Package ‘stocks’
August 31, 2018

Type Package
Title Stock Market Analysis
Version 1.1.4
License GPL-3
Date 2018-08-30
Author Dane R. Van Domelen
Maintainer Dane R. Van Domelen <vandomed@gmail.com>
Description Functions for analyzing stocks or other investments. Main features are loading and aligning historical data for ticker symbols, calculating performance metrics for individual funds or portfolios (e.g. annualized growth, maximum drawdown, Sharpe/Sortino ratio), and creating graphs. C++ code is used to improve processing speed where possible.
Depends rbenchmark, quantmod
Imports dvmisc, graphics, grDevices, Hmisc, lubridate, methods, RColorBrewer, Rcpp (>= 0.12.15), stats, TTR, zoo
Suggests knitr, rmarkdown, pander, printr
LinkingTo Rcpp
RoxygenNote 6.0.1
NeedsCompilation yes
Repository CRAN
Date/Publication 2018-08-31 04:30:03 UTC

R topics documented:

beta_trailing50 .......................................................... 3
contango_hedged ..................................................... 4
contango_simple .......................................................... 5
convert_gain ............................................................ 7
daily_yearly .............................................................. 8
diffs ................................................................. 8
gains_graph ............................................................. 9
gains_prices ............................................................ 10
<table>
<thead>
<tr>
<th>Topics Documented</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>gains_rate</td>
<td>11</td>
</tr>
<tr>
<td>growth_graph</td>
<td>12</td>
</tr>
<tr>
<td>highyield_etfs</td>
<td>13</td>
</tr>
<tr>
<td>largest_etfs</td>
<td>14</td>
</tr>
<tr>
<td>load_gains</td>
<td>14</td>
</tr>
<tr>
<td>load_prices</td>
<td>15</td>
</tr>
<tr>
<td>mdd</td>
<td>17</td>
</tr>
<tr>
<td>metrics</td>
<td>18</td>
</tr>
<tr>
<td>onemetric_graph</td>
<td>19</td>
</tr>
<tr>
<td>onemetric_overtime_graph</td>
<td>20</td>
</tr>
<tr>
<td>pchanges</td>
<td>22</td>
</tr>
<tr>
<td>pdiffs</td>
<td>23</td>
</tr>
<tr>
<td>prices_gains</td>
<td>23</td>
</tr>
<tr>
<td>prices_rate</td>
<td>24</td>
</tr>
<tr>
<td>ratios</td>
<td>25</td>
</tr>
<tr>
<td>rrr</td>
<td>26</td>
</tr>
<tr>
<td>sector_spdr_etfs</td>
<td>27</td>
</tr>
<tr>
<td>sharpe</td>
<td>27</td>
</tr>
<tr>
<td>sortino</td>
<td>28</td>
</tr>
<tr>
<td>stocks</td>
<td>28</td>
</tr>
<tr>
<td>targetall</td>
<td>29</td>
</tr>
<tr>
<td>targetbeta_twofunds</td>
<td>30</td>
</tr>
<tr>
<td>threefunds_graph</td>
<td>33</td>
</tr>
<tr>
<td>ticker_dates</td>
<td>35</td>
</tr>
<tr>
<td>twofunds_graph</td>
<td>36</td>
</tr>
<tr>
<td>twometrics_graph</td>
<td>39</td>
</tr>
<tr>
<td>vanguard_balanced_funds</td>
<td>40</td>
</tr>
<tr>
<td>vanguard_bond_etfs</td>
<td>41</td>
</tr>
<tr>
<td>vanguard_bond_funds</td>
<td>41</td>
</tr>
<tr>
<td>vanguard_etfs</td>
<td>41</td>
</tr>
<tr>
<td>vanguard_funds</td>
<td>41</td>
</tr>
<tr>
<td>vanguard_igrade_etfs</td>
<td>42</td>
</tr>
<tr>
<td>vanguard_igrade_funds</td>
<td>42</td>
</tr>
<tr>
<td>vanguard_international_etfs</td>
<td>42</td>
</tr>
<tr>
<td>vanguard_international_funds</td>
<td>43</td>
</tr>
<tr>
<td>vanguard_largecap_etfs</td>
<td>43</td>
</tr>
<tr>
<td>vanguard_largecap_funds</td>
<td>43</td>
</tr>
<tr>
<td>vanguard_midcap_etfs</td>
<td>44</td>
</tr>
<tr>
<td>vanguard_midcap_funds</td>
<td>44</td>
</tr>
<tr>
<td>vanguard_sector_etfs</td>
<td>44</td>
</tr>
<tr>
<td>vanguard_sector_funds</td>
<td>44</td>
</tr>
<tr>
<td>vanguard_smallcap_etfs</td>
<td>45</td>
</tr>
<tr>
<td>vanguard_smallcap_funds</td>
<td>45</td>
</tr>
<tr>
<td>vanguard_stock_etfs</td>
<td>45</td>
</tr>
<tr>
<td>vanguard_stock_funds</td>
<td>46</td>
</tr>
<tr>
<td>vanguard_targetdate_funds</td>
<td>46</td>
</tr>
<tr>
<td>vanguard_targetrisk_funds</td>
<td>46</td>
</tr>
<tr>
<td>vanguard_taxexempt_bond_funds</td>
<td>47</td>
</tr>
</tbody>
</table>
**beta_trailing50**

Beta for Last 50 Daily Gains

**Description**

Calculates beta for a ticker symbol based on the previous 50 daily gains.

**Usage**

```r
beta_trailing50(ticker, bench = "SPY", ...)
```

**Arguments**

- `ticker`: Character string with ticker symbols that Yahoo! Finance recognizes.
- `bench`: Character string with ticker symbol for benchmark.
- `...`: Arguments to pass to `load_gains`.

**Value**

Numeric value.

**References**


**Examples**

```r
# Not run:
## Calculate TLT's beta based on the previous 50 daily gains
beta_trailing50("TLT")
```

```r
## End(Not run)
```
Backtest a Hedged Contango-Based Volatility Trading Strategy

Description

Implements the following strategy: Each day, hold XIV/SPXU (weighted for zero beta) if contango > xiv.spux.cutpoint, hold VXX/UPRO (weighted for zero beta) if contango < vxx.upro.cutpoint, and hold cash otherwise. Perhaps not very useful since XIV closed on Feb. 20, 2018.

Usage

contango_hedged(contango, xiv.spux.gains = NULL, vxx.upro.gains = NULL, xiv.spux.cutpoint = 6.36, vxx.upro.cutpoint = 5.45, xiv.allocation = 0.46, vxx.allocation = 0.46, xiv.beta = NULL, vxx.beta = NULL, initial = 10000)

Arguments

contango Numeric vector of contango values at the end of each trading day.

xiv.spux.gains 2-column numeric matrix with gains for XIV and SPXU. Should have the same number of rows as contango and be date-shifted one value to the right. For example, the first row should have the XIV and SPXU gains for the day AFTER the first contango value.

vxx.upro.gains 2-column numeric matrix with gains for VXX and UPRO. Should have the same number of rows as contango and be date-shifted one value to the right. For example, the first row should have the VXX and UPRO gains for the day AFTER the first contango value.

xiv.spux.cutpoint Numeric value giving the contango cutpoint for XIV/SPXU position. For example, if xiv.spux.cutpoint = 5, XIV/SPXU will be held whenever contango is greater than 5%.

vxx.upro.cutpoint Numeric value giving the contango cutpoint for VXX/UPRO position. For example, if vxx.upro.cutpoint = -5, VXX/UPRO will be held whenever contango is less than -5%.

xiv.allocation Numeric value specifying XIV allocation for XIV/SPXU position. For example, if set to 0.46, 46% is allocated to XIV and 54% to SPXU when contango > xiv.spux.cutpoint.

vxx.allocation Numeric value specifying VXX allocation for VXX/UPRO position. For example, if set to 0.46, 46% is allocated to VXX and 54% to UPRO when contango < vxx.upro.cutpoint.

xiv.beta Numeric value specifying XIV’s beta. If specified, the function figures out what xiv.allocation needs to be for zero-beta XIV/SPXU positions. For example, if set to 3.5, then 46.2% XIV/53.8% SPXU achieves zero beta.
contango_simple

vxx.beta  Numeric value indicating VXX’s beta. If specified, the function figures out what vxx.allocation needs to be for zero-beta VXX/UPRO positions. For example, if set to -3.5, then 46.2% VXX/53.8% UPRO achieves zero beta.

initial  Numeric value giving the initial value of the portfolio.

details  You can find historical contango values from The Intelligent Investor Blog. You can click the first link at http://investing.kuchita.com/2012/06/28/xiv-data-and-pricing-model-since-vix-futures-available to download a zip file containing an Excel spreadsheet. Then, you will need to calculate whatever version of "contango" you prefer. I typically define contango as what percent higher the second-month VIX futures are compared to the first-month futures, i.e. dividing the "2nd mth" column by the "1st mth" column, subtracting 1, and then multiplying by 100.

To load daily gains for XIV, SPXU, VXX, and UPRO, you can use load_gains, which uses the quantmod package to load data from Yahoo! Finance. You will have to specify the from and to inputs to match the date range for your contango values.

value  List containing:

1. Character vector named holdings indicating what fund was held each day (XIV/SPXU, VXX/UPRO, or cash).
2. Numeric vector named port.gains giving the portfolio gain for each day, which will be 0 for days that cash was held and the weighted XIV/SPXU or VXX/UPRO gain for days that one of those positions was held.
3. Numeric vector named port.balances giving the portfolio balance each day.
4. Numeric value named trades giving the total number of trades executed.


contango_simple  Backtest a Simple Contango-Based Volatility Trading Strategy

description  Simple strategy: Each day, hold XIV if contango > xiv.cutpoint, hold VXX if contango < vxx.cutpoint, and hold cash otherwise. Perhaps not very useful since XIV closed on Feb. 20, 2018.

usage  contango_simple(contango, xiv.gains = NULL, vxx.gains = NULL, xiv.cutpoint = 0, vxx.cutpoint = -Inf, initial = 10000)
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contango</td>
<td>Numeric vector of contango values at the end of each trading day.</td>
</tr>
<tr>
<td>xiv.gains</td>
<td>Numeric vector of gains for XIV. Should be same length as contango and date-</td>
</tr>
<tr>
<td></td>
<td>shifted one value to the right. For example, the first value of xiv.gains</td>
</tr>
<tr>
<td></td>
<td>should be the XIV gain for the day AFTER the first contango value.</td>
</tr>
<tr>
<td>vxx.gains</td>
<td>Numeric vector of gains for VXX. Should be same length as contango and date-</td>
</tr>
<tr>
<td></td>
<td>shifted one value to the right. For example, the first value of vxx.gains</td>
</tr>
<tr>
<td></td>
<td>should be the VXX gain for the day AFTER the first contango value.</td>
</tr>
<tr>
<td>xiv.cutpoint</td>
<td>Numeric value giving the contango cutpoint for XIV, in percent.</td>
</tr>
<tr>
<td>vxx.cutpoint</td>
<td>Numeric value giving the contango cutpoint for VXX, in percent.</td>
</tr>
<tr>
<td>initial</td>
<td>Numeric value giving the initial value of the portfolio.</td>
</tr>
</tbody>
</table>

Details

You can find historical contango values from The Intelligent Investor Blog. You can click the first link at [http://investing.kuchita.com/2012/06/28/xiv-data-and-pricing-model-since-vix-futures-available](http://investing.kuchita.com/2012/06/28/xiv-data-and-pricing-model-since-vix-futures-available) to download a zip file containing an Excel spreadsheet. Then, you will need to calculate whatever version of "contango" you prefer. I typically define contango as what percent higher the second-month VIX futures are compared to the first-month futures, i.e. dividing the "2nd mth" column by the "1st mth" column, subtracting 1, and then multiplying by 100.

I think the most common approach for contango-based volatility strategies is holding XIV (inverse volatility) when contango is above some value (e.g. 0%, 5%, or 10%), and holding cash otherwise. You can do that with this function by leaving vxx.cutpoint as -Inf. However, you may also want to hold VXX (volatility) when contango is below some value (e.g. 0%, -5%, -10%), also known as "backwardation". You can implement an XIV-only, VXX-only, or XIV and VXX strategy with this function.

To load daily gains for XIV and/or VXX, you can use load_gains, which uses the quantmod package [1] to load data from Yahoo! Finance. You will have to specify the from and to inputs to match the date range for your contango values.

Value

List containing:

1. Character vector named holdings indicating what fund was held each day (XIV, VXX, or cash).
2. Numeric vector named port.gains giving the portfolio gain for each day, which will be 0 for days that cash was held and the XIV or VXX gain for days that XIV or VXX was held.
3. Numeric vector named port.balances giving the portfolio balance each day.
4. Numeric value named trades giving the total number of trades executed.

References

**convert_gain** | **Convert Gain from One Time Interval to Another**

**Description**

For example, you can use this function to figure out that an 8 trading days is 31.9

**Usage**

`convert_gain(gain, units.in = 1, units.out = 1)`

**Arguments**

- `gain` Numeric value specifying a gain, e.g. 0.005 for 0.5 a vector of gains.
- `units.in` Numeric value giving the time period over which the gain was achieved.
- `units.out` Numeric value giving the time period you want to convert to.

**Value**

Numeric value or vector.

**Examples**

```r
# Calculate annualized gain for an 8% gain over a 70-day period
convert_gain(gain = 0.08, units.in = 70, units.out = 252)

# Calculate the annual growth rate of a fund that gains 0.02% per day
convert_gain(gain = 0.0002, units.in = 1, units.out = 252)

# Calculate the annual growth rate of a fund that gains 1% per week
convert_gain(gain = 0.01, units.in = 1, units.out = 52)

# You invest in AAPL and gain 0.5% in 17 business days. Express as a 5-year # growth rate.
convert_gain(gain = 0.005, units.in = 17, units.out = 252 * 5)

# Your portfolio has tripled in a 13-year period. Calculate your average # annual gain.
convert_gain(gain = 2, units.in = 13, units.out = 1)
```
**daily_yearly**  
*Convert Daily Gain to X-year Gain*

**Description**

For example, you can use this function to calculate that an investment that gains 0.1 days).

**Usage**

daily_yearly(gain, years = 1)

**Arguments**

gain  
Numeric value specifying a gain, e.g. 0.005 for 0.5 a vector of gains.

years  
Numeric value.

**Value**

Numeric value or vector.

**Examples**

# Calculate annual gain for an investment that gains 0.1% per day
daily_yearly(gain = 0.001)

# Calculate 5-year gains corresponding to various daily gains
daily_yearly(gain = seq(0, 0.001, 0.0001), years = 5)

**diffs**  
*Lagged Differences (Alternate Implementation)*

**Description**

Calculates differences between subsequent (or lagged) elements of a vector. Very similar to `diff`, but written in C++.

**Usage**

diffs(x, lag = 1L)

**Arguments**

x  
Numeric vector.

lag  
Numeric value (e.g. 2 for differences between 1st and 3rd element, 2nd and 4th, ...).
**gains_graph**

**Value**

Numeric vector.

**Examples**

```r
# Generate 1 million values from Poisson(3) distribution
x <- rpois(1000000, 3)

# Calculate vector of differences between subsequent values
y <- diffs(x)

# Could get same result from base R function diff
z <- diff(x)
all.equal(y, z)

# But diffs is faster
benchmark(diffs(x), diff(x), replications = 100)
```

---

**gains_graph**  
*Scatterplot of Investment Gains*

**Description**

Useful for visualizing relationship between one (or several) investments and a benchmark. First fund in `tickers`, `gains`, or `prices` is used as the benchmark.

**Usage**

```r
gains_graph(tickers = NULL, ..., gains = NULL, prices = NULL, orders = 1, add.plot = FALSE, include.legend = TRUE, colors = NULL, lty = NULL, plot.list = NULL, points.list = NULL, legend.list = NULL, pdf.list = NULL, bmp.list = NULL, jpeg.list = NULL, png.list = NULL, tiff.list = NULL)
```

**Arguments**

- **tickers** Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.
- **...** Arguments to pass along with `tickers` to `load_gains`.
- **gains** Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).
- **prices** Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).
orders: Numeric vector specifying the orders of linear regression models for each y-axis investment. Set to 1 for simple linear regression, 2 for linear regression with first- and second-order terms, and so on.

add.plot: Logical value for whether to add plot data to current plot frame rather than open a new one.

include.legend: Logical value.

colors: Character vector of colors for each curve.

lty: Numeric vector specifying line types for each curve.

plot.list: List of arguments to pass to `plot`.

points.list: List of arguments to pass to `points`.

legend.list: List of arguments to pass to `legend`.

pdf.list: List of arguments to pass to `pdf`.

bmp.list: List of arguments to pass to `bmp`.

jpeg.list: List of arguments to pass to `jpeg`.

png.list: List of arguments to pass to `png`.

tiff.list: List of arguments to pass to `tiff`.

Value

In addition to the graph, a list containing fitted linear regression models returned by `lm` for each investment vs. the benchmark.

References


Examples

```r
## Not run:
# Plot daily gains for SSO and UPRO vs. VFINX
fig <- gains_graph(c("VFINX", "SSO", "UPRO"))

## End(Not run)
```

---

**gains_prices**  
*Convert Gains to Prices*

**Description**

Calculates prices based on initial balance and vector of gains.
gains_rate

Usage

gains_prices(gains, initial = 10000)

Arguments

gains Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).
initial Numeric value.

Value

Numeric value if gains is a vector, numeric matrix if gains is a matrix.

Examples

# Simulate daily gains over a 5-year period
set.seed(123)
gains <- rnorm(n = 252 * 5, mean = 0.001, sd = 0.02)

# Plot balance over time if initial balance is $10,000
prices <- gains_prices(gains)
plot(prices)

gains_rate

Calculate Growth Rate From a Vector of Gains

Description

The formula is simply: prod(gains + 1) - 1. If units_rate is specified, then it converts to x-unit growth rate.

Usage

gains_rate(gains, units_rate = NULL)

Arguments

gains Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).
units_rate Numeric value specifying the number of units for growth rate calculation, if you want something other than total growth. For annualized growth rate, set to 252 if gains has daily gains, 12 if gains has monthly gains, etc.

Value

Numeric value if gains is a vector, numeric matrix if gains is a matrix.
Examples

# Create vector of daily gains for a hypothetical stock
daily.gains <- c(-0.02, -0.01, 0.01, 0.02, 0.01)

# Overall growth is 0.95%
gains_rate(daily.gains)

# Average daily growth is 0.19%
gains_rate(daily.gains, 1)

# Corresponds to 61.0% annual growth
gains_rate(daily.gains, 252)

growth_graph

Graph Investment Growth

Description

Useful for comparing performance of investments over time.

Usage

growth_graph(tickers = NULL, ..., gains = NULL, prices = NULL,
initial = "10k", add.plot = FALSE, colors = NULL, lty = NULL,
plot.list = NULL, points.list = NULL, grid.list = NULL,
legend.list = NULL, pdf.list = NULL, bmp.list = NULL,
jpeg.list = NULL, png.list = NULL, tiff.list = NULL)

Arguments

tickers Character vector of ticker symbols that Yahoo! Finance recognizes, if you want
          to download data on the fly.
... Arguments to pass along with tickers to load_gains.
gains Numeric matrix with 1 column of gains for each investment (can be a vector if
          there is only one).
prices Numeric matrix with 1 column of prices for each investment (can be a vector if
          there is only one).
initial Numeric value specifying what value to scale initial prices to. Can also be char-
          acter string ending in "k", e.g. "10k" to graph growth of $10k without all the
          0's.
add.plot Logical value for whether to add plot data to current plot frame rather than open
          a new one.
colors Character vector of colors for each curve.
lty Numeric vector specifying line types for each curve.
plot.list     List of arguments to pass to plot.
points.list   List of arguments to pass to points.
grid.list     List of arguments to pass to grid.
legend.list   List of arguments to pass to legend.
pdf.list      List of arguments to pass to pdf.
bmp.list      List of arguments to pass to bmp.
jpeg.list     List of arguments to pass to jpeg.
png.list      List of arguments to pass to png.
tiff.list     List of arguments to pass to tiff.

Value

In addition to the graph, a list containing:

1. Numeric matrix named prices with prices for each investment.
2. Numeric vector named means with mean of gains for each investment.
3. Numeric matrix named corr.matrix with correlation matrix for gains for each investment.

References


Examples

```r
## Not run:
# Plot growth of $10k in VFINX and BRK-B
fig <- growth_graph(c("VFINX", "BRK-B"))
```

```
## End(Not run)
```

---

**highyield_etfs**  

*High-Yield ETFs from ETFdb.com and Inception Dates*

**Description**

High-Yield ETFs from ETFdb.com and Inception Dates

**Source**

http://etfdb.com/etfdb-category/high-yield-bonds/#etfs&sort_name=assets_under_management&sort_order=desc&page=2
**load_gains**

### Description

Download and align gains for a set of tickers from Yahoo! Finance, using the `quantmod` package.

### Usage

```r
load_gains(tickers, intercepts = NULL, slopes = NULL, from = "1950-01-01", to = Sys.Date(), time.scale = "daily", preto.days = NULL, prefrom.days = NULL, earliest = FALSE, latest = FALSE)
```

### Arguments

- **tickers**: Character vector with ticker symbols that Yahoo! Finance recognizes.
- **intercepts**: Numeric vector of values to add to daily gains for each ticker.
- **slopes**: Numeric vector of values to multiply daily gains for each ticker by. Slopes are multiplied prior to adding intercepts.
- **from**: Date or character string (e.g. "2015-01-15").
- **to**: Date or character string (e.g. "2016-01-30").
- **time.scale**: Character string controlling time frame for gains. Choices are "daily", "monthly", and "yearly".
- **preto.days**: Numeric value. If specified, function returns gains for `preto.days` trading days prior to `to`. To illustrate, to load the most recent 50 daily gains, you would leave `to` and `time.scale` as the defaults and set `preto.days = 50`.
- **prefrom.days**: Numeric value. If specified, function returns gains for `prefrom.days` trading days prior to `from`. Useful when you want to test a trading strategy starting on a particular date, but the strategy requires data leading up to that date (e.g. trailing beta).
load_prices

earliest Logical value for whether to retain only the subset of tickers with data going the furthest back. Set to FALSE if you want all tickers retained and gains over their mutual lifetimes.

latest Logical value for whether to retain only the subset of tickers with data going the furthest forward, e.g. dropping funds that were discontinued at some point.

Details

In aligning historical prices, dates on which not all funds have data are simply dropped. Messages are printed indicating which dates are dropped for which tickers.

Value

Numeric matrix.

References


Examples

```r
## Not run:
# Load gains for Netflix and Amazon over their mutual lifetimes
gains <- load_gains(c("NFLX", "AMZN"))

## End(Not run)
```

load_prices

Download and Align Historical Prices for a Set of Tickers

Description

Downloads and aligns historical prices for specified tickers from Yahoo! Finance, using the quantmod package.

Usage

```r
load_prices(tickers, intercepts = NULL, slopes = NULL,
           from = "1950-01-01", to = Sys.Date(), time.scale = "daily",
           preto.days = NULL, prefrom.days = NULL, initial = NULL,
           earliest = FALSE, latest = FALSE)
```
Arguments

- **tickers**: Character vector with ticker symbols that Yahoo! Finance recognizes.
- **intercepts**: Numeric vector of values to add to daily gains for each ticker.
- **slopes**: Numeric vector of values to multiply daily gains for each ticker by. Slopes are multiplied prior to adding intercepts.
- **from**: Date or character string (e.g. "2015-01-15").
- **to**: Date or character string (e.g. "2016-01-30").
- **time.scale**: Character string controlling time frame for gains. Choices are "daily", "monthly", and "yearly".
- **preto.days**: Numeric value. If specified, function returns gains for `preto.days` trading days prior to `to`. To illustrate, to load the most recent 50 daily gains, you would leave `to` and `time.scale` as the defaults and set `preto.days = 50`.
- **prefrom.days**: Numeric value. If specified, function returns gains for `prefrom.days` trading days prior to `from`. Useful when you want to test a trading strategy starting on a particular date, but the strategy requires data leading up to that date (e.g. trailing beta).
- **initial**: Numeric value specifying what value to scale initial prices to.
- **earliest**: Logical value for whether to retain only the subset of tickers with data going the furthest back. Set to `FALSE` if you want all tickers retained and gains over their mutual lifetimes.
- **latest**: Logical value for whether to retain only the subset of tickers with data going the furthest forward, e.g. dropping funds that were discontinued at some point.

Details

In aligning historical prices, dates on which not all funds have data are simply dropped. Messages are printed indicating which dates are dropped for which tickers.

Value

Numeric matrix.

References


Examples

```r
## Not run:
# Load prices for Netflix and Amazon over their mutual lifetimes
prices <- load_prices(c("NFLX", "AMZN"))

## End(Not run)
```
Maximum Drawdown

Description
Calculates maximum drawdown from vector of closing prices, highs and lows, or gains.

Usage
```r
mdd(prices = NULL, highs = NULL, lows = NULL, gains = NULL,
    indices = FALSE)
```

Arguments
- **prices**: Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).
- **highs**: Numeric vector of daily high prices.
- **lows**: Numeric vector of daily low prices.
- **gains**: Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).
- **indices**: Logical value for whether to include indices for when the maximum drawdown occurred.

Value
Numeric value, vector, or matrix depending on `indices` and whether there is 1 fund or several.

Examples
```r
## Not run:
# Simulate minute-to-minute stock gains over a 2-year period
set.seed(123)
stock.gains <- rnorm(6.5 * 60 * 252 * 2, 0.000005, 0.001)

# Convert to stock prices assuming an initial price of $9.50 per share
stock.prices <- gains_prices(gains = stock.gains, initial = 9.50)

# Plot minute-to-minute stock prices (200k data point, may be slow)
plot(stock.prices)

# Maximum drawdown based on stock prices
mdd(prices = stock.prices)

# Same answer using gains rather than prices
mdd(gains = stock.gains)

## End(Not run)
```
metrics

Calculate Various Performance Metrics

Description

Useful for comparing metrics for several investments. The first investment is used as the benchmark if requested metrics require one.

Usage

metrics(tickers = NULL, ..., gains = NULL, prices = NULL, 
perf.metrics = c("mean", "sd", "growth", "cagr", "mdd", "sharpe", "sortino", 
"alpha", "beta", "r.squared", "pearson", "spearman", "auto.pearson", 
"auto.spearman"))

Arguments

tickers Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.

Arguments to pass along with tickers to load_gains.

gains Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).

prices Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).

perf.metrics Character vector specifying metrics to calculate.

Value

List containing:

1. Numeric matrix named perf.metrics with performance metrics.

2. Numeric matrix named cor.mat with correlation matrix for gains for the various investments.

References


Examples

## Not run:
# Calculate performance metrics for SSO and UPRO, using SPY as benchmark
# for alpha and beta
metrics1 <- metrics(tickers = c("SPY", "SSO", "UPRO"))

## End(Not run)
onemetric_graph

Graph Performance Metric for Various Investments

Description

Useful for visualizing the performance of a group of investments. The first investment is used as the benchmark if the requested metric requires one.

Usage

onemetric_graph(tickers = NULL, ..., gains = NULL, prices = NULL,
   y.metric = "cagr", add.plot = FALSE, sort.tickers = TRUE,
   plot.list = NULL, points.list = NULL, axis.list = NULL,
   pdf.list = NULL, bmp.list = NULL, jpeg.list = NULL, png.list = NULL,
   tiff.list = NULL)

Arguments

tickers Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.

... Arguments to pass along with tickers to load.gains.

gains Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).

prices Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).

y.metric Character string specifying y-axis performance metric. Choices are:
"mean" or "sd" for mean or standard deviation of gains.
"growth" or "cagr" for total or annualized growth.
"mdd" for maximum drawdown.
"sharpe" or "sortino" for Sharpe or Sortino ratio.
"alpha", "beta", or "r.squared" for those metrics from a fitted linear regression on benchmark fund.
"pearson" or "spearman" for Pearson or Spearman correlation with benchmark fund.
"auto.pearson" or "auto.spearman" for Pearson or Spearman autocorrelation, defined as the correlation between subsequent gains.

add.plot Logical value for whether to add plot data to current plot frame rather than open a new one.

sort.tickers Logical value for whether to sort investments in decreasing order of the performance metric.

plot.list List of arguments to pass to plot.

points.list List of arguments to pass to points.

axis.list List of arguments to pass to axis.
onemetric_overtime_graph

Graph Performance Metric Over Time for Various Investments

Description

Useful for visualizing the performance of a group of investments over time. The first investment is used as the benchmark if the requested metric requires one.

Usage

onemetric_overtime_graph(tickers = NULL, ..., gains = NULL, prices = NULL, y.metric = "cagr", window.units = 50, add.plot = FALSE, colors = NULL, lty = NULL, plot.list = NULL, points.list = NULL, legend.list = NULL, pdf.list = NULL, bmp.list = NULL, jpeg.list = NULL, png.list = NULL, tiff.list = NULL)
Arguments

\textbf{tickers} \hspace{1cm} Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.

... \hspace{1cm} Arguments to pass along with \texttt{tickers} to \texttt{load_gains}.

\textbf{gains} \hspace{1cm} Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).

\textbf{prices} \hspace{1cm} Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).

\textbf{y.metric} \hspace{1cm} Character string specifying y-axis performance metric. Choices are:

- "mean" or "sd" for mean or standard deviation of gains.
- "growth" or "cagr" for total or annualized growth.
- "mdd" for maximum drawdown.
- "sharpe" or "sortino" for Sharpe or Sortino ratio.
- "alpha", "beta", or "r.squared" for those metrics from a fitted linear regression on benchmark fund.
- "pearson" or "spearman" for Pearson or Spearman correlation with benchmark fund.
- "auto.pearson" or "auto.spearman" for Pearson or Spearman autocorrelation, defined as the correlation between subsequent gains.

\textbf{window.units} \hspace{1cm} Numeric value specifying the width of the moving window.

\textbf{add.plot} \hspace{1cm} Logical value for whether to add plot data to current plot frame rather than open a new one.

\textbf{colors} \hspace{1cm} Character vector of colors for each curve.

\textbf{lty} \hspace{1cm} Numeric vector specifying line types for each curve.

\textbf{plot.list} \hspace{1cm} List of arguments to pass to \texttt{plot}.

\textbf{points.list} \hspace{1cm} List of arguments to pass to \texttt{points}.

\textbf{legend.list} \hspace{1cm} List of arguments to pass to \texttt{legend}.

\textbf{pdf.list} \hspace{1cm} List of arguments to pass to \texttt{pdf}.

\textbf{bmp.list} \hspace{1cm} List of arguments to pass to \texttt{bmp}.

\textbf{jpeg.list} \hspace{1cm} List of arguments to pass to \texttt{jpeg}.

\textbf{png.list} \hspace{1cm} List of arguments to pass to \texttt{png}.

\textbf{tiff.list} \hspace{1cm} List of arguments to pass to \texttt{tiff}.

Value

In addition to the graph, a numeric matrix containing the performance metric over time for each investment.

References

Examples

## Not run:

```r
# Plot BRK-B's 50-day alpha over time since the start of 2016
fig <- onemetric_overtime_graph(tickers = c("VFINX", "BRK-B"),
                                  y.metric = "alpha",
                                  from = "2016-01-01")
```

## End(Not run)

---

**pchanges**  
*Lagged Proportion Changes*

Description

Calculates proportion changes between subsequent (or lagged) elements of a vector.

Usage

```r
pchanges(x, lag = 1L)
```

Arguments

- `x`  
  Numeric vector.
- `lag`  
  Numeric value (e.g. 2 for differences between 1st and 3rd element, 2nd and 4th, ...).

Value

Numeric vector.

Examples

```r
# Generate 10 values from N(0, 1)
x <- rnorm(10)

# Calculate vector of proportion changes between subsequent values
(y <- pchanges(x))

# Equivalent base R computation
len <- length(x)
p1 <- x[2: len]
p2 <- x[1: (len - 1)]
y2 <- p1 / p2 - 1
all.equal(y, y2)
```
pdiffs

Lagged Proportion Differences

Description

Calculates proportion differences between subsequent (or lagged) elements of a vector.

Usage

pdiffs(x, lag = 1L)

Arguments

- **x**: Numeric vector.
- **lag**: Numeric value (e.g. 2 for differences between 1st and 3rd element, 2nd and 4th, ...).

Value

Numeric vector.

Examples

# Generate 10 values from N(0, 1)
x <- rnorm(10)

# Calculate vector of proportion differences between subsequent values
(y <- pdiffs(x))

# Equivalent base R computation
len <- length(x)
p1 <- x[2: len]
p2 <- x[1: (len - 1)]
y2 <- (p1 - p2) / (0.5 * (p1 + p2))
all.equal(y, y2)

prices_gains

Convert Prices to Gains

Description

Calculates gains based on vector or matrix of prices.
prices_rate

Calculate Growth Rate From a Vector of Prices

Usage

prices_rate(prices, units.rate = NULL)

Arguments

prices       Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).

units.rate   Numeric value specifying the number of units for growth rate calculation, if you want something other than total growth. For annualized growth rate, set to 252 if prices has daily prices, 12 if prices has monthly prices, etc.

Value

Numeric value if prices is a vector, numeric matrix if prices is a matrix.

Description

The formula is simply: \( \frac{\text{prices[length(prices)]}}{\text{prices[1]}} - 1 \). If units.rate is specified, then it converts to x-unit growth rate.

Examples

```r
## Not run:
# Load 2017 prices for Netflix and Amazon, and calculate growth of $10k
prices <- load_prices(c("NFLX", "AMZN"), initial = 1000)

# Calculate gains
gains <- prices_gains(prices)

## End(Not run)
```
Examples

```r
# Create vector of daily closing prices for a hypothetical stock
prices <- c(100.4, 98.7, 101.3, 101.0, 100.9)

# Overall growth is 0.50%
prices_rate(prices)

# Average daily growth is 0.12%
prices_rate(prices, 1)

# Corresponds to 36.7% annualized growth
prices_rate(prices, 252)
```

ratios

**Ratios of Subsequent Elements in a Vector**

**Description**

Calculates vector of ratios of a vector, i.e. ratio of x[2] to x[1], ratio of x[3] to x[2], and so forth.

**Usage**

```r
ratios(x)
```

**Arguments**

- `x` Numeric vector.

**Value**

Numeric vector.

**Examples**

```r
# Generate 10 values from N(0, 1)
x <- rnorm(10)

# Calculate vector of ratios
(y <- ratios(x))

# Slower base R computation
len <- length(x)
y2 <- x[2: len] / x[1: (len - 1)]
all.equal(y, y2)
```
Risk-Return Ratio

Description

Calculates risk-return ratio, defined as growth rate divided by maximum drawdown.

Usage

```r
rrr(prices = NULL, gains = NULL)
```

Arguments

- **prices**: Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).
- **gains**: Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).

Value

Numeric value or vector.

Examples

```r
# Simulate daily gains over a 5-year period
set.seed(123)
stock.gains <- rnorm(252 * 5, 0.0005, 0.01)

# Convert to daily balances assuming an initial balance of $10,000
daily.balances <- gains_prices(stock.gains + 1)

# Total return is about 1.23
daily.balances[length(daily.balances)] / daily.balances[1] - 1

# Maximum drawdown is about 0.19
mdd(prices = daily.balances)

# Ratio of these two is about 6.48

# Easier to calculate using rrr function
rrr(daily.balances)
```
sector_spdr_etfs  Sector SPDR ETFs and Inception Dates

Description

Sector SPDR ETFs and Inception Dates

Source

http://www.sectorspdr.com/sectorspdr/sectors/performance

sharpe  Sharpe Ratio

Description

Calculates Sharpe ratio from vector of gains or prices. The formula is: \( \frac{\text{mean}(\text{gains}) - rf}{\text{sd}(\text{gains})} \), where \( rf \) is some risk-free rate of return.

Usage

sharpe(gains = NULL, prices = NULL, rf = 0)

Arguments

gains  Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).

prices  Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).

rf  Numeric value.

Value

Numeric value.

Examples

# Simulate daily gains over a 5-year period
set.seed(123)
stock.gains <- rnorm(252 * 5, 0.0005, 0.01)

# Calculate Sharpe ratio using risk-free return of 0
sharpe(stock.gains)
sortino | Sortino Ratio
---|---

Description

Calculates Sortino ratio from vector of gains or prices. The formula is: \( \frac{\text{mean}(\text{gains}) - \text{rf}}{\text{sd}(\text{gains}[\text{gains} < 0])} \), where \( \text{rf} \) is some risk-free rate of return.

Usage

\[
\text{sortino}(\text{gains} = \text{NULL}, \text{prices} = \text{NULL}, \text{rf} = 0)
\]

Arguments

- **gains**: Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).
- **prices**: Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).
- **rf**: Numeric value.

Value

Numeric value or vector.

Examples

```r
# Simulate daily gains over a 5-year period
set.seed(123)
stock.gains <- rnorm(252 * 5, 0.0005, 0.01)

# Calculate Sortino ratio using risk-free return of 0
sortino(stock.gains)
```

stocks | Stock Market Analysis
---|---

Description

Functions for analyzing stocks or other investments. Main features are loading and aligning historical data for ticker symbols, calculating performance metrics for individual funds or portfolios (e.g. annualized growth, maximum drawdown, Sharpe/Sortino ratio), and creating graphs. C++ code is used to improve processing speed where possible.

Details
**Description**

Implements a trading strategy aimed at maintaining a fixed allocation to each of several funds, rebalancing when the effective allocations deviate too far from the targets.

**Usage**

```
targetall(tickers = NULL, intercepts = NULL, slopes = NULL, ...,
        tickers.gains = NULL, target.alls = NULL, tol = 0.05,
        rebalance.cost = 0, initial = 10000)
```
Arguments

tickers     Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.
intercepts  Numeric vector of values to add to daily gains for each ticker.
slopes      Numeric vector of values to multiply daily gains for each ticker by. Slopes are multiplied prior to adding intercepts.
...         Arguments to pass along with tickers to load_gains.
tickers.gains Numeric matrix of gains, where each column has gains for a particular fund.
target.alls Numeric vector specifying target allocations to each fund. If unspecified, equal allocations are used (e.g. 1/3, 1/3, 1/3 if there are 3 funds).
tol         Numeric value indicating how far the effective allocations can drift away from the targets before rebalancing.
rebalance.cost Numeric value specifying total cost of each rebalancing trade.
initial     Numeric value specifying what value to scale initial prices to.

Value

List containing:

1. Numeric matrix named fund.balances giving fund balances over time.
2. Numeric value named rebalance.count giving the number of rebalancing trades executed.

References


Examples

```r
## Not run:
# Backtest equal-allocation UPRO/VBLTX/VWEHX strategy
port <- targetall(tickers = c("UPRO", "VBLTX", "VWEHX"))
plot(port$fund.balances[, "Portfolio"])
```

---

targetbeta_twofunds   Backtest a Two-Fund Strategy that Targets a Certain Beta

Description

Implements a two-fund strategy where allocations to each fund are adjusted to maintain some user-specified portfolio beta. For example, you could back-test a zero-beta (i.e. market neutral) UPRO/VBLTX strategy using this function.
targetbeta_twofunds

Usage

targetbeta_twofunds(tickers = NULL, intercepts = NULL, slopes = NULL, ..., benchmark.ticker = NULL, reference.tickers = NULL, tickers.gains = NULL, benchmark.gains = NULL, reference.gains = NULL, target.beta = 0, tol = 0.15, window.units = 50, failure.method = "closer", maxall.tol = tol - 0.05, initial = 10000)

Arguments

tickers Character vector specifying 2 ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.
intercepts Numeric vector of values to add to daily gains for each ticker.
slopes Numeric vector of values to multiply daily gains for each ticker by. Slopes are multiplied prior to adding intercepts.
benchmark.ticker Character string specifying ticker symbol for benchmark index for calculating beta. If unspecified, the first fund in tickers is used as the benchmark.
reference.tickers Character vector of ticker symbols to include on graph as data points for comparative purposes.
tickers.gains Numeric matrix of gains, where each column has gains for a particular fund.
benchmark.gains Numeric vector of gains for the benchmark index for calculating beta. If unspecified, the first fund in tickers.gains is used as the benchmark.
reference.gains Numeric vector or matrix of gains for funds to include on graph as data points for comparative purposes.
target.beta Numeric value.
tol Numeric value specifying how far the effective portfolio beta has to deviate from target.beta to trigger a rebalancing trade.
window.units Numeric value specifying the width of the trailing moving window used to estimate each fund’s beta.
failure.method Character string or vector specifying method(s) to use when fund betas are such that the target portfolio beta cannot be achieved. Choices are "cash", "fund1", "fund2", "fund1.maxall", "fund2.maxall", "inverse1", "inverse2", and "closer". See Details.
maxall.tol Numeric value specifying tolerance to use when implementing the "fund1.maxall" or "fund2.maxall" failure method. To illustrate, if target.beta = 0, fund 1 has a current beta of 1, fund 2 has a current beta of 0.25, failure.method = "fund2.maxall", and maxall.tol = 0.1, a trade will be triggered that results in 40% fund 2 and 60% cash. The portfolio beta is 0.4 * 0.25 + 0.6 = 0.1. The reason you might want maxall.tol to be less than tol is to avoid frequently triggering another trade on the very next day, as fund 2’s beta changes a little and moves the portfolio beta outside of [target.beta - tol, target.beta + tol].
initial Numeric value specifying what value to scale initial prices to.
Details

The general implementation is as follows. Beta for each of the two funds is estimated based on the first \texttt{window.units} gains. Initial allocations are selected to achieve portfolio beta of \texttt{target.beta}. If that is not possible - for example, if \texttt{target.beta} = 0 and both funds have positive beta - then the action taken depends on what method is selected through the \texttt{failure.method} input (details below).

Assuming the target beta is attainable, the function moves over 1 day, and applies each fund’s gains for that day. It then re-calculates each fund’s beta based on the \texttt{window.units}-width interval, and determines the effective portfolio beta based on fund allocations and betas. If the effective beta is outside of \([\texttt{target.beta} - \texttt{tol}, \texttt{target.beta} + \texttt{tol}]\), a rebalancing trade is triggered. As before, if the target beta cannot be achieved, certain actions are taken depending on the selected method.

When outside of a trade because the target beta could not be achieved, the function attempts to rebalance each time it shifts over to a new day, regardless of the effective portfolio beta.

When \texttt{failure.method} = "cash", the entire portfolio balance is allocated to cash when the target beta cannot be achieved.

When \texttt{failure.method} = "fund1" (or "fund2"), the entire portfolio balance is allocated to the first (or second) fund when the target beta cannot be achieved.

When \texttt{failure.method} = "fund1.maxall" (or "fund2.maxall"), when the target beta cannot be achieved, fund 1 (or fund 2) is combined with cash, with the fund 1 (fund 2) allocation as high as possible while staying within \texttt{maxall.tol} of \texttt{target.beta}.

When \texttt{failure.method} = "inverse1" (or "inverse2"), an inverse version of the first (or second) fund is used when the target beta cannot be achieved. In many cases where the target beta cannot be achieved with the two funds, it can be achieved with an inverse version of one and the other. If the target beta still cannot be achieved, the entire portfolio balance is allocated to cash.

When \texttt{failure.method} = "closer", the entire portfolio balance is allocated to whichever fund has a beta closer to \texttt{target.beta}.

Value

For each method, a 4-element list containing:

1. Numeric matrix named \texttt{fund.balances} giving fund balances over time.
2. Numeric matrix named \texttt{fund.betas} giving fund betas over time.
3. Numeric vector named \texttt{effective.betas} giving effective portfolio beta over time.
4. Numeric value named \texttt{trades} giving the total number of trades executed.

References

Examples

```r
## Not run:
# Backtest zero-beta UPRO/VBLTX strategy
beta0 <- targetbeta_twofunds(tickers = c("UPRO", "VBLTX"), target.beta = 0)
plot(beta0$fund.balances[, "Portfolio"])

## End(Not run)
```

### threefunds_graph

**Graph One Performance Metric vs. Another for Three-Fund Portfolio as Allocation Varies**

**Description**

Useful for visualizing performance of three-fund portfolios, typically by plotting a measure of growth vs. a measure of volatility. Only works for one three-fund set at a time.

**Usage**

```r
threefunds_graph(tickers = NULL, intercepts = NULL, slopes = NULL, ..., 
benchmark.tickers = NULL, reference.tickers = NULL, 
tickers.gains = NULL, benchmark.gains = NULL, reference.gains = NULL, 
step.data = 0.0025, step.points = 0.1, step.curves = 0.2, 
x.metric = "sd", y.metric = "mean", tickerlabel.offsets = NULL, 
relabel.offsets = NULL, add.plot = FALSE, colors = NULL, lty = NULL, 
plot.list = NULL, points.list = NULL, text.list = NULL, 
pdf.list = NULL, bmp.list = NULL, jpeg.list = NULL, png.list = NULL, 
tiff.list = NULL)
```

**Arguments**

- `tickers` Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.
- `intercepts` Numeric vector of values to add to daily gains for each ticker.
- `slopes` Numeric vector of values to multiply daily gains for each ticker by. Slopes are multiplied prior to adding intercepts.
- `...` Arguments to pass along with tickers to `load_gains`.
- `benchmark.tickers` Character vector of length 1 or 2 indicating ticker symbols for benchmark indexes. Only used if `x.metric` and/or `y.metric` require benchmark indexes to calculate. For example, to plot correlation with SPY on the x-axis and correlation with TLT on the y-axis, set `x.metric = "pearson", y.metric = "pearson2"` (i.e. Pearson correlation with 2nd benchmark), and `benchmark.tickers = c("SPY", "TLT")`.
- `reference.tickers` Character vector of ticker symbols to include on graph as data points for comparative purposes.
**tickers.gains**  Numeric matrix of gains, where each column has gains for a particular fund.

**benchmark.gains**  Numeric vector or matrix of gains for 1 or 2 benchmark indexes. Only used if `x.metric` and/or `y.metric` require benchmark indexes to calculate. For example, to plot correlation with SPY on the x-axis and correlation with TLT on the y-axis, set `x.metric = "pearson"` and `y.metric = "pearson2"`, and input `benchmark.gains` as a 2-column matrix of gains for SPY and TLT.

**reference.gains**  Numeric vector or matrix of gains for funds to include on graph as data points for comparative purposes.

**step.data**  Numeric value specifying allocation increments for plotting curves.

**step.points**  Numeric value specifying allocation increments for adding data points on top of curves. Set to NULL to suppress data points.

**step.curves**  Numeric value specifying allocation increments for first fund in each set.

**x.metric**  Character string specifying x-axis performance metric. Choices are:
- "mean" or "sd" for mean or standard deviation of gains
- "growth" or "cagr" for total or annualized growth
- "mdd" for maximum drawdown
- "sharpe" or "sortino" for Sharpe or Sortino ratio
- "alpha", "beta", or "r.squared" for those metrics from a fitted linear regression on benchmark fund
- "pearson" or "spearman" for Pearson or Spearman correlation with benchmark fund
- "alpha2", "beta2", "r.squared2", "pearson2", or "spearman2" for same as previously described, but using the second benchmark index
- "auto.pearson" or "auto.spearman" for Pearson or Spearman autocorrelation, defined as the correlation between subsequent gains
- "allocation" for allocation to first fund in each pair.

**y.metric**  Same as `x.metric`, but for the y-axis

**tickerlabel.offsets**  Either a numeric vector of length 2 giving the x- and y-axis offsets for all ticker labels, or a 2-column matrix where each row gives the x- and y-axis offsets for a ticker.

**relabel.offsets**  Either a numeric vector of length 2 giving the x- and y-axis offsets for all reference ticker labels, or a 2-column matrix where each row gives the x- and y-axis offsets for a reference ticker.

**add.plot**  Logical value for whether to add plot data to current plot frame rather than open a new one.

**colors**  Character vector of colors for each curve.

**lty**  Numeric vector specifying line types for each curve.

**plot.list**  List of arguments to pass to `plot`.

**points.list**  List of arguments to pass to `points`. 
ticker_dates

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text.list</td>
<td>List of arguments to pass to text.</td>
</tr>
<tr>
<td>pdf.list</td>
<td>List of arguments to pass to pdf.</td>
</tr>
<tr>
<td>bmp.list</td>
<td>List of arguments to pass to bmp.</td>
</tr>
<tr>
<td>jpeg.list</td>
<td>List of arguments to pass to jpeg.</td>
</tr>
<tr>
<td>png.list</td>
<td>List of arguments to pass to png.</td>
</tr>
<tr>
<td>tiff.list</td>
<td>List of arguments to pass to tiff.</td>
</tr>
</tbody>
</table>

Value

In addition to the graph, a list containing:

1. List named portfolio.xy where each element is a two-column matrix of x- and y-axis values for a curve.
2. Numeric vector named means with mean gains for each fund.
3. Numeric matrix named corr.matrix with a correlation matrix for gains for each fund.

References


Examples

```r
## Not run:
# Plot mean vs. SD for UPRO/VBLTX/VWEHX portfolio, and compare to VFINX and BRK-B
fig <- threefunds_graph(tickers = c("VWEHX", "VBLTX", "UPRO"),
                         reference.tickers = c("VFINX", "BRK-B"))

## End(Not run)
```

---

**ticker_dates**  
*Get Yahoo! Finance Start/End Dates for Tickers*

**Description**

Typically useful for determining a time period over which to compare several funds.

**Usage**

ticker_dates(tickers, from = "1950-01-01", to = Sys.Date())
Arguments

tickers Character vector with ticker symbols that Yahoo! Finance recognizes.
from Date or character string (e.g. "2015-01-15").
to Date or character string (e.g. "2016-01-30").

Value

Data frame with ticker symbol, start date, end date, and number of trading days for each ticker.

References


Examples

```r
## Not run:
# See what dates are available for Apple and Amazon
ticker_dates(c("AAPL", "AMZN"))

## End(Not run)
```

twofunds_graph

Graph One Performance Metric vs. Another for Two-Fund Portfolios as Allocation Varies

Description

Useful for visualizing performance of two-fund portfolios, typically by plotting a measure of growth vs. a measure of volatility. First two investments are used as the first two-fund pair, next two as the second two-fund pair, and so on.

Usage

twofunds_graph(tickers = NULL, intercepts = NULL, slopes = NULL, ...
benchmark.tickers = NULL, reference.tickers = NULL,
tickers.gains = NULL, benchmark.gains = NULL, reference.gains = NULL,
step.data = 0.0025, step.points = 0.1, x.metric = "sd",
y.metric = "mean", tickerlabel.offsets = NULL, relabel.offsets = NULL,
add.plot = FALSE, colors = NULL, lty = NULL, plot.list = NULL,
points.list = NULL, text.list = NULL, pdf.list = NULL,
bmp.list = NULL, jpeg.list = NULL, png.list = NULL, tiff.list = NULL)
Arguments

tickers  Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.

intercepts  Numeric vector of values to add to daily gains for each ticker.

slopes  Numeric vector of values to multiply daily gains for each ticker by. Slopes are multiplied prior to adding intercepts.

...  Arguments to pass along with tickers to load_gains.

benchmark.tickers  Character vector of length 1 or 2 indicating ticker symbols for benchmark indexes. Only used if x.metric and/or y.metric require benchmark indexes to calculate. For example, to plot correlation with SPY on the x-axis and correlation with TLT on the y-axis, set x.metric = "pearson", y.metric = "pearson2" (i.e. Pearson correlation with 2nd benchmark), and benchmark.tickers = c("SPY", "TLT").

reference.tickers  Character vector of ticker symbols to include on graph as data points for comparative purposes.

tickers.gains  Numeric matrix of gains, where each column has gains for a particular fund.

benchmark.gains  Numeric vector or matrix of gains for 1 or 2 benchmark indexes. Only used if x.metric and/or y.metric require benchmark indexes to calculate. For example, to plot correlation with SPY on the x-axis and correlation with TLT on the y-axis, set x.metric = "pearson" and y.metric = "pearson2", and input benchmark.gains as a 2-column matrix of gains for SPY and TLT.

reference.gains  Numeric vector or matrix of gains for funds to include on graph as data points for comparative purposes.

step.data  Numeric value specifying allocation increments for plotting curves.

step.points  Numeric value specifying allocation increments for adding data points on top of curves. Set to NULL to suppress data points.

x.metric  Character string specifying x-axis performance metric. Choices are: "mean" or "sd" for mean or standard deviation of gains "growth" or "cagr" for total or annualized growth "mdd" for maximum drawdown "sharpe" or "sortino" for Sharpe or Sortino ratio "alpha", "beta", or "r.squared" for those metrics from a fitted linear regression on benchmark fund "pearson" or "spearman" for Pearson or Spearman correlation with benchmark fund "alpha2", "beta2", "r.squared2", "pearson2", or "spearman2" for same as previously described, but using the second benchmark index "auto.pearson" or "auto.spearman" for Pearson or Spearman autocorrelation, defined as the correlation between subsequent gains "allocation" for allocation to first fund in each pair.
y.metric Same as x.metric, but for the y-axis
tickerlabel.offsets Either a numeric vector of length 2 giving the x- and y-axis offsets for all ticker labels, or a 2-column matrix where each row gives the x- and y-axis offsets for a ticker.
relabel.offsets Either a numeric vector of length 2 giving the x- and y-axis offsets for all reference ticker labels, or a 2-column matrix where each row gives the x- and y-axis offsets for a reference ticker.
add.plot Logical value for whether to add plot data to current plot frame rather than open a new one.
colors Character vector of colors for each curve.
lty Numeric vector specifying line types for each curve.
plot.list List of arguments to pass to plot.
points.list List of arguments to pass to points.
text.list List of arguments to pass to text.
pdf.list List of arguments to pass to pdf.
bmp.list List of arguments to pass to bmp.
jpeg.list List of arguments to pass to jpeg.
png.list List of arguments to pass to png.
tiff.list List of arguments to pass to tiff.

Value

In addition to the graph, a list containing:

1. List named portfolio.xy where each element is a two-column matrix of x- and y-axis values for a fund pair.
2. Numeric vector named means with mean gains for each fund.
3. Numeric matrix named corr.matrix with a correlation matrix for gains for each fund.

References


Examples

```r
## Not run:
# Plot mean vs. SD for UPRO/VBLTX portfolio, and compare to VFINX and BRK-B
tfig1 <- twofunds_graph(tickers = c("UPRO", "VBLTX"),
                        reference.tickers = c("VFINX", "BRK-B"))

# Same funds, but annualized growth vs. maximum drawdown
tfig2 <- twofunds_graph(tickers = c("UPRO", "VBLTX"),
                        reference.tickers = c("VFINX", "BRK-B"),
```
Description

Useful for visualizing the performance of a group of investments. The first investment is used as the benchmark if `x.metric` or `y.metric` require one benchmark, and the first two investments are used as benchmarks if `x.metric` and `y.metric` require different benchmarks.

Usage

twometrics_graph(tickers = NULL, ..., gains = NULL, prices = NULL, 
x.metric = "mdd", y.metric = "cagr", tickerlabel.offsets = NULL, 
add.plot = FALSE, colors = NULL, plot.list = NULL, points.list = NULL, 
text.list = NULL, pdf.list = NULL, bmp.list = NULL, jpeg.list = NULL, 
png.list = NULL, tiff.list = NULL)

Arguments

tickers
Character vector of ticker symbols that Yahoo! Finance recognizes, if you want to download data on the fly.

... Arguments to pass along with tickers to `load_gains`.
gains Numeric matrix with 1 column of gains for each investment (can be a vector if there is only one).
prices Numeric matrix with 1 column of prices for each investment (can be a vector if there is only one).
x.metric Character string specifying x-axis performance metric. Choices are:
"mean" or "sd" for mean or standard deviation of gains.
"growth" or "cagr" for total or annualized growth.
"mdd" for maximum drawdown.
"sharpe" or "sortino" for Sharpe or Sortino ratio.
"alpha", "beta", or "r.squared" for those metrics from a fitted linear regression on benchmark fund.
"pearson" or "spearman" for Pearson or Spearman correlation with benchmark fund.
"alpha2", "beta2", "r.squared2", "pearson2", or "spearman2" for same as previously described, but using the second benchmark index.
"auto.pearson" or "auto.spearman" for Pearson or Spearman autocorrelation, defined as the correlation between subsequent gains.
y.metric Same as `x.metric`, but for the y-axis.
tickerlabel.offsets

Either a numeric vector of length 2 giving the x- and y-axis offsets for all ticker labels, or a 2-column matrix where each row gives the x- and y-axis offsets for a ticker.

add.plot

Logical value for whether to add plot data to current plot frame rather than open a new one.

colors

Character vector of colors for each curve.

plot.list

List of arguments to pass to plot.

points.list

List of arguments to pass to points.

text.list

List of arguments to pass to text.

pdf.list

List of arguments to pass to pdf.

bmp.list

List of arguments to pass to bmp.

jpeg.list

List of arguments to pass to jpeg.

png.list

List of arguments to pass to png.

tiff.list

List of arguments to pass to tiff.

Value

In addition to the graph, a data frame containing the performance metrics for each investment.

References


Examples

```r
## Not run:
# Plot annualized growth vs. maximum drawdown for VFINX, SSO, and UPRO
fig <- twometrics_graph(tickers = c("VFINX", "SSO", "UPRO"))

## End(Not run)
```

---

vanguard_balanced_funds

*Vanguard Balanced Mutual Funds and Inception Dates*

Description

Vanguard Balanced Mutual Funds and Inception Dates

Source

https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns
<table>
<thead>
<tr>
<th>vanguard_bond_etfs</th>
<th>Vanguard Bond ETFs and Inception Dates</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Vanguard Bond ETFs and Inception Dates</td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td><a href="https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns">https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>vanguard_bond_funds</th>
<th>Vanguard Bond Mutual Funds</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Vanguard Bond Mutual Funds</td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td><a href="https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns">https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>vanguard_etfs</th>
<th>Vanguard ETFs and Inception Dates</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Vanguard ETFs and Inception Dates</td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td><a href="https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns">https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>vanguard_funds</th>
<th>Vanguard Mutual Funds and Inception Dates</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Vanguard Mutual Funds and Inception Dates</td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td><a href="https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns">https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns</a></td>
</tr>
</tbody>
</table>
vanguard_igrade_etfs  Vanguard Investment-grade Bond ETFs and Inception Dates

Description
Vanguard Investment-grade Bond ETFs and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

vanguard_igrade_funds  Vanguard Investment-grade Bond Mutual Funds and Inception Dates

Description
Vanguard Investment-grade Bond Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns

vanguard_international_etfs  Vanguard International ETFs and Inception Dates

Description
Vanguard International ETFs and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns
Vanguard International Mutual Funds and Inception Dates

Description
Vanguard International Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns

Vanguard Large-cap Stock ETFs and Inception Dates

Description
Vanguard Large-cap Stock ETFs and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

Vanguard Large-cap Stock Mutual Funds and Inception Dates

Description
Vanguard Large-cap Stock Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns
vanguard_midcap_etfs  Vanguard Mid-cap Stock ETFs and Inception Dates

Description
Vanguard Mid-cap Stock ETFs and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

vanguard_midcap_funds  Vanguard Mid-cap Stock Mutual Funds and Inception Dates

Description
Vanguard Mid-cap Stock Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns

vanguard_sector_etfs  Vanguard Sector & Specialty ETFs and Inception Dates

Description
Vanguard Sector & Specialty ETFs and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

vanguard_sector_funds  Vanguard Sector Mutual Funds and Inception Dates

Description
Vanguard Sector Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns
**Vanguard Small-cap Stock ETFs and Inception Dates**

**Description**
Vanguard Small-cap Stock ETFs and Inception Dates

**Source**
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

---

**Vanguard Small-cap Stock Mutual Funds and Inception Dates**

**Description**
Vanguard Small-cap Stock Mutual Funds and Inception Dates

**Source**
https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns

---

**Vanguard Stock ETFs and Inception Dates**

**Description**
Vanguard Stock ETFs and Inception Dates

**Source**
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns
vanguard_stock_funds  Vanguard Stock Mutual Funds and Inception Dates

Description

Vanguard Stock Mutual Funds and Inception Dates

Source

https://investor.vanguard.com/mutual-funds/list?assetclass=bond#/mutual-funds/asset-class/month-end-returns

vanguard_targetdate_funds  Vanguard Target Date Mutual Funds

Description

Vanguard Target Date Mutual Funds

Source

https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

vanguard_targetrisk_funds  Vanguard Target Risk Mutual Funds and Inception Dates

Description

Vanguard Target Risk Mutual Funds and Inception Dates

Source

https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns
vanguard_taxexempt_bond_funds

Vanguard Tax-exempt Bond Mutual Funds and Inception Dates

Description
Vanguard Tax-exempt Bond Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

vanguard_traditional_funds

Vanguard Traditional Mutual Funds and Inception Dates

Description
Vanguard Traditional Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns

vanguard_treasury_etfs

Vanguard Treasury/Agency Bond ETFs and Inception Dates

Description
Vanguard Treasury/Agency Bond ETFs and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns
Vanguard Treasury/Agency Bond Mutual Funds and Inception Dates

Description
Vanguard Treasury/Agency Bond Mutual Funds and Inception Dates

Source
https://investor.vanguard.com/etf/list?assetclass=bond#/etf/asset-class/month-end-returns
Index

axis, 19
beta_trailing50, 3
bmp, 10, 13, 20, 21, 35, 38, 40
contango_hedged, 4
contango_simple, 5
convert_gain, 7
daily_yearly, 8
diff, 8
diffs, 8
gains_graph, 9
gains_prices, 10
gains_rate, 11
grid, 13
growth_graph, 12
highyield_etfs, 13
jpeg, 10, 13, 20, 21, 35, 38, 40
largest_etfs, 14
legend, 10, 13, 21
lm, 10
load_gains, 3, 5, 6, 9, 12, 14, 18, 19, 21, 30, 31, 33, 37, 39
load_prices, 15
mdd, 17
metrics, 18
onemetric_graph, 19
onemetric_overtime_graph, 20
pchanges, 22
pdf, 10, 13, 20, 21, 35, 38, 40
pdiffs, 23
plot, 10, 13, 19, 21, 34, 38, 40
png, 10, 13, 20, 21, 35, 38, 40
points, 10, 13, 19, 21, 34, 38, 40
prices_gains, 23
prices_rate, 24
ratios, 25
rrr, 26
sector_spdr_etfs, 27
sharpe, 27
sortino, 28
stocks, 28
stocks-package (stocks), 28
targetall, 29
targetbeta_twofunds, 30
text, 35, 38, 40
threefunds_graph, 33
ticker_dates, 35
tiff, 10, 13, 20, 21, 35, 38, 40
twofunds_graph, 36
twometrics_graph, 39
vanguard_balanced_funds, 40
vanguard_bond_etfs, 41
vanguard_bond_funds, 41
vanguard_etfs, 41
vanguard_funds, 41
vanguard_igrade_etfs, 42
vanguard_igrade_funds, 42
vanguard_international_etfs, 42
vanguard_international_funds, 43
vanguard_largecap_etfs, 43
vanguard_largecap_funds, 43
vanguard_midcap_etfs, 44
vanguard_midcap_funds, 44
vanguard_sector_etfs, 44
vanguard_sector_funds, 44
vanguard_smallcap_etfs, 45
vanguard_smallcap_funds, 45
vanguard_stock_etfs, 45
vanguard_stock_funds, 46
vanguard_targetdate_funds, 46
vanguard_targetrisk_funds, 46
vanguard_taxexempt_bond_funds, 47
vanguard_traditional_funds, 47
vanguard_treasury_etfs, 47
vanguard_treasury_funds, 48