Package: FinanceGraphs (via r-universe)

October 7, 2026

Title Flexible Graphs for Analysis of Financial Data and Time Series

Version 0.9.22

Description Flexible wrappers around R graphics modules 'dygraphs' <https://dygraphs.com/> and 'ggplot2' <https://ggplot2.tidyverse.org/> to visualize data commonly found in Financial Studies, with an emphasis on time series. Interactive time series plots include multiple options for incorporating external data such as forecasts and events. Other static plots useful for time series data include an intuitive and generic scatter plotter, a boxplot generator suitable for multiple time series, and event study plotters for time series analysis around sets of dates.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1.0)

Imports dplyr (>= 1.0.0), purrr (>= 1.2.0), tibble (>= 1.0.0), broom (>= 1.0.0), stringr (>= 1.2.0), forcats (>= 1.0.0), lubridate (>= 1.7.0), utils (>= 4.0.0), tidyr (>= 0.6.3), xts (>= 0.10.0), qlcal (>= 0.1.0), graphics (>= 4.0.0), grDevices (>= 4.5.0), ggplot2 (>= 4.0.0), ggrepel (>= 0.9.0), ggiraph (>= 0.9.0), knitr (>= 1.45), usethis (>= 3.2.0), cpm (>= 2.0.0), RegimeChange (>= 0.1.0), scales (>= 1.1.0), stats (>= 4.5.0), utils (>= 4.5.0), data.table (>= 1.9.8), dygraphs (>= 1.1.0)

Suggests ecp (>= 3.1.0), tidyquant (>= 1.0.0), RColorBrewer (>= 1.0.0), prophet (>= 1.1.0), forecast (>= 9.0.0), timetk (>= 2.9.0), sweep (>= 0.2.0), alphavantagepf (>= 0.3.0), testthat, rmarkdown (>= 2.25)

Roxygen list(markdown = TRUE)

LazyData true

VignetteBuilder knitr

URL https://github.com/derekholmes0/FinanceGraphs, https://derekholmes0.github.io/FinanceGraphs/

BugReports https://github.com/derekholmes0/FinanceGraphs/issues

Config/Needs/website rmarkdown

Config/roxygen2/version 8.1.0

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev git make libgit2-dev libicu-dev libpng-dev libuv1-dev libssl-dev python3 libx11-dev

Repository https://derekholmes0.r-universe.dev

Date/Publication 2026-10-07 12:26:12 UTC

RemoteUrl https://github.com/derekholmes0/financegraphs

RemoteRef HEAD

RemoteSha 9a6d1a33dcb962b128bdc97e96a197442bedc01e

Contents

consumer_sent . . . 3
earnings_ibm . . . 3
eqtypx . . . 4
eqtypx_melt . . . 4
eqtyrtn . . . 5
example_fcst_set . . . 5
fg_addbreakouts . . . 6
fg_cut_to_events . . . 7
fg_dates_of_interest . . . 8
fg_eventStudy . . . 9
fg_findTurningPoints . . . 11
fg_get_aes . . . 12
fg_prophet . . . 13
fg_ratingsEvents . . . 14
fg_RegimeChange . . . 15
fg_scatplot . . . 16
fg_seasonalstudy . . . 20
fg_signal_to_events . . . 22
fg_sweep . . . 23
fg_sync_group . . . 24
fg_tsboxplot . . . 25
fg_update_aes . . . 27
fgts_dygraph . . . 28
gendtstr . . . 33
imfdta . . . 35
nomfxdta . . . 35
ratings_db . . . 36
recession_indic . . . 36
reerdta . . . 37
yc_CMSUST . . . 37

Index **39**

consumer_sent *Consumer Sentiment Data*

Description

University of Michigan Consumer Sentiment Data, FRED code UMCSENT

Usage

```
consumer_sent
```

Format

`consumer_sent`:
A data frame with 120 rows and 3 columns

symbol FRED identifier
date Date of report
pricce Observation

Source

https://fred.stlouisfed.org/

earnings_ibm *IBM Earnings*

Description

IBM Earnings download

Usage

```
earnings_ibm
```

Format

`earnings_ibm`:
A `data.table` with 120 observations and e columns

reportedDate Earnings announcement date
reportedEPS Reported Earnings per Share
symbol Company

Source

https://business-science.github.io/tidyquant/

eqtypx *Equity Prices*

Description

Closing Equity Prices

Usage

```
eqtypx
```

Format

`eqtypx`:
A data table with 2529 obersavatin and 5 variables

date Date Of Observation
EEM EEM Closing Price
IBM IBM Closing Price
QQQ QQQ Closing Price
TLT TLT Closing Price

Source

https://business-science.github.io/tidyquant/

eqtypx_melt *Equity Prices*

Description

Closing Equity Prices (melted)

Usage

```
eqtypx_melt
```

Format

`eqtypx_melt`:
A tibble with 10116 observations and 3 columns

date Date Of Observation
variable Ticker
value Closing Price

Source

https://business-science.github.io/tidyquant/

eqtyrtn *Equity returns*

Description

Closing Equity returns and a rolling regression

Usage

```
eqtyrtn
```

Format

`eqtyrtn`:
A tibble with 2529 observations and 6 columns

date Date Of Observation
EEM EEM Log daily return
IBM IBM Log daily return
QQQ QQQ Log daily return
TLT TLT Log daily return
p_TLT_QQQ ROlling 66 business day regression of TLT on QQQ p.value
ibmdiv Dividends reported for IBM

Source

https://business-science.github.io/tidyquant/

example_fcst_set *Example forecasts*

Description

IBM Stock price forecasts from ets baseline ets model

Usage

```
example_fcst_set
```

Format

`example_fcst_set`**:**
A data table with forecasts for two equities

date Date
QQQ.f Forecast price for QQQ
QQQ.flo 80th percentile in Forecast CI for QQQ
QQQ.fhi 20th percentile in Forecast CI for QQQ
IBM.f Forecast price for IBM
IBM.flo 80th percentile in Forecast CI for IBM
IBM.fhi 20th percentile in Forecast CI for IBM

Source

https://business-science.github.io/tidyquant/

`fg_addbreakouts` *Add Breakout events to time series plots*

Description

Wrapper around the function `ecp::e.divisive()` to create events for `fgts_dygraph()`

Usage

```
fg_addbreakouts(indta, annotationstyle = "singleasdate", ...)
```

Arguments

`indta`	Time series `data.table` with a date as the first column and a value series as the second column.
`annotationstyle`	String in set (`singleasdate`,`singleasvalue`,'breakno')
`...`	Parameters passed to `ecp::e.divisive()`

Details

Event Helpers : fg_addbreakouts

Value

`data.table` suitable for passing into `fgts_dygraph()` via the `event_ds` parameter

Examples

```
if (requireNamespace("ecp", quietly = TRUE)) {
dta <- tail(eqtypx[,.(date,QQQ,TLT)],2*260)
fgts_dygraph(dta,event_ds=fg_addbreakouts(dta,min.size=66,R=40),title="With Breakouts")
}
```

`fg_cut_to_events` *Cut a univariate time series into colored event bars.*

Description

Event Helpers: fg_cut_to_events

Usage

```
fg_cut_to_events(
  indta,
  ncutsperside = 4,
  center = 0,
  extend = TRUE,
  invert = FALSE
)
```

Arguments

`indta`	Time series `data.table` with a date as the first column and a value series as the second column.
`ncutsperside`	: Integer with number of colors to use on each side of 'center'
`center`	: String or Double as follows: • Double (default 0) Normalize data by subtracting `center` • `"median"` Normalize data by subtracting median of all observations. • `"zscore"` Normalize data by using standard `scale()` function
`extend`	Logical (Default: TRUE) to extend data to today (`Sys.Date()`)
`invert`	Use opposite color schemes for data, i.e. "red" for good outcomes

Details

Always uses first date column and first numeric columns in data. If `indta` has multiple series, filter them before calling the function.

Value

`data.table` suitable for passing into `fgts_dygraph()` via the `event_ds` parameter

Examples

```
smalldta <- narrowbydtstr(eqtypx[,.(date,IBM,QQQ)],"-2y::")
events <- fg_cut_to_events(consumer_sent,center="zscore")
fgts_dygraph(smalldta,title="With Sentiment ranges",event_ds=events)
```

fg_dates_of_interest *Maintain Aethestics and Dates of Interest*

Description

fg_get_dates_of_interest() gets a set of time events for use in fg time series graphs fg_update_dates_of_interest() updates a set of time events for future use in time series graphs

Usage

```
fg_get_dates_of_interest(
  search_categories = "",
  use_default = TRUE,
  startdt = NULL,
  totoday = FALSE
)

fg_update_dates_of_interest(indta, replace = FALSE)
```

Arguments

search_categories	Grep string of categories to return.
use_default	(Default TRUE) use dedault dates if none else found.
startdt	Minimum date for events to be returned.
totoday	(Default: FALSE) Ends last date set returned (if applicable) with totoday if a date, Sys.Date()
indta	data.table with columns as shown in details.
replace	(Default: FALSE) If TRUE, replaces existing dates of interest with new set provided, otherwise replaces/inserts new rows only.

Details

Retrieves default dates of interest given a grepstring of categories. There are a default set of categories provided which may not be up to date. New data passed into fg_update_dates_of_interest() or fg_update_aes() persists across future loads of the package. Any duplicates in the new file will be taken out.

New doi data.frames must have at least three columns:

Column	Meaning
category	Grouping name (string) for a given set of dates of interest
eventid	Character string to be displayed at each event.
DT_ENTRY	Start Date of event
END_DT_ENTRY	Optional end of period to define regimes or ranges of events.

Value

data.table::data.table() of date or date ranges, or 'NULL"if new dates are added.

See Also

fgts_dygraph()

Examples

```
require(utils)
require(data.table)
tail(fg_get_dates_of_interest("fedmoves"),2)
# To add (for example) a new FOMC cut of 50bps on 6/16/2026:
newdoi <-data.table(category="fedmoves",eventid="F:-50",
            DT_ENTRY=as.Date("6/16/2026",format="%m/%d/%Y"))
fg_update_dates_of_interest(newdoi)
# Since this is in the future, we have to make the future now.
fg_get_dates_of_interest("fedmoves",totoday=as.Date("2026-12-31"))
fg_reset_to_default_state("doi")
fg_reset_to_default_state("all")
```

---

fg_eventStudy *Event Studies graphs*

Description

Summarizes and plots moves in data from a given set of event dates. Plots are designed to maintain reasonable aesthetics with as either time series or event dates increase.

Usage

```
fg_eventStudy(indata,dtset,output="path",changeas="diff",
              nbd_back=10,nbd_fwd=20,n_color_switch=5,
              title="Events",maxdelta=+Inf,meltvar="variable",verbose=FALSE)
```

Arguments

indata	A data.frame with at least one date column and multiple numeric columns. If melted, must also contain the character column specified in parameter meltvar#'
dtset	A list of dates or a data.table with a column of (event) dates and a character column with unique names for each date.
output	(Default path) : String with type of output desired. Choices are shown below by category:

- data,summary,stats : data.table with eVent moves by asset and business day relative event, a summary of events by asset and eventid, or statistics relative to a crossing of events and assets.

- path, pathbyvar, pathbyevent Show paths of time series moves, by both events and time series, just by time series, or just by event.
- lmbyvar, lmbyevent Show paths of time series moves by time series, or by event, but include linear regression of move ~ time.
- loessbyvar, loessbyevent Show paths of time series moves by time series, or by event, but smoothed loess curves of move ~ time.
- medbyvar,medbyevent Median moves by time series or by event.
- box, boxbyvar,boxbyevent Box plots of moves by both events and time series, just by time series, or just by event.
- scatter Scatter plot of cumulative move from event-nbd_back vs event+nbd_fwd, with medians and regions of movement.

changeas	(Default diff) Character string in c("diff","return","returnbps") describing how changes are displayed. Log returns are used.
nbd_back	(Default 10) Positive integer for number of days prior to event are considered.
nbd_fwd	(Default 20) Positive integer for number of days after event are considered.
n_color_switch	(Default 5) A positive integer after which colors are displayed as gradients instead of separate colors. See Examples.
title	Character string for title of graph
maxdelta	(Default +Inf) Integer to cut off the number of days forward shown, useful if you want to calculate full period statistics.
meltvar	(Default variable) Name of column describing distinct time series if indata is in long (melted) format,
verbose	(Default FALSE) Print Progress of calculations.

Details

Event Studies

Value

a ggplot() object with the events analysis requested by output parameter, or a data.frame with statistics if output in c("data"","summary","stats")

Examples

```
dtset <- fg_get_dates_of_interest("fedmoves",startdt="2024-01-01")[,.(DT_ENTRY,text=eventid2)]
fg_eventStudy(yc_CMSUST,dtset,title="Fed Cuts",output="stats")
fg_eventStudy(yc_CMSUST,dtset,title="Fed Cuts",output="pathbyevent")
fg_eventStudy(yc_CMSUST,dtset,title="Fed Cuts",output="medbyvar")
fg_eventStudy(yc_CMSUST,dtset,title="Fed Cuts",output="lmbyvar",n_color_switch=0)
fg_eventStudy(yc_CMSUST,dtset,nbd_back=3,nbd_fwd=10,title="Fed Cuts",output="boxbyvar")
fg_eventStudy(yc_CMSUST,dtset,title="Fed Cuts",output="scatter")
```

`fg_findTurningPoints` *Add turning points to time series graphs*

Description

Event Helpers : fg_findTurningPoints

Usage

```
fg_findTurningPoints(
  indta,
  rtn = "dates",
  method = "pctchg",
  npts = 10,
  pts_of_interest = "change",
  pctabovemin = 0.05,
  maxwindow = -1,
  addlast = FALSE,
  cpmmethod = "GLR",
  ...
)
```

Arguments

`indta`	Time series `data.table` with a date as the first column and a value series as the second column, or a `prophet::prophet()` object
`rtn`	string with what to return ('dates','data','all')
`method`	string describing method of finding Turning Points • `"pctchg"` : (Default) Find `npts` largest percentage changes with a miniumum window between them • `"cpm"` : Uaw cpm-package to find change points
`npts`	Number of change points to find
`pts_of_interest`	string in 'change' (default) or 'value'
`pctabovemin`	Minimum percentage change to look for.
`maxwindow`	Integer (default -1) which limits (if positive) the minimum number of observations between change points.
`addlast`	Logical (default: FALSE) to add an event with final observation.
`cpmmethod`	String (default: "GLM") passed to `cpm::processStream()`
`...`	Additional parameters passed to cpm-package

Value

`data.table` suitable for passing into `fgts_dygraph()` via the `event_ds` parameter

Examples

```
dta <-eqtypx[,.(date,QQQ,TLT)]
fgts_dygraph(dta,event_ds=fg_findTurningPoints(dta),title="With turningPoints")
```

`fg_get_aes` *Get aesthetic sets used in graphs*

Description

`fg_get_aes()` gets aethestic `data.frame` for use in graphs. `fg_get_aesstring()` takes a column from the `data.frame` retrieved by `fg_get_aes()` `fg_print_aes_list()` prints names of aesthetics used internally in FinanceGraph functions. `fg_display_colors()` Shows a plot with current colors. `fg_get_datemap()` returns a data.table of all dates and date mappings `fg_current_theme()` returns the current `ggplot2` theme used for scatter plots.

Usage

```
fg_get_aes(item = "", n_max = NA_integer_, asdataframe = FALSE)

fg_get_aesstring(
  item = "",
  n_max = NA_integer_,
  toget = "value",
  rtnifnotfound = FALSE
)

fg_display_colors(item = "")

fg_print_aes_list(grepstr = "")

fg_get_datemap()

fg_current_theme()
```

Arguments

`item`	(Default: "") A grep string for categories desired.
`n_max`	Maximum number of rows or entries to return. Required for `Rcolorbrewer` color aesthetics
`asdataframe`	(Default: FALSE) Return dataframe of parameters regardless of type. (See details)
`toget`	Column in the aes `data.frame` to paste together as a string.
`rtnifnotfound`	Return `NA_character_` if aes not found
`grepstr`	narrow list of internal aesthetics sets to functions from `grepstr`

Details

Maintain Colors

Value

`fg_get_aes()` returns `data.frame` of aesthetics, including sorting columns, help strings, and values, `fg_get_aesstring()` returns a list with just the character values of the requested aesthetic. `fg_print_aes_list()` returns a markdown ready character vector of aesthetic names used in each function `fg_display_colors()` returns a `ggplot2::ggplot()` object with colors and associated names for an aesthetic name

See Also

`fgts_dygraph()`, `fg_update_aes()`

Examples

```
# Data set, String
head(fg_get_aes("lines"),3)
fg_get_aesstring("lines")
#  Gradient colors are stored in a `data.frame` as in a set of "Blue Greens"
fg_get_aes("espath_gp",asdataframe=TRUE)
# To get the actual colors, we need to know how many:
fg_get_aes("espath_gp",n_max=8)
fg_display_colors("lines")
```

`fg_prophet` *Add forecasts from the prophet package*

Description

`fg_prophet` Augments a `prophet::predict.prophet()` output into `fgts_dygraph()` `forecastdataset` format.

Usage

```
fg_prophet(prophet_data, seriesname = "y")
```

Arguments

`prophet_data`	Data resulting from a `prophet::predict.prophet()` call
`seriesname`	(Default: `"y"`) Series name to attach forecast to

Details

Note that `prophet::predict.prophet()` loses the name of the series, the

Value

`data.table` suitable for passing into `fgts_dygraph()` via the `forecastdataset` parameter

Examples

```
if (requireNamespace("prophet", quietly = TRUE)) {
p_model <- eqtypx[,.(ds=date,y=QQQ)] |> narrowbydtstr(dtstr="-1y::") |> prophet::prophet()
p_fcst <- predict(p_model,prophet::make_future_dataframe(p_model,periods=60))
fgts_dygraph(eqtypx[,.(date,QQQ)],title="With Prophet Forecasts", roller=1,dtstartfrac=0.8,
      forecast_ds=fg_prophet(p_fcst,seriesname="QQQ"))
}
```

`fg_ratingsEvents` *Add colored ratings levels to time series graphs.*

Description

Event Helpers : fg_ratingsEvents

Calls `tidyquant::tq_get()` to get dividends for a given set of tickers. A previously created `data.frame` can also be input.

Created `event_ds` from alphavantagepf::av_get_pf quarterly earnings data.

Usage

```
fg_ratingsEvents(credit, ratings_db, agency = "S.P")

fg_tq_divs(tickers, divs_ds = NULL, ticker_in_label = TRUE)

fg_av_earnings(indt, field = "reportedEPS", ticker_in_label = FALSE)
```

Arguments

`credit` String with name of credit to look up in 'ratings_db'

`ratings_db` A 'data.table' or 'data.frame' with the all of the following columns:

column	description	type
CREDIT	Name of credit	character
AGENCY	Name of ratings agency	character
RATING	Rating assigned	character
WATCH	Watch denoted by anything with "+" or "-" in the string	character
DT_ENTRY	Date which ratings or ratings change was issued	Date

`agency` String (default 'S.P') with 'AGENCY to look up in 'ratings_db'

`tickers` List of tickers to get dividends for.

divs_ds	Alternatively a data.frame previously obtained using tidyquant::tq_get() with columns (symbol,date,value)
ticker_in_label	(Default: TRUE) Make label ticker and the earnings
indt	data.frame obtained from alphavantage earnings data.
field	(Default: reportedEPS) String in (reportedEPS,estimatedEPS,surprise,surprisePercentage)

Details

Investment grade ratings are shaded in blue, High Yield are in red. Darker areas are closest to the cutoff between the two.

Event Helpers: fg_tq_divs

Event Helpers: fg_av_earnings

Value

data.table suitable for passing into fgts_dygraph() via the event_ds parameter

data.table suitable for passing into fgts_dygraph() via the event_ds parameter

data.table suitable for passing into fgts_dygraph() via the event_ds parameter

Examples

```
data("nomfxdta")
copdta <- nomfxdta |> dplyr::filter(variable=="COP")
fgts_dygraph(copdta,title="COP with Ratings",dtstartfrac=0.3,
        event_ds=fg_ratingsEvents("COLOM",ratings_db,agency="S.P"))

if (requireNamespace("tidyquant", quietly = TRUE)) {
 fgts_dygraph(eqtypx,title="With divs",dtstartfrac=0.8,event_ds=fg_tq_divs(c("IBM","QQQ")))
}
if (requireNamespace("alphavantagepf", quietly = TRUE)) {
earnings = alphavantagepf::av_get_pf("IBM","EARNINGS") |>
        alphavantagepf::av_extract_df("quarterlyEarnings") |>
        fg_av_earnings()
toplot = dplyr::select(eqtypx,date,IBM)
fgts_dygraph(toplot,title="With earnings",dtstartfrac=0.8,event_ds=earnings)
}
```

---

fg_RegimeChange *Add Regime change events to time series graphs*

Description

Wrapper around the function RegimeChange::detect_regimes() to create events for fgts_dygraph()

Usage

```
fg_RegimeChange(indta, usereturns = TRUE, series = NULL, ...)
```

Arguments

indta	Time series data.table with a date as the first column and a value series as the second column.
usereturns	(default TRUE) LOgical to take log returns before changepoint calculations.
series	Which series in indta to apply changepoints to
...	Parameters passed to RegimeChange::detect_regimes()

Details

Change points with no avavilable confidence intervals will be shown, but not colored.

Value

data.table suitable for passing into fgts_dygraph() via the event_ds parameter

Examples

```
if (requireNamespace("RegimeChange", quietly = TRUE)) {
dta <- tail(eqtypx[,.(date,QQQ,TLT)],260)
eventdt = fg_RegimeChange(dta,bootstrap_reps=30)
fgts_dygraph(dta,event_ds=eventdt,title="With Breakouts")
}
```

---

fg_scatplot *Flexible and easy scatterplot generator, with time specific enhancements*

Description

Plots bivariate plots with some time-series specific enhancements. Rather than programmatically describing graph aesthetics, a simple formula-based approach is used. This approach allows quick specification of many customizaton options.

Usage

```
fg_scatplot(indata,plotform,type="scatter",datecuts=c(7,66),
                noscales="",xdecoration="",ydecoration="",annotatecorners="",
           tsize=3,psize=1,n_color_switch=7,n_hex_switch=400,repel=TRUE,jitter=c(0,0),
                title="",subtitle="",caption="",axislabels="",
           boundbox=c(),boundboxtype="",gridstyle=NA_character_,legendinside=FALSE,
                tformula=formula("y~x"),returnregresults=FALSE,
                keepcols="", meltvar="variable",melted=NULL)
```

Arguments

indata	data.frame with columns for (x,y) coordinates and possibly other categorical data or a date column. Alternatively, indata can be in long format with meltvar present. Note that aesthetic characteristics (if used) must be present for both long and wide input formats.
plotform	A text formula describing how to set up the graph. The formula is of the general form y ~ x + option:<column_name>,<aesthetic category> + ... where y is plotted against x and aesthetics for each point are controlled by one or more option clauses each followed by zero or more optional parameters. If the option applies to all points then the first parameter column_name must be in indata. By default, points or symbols are plotted. By general category, the options are Aesthetic options: • color:column<,aes_set> sets the color of each point or label by data in column • symbol:column<,aes_set> sets the symbol or shape cof each point or label by data in column • size:column<,aes_set> sets the size of each point by the data in column Date specific options: • doi:recent partitions data by number of days in datecutsprior to the last day in indata • doi:<doiset> partitions data by date ranges obtained from dates of interest set <doiset>. See fg_get_dates_of_interest() • point:<value|label|anno><all> adds highlights for either the last date in indata or the last date for each group (if all). value gives coordinates, label the label in the color column, while anno adds lines to each axis. Text options: • [text|label|labelhilight|tooltips]:column<,aes_set> : Plots the text in character column as text (without border), label (with border), filled in label, or mouse-over tooltip. (See details) Other annotations: • ellipse adds an ellipse around the points using ggplot2::stat_ellipse() • hull<:quantile> draws a convext hull around points with <quantile> (default 0) points removed. (See details) • xline<:level=0> draws a vertical line at level • yline<:level=0> draws a horizontal line at level • grid:<dotted|dotted_x|dotted_y|none> formats background grids
type	character string for the type of graph to plot: • scat plots points, text or labels • lm<one><noeqn><nofill> adds linear regression lines per color category or across all points (lmone). • loess<one><noeqn>adds loess line per per color category or across all points (loessone)

- density Creates a density plot

if noeqn is part of the string, equations are suppressed. If one is part of the string, no subcategories are used. 'nofill" removes confidence bands.

datecuts — list of integers (Default c(7,66) for days prior to last date to make date classes. (See examples and doi:recent as above.)

noscales — String to suppress guides with any of <color|size|symbol>

xdecoration, ydecoration — 2 element string list to add to either side of an axis label.

annotatecorners — 4 element string list to add notes to each of 4 quadrants of the graph. See examples.

tsize — default text size (with some scaled variations for graph parts such as titles)

psize — default point size.

n_color_switch — (Default 7): Number of distint color categories beyond which colors are taken from gradient scales, unless a color set is specified in a color part of plotform

n_hex_switch — (Default 400): Number of data points beyond which points are replaced with binned hexagons (see geom_hex())

repel — (Default TRUE) Text and labels are plotted using ggrepel()

jitter — Jitter parameters used by geom_text() or geom_label() if repel=FALSE. Default is no jitter.

title — Title to add to graph

subtitle — Subtitle to add to graph

caption — Caption to add to graph

axislabels — Semicolon separated string with x and y labels, e.g date;OAS

boundboxtype, boundbox — string describing how to use bounding boxes. If "identify" is part of the string, then data is truncated to the calculated bounding box and notations to that effect are added to the graph. If not, then points outside the box are dropped.

- prob|probidentify calculates bounding boxes from quantilies of x and y data. See vignette for details.
- value|valueidentify are minimum and maximum values of x and y axes to show. If boundbox is a list of two numbers, the y axis is truncated to those values, If boundbox is a list of 4 numbers c(xmin,xmax,ymin,ymax) data is truncated to that box.

gridstyle — String in <dotted|dotted_x|dotted_y|none> to contol grids as in grid option above.

legendinside — (Default: TRUE) Put all guides inside the graph.

tformula — (Default y~x) Formula used within lm or loess stats .

returnregresults — Return a two element list c(plot,regression data.frame). Only available for linear models, and uses the first amoung options c("color","symbol","size","alpha") as grouping variables

keepcols	list of indata columns to be kept with the graph data, useful for further faceting using ggplot2::facet_wrap()
meltvar	(Default "variable") If indata is melted, then this is used to create x and y categories.
melted	(Default:NULL) If FALSE forces data not to be melted if meltvar in indata

Details

fg_scatplot: Easy scatterplot generator, with time specific enhancements

indata can either be in wide ('date' ,'series1',...) format or normalized (long) format ('date','variable','value',...) format. This package infers date columns names from column types and casts or pivot_wider to get x and y columns. Note that aesthetic characteristics (if used) must be present for both long and wide input formats.

Default aesehetic sets used for portions of the graph each have their own names, which can be seeing by running fg_print_aes_list() and modified or added to using fg_update_aes(). The default theme can be modified using fg_replace_theme(). Both aesthetic changes and theme changes are persistent across R sessions.

Use of doi: in plotform string supercedes color aesthetics otherwise specified. (Is this true?)

If tooltips are used, the result of fg_scatplot must be viewed using print(girafe(ggobj=fg_scatplot(...))) See ggiraph::geom_point_interactive()

Winsorized hulls with quantile cutoff q are formed using the closest (by euclidean distance) 1-q points to the geograpic center of the entire set.

Captions are added if data is truncated or omitted by the bounding box procedure.

Value

A ggplot2::ggplot() object with desired graph, or a ggiraph::girafe() object if tooltips is in the plotform string.

Examples

```
# Simple text examples
require(data.table)
dt_mtcars=data.table(datasets::mtcars)
dt_mtcars$id=lapply(rownames(datasets::mtcars),\(x) last(strsplit(x," ")[[1]]))
fg_scatplot(dt_mtcars,"disp ~ hp + color:am + text:id","scatter",title="text basic")
fg_scatplot(dt_mtcars,"disp ~ hp + color:carb + label:id","scatter",
               n_color_switch=0,title="scat color switch")
# Plotting data with dates:
set.seed(1); ndates <- 400;  dlyvol <- 0.2/sqrt(260)
rtns <- cbind(cumsum(rnorm(ndates,sd=dlyvol)),cumsum(rnorm(ndates,sd=dlyvol)))
dttest <- data.table(date=seq(as.Date("2021-01-01"),as.Date("2021-01-01")+ndates-1),
            xtest=100*(1+rtns[,1]),ytest=100*(1+rtns[,2]),
            ccat=fifelse(runif(ndates)<=0.2,"Rare","mkt"))
# Making categories out of recent data
fg_scatplot(dttest,"ytest ~ xtest + doi:recent","scatter",datecuts=c(66,122),title="from recent")
fg_scatplot(dttest ,"ytest ~ xtest + color:ccat + doi:recent + point:label","scat",
              datecuts=c(7,66),title="recent w label")
```

```
# Makes categories out of event sets from [fg_get_dates_of_interest()]
fg_scatplot(dttest,"ytest ~ xtest + doi:regm","scatter",title="from a regime")
# Point graphing switches.
fg_scatplot(dttest,"ytest ~ xtest + color:ccat","lm",n_hex_switch=100,title="Hex Switch")
# Quick changes to aesthetic sets
fg_scatplot(dttest,"ytest ~ xtest + color:ccat,altlines_6","loess",title="Alternate colors")
# Extra summarizatons
fg_scatplot(dttest,"ytest ~ xtest + color:ccat + hull:0.1 + ellipse","lm",title="Curves")
# Annotations
fg_scatplot(dttest,"ytest ~ xtest + color:ccat + point:labelall","scat",title="Last Values")
fg_scatplot(dttest,"ytest ~ xtest + color:ccat + point:anno","scat",annotatecorners="NW;NE;SE;SW",
                legendinside = FALSE)
```

`fg_seasonalstudy` *Flexible Seasonality Graphs*

Description

Flexible and general seasonality graphs.

Usage

```
fg_seasonalstudy(
  indta,
  seasonaltype = "yr",
  seasonaldateset = NULL,
  day_offset = 0,
  bdaysonly = TRUE,
  graphtype = "flex",
  normalize = "",
  projectfwd = "",
  projfwd_wt = 0.9,
  yvar = "value",
  title = NULL,
  yrange = NULL,
  n_color_switch = 7,
  n_hex_switch = 20,
  line_aes_set = "lines",
  line_on_lastdate = TRUE,
  killbad_eop = FALSE,
  return_dates = FALSE
)
```

Arguments

`indta`	A data.frame with at least one date column and a numeric column with the name given by `yvar`

`seasonaltype` (default: `"yr"`) A string denoting the periodicity of the analysis, must be in one of `c("yr","qtr","mo","wk",IMMroll","optmo","optqtr")`. Not used if `seasonaldataset` is specified instead

`seasonaldateset` (default:NULL) An optional dataset with two columns: A Date column defining each period, and possibly a character variable with a period identifier.

`day_offset` (default: 0) Number of days (determined by bdaysonly) to offset each period. Applies only to `c("yr","qtr","mo","wk")`

`bdaysonly` (default:TRUE) Only consider NYSE business days.

`graphtype` (default: "flex") One of the following

graphtype	Description
`line`	A line for every period, with identifiers placed near the end
`hex`	A density plot for each day of the seasonal period, with the last seasonal period kept as a line
`flex`	A line graph if there are less than `n_hex_switch` seasonal periodss
`stat`	A line graph showing 10tyh, 50th, and 90th percentiles of values for each day in a seasonal period

`normalize` (default ""). How to normalize each period. Default is no normalization. Other options are

normalize	Description
`relative`	Older seasonal periods adjusted to match beginning of latest period
`index`	Each seasonal period is expressed as index from beginning of each period

`projectfwd` (default "") Project forward based on same number of days in seaonal period. Default is no projection.

normalize	Description
`mean`	Use mean (by days in seasonal period) of each cumulative percentage change from the start of the period
`weighted`	Weight previous period cumulative percantge change with an expoential decay using `projfwd_wt` ** (Periods ba

`projfwd_wt` See above

`yvar` (default:`"value"`) Series to use in `indta`

`title` (default: NULL) Title for graph.

`yrange` (default: NULL) y axis Range for which to focus data.

`n_color_switch` (default 7) Number of periods past which lines will be colored by a descending scale.

`n_hex_switch` (default 20) Number of periods past which a `hex` graph will be used, if `flex` is chosen above.

`line_aes_set` (default `"lines"`) Aes color set for discrete lines.

`line_on_lastdate` (default TRUE) Add a vertical line at the last observations day in period.

killbad_eop	(default FALSE): DO not show periods for which there are at least 60% of the mean number of observations per day of the seasonal period. If used, this helps to curtail extreme moves at the end of a period. For example, this would redact the 366th day of the year.
return_dates	(default: FALSE) Return list(graph,dates) instead of just the graph.

Details

Seasonality Graphs

Value

a ggplot() object displaying seasonality

Examples

```
require(data.table)
assetcols <- c("EEM","IBM","QQQ","TLT")
eqtyidx<-eqtyrtn[,(assetcols):=lapply(.SD,\(x) 100*(exp(cumsum(fcoalesce(x,0))))),
            .SDcols=assetcols]
fg_seasonalstudy(eqtyidx,yvar="TLT",seasonaltype="yr",normalize="index",projectfwd="mean")
# Earnings seasons
earnings_dates <- earnings_ibm[,.(reportedDate,divdt=format(reportedDate,"%Y%m%d"))]
fg_seasonalstudy(eqtyidx,yvar="IBM",seasonaldateset = earnings_dates,graphtype="stat",
   normalize="index",projectfwd="mean")
```

---

fg_signal_to_events *Add single date signals to time series graphs.*

Description

Event Helpers: fg_signal_to_events

Usage

```
fg_signal_to_events(signal_df, colormap)
```

Arguments

signal_df	A two-column data.frame with first being a date and second being any (factor-like) signal parameter.
colormap	A two column data.frame with the first being the possible signal (see Example) and the second a color. description

Details

This helper applies run-length encoding to match the signal in signal_df to the color in colormap

Value

data.table suitable for passing into fgts_dygraph() via the event_ds parameter

Examples

```
# A simple moving average strategy with threshold
require(data.table)
ma_signal<-eqtypx[,.(date,sig=cut(frollmean(EEM,5)-frollmean(EEM,20),
                    c(-10,-0.5,0.5,10),labels=c("long","flat","short")),EEM)]
colormap<-data.frame(sig=c("long","flat","short"),color=c("#f56462","white","#6161ff"))
fgts_dygraph(eqtypx[,.(date,EEM)],event_ds=fg_signal_to_events(ma_signal,colormap),
    dtstartfrac=0.8,roller=1,title="5/20 MA positions")
```

---

fg_sweep *Add forecasts from the sweep package*

Description

fg_sweep Augments a sweep::sw_sweep() output into fgts_dygraph() forecastdataset format. See Introduction to Sweep

Usage

```
fg_sweep(swept_data, confidence = 80)
```

Arguments

swept_data	Data resulting from a sweep::sw_sweep() call
confidence	(Default: 80) Confidence interval (in percent) to display

Details

Forecast Helpers

Value

data.table suitable for passing into fgts_dygraph() via the forecastdataset parameter

Examples

```
if (
    requireNamespace("forecast", quietly = TRUE) &
    requireNamespace("timetk", quietly = TRUE) &
    requireNamespace("sweep", quietly = TRUE)
) {
fcst_eqtypx <- timetk::tk_ts(eqtypx[,.(date,QQQ)]) |> forecast::ets() |>
      forecast::forecast(h=30) |> sweep::sw_sweep(timetk_idx=TRUE)
fcst_in <- fg_sweep(fcst_eqtypx)
```

```
toplot <- eqtypx[,.(date,IBM,QQQ)]
fgts_dygraph(toplot,title="With Forecasts", roller=1,dtstartfrac=0.7,forecast_ds=fcst_in)
}
```

`fg_sync_group` *Add a synchronizing identifier for dygraphs used in Shiny or Markdown contexts*

Description

Sets, gets, or resets a common name to be passed into `fgts_dygraph()` for synchronization.

Usage

```
fg_sync_group(gpname = "")
```

Arguments

gpname	A string or NULL

- `gpname=NULL` turns off dygraphs synchronization.
- `gpname=<string>` set the common group to `<string>`
- `gpname=""` (Default), just returns the current common group name.

Details

Group Synchronization

Use thie to set a common groupname for time scale synchronization (for Markdown or shiny apps), Only set it in the beginning, or when needed, and call with NULL to turn synchronization off.

Value

current groupname

See Also

`fgts_dygraph()`

Examples

```
fg_sync_group()
fg_sync_group("common")
fg_sync_group()
fg_sync_group(NULL)
```

`fg_tsboxplot` *Boxplots to summarize time series*

Description

Plots static summaries of time series in boxplot form.

Usage

```
fg_tsboxplot(indt,title="",xlab="",ylab="",
     breaks=c(7,30,90,360), doi="last", normalize="", orderby="",
     boxtype= "",
     dropset="",hilightcats="",
     addline="", #last/mean
     facetform="",
     ycoord=NULL,trimpctile=0,
     legend="insidetop",meltvar="variable",flip=FALSE,ptsize=3)
```

Arguments

`indt`	Input data.frame with at least one date variable and one or more vategorical variables, if melted.
`title`, `xlab`, `ylab`	TItles and Labels
`breaks`	A list or text as follows • `<doiset>` : A dates of interest category, see `fg_get_dates_of_interest()` • list of integers: A list of days for which to go back in time, e.g. `c(7,30,360)` creates intervals for the last week, 1 week to 1 month, etc. • list of reals in `[0,1]` fractions of the dates in each category, e.g `c(0.2,1)` creates intevals with the last 20pct of dates, and any older.
`doi`	Points or segments to overlay with latest observations, or changes since a particular date. • `"last"` (Default) Last date as a dot • `"last,<d1>"` Segment from date `d1` to last date in input data. • `"last,n"` Segment from nth date from the end to the end. • `"date,<d1>"` Levels as of date `d1` • `"none"` No points or segments.
`normalize`	Normalize data in some way prior to plotting. Choices are • `"byhistcat"`Transform data into percentiles within each variable and historical category • `"byvar"`,`"zbyvar"`Transform data into percentiles (or z-scores) within each variable and historical category
`orderby`	(Default `""`) Underlying categories are by default ordered as in `indt`, unless

- "value","-value" : Order by last value in series for each category or descending if "-value"
- "date,<d1>","-date,<d1>" : Order by value (or decreasing value) at date <d1>
- "alpha","-alpha" Order alphabetically in ascending or descnding orer.

boxtype	Formatting of boxplots. If in "violin,"viobycat" make a violin plot, otherwise show a full boxplot (with outliers turned off by default), with any aspects in c("nostaple","nomedian","nobox") taken out.
dropset	String or list with underlying categories to drop from graph
hilightcats	String or list of underlying categories to highlight with differnt color in label.
addline	in c("mean","last") Add a horizontal line across the mean of all observations or a smooth like across last observations
facetform	(Default: "") Any faceting formula which includes text or factor columns in indt. See examples and note that facets can also be added using ggplot2::facet_grid() to the output graph.
ycoord	(Default NULL) a two element list with limits on y corrdinates
trimpctile	(Default 0) trims data before any plotting to fall within c(trimpctile,1-trimpctile) percentiles within each variable.
legend	(Default "insidetop") Where to put the legend
meltvar	(Default: "variable" Name of variable with unit category.
flip	(Default FALSE) If TRUE then categories are arranged vertically
ptsize	(Default: 3) Size of points for doi parmeter

Details

fg_tsboxplot: Boxplots of time series

Value

A ggplot2::ggplot() object

Examples

```
fg_tsboxplot(eqtypx,breaks=c(7,30,360),normalize="byvar",hilightcats="QQQ",
       title="Equity prices, within ranges")
fg_tsboxplot(narrowbydtstr(eqtypx,"-2y::"),breaks="regm",normalize="byvar",
        hilightcats="QQQ",title="Equity prices, in regimes")
fg_tsboxplot(reerdta,breaks=c(0,0.2,0.5,1),doi="last",orderby="value",
        boxtype="nowhisker",facetform=". ~ REGION",title="Real Eff. Exch Rates")
fg_tsboxplot(reerdta,breaks=c(0,0.2,0.5,1),doi="last",orderby="value",
        addline="last",boxtype="violin",title="Real Eff. Exch Rates (Violin)")
```

fg_update_aes *Add aesthetic parameter sets to internal store.*

Description

fg_update_aes() updates or replaces default aesthestics (e.g. colors, linestyles, etc). fg_update_line_colors() replaces line colors only fg_reset_to_default_state() resets colors and/or dates of interest fg_replace_theme() Replaces default theme used in static plots fg_verbose() Toggles printing of aesthetics

Usage

```
fg_update_aes(indta, aestype = NA_character_, persist = TRUE, replace = FALSE)

fg_update_line_colors(colorlist, replace = FALSE, persist = TRUE)

fg_replace_theme(newTheme, persist = TRUE)

fg_verbose(item = "")

fg_reset_to_default_state(reset = "all")
```

Arguments

indta	data.table aesthetic data.fram with columns as shown in details.
aestype	(Default: NA) character string with type of aesthetic requested. If not provided in [fg_Update_aes()] the
persist	(Default: TRUE) Keep changes across invocations of the package.
replace	(Default: FALSE) Replaces existing dates of interest with new set provided, otherwise replaces/inserts new rows only.
colorlist	List with up to 14 new colors just for line (series) coloring
newTheme	A new ggplot2 theme
item	(Default: "") A grep string for categories desired.
reset	(Default: "all"), options in ("all","colors","doi") to reset to defaults with the package.

Details

Maintain Colors

For colors, New data passed into fg_update_aes() persists across future loads of the package unless persist=FALSE. New color datasets must have at least three columns:

Column	Meaning
category	Arbitrary aestehtic category, e.g. "lines" for line colors.

variable	Any string that can be sorted or grepped to map to data.
type	Aesthetic type, in c("color","colorrange","linetype","symbol","alpha")
value	String with value detired (e.g a color)

variable is used to prioritize colors, so (e.g. D01 will be the color of the first series in an input dataset)

If aestype=="colorrange" then a sequential scale of size n_max will be returned using details saved from fg_update_aes(). See scales::brewer_pal and colorbrewer

Value

No return value, as these are called for the side effects of adding to or replacing aesthetic sets.

See Also

fgts_dygraph(), fg_scatplot(), fg_get_aes()

Examples

```
# Data set, String
head(oldcolors <- fg_get_aes("lines"),3)
# then change as needed.  For example, to make the second line blue, and the 4th line red,
oldcolors[c(2,3),"value"] <- c("blue","tomato")
fg_update_aes( oldcolors )
head( fg_get_aes("lines"),3)
# to create a new category, make a similar `data.frame`, as in
newcolors <- data.frame(category=rep("mylines",3),variable=c("D01","D02","D03"),
                value=c("red","black","green"))
fg_update_aes( newcolors, aestype="color")
fg_get_aesstring("mylines")
#Theme replacement
require(ggplot2)
fg_replace_theme(ggplot2::theme_dark(),persist=FALSE)
fg_reset_to_default_state("all")
```

---

fgts_dygraph *Time series graphs using* dygraphs *package*

Description

Plots interactive time series graphs with many options for highlighting key events, regions and customizations.

Usage

```
fgts_dygraph( indata,
 title = "", xlab="", ylab = "", roller = "default", bg_opts = "hair,both;grid,both",
 splitcols = FALSE, stepcols = FALSE, hidecols = FALSE, hilightcols = FALSE,
 hilightwidth = 2, hilightstyle = "solid",
 events = "", event_ds = NULL,
 annotations = "", annotation_ds = NULL, forecast_ds = NULL,
 ylimits = NULL, dtstartfrac = 0, dtwindow = "", rebase = "", exportevents = FALSE,
 meltvar = "variable", dylegend = "always", fillGraph = FALSE, colorset="lines",
 groupnm = fg_sync_group(), verbose = FALSE,  extraoptions = list() )
```

Arguments

indata	Input data in long or wide format. THere must be at least one date column, one character column and one numeric column. Ideal format is `date,variable,value`
title	Title to put on top of graph
xlab, ylab	Labels for x and y axis
roller	Initial moving average value to smooth graphs. (See `dygraphs::dyRoller()`) Options are • `default` (Default) chose a smoothing parameter consistent with the length of the input series • `finest` No smoothing • integer >= 0 : User specified moving average length.
bg_opts	Semicomma separated options to change interactivity and background of charts. These semicomma separated options change pointer option and grids options. • `hair,<style>` passes `<style>` to `dygraphs::dyCrosshair()`. Default is "both" x and y crosshairs. • `grid,<x,y,both>` specifies which gri lines to show. Default is "both" • `norange` turns off date range selector.
splitcols, stepcols, hidecols	String or list of data series to show on a second y axis, to be shown as step plots, or to be hidden. Can also be `TRUE` in which case first series in the data is used Can also be a semicolon separated single string with mutiple series. `splitcols` is ignored if only one series is plotted.
hilightcols	String or list of data series to plot in different style than other series.
hilightwidth	(Default: 2) relative width of series specified in `hilightcols`
hilightstyle	(Default: solid). Line style of series specified in `hilightcols`. Options are (`solid,dashed,dotted,dotdash`)
events	String with possible events to add to graph. Options can be added together with ; and include • `doi,<eventsetname>` : Events in internal event list `eventsetname` from list maintained by `fg_update_dates_of_interest()`. • `seasonal,<type>` : Regularly spaced intervals of dates. See details below. • `minmax` : Locations of highest and lowest observations per series.

- dt,text,d1,<d2> : Text events starting at d1 and possibly ending at <d2>,both of the form yyyy-mm-dd. See details for adjustments.
- pt,d1,series,text : Text annotation for series at date d1 See details for adjustments.
- break,labelform : Breakouts as determined by fg_addbreakouts() with labelform in ("singleasdate","singleavalue","breakno")
- tp,n : Turning points on the first series as determined by fg_findTurningPoints()

event_ds — data.frame of events to be added to graph. See details and examples for specification.

annotations — string with annotations on individual series or along y axes. Options can be added together with ; and can include

- last,[value|label] : Value or name of latest observation for each series, placed at the end of the series
- last,[linevalue|linelabel] : Value or name of latest observation for each series, placed at the end of the seriesName of each series, placed at the end of the series.
- hline,y : Horizontal line at y‘
- range,ybeg,yend : Band placed between ybeg and yend

annotation_ds — data.frame of annotations added to graph. See details for specification.

forecast_ds — data.frame of forecasts to be displayed after the end of those in indt. Those typically are in wide format, with at minimum a (first) date column and series names of the form series.f``, series.flo and/or `series.fhi, where series is one of the plotted series in indt

ylimits — Two number vector of lower and upper limits of data to be displayed. Alternatively, a string of the form <seriesnm>,<q> will limit displayed data to the (q,1-q) quantiles of seriesnm

dtstartfrac — Fraction in (0,1] of dates in indt to start the range selector. See dygraphs::dyRangeSelector()

dtwindow — String to specify date ranges applied dygraphs::dyRangeSelector() of the form begin::end where either end can take the form "yyyy-mm-dd" or a relative date to the other end of the series, e.g -3m or -2w. Example: "-3m::-1m" defines a 2 month period 1 month back from the end of the series.

rebase — String of the form yyyy-mm-dd,<value> with <value> assumed 100 if not specified. This normalizes all series to <value> as of the given date. See examples.

exportevents — (Default: FALSE) Return list of the form c(<graph>,<event dataframe>) instead of just the graph.

meltvar — (Default: variable) Column name in indt with series names, if melted.

dylegend — (Default: "auto") Passed to dygraphs::dyLegend(), can be one of ("auto", "always", "onmouseover", "follow", "never")

fillGraph — (Default: FALSE) Shade area underneath each series.

colorset — (Default: "lines") Set of default colors to use. See Customization vignette .

groupnm — (Default: NULL, unless set via fg_sync_group()) Group name used in shiny or RMarkdown to synchronize graphs. See fg_sync_group() for details.

verbose — (Default: FALSE) Print extra details about what will be graphed.

extraoptions — Additional options passed to dygraphs::dyOptions()

Details

TIme series in Dygraph form

Input data can either be in wide ('date' ,'series1',...) format or normalized (long) format ('date','variable','value') format. This package infers date columns names from column types and seeks to be as agnostic as possible as to column names. Colors can be managed using `fg_update_aes()` and will persist across R sessions, See vignette for details. Series are grouped together into bands around a series `series` if their names end as in 'series.lo' or 'series.hi'. See examples and vignette for details.

Events are dates and date ranges to be highlighted in the graph. Multiple types of events can be strung together in semicolon delimited strings. Of the options outlined above, two additional details are

- `"doi,<category>"` gets events from `fg_get_dates_of_interest()` which can be added to or managed using `fg_update_dates_of_interest()`. Colors and label placement can be customized as necessary.
- `seasonal,<type>` puts regularly occurring events on the graph. `<type>` can be

<type>	description
"optex,mo|qtr"	Monthly and/or quarterly equity option expiration dates.
"roll"	IMM CDS roll dates
"daysfromroll"	Dates with same number of days to the next roll as last date plotted
"doq","doy","bdoy"	Dates with same day in quarter, in year, or business day of year to the last day plotted

Events can also be added using a `data.frame` passed via `event_ds` with the following columns:

column	description	type
date	(Required) Start date	Date
date_end	End date to specify range of a colored band	Date
text	(Required) Text to display	character
color	Color for line and text	character
eventonly	Only draw line for for start of event, no band	logical
strokePattern	One of ('solid','dashed' (Default) ,'dotted','dotdash')	character
loc	one of ('top','bottom' (Default))	character
series	Name of series to apply event to, if needed	character
category	Optional string used for exceptions. See notes below.	character

Many times, events depend on outside data or statistical analysis on the original data. The `event_ds` to be passed in can come from event helpers in `fg_cut_to_events()`, `fg_addbreakouts()`, `fg_findTurningPoints()`, or `fg_ratingsEvents()`. Event columns are processed as is, unless `category=="series_color"` which will replace `color` with that of its series.

Annotiations include any notes or highlights added to the graph on the 'y' axis or on an individual series. In addition to those passed via the `annotations` parameter, annotations can be added using a `data.frame` with the following columns:

column	description	type

date	(Required) Start date	Date
date_end	End date to specify range of a colored band	Date
text	(Required) Text to display	character
color	Color for line and text	character
eventonly	Only draw line for for start of event, no band	logical

Other notes:

- Using stepcols most often happens with lower frequency data, so an nafill is automatically performed.
- Dates in event types pt and dt are adjusted to next day in series if they do not already exist.

Value

Dygraph dygraphs plotting input data, with annotations and other customizations.

Examples

```
# See Vignette for more extensive examples.
# Basic Example
fgts_dygraph(eqtypx, title="Stock Prices", ylab="Adjusted Close")

# With series Highlights, finer resolution and focused date range
fgts_dygraph(eqtypx, dtstartfrac=0.8,hilightcols="IBM",hilightwidth=4,roller=3)

# Rebasing to 1/1/2022
fgts_dygraph(eqtypx, title="Rebased Prices", ylab="Adjusted Close",rebase="2022-01-01")

# Using bands (.lo, .hi)
toplot <- reerdta[REGION=="LATAM",.(cop=sum(value*(variable=="COL")),
              reer=mean(value),reer.lo=min(value),reer.hi=max(value)),by=.(date)]
fgts_dygraph(toplot,title="COP REER vs Latam peers",roller=3,hilightcols="cop",hilightwidth=4)

# Events Examples.  Notice how roller shortens with the series.
# See Vignette for more extensive examples
require(data.table)
smalldta <- narrowbydtstr(eqtypx[,.(date,TLT,EEM)],"-3y::")
fgts_dygraph(smalldta,events="doi,regm;doi,fedmoves")
fgts_dygraph(smalldta,events="date,FOMO,2025-01-01,2025-06-01;date,xmas,2025-12-25")

# Events passed in as data.frames
myevents = data.frame(end_date=as.Date(c("2024-03-10","2024-06-10")),
             date=as.Date(c("2024-01-10","2024-04-10")),
             text=c("range","event"),color=c("green","red"))
fgts_dygraph(smalldta,events="doi,fedmoves",event_ds=myevents)

# Annotations on y axis
fgts_dygraph(eqtypx,annotations="last,linevalue")
fgts_dygraph(eqtypx,annotations="hline,100,at100,red;hline,200,at200;range,300,400")
```

```
# use with helpers

smalldta <- narrowbydtstr(eqtypx[,.(date,IBM,QQQ)],"-2y::")
fgts_dygraph(smalldta,title="W TurnPts",event_ds=fg_findTurningPoints(smalldta[,.(date,QQQ)]))
fgts_dygraph(smalldta,title="W Sentiment",event_ds=fg_cut_to_events(consumer_sent,center="zscore"))
fgts_dygraph(smalldta,title="W dividends",event_ds=fg_tq_divs(c("IBM","QQQ")))

# Other helpers for use with credit ratings, breakouts, and earnings data are available.

# use with forecasts
require(forecast)
require(timetk)
require(sweep)
smalldta <- narrowbydtstr(eqtypx[,.(date,IBM,QQQ)],"-2y::")
fcst_one <- function(ticker) {
  fcst <-tk_ts(smalldta[,.SD,.SDcol=c("date",ticker)]) |> ets() |> forecast::forecast(h=30)
   fcst |> sweep::sw_sweep(timetk_idx=TRUE) |> fg_sweep()
  }
fpred <- merge(fcst_one("QQQ"),fcst_one("IBM"),by="date")
fgts_dygraph(smalldta,title="With Forecasts", dtstartfrac=0.7,rebase=",100",forecast_ds=fpred)
```

---

gendtstr *Date Utilities*

Description

COnverts a generic relative string defining one or two endpoints to exact dates or datestrings

Usage

```
gendtstr(x, today = Sys.Date(), rtn = "dtstr")

narrowbydtstr(
  xin,
  dtstr = "",
  includetoday = TRUE,
  windowdays = 0,
  invert = FALSE,
  addindicator = FALSE
)

extenddtstr(
  instr,
  begchg = 0,
  endchg = 0,
  mindt = NULL,
  maxdt = NULL,
  rtn = "",
  rtnstyle = "string"
)
```

Arguments

x	String describing generalized date as of today
today	Default Sys.Date()
rtn	string describing what to do: (list,datelist,fromtoday,totoday)
xin	Input data.frame or data.table with a Date column
dtstr	Generalized Date string of the form <yyyy-mm-dd>::<yyyy-mm-dd> or e.g. -3m::
includetoday	(Default: TRUE) pass either today Sys.Date() or Sys.Date()-1 to gendtstr
windowdays	(Default: 0)Number of additional days to add at beginning of series
invert	(Default: FALSE) Return dates not in dtstr
addindicator	(Default: FALSE) Returns original dataset with logical variable inrange if date is in desired range.
instr	Input generalized date string, data.table or xts dataset
begchg	(Default: 0) Number of calendar days to extend beginning
endchg	(Default: 0) Number of calendar days to extend end
mindt	Minimum date to return
maxdt	Maximum date to return
rtnstyle	REturn datestring or list

Value

an exact start date startdt and an exact end date enddt, in the following forms: If rtn="list" returns c(startdt,enddt), if rtn="first" then startdt, if rtn="days then an integer number of days from startdt to today otherwise (by default) "startdt::enddt"

Same form as xin, i.e. a data.table or data.frame

character string or list with new dates

Examples

```
gendtstr("-3m::")
gendtstr("-2y::-3m",today=as.Date("2025-03-15"))
narrowbydtstr(eqtypx,"-2m::-1m")
extenddtstr("-2m::-1m")
extenddtstr("-2m::-1m",begchg=-10,endchg=5)
```

imfdta *IMF Economic FOrecasts*

Description

IMF World Economic Outlook June 2025

Usage

```
imfdta
```

Format

`imfdta`:
A long format `data.table` with both historical economic data and projections.

CC ISO Country code
SUBJ IMF subject classification
value Value of historical data or forecast
variable Short abbreviation for SUBJ
ctryname Full country name correponding to CC
region Investment Region each country is in
usedccy Dominant currency used in Country
date Date of historical data or of forecast

Source

https://www.imf.org/en/publications/weo

nomfxdta *Nominal FX levels*

Description

Closing Nominal currency levels (local currency per dollar)

Usage

```
nomfxdta
```

Format

`nomfxdta`:
A tibble with 13134 observations and 3 columns

date Date Of Observation
variable CUrrency
value Currency/USD

Source

https://business-science.github.io/tidyquant/

---

ratings_db *Ratings Database*

Description

Ratings changes for a few select sovereigns.

Usage

```
ratings_db
```

Format

ratings_db:
A tibble with 110 observations and 5 columns

CREDIT Short Credit name for a few countries
AGENCY Ratings Agency
RATING Rating code for each country specific to each agency
WATCH Character indicator if ratings is watch positive or negative
DT_ENTRY Date of ratings annoucement

Source

https://ratingshistory.info/

---

recession_indic *Monthly recession indicator, FRED code RECPROUSM156N*

Description

Monthly recession indicator, FRED code RECPROUSM156N

Usage

```
recession_indic
```

Format

recession_indic:
A data frame with 120 rows and 3 columns

symbol FRED identifier
date Date of report
pricce Observation

Source

https://fred.stlouisfed.org/

---

reerdta *Real FX levels*

Description

Real Effective Exchange Rates

Usage

```
reerdta
```

Format

`reerdta`:
A tibble with 1920 observations and 3 columns

date Date Of Observation
variable ISO Country code
REGION investment region to which each country belongs
value Index of Real Effective Exchange Rates

Source

https://imf.org

---

yc_CMSUST *Constant Maturity UST rates*

Description

FRED calculated constant maturity interest rates

Usage

```
yc_CMSUST
```

Format

`yc_CMSUST`:
A long format `data.table` with constant maturity UST with tenors 2 year, 10 year and 30 year

variable Term of UST
date Date of observation
value Annualized Percent interest rae

Source

https://fred.stlouisfed.org/

Index

∗ **datasets**
consumer_sent, 3
earnings_ibm, 3
eqtypx, 4
eqtypx_melt, 4
eqtyrtn, 5
example_fcst_set, 5
imfdta, 35
nomfxdta, 35
ratings_db, 36
recession_indic, 36
reerdta, 37
yc_CMSUST, 37

alphavantagepf::av_get_pf, *14*

consumer_sent, 3
cpm-package, *11*
cpm::processStream(), *11*

data.table::data.table(), *9*
dygraphs::dyCrosshair(), *29*
dygraphs::dyLegend(), *30*
dygraphs::dyOptions(), *30*
dygraphs::dyRangeSelector(), *30*
dygraphs::dyRoller(), *29*

earnings_ibm, 3
ecp::e.divisive(), *6*
eqtypx, 4
eqtypx_melt, 4
eqtyrtn, 5
example_fcst_set, 5
extenddtstr *(*gendtstr*)*, 33

fg_addbreakouts, 6
fg_addbreakouts(), *30, 31*
fg_av_earnings *(*fg_ratingsEvents*)*, 14
fg_current_theme *(*fg_get_aes*)*, 12
fg_cut_to_events, 7
fg_cut_to_events(), *31*
fg_dates_of_interest, 8
fg_display_colors *(*fg_get_aes*)*, 12
fg_eventStudy, 9
fg_findTurningPoints, 11
fg_findTurningPoints(), *30, 31*
fg_get_aes, 12
fg_get_aes(), *28*
fg_get_aesstring *(*fg_get_aes*)*, 12
fg_get_datemap *(*fg_get_aes*)*, 12
fg_get_dates_of_interest *(*fg_dates_of_interest*)*, 8
fg_get_dates_of_interest(), *17*, *25*, *31*
fg_print_aes_list *(*fg_get_aes*)*, 12
fg_print_aes_list(), *19*
fg_prophet, 13
fg_ratingsEvents, 14
fg_ratingsEvents(), *31*
fg_RegimeChange, 15
fg_replace_theme *(*fg_update_aes*)*, 27
fg_replace_theme(), *19*
fg_reset_to_default_state *(*fg_update_aes*)*, 27
fg_scatplot, 16
fg_scatplot(), *28*
fg_seasonalstudy, 20
fg_signal_to_events, 22
fg_sweep, 23
fg_sync_group, 24
fg_sync_group(), *30*
fg_tq_divs *(*fg_ratingsEvents*)*, 14
fg_tsboxplot, 25
fg_update_aes, 27
fg_update_aes(), *8*, *13*, *19*, *27, 28*, *31*
fg_update_dates_of_interest *(*fg_dates_of_interest*)*, 8
fg_update_dates_of_interest(), *8*, *29*, *31*
fg_update_line_colors *(*fg_update_aes*)*, 27
fg_verbose *(*fg_update_aes*)*, 27

fgts_dygraph, 28
fgts_dygraph(), *6, 7*, *9*, *11*, *13–16*, *23, 24*, *28*

gendtstr, 33
geom_hex(), *18*
geom_label(), *18*
geom_text(), *18*
ggiraph::geom_point_interactive(), *19*
ggiraph::girafe(), *19*
ggplot(), *10*, *22*
ggplot2::facet_grid(), *26*
ggplot2::facet_wrap(), *19*
ggplot2::ggplot(), *13*, *19*, *26*
ggplot2::stat_ellipse(), *17*
ggrepel(), *18*

imfdta, 35

narrowbydtstr *(*gendtstr*)*, 33
nomfxdta, 35

prophet::predict.prophet(), *13*
prophet::prophet(), *11*

ratings_db, 36
recession_indic, 36
reerdta, 37
RegimeChange::detect_regimes(), *15, 16*

scale(), *7*
scales::brewer_pal, *28*
sweep::sw_sweep(), *23*

tidyquant::tq_get(), *14, 15*

yc_CMSUST, 37