R Bootcamp
Homework 2: Working with data objects

Marguerite A. Butler

February 2, 2018

DUE: Monday February 5 by 5pm (but please don't wait til the last minute!), turned in by posting to GitHub or by email to mbutler808@gmail.com. Please put "710 Homework 2" in the subject line.

Practice

Vectors and data frames

```
> x <- c( 1, 5, 7)
> x
```

```
[1] 1 5 7
```

```
> rep( x, times=2)
```

```
[1] 1 5 7 1 5 7
```

```
> rep( x, each=2)
```

```
[1] 1 1 5 5 7 7
```

```
> y <- rnorm(6)
> y
```

```
[1] -0.45039371  2.57843737  0.16857918  0.53228122  0.09552438 -0.27390916
```

```
> xxy <- data.frame(x,x,y)
> xxy
```

```
  x x.1           y
1 1   1 -0.45039371
2 5   5  2.57843737
3 7   7  0.16857918
4 1   1  0.53228122
5 5   5  0.09552438
6 7   7 -0.27390916
```

```
> xxy[-2]    ## What does the negative index do?
```

```
  x           y
1 1 -0.45039371
2 5  2.57843737
3 7  0.16857918
4 1  0.53228122
5 5  0.09552438
6 7 -0.27390916
```

Vectors and matrices

```
> x <- 1:10
> x
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
> class(x)
```

```
[1] "integer"
```

```
> dim(x)
```

```
NULL
```

```
> dim(x) <- c(2,5)   # what just happened?
> x
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
```

We just made a matrix from a vector by manipulating the attributes (here, dimension). Another way to accomplish the same thing:

```
> x <- 1:10
> class(x)

[1] "integer"

> x <- matrix(x, nrow=2)
> class(x)

[1] "matrix"

> dim(x)

[1] 2 5
```

Instructions

Write a script of R commands (a text file with the ending .R) completing the following exercises. Be sure to answer all questions in a comment (# The attributes of x are...). Be sure to include comments to explain the code so that anyone reading it can understand what the code does, even if they are not in the class. Make sure your script runs without error when sourced before you turn it in.

Problems

1. Matrix reshaping and indexing. You may want to review the help page for matrix, as well as chapter 5&6 Creating data objects and What is it? For question 1, indicate each part a-i in your script with comments (#1a, #1b, etc.).

   (a) Create a matrix with the values 1 through 20, filling four rows. Save it as "x".

   (b) What are the attributes of x? Put the answer in the code as a comment.

   (c) Change it to a matrix with 2 rows and 10 columns by changing its attribute.

(d) Change x to a vector.

(e) Change x to a matrix with four rows, this time filling it by rows rather than by columns (you may want to check the help page).

(f) Coerce x to a vector again. Is it in the same order as the previous vector? What does this tell you about R's default behavior when flattening matrices to vector?

(g) Create the original x matrix again. Select only the 3rd row, 4th column. What is it?

(h) Select rows 3 and 4, columns 4 and 5. Print it to the console by using the `print(x)` function.

(i) Select the first and last rows, first and last columns. Print it.

2. **Exploring data and plotting** For this problem, use the built-in dataset `iris`. We will do a typical preliminary data exploration where we check each variable for normality and do bivariate plots two variables at a time. You may want to check the names of the `iris` dataframe to do the exercises. For this problem, what you should include in the final script is explained in 2(d).

   (a) Let's start by checking normality of the data. We will use the function `qqnorm()`. It produces a normal QQ plot (a probability plot) and compares it to the theoretical QQ for normally-distributed data. If the data are normally distributed, this plot should show points following a straight line at a diagonal. Since there are four variables, let's make four plots on one page. Set this up in the plot environment using the code:

   ```
   > par(mfrow=c(2,2))
   ```

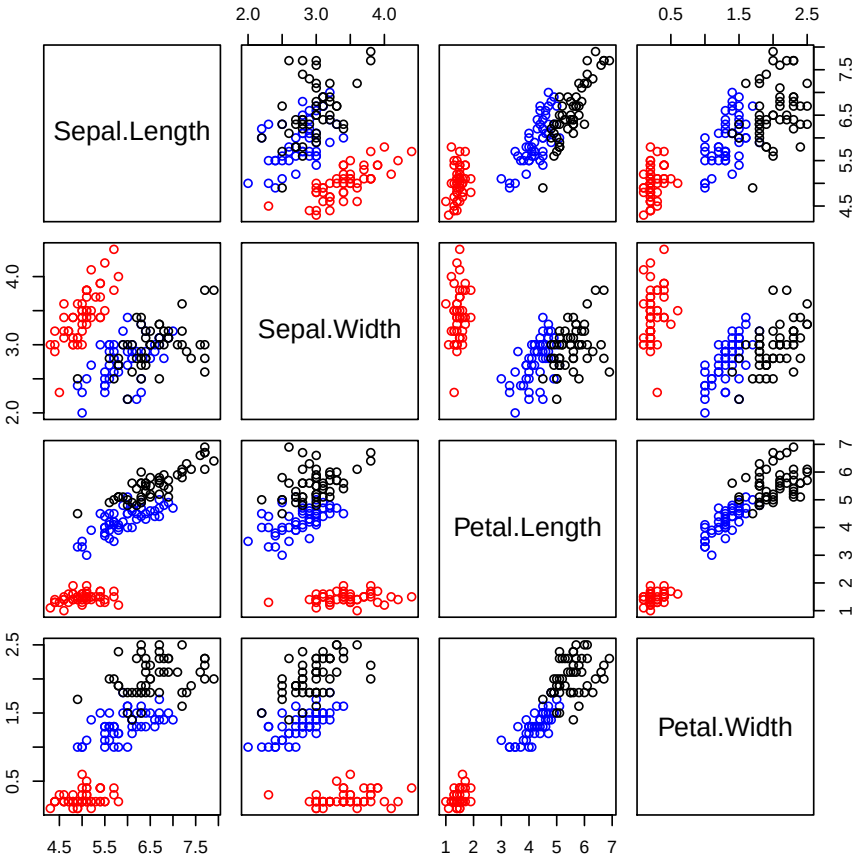
   Now make the four qqnorm plots (one for each variable), and they will fill the page by row first. Let's also add a title to each panel so we know what's what. Like this:

   ```
   > qqnorm( iris$Petal.Width, main="Petal Width")
   ```

   Make all four plots. Which variables are most normally distributed? Which are deviating the most? (We're just going to note it, but not do anything about it).

   (b) Make an new plot window using `quartz()` for Mac or `x11()` for PC or Linux. Now let's make some scatterplots. The plot method for data frames will automatically make a set of bivariate plots for each pair of variables (i.e., just use the `plot( iris )` function on the data frame). Plot the dataframe including only the numeric columns (exclude the species name column). Do you notice anything about the scatterplots? Do they correspond in any way to the dip seen in the QQ plots of some of the variables?

(c) Do you think there might be species differences in the data? Remake the scatterplots, but this time let's color code the species so that we can see the clouds of points for each species on each plot. We do this by creating a color vector, with one entry for each row of the `iris` dataframe, where setosa gets a value of `"red"`, versicolor gets a value of `"blue"`, and virginica gets a value of `"black"`. First find out how the species are arranged in the levels of the factor by printing the vector of `Species` to screen (or check the attributes). Create a vector of colors to correspond to the species names. Set the `col=` argument in `plot()` to the color vector that you just created. Your plot should look like this:



What do you see? Are the species different? In what way? Which are most different?

(d) Remake the QQ plots and the bivariate scatter with color, but this time make pdfs. Turn in your finished script that creates these plots as pdfs and answers the questions in comments.

CONGRATULATIONS! You just finished your second homework assignment.