

# How to Present Tables in Plot Devices

Peter Carl

Chicago R User Group Meetup: R Output

# Outline

1 Overview

2 Example

3 Potential Solutions

# Overview

Graphics in R are plotted on a graphics device

- Depending on the OS, in an interactive R session the default device is the screen, using `windows()`, `X11()`, or `quartz()`.
- Common graphics file formats use the `bmp()`, `jpeg()`, `png()`, and `tiff()` devices.
- Other useful file devices include `postscript()`, `pdf()`, `pictex()`, `xfig()`, and `bitmap()`.

Why would we display tabular data on a plot device?

- Reviewing results in a terminal isn't usually effective
- Garner benefits from formatting
- Combining graphics and tables can be very powerful

Some solutions, with a focus on `textplot`

# Set up an example

```

> library('PerformanceAnalytics')
> data(managers)
> #managers=read.csv("/home/peter/dev/R/managers.csv",row.names=1)
> head(managers)

```

	HAM1	HAM2	HAM3	HAM4	HAM5	HAM6	EDHEC	LS	EQ	SP500	TR	US	10Y	TR	US	3m	TR
1996-01-31	0.0074	NA	0.0349	0.0222	NA	NA			NA	0.0340		0.00380		0.00456			
1996-02-29	0.0193	NA	0.0351	0.0195	NA	NA			NA	0.0093		-0.03532		0.00398			
1996-03-31	0.0155	NA	0.0258	-0.0098	NA	NA			NA	0.0096		-0.01057		0.00371			
1996-04-30	-0.0091	NA	0.0449	0.0236	NA	NA			NA	0.0147		-0.01739		0.00428			
1996-05-31	0.0076	NA	0.0353	0.0028	NA	NA			NA	0.0258		-0.00543		0.00443			
1996-06-30	-0.0039	NA	-0.0303	-0.0019	NA	NA			NA	0.0038		0.01507		0.00412			

```

> dim(managers)
[1] 132 10
> colnames(managers)
[1] "HAM1" "HAM2" "HAM3" "HAM4" "HAM5" "HAM6" "EDHEC"

```

# Set up an example

```
> manager.col = 1
> peers.cols = c(2,3,4,5,6)
> indexes.cols = c(7,8)
> Rf.col = 10
> peer.colorset=c("red", rep("darkorange", 2), rep("gray", 5))
> ham1.downside = t(table.DownsideRisk(managers[,c(manager.col,
+ indexes.cols, peers.cols)],Rf=.03/12))
```

# Construct a table example

```
> ham1.downside
```

	Semi Deviation	Gain Deviation	Loss Deviation	Downside Deviation	Downside Deviation (MAR=10%)
HAM1	0.0191	0.0169	0.0211		0.0178
EDHEC LS EQ	0.0145	0.0143	0.0118		0.0138
SP500 TR	0.0325	0.0250	0.0300		0.0323
HAM2	0.0201	0.0347	0.0107		0.0164
HAM3	0.0237	0.0290	0.0191		0.0214
HAM4	0.0395	0.0311	0.0365		0.0381
HAM5	0.0324	0.0313	0.0324		0.0347
HAM6	0.0175	0.0149	0.0128		0.0161

Modified ES (95%)

HAM1	-0.0610
EDHEC LS EQ	-0.0346
SP500 TR	-0.0944
HAM2	-0.0614
HAM3	-0.0440
HAM4	-0.1176
HAM5	-0.0974
HAM6	-0.0390

# textplot

Gregory R. Warnes' package, `gplots`, includes the `textplot` function

- Displays text output in a graphics window
- Provides the equivalent of `print`
- Creates a new plot and displays a table using the largest font that will fit in the plotting region
- Several other good things in the package, too
- `testplot` function added to `PerformanceAnalytics`

# textplot example

```
> #args(textplot)
> textplot(ham1.downside); box(col="lightblue")
```

	Semi Deviation	Gain Deviation	Loss Deviation	Downside Deviation (MAR=10%)	Downside Deviation (Rf=3%)	Downside Deviation (0%)	Maximum Drawdown	Historical VaR (95%)	Historical ES (95%)	Modified VaR (95%)	Modified ES (95%)
<b>HAM1</b>	0.0191	0.0169	0.0211	0.0178	0.0154	0.0145	0.1518	-0.0258	-0.0513	-0.0342	-0.061
<b>EDHEC I.S EQ</b>	0.0145	0.0143	0.0118	0.0138	0.0109	0.0098	0.1075	-0.0203	-0.0342	-0.0235	-0.0346
<b>SP500 TR</b>	0.0325	0.025	0.03	0.0323	0.0295	0.0283	0.4473	-0.0669	-0.0933	-0.0683	-0.0944
<b>HAM2</b>	0.0201	0.0347	0.0107	0.0164	0.0129	0.0116	0.2399	-0.0294	-0.0331	-0.0276	-0.0614
<b>HAM3</b>	0.0237	0.029	0.0191	0.0214	0.0185	0.0174	0.2894	-0.0425	-0.0555	-0.0368	-0.044
<b>HAM4</b>	0.0395	0.0311	0.0365	0.0381	0.0353	0.0341	0.2874	-0.0799	-0.1122	-0.0815	-0.1176
<b>HAM5</b>	0.0324	0.0313	0.0324	0.0347	0.0316	0.0304	0.3405	-0.0733	-0.1023	-0.0676	-0.0974
<b>HAM6</b>	0.0175	0.0149	0.0128	0.0161	0.0133	0.0121	0.0788	-0.0341	-0.0392	-0.0298	-0.039

## Hmisc:::format.df

The `Hmisc` package by Frank E. Harrell, Jr., and Richard M. Heiberger contains several functions useful for data analysis

- Includes functions for advanced table making, character string manipulation, and conversion of `S` objects to LaTeX code, and many others.
- `format.df` does rounding and decimal alignment for `data.frames`, similar to `format` in `base`
- Generates a character matrix containing the formatted data
- Useful for formatting tables in LaTeX or HTML, as well

# Hmisc:::format.df example

```
> library(Hmisc)
> args(format.df)

function (x, digits, dec = NULL, rdec = NULL, cdec = NULL, numeric.dollar = !dcolumn,
  na.blank = FALSE, na.dot = FALSE, blank.dot = FALSE, col.just = NULL,
  cdot = FALSE, dcolumn = FALSE, matrix.sep = " ", scientific = c(-4,
  4), math.row.names = FALSE, already.math.row.names = FALSE,
  math.col.names = FALSE, already.math.col.names = FALSE, double.slash = FALSE,
  format.Date = "%m/%d/%Y", format.POSIXt = "%m/%d/%Y %H:%M:%OS",
  ...)
NULL

> ham1.f.downside = format.df(ham1.downside, na.blank=TRUE, numeric.dollar = FALSE, cdec=rep(4,d
```

## Hmisc:::format.df example

```
> ham1.f.downside
```

	Semi Deviation	Gain Deviation	Loss Deviation	Downside Deviation (MAR=10\%)	Downside
HAM1	"0.0191"	"0.0169"	"0.0211"	"0.0178"	"0.0154"
EDHEC LS EQ	"0.0145"	"0.0143"	"0.0118"	"0.0138"	"0.0109"
SP500 TR	"0.0325"	"0.0250"	"0.0300"	"0.0323"	"0.0295"
HAM2	"0.0201"	"0.0347"	"0.0107"	"0.0164"	"0.0129"
HAM3	"0.0237"	"0.0290"	"0.0191"	"0.0214"	"0.0185"
HAM4	"0.0395"	"0.0311"	"0.0365"	"0.0381"	"0.0353"
HAM5	"0.0324"	"0.0313"	"0.0324"	"0.0347"	"0.0316"
HAM6	"0.0175"	"0.0149"	"0.0128"	"0.0161"	"0.0133"

	Modified VaR (95\%)	Modified ES (95\%)
HAM1	"-0.0342"	"-0.0610"
EDHEC LS EQ	"-0.0235"	"-0.0346"
SP500 TR	"-0.0683"	"-0.0944"
HAM2	"-0.0276"	"-0.0614"
HAM3	"-0.0368"	"-0.0440"
HAM4	"-0.0815"	"-0.1176"
HAM5	"-0.0676"	"-0.0974"
HAM6	"-0.0298"	"-0.0390"

```
attr(,"col.just")
```

```
[1] "r" "r"
```

# PerformanceAnalytics::textplot

The PerformanceAnalytics package extends the `gplots::textplot` function

- Equivalent of `print` except that the output is displayed as a plot
- Fixes some of the layout math
- Adds column and row name word wrapping
- Adds color to the table elements
- Adds vertical alignment for headers and data

# PerformanceAnalytics::textplot example

```
> require(PerformanceAnalytics)
> args(PerformanceAnalytics::textplot)

function (object, halign = "center", valign = "center", cex,
  max.cex = 1, cmar = 2, rmar = 0.5, show.rownames = TRUE,
  show.colnames = TRUE, hadj = 1, vadj = NULL, row.valign = "center",
  heading.valign = "bottom", mar = c(0, 0, 0, 0) + 0.1, col.data = par("col"),
  col.rownames = par("col"), col.colnames = par("col"), wrap = TRUE,
  wrap.colnames = 10, wrap.rownames = 10, ...)

NULL
```

# PerformanceAnalytics::textplot example

```
> PerformanceAnalytics::textplot(ham1.f.downside, halign = "center", valign = "top", row.valign = "top",
> box(col="lightblue"))
```

	Semi Deviation	Gain Deviation	Loss Deviation	Downside Deviation (MAR=10%)	Downside Deviation (Rf=3%)	Downside Deviation (0%)	Maximum Drawdown	Historical VaR (95%)	Historical ES (95%)	Modified VaR (95%)	Modified ES (95%)
HAM1	0.0191	0.0169	0.0211	0.0178	0.0154	0.0145	0.1518	-0.0258	-0.0513	-0.0342	-0.0310
EDHEC LS EQ	0.0145	0.0143	0.0118	0.0138	0.0109	0.0098	0.1075	-0.0203	-0.0342	-0.0235	-0.0346
SP500 TR	0.0325	0.0250	0.0300	0.0323	0.0295	0.0283	0.4473	-0.0669	-0.0933	-0.0683	-0.0944
HAM2	0.0201	0.0347	0.0107	0.0164	0.0129	0.0116	0.2399	-0.0294	-0.0331	-0.0276	-0.0314
HAM3	0.0237	0.0290	0.0191	0.0214	0.0185	0.0174	0.2894	-0.0425	-0.0555	-0.0368	-0.0440
HAM4	0.0395	0.0311	0.0365	0.0381	0.0353	0.0341	0.2874	-0.0799	-0.1122	-0.0815	-0.1176
HAM5	0.0324	0.0313	0.0324	0.0347	0.0316	0.0304	0.3405	-0.0733	-0.1023	-0.0676	-0.0974
HAM6	0.0175	0.0149	0.0128	0.0161	0.0133	0.0121	0.0788	-0.0341	-0.0392	-0.0298	-0.0390

# Other Possibilities

What else is available?

- A very promising package presented at useR! 2010, `tabularR`
- Dump results to a spreadsheet, perhaps with `XLConnect`
- Finally learn  $\LaTeX$  and Sweave
- What did I miss? Any feedback would be much appreciated ...