# Package: tfplot (via r-universe)

August 16, 2024

**Version** 2015.12-1

**Title** Time Frame User Utilities

**Description** Utilities for simple manipulation and quick plotting of time series data. These utilities use the 'tframe' package which provides a programming kernel for time series. Extensions to 'tframe' provided in 'tframePlus' can also be used. See the Guide vignette for examples.

**Depends** R (>= 2.12.0), tframe

**Imports** stats, graphics, grDevices

**Suggests** googleVis

**Enhances** tframePlus

**LazyLoad** yes

**License** GPL-2

**Copyright** 1993-1996,1998-2011 Bank of Canada. 1997,2012-2014 Paul Gilbert

**Author** Paul Gilbert <pgilbert.ttv9z@ncf.ca>

**Maintainer** Paul Gilbert <pgilbert.ttv9z@ncf.ca>

**URL** http://tsanalysis.r-forge.r-project.org/

**Repository** https://r-forge.r-universe.dev

**RemoteUrl** https://github.com/r-forge/tsanalysis

**RemoteRef** HEAD

**RemoteSha** 24081e0c434465ca8d78b5de290f47bf916413e8

## Contents

---

`addDate` *Add Periods to a Date*

---

## Description

Add periods to two element start date of given frequency to give a new date. NULL periods is treated as 0.

## Usage

```
addDate(date, periods, freq)
```

## Arguments

| | |
|---|---|
| `date` | A two element date as used by tsp i.e c(year, period). |
| `periods` | A number of periods. |
| `freq` | The number of periods in a year. |

## Value

A two element date.

## Note

A useful utility not strictly part of tframe.

## See Also

`tfExpand`

## Examples

```
addDate(c(1998,1), 20, 12)
```

diffLog *Calculate the difference of log data*

## Description

Calculate the difference from lag periods prior for log of data.

## Usage

```
    diffLog(obj, lag=1, base = exp(1),
         names=paste("diff of log of ", seriesNames(obj)))
    ## Default S3 method:
diffLog(obj, lag=1, base = exp(1),
         names=paste("diff of log of ", seriesNames(obj)))
```

## Arguments

| | |
|---|---|
| obj | A time series object. |
| lag | The difference is calculated relative to lag periods prior. |
| base | Base to use when calculating logrithms. |
| names | names for the new series (but is details). |

## Details

The result is a time series of the difference relative to lag periods prior for the log of the data. lag data points are lost from the beginning of the series. Negative values will result in NAs.

The names are not applied to the new series if the global option ModSeriesNames is FALSE. This can be set with options(ModSeriesNames=FALSE). This provides a convenient mechanism to prevent changing series labels on plot axis, when the title may indicate that data is in year-to-year percent change so the axis label does not need this.

## Value

A time series vector or matrix.

## Examples

```
z <- matrix(100 + rnorm(200),100,2)
z[z <= 0] <- 1 # not to likely, but it can happen
z <- diffLog(z)
```

percentChange *Various Time Series Calculations*

## Description

Calculate various conversions of time series.

## Usage

```
    percentChange(obj, ...)
    ## Default S3 method:
percentChange(obj, base=NULL, lag=1, cumulate=FALSE, e=FALSE, ...)

    ytoypc(obj, names = paste("y to y %ch", seriesNames(obj)))
    ## Default S3 method:
ytoypc(obj, names = paste("y to y %ch", seriesNames(obj)))

    annualizedGrowth(obj, ...)
    ## Default S3 method:
annualizedGrowth(obj, lag=1, freqLagRatio=frequency(obj)/lag,
        names=paste("Annual Growth of", seriesNames(obj)), ...)
```

## Arguments

| | |
|---|---|
| obj | An object on which the calculation is to be done. The default method works for a time series vector or matrix (with columns corresponding to series, which are treated individually). |
| e | If e is TRUE the exponent of the series is used (after cumulating if cumulate is TRUE). e can be a logical vector with elements corresponding to columns of obj. |
| base | If base is provided it is treated as the first period value (that is, prior to differencing). It is prefixed to the m prior to cumulating. It should be a vector of length dim(m)[2]. (If e is TRUE then base should be log of the original data). |
| lag | integer indicating the number of periods relative to which the change should be calculated. |
| cumulate | logical indicating if the series should be cumulated before the percent change is calculated. |
| freqLagRatio | the ratio of obj's frequency to the number of lags. |
| names | gives new names to be given to the calculated series. |
| ... | arguments passed to other methods. |

### Details

percentChange calculate the percent change relative to the data lag periods prior. If cumulate is TRUE then the data is cumulated first. cumulate can be a logical vector with elements corresponding to columns of obj.

The result is a time series of the year over year percent change. This uses percentChange with lag=frequency(obj).

The names are not applied to the new series if the global option ModSeriesNames is FALSE. This can be set with options(ModSeriesNames=FALSE). This provides a convenient mechanism to prevent changing series labels on plot axis, when the title may indicate that data is in year-to-year percent change so the axis label does not need this.

annualizedGrowth calculates the year to year percentage growth rate using 100*((obj/shift(obj, periods= -lag))^freqLagRatio - 1). The default gives the annualized one period growth. If lag is equal to the frequency of obj then the result is year-over-year growth.

### Value

A time series or time series matrix.

### See Also

diff

### Examples

```
z <- ts(matrix(100 + rnorm(200),100,2), start=c(1990,1), frequency=12)
z[z == 0] <- 1 # not to likely, but it can happen
zyypc <- ytoypc(z)
zpc <- percentChange(z)
zag <- annualizedGrowth(z)
```

---

## tfplot *Plot Tframed Objects*

---

### Description

Plot tframe or tframed objects.

### Usage

```
    tfplot(x, ...)

    ## Default S3 method:
tfplot(x, ..., tf=tfspan(x, ...), start=tfstart(tf), end=tfend(tf),
       series=seq(nseries(x)),
        Title=NULL, title=Title, subtitle=NULL,
        lty = 1:5, lwd = 1, pch = 1, col = 1:6, cex = NULL,
```

```
       xlab=NULL, ylab=seriesNames(x), xlim = NULL, ylim = NULL,
       graphs.per.page=5, par=NULL, reset.screen=TRUE,
       Xaxis="auto", L1=NULL,
YaxisL=TRUE, YaxisR=FALSE, Yaxis.lab.rot = "vertical",
splitPane=NULL,
       lastObs = FALSE, source = NULL,
       footnote = NULL, footnoteLeft = footnote, footnoteRight = NULL,
legend=NULL, legend.loc="topleft")
   tfOnePlot(x, tf=tframe(x), start=tfstart(tf), end=tfend(tf),
       Title=NULL, title=Title,  subtitle=NULL,
lty=1:5, lwd=1, pch=1, col=1:6, cex=NULL,
       xlab=NULL, ylab=NULL, xlim=NULL, ylim=NULL, par=NULL,
Xaxis="auto", L1=NULL,
YaxisL=TRUE, YaxisR=FALSE, Yaxis.lab.rot = "vertical",
splitPane=NULL,
lastObs=FALSE,  source=NULL,
footnote=NULL, footnoteLeft=footnote, footnoteRight=NULL,
legend=NULL, legend.loc="topleft")
```

## Arguments

| | |
|---|---|
| x | a tframe or tframed object to plot. |
| ... | any additional tframed objects for the same plot. |
| start | start of plot. (passed to tfwindow) |
| end | end of plot. (passed to tfwindow) |
| tf | a tframe or tframed object which can be used to specify start and end. |
| series | series to be plotted. (passed to selectSeries) |
| title | string to use for plot title (but see details). |
| Title | synonym for title. |
| subtitle | string to use for plot subtitle (but see details). |
| lty | passed to plot. See also par. |
| lwd | passed to plot. See also par. |
| pch | passed to plot. See also par. |
| col | passed to plot. See also par. |
| cex | passed to plot. See also par. |
| xlab | string to use for x label (passed to plot). |
| ylab | string to use for y label (passed to plot). |
| xlim | passed to plot. See also par. |
| ylim | passed to plot. See also par. |
| Xaxis | If equal 'auto' then an attempt is made at a better format for the x-axis tick marks and their labels. A value of NULL produces the result using plot defaults (as previously). |
| YaxisL | logical indicating if a left Y axis should be on the graph. |

| | |
|---|---|
| YaxisR | logical or numeric indicating if a right Y axis should be on the graph. A numeric value indicates its scale relative to the left axis. |
| Yaxis.lab.rot | 'vertical' or 'horizontal' indicating the orientation of labels on the Y axis. |
| L1 | A character vector used for the minor tick marks. The default is in english (e.g. the first letter of each month). It should be the same length as the frequency of x. |
| lastObs | Logical indicating if the date of the last observation should be printed below the graph, flushed right. |
| splitPane | An integer indicating the number of last observations that should be put in a second right panel (to show more detail at the end). NULL indicates no second panel. |
| source | String printed below the graph, flushed left. |
| footnote | Synonym for footnoteLeft. |
| footnoteLeft | String printed below lastObs and source, flushed left. |
| footnoteRight | String printed below lastObs and source, flushed right. |
| legend | NULL (indicating no legend) or a vector strings to be used for a legend (see legend) |
| legend.loc | indication of placement of the legend (see legend) |
| graphs.per.page | integer indicating number of graphs to place on a page. |
| par | a list of arguments passed to par() before plotting.) |
| reset.screen | logical indicating if the plot window should be cleared before starting. If this is not TRUE then par values will have no effect. |

## Details

In many cases these are the same as plot methods, however, tfplot puts different series in the object x in different plot panels, whereas plot usually puts them in the same panel. For this reason, tfplot tends to work better when the scale of the different series are very different. If additional objects are supplied, then they should each have the same number of series as x and all corresponding series will be plotted in the same panel.

tfplot provides an alternate generic mechanism for plotting time series data. New classes of time series may define there own tfplot (and plot) methods.

tfplot does calls to tfOnePlot for each panel. tfOnePlot may give slightly better control, especially in cases where all series are to go on one plot. The functions are intended to provide a convenient way to do some usual things. Ultimately tfOnePlot calls plot, title, and mtext, so even more control of plot details can be achieved by calling those functions directly.

The start and end arguments to tfplot determine the start and end of the plot. The argument tf is an alternate way to specify the start and end. It is ignored if start and end are specified.

If xlim and ylim are not NULL they should be a vector of two elements giving the max and min, which are applied to all graphs, or a list of length equal to the number of series to be plotted with each list element being the two element vector for the corresponding plot limits.

Xaxis provides a mechanism to try and achieve a better default axis. If equal 'auto' then an attempt is made at a format with large tick marks for years and smaller tick marks for periods (months

or quarters). If the number of years is sufficiently small, so there is enough space, then period indications are added. The default, indicated by `L1=NULL`, is the c('Q1', Q2', Q3', 'Q4') will be used for quarterly data and c("J","F","M","A","M","J","J","A","S","O","N","D") for monthly data. Different values can be specified by setting `L1`. It should be the same length as the frequency of `x`. If `Xaxis` is set to `NULL` then the result is to use plot defaults (as prior to the addition of the `Xaxis` argument in version 2013.11-1). Currently `Xaxis='auto'` only affects annual, monthly, and quarterly data, and the affect on annual data is marginal.

`YaxisL` set `TRUE` or `FALSE` controls if left axis tick marks and labels are put on the plot. If `YaxisR` is `FALSE` then right axis tick marks and labels are not put on the plot. If it is `TRUE` then they are put on the plot with the same scale as the left axis (or as it would have if it were plotted). If `YaxisR` is a numeric value then the right axis is put on the plot with the scale of the left axis multiplied by the numeric value. The data is plotted using the left scale, so the user must appropriately adjust any values to be read on the right scale (divide by YaxisR). `YaxisR` can be a vector of length equal to the number of series in `x`, in which case a scale element is applied to the corresponding plot panel. If `YaxisR` is shorter it is recycled, so a scalar value is applied to all panels.

The `title` is not put on the plot if the global option PlotTitles is FALSE. This can be set with `options(PlotTitles=FALSE)`. This provides a convenient mechanism to omit all titles when the title may be added separately (e.g. in Latex).

Similarly, `options(PlotPlotSubtitles=FALSE)`, `options(PlotSources=FALSE)`, and `options(PlotFootnotes=FALSE)` can be used to suppress printing of these.

Footnotes can contain `"\n"` to produce multiline, or multiple footnotes. However, if `source` and `lastObs` are speciied then the overlap can be messy. In this case a better result might be obtained by specifying the source as part of the footnote.

If `subtitle`, `source`, `footnoteLeft`, `footnoteRight` or `legend.loc` have length less than the number of panels then they are replicated, so typically they should have one element that is applied to each panel, or be vectors with one element for each panel. For `tfOnePlot` these should all have length `1`. If `legend` is a matrix then a column will be used for each panel, otherwise the vector will be passed to each panel. (Typically this vector has length equal to the number of series in each panel graph.)

The `par` argument can be used to pass other graphics parameters to `tfplot` and `tfOnePlot` (see `par`). These are set by a call `par(par)` in `tfplot` or `tfOnePlot`. `tfplot` makes this call and does not pass `par` to `tfOnePlot`, so the result may sometimes be different from making a direct call to `tfOnePlot` and providing the `par` argument. Some of the margin (`mar`) setting are overridden by split plots, so the results may not be predictable for this case.

## Value

None.

## Side Effects

An object is plotted.

## See Also

`tfprint`, `tframe`, `tframed`, `print`, `plot`, `legend`, `par`

## Examples

```
tfplot(ts(rnorm(100), start=c(1982,1), frequency=12))
tfplot(ts(rnorm(100), start=c(1982,1), frequency=12), start=c(1985,6))
```

---

`tfVisPlot` *Plot Tframed Objects using googleVis*

---

## Description

Plot tframe or tframed objects using googleVis, which allows pointing to lines on the plot in a browser to display extra information.

## Usage

```
tfVisPlot(x, tf = tframe(x), start = tfstart(tf), end = tfend(tf),
              options=list(title=NULL), ...)
```

## Arguments

| | |
|---|---|
| `x` | a tframe or tframed object to plot. |
| `...` | any additional tframed objects for the same plot. |
| `start` | start of plot. (passed to tfwindow) |
| `end` | end of plot. (passed to tfwindow) |
| `tf` | a tframe or tframed object which can be used to specify start and end. |
| `options` | passed to googleVis, including title. |

## Details

This function produces a line plot of time series `x` in a web browser using `gvisLineChart` from package **googleVis**. The advantage of this relative to `tfplot` and `tfOnePlot` is that additional information about the series or points are displayed when the mouse pointer is close to a point. This can be useful, for example, to distinguish a particular vintage among several vintages in a graph. See package **googleVis** for more details.

## Value

None.

## Side Effects

An object is plotted in a browser.

## See Also

`tfplot`, `tfOnePlot`, `gvisLineChart`

## Examples

```
## Not run:
 z <- ts(matrix(rnorm(1000),100,10), start=c(1982,1), frequency=12)
 seriesNames(z) <- paste("Series", 1:10)
 if (requireNamespace("googleVis"))
    tfVisPlot(z, options=list(title="Random Number Series"))

## End(Not run)
```

---

`tsScan` *Read and Write Time Series to Files*

---

## Description

Read and write time series to files.

## Usage

```
tsScan(file="", skip=1, nseries=1, sep=",",
    na.strings=c("NA", "NC", "ND"), ...)

tsWrite(x, file="data", header=TRUE, sep=",", digits=16)
```

## Arguments

| | |
|---|---|
| `file` | name of file to read or write. |
| `x` | A time series or time series matrix. |
| `skip` | number of lines to skip at start of file before reading data. |
| `nseries` | number of columns of series to expect. |
| `sep` | field separator. |
| `na.strings` | charaters that should be treated as NA. |
| `header` | a logical indicating is a header line should be written. |
| `digits` | number of significant digits to print. |
| `...` | additional arguments passed to scan. |

## Details

Read and write a file with time series data. By default the file is comma separated values (csv) with one header line (the series names on write, ignored on read). The year and period are the first two columns, with series in following columns. These are wrappers for `scan` and `write`.

Beware that short digits settings will result in truncated data.

## Value

A time series vector or matrix.

## See Also

scan, write

## Examples

```
z <- ts(matrix(100 + rnorm(200),100,2), start=c(1991,1), frequency=4)
tsWrite(z, file="tmp.test.data.csv")
zz <- tsScan("tmp.test.data.csv", nseries=2)

max(abs(z - zz))
```

# Index