = TRUE)
```

Arguments

HL	Object that is coercible to xts or matrix and contains either a High-Low price series, or a Close price series.
change	Minimum price movement, either in dollars or percent (see percent).
percent	Use percentage or dollar change?
retrace	Is change a retracement of the previous move, or an absolute change from peak to trough?
lastExtreme	If the extreme price is the same over multiple periods, should the extreme price be the first or last observation?

Details

The Zig Zag is non-predictive. The purpose of the Zig Zag is filter noise and make chart patterns clearer. It's more a visual tool than an indicator.

Value

A object of the same class as `HL` or a vector (if `try.xts` fails) containing the Zig Zag indicator.

Warning

The last value of the ZigZag indicator is unstable (i.e. unknown) until the turning point actually occurs. Therefore this indicator isn't well-suited for use for systematic trading strategies.

Note

If High-Low prices are given, the function calculates the max/min using the high/low prices. Otherwise the function calculates the max/min of the single series.

Author(s)

Joshua Ulrich

References

The following site(s) were used to code/document this indicator:
https://www.fmlabs.com/reference/default.htm?url=ZigZag.htm
https://www.linnsoft.com/techind/zig-zag-indicator-zig-zzo
https://www.linnsoft.com/techind/zig-zag-oscillator-indicator-zzo
https://www.metastock.com/Customer/Resources/TAAZ/#127
https://school.stockcharts.com/doku.php?id=technical_indicators:zigzag

Examples

```
## Get Data and Indicator ##
data(ttrc)
zz <- ZigZag( ttrc[,c("High", "Low")], change=20 )
```

Index