Stats 598z: Homework 1

Due before midnight on Friday, Jan 26

Important:`R` code, tables and figures should be part of a single .pdf or .html files from `R Markdown` and `knitr`. See the class reading lists for a short tutorial.

1 Problem 1: The `seq()` function [30]

1. We saw the '`:`' operator generate integer vectors. A more flexible function in `seq()`. Skim through the `R` documentation to try and understand what this does. In particular, with examples explain the `from, to, by` and `length.out` options. [8pts]
2. Note that you don't always have to specify all options, some missing values take default values. Explain what `seq(10)` and `seq(3,10)` do. [4pts]
3. What happens when you pass `seq()` a vector (of anything)? [4pts]
4. What happens when you pass `seq()` an integer vector of length 1? Note that this behaviour is inconsistent with the previous question and is often a source of bugs. [3pts]
5. The `seq_along()` function avoids this inconsistency. Show this by using this with vector inputs from the previous two questions. [4pts]
6. `seq_len(num)` is shorthand[1] for `seq(from = 1, to = num, by = 1)`. Use `seq_len` (and some basic arithmetic) to generate an *integer*-vector from 3 to 10. [4pts]
7. `seq_len(num)` resembles `1:num` unless num is negative. What happens then for both cases? [3pts]

2 Problem 2: Vectors in `R` [25]

`R` comes with a few built-in constant to make life easy for you. One such constant is `letters`.

1. Print out `letters`. Also, print a summary of it. [3pts]
2. What kind of data type is `letters`? Include the commands you used to find this. [3pts]
3. What is the length of `letters`? Don't count! [2pts]
4. How would you access every alternate element of `letters`? Save the first, third, fifth etc. in an appropriately named variable. [4pts]
5. How would you store `letters` backwards in an appropriately named variable? [3pts]
6. How would you store every alternate letters backwards in an appropriately named variable? [4pts]
7. Save the first 16 letters in a 4×4 matrix. [3pts]
8. From the documentation of `letters`, mention two other built-in constants and their data types. [3pts]

[1] not exactly: it's actually a more efficient implementation

3 Problem 3: Matrices in R [45]

`runif()` generates numbers uniformly between 0 and 1 (we saw `rnorm()` in class).

1. Generate 20 numbers uniformly between 0 and 1 and store them in a 4×5 matrix. [3pts]
2. How would you index the entire third row of this matrix? What kind of data-structure is this? What are its dimensions (if any)? [2pts]
3. Repeat the previous with `drop=FALSE`. See `help('[')` for this option. [2pts]
4. `sum()` calculates the sum of all elements in the `R` object. What is the sum of your matrix? [2pts]
5. How would you transform your matrix so that the sum of all elements is 1? [3pts]
6. `rowSums()` and `colSums()` have self-explanatory names (though you should skim the documentation). Print their outputs for your matrix. [3pts]
7. How would you transform your matrix so that its rows add up to one? Use recycling (and recall `R` matrices are column-major). You should not use `for` loops, just material from the lectures [7pts]
8. What does the function `t()` do? [2pts]
9. How would you transform your matrix so that its columns add up to one? Your answer will be similar to the previous one, except you will additionally have to use `t()`. [7pts]
10. Use `runif()` to generate a new 2×9 matrix. Starting with this matrix, generate a new 3×3 matrix by picking every other (i.e. alternate) element. There are two possibilities: every other element as you move along the rows, and every other element as you move along the columns. Provide code and results for both cases. Again, you might have to use `t()`. [7pts]
11. Look at the documentation for the `cbind()` function. The example at the end has four lines:

    ```
    m <- cbind(1, 1:7) # the '1' (= shorter vector) is recycled
    m
    m <- cbind(m, 8:14)[, c(1, 3, 2)] # insert a column
    m
    ```

    Run these and print the output. Explain what is happening. Line 3 is a sequence of operations, and it might help to break them down into the individual operations. [7pts]