School Nova Computer Science 202

Homework 10 (due 12/12/2020)

We are going to a create an alternative SIR model based on random matching (instead of a grid). Importantly, the model will be mostly based on NUMPY.

Task 1

There are N = 100 agents in the model. And for each agent we need to store data for 8 variables. In other words, create an N by 8 numpy array initially composed of all zeros. Use np.int32 as the dtype.

Task 2

The first variable (column 0) in the array will be a numerical ID, which starts at 0 and increases to N (excluded). Use np.arange for this task. (Note: at the end of the homework I will give you an example of how the final numpy array may look like).

Task 3

Column 1 (the second column) is the initial Susceptible status. All agents begin as 1. Do not use for loop to assign 1 to each agent (remember that numpy arrays allow element-wise assignments and math; see the classwork code).

Task 4

Column 2 is the Infected status: 0 if not Infected, 1 if infected. Randomly assign 1 to some agents. Assume that each agent's probability of becoming Infected is 0.1. Do not use loops. Use np.random.binomial(1, 0.1, N).

Task 5

Those agents who become Infected (=1) are no longer Susceptible. Change their Susceptible status to 0. Again: no loops.

Task 6

Column 3 (Recovered) values are zero. Column 4 is wearing a mask (0-no mask, 1-mask on). Assume that the probability of wearing a mask is 0.8. Generate the values. No loops (if you forgot how to do it; see Task 4).

Task 7

Column 5 is the number of days the person has been sick. Assume that it takes 7 days to recover. In other words, assign 7 in column 5 to all agents who are Infected. Guess what: no loops.

Task 8

Assume that every day, each agent visits one other agent. In other words, each agent has *two* meetings (one as a host, and one as a guest). Column 6 will represent the id of an agent who "you" visit (there will be an example below to clarify). Agents randomly choose whom to visit. Instead of a loop, you can use np.random.choice(N, N, replace = False). Note: given the formula, it is possible that an agent chooses to visit self – let's not worry about this possibility (assume that some agents isolate!).

Task 9
Given the information in column 6, update column 7 showing who visited you (your guest's ID). In other words, column 6 shows who you visit when you are a guest, while column 7 shows the id of the agent who visits you when you are a host. For this task, you will probably need to use a for loop since the numpy solution is not easy (but certainly possible!).

Your final numpy array should look something like the below (example for N = 10):

```
[[0 0 1 0 1 7 3 6]
 [1 1 0 0 1 0 8 9]
 [2 0 1 0 1 7 7 4]
 [3 1 0 0 1 0 5 0]
 [4 0 1 0 0 7 2 5]
 [5 1 0 0 1 0 4 3]
 [6 1 0 0 0 0 0 8]
 [7 1 0 0 1 0 9 2]
 [8 1 0 0 1 0 6 1]
 [9 1 0 0 1 0 1 7]]
```

To remind you, each row represents an agent. Columns: 0-ID, 1-Susceptible, 2-Infected, 3-Recovered, 4-Mask ON, 5-Days Sick, 6-id of an agent who you visit, 7-id of an agent who visits you. (Verify that 6 and 7 match; For example agent 0 visits agent 3: A[0, 6] = 3. We can verify that when 3 is the host, the guest is indeed agent 0: A[3, 7] = 0. And so on).

Task 10
Using np.sum calculate and display information about the total number of Susceptible and Infected agents. (Hint: you need to specify the appropriate column when using np.sum).

Task 11* (optional)
Think how you would implement a disease transmission based on the values from the numpy array. Feel free to use for loops for this (optional) task. Eventually, we will explore if any of the for loops can be avoided thanks to the numpy functionality.