

Package ‘RTL’

April 7, 2021

Type Package

Title Risk Tool Library

Version 0.1.6

Date 2021-04-07

Description Collection of functions and metadata to complement core packages in Finance and Commodities, including futures expiry tables and <https://www.morningstar.com/products/commodities-and-energy> API functions. See <https://github.com/risktoollib/RTL>.

Depends R (>= 4.0)

License GPL (>= 3)

Encoding UTF-8

LazyData true

LazyDataCompression xz

URL <https://github.com/risktoollib/RTL>

Suggests plotly

RoxygenNote 7.1.1

Imports zoo, xts, stats, magrittr, tibble, dplyr, tidyr, ggplot2, httr, stringr, purrr, lubridate, timetk, PerformanceAnalytics, tibbletime, quantmod, forecast, tidyquant, readr, Quandl, fitdistrplus, tsibble, feasts, fabletools, jsonlite, sp, RCurl, rugarch, lpSolve, rlang

NeedsCompilation no

Author Philippe Cote [aut, cre],
Nima Safaian [aut]

Maintainer Philippe Cote <pcote@ualberta.ca>

Repository CRAN

Date/Publication 2021-04-07 13:10:10 UTC

R topics documented:

bond	3
cancrudeassays	4
cancrudeassayssum	4
cancrudeprices	5
chart_eia_sd	5
chart_eia_steo	6
chart_fwd_curves	7
chart_pairs	8
chart_PerfSummary	8
chart_spreads	9
chart_zscore	10
CRReuro	12
crudeassaysBP	13
crudeassaysXOM	13
crudepipelines	14
crudes	14
dflong	15
dfwide	15
df_fut	15
distdescplot	16
eia2tidy	16
eiaStocks	17
eiaStorageCap	18
expiry_table	18
fitOU	19
fizdiffs	19
garch	20
getCurve	20
getGenscapePipeOil	22
getGenscapeStorageOil	23
getIRswapCurve	24
getPrice	25
getPrices	27
holidaysOil	28
ir_df_us	28
lngterminals	29
nghubs	29
ngpipelines	30
ngstorage	30
npv	31
planets	32
productspipelines	32
productsterminals	33
promptBeta	33
ref.opt.inputs	34
ref.opt.outputs	34

refineries	35
refineryLP	35
returns	36
rolladjust	36
simGBM	37
simOU	38
simOUJ	39
stl_decomp	40
swapCOM	41
swapFutWeight	42
swapInfo	43
swapIRS	44
tickers_eia	45
tradeCycle	46
tradeprocess	46
tradeStats	46
usSwapCurves	47
usSwapCurvesPar	47
usSwapIR	48
usSwapIRdef	48

Index	49
--------------	-----------

bond	bond
------	------

Description

Compute bond price, cash flow table and duration

Usage

```
bond(ytm = 0.05, C = 0.05, T2M = 1, m = 2, output = "price")
```

Arguments

ytm	Yield to Maturity
C	Coupon rate per annum
T2M	Time to maturity in years
m	Periods per year for coupon payments e.g semi-annual = 2.
output	"price", "df" or "duration"

Value

Price, cash flows data frame and/or duration

Author(s)

Philippe Cote

Examples

```
bond(ytm = 0.05, C = 0.05, T2M = 1, m = 2, output = "price")  
bond(ytm = 0.05, C = 0.05, T2M = 1, m = 2, output = "df")  
bond(ytm = 0.05, C = 0.05, T2M = 1, m = 2, output = "duration")
```

cancrudeassays	<i>cancrudeassays</i>
----------------	-----------------------

Description

Data set with historical Canadian Crude Assays.

Usage

```
cancrudeassays
```

Format

data frame

Source

<https://crudemonitor.ca/>

cancrudeassayssum	<i>cancrudeassayssum</i>
-------------------	--------------------------

Description

Data set with historical Canadian Crude Assays Statistics.

Usage

```
cancrudeassayssum
```

Format

data frame

Source

<https://crudemonitor.ca/>

can crudeprices	<i>can crudeprices</i>
-----------------	------------------------

Description

Randomized dataset of Canadian Crude monthly prices versus WTi Calendar Month Average.

Usage

```
can crudeprices
```

Format

```
data frame
```

chart_eia_sd	chart_eia_sd
--------------	--------------

Description

Supply Demand Balance from EIA Short Term Energy Outlook.

Usage

```
chart_eia_sd(
  market = "mogas",
  key = "your EIA.gov API key",
  from = "2011-01-01",
  legend.pos = list(x = 0.4, y = 0.53),
  output = "chart"
)
```

Arguments

market	"mogas", "dist", "jet" or "resid".
key	Your private EIA API token.
from	Date as character "2020-07-01". Default to all dates available.
legend.pos	Defaults to list(x = 0.4, y = 0.53)
output	"chart" for plotly object or "data" for dataframe.

Value

A plotly object or a dataframe

Author(s)

Philippe Cote

Examples

```
## Not run:
chart_eia_sd(key = key, market = "mogas")

## End(Not run)
```

chart_eia_steo	chart_eia_steo
----------------	----------------

Description

Supply Demand Balance from EIA Short Term Energy Outlook.

Usage

```
chart_eia_steo(
  market = "globalOil",
  key = "your EIA.gov API key",
  from = "2018-07-01",
  fig.title = "EIA STEO Global Liquids SD Balance",
  fig.units = "million barrels per day",
  legend.pos = list(x = 0.4, y = 0.53),
  output = "chart"
)
```

Arguments

market	"globalOil" only currently implemented.
key	Your private EIA API token.
from	Date as character "2020-07-01". Default to all dates available.
fig.title	Defaults to "EIA STEO Global Liquids SD Balance".
fig.units	Defaults to "million barrels per day"
legend.pos	Defaults to list(x = 0.4, y = 0.53)
output	"chart" for plotly object or "data" for dataframe.

Value

A plotly object or a dataframe

Author(s)

Philippe Cote

Examples

```
## Not run:  
chart_eia_steo(key = EIAkey, market = "globalOil")  
  
## End(Not run)
```

chart_fwd_curves chart_fwd_curves

Description

Returns a plot of forward curves through time

Usage

```
chart_fwd_curves(df = dfwide, cmdty = "cmewti", weekly = FALSE, ...)
```

Arguments

df	Wide dataframe with date column and multiple series columns (multivariate)
cmdty	Futures contract code in expiry_table object: unique(expiry_table\$cmdty)
weekly	TRUE if you want weekly forward curves
...	other graphical parameters

Value

plot of forward curves through time

Author(s)

Philippe Cote

Examples

```
## Not run:  
df <- dfwide %>% dplyr::select(date, dplyr::starts_with("CL"))  
chart_fwd_curves(df = df, cmdty = "cmewti", weekly = TRUE,  
main="WTI Forward Curves", ylab="$ per bbl", xlab="", cex=2)  
  
## End(Not run)
```

chart_pairs	chart_pairs
-------------	-------------

Description

Pairwise scatter chart for timeseries.

Usage

```
chart_pairs(df = df, title = "Time Series Pairs Plot")
```

Arguments

df	Wide data frame
title	Chart title

Value

A plotly object

Author(s)

Philippe Cote

Examples

```
df <- dfwide %>% dplyr::select(date,CL01,NG01,H001,RB01)
chart_pairs(df = df, title = "example")
```

chart_PerfSummary	chart_PerformanceSummary
-------------------	--------------------------

Description

Multi Asset Display of Cumulative Performance and Drawdowns

Usage

```
chart_PerfSummary(
  ret = ret,
  geometric = TRUE,
  main = "Cumulative Returns and Drawdowns",
  linesize = 1.25
)
```


Arguments

ret	Wide dataframe univariate or multivariate of percentage returns.
geometric	Use geometric returns TRUE or FALSE.
main	Chart title.
linesize	Size of lines in chart and legend.

Value

Cumulative performance and drawdown charts.

Author(s)

Philippe Cote

Examples

```
df <- dflong %>% dplyr::filter(series %in% c("CL01","CL12","CL36"))
ret <- returns(df=df,retType="rel",period.return=1,spread=TRUE)
ret <- data.frame(rolladjust(x=ret,commodityname=c("cmewti"),rolltype=c("Last.Trade")))
chart_PerfSummary(ret=ret, geometric=TRUE, main="Cumulative Returns and Drawdowns",linesize=1.25)
```

chart_spreads	chart_spreads
---------------	---------------

Description

Chart spreads in specific futures contracts for multiple years.

Usage

```
chart_spreads(
  cpairs = cpairs,
  daysFromExpiry = 200,
  from = "2012-01-01",
  conversion = c(1, 1),
  feed = "CME_NymexFutures_EOD",
  iuser = "x@xyz.com",
  ipassword = "pass",
  title = "March/April ULSD Nymex Spreads",
  yaxis = "$ per bbl",
  output = "chart"
)
```

Arguments

cpairs	Data frame of contract pairs - see example.
daysFromExpiry	Number of days (numeric) from expiry to compute spreads.
from	From date as character string
conversion	Defaults to c(1,1) first and second contracts. 42 from \$ per gallons to bbls.
feed	Morningstar Feed Table.
iuser	Morningstar user name as character - sourced locally in examples.
ipassword	Morningstar user password as character - sourced locally in examples.
title	Title for chart.
yaxis	y-axis label.
output	"chart" for plotly object or "data" for dataframe.

Value

A plotly object or a dataframe

Author(s)

Philippe Cote

Examples

```
## Not run:
cpairs <- dplyr::tibble(year = c("2014", "2019", "2020"),
  first = c("@H04H", "@H09H", "@H00H"),
  second = c("@CL4J", "@CL9J", "@CL0J"))
chart_spreads(cpairs = cpairs, daysFromExpiry = 200, from = "2012-01-01",
  conversion = c(42,1), feed = "CME_NymexFutures_EOD",
  iuser = "x@xyz.com", ipassword = "pass",
  title = "March/April ULSD Nymex Spreads",
  yaxis = "$ per bbl",
  output = "data")

## End(Not run)
```

chart_zscore

chart_zscore

Description

Supports analytics and display of seasonal data. Z-Score is computed on residuals conditional on their seasonal period. Beware that most seasonal charts in industry e.g. (NG Storage) is not detrended so results once you apply an STL decomposition will vary from the unadjusted seasonal plot.

Usage

```
chart_zscore(
  df = df,
  title = "NG Storage Z Score",
  per = "yearweek",
  output = "zscore",
  chart = "seasons"
)
```

Arguments

df	Long data frame with columns series, date and value
title	Default is a blank space returning the unique value in df\$series.
per	Frequency of seasonality "yearweek" (DEFAULT). "yearmonth", "yearquarter"
output	"stl" for STL decomposition chart, "stats" for STL fitted statistics. "res" for STL fitted data. "zscore" for residuals Z-score, "seasonal" for standard seasonal chart.
chart	"seasons" for feasts::gg_season() (DEFAULT) "series" for feasts::gg_subseries()

Value

Time series of STL decomposition residuals Z-Scores, or standard seasonal chart with feasts package.

Author(s)

Philippe Cote

Examples

```
## Not run:
df <- eiaStocks %>% dplyr::filter(series == "NGLower48")
title <- "NGLower48"
chart_zscore(df = df, title = " ", per = "yearweek", output = "stl", chart = "seasons")
chart_zscore(df = df, title = " ", per = "yearweek", output = "stats", chart = "seasons")
chart_zscore(df = df, title = " ", per = "yearweek", output = "res", chart = "seasons")
chart_zscore(df = df, title = " ", per = "yearweek", output = "zscore", chart = "seasons")
chart_zscore(df = df, title = " ", per = "yearweek", output = "seasonal", chart = "seasons")

## End(Not run)
```

CRReuro

CRReuro

Description

European option binomial model on a stock without dividends. For academic purpose only. Use `fOptions::CRRBinomialTreeOptions` for real-life usage.

Usage

```
CRReuro(S, X, sigma, r, T2M, N, type)
```

Arguments

S	Stock price.
X	Strike price.
sigma	Implied volatility e.g. 0.20
r	Risk-free rate.
T2M	Time to maturity in years
N	Number of time steps. Internally $dt = T2M/N$.
type	"call" or "put"

Value

List of asset price tree, option value tree and option price.

Author(s)

Philippe Cote

Examples

```
CRReuro(S=100, X=100, sigma=0.2, r=0.1, T2M=1, N=5, type="call")
```

crudeassaysBP

crudeassaysBP

Description

Crude Assays from BP.

Usage

crudeassaysBP

Format

data frame

Source

<https://www.bp.com/en/global/bp-global-energy-trading/features-and-updates/technical-downloads/crudes-assays.html>

crudeassaysXOM

crudeassaysXOM

Description

Crude Assays from ExxonMobil.

Usage

crudeassaysXOM

Format

data frame

Source

<https://corporate.exxonmobil.com/Crude-oils/Crude-trading/Crude-oil-blends-by-API-gravity-and-by-s-APIgravity>

crudepipelines

crudepipelines

Description

GIS data set for North American crude pipelines.

Usage

crudepipelines

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

crudes

crudes

Description

Crude oil qualities.

Usage

crudes

Format

data frame

Source

Canadian Crude Monitor and BP Crude Assays

dflong	<i>dflong</i>
--------	---------------

Description

Futures settlement data set.

Usage

dflong

Format

data frame #' @source <https://www.morningstar.com/products/commodities-and-energy>

dfwide	<i>dfwide</i>
--------	---------------

Description

Futures settlement data set.

Usage

dfwide

Format

data frame #' @source <https://www.morningstar.com/products/commodities-and-energy>

df_fut	<i>df_fut</i>
--------	---------------

Description

Futures settlement data set.

Usage

df_fut

Format

data frame #' @source <https://www.morningstar.com/products/commodities-and-energy>

distdescplot	distdescplot
--------------	--------------

Description

Provides a summary of returns distribution

Usage

```
distdescplot(x = x)
```

Arguments

x Wide dataframe with date column and single series (univariate).

Value

Multiple plots describing the distribution.

Author(s)

Philippe Cote

Examples

```
x <- dflong %>% dplyr::filter(series=="CL01")
x <- returns(df=x, retType="rel", period.return=1, spread=TRUE)
x <- rolladjust(x=x, commodityname=c("cmewti"), rolltype=c("Last.Trade"))
distdescplot(x=x)
```

eia2tidy	eia2tidy
----------	----------

Description

Extracts data from the Energy Information Administration (EIA) API to tibble format with optional custom series name. Makes a clean wrapper for use with purrr for multiple series extraction. Query Browser at <https://www.eia.gov/opendata/qb.php>.

Usage

```
eia2tidy(ticker, key, name = " ")
```


Arguments

ticker EIA series name.
 key Your private EIA API token as character "<yourapikey>".
 name Name you want to give the series. Defaults to ticker if set to " "

Value

A tibble object with class date for weekly, monthly, quarterly or annual data and class POSIXct for hourly.

Author(s)

Philippe Cote

Examples

```

## Not run:
Single Series
RTL::eia2tidy(ticker = "PET.MCRFPTX2.M", key = "<yourapikey>", name = "TexasProd")
Multiple Series
eia_df <-tibble::tribble(~ticker, ~name,
  "PET.W_EPC0_SAX_YCUOK_MBBL.W", "CrudeCushing",
  "NG.NW2_EPG0_SWO_R48_BCF.W", "NGLower48") %>%
  dplyr::mutate(key = "EIAkey") %>%
  dplyr::mutate(df = purrr::pmap(list(ticker,key,name),.f=RTL::eia2tidy)) %>%
  dplyr::select(df) %>% tidyr::unnest(df)

## End(Not run)

```

eiaStocks

eiaStocks

Description

EIA weekly crude, NG, ULSD and RBOB stocks.

Usage

```
eiaStocks
```

Format

data frame

Source

<https://www.eia.gov>

eiaStorageCap	<i>eiaStorageCap</i>
---------------	----------------------

Description

EIA crude storage capacity in thousand bbls.

Usage

eiaStorageCap

Format

data frame

Source

<https://www.eia.gov/petroleum/storagecapacity/>

expiry_table	<i>expiry_table</i>
--------------	---------------------

Description

This dataframe provides detailed information on major futures contracts specifications pertaining to last settlement, notices and delivery dates. It also provides tickers in some data service.

Usage

expiry_table

Format

data frame

`fitOU``fitOU`

Description

Parameter estimation for Ornstein–Uhlenbeck process

Usage

```
fitOU(spread)
```

Arguments

`spread` Spread time series.

Value

List of alpha, mu and sigma estimates

Author(s)

Philippe Cote

Examples

```
spread <- simOU(mu=5, theta=.5, sigma=0.2, T=5, dt=1/250)
fitOU(spread)
```

`fizdiffs``fizdiffs`

Description

Randomized data set for education purpose of selected physical crude differentials to WTI.

Usage

```
fizdiffs
```

Format

data frame

garch garch

Description

Computes annualised Garch(1,1) volatilities using fGarch package.

Usage

```
garch(x = x, out = TRUE)
```

Arguments

x Wide dataframe with date column and single series (univariate).
out "chart" to return chart, "data" to return data or "fit" for garch fit output

Value

plot.xts object or xts series

Author(s)

Philippe Cote

Examples

```
x <- dflong %>% dplyr::filter(series=="CL01")  
x <- returns(df=x, retType="rel", period.return=1, spread=TRUE)  
x <- rolladjust(x=x, commodityname=c("cmewti"), rolltype=c("Last.Trade"))  
summary(garch(x=x, out="fit"))  
garch(x=x, out="chart")  
garch(x=x, out="data")
```

getCurve getCurve

Description

Returns forward curves from Morningstar API. See below for current feeds supported. You need your own credentials with Morningstar.

Usage

```

getCurve(
  feed = "Crb_Futures_Price_Volume_And_Open_Interest",
  contract = "CL",
  date = "2020-08-10",
  fields = c("Open, High, Low, Close"),
  iuser = "x@xyz.com",
  ipassword = "pass"
)

```

Arguments

feed	Morningstar Feed Table e.g "Crb_Futures_Price_Volume_And_Open_Interest".
contract	Morningstar contract root e.g. "CL" for CME WTI and "BG" for ICE Brent.
date	From date as character string.
fields	Defaults to c("Open, High, Low, Close").
iuser	Morningstar user name as character - sourced locally in examples.
ipassword	Morningstar user password as character - sourced locally in examples.

Value

wide data frame

Current Feeds Supported

- Crb_Futures_Price_Volume_And_Open_Interest
- CME_NymexFuturesIntraday_EOD
- ICE_EuroFutures and ICE_EuroFutures_continuous

Author(s)

Philippe Cote

Examples

```

## Not run:
# CME WTI Futures
getCurve(feed = "Crb_Futures_Price_Volume_And_Open_Interest",contract = "CL",
date = "2020-07-13",fields = c("Open, High, Low, Close"),
iuser = "x@xyz.com", ipassword = "pass")

getCurve(feed = "Crb_Futures_Price_Volume_And_Open_Interest",contract = "BG",
date = "2020-07-13",fields = c("Open, High, Low, Close"),
iuser = "x@xyz.com", ipassword = "pass")

## End(Not run)

```

```
getGenscapePipeOil    getGenscapePipeOil
```

Description

Returns oil pipeline flows in barrels per day data from Genscape API. You need your own credentials. Refer to API documentation for argument values. It is assumed if you use this function that you know the pipelines you need to extract to build supply demand balances. Use the online API to identify the pipeline IDs. <https://developer.genscape.com/docs/services/oil-transportation/operations/GetPipelineFlowValues>

Usage

```
getGenscapePipeOil(
  frequency = "daily",
  regions = "Canada",
  pipelineIDs = c(97),
  revision = "revised",
  limit = 5000,
  offset = 0,
  startDate = "2015-01-01",
  endDate = as.character(Sys.Date()),
  apikey = "yourapikey"
)
```

Arguments

frequency	"daily" DEFAULT.
regions	See API webpage. Multiple values separated by commas e.g. "Canada", "Gulf-Coast").
pipelineIDs	See API webpage. c(98,54...) for specific pipes.
revision	See API webpage.
limit	See API webpage. Max 5000
offset	See API webpage.
startDate	"yyyy-mm-dd" as character string
endDate	"yyyy-mm-dd" as character string
apikey	Your API key as a character string.

Value

wide data frame

Author(s)

Philippe Cote

Examples

```
## Not run:
getGenscapePipeOil(frequency = "daily", regions = "Canada", pipelineIDs = c(97),
  revision = "revised", limit = 5000, offset = 0,
  startDate = "2015-01-01", endDate = as.character(Sys.Date()),
  apikey = "yourapikey")

## End(Not run)
```

```
getGenscapeStorageOil getGenscapeStorageOil
```

Description

Returns oil storage data from Genscape API. You need your own credentials. Refer to API documentation for argument values. <https://developer.genscape.com/docs/services/oil-storage/operations/StorageVolumeByOwnerGet>

Usage

```
getGenscapeStorageOil(
  feed = "owner-volumes",
  regions = "Canada",
  products = "Crude",
  revision = "revised",
  limit = 5000,
  offset = 0,
  startDate = "2011-01-01",
  endDate = as.character(Sys.Date()),
  apikey = "yourapikey"
)
```

Arguments

feed	"owner-volumes" DEFAULT or "tank-volumes"
regions	See API webpage. Multiple values separated by commas e.g. "Canada, Cushing").
products	See API webpage. Multiple values separated by commas e.g. "Crude, JetFuel").
revision	See API webpage.
limit	See API webpage. Max 5000
offset	See API webpage.
startDate	"yyyy-mm-dd" as character string
endDate	"yyyy-mm-dd" as character string
apikey	Your API key as a character string.

Value

wide data frame

Author(s)

Philippe Cote

Examples

```
## Not run:
getGenscapeStorageOil(feed = "owner-volumes", regions = "Canada", products = "Crude",
  evision = "revised", limit = 5000, offset = 0, startDate = "2011-01-01", endDate = "2020-11-01"
  apikey = "<yourapikey>")

## End(Not run)
```

getIRswapCurve	getIRswapCurve
----------------	----------------

Description

Extract historical data for tsQuotes in RQuantlib to bootstrap swap curve using Morningstar and FRED as data source.

Usage

```
getIRswapCurve(
  currency = "USD",
  from = "2019-01-01",
  iuser = "x@xyz.com",
  ipassword = "pass"
)
```

Arguments

currency	Currently only USD LIBOR implemented.
from	From date as character string
iuser	Morningstar user name as character - sourced locally in examples.
ipassword	Morningstar user password as character - sourced locally in examples.

Value

wide data frame

Author(s)

Philippe Cote

Examples

```
## Not run:
getIRswapCurve(currency="USD", from="2019-08-26", iuser = username, ipassword = password)

## End(Not run)
```

```
getPrice          getPrice
```

Description

Returns data from Morningstar API. See below for current feeds supported. You need your own credentials with Morningstar. In examples sourced locally.

Usage

```
getPrice(
  feed = "CME_NymexFutures_EOD",
  contract = "@CL21Z",
  from = "2020-09-01",
  iuser = "x@xyz.com",
  ipassword = "pass"
)
```

Arguments

feed	Morningstar Feed Table.
contract	Morningstar key.
from	From date as character string
iuser	Morningstar user name as character - sourced locally in examples.
ipassword	Morningstar user password as character - sourced locally in examples.

Value

wide data frame

Current Feeds Supported

- CME_CbotFuturesEOD and CME_CbotFuturesEOD_continuous
- CME_NymexFutures_EOD and CME_NymexFutures_EOD_continuous
- CME_NymexOptions_EOD
- CME_CmeFutures_EOD and CME_CmeFutures_EOD_continuous
- CME_Comex_FuturesSettlement_EOD and CME_Comex_FuturesSettlement_EOD_continuous
- LME_AskBidPrices_Delayed

- SHFE_FuturesSettlement_RT
- ICE_EuroFutures and ICE_EuroFutures_continuous
- ICE_NybotCoffeeSugarCocoaFutures and ICE_NybotCoffeeSugarCocoaFutures_continuous
- CME_STLPCPC_Futures
- CFTC_CommitmentsOfTradersCombined. Requires multiple keys. Separate them by a space e.g. "N10 06765A NYME 01".
- Morningstar_FX_Forwards. Requires multiple keys. Separate them by a space e.g. "USD-CAD 2M".
- ERCOT_LmpsByResourceNodeAndElectricalBus.
- PJM_Rt_Hourly_Lmp.

Author(s)

Philippe Cote

Examples

```
## Not run:
getPrice(feed="CME_NymexFutures_EOD",contract="@CL21Z",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="CME_NymexFutures_EOD_continuous",contract="CL_006_Month",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="CME_NymexOptions_EOD",contract="@L021ZP4000",
from="2020-03-15",iuser = username, ipassword = password)
getPrice(feed="CME_CbotFuturesEOD",contract="C0Z",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="CME_CbotFuturesEOD_continuous",contract="ZB_001_Month",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="CME_CmeFutures_EOD_continuous",contract="HE_006_Month",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="Morningstar_FX_Forwards",contract="USDCAD 2M",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="CME_CmeFutures_EOD",contract="LH0N",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="CME_CmeFutures_EOD_continuous",contract="HE_006_Month",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="ICE_EuroFutures",contract="BRN0Z",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="ICE_EuroFutures_continuous",contract="BRN_001_Month",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="ICE_NybotCoffeeSugarCocoaFutures",contract="SB21H",
from="2019-08-26",iuser = username, ipassword = password)
getPrice(feed="ICE_NybotCoffeeSugarCocoaFutures_continuous",contract="SF_001_Month",
from="2019-08-26",iuser = username, ipassword = password)

## End(Not run)
```

`getPrices``getPrices`

Description

Multiple Morningstar API calls using getPrice functions. Refer to 'getPrices()' for list of currently supported data feeds.

Usage

```
getPrices(  
  feed = "CME_NymexFutures_EOD",  
  contracts = c("CL9Z", "CL0F", "CL0M"),  
  from = "2019-01-01",  
  iuser = "x@xyz.com",  
  ipassword = "pass"  
)
```

Arguments

feed	Morningstar Feed Table
contracts	Symbols vector
from	From date as character string
iuser	Morningstar user name as character - sourced locally in examples.
ipassword	Morningstar user password as character - sourced locally in examples.

Value

wide data frame

Author(s)

Philippe Cote

Examples

```
## Not run:  
getPrices(feed = "CME_NymexFutures_EOD", contracts = c("@CL0Z", "@CL1F", "@CL21H", "@CL21Z"),  
  from = "2020-01-01", iuser = username, ipassword = password)  
  
## End(Not run)
```

holidaysOil	<i>holidaysOil</i>
-------------	--------------------

Description

Holiday calendars for NYMEX and ICE Brent

Usage

```
holidaysOil
```

Format

data frame

ir_df_us	ir_df_us
----------	----------

Description

Extracts US Treasury Zero Rates

Usage

```
ir_df_us(quandlkey = quandlkey, ir.sens = 0.01)
```

Arguments

quandlkey	Your Quandl key "quandlkey"
ir.sens	Creates plus and minus IR sensitivity scenarios with specified shock value.

Value

Data frame of zero rates

Author(s)

Philippe Cote

Examples

```
## Not run:
us.df <- ir_df_us(quandlkey = quandlkey,ir.sens=0.01)

## End(Not run)
```

Ingterminals

Ingterminals

Description

GIS data set for North American LNG terminals.

Usage

Ingterminals

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

nghubs

nghubs

Description

GIS data set for North American NG Trading Hubs.

Usage

nghubs

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

ngpipelines

ngpipelines

Description

GIS data set for North American NG pipelines.

Usage

ngpipelines

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

ngstorage

ngstorage

Description

GIS data set for North American NG storage.

Usage

ngstorage

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

npv	npv
-----	-----

Description

Compute NPV

Usage

```
npv(
  init.cost = -375,
  C = 50,
  cf.freq = 0.25,
  TV = 250,
  T2M = 2,
  disc.factors = us.df,
  BreakEven = FALSE,
  BE.yield = 0.01
)
```

Arguments

init.cost	Initial investment cost
C	Periodic cash flow
cf.freq	Cash flow frequency in year fraction e.g. quarterly = 0.25
TV	Terminal Value
T2M	Time to Maturity in years
disc.factors	Data frame of discount factors using ir.df.us() function.
BreakEven	TRUE when using a flat discount rate assumption.
BE.yield	Set the flat IR rate when BeakEven = TRUE.

Value

List of NPV and NPV Data frame

Author(s)

Philippe Cote

Examples

```
## Not run:
us.df <- ir_df_us(quandlkey = quandlkey, ir.sens=0.01)
npv(init.cost=-375, C=50, cf.freq=.5, TV=250, T2M=2,
  disc.factors=us.df, BreakEven=TRUE, BE.yield=.0399)$npv
npv(init.cost=-375, C=50, cf.freq=.5, TV=250, T2M=2,
```

```
disc.factors=us.df,BreakEven=TRUE,BE.yield=.0399)$df
## End(Not run)
```

planets	<i>planets</i>
---------	----------------

Description

Planet metrics from NASA

Usage

planets

Format

data frame

Source

<https://nssdc.gsfc.nasa.gov/planetary/factsheet/index.html>

productspipelines	<i>productspipelines</i>
-------------------	--------------------------

Description

GIS data set for North American products pipelines.

Usage

productspipelines

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

productsterminals *productsterminals*

Description

GIS data set for North American products terminals.

Usage

productsterminals

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

promptBeta promptBeta

Description

Returns betas of futures contracts versus front futures contract.

Usage

promptBeta(x = x, period = "all", betatype = "all", output = "chart")

Arguments

x	Wide dataframe with date column and multiple series columns (multivariate).
period	"all" or numeric period of time in last n periods as character eg "100".
betatype	"all" "bull" "bear".
output	"betas" or "chart"

Value

betas data frame or plotly chart of betas

Author(s)

Philippe Cote

Examples

```
## Not run:
x <- dflong %>% dplyr::filter(grepl("CL",series))
x <- x %>% dplyr::mutate(series = readr::parse_number(series)) %>% dplyr::group_by(series)
x <- RTL::returns(df = x,retType = "abs",period.return = 1,spread = TRUE)
x <- RTL::rolladjust(x = x,commodityname = c("cmewti"),rolltype = c("Last.Trade"))
x <- x %>% dplyr::filter(!grepl("2020-04-20|2020-04-21",date))
promptBeta(x = x,period = "all",betatype = "all",output = "chart")
promptBeta(x = x,period = "all",betatype = "all",output = "betas")
promptBeta(x = x,period = "100",betatype = "all",output = "betas")

## End(Not run)
```

ref.opt.inputs

ref.opt.inputs

Description

Simple refinery input to be used in running LP modeling for education purposes.

Usage

```
ref.opt.inputs
```

Format

data frame

ref.opt.outputs

ref.opt.ouputs

Description

Simple refinery outputs and constraints to be used in running LP modeling for education purposes.

Usage

```
ref.opt.outputs
```

Format

data frame

refineries	<i>refineries</i>
------------	-------------------

Description

GIS data set for North American refineries.

Usage

refineries

Format

data frame

Source

https://www.eia.gov/maps/layer_info-m.php

refineryLP	refineryLP
------------	------------

Description

Plain vanilla refinery optimization LP model.

Usage

refineryLP(crudes = ref.opt.inputs, products = ref.opt.outputs)

Arguments

crudes	Data frame of crude inputs
products	Data frame of product outputs and max outputs.

Value

Optimal crude slate and profits

Author(s)

Philippe Cote

Examples

refineryLP(crudes = ref.opt.inputs, products = ref.opt.outputs)

returns	returns
---------	---------

Description

Computes periodic returns from a dataframe ordered by date

Usage

```
returns(df = dflong, retType = "abs", period.return = 1, spread = FALSE)
```

Arguments

df	Long dataframe with colnames = c("date","value","series")
retType	"abs" for absolute, "rel" for relative, or "log" for log returns.
period.return	Number of rows over which to compute returns.
spread	TRUE if you want to spread into a long dataframe.

Value

A dataframe object of returns.

Author(s)

Philippe Cote

Examples

```
returns(df = dflong,retType = "abs",period.return = 1,spread = TRUE)
returns(df = dflong,retType = "abs",period.return = 1,spread = FALSE)
```

rolladjust	rolladjust
------------	------------

Description

Returns a xts price or return object adjusted for contract roll. The methodology used to adjust returns is to remove the daily returns on the day after expiry and for prices to adjust historical rolling front month contracts by the size of the roll at each expiry. This is conducive to quantitative trading strategies as it reflects the PL of a financial trader.

Usage

```
rolladjust(x, commodityname = c("cmewti"), rolltype = c("Last.Trade"), ...)
```

Arguments

x An xts object of prices or returns.
commodityname Name of commodity in expiry_table. See example below for values.
rolltype Type of contract roll: "Last.Trade" or "First.Notice".
... Other parms

Value

Roll-adjusted xts object of returns

Author(s)

Philippe Cote

Examples

```

unique(expiry_table$comdty) # for list of commodity names
ret <- returns(df=dflong,retType="abs",period.return=1,spread=TRUE)[,1:2]
rolladjust(x=ret,commodityname=c("cmewti"),rolltype=c("Last.Trade"))

```

simGBM

simGBM

Description

Simulates a Geometric Brownian Motion process

Usage

```
simGBM(S0 = 10, drift = 0, sigma = 0.2, T2M = 1, dt = 1/12)
```

Arguments

S0 Spot price at t=0
drift Drift term in percentage
sigma Standard deviation
T2M Maturity in years
dt Time step in period e.g. 1/250 = 1 business day.

Value

A numeric vector of simulated values

Author(s)

Philippe Cote

Examples

```
simGBM(S0=10,drift=0,sigma=0.2,T2M=1,dt=1/12)
```

```
simOU
```

```
simOU
```

Description

Simulates a Ornstein–Uhlenbeck process

Usage

```
simOU(S0 = 5, mu = 5, theta = 0.5, sigma = 0.2, T2M = 1, dt = 1/12)
```

Arguments

S0	S at t=0
mu	Mean reversion level
theta	Mean reversion speed
sigma	Standard deviation
T2M	Maturity in years
dt	Time step size e.g. 1/250 = 1 business day.

Value

A numeric vector of simulated values

Author(s)

Philippe Cote

Examples

```
simOU(S0=5,mu=5,theta=.5,sigma=0.2,T2M=1,dt=1/12)
```

 simOUJ

 simOUJ

Description

Simulates a Ornstein–Uhlenbeck process with Jumps

Usage

```
simOUJ(
  S0 = 5,
  mu = 5,
  theta = 10,
  sigma = 0.2,
  jump_prob = 0.05,
  jump_avesize = 2,
  jump_stdv = 0.05,
  T2M = 1,
  dt = 1/250
)
```

Arguments

S0	S at t=0
mu	Mean reversion level
theta	Mean reversion speed
sigma	Standard deviation
jump_prob	Probability of jumps
jump_avesize	Average size of jumps
jump_stdv	Standard deviation of jump average size
T2M	Maturity in years
dt	Time step size e.g. 1/250 = 1 business day.

Value

A numeric vector of simulated values

Author(s)

Philippe Cote

Examples

```
simOUJ(S0=5,mu=5,theta=.5,sigma=0.2,jump_prob=0.05,jump_avesize = 3,jump_stdv = 0.05,T2M=1,dt=1/12)
```

stl_decomp	stl_decomp
------------	------------

Description

Provides a summary of returns distribution

Usage

```
stl_decomp(x = x, output = "chart", s.window = 13, s.degree = 1, ...)
```

Arguments

x	Wide dataframe with date column and single series (univariate).
output	"chart" to see output as a graph. "data" for results as a list.
s.window	Either the character string "periodic" or the span (in lags) of the loess window for seasonal extraction, which should be odd. This has no default.
s.degree	Degree of locally-fitted polynomial in seasonal extraction. Should be zero or one.
...	Other parms

Value

a chart or list object of results

Author(s)

Philippe Cote

Examples

```
x <- dflong %>% dplyr::filter(series=="CL01")
stl_decomp(x,output="chart",s.window=13,s.degree=1)
stl_decomp(x,output="data",s.window=13,s.degree=1)
```

swapCOM	swapCOM
---------	---------

Description

Commodity swap pricing from exchange settlement

Usage

```
swapCOM(
  futures = futs,
  futuresNames = c("CL0M", "CL0N"),
  pricingDates = c("2020-05-01", "2020-05-30"),
  contract = "cmewti",
  exchange = "nymex"
)
```

Arguments

futures	Wide data frame of futures prices for the given swap pricing dates
futuresNames	Tickers of relevant futures contracts
pricingDates	Vector of start and end pricing dates as character. See example.
contract	Contract code in data(expiry_table). sort(unique(expiry_table\$cmdty)) for options.
exchange	Exchange code in data(holidaysOil). Currently only "nymex" and "ice" supported.

Value

Data frame of historical swap prices.

Author(s)

Philippe Cote

Examples

```
## Not run:
c <- paste0("CL0", c("M", "N", "Q"))
futs <- getPrices(feed="CME_NymexFutures_EOD", contracts = c, from="2019-08-26",
  iuser = username, ipassword = password)
swapCOM(futures = futs, futuresNames=c("CL0M", "CL0N"),
  pricingDates = c("2020-05-01", "2020-05-30"), contract = "cmewti", exchange = "nymex")

## End(Not run)
```

swapFutWeight	swapFutWeight
---------------	---------------

Description

Returns the percentage weight of the future in Calendar Month Average swaps

Usage

```
swapFutWeight(
  Month = "2020-09-01",
  contract = "cmewti",
  exchange = "nymex",
  output = "first.fut.weight"
)
```

Arguments

Month	First calendar day of the month.
contract	Contract code in data(expiry_table). sort(unique(expiry_table\$cmdty)) for options.
exchange	Exchange code in data(holidaysOil). Currently only "nymex" and "ice" supported.
output	Either "numDaysFut1", "numDaysFut2" or "first.fut.weight"

Value

What you defined in outputs. If first.fut.weight, to compute swap 1 - first.fut.weight =

Author(s)

Philippe Cote

Examples

```
swapFutWeight(Month = "2020-09-01",
  contract = "cmewti", exchange = "nymex", output = "first.fut.weight")
```

swapInfo	swapInfo
----------	----------

Description

Returns dataframe required to price a WTI averaging instrument based on first line settlements.

Usage

```
swapInfo(
  date = "2020-05-06",
  feeds = dplyr::tibble(feed = c("Crb_Futures_Price_Volume_And_Open_Interest",
    "CME_NymexFutures_EOD_continuous"), ticker = c("CL", "CL_001_Month")),
  contract = "cmewti",
  exchange = "nymex",
  iuser = "x@xyz.com",
  ipassword = "pass",
  output = "all"
)
```

Arguments

date	Character date as of which you want to extract daily settlement and forward values.
feeds	Feeds for Morningstar getCurve() and getPrice().
contract	Contract code in data(expiry_table). sort(unique(expiry_table\$cmdty)) for options.
exchange	Exchange code in data(holidaysOil). Defaults to "nymex".
iuser	Morningstar user name as character - sourced locally in examples.
ipassword	Morningstar user password as character - sourced locally in examples.
output	"chart" or "all"

Value

Plot or a list of data frame and plot if output = "all".

Author(s)

Philippe Cote

Examples

```
## Not run:
feeds = dplyr::tibble(feed = c("Crb_Futures_Price_Volume_And_Open_Interest",
                              "CME_NymexFutures_EOD_continuous"),
                    ticker = c("CL", "CL_001_Month"))
swapInfo(date = "2020-05-06", feeds = feeds, contract = "cmewti", exchange = "nymex",
         iuser = "x@xyz.com", ipassword = "pass", output = "all")

## End(Not run)
```

 swapIRS

 swapIRS

Description

Commodity swap pricing from exchange settlement

Usage

```
swapIRS(
  trade.date = lubridate::today(),
  eff.date = lubridate::today() + 2,
  mat.date = lubridate::today() + 2 + lubridate::years(2),
  notional = 1e+06,
  PayRec = "Rec",
  fixed.rate = 0.05,
  float.curve = usSwapCurves,
  reset.freq = 3,
  disc.curve = usSwapCurves,
  convention = c("act", 360),
  bus.calendar = "NY",
  output = "price"
)
```

Arguments

trade.date	Date object. Defaults to today().
eff.date	Date object. Defaults to today() + 2 days.
mat.date	Date object. Defaults to today() + 2 years.
notional	Numeric value of notional. Defaults to 1,000,000.
PayRec	"Pay" or "Rec" fixed.
fixed.rate	Numeric fixed interest rate. Defaults to 0.05.
float.curve	List of interest rate curves. Defaults to data("usSwapCurves").
reset.freq	Numeric where 1 = "monthly", 3 = quarterly, 6 = Semi annual 12 = yearly.
disc.curve	List of interest rate curves. Defaults to data("usSwapCurves").

convention	Vector of convention e.g. c("act",360) c(30,360),...
bus.calendar	Banking day calendar. Not implemented.
output	"price" for swap price or "all" for price, cash flow data frame, duration.

Value

List of swap price, cash flow data frame, duration.

Author(s)

Philippe Cote

Examples

```
data("usSwapCurves")
swapIRS(trade.date = as.Date("2020-01-04"), eff.date = as.Date("2020-01-06"),
mat.date = as.Date("2022-01-06"), notional = 1000000,
PayRec = "Rec", fixed.rate=0.05, float.curve = usSwapCurves, reset.freq=3,
disc.curve = usSwapCurves, convention = c("act",360),
bus.calendar = "NY", output = "all")
```

tickers_eia	<i>tickers_eia</i>
-------------	--------------------

Description

Supports automated upload of EIA data through its API by categories. Data frame organized by Supply Demand categories and products.

Usage

```
tickers_eia
```

Format

data frame

Source

<https://www.eia.gov/>

tradeCycle	<i>tradeCycle</i>
------------	-------------------

Description

Crude Trading Trade Cycles

Usage

tradeCycle

Format

data frame

tradeprocess	<i>tradeprocess</i>
--------------	---------------------

Description

Data set for explaining the various ways to monetize a market view.

Usage

tradeprocess

Format

data frame

tradeStats	tradeStats
------------	------------

Description

Compute list of risk reward metrics

Usage

tradeStats(x, Rf = 0)

Arguments

x	xts object of returns
Rf	Risk-free rate

Value

List of risk/reward metrics.

Author(s)

Philippe Cote

Examples

```
library(quantmod)
getSymbols("SPY", return.class = "zoo")
SPY$retClCl <- na.omit(quantmod::Delt(Cl(SPY),k=1,type='arithmetic'))
tradeStats(x=SPY$retClCl,Rf=0)
```

usSwapCurves

usSwapCurves

Description

USD IR Discount, Forward and Zero curves from RQuantlib::DiscountCurve

Usage

usSwapCurves

Format

List #' @source Morningstar and FRED

usSwapCurvesPar

usSwapCurvesPar

Description

USD IR Discount, Forward and Zero curves from RQuantlib::DiscountCurve - Parallel toy data set

Usage

usSwapCurvesPar

Format

data frame

usSwapIR

usSwapIR

Description

USD Interest Rate Swap Curve for RQuantlib bootstrapping. See usSwapIRdef for sources and tickers.

Usage

usSwapIR

Format

data frame #' @source Morningstar and FRED

usSwapIRdef

usSwapIRdef

Description

USD Interest Rate Swap Curve definitions with sources and tickers

Usage

usSwapIRdef

Format

data frame #' @source Morningstar and FRED

Index

* datasets

- cancrudeassays, 4
 - cancrudeassayssum, 4
 - cancrudeprices, 5
 - crudeassaysBP, 13
 - crudeassaysXOM, 13
 - crudepipelines, 14
 - crudes, 14
 - df_fut, 15
 - dflong, 15
 - dfwide, 15
 - eiaStocks, 17
 - eiaStorageCap, 18
 - expiry_table, 18
 - fizdiffs, 19
 - holidaysOil, 28
 - lngterminals, 29
 - nghubs, 29
 - ngpipelines, 30
 - ngstorage, 30
 - planets, 32
 - productspipelines, 32
 - productsterminals, 33
 - ref.opt.inputs, 34
 - ref.opt.outputs, 34
 - refineries, 35
 - tickers_eia, 45
 - tradeCycle, 46
 - tradeprocess, 46
 - usSwapCurves, 47
 - usSwapCurvesPar, 47
 - usSwapIR, 48
 - usSwapIRdef, 48
- bond, 3
- cancrudeassays, 4
 - cancrudeassayssum, 4
 - cancrudeprices, 5
 - chart_eia_sd, 5
 - chart_eia_steo, 6
 - chart_fwd_curves, 7
 - chart_pairs, 8
 - chart_PerfSummary, 8
 - chart_spreads, 9
 - chart_zscore, 10
 - CRReuro, 12
 - crudeassaysBP, 13
 - crudeassaysXOM, 13
 - crudepipelines, 14
 - crudes, 14
 - df_fut, 15
 - dflong, 15
 - dfwide, 15
 - distdescplot, 16
 - eia2tidy, 16
 - eiaStocks, 17
 - eiaStorageCap, 18
 - expiry_table, 18
 - fitOU, 19
 - fizdiffs, 19
 - garch, 20
 - getCurve, 20
 - getGenscapePipeOil, 22
 - getGenscapeStorageOil, 23
 - getIRswapCurve, 24
 - getPrice, 25
 - getPrices, 27
 - holidaysOil, 28
 - ir_df_us, 28
 - lngterminals, 29
 - nghubs, 29
 - ngpipelines, 30

ngstorage, 30
npv, 31

planets, 32
productspipelines, 32
productsterminals, 33
promptBeta, 33

ref.opt.inputs, 34
ref.opt.outputs, 34
refineries, 35
refineryLP, 35
returns, 36
rolladjust, 36

simGBM, 37
simOU, 38
simOUJ, 39
stl_decomp, 40
swapCOM, 41
swapFutWeight, 42
swapInfo, 43
swapIRS, 44

tickers_eia, 45
tradeCycle, 46
tradeprocess, 46
tradeStats, 46

usSwapCurves, 47
usSwapCurvesPar, 47
usSwapIR, 48
usSwapIRdef, 48