Graphics in R

Timothée Bonnet

February 20, 2020

BDSI / RSB

R-Stats workshops, general info

- Schedule on Wattle

R-Stats workshops, general info

- Schedule on Wattle
- Slack RSB-R-Stats-Biology email me if you are interested

R-Stats workshops, general info

- Schedule on Wattle
- Slack RSB-R-Stats-Biology email me if you are interested
- Name tag please!

R-Stats workshops, general info

- Schedule on Wattle
- Slack RSB-R-Stats-Biology email me if you are interested
- Name tag please!
- Green and orange sticky notes!

R-Stats workshops, general info

- Schedule on Wattle
- Slack RSB-R-Stats-Biology email me if you are interested
- Name tag please!
- Green and orange sticky notes!
- Please fill in survey
  `https://anu.au1.qualtrics.com/jfe/form/SV_23HWGb8pGSfyHxr`

Today

- A taste of base plot
- Playing with ggplot
- Demo of what R can do
- Reproducibility: do everything in R (no external editing!)

Today

- A taste of base plot
- Playing with ggplot
- Demo of what R can do
- Reproducibility: do everything in R (no external editing!)

Take notes in a R script. Comment using

```
# comment within R code
#' comment converted to Markdown
```

Compile HTML, Word or PDF with `CTRL + Maj + K`

Today

- A taste of base plot
- Playing with ggplot
- Demo of what R can do
- Reproducibility: do everything in R (no external editing!)

Take notes in a R script. Comment using

```
# comment within R code
#' comment converted to Markdown
```

Compile HTML, Word or PDF with `CTRL + Maj + K`

If you are bored, go to the end for some exercises

the plot() function

Basic scatterplot:

```r
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```

the plot() function

Basic scatterplot:

```r
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```

Change plotting range:

```r
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     xlim=c(0,max(iris$Sepal.Length)))
```

the plot() function

Basic scatterplot:

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```

Change plotting range:

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     xlim=c(0,max(iris$Sepal.Length)))
```

Axes labels

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     xlab="Sepal length")
```

the plot() function

Basic scatterplot:

```r
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```

Change plotting range:

```r
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     xlim=c(0,max(iris$Sepal.Length)))
```

Axes labels

```r
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     xlab="Sepal length")
```

Practice: Change the y-axis range and label

graphical parameters

```r
?par()
```

graphical parameters

```r
?par()
```

default:

```r
par(las=0, mar=c(4,4,4,4)+0.1, lwd=1, pch=1, cex.axis=1)
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```

better?

```r
par(las=1, mar=c(4,4,1,1)+0.1, lwd=5, pch=16, cex.axis=1.5)
plot(x = iris$Sepal.Length, y = iris$Sepal.Width)
```

Colors

By group:

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=iris$Species)
```

Colors

By group:

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=iris$Species)
```

Customized palette

```
mycol <- c("orange", "dark red", "brown")

plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=mycol[iris$Species])
```

Generate palettes

By value:

```r
colval <- rgb(red = iris$Sepal.Length,green = iris$Sepal.Width,
             blue = max(iris$Sepal.Length)-iris$Sepal.Length,
             maxColorValue = max(iris$Sepal.Length))

plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=colval)
```

Generate palettes

By value:

```r
colval <- rgb(red = iris$Sepal.Length,green = iris$Sepal.Width,
             blue = max(iris$Sepal.Length)-iris$Sepal.Length,
             maxColorValue = max(iris$Sepal.Length))

plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=colval)
```

Pre-defined gradients

```r
custcol <- rainbow(20)
plot(1:20, col=custcol, pch=16, cex=3)

custcol <- terrain.colors(20)
plot(1:20, col=custcol, pch=16, cex=3)
```

Adding stuff

Always in successive layers starting with a primary plot

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=mycol[iris$Species], pch=16)
```

Adding stuff

Always in successive layers starting with a primary plot

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=mycol[iris$Species], pch=16)
```

lines:

```
abline(lm(Sepal.Width~Sepal.Length, data=iris), lwd=5)
lines(x = 5:8, y=5:8/2, lty=2, lwd=2)
segments(x0 = 4.5, y0 = 2, x1 = 6, y1=4)
```

Adding stuff

Always in successive layers starting with a primary plot

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=mycol[iris$Species], pch=16)
```

lines:

```
abline(lm(Sepal.Width~Sepal.Length, data=iris), lwd=5)
lines(x = 5:8, y=5:8/2, lty=2, lwd=2)
segments(x0 = 4.5, y0 = 2, x1 = 6, y1=4)
```

legend:

```
legend(x="topright", legend = unique(iris$Species),
       col=mycol, pch = 16)
```

Adding stuff

Always in successive layers starting with a primary plot

```
plot(x = iris$Sepal.Length, y = iris$Sepal.Width,
     col=mycol[iris$Species], pch=16)
```

lines:

```
abline(lm(Sepal.Width~Sepal.Length, data=iris), lwd=5)
lines(x = 5:8, y=5:8/2, lty=2, lwd=2)
segments(x0 = 4.5, y0 = 2, x1 = 6, y1=4)
```

legend:

```
legend(x="topright", legend = unique(iris$Species),
       col=mycol, pch = 16)
```

text

```
text(x = 5, y=4, labels = "whatever in the plot")
```

Other graphical functions

Barplot

```
barplot(Freq ~ Class + Survived, data = as.data.frame(Titanic),
        subset = Age == "Adult" & Sex == "Male",
        ylab = "# passengers", legend = TRUE, beside=TRUE)
```

Other graphical functions

Barplot

```
barplot(Freq ~ Class + Survived, data = as.data.frame(Titanic),
       subset = Age == "Adult" & Sex == "Male",
       ylab = "# passengers", legend = TRUE, beside=TRUE)
```

Boxplot

```
boxplot(Freq ~ Class + Survived,data = as.data.frame(Titanic))
```

Other graphical functions

Barplot

```
barplot(Freq ~ Class + Survived, data = as.data.frame(Titanic),
       subset = Age == "Adult" & Sex == "Male",
       ylab = "# passengers", legend = TRUE, beside=TRUE)
```

Boxplot

```
boxplot(Freq ~ Class + Survived,data = as.data.frame(Titanic))
```

Histogram

```
a <- rnorm(1000); hist(a)
```

Other graphical functions

Barplot

```
barplot(Freq ~ Class + Survived, data = as.data.frame(Titanic),
        subset = Age == "Adult" & Sex == "Male",
        ylab = "# passengers", legend = TRUE, beside=TRUE)
```

Boxplot

```
boxplot(Freq ~ Class + Survived,data = as.data.frame(Titanic))
```

Histogram

```
a <- rnorm(1000); hist(a)
```

Matrix image

```
data("volcano")
image(volcano)
```

Panel layout

Simple option: mfrow

```
par(mfrow=c(2,3))
plot(1,1)
hist(rnorm(20))
image(diag(10))
plot(density(rnorm(100)))
curve(expr = sin(1/x), from = -1, to = 1, n = 10^4)
```

Panel layout

Simple option: mfrow

```
par(mfrow=c(2,3))
plot(1,1)
hist(rnorm(20))
image(diag(10))
plot(density(rnorm(100)))
curve(expr = sin(1/x), from = -1, to = 1, n = 10^4)
```

More flexible: layout

```
layout(matrix(c(1,2,3,2), 2, 2, byrow = TRUE),
       widths = c(1,2) , heights = c(2,3))
layout.show(3) # visualize the layout
plot(1,1)
hist(rnorm(20))
image(diag(10))
```

Should you use base plot?

Pros

- You control every pixel
- Simple graphes take little code, very easy
- Stable, few dependencies

Should you use base plot?

Pros

- You control every pixel
- Simple graphes take little code, very easy
- Stable, few dependencies

Cons

- Few black-box ready solutions
- Non-standard graphes take lots of code and knowledge of par()

Should you use base plot?

Pros

- You control every pixel
- Simple graphes take little code, very easy
- Stable, few dependencies

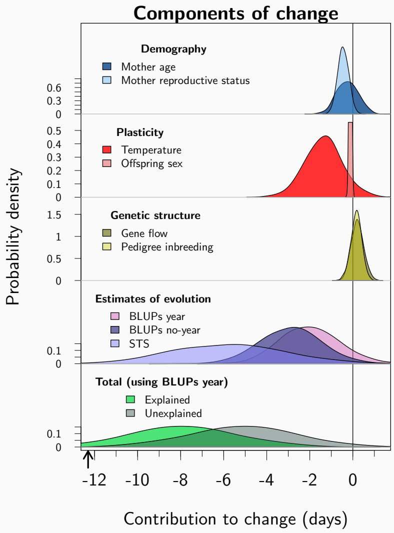
Cons

- Few black-box ready solutions
- Non-standard graphes take lots of code and knowledge of par()

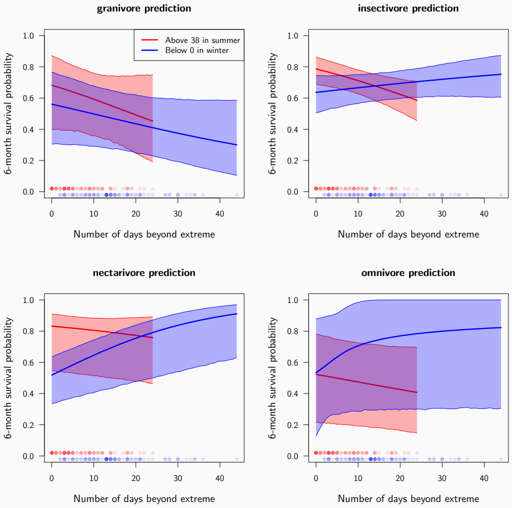
Pro/Con It depends who you ask

- Works in successive layers NOT interacting
- Base-R logic

Personally I like it...



Personally I like it...



Should you use ggplot?

Pros

- Complex graphes with little code
- Lots of pre-made extensions in packages
- Modular, interactive layers

Should you use ggplot?

Pros

- Complex graphes with little code
- Lots of pre-made extensions in packages
- Modular, interactive layers

Cons

- Logic different from R
- Unexpected behaviour and breaking dependencies
- Fine control more difficult

how ggplot works

```
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width))
```

how ggplot works

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width))
```

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point()
```

how ggplot works

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width))
```

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point()
```

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point() + facet_wrap(~Species)
```

1. Layers interact according to grammatic rules; not on top of each other

how ggplot works

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point() + geom_smooth(method = "lm")
```

how ggplot works

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point() + geom_smooth(method = "lm")
```

```r
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width,
                      color= Species)) +
  geom_point() + geom_smooth(method = "lm")
```

2. Aesthetics, aes(), controls the behaviour of all layers (unless stated otherwise)

how ggplot works

```r
basicplot <- ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width))
  geom_point()
basicplot

(smoothline <- basicplot + theme_linedraw() + geom_smooth())
(polarline <- basicplot + geom_line() + coord_polar())
smoothline + polarline
```

how ggplot works

```r
basicplot <- ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width))
  geom_point()
basicplot

(smoothline <- basicplot + theme_linedraw() + geom_smooth())
(polarline <- basicplot + geom_line() + coord_polar())
smoothline + polarline
```

3. Intermediate plots can be saved and modified (not final plots)

Exercise

```r
download.file(url="https://ndownloader.figshare.com/files/2292169",
              destfile = "dat.csv")
dat<- read.csv("dat.csv", stringsAsFactors = FALSE)
```

1. Load these data, inspect them in R
2. (Create a subset with rodents only)
3. Create some graphics to describe the relationship between hind foot length and weight. Does that relationship differ among rodent genus? Try log-scales.

Exercise

```r
download.file(url="https://ndownloader.figshare.com/files/2292169",
              destfile = "dat.csv")
dat<- read.csv("dat.csv", stringsAsFactors = FALSE)
```

1. Load these data, inspect them in R
2. (Create a subset with rodents only)
3. Create some graphics to describe the relationship between hind foot length and weight. Does that relationship differ among rodent genus? Try log-scales.
4. How do captures vary through months? Across years? Did some genus become more frequent? (try geom_bar())

Exercise

```r
download.file(url="https://ndownloader.figshare.com/files/2292169",
              destfile = "dat.csv")
dat<- read.csv("dat.csv", stringsAsFactors = FALSE)
```

1. Load these data, inspect them in R
2. (Create a subset with rodents only)
3. Create some graphics to describe the relationship between hind foot length and weight. Does that relationship differ among rodent genus? Try log-scales.
4. How do captures vary through months? Across years? Did some genus become more frequent? (try geom_bar())
5. Was the change in abundance of the genus Chaetodipus similar in all plot types?
6. Imagine more questions you could ask graphically!
7. (Change the color theme, axes labels... make your figure publishable!)

You CAN do all your graphics in R

Maps (example with ggplot)

```r
library("sf")
library("ggplot2")
library("rnaturalearth")
library("rnaturalearthdata")
world <- ne_countries(scale = "medium", returnclass = "sf")

ggplot(data = world) +
    geom_sf()
```

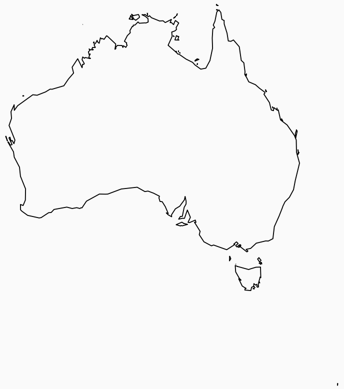
Maps (example with ggplot)

```r
coordcrs <- paste("+proj=laea",
                   "+lat_0=52",
                    "+lon_0=10",
                  "+x_0=4321000",
                  "+y_0=3210000",
                  "+ellps=GRS80",
                  "+units=m +no_defs")

ggplot(data = world) +
    geom_sf(aes(fill = pop_est)) + theme_bw()+
    coord_sf(crs =coordcrs)+
  scale_fill_viridis_c(option = "plasma", trans = "sqrt")
```

Maps (example with R base)

```
library(maps) map("world", regions = "australia")
```



Phylogeny example

```r
library(MCMCtreeR)
data(MCMCtree.output)
attach(MCMCtree.output)
names(MCMCtree.output)

phy <- readMCMCtree(MCMCtree.phy, from.file = FALSE)

MCMC.tree.plot(phy, analysis.type = "MCMCtree",
    cex.tips = 0.2, time.correction = 100,
    plot.type = "phylogram", lwd.bar = 2,
    scale.res = c("Eon", "Period"),
    node.method = "bar", col.age = "navy",
    no.margin = TRUE, label.offset = 4)
```

3D plot

```r
library("plot3D")
x1 <- rnorm(100) ; x2 <- rnorm(100) + 0.5*x1
y <- -0.5*x1 + 0.5*x2 ; fit <- lm(y ~ x1 + x2)
grid.lines = 26
x1.pred <- seq(min(x1), max(x1), length.out = grid.lines)
x2.pred <- seq(min(x2), max(x2), length.out = grid.lines)
x1x2 <- expand.grid( x1 = x1.pred, x2 = x2.pred)
y.pred <- matrix(predict(fit, newdata = x1x2),
nrow = grid.lines, ncol = grid.lines)
fitpoints <- predict(fit)
scatter3D(x1, x2, y, pch = 18, cex = 2,
theta =18, phi = -18, ticktype = "detailed",
xlab = "x1", ylab = "x2", zlab = "y",
surf = list(x = x1.pred, y = x2.pred, z = y.pred,
facets = NA, fit = fitpoints), main = "")
```

3D plot

(NB: you need RGL on your laptop for the following to work, that's not part of R)

```r
library("plot3Drgl")
plotrgl(lighting = FALSE, new=TRUE)
```

Want more?

https://www.r-graph-gallery.com/

Extra ggplot

Create your own theme

```r
?ggplot2::theme

custom_theme <- theme(
plot.title = element_text(color = "blue",
face = "bold", size=rel(2)),
panel.grid.major = element_line(colour="red",
linetype = "dashed"),
panel.background = element_rect(fill = "green")
)

basicplot + ggtitle("Your own theme") + custom_theme
```

Extra ggplot

Easy layout

```r
# install.packages("devtools")
devtools::install_github("thomasp85/patchwork")

library(patchwork)

p1 <- ggplot(mtcars) + geom_point(aes(mpg, disp))
p2 <- ggplot(mtcars) + geom_boxplot(aes(gear, disp, group = gear))

p1 + p2

p3 <- ggplot(mtcars) + geom_smooth(aes(disp, qsec))
p4 <- ggplot(mtcars) + geom_bar(aes(carb))

(p1 | p2 | p3) /
      p4
```

Extra ggplot

Adding marginal histograms

```
library("ggExtra")
df <- data.frame(x = rnorm(1000, 50, 10),
                 y = rnorm(1000, 50, 10),
    age = sample(c("a","b"), size = 1000, replace = TRUE))

p <- ggplot(df, aes(x, y, color=age)) + geom_point() +
  theme_classic()+
  theme(legend.position="left")+
  scale_color_brewer(palette = "Spectral", labels=c("a","b"))

ggExtra::ggMarginal(p, type = "histogram",
                groupFill=TRUE, groupColour = FALSE)
```

Please fill in survey

https://anu.au1.qualtrics.com/jfe/form/SV_23HWGb8pGSfyHxr