Base Graphics (part 3)

Graphics

Gaston Sanchez

R Coding Compendium

Donate

About

This is the third part on the traditional system for creating graphics in R.

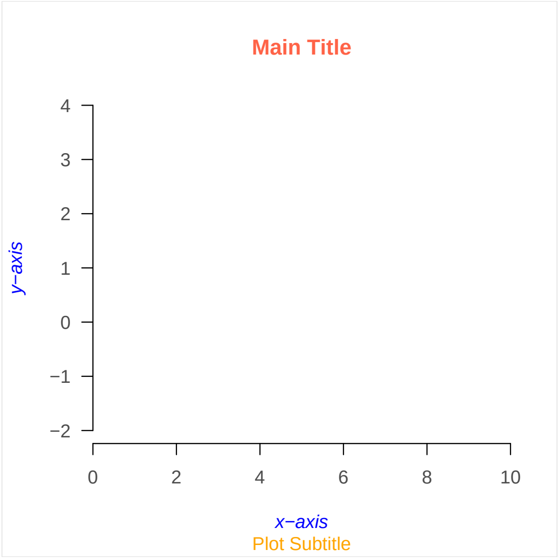
Plots from scratch

Customizing Annotations

It is also possible to create a plot from scratch. Although this procedure is less documented, it is extremely flexible and powerful:

- call `plot.new()` to start a new plot frame
- call `plot.window()` to define coordinates
- then call low-level functions:
- typical options involve `axis()`
- then `title()` (title, subtitle)
- after that call other function: e.g. `points()`, `lines()`, etc

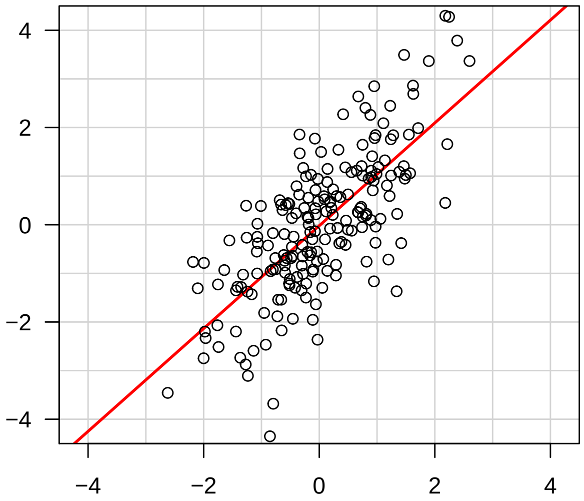
```r
plot.new()
plot.window(xlim = c(0, 10), ylim = c(-2, 4), xaxs = "i")
axis(side = 1, col.axis = "grey30")
axis(side = 2, col.axis = "grey30", las = 1)
title(main = "Main Title",
      col.main = "tomato",
      sub = "Plot Subtitle",
      col.sub = "orange",
      xlab = "x-axis",
      ylab = "y-axis",
      col.lab = "blue",
      font.lab = 3)
box("figure", col = "grey90")
```


Main Title
4
3
2
1
0
−1
−2
y−axis
0
2
4
6
8
10
x−axis
Plot Subtitle

```r
set.seed(5)
x <- rnorm(200)
y <- x + rnorm(200)

plot.new()
plot.window(xlim = c(-4.5, 4.5), xaxs = "i",
            ylim = c(-4.5, 4.5), yaxs = "i")
z <- lm(y ~ x)
abline(h = -4:4, v = -4:4, col = "lightgrey")
abline(a = coef(z)[1], b = coef(z)[2], lwd = 2, col = "red")
points(x, y)
axis(side = 1)
axis(side = 2, las = 1)
box()
title(main = "A Fitted Regression Line")
```

A Fitted Regression Line



Creating a Plot from Scratch

- Start a new plot with `plot.new()`
- `plot.new()` opens a new (empty) plot frame
- `plot.new()` chooses a default plotting region

Setting Up Coordinates

After starting with `plot.new()`, use `plot.window()` to set up the coordinate system for the plotting frame

```
# axis limits (0,1)x(0,1)
plot.window(xlim = c(0, 1), ylim = c(0, 1))
```

By default `plot.window()` produces axis limits which are expanded by 6% over those actually specified.

The default limits expansion can be turned-off by specifying `xaxs = "i"` and/or `yaxs = "i"`

```
plot.window(xlim, ylim, xaxs = "i")
```

Aspect Ratio Control

Another important argument is `asp`, which allows us to specify the **aspect ratio**

```
plot.window(xlim, ylim, xaxs = "i", asp = 1)
```

`asp = 1` means that unit steps in the `x` and `y` directions produce equal distances in the `x` and `y` directions on the plot.

(Important for avoiding distortion of circles that look like ellipses)

Drawing Axes

The ‘axis() function can be used to draw axes at any of the four sides of a plot.

- `side = 1` below the graph
- `side = 2` to the left of the graph
- `side = 3` above the graph
- `side = 4` to the right of the graph

Customizing Axes

Axes can be customized via several arguments (see `?axis`)

- location of tick-marks
- labels of axis
- colors
- sizes
- text fonts
- text orientation

Plot Annotation

The function `title()` allows us to include labels in the margins

- `main` main title above the graph
- `sub` subtitle below the graph
- `xlab` label for the x-axis
- `ylab` label for the y-axis

Customizing Annotations

The annotations can be customized with additional arguments for the fonts, colors, and size (expansion)

- `font.main`, `col.main`, `cex.main`
- `font.sub`, `col.sub`, `cex.sub`
- `font.lab`, `col.lab`, `cex.lab`

Drawing Arrows

Arrows can be drawn with the function:

```
arrows(x0, y0, x1, y1, code = int,
       length = num, angle = num)
```

- The `x0, y0, x1, y1` arguments give the start and end coordinates.
- `code=1` head at the start, `code=2` head at the end, `code=3` head at both ends
- `length` of the arrow head and `angle` to the shaft

Drawing Arrows

```r
plot.new()
plot.window(xlim = c(0, 1), ylim = c(0, 1))
arrows(0.05, 0.075, 0.45, 0.9, code = 1)
arrows(0.55, 0.9, 0.95, 0.075, code = 2)
arrows(0.1, 0, 0.9, 0, code = 3)
text(0.5, 1, "A", cex = 1.5)
text(0, 0, "B", cex = 1.5)
text(1, 0, "C", cex = 1.5)
```

$$\begin{tikzcd}
 & A \arrow[dl] \arrow[dr] & \\
B \arrow[rr, leftrightarrow] & & C
\end{tikzcd}$$

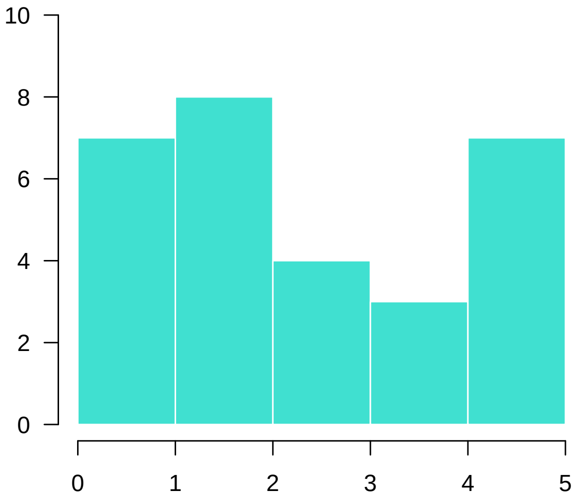
Drawing Rectangles

Rectangles can be drawn with the function:

```
rect(x0, y0, x1, y1, col = str, border = str)
```

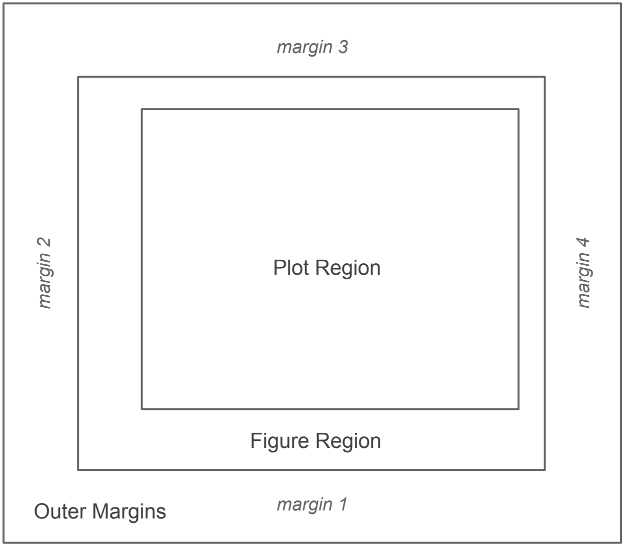
- `x0, y0, x1, y1` give the coordinates of diagonally opposite corners of the rectangles/
- `col` specifies the color of the interior.
- `border` specifies the color of the border/

```r
# barplot "manually" constructed
plot.new()
plot.window(xlim = c(0, 5), ylim = c(0, 10))
rect(0:4, 0, 1:5, c(7, 8, 4, 3),
     col = "turquoise",
     border = "white")
axis(1)
axis(2, las = 1)
```



10
8
6
4
2
0
0
1
2
3
4
5

Plotting Regions

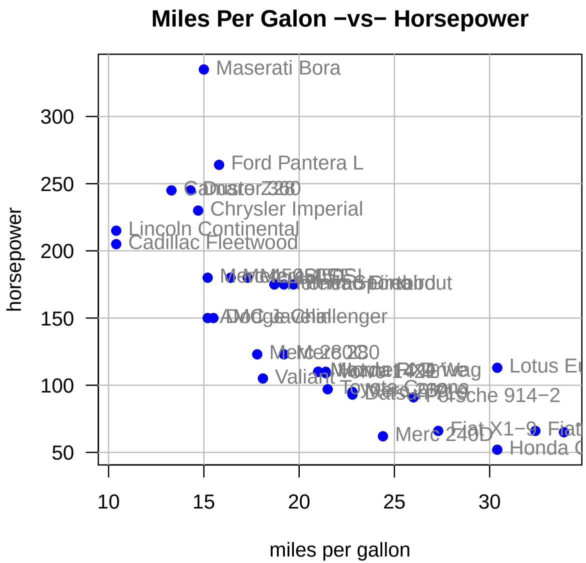

margin 3
margin 2
Plot Region
margin 4
Figure Region
Outer Margins
margin 1

Adjusting the Margins

Margins can be adjusted with the `par()` function in various ways:

- In inches: `par(mai = c(2, 2, 1, 1))`
- In lines of text: `par(mar = c(4, 4, 2, 2))`
- Width and Height in inches: `par(pin = c(5, 4))`

```r
# simple scatter-plot
op <- par(mar = c(5, 4, 3, 1))
plot(mtcars$mpg, mtcars$hp, type = "n", las = 1,
     xlab = "miles per gallon", ylab = "horsepower")
# grid lines
abline(v = seq(from = 10, to = 30, by = 5), col = 'gray')
abline(h = seq(from = 50, to = 300, by = 50), col = ' gray')
# points
points(mtcars$mpg, mtcars$hp, pch = 19, col = "blue")
# text (point labels)
text(mtcars$mpg, mtcars$hp, labels = rownames(mtcars),
     pos = 4, col = "gray50")
# title
title("Miles Per Galon -vs- Horsepower")
# reset graphical margins
par(op)
```



Miles Per Galon −vs− Horsepower
horsepower
miles per gallon
300
250
200
150
100
50
10
15
20
25
30
Maserati Bora
Ford Pantera L
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Mazda RX4 Wag
Lotus Eu
Valiant
Porsche 914−2
Fiat X1−9
Merc 240D
Honda C