FinTᴇX

Enrico Schumann
mailto:es@enricoschumann.net

PMWR provides several methods for `toLatex`.

Monthly returns

For a timeseries (e.g. zoo or xts), the function `returns` provides monthly returns.

```
> returns(DAX, period = "month")
```

```
      Jan Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec  YTD
2014 -1.0 4.1 -1.4  0.5  3.5 -1.1 -4.3  0.7  0.0 -1.6 7.0 -1.8 4.3
2015  9.1 6.6  5.0 -4.3 -0.4 -4.1  3.3 -9.3 -5.8 12.3 4.9 -5.6 9.6
```

To have such a table placed into a LᴬTᴇX file, you can put the following snippet into a Sweave file.

```
\begin{tabular}{rrrrrrrrrrrrrr}
<<results=tex,echo=false>>=
toLatex(returns(DAX, period = "month"), ytd = "\\textsc{ytd}")
@
\end{tabular}
```

The results will look like this:

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	YTD
2014	-1.0	4.1	-1.4	0.5	3.5	-1.1	-4.3	0.7	0.0	-1.6	7.0	-1.8	4.3
2015	9.1	6.6	5.0	-4.3	-0.4	-4.1	3.3	-9.3	-5.8	12.3	4.9	-5.6	9.6

NAVseries

Summaries of NAVseries contain a number of statistics that can be placed into LᴬTᴇX templates.

```
> returns(DAX, period = "annualised")
```

```
6.9%  [02 Jan 2014 -- 30 Dec 2015]
```

To do so, call `toLatex` with a summary of one or more NAVseries, and a `template`.

```
> toLatex(summary(as.NAVseries(DAX,  title = "DAX"),
                   as.NAVseries(REXP, title = "REXP")),
          template = "%title: %return\\% \\\\")
```

```
DAX: 6.9\% \\
REXP: 3.8\% \\
```

In recent versions of R, you can use raw character constants of the form `r"(...)"`, which reduces the need for escaping. See `?Quotes`.

Note that the template was recycled, i.e. it was used for both series. We may also pass separate templates for each series.

```
> tmpl <- c("Equities (%title) made %return\\%, with a drawdown of %mdd\\%;",
            "bonds (%title) returned %return\\%.")
> toLatex(summary(as.NAVseries(DAX,  title = "DAX"),
                  as.NAVseries(REXP, title = "REXP")),
         template = tmpl)
```

```
Equities (DAX) made 6.9\%, with a drawdown of 23.8\%;
bonds (REXP) returned 3.8\%.
```

The keyword %sparkline adds a sparkline:

```
> toLatex(summary(as.NAVseries(DAX,  title = "DAX")),
         template = "The DAX %sparkline made %return\\% during the period.")
```

The DAX made 6.9% during the period. It uses the LATEX package sparklines (https:

//ctan.org/pkg/sparklines).
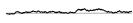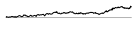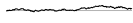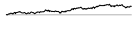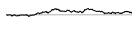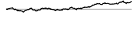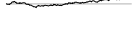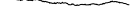
Since templates are recycled, we can easily create rows for LATEX tables, such as this one:

		Return p.a.	Volatility
DAX		6.9	18.0
REXP		3.8	1.9

…which is produced by the following call:

```
> toLatex(summary(as.NAVseries(DAX,  title = "DAX"),
                  as.NAVseries(REXP, title = "REXP")),
         template = "%title & %sparkline & %return & %volatility  \\\\")
```

When several NAV series are passed to toLatex, all sparkline plots use the same y-scale. It is then straightforward to produce tables such as the following one, in which we have sorted 50 random series by total return. (See the vignette source for the exact code, but the key command is toLatex(do.call(summary, <list of NAVseries>))).

Return	Vol		Return	Vol		Return	Vol	
40.0	1.0		5.2	1.0		-13.7	1.0	
29.7	1.0		4.6	0.9		-14.4	1.0	
23.5	1.0		4.4	1.1		-14.6	1.0	
23.2	1.0		3.4	1.0		-18.4	1.0	
21.9	1.0		3.3	1.0		-20.1	0.9	
20.4	0.9		2.5	1.0		-21.0	1.0	
18.1	1.0		2.3	1.0		-21.2	0.9	
14.0	1.1		1.3	1.0		-22.2	1.0	
13.3	1.0		-0.1	1.0		-22.3	1.0	
12.7	0.9		-0.2	1.0		-22.9	1.0	
11.7	1.0		-1.9	1.0		-25.9	1.1	
11.0	1.0		-4.3	1.0		-26.0	1.0	
8.5	1.0		-5.1	1.0		-27.9	1.0	
8.0	1.0		-7.6	1.1		-28.3	1.0	
7.8	1.1		-8.7	1.1		-34.9	1.0	
6.3	0.8		-10.8	1.1		-39.5	1.0	
6.2	1.0		-12.2	1.0				