Time Series tfplot Package

June 2, 2021

The functions in this package are made available with

```
> library("tfplot")
```

The code from the vignette that generates this guide can be loaded into an editor with *edit(vignette("Guide", package="tfplot"))*. This uses the default editor, which can be changed using *options()*.

The functions in this package rely on the *tframe* package. While that package contains underlying programing support, and some functions that might occasionally be called by users, the functions in this package are meant primarily to be called by users. They include *tfplot*, *tfOnePlot*, *tfVisPlot*, *diffLog*, *percentChange*, *ytoypc*, *annualizedGrowth*, *trimNA*, *tsWrite*, and *tsScan*.

The intention of these function is to have defaults which generally work fairly well for producing graphs often and on an ad hoc basis. That may include much of what is done in preliminary research. Producing graphs for publication will require more work. In many cases that can be accomplished by specifying arguments to the functions, as examples here will illustrate. However, not all special effects can be accomplished with these tools. The emphasis is more on simplicity than total flexibility.

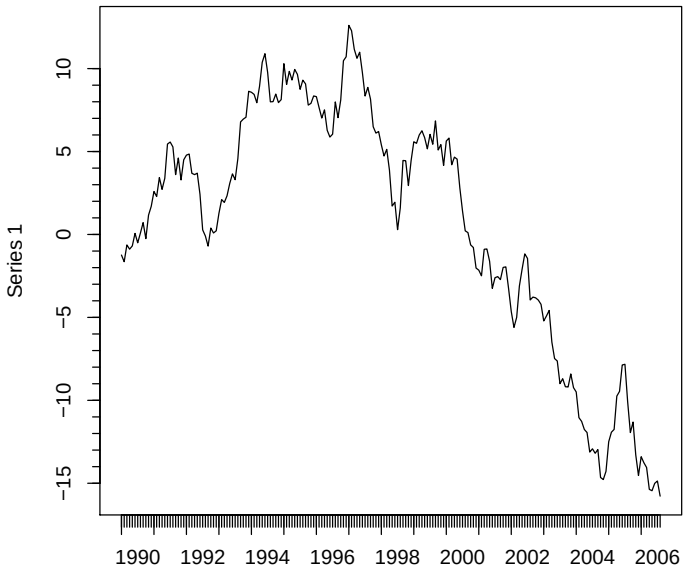
Below are some simple examples using generated data, but see the Guide vignettes for packages *TSgetSymbol*, *TShistQuote*, *TSxls*, *TSzip* for example with real data taken from the web. Also see the help for specific functions to get more details.

For illustration, two univariate and two multivariate series can be generated with:

```
>  z  <- ts(cumsum(rnorm(200)), start=c(1990, 1), frequency=12)
>  x  <- ts(cumsum(rnorm(200)), start=c(1990, 1), frequency=12)
>  zz <- ts(apply(matrix(rnorm(300), 100,3),2, cumsum),
             start=c(1990, 1), frequency=4)
>  xx <- ts(apply(matrix(rnorm(300), 100,3),2, cumsum),
             start=c(1990, 1), frequency=4)
```

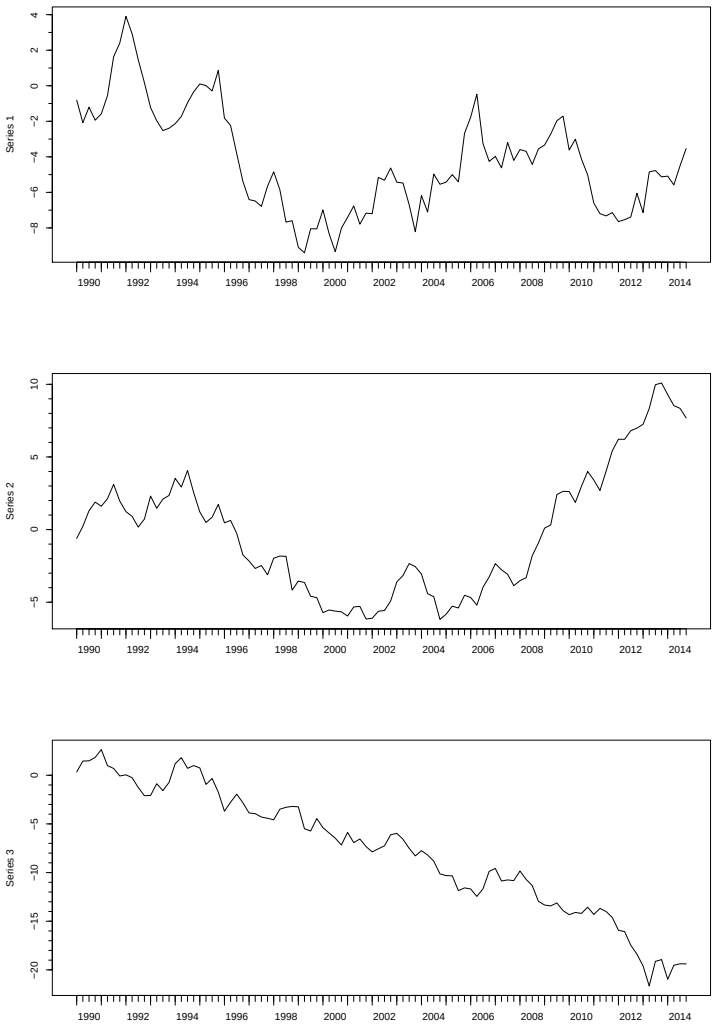
The simplest use produces a graph:

```
> tfplot(z)
```



or a panel for each series in the multivariate case:

```
> tfplot(zz)
```

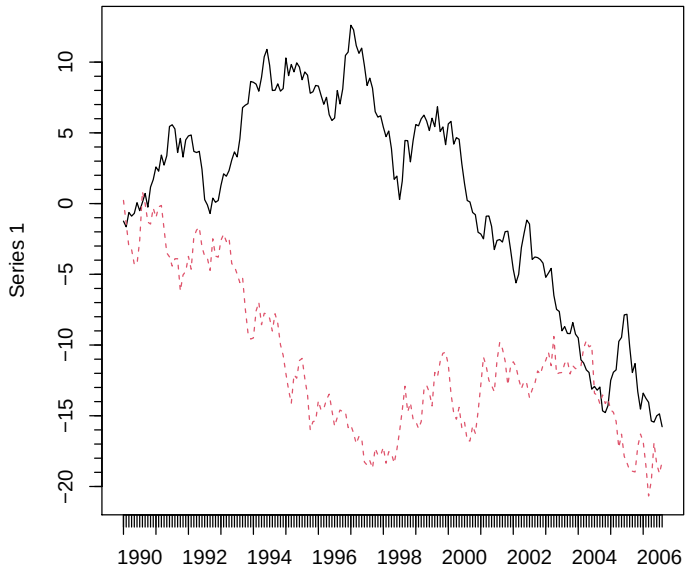


Multiple panels, as in the last graph are especial convenient when the series have different magnitudes, so showing them on the same graph does not work.

Font sizes can be controlled. Defaults are used here for simplicity, although the defaults generally work better on screen output than they do in vignettes.

If series are provided in multiple arguments then a single graph with the series from each argument are produced:

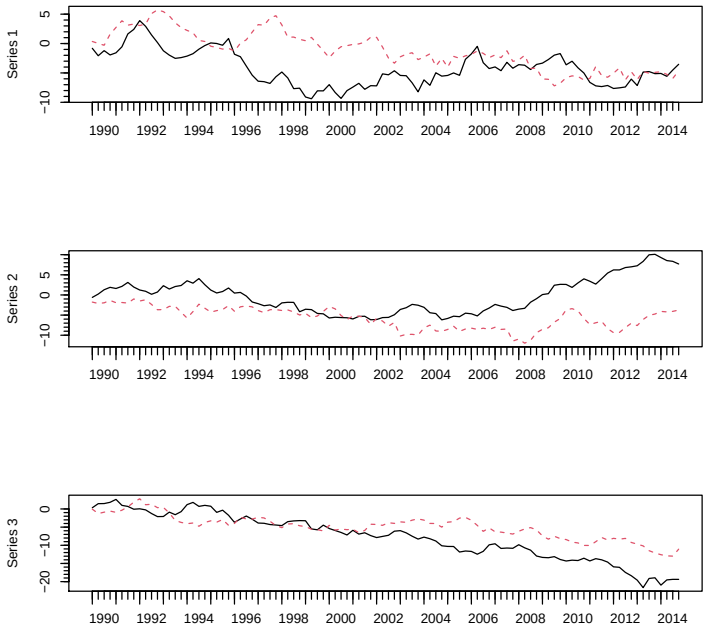
```
>   tfplot(z, x)
```



These graphs show only two time series objects, but more can be supplied as additional arguments.

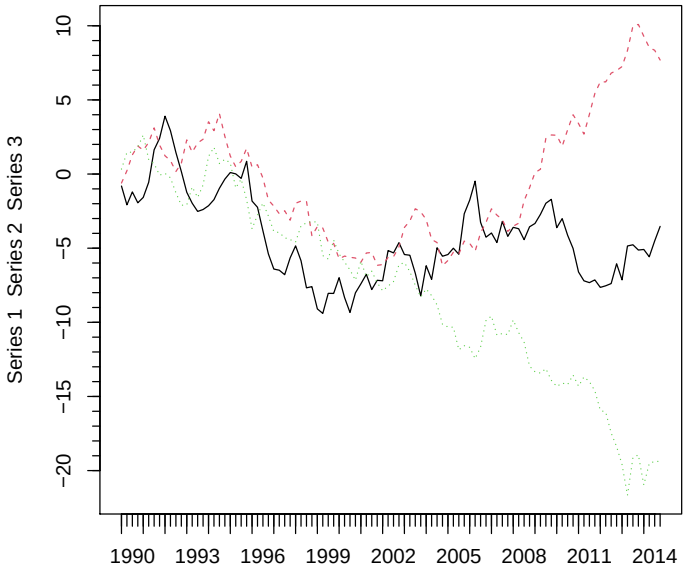
Each multivariate series argument should have the same number of series, and panels with the corresponding series from each argument are produced:

```
> tfplot(zz, xx)
```

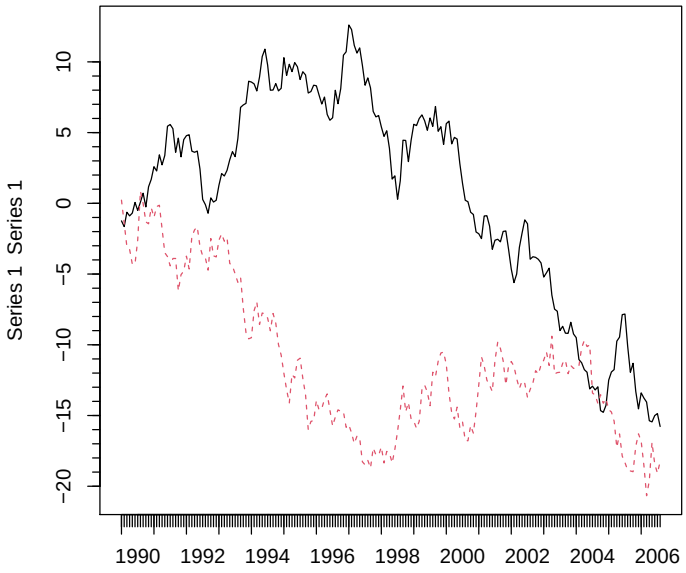


In the case of multivariate series, putting all series onto the same panel can be done with *tfOnePlot*:

```
>   par(mfcol=c(1,1))
>   tfOnePlot(zz)
```



```
>   tfOnePlot(tbind(z, x))
```

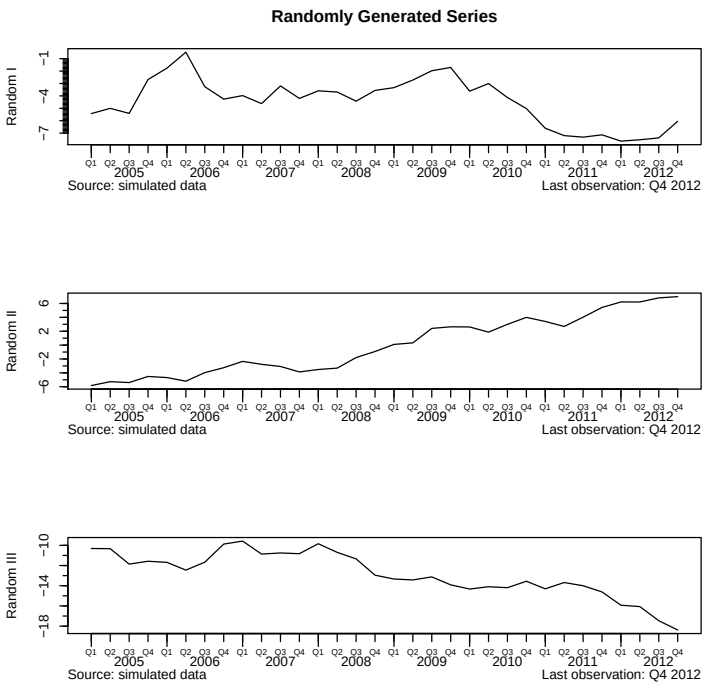


In the above *par(mfcol=c(1,1))* sets the graphic device to contain one panel. Often this is not necessary, but ocassionally the device is left in a different state by a previous command. Also, sometimes it is interesting to set it differently on purpose, for example, *par(mfcol=c(2,2))* could be used to put four graphs in two rows and two columns on a page.

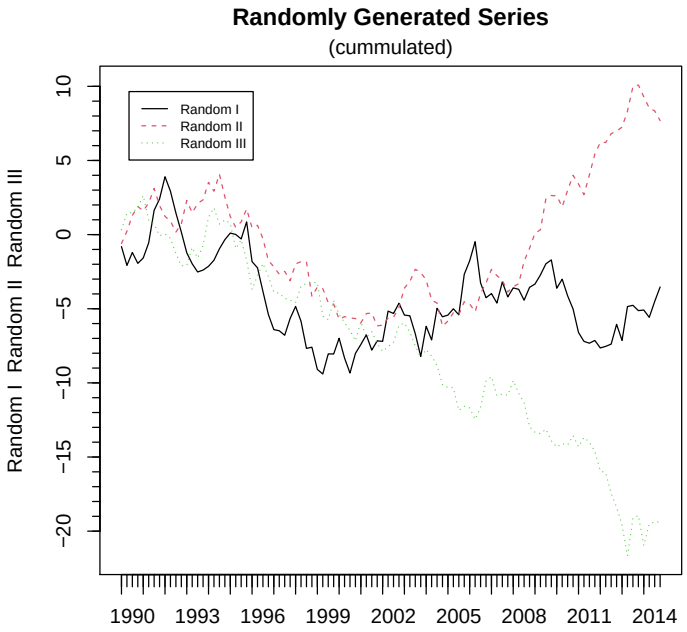
The above general rules apply in most cases, so the examples below will not be shown for all the above cases.

If the series have names then these will be used. (If the series are extracted from a database then the names may automatically be available):

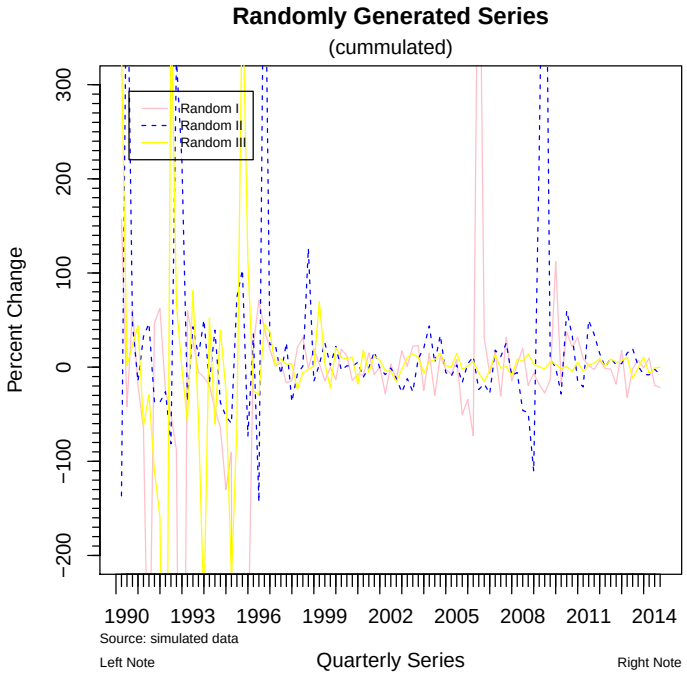
```
>   seriesNames(zz) <- c("Random I","Random II","Random III")
>   tfplot(zz, Title="Randomly Generated Series",
       start=c(2005,1), end=c(2012,4), lastObs=TRUE,
       source="Source: simulated data")
```



```
>   par(mfcol=c(1,1))
>   tfOnePlot(zz, Title="Randomly Generated Series", subtitle="(cummulated)",
        legend=c("Random I", "Random II", "Random III"))
```



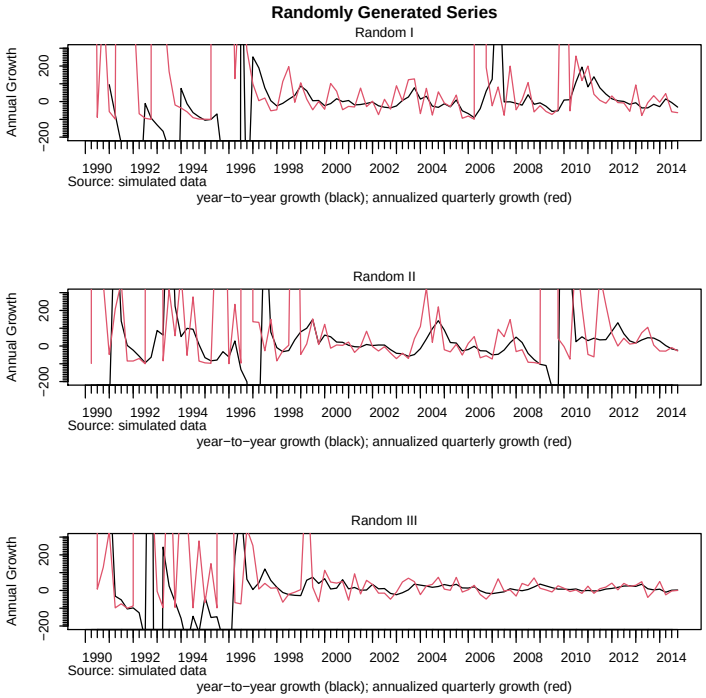
```
>   tfOnePlot(percentChange(zz), Title="Randomly Generated Series",
        subtitle="(cummulated)",
        lty=c("solid", "dashed", "solid"), col=c("pink", "blue", "yellow"),
        legend=c("Random I", "Random II", "Random III"),
        ylab="Percent Change", xlab="Quarterly Series",
        ylim=c(-200, 300),
        source="Source: simulated data",
        footnote="Left Note", footnoteRight ="Right Note")
```



See ?par for additional information on setting *lty* and *col*.

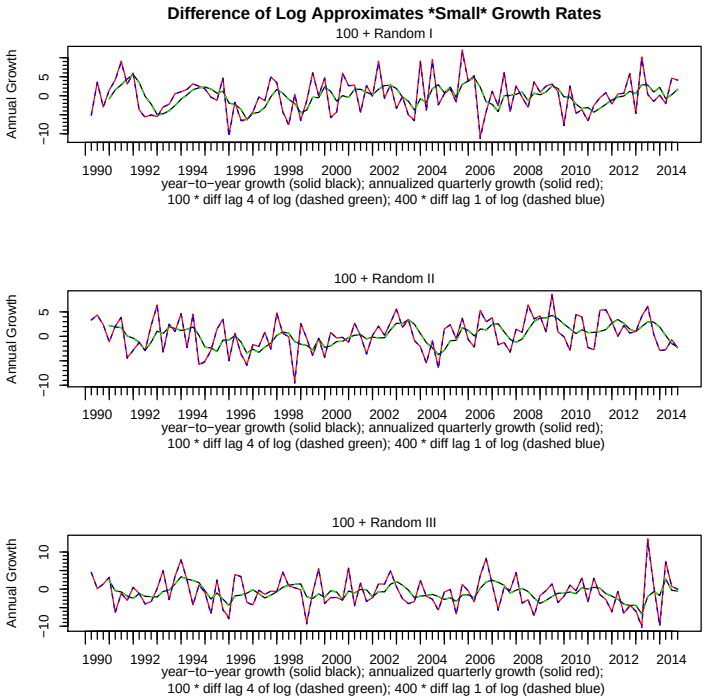
Panels for different series can be especial useful for comparing different transformations of the same series, for example year-to-year growth compared with annualized period-to-period growth:

```
>   tfplot(ytoypc(zz), annualizedGrowth(zz),
        Title="Randomly Generated Series",
        subtitle=c("Random I","Random II","Random III"),
        lty=c("solid", "solid"),
        ylab=c("Annual Growth", "Annual Growth", "Annual Growth"),
        xlab="year-to-year growth (black); annualized quarterly growth (red)",
        ylim=c(-200, 300),
        source="Source: simulated data")
```



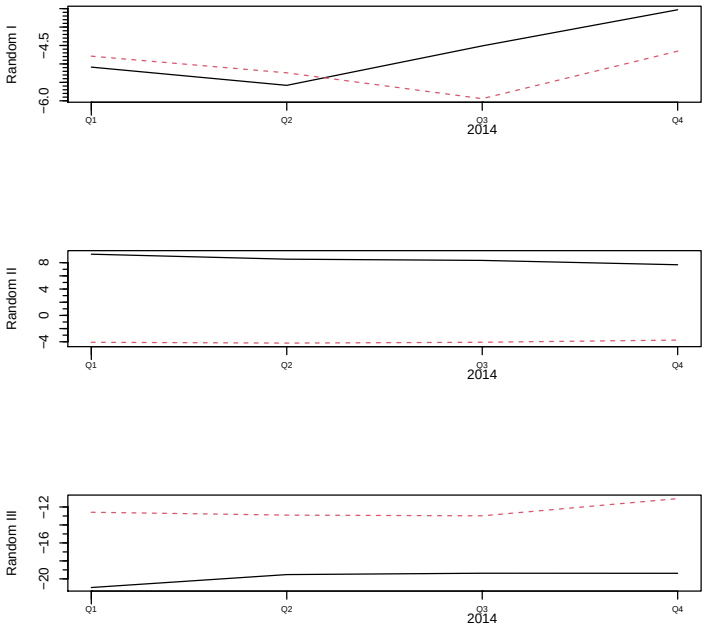
or the difference of logrithms approximation to growth rates, often used by economists:

```
>   tfplot(ytoypc(100+zz), annualizedGrowth(100+zz),
        100 * diffLog(100+zz, lag=4), 400 * diffLog(100+zz, lag=1),
        Title="Difference of Log Approximates *Small* Growth Rates",
        subtitle=c("100 + Random I","100 + Random II","100 + Random III"),
        lty=c("solid", "solid", "dashed", "dashed"),
        col=c("black", "red",   "green",  "blue"),
        ylab=c("Annual Growth", "Annual Growth", "Annual Growth"),
        xlab="year-to-year growth (solid black); annualized quarterly growth (solid red); \
    100 * diff lag 4 of log (dashed green); 400 * diff lag 1 of log (dashed blue)")
```



The function *tfplot* calls *tfOnePlot* for each panel. It, in turn, uses function *plot* from the *graphics* package. *plot* automatically determines tick marks and labelling of the x-axis.

```
>   tfplot(zz, xx, start=c(2014,1))
```



This will also be influenced by the type of time series object. For example, *zoo* objects may result in different (generally better) labeling than *ts* objects, possibly indicating quarters or months rather than fractions of a year. In general, the handling of axis labelling is an area that could use improvement (contributions welcomed).

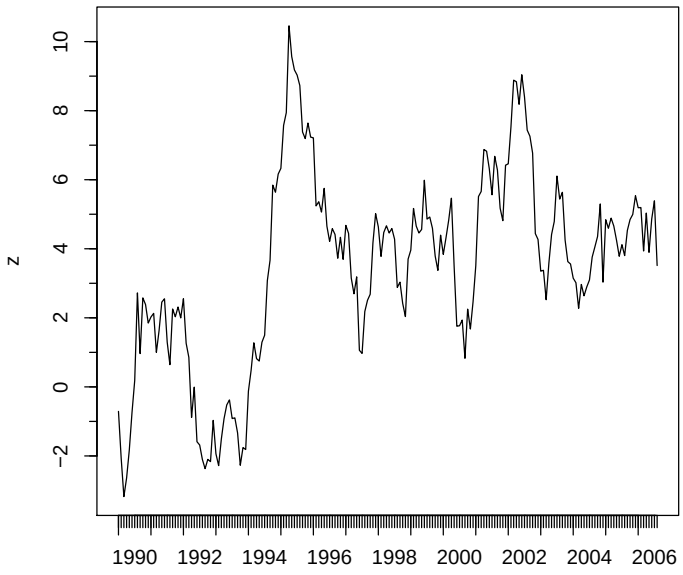
The function *tfVisPlot* can be used to generate plots in a browser, with the added functionality that your mouse can point at the graph and find additional information like the date of a point, or the series in a multi-series graph. This can be very useful when looking for outliers and other anomolies in the data. See ?tfVisPlot for more details.

Often one needs to deal with time series that have slightly different time frames, that is, start or end at different dates. Also, series are sometimes padded with *NA* on the ends, for a variety of reasons, effectively changing the start and end date of the object from that of the actually available data. Below are examples of some ways to deal with this.

```
>   z  <- ts(cumsum(rnorm(200)), start=c(1990, 1), frequency=12)
>   x  <- ts(c(rep(NA,105),cumsum(rnorm(100)),rep(NA,10)),
                                start=c(1995, 1), frequency=12)
>   seriesNames(z) <- "z"
>   seriesNames(x) <- "x"
>   start(z)
>   end(z)
>   Tobs(z)
>   tframe(z)
>   start(x)
>   end(x)
>   Tobs(x)
>   tframe(x)
>   start(trimNA(x))
>   end(trimNA(x))
>   Tobs(trimNA(x))
>   tframe(trimNA(x))
```

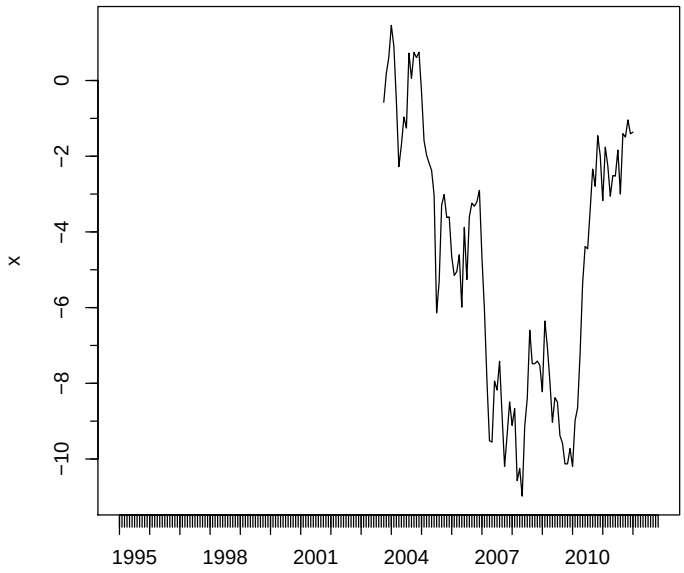
The *tfplot* default is to plot for the whole length of the object.

```
>   tfplot(z)
```


z
10
8
6
4
2
0
−2
1990
1992
1994
1996
1998
2000
2002
2004
2006

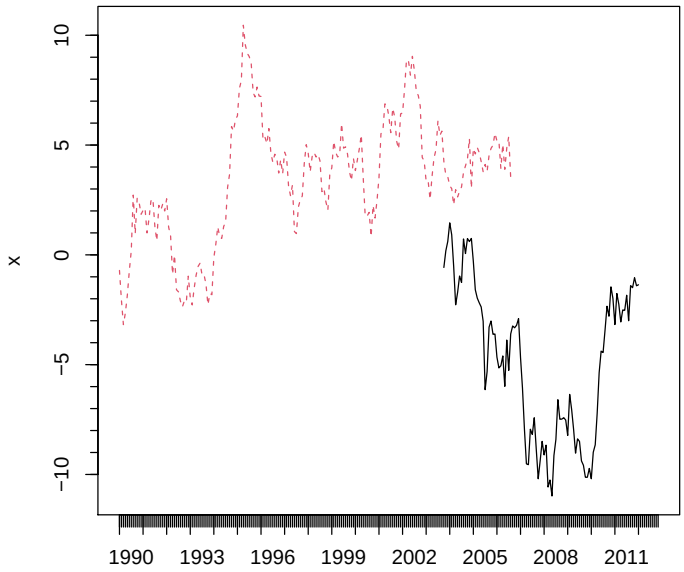
For objects with *NAs* this means that some parts of the graph are blank.

```
>   tfplot(x)
```



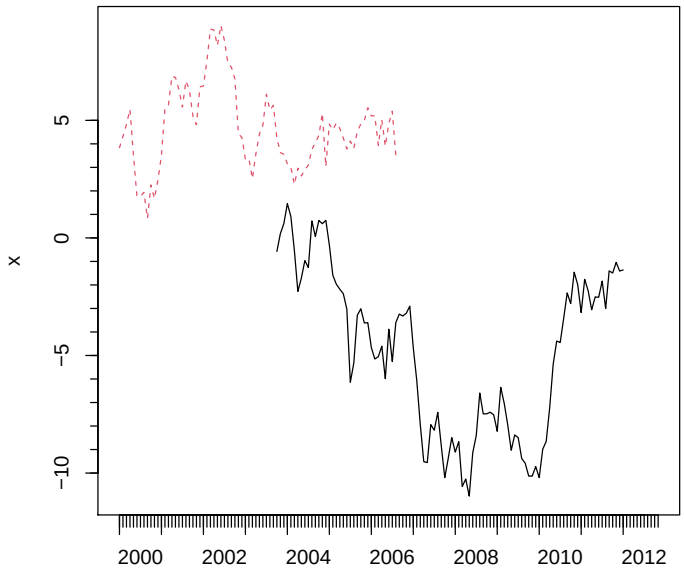
When there is more than one object, the combined time frame is used (spliced in time):

```
> tfplot(x,z)
```

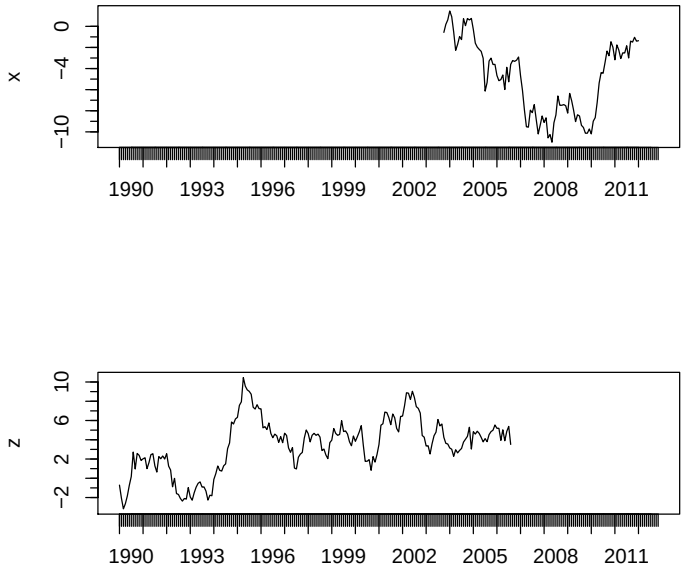


But specific dates can also be provided:

```
>   tfplot(x,z, start=c(2000,1))
```

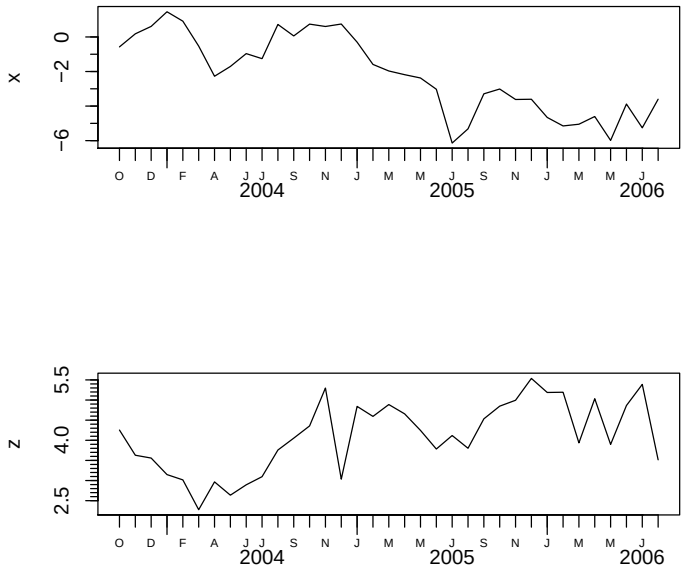


```
>   tfplot(tbind(x,z))
```



The function *trimNA* can be used to trim *NAs* from the ends of a series:

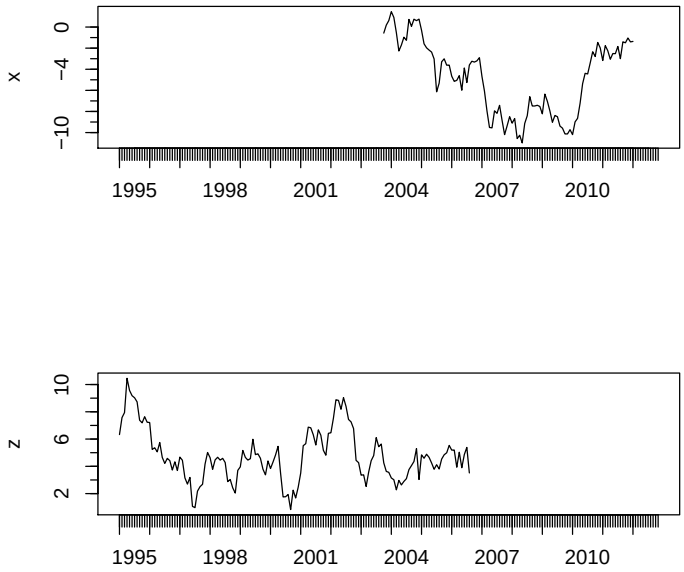
```
>   tfplot(trimNA(tbind(x,z)))
```



For a multivariate timeseries object it trims time periods for which any observation is *NA*. Options allow triming only the start or end of the object.

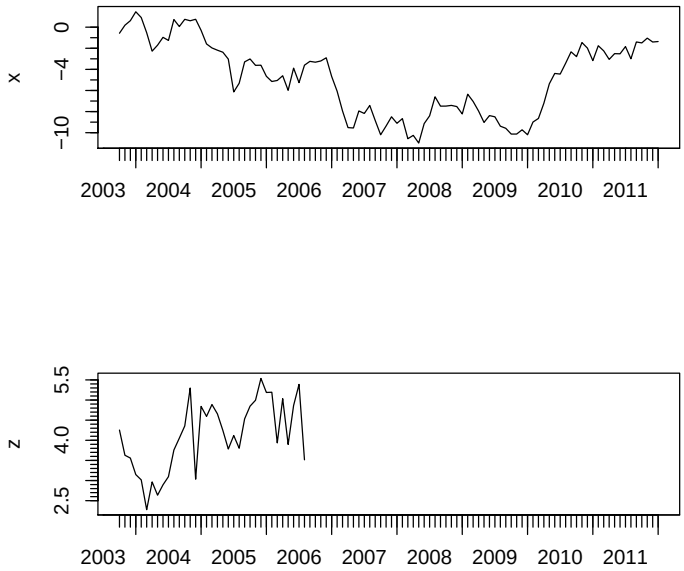
The time frame can also be specified using the time frame from one of the objects,

```
>   tfplot(tbind(x,z), tf=tframe(x))
```



Or, using the time frame of the object with *NAs* removed:

```
>   tfplot(tbind(x,z), tf=tframe(trimNA(x)))
```



As mentioned above, the guides for several of the *TSdbi* packages have additional examples. In particular, see the guide for *TShistQuote* for examples with higher frequency data and *zoo* objects.