Intro R plotting

Roberto Molowny-Horas

April 24-25, 2023

Standard R includes quite a few functions to plot data. Although they are not as sophisticated as those offered by some purpose-built R packages (e.g. **ggplot2**, later this week), they are a handy resource that often needs very little preparation for “quick and dirty” illustration of modeling results. Let’s see some examples.

Scatter plots

Scatter plots are helpful at showing trends in observational data and as pre-visualization tool before more complex statistical analysis is carried out.

```
x <- 1:19
y <- 5 + .3*x + rnorm(length(x))
color <- sample(c("blue", "red", "green"), length(x), replace = T)
type <- c("Type A", "Type B", "Type C")
plot(x, y,
     xlim = c(0, 20), ylim = c(0, 15), cex.axis = 1.5,
     xlab = "X axis", ylab = "Y axis", cex.lab = 1.5,
     main = "Scatter plot example", cex.main = 2,
     col = color, pch = 16, cex = 2)
```

```
# An equivalent way.
plot(y ~ x,
     xlim = c(0, 20), ylim = c(0, 15), cex.axis = 1.5,
     xlab = "X axis", ylab = "Y axis", cex.lab = 1.5,
     main = "Scatter plot example", cex.main = 2,
     col = color, pch = 16, cex = 2)
```

```
# Yet another way.
df <- data.frame(X = x, Y = y, Color = color)
attach(df)
plot(X, Y,
     xlim = c(0, 20), ylim = c(0, 15), cex.axis = 1.5,
     xlab = "X axis", ylab = "Y axis", cex.lab = 1.5,
     main = "Scatter plot example", cex.main = 2,
     col = Color, pch = 16, cex = 2)
detach(df)

# And yet another.
with(df,
     plot(X, Y,
     xlim = c(0,20), ylim = c(0, 15), cex.axis = 1.5,
     xlab = "X axis", ylab = "Y axis", cex.lab = 1.5,
     main = "Scatter plot example", cex.main = 2,
     col = Color, pch = 16, cex = 2))

# Finally, a nice legend.
legend("bottomright",
       legend = type,
       col = c("blue", "red", "green"),
       pch = 16,
       cex = 1.5)
```

We have introduced some new functionalities of R, like **sample**, **attach/detach** or **with**. They may or may not be useful depending on the circumstances. They allow you to write in R in a more compact way, though you would then pay the price of having a code that is somehow more cluttered and harder to debug. Have a look at the **ggplot2** package for a far more complete, sophisticated and, probably, more intuitive way of plotting.

Boxplots

Boxplots are useful for categorical data.

```
df$Type <- sample(type, length(df$X), replace = T)

boxplot(Y ~ Type, data = df,
        xlab = "Color times Type", ylab = "Y values", cex.lab = 1.5,
        main = "Boxplot example", cex.main = 2,
        varwidth = T,
        boxwex = .5)
```

```
# A more complicated example.
boxplot(Y ~ Color + Type, data = df,
        xlab = "Color times Type", ylab = "Y values", cex.lab = 1.5,
        main = "Boxplot example", cex.main = 2,
        varwidth = T,
        boxwex = .5)
```

Histograms

To show the distribution of a set of numbers, histograms are very useful.

```
z1 <- runif(400, min = -1, max = 2)
z2 <- rnorm(1000, mean = 2)
z <- c(z1, z2)
par(mfrow = c(2, 2))
hist(z2, breaks = seq(-10, 10, by = .1), xlim=c(-2,6),
     main = "Normal component", xlab = "X values", col = "red")
hist(z1, breaks = seq(-10, 10, by = .1), xlim = c(-2, 6),
     main = "Uniform component", xlab = "X values",col = "blue")
hist(z2, breaks = seq(-10, 10, by = .1), xlim = c(-2, 6),
     main = "Normal component", xlab = "X values", col = "red")
hist(z1, breaks = seq(-10, 10, by = .1), xlim = c(-2, 6),
     main = "Uniform component", xlab = "X values", col = "blue", add = T)
hist(z, breaks = seq(-10, 10, by = .1), xlim = c(-2 ,6),
     main = "All data", xlab = "X values")
```

```
par(mfcol = c(1, 1))   # To go back to individual plots.
```

---

*Exercise. Plot four histograms using **z1 <- runif(1000)**, **z2 <- runif(1000)**, **z3 <- runif(1000)** and **z4 <- (z1+z+z3)/3**. Should we change the **breaks** attribute?*