Visualizing in R
– advanced plotting

Norsk statistikermøte, Halden, 10. juni 2013

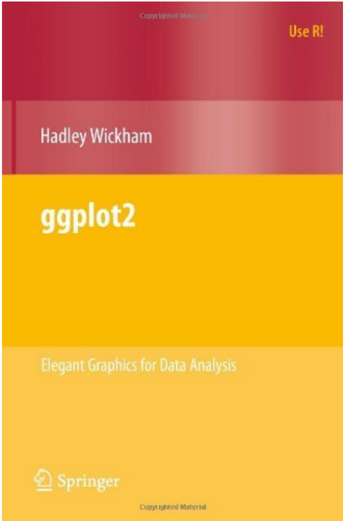
Elisabeth Orskaug
Thordis Thorarinsdottir

Norsk Regnesentral

Outline of the lectures

- Monday
  - Lecture I: **Basic plotting** (Thordis)
  - Lecture II: **Displaying multivariate data** (Thordis)
  - Lecture III: **Introduction to ggplot2** (Thordis)
- Tuesday
  - Lecture IV: **Spatial maps with ggplot2** (Elisabeth)
  - Lecture V: **RgoogleMaps** (Elisabeth)
  - Lecture VI: **Animations** (Elisabeth)

Further reading



Copyrighted Material

Use R!

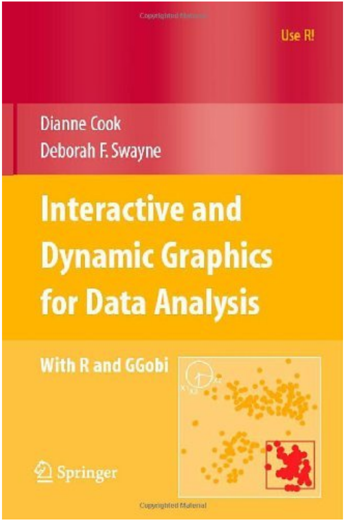
Hadley Wickham

ggplot2

Elegant Graphics for Data Analysis

Springer

Copyrighted Material



Copyrighted Material

Use R!

Dianne Cook
Deborah F. Swayne

Interactive and Dynamic Graphics for Data Analysis

With R and GGobi

Springer

Copyrighted Material

Before we start

- Download the course material from

  `http://www.nr.no/~thordis/vis.html`

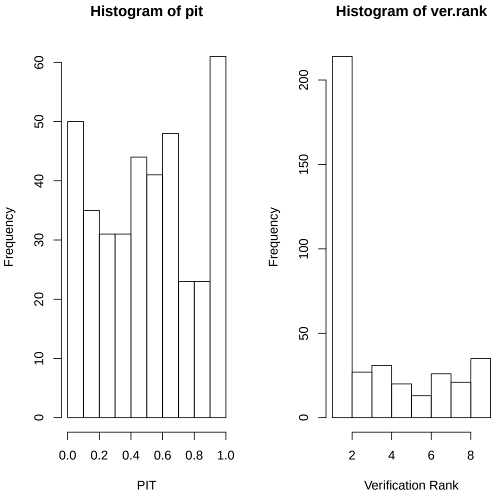
- Open `R` and set your working directory to

  `../VisualizationInR/`

1. Basic plotting

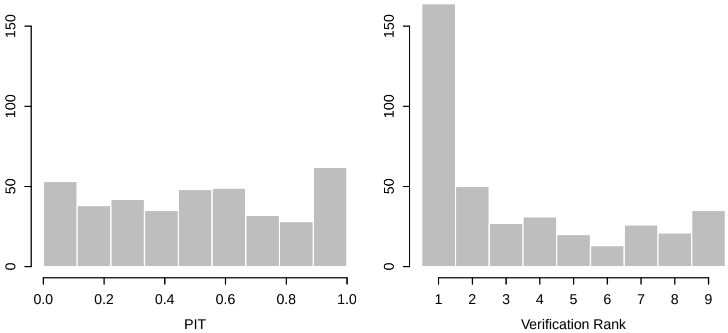
- Histograms
- Polygons and transparent layers
- Plotting object classes
- Colors and outputs
- Assignment

What is wrong here?



Both histograms are based on 387 variables.
Variables on the left take values in $[0, 1]$ while those on the right take valeus in $\{1, 2, \ldots, 9\}$.

Better



Both histograms are based on 387 variables.
Variables on the left take values in $[0, 1]$ while those on the right take valeus in $\{1, 2, \ldots, 9\}$.

Let us look at the code

```
pdf(file="../hist1.pdf")
par(mfrow=c(1,2))
hist(pit, xlab="PIT")
hist(ver.rank, xlab="Verification Rank")
dev.off()
```

```
pdf(file="../hist2.pdf", width=6, height=3, points=8)
par(mfrow=c(1,2), mar=c(5,3,1,1), mex=0.75)
hist(pit, xlab="PIT", ylab="", ylim=c(0,200),
     breaks=c(0:9)/9, col="gray", border="white", main="")
hist(ver.rank, xlab="Verification Rank", ylab="",
     ylim=c(0,200), breaks=c(0:9)+0.5, col="gray",
     border="white", main="", axes=FALSE)
axis(1, at=1:9)
axis(2)
dev.off()
```

All of this is documented in the R help files

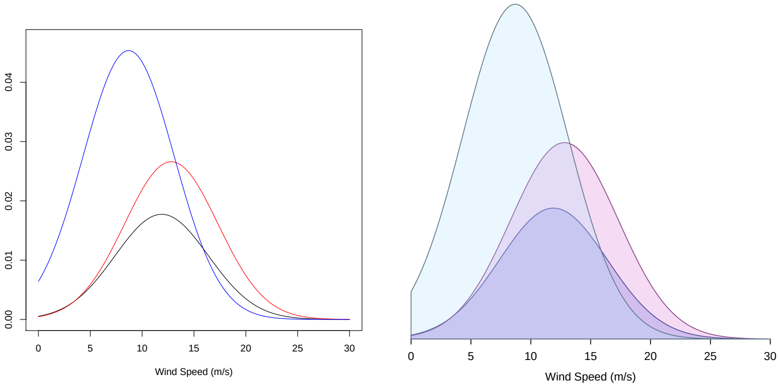
```
> ?par
```

```
'mar' A numerical vector of the form 'c(bottom, left, top, right)'
     which gives the number of lines of margin to be specified on
     the four sides of the plot.  The default is 'c(5, 4, 4, 2) +
     0.1'.

'mex' 'mex' is a character size expansion factor which is used to
     describe coordinates in the margins of plots. Note that this
     does not change the font size, rather specifies the size of
     font (as a multiple of 'csi') used to convert between 'mar'
     and 'mai', and between 'oma' and 'omi'.

     This starts as '1' when the device is opened, and is reset
     when the layout is changed (alongside resetting 'cex').
```

Polygons and transparent layers



Simple plot

```
x <- seq(0, 30, by=0.01)
x.mu <- c(11.89, 12.81, 8.70)
x.sigma <- c(4.5, 4.5, 4.4)

pdf(file="../dist1.pdf")
y <- 0.2*dtnorm(x, x.mu[1], x.sigma[1])
plot(x, y, type="l", ylim=c(0, 0.047),
     xlab="Wind Speed (m/s)", ylab="")
y <- 0.3*dtnorm(x, x.mu[2], x.sigma[2])
lines(x, y, col="red")
...
dev.off()
```

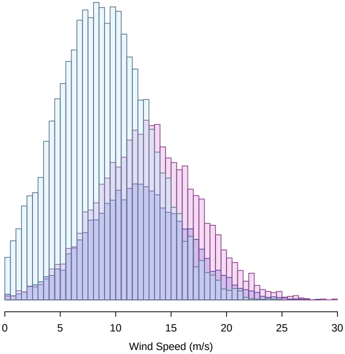
Similarly, we can use `abline()` to add straight lines, or `points()` to add points to a plot.

Code for polygons and transparent layers

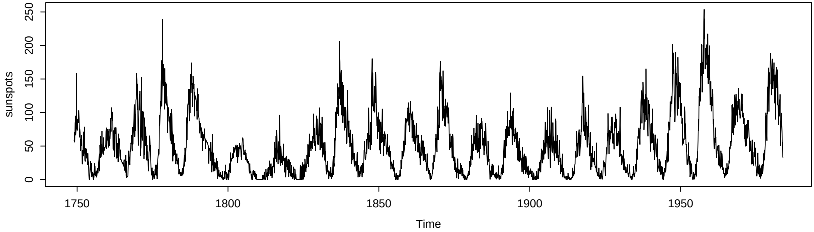
```
pdf(file="../dist2.pdf", width=5, height=5, points=10)
par(mar=c(5,1,1,1), mex=0.75)
y <- 0.2*dtnorm(x, x.mu[1], x.sigma[1])
n <- length(y)
plot(x=NULL, y=NULL, xlim=c(0,30), ylim=c(0, 0.047),
     xlab="Wind Speed (m/s)", ylab="", axes=FALSE, yaxs="i")
axis(1)
polygon(c(x, rev(x), x[1]), c(rep(0,n), rev(y), y[1]),
        col="#4169E140", border="#27408B")
...
dev.off()
```

We can create a similar plot with histograms

```
pdf(...)
par(...)
z <- rtnorm(10000, x.mu[1],
            x.sigma[1], lower=0)
y <- hist(z, breaks=50,
          plot=FALSE)
y$density <- 0.2*y$density
plot(y, freq=FALSE,  ...)
axis(1)
...
plot(..., add=TRUE)
...
dev.off()
```

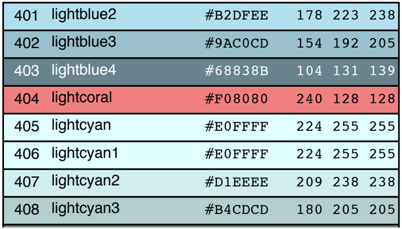


Many object classes have built-in plot functions

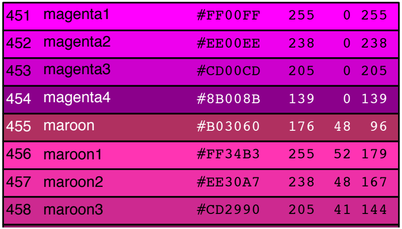


```
> data(sunspots)
> class(sunspots)
[1] "ts"
> pdf(...)
> par(...)
> plot(sunspots)
> dev.off()
```

R has vast color options



401	lightblue2	#B2DFEE	178	223	238
402	lightblue3	#9AC0CD	154	192	205
403	lightblue4	#68838B	104	131	139
404	lightcoral	#F08080	240	128	128
405	lightcyan	#E0FFFF	224	255	255
406	lightcyan1	#E0FFFF	224	255	255
407	lightcyan2	#D1EEEE	209	238	238
408	lightcyan3	#B4CDCD	180	205	205



451	magenta1	#FF00FF	255	0	255
452	magenta2	#EE00EE	238	0	238
453	magenta3	#CD00CD	205	0	205
454	magenta4	#8B008B	139	0	139
455	maroon	#B03060	176	48	96
456	maroon1	#FF34B3	255	52	179
457	maroon2	#EE30A7	238	48	167
458	maroon3	#CD2990	205	41	144

The full chart is available at

http://research.stowers-institute.org/efg/R/Color/Chart/ColorChart.pdf

The colors can be made transparent by adding a number behind the code, “#RRGGBBAA”, where the portion AA is the opacity/transparency.

If using color figures in a paper, try to pick colors that can be differentiated in gray-scale as well

Recommended graphics output

Software	Recommended graphics device
latex	ps
pdflatex	pdf, png (600 dpi)
MS Office	png (600 dpi)
Open Office	png (600 dpi)
web	png (72 dpi)

If using LaTeX, you can use

```
DeclareGraphicsExtensions{.png, .pdf}
```

This way, png files are picked before pdf. You can then produce all the figures in pdf, and afterwards go back and re-render the big ones as png.

Saving your plots

```
pdf(file, width, height, fonts, bg, fg, pointsize, ...)
```

```
postscript(file, fonts, bg, fg, width, height, horizontal,
           pointsize, ...)
```

```
png(filename, width = 480, height = 480, units = "px",
    pointsize = 12, bg = "white",  res = NA, ...)
```

Assignment: The model

Section 1.3

We are given a set of temperature forecasts $T_1, \dots, T_{20}$ at a single location.

We want to visualize these forecasts, the observation, and two alternative probabilistic forecasts,

$$f(x) = \frac{1}{\sigma_{T_1,\dots,T_{20}}} \varphi\left(\frac{x - \mu_{T_1,\dots,T_{20}}}{\sigma_{T_1,\dots,T_{20}}} \right)$$

and

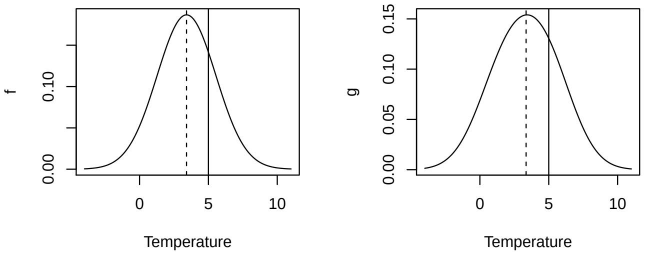
$$g(x) = \sum_{i=1}^{20} \omega_i \frac{1}{\sigma} \varphi\left(\frac{x - \mu_{T_i}}{\sigma} \right),$$

where φ denotes the density of the standard normal distribution.

Assignment: The data

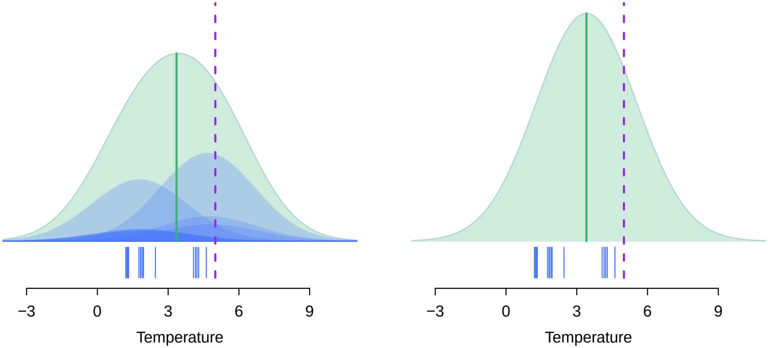
ens	vector of 20 temperature forecasts
obs	the observed temperature
x	a vector of *x*-values for plotting
f	the density values for forecast f
med.f	the median value of f
g	the density values for forecast g
med.g	the median value of g
p	the 20 densities that make up g (one column each)
weights	the weights for the densities in p

Assignment 1: A simple example



```
par(mfrow=c(1,2))
plot(x, f, type="l", xlab="Temperature")
abline(v=obs); abline(v=med.f, lty=2)
plot(x, g, type="l", xlab="Temperature")
abline(v=obs); abline(v=med.g, lty=2)
```

Assignment 1: Suggested solution



Assignment 1: Code for solution

```
pdf(file="fig//temp2.pdf", width=8, height=3.5, points=11)
par(mfrow=c(1,2), mex=0.7, mar=c(4.5,1,1,1)+0.01)
k <- 0.03
x.min <- min(x)
x.max <- max(x)
dens.max <- max(c(f, g))
z <- 0*x
dens.color <- "#4876FF40"
dens2.color <- "#3CB37140"
## plot forecast g
plot(x, g+k, type="l", col=dens.color, axes=FALSE, xlab="Temperature",
     ylab="", xlim=c(x.min,x.max), ylim=c(0,dens.max+k))
axis(1, at=c(-3,0,3,6,9))
polygon(c(x, rev(x), x[1]), c(z+k, rev(g+k), z[1]+k), col=dens2.color, border=dens2.color)
## add the mixture densities
for(i in 1:20)
{
  p.tmp <- weights[i]*p[,i]
  polygon(c(x, rev(x), x[1]), c(z+k, rev(p.tmp+k), z[1]+k), col=dens.color, border=dens.color)
}
## plot ensemble forecasts
for (j in 1:length(ens)) lines(c(ens[j], ens[j]), c(0, 0.025), lty=1, lwd=1, col="royalblue1")
## plot forecast median and observation
ind <- which.min(abs(x-med.g))
lines(c(med.g, med.g),c(k, g[ind]+k), lwd=2, lty=1, col="mediumseagreen")
lines(c(obs, obs), c(0, 0.25), lwd=2, lty=2, col="purple")
...
dev.off()
```