Assignment 02

DSA2101 AY 22/23 Sem I

Material Covered

This assignment covers topics 1 and 2 from the lecture notes, and tutorials 1 to 4. The main learning outcomes are:

1. To be familiar with base R programming and plotting.
2. To know how to create named "sections" in an Rmd document.
3. To be able to identify errors in others' code.
4. To know how S3 object-oriented programming works in R.
5. To practice with web-scraping in R.
6. To practice with file input and output in R.

Question 1 (5 marks)

The following algorithm (Box-Muller) generates a pair of i.i.d random variables from a $N(0,1)$ distribution.

1. Generate $U_1, U_2 \sim Unif(0,1)$.
2. Define:

$$X_1 = \sqrt{-2\log(U_1)}\cos(2\pi U_2) \quad (1)$$
$$X_2 = \sqrt{-2\log(U_1)}\sin(2\pi U_2) \quad (2)$$

3. Take X_1 and X_2 as i.i.d draws from $N(0,1)$.

Tasks

1. Write an R function `box_muller` with a single argument `n`, that returns `n` random variables from $N(0,1)$ using the above algorithm.
2. Use your function to generate 1000 random variables, and create a histogram.
3. In an Rmd section entitled "Question 1.3", state *two* ways in which you will test your output to verify that the algorithm is working correctly.

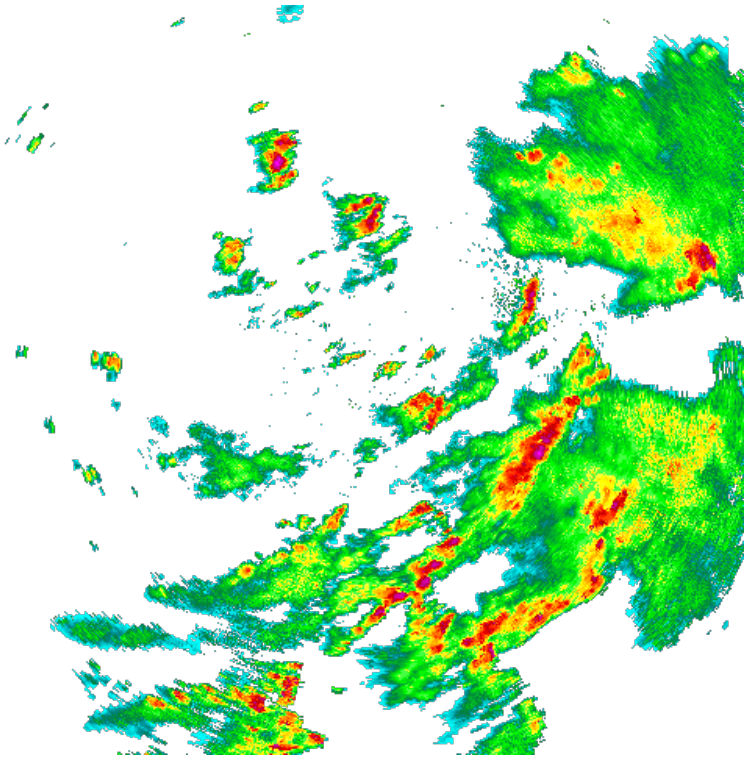
Notes

1. You may use ggplot or base R plots. But ensure that your plot has a title.
2. We will run your function, and test the output to see if it is close to Normal.
3. Remember to handle odd values of `n`.
4. For question 1.3, I am just looking for heuristic methods, not rigorously proved methods. When we implement a procedure, we should be able to test our function to see if it works properly. You can describe graphical or numerical procedures, from your knowledge in ST2131. You should include what you expect to see if the function works/ does not work. Please keep your answer to about 200 words max.

Question 2 (5 marks)

The following website from NEA displays radar scans around Singapore. Download the 240km radar scans between 2022-07-31 01:00:00 (am) and 2022-07-31 03:00:00 (am) at five-minute intervals, inclusive of end points. Save the images to a zip file named `radar_files.zip` in your `data` folder.

Here is an example of the images you would obtain:



Notes

1. The zip file structure should not have any directories within it. When I unzip it, all the files should appear in my `data` folder.
2. Make sure that only one zip file remains after your code is done; any interim files should be deleted.
3. You may need to look up the help pages for `zip` and `unlink`. If your operating system is Windows, you may have to install Rtools in order to create zip files from within R.

Question 3 (5 marks)

In tutorial 3, we read in the grammatical facial expressions data, and wrote a plot method for objects of class `gfe`. When we deal with non-tabular or unconventional datasets, we usually have to think of a data structure for them, and write methods to explore, summarise and manipulate them within R. The `[` method to extract subsets is one such method. Write an extract method `[` for objects of class `gfe` that works this way:

```
gfe_obj <- read.fe_data("../data/grammatical_facial_expression",
                        "b", "emphasis")
# original structure
str(gfe_obj)
```

```
## List of 2
##  $ frames: num [1:100, 1:3, 1:1344] 329 327 323 320 317 ...
##  $ info  :'data.frame':  1344 obs. of  4 variables:
```

```
##   ..$ timestamp        : num [1:1344] 1.39e+09 1.39e+09 1.39e+09 1.39e+09 1.39e+09 ...
##   ..$ user             : chr [1:1344] "b" "b" "b" "b" ...
##   ..$ facial_expression: chr [1:1344] "emphasis" "emphasis" "emphasis" "emphasis" ...
##   ..$ fe_present       : num [1:1344] 0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "class")= chr "gfe"
```

```
sub_gfe_obj <- gfe_obj[1:3]
# new structure
str(sub_gfe_obj)
```

```
## List of 2
##  $ frames: num [1:100, 1:3, 1:3] 329 327 323 320 317 ...
##  $ info  :'data.frame':  3 obs. of  4 variables:
##   ..$ timestamp        : num [1:3] 1.39e+09 1.39e+09 1.39e+09
##   ..$ user             : chr [1:3] "b" "b" "b"
##   ..$ facial_expression: chr [1:3] "emphasis" "emphasis" "emphasis"
##   ..$ fe_present       : num [1:3] 0 0 0
##  - attr(*, "class")= chr "gfe"
```

Notes

1. To test your code, we will run it with different types of index vectors, and inspect the structure of the resulting object.
2. Please make sure that you handle recycling, logical indexing and negative indexing in your function.

Question 4 (3 marks)

The file `est_poly_fns.rdt` contains three functions that were meant to be answers for question 2 of assignment 1. The function was supposed to return a numeric matrix of `n` interpolated points, given certain inputs. Find the mistakes in each of them, and describe your findings in a section entitled "Question 4".

To load the functions into your workspace, you can do:

```
load("../data/est_poly_fns.rdt")
```

Notes

There is no need to fix the functions. We will not be running them.

Requirements

1. Your code in your file should create the following objects when we run the code:
   - a function named `box_muller`
   - an S3 function `[` that works on objects of class `gfe`.
2. The knitted html file should contain:
   - a histogram
   - two markdown text sections entitled "Question 1.3" and "Question 4" (not comments inside code chunks).

Reminders

We have a big class, but we strive to provide individual feedback. Please adhere to the following practices; marks will be deducted if you do not adhere to them.

1. Use relative paths for reading and writing data. Your Rmd file should be in `src` folder, and your data should be in `data` folder, at the same level.
2. Your Rmd code should **not** contain calls to `setwd`, `install.packages`, `View`, or `edit`. The latter two are meant for interactive use.
3. To test if we will be able to knit your code, clear out your environment and then knit the file. This ensures that your code does not use stray objects in your global workspace.
4. *The assignment is individual work.* If we find a suspiciously high similarity between individuals, we will not hesitate to award 0 marks.
5. Late submissions will not be graded. Email submissions will not be graded. Only submissions through the Canvas assignment portal will be graded.
6. It is ok to submit multiple solution files, but we will only grade the most recent one. You do not have to delete the previous ones.
7. Avoid the use of `for` loops in this tutorial.