MA 354: Data Analysis I – Fall 2021
Homework 0:

Complete the following opportunities to use what we've talked about in class. These questions will be graded for correctness, communication and succinctness. Ensure you show your work and explain your logic in a legible and refined submission.

0. **Complete weekly diagnostics.**

1. Simple tasks.

   (a) Print the first 25 even integers using the `seq()` function.
   Solution:

   ```
   seq(2,50,2)

   ##  [1]  2  4  6  8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50
   ```

   (b) Print the square root of the first 50 odd natural numbers

   (c) Evaluate how many numbers between 1 and 100 are divisible by 3.

   (d) Evaluate $y = exp(1) - \left(1 + \frac{1}{x}\right)^x$ over a sequence from 10 to 200 (by 10). Where does it appear that y is approaching as x increases? For a visual representation check `?plot`.

   (e) Write a function that takes a vector of coordinates in $\mathbb{R}^2$ and converts them to polar. For (x, y) we want

   $$r = \sqrt{x^2 + y^2}$$
   $$\theta = \tan^{-1}\left(\frac{y}{x}\right).$$

   Make sure to return $\theta \in (0, 2\pi)$. Note that if (x, y) is in the second, or third quadrant we must add π to `R`'s result for θ, and if (x, y) is in the fourth quadrant we must add 2π.
   Hint: Running `?sin` will provide documentation on trigonometric functions.

   (f) Add an optional argument to your function for part (e) that allows the user to specify that they want degrees instead of radians. Note that radians should be the default.

   (g) Demonstrate your function for points $(-3, 10), (5, -8), (13, 13), (-2, -10)$, and $(0, -8)$ in both radians and degrees.

2. Consider a tug-of-war competition for robots. In each matchup, two robots take turns tugging the rope until the marker indicates that one of the robots won.

   - The match starts with the marker at 0.
   - Robot A pulls the rope – use `runif(n=1,min=0,max=0.50)` to simulate the magnitude of the pull. Adding the simulated value to the marker position gives the new position of the marker.
   - Robot B pulls the rope in the opposite direction – use `runif(n=1,min=-0.50,max=0)` to simulate the magnitude of the pull. Adding the simulated value to the marker position gives the new position of the marker.
   - The two robots continue taking turns until the marker moves past -0.50 or 0.50.

   (a) Write a function that simulates a tug-of-war match. When called, it should simulate a tug-of-war contest, and return "Robot A" when robot A wins, and "Robot B" when robot B wins.

   (b) Report the results of 10,000 simulated robot tug of war battles. Is the game fair?

   (c) What can be done to make this game more fair? Be creative!

3. In class, we wrote the following function for a project Euler problem exploring divisibility streaks.

```
streak <- function(n){
  k <- 0 # starting k
  repeat{
    remainder <- (n + k) %% (k+1) # is n+k divisible by k+1

    if(remainder == 0){ #is the remainder 0?
      k<-k+1
    }else{ #the remainder isn't 0 --> streak ended
      break
    }
  }
  k #return(k)
}
```

(a) Why does an issue occur when we plug in $n = 1$? Note that there is no error message – the worst!

(b) Fix the issue in part (a).

(c) Define a function $P(s, N)$ to be the number of integers n, $1 < n < N$, for which streak(n) $= s$. Complete this using a loop.

(d) Redo part (c) using an apply-function-based approach.

(e) Write code to find $\sum_{i=1}^{31} P(i, 4^i)$. Note that the apply function will not work because would have to create a vector of size 10^{31}, which would make us run out of memory (unless you have a *wild* computer). The approach using the loop approach can work, but it could take days or weeks to finish (depending on your processor). Report the code to find the solution, but make sure not to evaulate it. Instead, you can test this function by evaluating $\sum_{i=1}^{5} P(i, 4^i)$ noting that this function could hypothetically be used to evaluate the original sum with enough time or computational power.

(f) The standard approach to solving part (e) is rather inefficient. Let's write a more a efficient function. Note this would still take days to complete the calculation from part (e). Instead, you can test this function by evaluating $\sum_{i=1}^{5} P(i, 4^i)$. Write a function that does the following:

   i. takes $I \in \mathbb{N}$ as input

   ii. creates an $I \times I$ matrix – think of this as a table where rows are values of i (1:I), columns are values streaks (1:I), cell entries are counts of observations)

   iii. for each value of interest $i = 1, 2, ..., I$

      A. calculates the streaks for the values on $\left[4^{i-1} + 1, 4^i\right]$

      B. updates the i^{th} row of the matrix

   iv. At the end, use the matrix to evaluate the sum.
   Hint: You may find the functions `upper.tri()`, `lower.tri()`, or `diag()` helpful.

4. Recently, I worked with Dr. Brown-Iannuzzi (UVA) to evaluate how people's mental visualizations of "welfare recipients" and "illegal voters" shape their policy attitudes. To do this, we have been using reverse correlation techniques (Dotsch et al., 2008; Mangini and Biederman, 2004; Schmitz et al., 2021). Within these tasks, participants are asked to choose, across many different trials, which of two images looks more like a "welfare recipient," for example.

   The images presented to participants on each trial are generated by superimposing random visual noise over a merged photo representing people of different races and genders. Across many trials, we can then average participants' image selections to get an average image composite of whom people envision when they think of different social categories. For example, Figure 1 depicts the average image participants envisioned when asked to consider a "social security recipient" and "not a social security recipient." Critically, we have further linked characteristics of these photos (e.g., skin tone) to participants' degree of belief that social security recipients are hardworking.

   In the chunk of code below, I load the images as pixel matrices using the imager package (Barthelme, 2021).

```r
library(imager) # install.packages("imager") to install
img.base <- as.matrix(load.image("base.jpg"))
# image of NOT SS recipient
img.young <- as.matrix(load.image("notSSrecip.jpg")) # Noisy
# image of SS recipient
img.old <- as.matrix(load.image("SSrecip.jpg"))  # Noisy
```

   The plot is created as follows. Note that I use the ggmap package (Kahle and Wickham, 2013) to plot the pixel matrix, and the patchwork package (Pedersen, 2020) to combine the graphs.

```r
library(ggmap)
library(patchwork)
g1<-ggimage(img.base, fullpage = FALSE) +
  ggtitle("Base Image")+
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
        axis.text.y=element_blank(),axis.ticks=element_blank(),
        axis.title.x=element_blank(),
        axis.title.y=element_blank(),legend.position="none")
g2<-ggimage(img.old, fullpage = FALSE) +
  ggtitle("Social Security Recipient")+
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
        axis.text.y=element_blank(),axis.ticks=element_blank(),
        axis.title.x=element_blank(),
        axis.title.y=element_blank(),legend.position="none")
g3<-ggimage(img.young, fullpage = FALSE) +
  ggtitle("Non Social Security Recipient")+
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
        axis.text.y=element_blank(),axis.ticks=element_blank(),
        axis.title.x=element_blank(),
        axis.title.y=element_blank(),legend.position="none")

g1+g2+g3
```

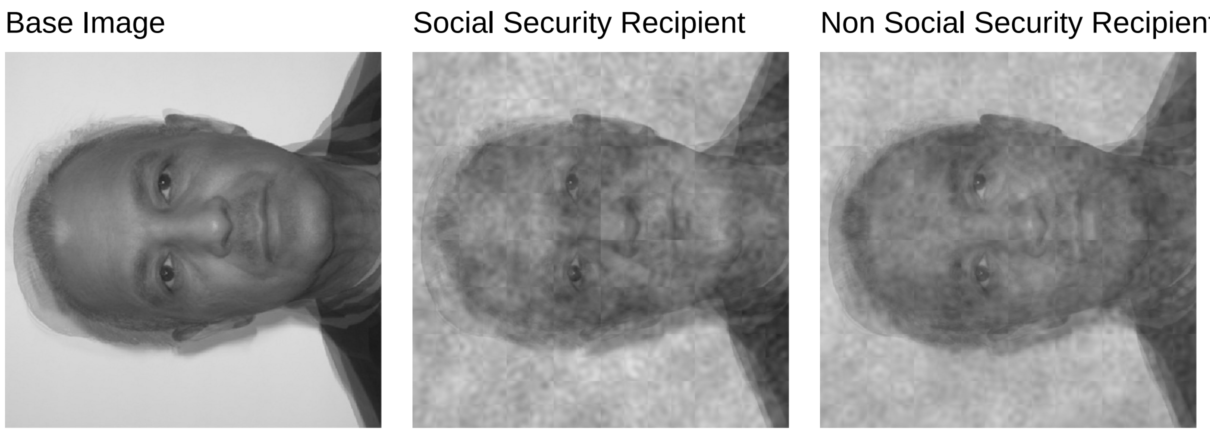

Figure 1: The Base image (left), and the resulting visual representations of a social security recipient (middle) and non social security recipient (right).

(a) Fix Figure 1 by manipulating the matrix so the pictures are right-side-up.
Hint: There is a much easier way than writing your own function to rotate a matrix. Use a built in function.

(b) Subtract the base image from the social security recipient image and the non social security recipient image to get an idea about what the added noise looks like. What do you notice about the noise?

(c) **Optional:** We'll discuss plotting in detail in Chapter 5, but as a warm up and a nod to our first reading (De Paolis, 2019), attempt creating a plot that shows the difference between the two noisy images. This might help elucidate the visual differences in the images generated when selecting Social Security recipients compared to non Social Security recipients. Complete a quick search on Google Scholar to see if you can't find some work that corroborates what you find.

References

Barthelme, S. (2021). *imager: Image Processing Library Based on 'CImg'*. R package version 0.42.10.

De Paolis, D. (2019). Am i an expert developer or just an expert googler? [Online; accessed 28-August-2021].

Dotsch, R., Wigboldus, D. H., Langner, O., and van Knippenberg, A. (2008). Ethnic out-group faces are biased in the prejudiced mind. *Psychological science*, 19(10):978–980.

Kahle, D. and Wickham, H. (2013). ggmap: Spatial visualization with ggplot2. *The R Journal*, 5(1):144–161.

Mangini, M. C. and Biederman, I. (2004). Making the ineffable explicit: Estimating the information employed for face classifications. *Cognitive Science*, 28(2):209–226.

Pedersen, T. L. (2020). *patchwork: The Composer of Plots*. R package version 1.1.1.

Schmitz, M., Rougier, M., and Yzerbyt, V. (2021). Introducing the brief reverse correlation.