PS941: Practical Report 2

AUTHOR
Lukasz Walasek

PUBLISHED
March 7, 2025

For your second PS941 practical report, you will need to complete two activities. These are equally weighted, and you must complete both to achieve the maximum score. Please read the following instructions carefully. Each challenge requires a written response and annotated R code (**Important: Please make sure to submit RAW R Code with your report**). You can choose one of the following formats but please remember to always include raw R code in uncompiled .Rmd or .qmd file, or plain R script.

R Markdown: Submit a single '.Rmd' file, along with a compiled document in either a .pdf or .html format.

LaTeX: Submit both the compiled PDF and the '.tex' source file(s). I recommend using Overleaf.

Quarto: Submit a '.qmd' file (similar to your seminar materials) containing your responses and code snippets. You can even use the source file for these instructions as a template.

Word Document: Submit your report as a Word document. In addition, submit a separate file containing all the raw 'R' code.

Whichever option you choose, I recommend that you compress all files into a .zip format and submit your work as one file to avoid issues with Tabula.

Please note that there is no strict word limit for this report, but you should aim for approximately 500-1000 words per challenge. The R code does not count towards this total. In other words, what matters is that your code works, not that how conciesly you can write it.

Challenge 1 - Network Analysis

Your task is to create your own social network and then visualize and evaluate it along several metrics. To do this, you need to follow these steps:

1. Write down a list of people you know personally, as quickly as you can, between 15 and 20 people. These represent the nodes in your network.

2. Next, produce a weighted adjacency matrix indicating how well these people know one another (on a scale from 0 to 4) — this is data you can input based on your personal knowledge of the people and their relationships. You can use any software you like for this, but an excel sheet is straightforward. Make sure you attach these data with your submission so that the marker can reproduce your analysis.

3. For every possible threshold between 0 and 4, visualize the unweighted and undirected network, and so that node labels are clear and legible. You may need to try different layouts and you may need to relabel your nodes (e.g., with initials).

4. Choose what you see as the most informative unweighted network above, and produce a table of centrality measures for each of the nodes: degree, betweenness, eigenvector centrality, closeness, and clustering coefficient. Briefly explain what each of these measures tells us about your social network.

5. Provide evidence that this network is or is not assortative by degree. Explain what your result means.

6. For all the thresholds you used for keeping edges in your network, show which of these thresholds produces a network that is least like an Erdös-Renyi random graph in terms of clustering coefficient. To do this, generate 200 Erdös-Renyi random graphs of the same size and density as your network for each threshold and compare the distribution of clustering coefficients with the observed value for your social network at that threshold. You can use a one-sample t-test for each comparison. Report the analysis and visualize the winning thresholded network alongside an example ER random graph. ER random graphs are explained in detail in chapter 3 of your textbook.

Note: If you are interested in learning more about how people report their social networks, see Hills, T. T., & Pachur, T. (2012). Dynamic search and working memory in social recall. Journal of Experimental Psychology: Learning, Memory, and Cognition, 38(1), 218.

Challenge 2 - Agent Based Modeling

Your second task is to propose and test two extensions of Granovetter's threshold model. Your starting point is the simple implementation of the model, which is described below. In this implementation, agents are located in a square grid and each one is assigned some threshold value. On each iteration, a randomly picked agent inspects eight of its immediate neighbours to see if they are active (1) or passive (0). If the proportion of active agents is higher than the agent's threshold value, the agent becomes active too.

Your task is as follows:

1. Think about the type of social process that you would like to model with this framework. You can choose anything you like, using the model to simulate the rising number of people who choose to riot, get married, adopt some technology, or just express some attitude. While it is completely up to you what you choose, in your answer you should explain how the model captures the key dynamics of your chosen social process.

2. You need to extend the model in two ways.

a. First, you are asked to explore how the distribution of thresholds in the population may influence the model's behaviour. Using simulations, show how the model behaves as you change the initial setup of thresholds of individual agents.

b. Second, you should extend the core behaviour of your agents. The goal here is to make your simulation more realistic in how it reflects basic psychological and social processes of individual agents. You are free to change any aspect of the model, and you can draw inspiration from the activities that we will discuss in our seminars.

3. You need to showcase, through simulation, how the two extensions (above) of the model influence its behaviour. Crucially, you should discuss how your results help us understand the psychological dynamics of your chosen social process. You should complement your answer with visualization that summarize the behaviour of your model.

Note: We will be enhancing this model during our final seminar session, and you are welcome to use the improved version for this assignment. You can also create your own independent implementation of the Granovetter's model. None of this is required however, and it will not impact your grade.

Granovetter Model

```
library(reshape2)

# Setup the number of cycles in the simulation.
cycle <- seq(1:10000)

# Define the grid size
grid_size <- 32
# Populate your grid with threshold values drawn at random from a beta distribution with parameters shape1
thresholds <- matrix(rbeta(n = grid_size^2, shape1 = 5, shape2 = 5),
nrow = grid_size, ncol = grid_size)
# Create a matrix of agents being active or not.
activity <- matrix(0, nrow = grid_size, ncol = grid_size)
# Randomly allocate some agents to be active from the start. This can be useful to get the simulation goir
activity[sample(1:(grid_size^2), 100)] <- 1

# Function to get the sum of thresholds of immediate neighbours.
get_neighbour_sum <- function(x, y, grid) {
  neighbours <- c(grid[x-1, y], grid[x+1, y], grid[x, y-1], grid[x, y+1],
  grid[x-1,y-1], grid[x+1,y+1],grid[x-1,y+1], grid[x+1,y-1])
```

```r
  return(sum(neighbours, na.rm = TRUE))
}

# Simulation
r <- vector()
for (i in 1:length(cycle)) {
  agent_x <- sample(2:(grid_size-1), 1)
  agent_y <- sample(2:(grid_size-1), 1)

  neighbour_sum <- get_neighbour_sum(agent_x, agent_y, activity)

  # Change agent to active if his threshold is below the proportion of surrounding active agents.
  if (thresholds[agent_x, agent_y] < (neighbour_sum / 8)) {
    activity[agent_x, agent_y] <- 1
  }

  r[i] <- sum(activity)
  # Optional methods to inspect behaviour of the simulation.
  #Sys.sleep(0.05)
  #print(neighbour_sum)
}

# Plot the results
plot(r, type = "l", main = "Number of Active Agents Over Time", xlab = "Time", ylab = "Number of Active Ag
```