The TTR Package

October 25, 2007

Type Package

Title Technical Trading Rules

Version 0.13-1

Date 2007-10-23

Author Josh Ulrich

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

Description Functions and data to construct technical trading rules with R.

License GPL version 2 or later

R topics documented:

- ADX .......................................................... 2
- ATR .......................................................... 3
- CCI .................................................................. 5
- CLV .................................................................. 6
- CMO ............................................................. 7
- DPO ............................................................. 8
- EMV .............................................................. 9
- KST ................................................................ 10
- MovingAverages ................................................. 12
- OBV ............................................................. 14
- Oscillators ....................................................... 15
- RSI ............................................................. 16
- SAR ............................................................. 18
- TDI ............................................................. 19
- TRIX ............................................................. 20
- TTR-package ................................................... 21
- VHF ............................................................. 23
- WPR ............................................................. 24
- WebData ......................................................... 25
- aroon .......................................................... 27
Welles Wilder's Directional Movement Index

**Description**

Directional Movement Index; developed by J. Welles Wilder.

**Usage**

```r
ADX(HLC, n=14, ma.adx=list("EMA", n=n, wilder=TRUE))
```

**Arguments**

- **HLC**
  - Object able to be coerced to a matrix, which contains High-Low-Close prices.
- **n**
  - Number of periods to use for DX calculation (not ADX calculation).
- **ma.adx**
  - A list whose first component is a string containing the ADX moving average function name; additional parameters may also be specified as named components.

**Details**

The $D_{p}/D_{n}$ (positive/negative) is the percentage of the true range that is up/down.

**Value**

A matrix containing the columns:

- **$D_{p}$**
  - The positive Direction Index.
- **$D_{n}$**
  - 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 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)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:
http://www.fmlabs.com/reference/DI.htm
http://www.fmlabs.com/reference/DX.htm
http://www.fmlabs.com/reference/ADX.htm
http://www.fmlabs.com/reference/ADXR.htm
http://linnsoft.com/tour/techind/dirInd.htm
http://linnsoft.com/tour/techind/adx.htm
http://linnsoft.com/tour/techind/adxr.htm
http://stockcharts.com/education/IndicatorAnalysis/indic_ADX.html

See Also

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

Examples

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

---

### **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, ma = list("EMA", n=14, wilder=TRUE))
```

**Arguments**

- **HLC**
  
  Object able to be coerced to a matrix, which contains High-Low-Close prices.

- **ma**
  
  A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components.
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 matrix containing the columns:

- tr: The true range of the series.
- atr: The average (as specified by ma) true range of the series.
- true.high: The true high of the series.
- true.low: The true low of the series.

Author(s)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:

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)
attr <- ATR(ttrc[,c("High","Low","Close")], ma = list("EMA", n=14, wilder=TRUE))
Description

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

Usage

```r
CCI(HLC, ma = list("SMA", n=20), c=0.015)
```

Arguments

- **HLC**: High-Low-Close price series to use. If only a univariate series is given, it will be used. See details.
- **ma**: A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components.
- **c**: Constant to apply to the mean deviation.

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 vector 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)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:

- http://stockcharts.com/education/IndicatorAnalysis/indic_CCI.html
See Also

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

Examples

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

<table>
<thead>
<tr>
<th>CLV</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Close Location Value</strong></td>
</tr>
</tbody>
</table>

Description

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

Usage

CLV(HLC)

Arguments

HLC Object able to be coerced to a matrix, which 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 vector containg the Close Location Values of a High-Low-Close price series.

Author(s)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:
http://stockcharts.com/education/IndicatorAnalysis/indic_AccumDistLine.html

See Also

See chaikinAD, which uses CLV.
**Examples**

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

---

**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 to use.
- `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 vector 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)**

Josh Ulrich

**References**

The following site(s) were used to code/document this indicator:

See Also

See RSI.

Examples

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

---

**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**

\[
\text{DPO}(x, \ ma = \text{list("SMA", n=10)}, \ shift = \text{ma$n/2+1}, \ percent = \text{FALSE})
\]

**Arguments**

- **x**  
  Price, volume, etc. series to use.
- **ma**  
  A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components.
- **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.

**Details**

The Detrended Price shows cycles and overbought / oversold conditions. Note the calculation shifts the results shift periods, so the last shift periods will be zero.

**Value**

A vector containing the DPO values.

**Note**

As stated above, the DPO can be used on any univariate series, not just price.

**Author(s)**

Josh Ulrich
EMV

References
The following site(s) were used to code/document this indicator:
http://www.fmlabs.com/reference/DPO.htm
http://www.equis.com/Customer/Resources/TAAZ/?c=3&p=48

See Also
See EMA, SMA, etc. for moving average options; and note Warning section. See oscillator for other oscillators.

Examples
```r
data(ttrc)
price.dpo <- DPO(ttrc[,"Close"])
volume.dpo <- DPO(ttrc[,"Volume"])
```

---

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

```r
EMV(HL, volume, ma=list("SMA", n=9), vol.divisor=10000)
```

Arguments

- **HL**: Object able to be coerced to a matrix, which contains a High-Low price series.
- **volume**: Vector or matrix of volume observations corresponding to the HL object.
- **ma**: A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components.
- **vol.divisor**: An increment to make the results larger and easier to work with.

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 matrix containing the columns:

- **emv**: The ease of movement values.
- **ma.emv**: 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)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:
http://www.fmlabs.com/reference/ArmsEMV.htm
http://www.equis.com/Customer/Resources/TAAZ/?c=3&p=51
http://linnsoft.com/tour/techind/arms.htm

See Also

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

Examples

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

---

**KST**

**Know Sure Thing**

Description

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

Usage

```r
KST(price, n = c(10, 15, 20, 30), ma1 = list("SMA", n=10), ma2 = ma1, ma3 = ma1,
    ma4 = list("SMA", n=15), ma.sig = list("SMA", n=10), wts = 1:4)
```

Arguments

- **price**
  Price series to use.
- **n**
  A vector of the number of periods to use in the ROC calculations.
- **ma1**
  A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components. There must be a `ma` (not including `ma.sig`) for each period in `n`.
- **ma2**
  See `ma1`.
KST

ma3 See ma1.
ma4 See ma1.
ma.sig A list whose first component is a string containing the signal moving average function name; additional parameters may also be specified as named components.
wts A vector the same length as n, of the weight for each period (need not sum to one).

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 vector 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)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:
http://www.pring.com/index.html
http://www.pring.com/movieweb/daily_kst.htm
http://www.pring.com/articles/article28.htm
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 oscillator for other oscillators.

Examples

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

Calculate various moving averages (MA) of a series.

Usage

SMA(x, n=10)
EMA(x, n=10, wilder=FALSE)
WMA(x, n=10, wts=1:n)
DEMA(x, n=10)
EVWMA(price, volume, n=10)
ZLEMA(x, n=10)

Arguments

x Vector to be averaged.
price Vector of prices to be averaged.
volume Volume series corresponding to price series, or a constant. See Notes.
n Number of periods to average over.
wts Vector of weights. Length of wts vector must equal the length of x, or n (the default).
wilder logical; if TRUE, a Welles Wilder type EMA will be calculated; see notes.

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 the values of wts as weights.

DEMA is calculated as: DEMA = 2 * EMA(x, n) - EMA(EMA(x, n), n).

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 to minimize the cumulative effect.

Value

SMA Simple moving average.
EMA Exponential moving average.
WMA Weighted moving average.
MovingAverages

DEMA  Double-exponential moving average.
EVWMA  Elastic, volume-weighted moving average.
ZLEMA  Zero lag exponential 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 $1/n$.

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.

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)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:
http://www.fmlabs.com/reference/ExpMA.htm
http://www.fmlabs.com/reference/WeightedMA.htm
http://www.fmlabs.com/reference/DEMA.htm
http://linnsoft.com/tour/techind/evwma.htm

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"], 20)
  zlema.20 <- ZLEMA(ttrc[,"Close"], 20)

  ## Example of short-term instability of EMA
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 to use.
volume   Vector or matrix of volume observations corresponding 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 vector containing the OBV values.

Note

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

Author(s)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:
http://www.fmlabs.com/reference/OBV.htm
http://www.equis.com/Customer/Resources/TAAZ?c=3\p=82
http://stockcharts.com/education/IndicatorAnalysis/indic-obv.htm
Oscillators

See Also

See chaikinAD.

Examples

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

Oscillators

Description

The oscillators documented in this page compare a fast moving average (MA) of a series with a slow MA of the same series. The most popular price oscillator is probably the MACD, which was developed by Gerald Appel.

Usage

oscillator(x, ma.slow = list("EMA", n=20), ma.fast = list("EMA", n=10), ma.sig = list("EMA", n=10), percent = FALSE)
MACD(x)

Arguments

x Series to use; usually price, but can be volume, etc.
ma.slow A list whose first component is a string containing the slow moving average function name; additional parameters may also be specified as named components.
ma.fast Similar to ma.slow, but for the fast moving average.
ma.sig Similar to ma.slow, but for the signal line 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.

Details

The oscillator function either subtracts the short MA from the long MA, or finds the rate of change between the short MA and the long MA.

Value

The MACD function returns a matrix containing the columns:

macd The MACD line.
signal The MACD signal line (a moving average of the MACD).
oscillator 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 function is provided for convenience. Time periods for the MACD are often given as 26 and 12, but the function originally used exponential constants of 0.075 and 0.15, which are closer to 25.6667 and 12.3333 periods.

Author(s)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:
http://www.fmlabs.com/reference/MACD.htm
http://www.fmlabs.com/reference/PriceOscillator.htm
http://stockcharts.com/education/IndicatorAnalysis/indic_MACD1.html
http://stockcharts.com/education/IndicatorAnalysis/indic_priceOscillator.html

See Also

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

Examples

data(ttrc)

macd <- MACD(ttrc,"Close")
osc <- oscillator(ttrc,"Close", ma.slow = list("EMA",n=26),
                ma.fast = list("EMA",n=12), ma.sig = list("EMA", n=9) )

# Should be TRUE
identical(macd,osc)

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, ma.up=list("EMA", n=14, wilder=TRUE), ma.down=ma.up)
Arguments

price: Price series to use.

ma.up: A list whose first component is a string containing the upward price movement moving average function name; additional parameters may also be specified as named components.

ma.down: Similar to ma.up, but for the downward price movement moving average.

Details

The RSI calculation is \( RSI = 100 - \frac{100}{1 + RS} \), where \( RS \) is the smoothed ratio of 'average' gains over 'average' losses. The 'average' aren’t true averages, since they’re divided by the value of \( n \) not the number of gain/loss periods.

Value

A vector 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 stochastic on RSI values.

Author(s)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:

- http://www.equis.com/Customer/Resources/TAAZ/?c=3&p=100

See Also

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

Examples

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

*Parabolic Stop-and-Reverse*

**Description**

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

**Usage**

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

**Arguments**

- `HL`: Object able to be coerced to a matrix, which contains High-Low prices.
- `accel`: 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 vector containing the Parabolic Stop and Reverse values.

**Author(s)**

Josh Ulrich

**References**

The following site(s) were used to code/document this indicator:


**See Also**

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

**Examples**

```r
data(ttrc)
sar <- SAR(ttrc[,c("High","Low")])
```
Trend Detection Index (TDI) attempts to identify starting and ending trends. Developed by M. H. Pee.

Usage

TDI(price, n = 20)

Arguments

price Price series to use.
n Number of periods to use.

Details

The TDI is the (1) absolute value of the \( n \)-day sum of the \( n \)-day momentum, minus the quantity of (2) \( 2n \)-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 matrix 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)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:

http://www.linnsoft.com/tour/techind/tdi.htm
TRIX

See Also

See aroon, CCI, ADX, VHF 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, ma1 = list("EMA", n=20), ma2 = ma1, ma3 = ma1, ma.sig = list("EMA", n=9), percent = TRUE)

Arguments

price  
A list series to use.

ma1     
A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components.

ma2     
See ma1.

ma3     
See ma1.

ma.sig  
Similar to ma1, but for the signal moving average.

percent 
logical; if TRUE, the rate of change is calculated using the ROC function, otherwise the momentum function is used.

Details

The TRIX is calculated as follows:

\[
3\text{MA} = \text{ma3}(\text{ma2} (\text{ma1}(\text{price})))
\]

\[
\text{trix} = 100 \times \left[ \frac{3\text{MA}(t)}{3\text{MA}(t-1)} - 1 \right]
\]

Value

A vector 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)
Josh Ulrich

References
The following site(s) were used to code/document this indicator:
http://www.equis.com/Customer/Resources/TAAZ/?c=3&p=114
http://www.linnsoft.com/tour/techind/trix.htm
http://stockcharts.com/education/IndicatorAnalysis/indic_trix.htm

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

Examples

data(ttrc)
trix <- TRIX( ttrc[,"Close"] )

TTR-package 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

Package: TTR
Type: Package
Version: 0.13-1
Date: 2007-10-23
License: GPL Version 2 or later.

Users will probably be most interested in the following functions:
bollingerBands
changes
ADX
MovingAverages
Oscillators
RSI
stochastic
WebData
Author(s)

Josh Ulrich

Maintainer: Josh Ulrich

References

The following site(s) were used to code/document this package:
http://www.fmlabs.com/reference/default.htm
http://www.equis.com/Customer/Resources/TAAZ/?p=0
http://www.linnsoft.com/tour/technicalindicators.htm
http://stockcharts.com/education/IndicatorAnalysis/

Examples

data(ttrc)

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

# Directional Movement Index
dmi.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
stoch.osc <- stochastic(ttrc[,c("High","Low","Close")])

### Note: you must have a working internet connection
### for the examples below to work!

# Fetch U.S. symbols from the internet
nyse.symbols <- stockSymbols("NYSE")

# Fetch Yahoo! Finance data from the internet
ibm <- getYahooData("IBM", 19990404, 20050607)
**VHF**

*Vertical Horizontal Filter*

---

**Description**

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

**Usage**

\[ \text{VHF}(\text{price}, n = 28) \]

**Arguments**

- **price**
  - Object able to be coerced to a matrix, which contains either 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 vector 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)**

Josh Ulrich

**References**

The following site(s) were used to code/document this indicator:


**See Also**

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

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

---

**WPR**

**William’s % R**

Description

William’s % R.

Usage

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

Arguments

- **HLC**: High-Low-Close price series to use. 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 vector containing the William’s %R values.

Note

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

Author(s)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:

- http://linnsoft.com/tour/techind/willR.htm
See Also

See `stochastic`.

Examples

data(ttrc)
stoch.osc <- stochastic(ttrc[,c("High","Low","Close")])
stoch.wpr <- WPR(ttrc[,c("High","Low","Close")])

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

WebData Fetch Internet Data

Description

Get investment data from the internet.

Usage

```r
getYahooData(symbol, start, end, freq = "daily", type = "price", adjust = TRUE, quiet = FALSE)
stockSymbols(exchange = c("AMEX","NASDAQ","NYSE"), sort.by = c("Exchange","Symbol"), quiet = FALSE)
```

Arguments

- **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.
- **quiet**: Logical; if TRUE, status messages will be printed to the console.
- **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”.

Details

`getYahooData` fetches individual stock data from the Yahoo! Finance website. It also adjusts price for splits and dividends, and volume for splits.

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

Value

`getYahooData` returns a numeric matrix containing the columns:

<table>
<thead>
<tr>
<th>Date</th>
<th>Trade date, in CCYYMMDD format.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>Open price.</td>
</tr>
<tr>
<td>High</td>
<td>High price.</td>
</tr>
<tr>
<td>Low</td>
<td>Low price.</td>
</tr>
<tr>
<td>Close</td>
<td>Close price.</td>
</tr>
<tr>
<td>Volume</td>
<td>Volume.</td>
</tr>
</tbody>
</table>

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

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)

Josh Ulrich

Examples

```r
### Note: you must have a working internet
### connection for these examples to work!
ibm <- getYahooData("IBM", 19990404, 20050607)
nyse.symbols <- stockSymbols("NYSE")
```
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

\texttt{aroon(HL, n=20)}

Arguments

- \texttt{HL} Object able to be coerced to a matrix, which contains either a High-Low price series, or a Close price series.
- \texttt{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 \((1/n) \times 100\).

Value

A matrix containing the columns:

- \texttt{aroon.up} The Aroon up indicator.
- \texttt{aroon.dn} The Aroon down indicator.
- \texttt{oscillator} The Aroon oscillator \texttt{(aroon.up - aroon.dn)}.

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 aroon.up(dn) is between 70 and 100. Strong trends are indicated when when the aroon.up(dn) is above 70 while the aroon.dn(up) is below 30. Also, crossovers may be useful.

Author(s)

Josh Ulrich
References

The following site(s) were used to code/document this indicator:
http://www.fmlabs.com/reference/Aroon.htm
http://www.fmlabs.com/reference/AroonOscillator.htm
http://www.linnsoft.com/tour/techind/aroon.htm
http://stockcharts.com/education/IndicatorAnalysis/indic-Aroon.htm

See Also

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

Examples

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

bollingerBands  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

bollingerBands(HLC, ma = list("SMA", n=20), sd = list(FUN="sd", n=2))

Arguments

- **HLC**  High-Low-Close price series to use. If only a univariate series is given, it will be used. See details.
- **ma**  A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components.
- **sd**  A list of standard deviation input.

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$n 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 matrix containing the columns:

- **dn**: The lower Bollinger Band.
- **ma**: The middle Moving Average (see notes).
- **up**: The upper Bollinger Band.
- **pct.b**: 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)**

Josh Ulrich

**References**

The following site(s) were used to code/document this indicator:
- [http://stockcharts.com/education/IndicatorAnalysis/indic_BBWidth.htm](http://stockcharts.com/education/IndicatorAnalysis/indic_BBWidth.htm)

**See Also**

See [EMA](#), [SMA](#), etc. for moving average options; and note Warning section.

**Examples**

```r
## 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 <- bollingerBands( ttrc[,c("High","Low","Close")])
bbands.close <- bollingerBands( ttrc,"Close")
```
chaikinAD

| 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**

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

**Arguments**

- **HLC**: Object able to be coerced to a matrix, which 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 vector 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)**

Josh Ulrich

**References**

The following site(s) were used to code/document this indicator:

- [http://www.fmlabs.com/reference/AccumDist.htm](http://www.fmlabs.com/reference/AccumDist.htm)
- [http://stockcharts.com/education/IndicatorAnalysis/indic_AccumDistLine.html](http://stockcharts.com/education/IndicatorAnalysis/indic_AccumDistLine.html)
chaikinMF

See Also

See OBV, and CLV.

Examples

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

chaikinMF 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

chaikinMF(HLC, volume, n = 20)

Arguments

HLC Object able to be coerced to a matrix, which 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 vector 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)

Josh Ulrich
References

The following site(s) were used to code/document this indicator:
http://www.linnsoft.com/tour/techind/cmf.htm

See Also

See CLV, and chaikinAD.

Examples

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

chaikinOscillator Chaikin Oscillator

Description

The Chaikin Oscillator is an oscillator applied to the Chaikin Accumulation / Distribution line.

Usage

chaikinOscillator(HLC, volume, ma.slow = list("EMA", n=10),
                 ma.fast = list("EMA", n=3), percent=FALSE)

Arguments

HLC Object able to be coerced to a matrix, which contains High-Low-Close prices.
volume Vector or matrix of volume observations corresponding to the HLC object.
ma.slow A list whose first component is a string containing the slow moving average function name; additional parameters may also be specified as named components.
ma.fast A list whose first component is a string containing the fast moving average function name; additional parameters may also be specified as named components.
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.

Details

The Chaikin Oscillator is calculated similarly to other price oscillators: by subtracting a slow moving average of the A/D line from a fast moving average of the A/D line.
**Value**

A vector containing the Chaikin Oscillator values.

**Note**

When the Chaikin Oscillator crosses above (below) zero, it indicates a buy (sell) signal. Also look for price divergence to indicate bullish or bearish conditions.

**Author(s)**

Josh Ulrich

**References**

The following site(s) were used to code/document this indicator:

http://www.fmlabs.com/reference/CaiaikinOscillator.htm

http://www.equis.com/Customer/Resources/TAAZ/?c=3&p=41

http://stockcharts.com/education/IndicatorAnalysis/indic_ChaikinOscillator.html

**See Also**

See EMA, SMA, etc. for moving average options; and note Warning section. Also see CLV, chaikinAD, and oscillator.

**Examples**

```r
data(ttrc)
oscillator <- chaikinOscillator(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**

```r
chaikinVolatility(HL, ma = list("EMA", n=10))
```

**Arguments**

- **HL**: Object able to be coerced to a matrix, which contains High-Low prices.
- **ma**: A list whose first component is a string containing the moving average function name; additional parameters may also be specified as named components.
Details

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

Value

A vector 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)

Josh Ulrich

References

The following site(s) were used to code/document this indicator:

See Also

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

Examples

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

<table>
<thead>
<tr>
<th>changes</th>
<th>Rate of Change / Momentum</th>
</tr>
</thead>
</table>

Description

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

Usage

```r
ROC(x, n = 1, type = c("discrete", "continuous"), na = NA)
momentum(x, n = 1, na = NA)
```
stochastic

Arguments

x  Price, volume, etc. series to use.
n  Number of periods to use.
type  Compounding type; either "discrete" (the default) or "continuous".
na  How should periods prior to n be represented? Default is NA.

Details

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

Value

ROC returns a vector containing the rate-of-change (or return) values.
momentum returns a vector containing the differenced price series.

Author(s)

Josh Ulrich

Examples

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

stochastic

Stochastic Oscillator

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.

Usage

stochastic(HLC, n.fastK=14, ma.fastD=list("SMA", n=3), ma.slowD=ma.fastD)

Arguments

HLC  High-Low-Close price series to use. If only a univariate series is given, it will be used. See details.
n.fastK  Number of periods for fast %K (i.e. the number of past periods to use).
ma.fastD  A list whose first component is a string containing the fast %D moving average function name; additional parameters may also be specified as named components.
ma.slowD  Similar to ma.fastD, but for the fast %D moving average.
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. 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).

Value

A matrix containing the columns:

- fastK Fast %K
- fastD Fast %D
- slowD Slow %D

Note

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

Josh Ulrich

References

The following site(s) were used to code/document this indicator:

- [http://linnsoft.com/tour/techind/stoc.htm](http://linnsoft.com/tour/techind/stoc.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

```r
data(ttrc)
stoch.osc <- stochastic(ttrc[,c("High","Low","Close")])
stoch.wpr <- WPR(ttrc[,c("High","Low","Close")])
plot(tail(stoch.osc[,"fastK"], 100), type="l", main="Fast %K and Williams %R",
```
ttr.tools

Miscellaneous Tools

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

Usage

```r
growth(price, signals, ...)
rollFUN(x, n, FUN, ...)
lags(x, n = 1)
wilderSum(x, n = 10)
```

Arguments

- **price**: Price series to use.
- **signals**: Signals to use (defaults to vector of ones). Use ‘0’ for no position, ‘1’ for long position, and ‘-1’ for short position.
- **x**: Object able to be coerced to a matrix.
- **n**: Number of periods to use.
- **FUN**: Function to calculate over rolling subsets of data.
- **...**: Further arguments to be passed from or to other methods.

Details

- **growth**: calculates the growth of an investment using given prices and signals.
- **rollFUN**: allows the calculation of rolling functions.
- **lags**: calculates the lags of a given series.
- **wilderSum**: calculates a Welles Wilder style weighted sum.

Value

- **growth**: returns a vector of the growth of the investment.
- **rollFUN**: returns a vector of function results for the rolling subsets of data.
- **lags**: returns a matrix of lagged values of the original vector.
- **wilderSum**: returns a vector of weighted sums.
Note
In \texttt{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)
Josh Ulrich

---

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.

Usage
data(ttrc)

Format
The format is:

<table>
<thead>
<tr>
<th>Date:</th>
<th>Class ‘Date’ num [1:5550]</th>
</tr>
</thead>
<tbody>
<tr>
<td>5480</td>
<td>5481 5482 5485 5486 ...</td>
</tr>
<tr>
<td>Open:</td>
<td>num</td>
</tr>
<tr>
<td>3.18</td>
<td>3.09 3.11 3.09 3.10 ...</td>
</tr>
<tr>
<td>High:</td>
<td>num</td>
</tr>
<tr>
<td>3.18</td>
<td>3.15 3.12 3.12 3.12 ...</td>
</tr>
<tr>
<td>Low:</td>
<td>num</td>
</tr>
<tr>
<td>3.08</td>
<td>3.09 3.08 3.07 3.08 ...</td>
</tr>
<tr>
<td>Close:</td>
<td>num</td>
</tr>
<tr>
<td>3.08</td>
<td>3.11 3.09 3.10 3.11 ...</td>
</tr>
<tr>
<td>Volume:</td>
<td>num</td>
</tr>
<tr>
<td>1870906</td>
<td>3099506 2274157 2086758 2166348 ...</td>
</tr>
</tbody>
</table>

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

Source
Randomly generated.

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

*Topic datasets
  ttrc, 38

*Topic package
  TTR-package, 21

*Topic ts
  ADX, 1
  aroon, 27
  ATR, 3
  bollingerBands, 28
  CCI, 4
  chaikinAD, 30
  chaikinMF, 31
  chaikinOscillator, 32
  chaikinVolatility, 33
  changes, 34
  CLV, 5
  CMO, 6
  DPO, 7
  EMV, 8
  KST, 9
  MovingAverages, 11
  OBV, 13
  Oscillators, 14
  RSI, 16
  SAR, 17
  stochastic, 35
  TDI, 18
  TRIX, 20
  ttr.tools, 37
  VHF, 22
  WebData, 25
  WPR, 24
  %D(stochastic), 35
  %K(stochastic), 35

  ADX, 1, 5, 18, 19, 21, 23, 28
  aroon, 2, 5, 19, 23, 27
  ATR, 2, 3, 18
  bollingerBands, 21, 28

  CCI, 2, 4, 19, 23, 28
  chaikinAD, 6, 14, 30, 32, 33
  chaikinMF, 31
  chaikinOscillator, 32
  chaikinVolatility, 4, 33
  changes, 21, 34
  CLV, 5, 31–33
  CMO, 6, 17
  DEMA(MovingAverages), 11
  DI(ADX), 1
  DPO, 7
  DX, 4
  DX(ADX), 1
  EMA, 2, 4, 5, 8, 9, 11, 16, 17, 21, 29, 33, 34, 36
  EMA(MovingAverages), 11
  EMV, 8
  EVWMA(MovingAverages), 11
  getYahooData(WebData), 25
  growth(ttr.tools), 37
  KST, 9
  lags(ttr.tools), 37
  MACD(Oscillators), 14
  momentum(changes), 34
  MovingAverages, 11, 21
  OBV, 13, 31
  oscillator, 8, 11, 33
  oscillator(Oscillators), 14
  Oscillators, 14, 21
  ROC, 11
  ROC(changes), 34
  rollFUN(ttr.tools), 37
  RSI, 7, 16, 21
  SAR, 17
SMA, 2, 4, 5, 8, 9, 11, 16, 17, 21, 29, 33, 34, 36
SMA (MovingAverages), 11
stochastic, 17, 21, 24, 35
stockSymbols (WebData), 25

TDI, 2, 5, 18, 23, 28
TR, 34
TR (ATR), 3
TRIX, 20
TTR (TTR-package), 21
TTR-package, 21
ttr.tools, 37
ttrc, 38

VHF, 2, 5, 19, 22, 28

WebData, 21, 25
wilderSum, 13
wilderSum (ttr.tools), 37
WMA (MovingAverages), 11
WPR, 24, 36

ZLEMA (MovingAverages), 11