Package 'ichimoku'

February 5, 2024

Type Package

Title Visualization and Tools for Ichimoku Kinko Hyo Strategies

Version 1.5.0

Description An implementation of 'Ichimoku Kinko Hyo', also commonly known as 'cloud charts'. Static and interactive visualizations with tools for creating, backtesting and development of quantitative 'ichimoku' strategies. As described in Sasaki (1996, ISBN:4925152009), the technique is a refinement on candlestick charting, originating from Japan and now in widespread use in technical analysis worldwide. Translating as 'one-glance equilibrium chart', it allows the price action and market structure of financial securities to be determined 'at-a-glance'. Incorporates an interface with the OANDA fxTrade API <https://developer.oanda.com/> for retrieving historical and live streaming price data for major currencies, metals, commodities, government bonds and stock indices.

BugReports https://github.com/shikokuchuo/ichimoku/issues

License GPL (>= 3)

URL https://shikokuchuo.net/ichimoku/, https://github.com/shikokuchuo/ichimoku/

Encoding UTF-8

LazyData true

Depends R (>= 3.5)

Imports ggplot2 (>= 3.4.0), mirai (>= 0.12.0), nanonext (>= 0.12.0), RcppSimdJson (>= 0.1.9), secretbase, shiny (>= 1.4.0), xts, zoo

LinkingTo RcppSimdJson, xts

Suggests keyring, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation yes

Author Charlie Gao [aut, cre] (<https://orcid.org/0000-0002-0750-061X>),
Hibiki AI Limited [cph]

Maintainer Charlie Gao <charlie.gao@shikokuchuo.net>

Repository CRAN

Date/Publication 2024-02-05 21:10:02 UTC

R topics documented:

ichimoku-package *ichimoku: Visualization and Tools for Ichimoku Kinko Hyo Strategies*

Description

An implementation of 'Ichimoku Kinko Hyo', also commonly known as 'cloud charts'. Static and interactive visualizations with tools for creating, backtesting and development of quantitative 'ichimoku' strategies. As described in Sasaki (1996, ISBN:4925152009), the technique is a refinement on candlestick charting, originating from Japan and now in widespread use in technical analysis worldwide. Translating as 'one-glance equilibrium chart', it allows the price action and market structure of financial securities to be determined 'at-a-glance'. Incorporates an interface with the OANDA fxTrade API https://developer.oanda.com/ for retrieving historical and live streaming price data for major currencies, metals, commodities, government bonds and stock indices.

Principal ichimoku functions

Data & Visualization

- ichimoku to create an ichimoku object from price data.
- plot.ichimoku / iplot to plot (interactive) cloud charts from ichimoku objects.
- archive for reading/writing objects to/from archive files with data verification.
- oanda to retrieve price data from the OANDA fxTrade API.

Strategies & ML

- strat to augment an ichimoku object with a strategy, including combined and asymmetric strategies.
- autostrat to automatically evaluate and rank top-performing strategies.
- mlgrid to generate a numeric representation of the ichimoku cloud chart.
- relative to produce a statistical summary of the latest ichimoku numeric representation relative to historical values.

Real-time

- oanda_chart to plot real-time ichimoku cloud charts using OANDA data.
- oanda_studio a complete live analysis environment using OANDA data implemented in R Shiny.
- oanda_stream / oanda_quote to obtain the latest live data stream / quote from the OANDA fxTrade API.
- oanda_view for a market overview showing the relative performance of constituents.
- oanda_orders / oanda_positions to retrieve the aggregate OANDA fxTrade order / position book.

Author(s)

Charlie Gao <charlie.gao@shikokuchuo.net> (ORCID)

References

Sasaki, H. (1996), *ichimoku kinkouhyou no kenkyuu*. Tokyo, Japan: Toushi Radar.

See Also

Useful links:

- https://shikokuchuo.net/ichimoku/
- https://github.com/shikokuchuo/ichimoku/
- Report bugs at https://github.com/shikokuchuo/ichimoku/issues

---

`archive` *Read/write Objects <> Archive Files with Data Verification*

Description

Read and write objects to/from archival storage in the native RData format, with verification of data integrity.

Usage

```
archive(..., object, file)
```

Arguments

`...`	unnamed arguments will be parsed as 'file' if there is only one argument, 'object' and 'file' if there are two arguments.
`object`	(for write operations) an object.
`file`	the name of the file or a connection where the object is saved to or read from.

Details

For read operations: specify only 'file', or alternatively if no arguments are specified, a system dialog will be opened allowing a file to be chosen interactively. 'file' is read and the return value may be assigned to an object. A confirmation message is issued if the file read operation has been successful.

For write operations: specify both 'object' and 'file'. If only 'object' is specified and 'file' is left empty (see examples), a system dialog will be opened allowing the file save location to be chosen interactively. 'object' will be written to 'file'. A confirmation message is issued if the file write operation has been successful.

Value

For read operations: the object originally archived.

For write operations: the filename supplied. 'object' is written to 'file'.

Data Verification

A SHA3-256 hash of the original object is written to the archive. This allows the data integrity of the restored object to be verified when the archive is read back.

For write operations: confirmation of the SHA3-256 hash written to file is displayed.

For read operations: a 'data verified' message is issued if the SHA256 hash found within the data file has been authenticated.

Further Details

Please refer to the reference vignette by calling: `vignette("reference", package = "ichimoku")`

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
file <- tempfile()

archive(cloud, file)

restored <- archive(file)

unlink(file)

if (interactive()) {
# Only run examples in interactive R sessions
# Read file to 'object' using system dialog:
object <- archive()

# Write 'cloud' to file using system dialog:
archive(cloud, )
}
```

---

as.data.frame.ichimoku

Convert ichimoku to data.frame

Description

An optimised 'ichimoku' to 'data.frame' constructor.

Usage

```
## S3 method for class 'ichimoku'
as.data.frame(x, row.names, optional, keep.attrs = FALSE, ...)
```

Arguments

x	an object of class 'ichimoku'.
row.names	not used.
optional	not used.
keep.attrs	[default FALSE] if set to TRUE, will preserve any custom attributes set on the original object.
...	arguments passed to or from other methods.

Details

This function is an S3 method for the generic function as.data.frame() for class 'ichimoku'. It can be invoked by calling as.data.frame(x) on an object 'x' of class 'ichimoku'.

Value

A 'data.frame' object. The ichimoku object index is preserved as the first column with header 'index'.

Examples

```
cloud <- ichimoku(sample_ohlc_data)
df <- as.data.frame(cloud)
str(df)

df2 <- as.data.frame(cloud, keep.attrs = TRUE)
str(df2)
```

---

autostrat *Automated Ichimoku Strategies*

Description

Generate a list of the top performing ichimoku cloud strategies based on simple indicator conditions of the form 'c1 > c2' (level 1), complex combined strategies of the form 'c1 > c2 & c3 > c4' (level 2), or complex asymmetric strategies of the form 'c1 > c2 x c3 > c4' (level 3).

Usage

```
autostrat(x, n = 8, dir = c("long", "short"), level = 1, quietly)
```

Arguments

x	an ichimoku object.
n	[default 8] select top 'n' number of strategies to return.
dir	[default 'long'] trade direction, either 'long' or 'short'.
level	[default 1] to return simple strategies. For complex strategies, set level to 2 to return combined strategies of the form 's1 & s2' or level to 3 to return asymmetric strategies of the form 's1 x s2'.
quietly	(optional) if set to TRUE, will suppress printing of additional output to the console and return quietly.

Details

Ichimoku objects for each strategy are returned as a list. The cumulative log returns for all strategies as well as the summaries for the top 'n' strategies are saved as attributes to the list. This information may be retrieved by using `look` on the returned list.

Each individual ichimoku object may be accessed via its position in the list, e.g. [[1]] for the 1st item.

Value

Returned invisibly, a list of 'n' ichimoku objects containing strategies, with attributes 'logret' (a vector of cumulative log returns for all strategies) and 'summary' (a matrix of summaries for the top 'n' strategies).

In addition, the strategy summaries are printed to the console.

Further Details

Please refer to the strategies vignette by calling: `vignette("strategies", package = "ichimoku")`

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")

stratlist <- autostrat(cloud, n = 3, quietly = TRUE)
look(stratlist)
strat <- stratlist[[2]]
summary(strat)

autostrat(cloud, n = 1, dir = "short", level = 2)
autostrat(cloud, n = 1, dir = "long", level = 3)
```

coredata *Extract the Core Data of Ichimoku Objects*

Description

Method for extracting the core data matrix of ichimoku objects.

Usage

```
## S3 method for class 'ichimoku'
coredata(x, fmt, ...)
```

Arguments

x	an object of class 'ichimoku'.
fmt	(optional) set to TRUE to retain the index as row names of the returned matrix, or a character string passed on to the 'format' argument of `format.POSIXct()` to format these values in a specific way.
...	arguments passed to or from other methods.

Details

This function is an S3 method for the generic function coredata() for class 'ichimoku'. It can be invoked by calling coredata(x) on an object 'x' of class 'ichimoku'.

Value

A numeric matrix containing the ichimoku object data, stripped of the index unless 'fmt' is specified in which case the index will be retained as character values in the matrix row names.

Examples

```
cloud <- ichimoku(sample_ohlc_data)
coredata(cloud)[101:120, ]
```

df_append *Append New Data to Dataframe*

Description

Update a 'data.frame' object with new data. Can be used to append new updated time series data to an existing dataframe, where each observation is indexed by a unique timestamp/identifier in a key column.

Usage

```
df_append(old, new, key = "time", keep.attr = "timestamp")
```

Arguments

old	data.frame object containing existing data.
new	data.frame object containing new data.
key	[default 'time'] column name used as key, provided as a character string.
keep.attr	[default 'timestamp'] name of an attribute in 'new' to retain, if present, provided as a character string.

Details

Can be used to update price dataframes retrieved by oanda. The function is designed to update existing data with new values as they become available. As opposed to df_merge, the data in 'new' will overwrite the data in 'old' rather than create duplicates.

Value

A data.frame of the existing data appended with the new data. If the data in 'new' contains data with the same value for the key column as 'old', the data in 'new' will overwrite the data in 'old'.

If the attribute specified by 'keep.attr' is present in 'new', this is retained. All other non-required attributes are dropped.

Examples

```
data1 <- sample_ohlc_data[1:8, ]
data1
data2 <- sample_ohlc_data[7:10, ]
data2
df_append(data1, data2)
```

---

df_merge *Merge Dataframes*

Description

Full join on an arbitrary number of 'data.frame' objects passed as arguments, preserving all unique entries. Can be used to combine historical time series data where each observation is indexed by a unique timestamp and all periods are complete.

Usage

```
df_merge(...)
```

Arguments

...	data.frame objects to combine.

Details

Can be used to join price dataframes retrieved by oanda. The function is designed to join complete historical data. If the data to be merged contains data with incomplete periods, all entries are preserved rather than updated. If incomplete periods are detected within the data, a warning is issued, and the resulting dataframe should be manually checked in case it contains unwanted duplicates. Use df_append for updating dataframes with new values.

Value

A data.frame containing all unique entries in the objects passed as argument.

Examples

```
data1 <- sample_ohlc_data[1:6, ]
data1
data2 <- sample_ohlc_data[4:10, ]
data2
df_merge(data1, data2)
```

---

hasStrat *hasStrat*

Description

A function for checking if an object contains a strategy.

Usage

```
hasStrat(x)
```

Arguments

x	an object.

Details

Designed to be used by ichimoku functions that are either S3 methods for class 'ichimoku' or after validation that 'x' is an ichimoku object, hence there is no check on the class of 'x' within this function.

Value

A logical value of TRUE if the 'strat' attribute of 'x' is set, otherwise FALSE.

Examples

```
cloud <- ichimoku(sample_ohlc_data)
strat <- strat(cloud)

# TRUE:
hasStrat(strat)
# FALSE:
hasStrat(cloud)
```

---

ichimoku *ichimoku*

Description

Create an ichimoku object containing values for all components of the Ichimoku Kinko Hyo cloud chart. The object encapsulates a date-time index, OHLC pricing data, candle direction, the cloud lines Tenkan-sen, Kijun-sen, Senkou span A, Senkou span B and Chikou span, as well as values for the cloud top and cloud base.

Usage

```
ichimoku(x, ticker, periods = c(9L, 26L, 52L), keep.data = FALSE, ...)

## S3 method for class 'ichimoku'
ichimoku(x, ticker, periods = c(9L, 26L, 52L), keep.data = FALSE, ...)

## S3 method for class 'xts'
ichimoku(x, ticker, periods = c(9L, 26L, 52L), keep.data = FALSE, ...)

## S3 method for class 'data.frame'
ichimoku(x, ticker, periods = c(9L, 26L, 52L), keep.data = FALSE, ...)

## S3 method for class 'matrix'
ichimoku(x, ticker, periods = c(9L, 26L, 52L), keep.data = FALSE, ...)

## Default S3 method:
ichimoku(x, ticker, periods = c(9L, 26L, 52L), keep.data = FALSE, ...)
```

Arguments

x	a data.frame or other compatible object, which includes xts, data.table, tibble, and matrix.
ticker	(optional) specify a ticker to identify the instrument, otherwise this is set to the name of the input object.
periods	[default c(9L, 26L, 52L)] a vector defining the length of periods used for the cloud. This parameter shoud not normally be modified as using other values would be invalid in the context of traditional ichimoku analysis.

keep.data	[default FALSE] set to TRUE to retain additional data present in the input object as additional columns and/or attributes.
...	additional arguments, for instance 'holidays', passed along to tradingDays for calculating the future cloud on daily data.

Details

Calling an ichimoku object automatically invokes its print method, which by default produces a printout of the data to the console as well as a plot of the cloud chart to the graphical device.

For further options, use plot() on the returned ichimoku object to pass further arguments for customising the chart. Use iplot() for interactive charting.

Where an ichimoku object is passed to ichimoku(), the ichimoku object is re-calculated using the OHLC pricing data contained within.

Value

An ichimoku object with S3 classes of 'ichimoku', 'xts' and 'zoo'.

Ichimoku Object Specification

Index:

- index(object): date-time index [POSIXct]

Columns [numeric]:

- object$open: opening price
- $high: high price
- $low: low price
- $close: closing price
- $cd: candle direction (-1 = down, 0 = flat, 1 = up)
- $tenkan: Tenkan-sen
- $kijun: Kijun-sen
- $senkouA: Senkou span A
- $senkouB: Senkou span B
- $chikou: Chikou span
- $cloudT: cloud Top (max of senkouA, senkouB)
- $cloudB: cloud Base (min of senkouA, senkouB)

Attributes:

- attributes(object)$periods: parameters used to calculate the cloud [integer vector of length 3]
- $periodicity: periodicity of the data in seconds [numeric]
- $ticker: instrument identifier [character]

Further Details

`ichimoku()` requires OHLC (or else HLC) price data as input to calculate the cloud chart values.

If only single series price data is supplied, a *pseudo* OHLC series is generated and a *pseudo* cloud chart is returned.

A faster technical utility version of this function is available in `.ichimoku` for use when the data is already in the required format.

Please refer to the reference vignette by calling: `vignette("reference", package = "ichimoku")`

Examples

```
TKR <- sample_ohlc_data

cloud <- ichimoku(TKR)
cloud

kumo <- ichimoku(TKR, ticker = "TKR Co.", periods = c(9, 26, 52), keep.data = TRUE)
summary(kumo)
print(kumo, plot = FALSE, rows = 10)
plot(kumo, theme = "solarized", type = "bar", custom = "volume")
```

---

index *Extract the Index of Ichimoku Objects*

Description

Method for extracting the date-time index of ichimoku objects.

Usage

```
## S3 method for class 'ichimoku'
index(x, subset, ...)
```

Arguments

`x`	an object of class 'ichimoku'.
`subset`	an integer or logical value or vector by which to subset the index.
`...`	arguments passed to or from other methods.

Details

This function is an S3 method for the generic function index() for class 'ichimoku'. It can be invoked by calling index(x) on an object 'x' of class 'ichimoku'.

Subsetting by specifying the 'subset' parameter subsets using the numerical values underlying the POSIXct times and results in a faster operation than usual subset operators such as '['.

Value

The date-time index of the ichimoku object as a vector of POSIXct values.

Examples

```
cloud <- ichimoku(sample_ohlc_data)
index(cloud)[101:110]
index(cloud, 101:110)
```

`iplot` *Interactive Ichimoku Cloud Plot*

Description

Plot Ichimoku Kinko Hyo cloud charts from ichimoku objects in R Shiny, allowing full customisation of chart elements in an interactive environment. Intuitive cursor infotip provides ready access to the data directly from the chart.

Usage

```
iplot(
  x,
  ticker,
  subtitle,
  theme = c("classic", "dark", "mono", "noguchi", "okabe-ito", "solarized"),
  strat = TRUE,
  type = c("none", "r", "s", "bar", "line"),
  custom,
  ...,
  launch.browser = TRUE
)
```

Arguments

`x`	an object of class 'ichimoku'.
`ticker`	(optional) specify a ticker (or other text) to include in the chart heading. If not set, the ticker saved within the ichimoku object will be used.
`subtitle`	(optional) specify a subtitle to display under the chart title.
`theme`	[default 'classic'] with further choices of 'dark', 'mono', 'noguchi', 'okabe-ito' or 'solarized'.
`strat`	[default TRUE] if the ichimoku object contains a strategy, the periods for which the strategy results in a position will be shaded, and the strategy printed as the chart subtitle (if not otherwise specified). Set to FALSE to turn off this behaviour.

type	[default 'none'] type of sub-plot to display beneath the ichimoku cloud chart, with a choice of 'none', 'r' or 's' for the corresponding oscillator type, and 'bar' or 'line' for custom plots.
custom	(optional) character string (containing a regular expression) matching the column name of the variable to be displayed as sub-plot. Specify `type = 'bar'` or `type = 'line'`, otherwise other type settings will take precedence.
...	additional parameters passed along to the 'options' argument of `shiny::shinyApp()`.
launch.browser	[default TRUE] If TRUE, the system's default web browser will be launched automatically after the app is started. The value of this argument can also be a function to call with the application's URL. To use the default Shiny viewer in RStudio, please specify `getOption("shiny.launch.browser")`.

Details

For further details please refer to the reference vignette by calling: `vignette("reference", package = "ichimoku")`

Value

A Shiny app object with class 'shiny.appobj'. With default arguments, the Shiny app is launched in the default browser.

Examples

```
if (interactive()) {
# Only run examples in interactive R sessions
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
iplot(cloud)

# To open in RStudio viewer instead of default browser
iplot(cloud, launch.browser = getOption("shiny.launch.browser"))
}
```

`is.ichimoku` *is.ichimoku*

Description

A function for checking if an object is an ichimoku object.

Usage

```
is.ichimoku(x)
```

Arguments

x	an object.

Value

A logical value of TRUE if 'x' is of class 'ichimoku', otherwise FALSE.

Examples

```
cloud <- ichimoku(sample_ohlc_data)

# TRUE:
is.ichimoku(cloud)
# FALSE:
is.ichimoku(sample_ohlc_data)
```

---

look *Look at Informational Attributes*

Description

Inspect the informational attributes of objects.

Usage

```
look(x = .Last.value)
```

Arguments

x	an object (optional). If 'x' is not supplied, `.Last.value` will be used instead.

Details

Note: autostrat list attributes may be accessed directly using `look(x)$logret` and `look(x)$summary`.

Value

For objects created by the ichimoku package, a pairlist of attributes specific to that data type.

For other objects, a pairlist of non-standard attributes for matrix / data.frame / xts classes, or else invisible NULL if none are present.

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
look(cloud)

stratlist <- autostrat(cloud, n = 3)
look(stratlist)

strat <- stratlist[[1]]
look(strat)
```

```
grid <- mlgrid(cloud)
look(grid)

## Not run:
# OANDA API key required to run this example
prices <- oanda("USD_JPY")
look(prices)

## End(Not run)
```

`matrix_df` *Convert matrix to data.frame*

Description

An optimised 'matrix' to 'data.frame' constructor.

Usage

```
matrix_df(x, keep.attrs = FALSE)
```

Arguments

`x`	a matrix.
`keep.attrs`	[default FALSE] if set to TRUE, will preserve any custom attributes set on the original object.

Details

The optimised data.frame constructors are used internally within the package and made available as utilities. Please note that no data validation or checking is performed.

Value

A 'data.frame' object. If the matrix has row names, these are retained by the dataframe.

Examples

```
cloud <- ichimoku(sample_ohlc_data)
mcloud <- as.matrix(cloud)
df <- matrix_df(mcloud)
str(df)
str(rownames(df))
```

mlgrid *mlgrid Numeric Representation*

Description

Create a grid of ichimoku indicator conditions and next period returns. The grid facilitates the comparison of strategy returns and provides a basis for use in machine learning applications. Translates the visual representation of the relationship between cloud chart elements into a numerical format for further analysis.

Usage

```
mlgrid(
  x,
  y = c("logret", "ret", "none"),
  k = 1L,
  dir = c("long", "short"),
  type = c("boolean", "numeric", "z-score"),
  format = c("dataframe", "matrix"),
  unique = TRUE,
  expr = list()
)
```

Arguments

x	an ichimoku object.
y	[default 'logret'] choose target variable 'logret' (log returns), 'ret' (discrete returns), or 'none'.
k	[default 1L] number of periods time horizon over which to calculate target variable 'y'.
dir	[default 'long'] trade direction, either 'long' or 'short'.
type	[default 'boolean'] choose 'boolean', 'numeric' or 'z-score'. 'boolean' creates a grid of dummy variables for ichimoku indicator conditions of the form 1 if c1 > c2, 0 otherwise. 'numeric' creates a grid of the numeric difference c1 - c2. 'z-score' standardises the numeric grid by the mean and standard deviation of each feature.
format	[default 'dataframe'] select either 'dataframe' or 'matrix' for the format of returned object.
unique	[default TRUE] to return only unique combinations of c1 and c2. Set to FALSE to return both c1 > c2 and c2 > c1.
expr	[default list()] (for advanced use only) a named list of quoted language objects or expressions, which are evaluated internally within the function, referencing intermediate objects such as: 'core', the coredata matrix of the ichimoku object, 'xlen' the number of observations, or any of the function parameters etc. Each evaluated expression must return a vector of length 'xlen' for inclusion in the grid.

Details

The date-time index corresponds to when the indicator condition is met at the close for that period. The return is the k-period return achieved by transacting at the immediately following opening price until the next opening price.

Only valid combinations are included. This excludes any combination involving 'open' as it is in effect a lagged indicator and not contemporaneous. The following trivial or highly-collinear pairs are also excluded: (high, close), (low, close), (low, high), (cloudTop, Senkou A), (cloudBase, senkou A), (cloudTop, senkouB), (cloudBase, senkouB), (cloudBase, cloudTop).

Value

A data.frame or matrix in a 'tidy' format with one observation per row and one feature per column with the target 'y' as the first column (unless set to 'none').

The 'y' and 'k' parameters, trade direction and grid type are set as attributes, with 'means' and 'sdevs' also populated for type 'z-score' to return the mean and standard deviation for each column. To view these, use `look` on the returned object.

Further Details

Please refer to the strategies vignette by calling: `vignette("strategies", package = "ichimoku")`

See Also

`autostrat` which uses `mlgrid()` to enumerate all valid return combinations.

`relative` which uses `mlgrid()` to relate the latest observed numeric representation to historical values.

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
grid <- mlgrid(cloud, y = "ret", k = 2, dir = "short", type = "z-score")
str(grid)

custom <- mlgrid(cloud, type = "numeric", expr = list(cd = quote(core[, "cd"])))
str(custom)
```

`more` *Print More Rows of Ichimoku Objects*

Description

After calling or invoking the default print method for ichimoku objects, the console output will display `-- omitted x rows` if the entire data does not fit on-screen. Use `more()` to display more rows.

Usage

```
more(rows)
```

Arguments

rows	(optional) specify the number of rows to print; defaults to all rows if not supplied or non-numeric.

Value

The ichimoku object contained in `.Last.value` (invisibly) or else invisible NULL (if .Last.value is not an ichimoku object). The ichimoku object data is printed to the console.

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
cloud
more(25)
more()
```

---

oanda *OANDA Price Data*

Description

Retrieve price data for major currencies, metals, commodities, government bonds and stock indices from the OANDA fxTrade API.

Usage

```
oanda(
  instrument,
 granularity = c("D", "W", "M", "H12", "H8", "H6", "H4", "H3", "H2", "H1", "M30", "M15",
    "M10", "M5", "M4", "M2", "M1", "S30", "S15", "S10", "S5"),
  count = NULL,
  from = NULL,
  to = NULL,
  price = c("M", "B", "A"),
  server,
  apikey,
  quietly
)
```

Arguments

instrument	string containing the base currency and quote currency delimited by '_' or '-' (e.g. "USD_JPY" or "usd-jpy"). Use the oanda_instruments function to return a list of all valid instruments.
granularity	[default "D"] the granularity of the price data to fetch, one of "M", "W", "D", "H12", "H8", "H6", "H4", "H3", "H2", "H1", "M30", "M15", "M10", "M5", "M4", "M2", "M1", "S30", "S15", "S10", "S5".
count	(optional) the number of periods to return. The API supports a maximum of 5000 for each individual request, and defaults to 500 if not specified. If both 'from' and 'to' are specified, 'count' is ignored, as the time range combined with 'granularity' will determine the number of periods to return.
from	(optional) the start of the time range for which to fetch price data, in a format convertible to POSIXct by as.POSIXct(), for example "2020-02-01".
to	(optional) the end of the time range for which to fetch price data, in a format convertible to POSIXct by as.POSIXct(), for example "2020-06-30".
price	[default "M"] pricing component, one of "M" (midpoint), "B" (bid) or "A" (ask).
server	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by oanda_switch.
apikey	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable OANDA_API_KEY or by oanda_set_key. Can also be entered interactively if not specified.
quietly	(optional) if set to TRUE, will suppress printing of auxiliary output to the console and return quietly.

Details

This function queries the OANDA fxTrade API.

Requires an fxTrade account with OANDA https://www.oanda.com/forex-trading/. If you do not already hold a live account, you may register for an OANDA fxTrade practice / demo account. There is a link on your OANDA fxTrade account profile page 'Manage API Access' (My Account -> My Services -> Manage API Access). From there, a personal access token to use with the OANDA API can be generated, as well as revoked.

The oanda_set_key function can be used to save the API key in the system credential store so that it is automatically recognised in future (requires the 'keyring' package to be installed).

Value

A data.frame containing the price data requested.

Further Details

Please refer to the OANDA fxTrade API vignette by calling: vignette("xoanda", package = "ichimoku").

'OANDA' and 'fxTrade' are trademarks owned by OANDA Corporation, an entity unaffiliated with the ichimoku package.

Examples

```
## Not run:
# OANDA fxTrade API key required to run these examples
prices <- oanda("USD_JPY")
ichimoku(prices)

oanda("EUR_JPY", granularity = "H1", count = 250, from = "2020-02-01", price = "B")

## End(Not run)
```

---

oanda_chart *OANDA Real-time Cloud Charts*

Description

Plot real-time Ichimoku Kinko Hyo cloud charts for major currencies, metals, commodities, government bonds and stock indices using OANDA fxTrade API data.

Usage

```
oanda_chart(
  instrument,
  granularity = c("D", "W", "M", "H12", "H8", "H6", "H4", "H3", "H2", "H1", "M30", "M15",
    "M10", "M5", "M4", "M2", "M1", "S30", "S15", "S10", "S5"),
  refresh = 5,
  count = 250,
  price = c("M", "B", "A"),
  theme = c("classic", "dark", "mono", "noguchi", "okabe-ito", "solarized"),
  type = c("none", "r", "s"),
  limit,
  server,
  apikey,
  ...,
  periods = c(9L, 26L, 52L)
)
```

Arguments

instrument	string containing the base currency and quote currency delimited by '_' or '-' (e.g. "USD_JPY" or "usd-jpy"). Use the oanda_instruments function to return a list of all valid instruments.
granularity	[default "D"] the granularity of the price data to fetch, one of "M", "W", "D", "H12", "H8", "H6", "H4", "H3", "H2", "H1", "M30", "M15", "M10", "M5", "M4", "M2", "M1", "S30", "S15", "S10", "S5".
refresh	[default 5] data refresh interval in seconds, with a minimum of 1.

count	[default 250] the number of periods to return. The API supports a maximum of 5000. Note that fewer periods are actually shown on the chart to ensure a full cloud is always displayed.
price	[default "M"] pricing component, one of "M" (midpoint), "B" (bid) or "A" (ask).
theme	[default 'classic'] with further choices of 'dark', 'mono', 'noguchi', 'okabe-ito' or 'solarized'. Alternatively, supply a vector of 12 colour values (hex codes or names) as a user-defined theme.
type	[default 'none'] type of sub-plot to display beneath the ichimoku cloud chart, with a choice of 'none', 'r' or 's' for the corresponding oscillator type.
limit	(optional) specify a time in seconds by which to limit the session. The session will end with data returned automatically after the specified time has elapsed.
server	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by oanda_switch.
apikey	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable OANDA_API_KEY or by oanda_set_key. Can also be entered interactively if not specified.
...	additional arguments passed along to ichimoku for calculating the ichimoku cloud or autoplot to set chart parameters.
periods	[default c(9L, 26L, 52L)] a vector defining the length of periods used for the cloud. This parameter shoud not normally be modified as using other values would be invalid in the context of traditional ichimoku analysis.

Details

This function polls the OANDA fxTrade API for the latest live prices and updates the plot in the graphical device at each refresh interval. Use the 'Esc' key to stop updating.

To access the underlying data, assign the function to an object, for example: `cloud <- oanda_chart("USD_JPY")`.

Value

The ichimoku object underlying the chart (invisibly) on function exit. A plot of the ichimoku chart for the price data requested is output to the graphical device at each refresh interval.

Further Details

Please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Examples

```
## Not run:
# OANDA fxTrade API key required to run these examples
oanda_chart("USD_JPY")
oanda_chart("EUR_JPY", granularity = "H1", refresh = 3, count = 300, price = "B", theme = "mono")
```

```
# Save data underlying chart at time of function exit
cloud <- oanda_chart("USD_JPY")

## End(Not run)
```

`oanda_instruments` *Available OANDA Instruments*

Description

Return list of instruments including major currencies, metals, commodities, government bonds and stock indices for which pricing data is available from the OANDA fxTrade API.

Usage

```
oanda_instruments(server, apikey)
```

Arguments

`server`	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by `oanda_switch`.
`apikey`	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable `OANDA_API_KEY` or by `oanda_set_key`. Can also be entered interactively if not specified.

Details

This function returns a data.frame listing the instrument names available for an account associated with the supplied OANDA fxTrade API key.

For further details please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Value

A data.frame containing the instrument name, full display name, and type.

Examples

```
## Not run:
# OANDA fxTrade API key required to run this example
oanda_instruments()

## End(Not run)
```

oanda_orders *OANDA Order Book*

Description

Provides a summary of the aggregate orders posted by OANDA fxTrade clients at each price level.

Usage

```
oanda_orders(instrument, time, server, apikey)
```

Arguments

instrument	string containing the base currency and quote currency delimited by '_' or '-' (e.g. "USD_JPY" or "usd-jpy"). Use the oanda_instruments function to return a list of all valid instruments.
time	(optional) the time for which to retrieve the order book, in a format convertible to POSIXct by as.POSIXct(). If not specified, the most recent order book will be retrieved.
server	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by oanda_switch.
apikey	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable OANDA_API_KEY or by oanda_set_key. Can also be entered interactively if not specified.

Details

This feature has been implemented by OANDA only for certain major currency pairs and should be considered experimental.

Note: as certain orders are placed far from the market price, only the interquartile range of order levels is shown on the chart. The returned data frame does however contain the entire order book.

For further details please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Value

Invisibly, a data frame of the order book with parameters saved as attributes. A chart showing the percentage long and short orders at each price level is output to the graphical device.

Examples

```
## Not run:
# OANDA fxTrade API key required to run this example
oanda_orders("USD_JPY")

## End(Not run)
```

`oanda_positions` *OANDA Position Book*

Description

Provides a summary of the aggregate positions held by OANDA fxTrade clients at each price level.

Usage

```
oanda_positions(instrument, time, server, apikey)
```

Arguments

`instrument`	string containing the base currency and quote currency delimited by '_' or '-' (e.g. "USD_JPY" or "usd-jpy"). Use the `oanda_instruments` function to return a list of all valid instruments.
`time`	(optional) the time for which to retrieve the position book, in a format convertible to POSIXct by `as.POSIXct()`. If not specified, the most recent position book will be retrieved.
`server`	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by `oanda_switch`.
`apikey`	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable `OANDA_API_KEY` or by `oanda_set_key`. Can also be entered interactively if not specified.

Details

This feature has been implemented by OANDA only for certain major currency pairs and should be considered experimental.

For further details please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Value

Invisibly, a data frame of the position book with parameters saved as attributes. A chart showing the percentage long and short positions at each price level is output to the graphical device.

Examples

```
## Not run:
# OANDA fxTrade API key required to run this example
oanda_positions("USD_JPY")

## End(Not run)
```

`oanda_quote` *OANDA Quote Latest Price*

Description

Provides a single line price quote for an instrument.

Usage

```
oanda_quote(instrument, price = c("M", "B", "A"), server, apikey)
```

Arguments

`instrument`	string containing the base currency and quote currency delimited by '_' or '-' (e.g. "USD_JPY" or "usd-jpy"). Use the `oanda_instruments` function to return a list of all valid instruments.
`price`	[default "M"] pricing component, one of "M" (midpoint), "B" (bid) or "A" (ask).
`server`	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by `oanda_switch`.
`apikey`	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable `OANDA_API_KEY` or by `oanda_set_key`. Can also be entered interactively if not specified.

Details

This function is designed for interactive use.

For further details please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Value

Invisible NULL. The instrument, timestamp, daily open, high, low and last prices, percentage change from the open, and the pricing component (M for mid, B for bid, A for ask) is output to the console.

Examples

```
## Not run:
# OANDA fxTrade API key required to run this example
oanda_quote("USD_JPY")

## End(Not run)
```

`oanda_set_key` *Set OANDA fxTrade API Key*

Description

Save OANDA fxTrade API key (personal access token) to the system credential store.

Usage

```
oanda_set_key()
```

Details

The key is read interactively. Separate keys can be set for practice and live accounts - please choose the correct account type when prompted.

This function only needs to be called once to set the key; it does not need to be called each session.

This function has a dependency on the 'keyring' package.

Value

Invisible NULL. A key is set in the default keyring under the service name 'OANDA_API_KEY' for practice accounts or 'OANDA_LIVE_KEY' for live accounts.

Further Details

Please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Examples

```
if (interactive()) {
# Only run example in interactive R sessions
oanda_set_key()
}
```

oanda_stream *OANDA Streaming Data*

Description

Stream live price and liquidity data for major currencies, metals, commodities, government bonds and stock indices from the OANDA fxTrade Streaming API.

Usage

```
oanda_stream(instrument, display = 8L, limit, server, apikey)
```

Arguments

instrument	string containing the base currency and quote currency delimited by '_' or '-' (e.g. "USD_JPY" or "usd-jpy"). Use the oanda_instruments function to return a list of all valid instruments.
display	[default 8L] integer rows of data to display in the console at any one time.
limit	(optional) specify a time in seconds by which to limit the streaming session. The session will end with data returned automatically after the specified time has elapsed.
server	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by oanda_switch.
apikey	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable OANDA_API_KEY or by oanda_set_key. Can also be entered interactively if not specified.

Details

This function connects to the OANDA fxTrade Streaming API. Send an interrupt using the 'Esc' key or 'Ctrl+c' to stop the stream and return the session data.

Note: only messages of type 'PRICE' are processed. Messages of type 'HEARTBEAT' consisting of only a timestamp are discarded.

Value

Returned invisibly, a dataframe containing the data for the streaming session on function exit. The latest rows of the dataframe are printed to the console, as governed by the 'display' argument.

Streaming Data

Summarised from the streaming API documentation:

- Pricing stream does not include every single price created for the Account
- At most 4 prices are sent per second (every 250 milliseconds) for each instrument
- If more than one price is created during the 250 millisecond window, only the price in effect at the end of the window is sent
- This means that during periods of rapid price movement, not every price is sent
- Pricing windows for different connections to the stream are not all aligned in the same way (e.g. to the top of the second)
- This means that during periods of rapid price movement, different prices may be observed depending on the alignment for the connection

Further Details

Please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Examples

```
## Not run:
# OANDA fxTrade API key required to run this example
data <- oanda_stream("USD_JPY", display = 8L)

## End(Not run)
```

---

oanda_studio *OANDA Studio Interactive Live Analysis*

Description

Interactive and fully-customisable R Shiny environment providing real-time Ichimoku Kinko Hyo cloud charts for major currencies, metals, commodities, government bonds and stock indices using OANDA fxTrade API data. Intuitive cursor infotip provides ready access to the data directly from the chart.

Usage

```
oanda_studio(
  instrument = "USD_JPY",
 granularity = c("D", "W", "M", "H12", "H8", "H6", "H4", "H3", "H2", "H1", "M30", "M15",
    "M10", "M5", "M4", "M2", "M1", "S30", "S15", "S10", "S5"),
  refresh = 5,
  count = 300,
  price = c("M", "B", "A"),
```

```
  theme = c("classic", "dark", "mono", "noguchi", "okabe-ito", "solarized"),
  type = c("none", "r", "s"),
  server,
  apikey,
  new.process = FALSE,
  ...,
  launch.browser = TRUE,
  periods = c(9L, 26L, 52L)
)
```

Arguments

instrument	[default 'USD_JPY'] string containing the base currency and quote currency delimited by a '_'. Use the oanda_instruments function to return a list of all valid instruments.
granularity	[default "D"] the granularity of the price data to fetch, one of "M", "W", "D", "H12", "H8", "H6", "H4", "H3", "H2", "H1", "M30", "M15", "M10", "M5", "M4", "M2", "M1", "S30", "S15", "S10", "S5".
refresh	[default 5] data refresh interval in seconds, with a minimum of 1.
count	[default 300] the number of periods to return, from 100 to 800. Note that fewer periods are actually shown on the chart to ensure a full cloud is always displayed.
price	[default "M"] pricing component, one of "M" (midpoint), "B" (bid) or "A" (ask).
theme	[default 'original'] with alternative choices of 'conceptual', 'dark', 'fresh', 'mono', or 'solarized'.
type	[default 'none'] type of sub-plot to display beneath the ichimoku cloud chart, with a choice of 'none', 'r' or 's' for the corresponding oscillator type.
server	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by oanda_switch.
apikey	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable OANDA_API_KEY or by oanda_set_key. Can also be entered interactively if not specified.
new.process	[default FALSE] if TRUE, will start the shiny session in a new R process, unblocking the current process and allowing continued use of the R console.
...	additional arguments passed along to ichimoku for calculating the ichimoku cloud, autoplot to set chart parameters, or the 'options' argument of shiny::shinyApp().
launch.browser	[default TRUE] If TRUE, the system's default web browser will be launched automatically after the app is started. The value of this argument can also be a function to call with the application's URL. To use the default Shiny viewer in RStudio, please specify getOption("shiny.launch.browser").
periods	[default c(9L, 26L, 52L)] a vector defining the length of periods used for the cloud. This parameter shoud not normally be modified as using other values would be invalid in the context of traditional ichimoku analysis.

Details

This function polls the OANDA fxTrade API for the latest prices and updates a customisable reactive Shiny app at each refresh interval.

Value

Invisible NULL, or a 'mirai' if 'new.process' is specified as TRUE. With default arguments, a Shiny app is launched in the default browser.

Further Details

Please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Examples

```
## Not run:
# OANDA fxTrade API key required to run these examples
oanda_studio()

# To open in RStudio viewer instead of default browser
oanda_studio(launch.browser = getOption("shiny.launch.browser"))

## End(Not run)
```

---

oanda_switch *Switch Default OANDA Server*

Description

Switch the default OANDA fxTrade server from 'practice' to 'live' or vice versa. Settings persist for the current session only.

Usage

```
oanda_switch()
```

Details

The default server at the start of a new session is the practice server. Call this function to switch to the live server.

This function can be used to toggle between the practice and live servers. Any cached variables for API key, account or instruments list are cleared each time this function is called.

For further details please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Value

Invisible NULL. A message informs the resulting default server setting.

Examples

```
oanda_switch()
oanda_switch()
```

`oanda_view` *OANDA View Market Performance*

Description

Provides a snapshot overview of markets on an intraday basis, showing the relative performance of individual constituents.

Usage

```
oanda_view(
  market = c("allfx", "bonds", "commodities", "fx", "metals", "stocks"),
  price = c("M", "B", "A"),
  server,
  apikey
)
```

Arguments

`market`	string specifying the market: 'allfx' for all available currencies, 'bonds' for government bonds, 'commodities' for commodities, 'fx' for major currency pairs, 'metals' for metals and 'stocks' for global stock markets.
`price`	[default "M"] pricing component, one of "M" (midpoint), "B" (bid) or "A" (ask).
`server`	(optional) specify the "practice" or "live" server according to the account type held. If not specified, will default to "practice", unless this has been changed by `oanda_switch`.
`apikey`	(optional) string containing the OANDA fxTrade API key (personal access token), or function that returns this string. Does not need to be specified if already stored as the environment variable `OANDA_API_KEY` or by `oanda_set_key`. Can also be entered interactively if not specified.

Details

This function is designed for interactive use.

For further details please refer to the OANDA fxTrade API vignette by calling: `vignette("xoanda", package = "ichimoku")`.

Value

A data.frame containing the daily open, high, low and last prices, along with the percentage price change from the open, ordered by the percentage change. The instrument names are set as row names.

The first timestamp retrieved and the pricing component are printed to the console as well as saved as attributes to the dataframe. The dataframe is also printed to the console.

Examples

```
## Not run:
# OANDA fxTrade API key required to run this example
oanda_view("fx")

## End(Not run)
```

`plot.ichimoku` *Plot Ichimoku Cloud Chart*

Description

Plot Ichimoku Kinko Hyo cloud charts from ichimoku objects.

Usage

```
## S3 method for class 'ichimoku'
plot(
  x,
  window,
  ticker,
  subtitle,
  theme = c("classic", "dark", "mono", "noguchi", "okabe-ito", "solarized"),
  strat = TRUE,
  type = c("none", "r", "s", "bar", "line"),
  custom,
  ...
)
```

Arguments

`x`	an object of class 'ichimoku'.
`window`	(optional) a date-time window to subset the plot, in ISO-8601 compatible range strings of the format used for 'xts' objects, for example "2020-02-15/2020-08-15" or "2020-02-15/", "/2020-08" or "2020-07".
`ticker`	(optional) specify a ticker (or other text) to include in the chart heading. If not set, the ticker saved within the ichimoku object will be used.

subtitle	(optional) specify a subtitle to display under the chart title.
theme	[default 'classic'] with further choices of 'dark', 'mono', 'noguchi', 'okabe-ito' or 'solarized'. Alternatively, supply a vector of 12 colour values (hex codes or names) as a user-defined theme.
strat	[default TRUE] if the ichimoku object contains a strategy, the periods for which the strategy results in a position will be shaded, and the strategy printed as the chart subtitle (if not otherwise specified). Set to FALSE to turn off this behaviour.
type	[default 'none'] type of sub-plot to display beneath the ichimoku cloud chart, with a choice of 'none', 'r' or 's' for the corresponding oscillator type, and 'bar' or 'line' for custom plots.
custom	(optional) character string (containing a regular expression) matching the column name of the variable to be displayed as sub-plot. Specify `type = 'bar'` or `type = 'line'`, otherwise other type settings will take precedence.
...	additional arguments passed along to the print method for 'ggplot' objects.

Details

This function is an S3 method for the generic function plot() for class 'ichimoku'. It can be invoked by calling plot(x) on an object 'x' of class 'ichimoku'.

For further details please refer to the reference vignette by calling: `vignette("reference", package = "ichimoku")`

Value

The ichimoku object supplied (invisibly). The requested plot is output to the graphical device.

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
plot(cloud)
plot(cloud, window = "2020-05-01/2020-12-01", theme = "dark")
plot(cloud, window = "2020-05/", ticker = "TKR Co.", theme = "noguchi", type = "s")
plot(cloud, window = "/2020-11-02", subtitle = "Sample Price Data", theme = "mono", type = "r")

kumo <- ichimoku(sample_ohlc_data, ticker = "TKR", keep.data = TRUE)
plot(kumo, window = "2020-05/", theme = "solarized", type = "bar", custom = "volume")
plot(kumo, window = "2020-05/", theme = "okabe-ito", type = "line", custom = "volume")
```

---

print.ichimoku *Print Ichimoku Objects*

Description

Default print method for ichimoku objects to enable automatic plotting of the ichimoku cloud chart.

Usage

```
## S3 method for class 'ichimoku'
print(x, plot = TRUE, rows = 26L, ...)
```

Arguments

x	an object of class 'ichimoku'.
plot	[default TRUE] set to FALSE to prevent automatic plotting of the ichimoku cloud chart.
rows	[default 26L] integer number of rows to print.
...	additional arguments passed along to the xts print and plot.ichimoku methods.

Details

This function is an S3 method for the generic function print() for class 'ichimoku'. It can be invoked by calling print(x) on an object 'x' of class 'ichimoku'.

Value

The ichimoku object supplied (invisibly). The data is printed to the console. The ichimoku cloud chart is also output to the graphical device depending on the parameters set.

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")

print(cloud)
print(cloud, plot = FALSE, rows = 20L)
```

---

relative *Relative Numeric Representation*

Description

Produce a statistical summary of the latest numeric representation of the ichimoku cloud chart relative to historical values. For determining whether current trading falls within or outside of normal ranges.

Usage

```
relative(x, order = FALSE, signif = 0.2, quietly)
```

Arguments

x	an ichimoku object.
order	[default FALSE] set to TRUE to order the results by the absolute 'z-score'.
signif	[default 0.2] set a significance threshold for which if 'p' is equal or lower, the element will be starred with a '*'.
quietly	(optional) if set to TRUE, will suppress printing of additional output to the console and return quietly.

Details

'mean(X)' is the mean value for each element X, 'sd(X)' the standard deviation, and 'X[n]' the nth or latest observed values.

'res' is the residual X[n] - mean(X) and represents a centred measure of deviation for the latest observed value.

The 'z-score' (or standard score) is calculated as res / sd(X) and is a centred and scaled measure of deviation for the latest observed value.

'p >= |z|' represents the empirical probability of the latest observed absolute 'z-score' or greater.

'p*' will display a star if 'p >= |z|' is less than or equal to the value of the argument 'signif'.

'E(|res|)|p' represents the mean or expected absolute value of 'res', conditional upon the absolute 'z-score' being greater than equal to the latest observed absolute 'z-score'.

Value

A data frame containing a statistical summary of the latest ichimoku cloud chart representation in relation to historical values.

In addition, the time index of the latest observed values and total number of datapoints are printed to the console.

Further Details

Please refer to the strategies vignette by calling: `vignette("strategies", package = "ichimoku")`

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
statistics <- relative(cloud, quietly = TRUE)
relative(cloud, signif = 0.4)
relative(cloud, order = TRUE, signif = 0.4)
```

sample_ohlc_data *Sample OHLC Price Data*

Description

Simulated prices for a hypothetical financial asset. Created for the purpose of demonstrating package functions in examples and vignettes only.

Usage

```
sample_ohlc_data
```

Format

A data frame with 256 observations of 6 variables:

- time - timestamp of observation [POSIXct]
- open - opening price [numeric]
- low - low price [numeric]
- high - high price [numeric]
- close - closing price [numeric]
- volume - volume [integer]

Source

Not applicable: simulated data

Examples

```
head(sample_ohlc_data)
```

str.ichimoku *Display the Structure of Ichimoku Objects*

Description

Compactly display the internal structure of ichimoku objects.

Usage

```
## S3 method for class 'ichimoku'
str(object, ...)
```

Arguments

object	an object of class 'ichimoku'.
...	arguments passed to or from other methods.

Details

This function is an S3 method for the generic function str() for class 'ichimoku'. It can be invoked by calling str(x) on an object 'x' of class 'ichimoku'.

Value

Invisible NULL. A compact display of the structure of the object is output to the console.

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
str(cloud)

strat <- strat(cloud)
str(strat)
```

---

strat *Create Ichimoku Strategies*

Description

Create ichimoku cloud strategies using the indicator condition 'long / short while c1 > c2'. Complex strategies can be formulated as combined 'c1 > c2 & c3 > c4' (both conditions must be satisfied) or asymmetric 'c1 > c2 x c3 > c4' (where 'c1 > c2' denotes the entry and 'c3 > c4' the exit indicator).

Usage

```
strat(
  x,
  c1 = c("close", "chikou", "open", "high", "low", "tenkan", "kijun", "senkouA",
    "senkouB", "cloudT", "cloudB"),
 c2 = c("tenkan", "kijun", "senkouA", "senkouB", "cloudT", "cloudB", "chikou", "close",
    "open", "high", "low"),
 c3 = c("close", "chikou", "open", "high", "low", "tenkan", "kijun", "senkouA",
    "senkouB", "cloudT", "cloudB"),
 c4 = c("tenkan", "kijun", "senkouA", "senkouB", "cloudT", "cloudB", "chikou", "close",
    "open", "high", "low"),
  dir = c("long", "short"),
  type = 2
)
```

Arguments

x	an ichimoku object.
c1	[default 'close'] column name specified as a string.
c2	[default 'tenkan'] column name specified as a string.
c3	(optional) column name specified as a string.
c4	(optional) column name specified as a string.
dir	[default 'long'] trade direction, either 'long' or 'short'.
type	[default 2] if 'c3' and 'c4' are specified, type 2 will create the combined strategy 'c1 > c2 & c3 > c4' whilst type 3 will create the asymmetric strategy 'c1 > c2 x c3 > c4'.

Details

The following assumption applies to all strategies: confirmation of whether a condition is satisfied is received at the 'close' of a particular period, and a transaction is initiated at the immediately following 'open'. All transactions occur at the 'open'.

By default, the periods in which the strategy results in a position is shaded on the ichimoku cloud chart and the strategy is printed as the chart message (if not otherwise specified). To turn off this behaviour, pass the `strat = FALSE` argument to plot() or iplot().

Value

An ichimoku object augmented with the strategy.

Ichimoku Object Specification for Strategies

The ichimoku object is augmented with the following additional elements:

Columns [numeric]:

- `$cond`: a boolean vector if the indicator condition is met
- `$posn`: a boolean vector indicating if a position is held
- `$txn`: a vector representing the transactions to implement the position (1 = enter position, -1 = exit position)
- `$logret`: a vector of log returns
- `$slogret`: a vector of log returns for the strategy
- `$ret`: a vector of discrete returns
- `$sret`: a vector of of discrete returns for the strategy

Attributes:

- `$strat`: the strategy summary [matrix]

The strategy summary may be accessed by the `summary()` method for ichimoku objects or via `look`.

Complex Strategies

For complex strategies: 's1' denotes the strategy 'c1 > c2' and 's2' denotes the strategy 'c3 > c4'.

- Combined strategy 's1 & s2': indicator conditions in 's1' and 's2' have to be met simultaneously for a position to be taken. The column `$cond` will show when both conditions are met
- Asymmetric strategy 's1 x s2': indicator condition in 's1' has to be met to enter a position, and indicator condition in 's2' to exit a position. These rules are applied recursively over the length of the data. The column `$cond` will show when the indicator condition is met in 's1'

Further Details

Please refer to the strategies vignette by calling: `vignette("strategies", package = "ichimoku")`

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")

strat <- strat(cloud, c1 = "tenkan", c2 = "cloudB", dir = "short")
summary(strat)
plot(strat)

strat2 <- strat(cloud, c1 = "cloudT", c2 = "kijun", c3 = "cloudT", c4 = "close")
summary(strat2)
plot(strat2)
```

---

stratcombine *Combine Ichimoku Strategies*

Description

Create custom strategies from combining existing strategies contained in ichimoku objects 's1' and 's2' to form 's1 & s2'.

Usage

```
stratcombine(s1, s2)
```

Arguments

s1	an ichimoku object containing a strategy.
s2	an ichimoku object containing a strategy.

Details

The combined strategy 's1 & s2' means indicator conditions in 's1' and 's2' have to be met simulateneously for a trade position to be taken.

The boolean values showing whether these conditions are met are stored in the 'cond' column.

The strategy summary may be accessed by the `summary()` method for ichimoku objects or via `look`.

Value

An ichimoku object augmented with the combined strategy.

Further Details

Please refer to the strategies vignette by calling: `vignette("strategies", package = "ichimoku")`

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
strat1 <- strat(cloud, c1 = "close", c2 = "kijun")
strat2 <- strat(cloud, c1 = "cloudB", c2 = "tenkan")
cstrat <- stratcombine(strat1, strat2)
summary(cstrat)
plot(cstrat)
```

`summary.ichimoku` *Summary of Ichimoku Objects and Strategies*

Description

Display summary information for an ichimoku object or its strategy.

Usage

```
## S3 method for class 'ichimoku'
summary(object, strat = TRUE, ...)
```

Arguments

`object`	an object of class 'ichimoku'.
`strat`	[default TRUE] to show the strategy summary if present. Set to FALSE to show the object summary instead.
`...`	arguments passed to or from other methods.

Details

This function is an S3 method for the generic function summary() for class 'ichimoku'. It can be invoked by calling summary(x) on an object 'x' of class 'ichimoku'.

Performs basic validation for an ichimoku object and will inform if an ichimoku object contains invalid information.

Value

A matrix containing the strategy summary, if present and 'strat' is set to TRUE, otherwise a character vector containing an abbreviated object summary (the full object summary is output to the console).

Examples

```
cloud <- ichimoku(sample_ohlc_data, ticker = "TKR")
summary(cloud)

strat <- strat(cloud)
summary(strat)
```

---

tradingDays *Select Trading Days*

Description

Used by ichimoku to subset a vector of dates to trading days.

Usage

```
tradingDays(x, holidays, ...)
```

Arguments

x	a vector of POSIXct date objects.
holidays	(optional) a vector, or function which outputs a vector, of dates defined as holidays. Set to NULL for a continuously-traded market.
...	other arguments not used by this function.

Details

New Year's Day (01-01) and Christmas Day (12-25) are defined as holidays by default if 'holidays' is not specified.

Value

A vector of logical values: TRUE if the corresponding element of 'x' is a weekday and not a holiday, FALSE otherwise.

Or, if the parameter 'holidays' is set to NULL, a vector of TRUE values of the same length as 'x'.

Examples

```
dates <- seq(from = as.POSIXct("2020-01-01"), by = "1 day", length.out = 7)
dates
tradingDays(dates)
tradingDays(dates, holidays = c("2020-01-02", "2020-01-03"))
tradingDays(dates, holidays = NULL)
```

---

xts_df *Convert xts to data.frame*

Description

An optimised 'xts' to 'data.frame' constructor.

Usage

```
xts_df(x, keep.attrs = FALSE)
```

Arguments

x	an 'xts' object.
keep.attrs	[default FALSE] if set to TRUE, will preserve any custom attributes set on the original object.

Details

The optimised data.frame constructors are used internally within the package and made available as utilities. Please note that no data validation or checking is performed.

Value

A 'data.frame' object. The 'xts' index is preserved as the first column with header 'index'.

Examples

```
cloud <- ichimoku(sample_ohlc_data)
df <- xts_df(cloud)
str(df)

df2 <- xts_df(cloud, keep.attrs = TRUE)
str(df2)
```

Index