Meeting 2

Anh Le

September 12, 2014

1 Readings

- Sec 1.3 Data frames and matrices. These are two most frequently used data structures in R, so understand it well.
- Sec 1.5 Graphics in R. Sec. 1.5.1 -¿ 1.5.3, 1.5.5

  I'm in a bit of dilemma about how to teach you guys graphing in R.

  There are three ways / package to do that in R: base R vs lattice vs ggplot2. See the difference in this StackOverflow answer – notice how there's no Cons for ggplot2 :-).

  The `gg` in `ggplot2` stands for "Grammar of Graphics". `ggplot2` is by far the best in the business of graphing. When other languages (like Python) wants to improve its graphing toolkit, they all copy from R's `ggplot2`.

  So keep that in mind as a very strong recommendation that you learn `ggplot2` instead of base R graphics.

2 In-class discussion

2.1 Data frame vs matrix

Data frame allows us to hold multiple types of data (e.g. numeric, string, factors), whereas a matrix only contains one type of data. This means that data frame is useful for interactive data analysis (since real life data is messy), while matrix is good for programming (where the program we write should not be surprised by a data type it can't handle).

Let's prove the stated difference between data frame (df) and matrix (mat) to ourselves:

```
df <- data.frame(col1 = 1:4, col2 = c("one", "two", "three", "four"),
                 stringsAsFactors=FALSE)
mat <- matrix(c(1:4, "one", "two", "three", "four"), ncol = 2)

df
```

```
##   col1  col2
## 1    1   one
## 2    2   two
## 3    3 three
## 4    4  four

str(df) # str() stands for structure -- a useful function to understand a data object

## 'data.frame': 4 obs. of  2 variables:
##  $ col1: int  1 2 3 4
##  $ col2: chr  "one" "two" "three" "four"

mat

##      [,1] [,2]
## [1,] "1"  "one"
## [2,] "2"  "two"
## [3,] "3"  "three"
## [4,] "4"  "four"

str(mat)

##  chr [1:4, 1:2] "1" "2" "3" "4" "one" "two" "three" ...
```

Another useful practice I used above is to use `stringsAsFactors=FALSE` whenever you import data into R's data frames. Without specifying that option, the default is `stringsAsFactors=TRUE`, meaning that R will automatically convert your strings into factors (aka categorical variables) and guess the levels.

For example, if you data has a column of country names, i.g. ”USA”, ”UK”, ”France”, R will try to convert this into a factor (categorical variable). However, without understanding the data many times it will guess wrong, so the best practice is to set `stringsAsFactors=FALSE` always.

3 In-class Exercises

See our website.