Student Name: Mai Le

ID: i6375777

Email: [mai.le@student.maastrichtuniversity.nl](mailto:mai.le@student.maastrichtuniversity.nl)

**Matlab Assignment**

1. Explain the logic behind the code for the people matrix (5 marks)

This global function is built to simulate the virus infection of the population. To achieve it, the function needs inputs, which are the number of people in the population, the size of the environment, the radius of infection, the chance of infection per timestep, the length of infection, and the simulation length, and the output is the days needed to reach no infection in the population. Firstly, we need to assign particular values for inputs and create the people matrix which consists of six columns (columns 1 and 2 are the coordinates of people in the environment, column 3 is age, column 4 is gender, column 5 is the number of days left infected, and column 6 is immune status) and the row number equal to the number of people in the population. We assume that one person is infected at the start and then spreads the virus to the population. This infected person is identified randomly in the population, and her/his count of days left infected is assigned to the length of infection value. To know who has the potential to be infected due to interaction with the infected person, the distance between the first infected person and other people in the population will be calculated, while their possibilities of infection are generated randomly. If their physical distances are under the radius of infection (which means that they don't maintain social distance with infected one), and their possibilities of infection are higher than the threshold of being infected, called the chance of infection per timestep, they would be considered as newly infected people at the end of the current timestep. However, we also need to consider the immune status of these people to know who is exactly the newly infected people in the population because some infected people will recover and acquire immunity, or some healthy people were vaccinated and already have immunity, thus protecting them from getting infected again. Therefore, we need to consider the immune status of infected people to identify who are exactly the newly infected ones. They are people who will be infected and haven’t obtained the immune yet. After being infected, the number of days left infected is initially set as the length of infection. To start the simulation of the next timestep, the infected days of all infected people are subtracted to one day at the end of the current timestep, and their positions are changed randomly but still in the environment. If infected day values reach 1, they will become immune in the next simulation. These steps are repeated to simulate the population infection in the next timestep. When no one in the population is infected at a particular timestep, this is the day that no infection in the population is reached.

b) Does this virus simulation work in discrete or continuous time?  Explain your answer.  (5 marks)

This virus simulation works in discrete time because it only simulates the virus infection in a specific timestep—a particular day. It doesn’t provide information about population infection continuously during the day.

c) The most complex line in this program is line 92, which uses the function *repmat*.  Look up what this function does. Can you suggest alternative(s) to this line of code. (It can be multi-line) (10 marks)

The function *repmat(A,m,n)* returns a matrix containing m copies of A in the row dimension and n copies of A in the column dimension. The function *not(A)* returns a table of logical values of the same size as A, whereas the output contains logical 1 (true) values where A is zero and logical 0 (false) values where A is nonzero.

The logic behind the code of line 92: this code is used to find people who haven’t been infected yet in this timestep (their values in column 5 = 0) and haven’t acquired immunity yet (their values in column 6 = 0). The code uses *not* function to return the logical values of these people as 1 (true). The *repmat* function is used to give these matrices a new size so that they can be multiplied together. After multiplying entries of three matrices, entry 1 (true) returns people who have the potential for infection, haven’t been infected, and haven’t been immune before. They will be newly infected people.

The logic behind my code: I use the *find* function to find the indices of people who haven’t been infected yet (their infected days left = 0) and haven’t acquired immunity yet (their immune status = 0); these indices are stored in an array - I. After that, I create the zeros matrix, which is the same size as the potentiallyInfected2 matrix, called Alt. If the order of a column of the Alt matrix is similar to one of the values in array I, this column is then equal to the multiplication of the corresponding column from the potentiallyInfected2 matrix and 1 (because 1 = true value for people who haven’t been infected and haven’t got immune yet). As a result, entry 1 in the Alt matrix means people who are potentially infected, haven’t been infected, and haven’t obtained immunity before, they will be newly infected people at the end of this timestep.

**The codes are in file virus\_simulation\_2024\_assign.m from line 90 to line 97**

%% Question c - Alternative codes for code line 129

%Find the indices of people who haven't been infected and haven't acquired the

%immune at the beginning of the simulated timestep

I = find(and(people(:,5)==0,people(:,6)==0));

%Create a zeros matrix with the same size as the potentiallyInfected2 matrix

Alt = zeros(size(potentiallyInfected,1),numberPeople);

%Identify newly infected people in the Alt matrix (entry = 1)

Alt(:,I)= potentiallyInfected2(:,I)\*1;

d)  You have been given the demographics of the Netherlands. Load the data into the given code as a matrix or table. Find the sex distribution from the demographics. Add a column to the people matrix which denotes the sex of the person, distributed according to the demographics matrix.

Similarly, you can find the distribution of different age groups in the Netherlands. Add another column to the matrix which is the age of the person. Use the age distribution from the demographic matrix to randomly assign ages to the people. (15 marks)

 (*Tip: You can use built-in functions in MATLAB to perform weighted sampling*)

1. Load the data into the given code as a table and do the calculations

Because the given Excel file has both text and numbers, when I used the *readmatrix* function to import the Excel file to Matlab, Matlab could not read the first column as text; it returned the “***NaN”*** in all rows. Therefore, I changed my code to [num,txt,raw] = xlsread('D:\UM\MATLAB\Data\population') to import the Excel file. As a result, Matlab can read it both in text and numbers.

Moreover, I used the *cell2mat* function to change the data of the M table from cell format to matrix format so that I could do basic calculations with them. When I didn’t change the format, I got the error: Operator '+' is not supported for operands of type 'cell'.

**The codes are in file virus\_simulation\_2024\_assign.m from line 8 to line 28**

%% Now assign gender 0 = male, 1 = female to column 4 of the people matrix

%read the table from the Netherlands population as of 29th August 2022

[num,txt,raw] = xlsread('D:\UM\MATLAB\Data\population');

M0 = raw(2:21,1:3);

%Assign the name for each column of table M

M = array2table(M0,'VariableNames',{'age group','men','women'});

%Calculate the total number of people in each age group

M.total = cell2mat(M{:,2}) + cell2mat(M{:,3});

%Calculate the percentage of males in each age group

M.menpercent=cell2mat(M{:,2})./M.total\*100;

%Calculate the percentage of females in each age group

M.womenpercent = cell2mat(M{:,3})./M.total\*100;

%Calculate the percentage of males in the Netherlands population

MalePerc = mean(M.menpercent);

%Calculate the percentage of females in the Netherlands population

FemalePerc= mean(M.womenpercent);

%Thus, the gender distribution of the Netherlands population is

% 46.7342 % male : 53.2658 % female.

%Now, we need to assign this distribution randomly to column 4 of the

%people matrix with male = 0 and female = 1

1. Assign the gender of people in the simulated population based on the gender distribution of the Netherlands population to male = 0 and female = 1.

After analyzing the Netherlands population, I found that the percentage of males is 46.7342 % and the percentage of females is 53.2658 %. Next, I randomly assigned this distribution to column 4 of the people matrix, with male = 0 and female = 1.

The logic behind my codes: From the gender distribution, I first calculate the number of females in the simulated population, called the femalenumber variable. Then I assigned the first number of people (equal to femalenumber) to value 1 in column 4. The value of the remaining rows is 0 for males. To rearrange gender randomly, I use the *randperm* function to generate a row vector containing a random permutation of the integers from 1 to the number of people in the population, which serve as the indices of random gender values.

**The codes are in the file virus\_simulation\_2024\_assign.m from line 30 to line 36**

%Calculate the number of females in the simulated population

femalenumber = round((FemalePerc/100)\*numberPeople);

%Assign the gender of these people to 1

people(1:femalenumber,4) = 1;

%Rearrange the gender of the simulated population randomly

randomOrder = randperm(numberPeople);

people(:,4) = people(randomOrder,4);

1. Assign the age of people in the simulated population based on the age distribution of the Netherlands population.

After analyzing the Netherlands population, I got the percentage of each age group. Next, I randomly assigned this distribution to column 3 of the people matrix.

The logic behind my codes:

From the age distribution, I first calculate the number of people at each age group in the simulated population, called agenumber variable, using the *round* function. However, rounding numerous decimal numbers may lead to a loss in the total value. Thus, I need to check whether the sum of people from all age groups is not equal to the number of people in the population. If so, I have to adjust agenumber to the same value as numberPeople by adding the final value of agenumber withthe different value between numberPeople and the sum of agenumber. Next, I randomly assign age values to people in each age group based on the age distribution. My codes are inspired by the table of the number of people in each age group:

|  |  |  |  |
| --- | --- | --- | --- |
| Age group | Number of people | Index of the first person | Index of the last person |
| 0-4 | 49 | 1 | 49 |
| 5-9 | 52 | 49+1 | 49+52 |
| 10-14 | 55 | 49+52+1 | 49+52+55 |
| 15-19 | 60 | 49+52+55+1 | 49+52+55+60 |
| 20-24 | 63 | 49+52+55+60+1 | 49+52+55+60+63 |
| … |  |  |  |
| i | agenumber(i) | age\_cum(i-1)+1 | age\_cum(i) |

From the table:

In age group 0-4, the index of the first person is 1, and the index of the last person is similar to the number of people in this age group. From the age group 5-9 onwards, the index of the first person is equal to that of the last person of the previous age group plus 1. The index of the last person is equal to the cumulative sum of people from all previous age groups.

Thus, in a particular age group (i) which has the number of people equal to agenumber(i), the index of the first person is equal to that of the last person of the previous age group (i-1) plus 1. The index of the last person is equal to the cumulative sum of people from all previous age groups (from group 1 to group i)

**The codes are in the file virus\_simulation\_2024\_assign.m from line 38 to line 60**

%% Assign age to each of these people

%Calculate the percentage of each age group from the Netherlands population

M.agepercent = M.total./sum(M.total)\*100;

%Calculate the number of people in each age group of the people matrix

agenumber = round((M.agepercent/100)\*numberPeople);

%Check if the sum of people from all age groups is not equal to the total

%number of people due to rounding

sum(agenumber) == numberPeople;

%If no, we need to adjust the agenumber to the same value as numberPeople

agenumber(end) = agenumber(end) + (numberPeople-sum(agenumber));

%Create a zero array of age of all people in the simulated population

age\_array=zeros(numberPeople,1);

%Calculate the cumulative sum of people in each age group as the

%indices for age\_array

age\_cum=cumsum(agenumber);

%Randomly assign the age for people in age group 0-5

age\_array(1:age\_cum(1)) = randi([(1-1)\*5,(1\*5-1)],agenumber(1),1);

%Randomly assign the age for people from the age group 5-10 to 95 and above

for i=2:20

age\_array((age\_cum(i-1)+1):age\_cum(i))=randi([(i-1)\*5,(i\*5-1)],agenumber(i),1);

end

%Assign column 3 of the people matrix to the values of age\_array

people(:,3)=age\_array;

e) Column 5 records the length of infection of each person. One article suggests that people above the age of 60 have a longer length of infection. It also suggests that males over 60 have an even longer length of infection than males below 60. Include this in the model. Add 2 days to the length of infection if the person is above 60 years of age. Add another 2 days if the person is a male above 60 years. (25 marks)

The logic behind my codes: ind returns the indices of newly infected people. First, I find the index of (1) newly infected people who are male and over 60 years old, (2) newly infected people who are female and over 60 years old, and (3) newly infected people who are under or equal to 60 years old. After that, I assign the length of infection in column 5 of the people matrix to lengthOfInfection +4 days for newly infected people who are male and over 60 years old, to lengthOfInfection +2 days for newly infected people who are female and over 60 years old, to lengthOfInfection days for newly infected people who are under or equal to 60 years old based on their indices.

**The codes are in the file virus\_simulation\_2024\_assign.m from line 99 to line 109**

%% Assign length of infection of males over 60 to 7, and of females over 60 to 5, and others to 3

%Find the indices of newly infected people who are males over 60

maleover60 = ind(and(people(ind,3)>60,people(ind,4)==0));

%Find the indices of newly infected people who are females over 60

femaleover60 = ind(and(people(ind,3)>60,people(ind,4)==1));

%Find the indices of newly infected people who are under or equal to 60

upto60 = ind(people(ind,3)<=60);

%Assign the length of infection of newly infected people

people(maleover60,5)=lengthOfInfection+4;

people(femaleover60,5)=lengthOfInfection+2;

people(upto60,5)=lengthOfInfection;

***\*Note: I changed the simulationLength to 50 because the for loop cannot run if the simulationLength is 100.***

f) Right now in the simulation, only one person is infected in the beginning. Make a plot of the number of days needed to reach no infection versus the number of infected people at the start. (20 marks)

The logic behind my codes:

I use the *for* loop with an index equal to the number of infected people at the start, ranging from 1 to 30, to simulate the days needed to reach no infection in the population for different numbers of infected people at the start. For a particular number of infected people at the beginning, I identify these people exactly in the population by randomly generating their indices, from 1 to the number of people, by the *randperm* function. Their days of infection are then assigned to the lengthOfInfection value. Next, I need to simulate the virus spreading in the population when these people move around the environment and to identify the last day of infection. To achieve this, I use a *while* loop with the true statement that there are still existing infected people in the population (that is, the sum of days left infected of all people is non-zero). If so, the newly infected people are identified in the same way as the codes given by the teacher, and the days existing infection in the population is increased to 1 at the end of the simulation. The *while* loop is stopped when no one in the population is infected (that is, the sum of days left infected of all people is zero). The days that infection still exists in the population resulting from the final *while* loop is considered the final day of infection or the days needed to reach no infection in the population. To begin the new *for* loop with a different number of people infected at the start, I need to reset all the length of infection values in column 5 and all the immune statuses in column 6 to 0 because, at the start, no one is infected or immune until the initially infected people are assigned. After that, new *for* loops simulate the days needed to reach no infection in the population with different numbers of infected people at the beginning, ranging from 1 to 30. As a result, I obtain an array of 30 values of infection days corresponding to 30 numbers (from 1 to 30) of initially infected people.

Because I did not use the *for* loop to simulate the days needed to reach no infection in the population, the inputs of the global function don't include simulationLength.

**The codes are in the file virus\_simulation\_2024\_partf.m**

function infectionDays = virus\_simulation\_2024\_partf(numberPeople, sizeEnvironment, radiusOfInfection, chanceOfInfectionPerTimestep, lengthOfInfection)

numberPeople = 1000;

sizeEnvironment = 100;

lengthOfInfection = 3;

people = zeros(numberPeople, 6);

people(:,1:2)=ceil(rand(numberPeople,2)\*sizeEnvironment);

%% Assign gender 0 = male, 1 = female based on gender distribution of the Netherlands population

%Read table from Netherlands population as of 29th August 2022

[num,txt,raw] = xlsread('D:\UM\MATLAB\Data\population');

M0 = raw(2:21,1:3);

M = array2table(M0,'VariableNames',{'age group','men','women'});

M.total = cell2mat(M{:,2}) + cell2mat(M{:,3});

M.menpercent=cell2mat(M{:,2})./M.total\*100;

M.womenpercent = cell2mat(M{:,3})./M.total\*100;

MalePerc = mean(M.menpercent);

FemalePerc= mean(M.womenpercent);

femalenumber = round((FemalePerc/100)\*numberPeople);

people(1:femalenumber,4) = 1;

randomOrder = randperm(numberPeople);

people(:,4) = people(randomOrder,4);

%% Assign age to each of these people based on the age distribution of the Netherlands population

M.agepercent = M.total./sum(M.total)\*100;

agenumber = round((M.agepercent/100)\*numberPeople);

sum(agenumber) == numberPeople;

agenumber(end) = agenumber(end) + (numberPeople-sum(agenumber));

age\_array=zeros(numberPeople,1);

age\_cum=cumsum(agenumber);

age\_array(1:age\_cum(1)) = randi([(1-1)\*5,(1\*5-1)],agenumber(1),1);

for i=2:20

age\_array((age\_cum(i-1)+1):age\_cum(i))=randi([(i-1)\*5,(i\*5-1)],agenumber(i),1);

end

people(:,3)=age\_array;

%% Simulate the days needed to reach no infection for initially infected people ranging from 1 to 30

for n = 1:30

%Assign the number of initially infected people at each iteration

infectedNumber(n) = n;

%Assign randomly the indices of n initially infected people

infected(1:n,n) = randperm(numberPeople, infectedNumber(n));

%Assign the length of infection of initially infected people to

%lengthOfInfection

people(infected(1:n,n), 5) = lengthOfInfection;

%Set the radius of infection and change of infection per timestep

radiusOfInfection = 5;

chanceOfInfectionPerTimestep = 0.5;

%Next, we identify the final day of infection at which no one is infected in the population

%Set the first day of infection as 1

time=1;

%Run while loop to simulate the days needed to reach no infection in the population

while sum(people(:,5))~=0

%Identify newly infected people

distances=squareform(pdist(people(:,1:2)));

notSociallyDistanced=and(distances<radiusOfInfection, distances>0);

potentiallyInfected=notSociallyDistanced(people(:,5)>0,:);

if size(potentiallyInfected,1)>0

infectionRand = rand(size(potentiallyInfected,1), numberPeople);

potentiallyInfected2 = potentiallyInfected .\* (infectionRand>chanceOfInfectionPerTimestep);

notImmune = potentiallyInfected2.\*(not(repmat(people(:,5)>0,1,size(potentiallyInfected,1))))'.\*(repmat(not(people(:,6)),1,size(potentiallyInfected,1)))';

ind = find(sum(notImmune,1)>0);

end

%% Assign length of infection of males over 60 to 7, of females over 60 to 5, and others to 3

maleover60 = ind(and(people(ind,3)>60,people(ind,4)==0));

femaleover60 = ind(and(people(ind,3)>60,people(ind,4)==1));

upto60 = ind(people(ind,3)<=60);

people(maleover60,5)=lengthOfInfection+4;

people(femaleover60,5)=lengthOfInfection+2;

people(upto60,5)=lengthOfInfection;

%% Next we need to set up the new day

people(people(:,5)==1,6)=1;

people(people(:,5)>0,5)=people(people(:,5)>0,5)-1;

move=ceil(rand(numberPeople,2)\*3)-2;

people(:,1:2)=people(:,1:2)+move;

people(people(:,1)<1,1)=sizeEnvironment;

people(people(:,2)<1,2)=sizeEnvironment;

people(people(:,1)>sizeEnvironment,1)=1;

people(people(:,2)>sizeEnvironment,2)=1;

%time is increased by one unit until no one is infected anymore in the population

time = time+1;

end

infectionDays(n) = time;

%To start a new for loop with a new number of initially infected people, we need to

%reset all the length of infection values and all the immune status to 0 at the

%end of the current loop

people(:,5:6)=0;

end

%plot the days needed to reach no infection versus the number of innitially

%infected people in the population

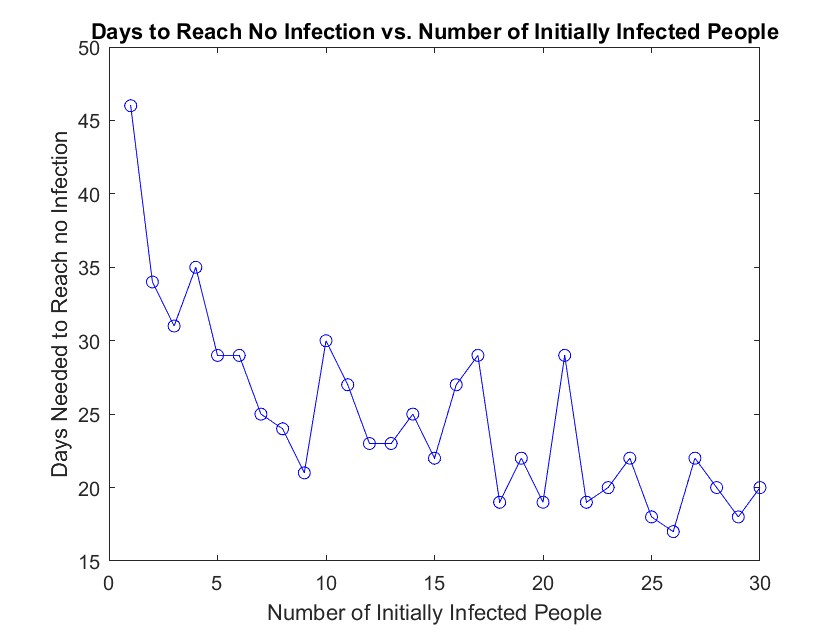
plot(infectionDays,'b-o');

xlabel('Number of Initially Infected People');

ylabel('Days Needed to Reach no Infection');

title('Days to Reach No Infection vs. Number of Initially Infected People');

end



**Figure**: The plot of the days needed to reach no infection in the population versus the number of infected people at the start.

g) The following script (next page), created by ChatGPT, is designed to be inserted at line 128 of “*virus\_simulation\_2024.m”* function to simulate the addition of a vaccination center. The aim is to randomly place a vaccination center that can vaccinate a certain number of people per day. This vaccination site can only vaccinate people who are within a distance of 8\*8 from the center of the vaccination site. Only vaccinated individuals are immune.

Your task is to critically evaluate this implementation. Do you agree with the approach taken, or do you find any issues with the script? Please describe any problems you identify and suggest how to fix them. (20 marks)

% location of the vaccination center

vaccinationCenter = ceil(rand(1,2) \* sizeEnvironment); => same

vaccinationRadius = 8; => same

% Calculate the distance of each person from the vaccination center

distancesToCenter = sqrt((people(:,1) - vaccinationCenter(1)).^2 + (people(:,2) - vaccinationCenter(2)).^2); same

% Identify people who are within the vaccination radius

eligibleForVaccination = distancesToCenter <= vaccinationRadius; => diff -> return logical results <= : true = 1,

% Randomly select people to vaccinate => it is not randomly select, it select exactly eligible people

eligibleIndices = find(eligibleForVaccination); => returns the indices of eligible for vaccination

vaccinatedIndices = eligibleIndices;

people(vaccinatedIndices, 6) = 1; => identify vaccination people

**Answer:**

I don't agree with ChatGPT's approach. I found two issues with the codes it gave to create a vaccination center randomly that can vaccinate a certain number of people per day.

The first one is that ChatGPT's codes do not consider vaccination capacity. ChatGPT’s codes indicate that all eligible people for vaccination can be vaccinated. However, in fact, the vaccination center can only vaccinate a maximum of a particular number of people per day, called vaccination capacity. Thus, I suggest adding the vaccination capacity to the code, and if the number of vaccinated people is equal to the vaccination capacity, the vaccination process will be stopped.

The second issue is that the codes from ChatGPT lack consideration for the infection and the immune status of eligible people for vaccination. From codes given by ChatGPT, we understand that all people whose distance to the vaccination center under the radius of vaccination will be vaccinated, regardless of the circumstance that they are being infected or they have already acquired immunity after infection. However, in fact, only people who have never been infected or haven’t gotten immune yet have the ability to be vaccinated. Because if they have recovered from the virus infection, they will have immunity, or if they have already acquired immunity, they don’t need to be vaccinated. After that, if these people are in an area where vaccination is possible, they will be vaccinated. Thus, I suggest finding people who have not being infected (that is, their length of infection = 0) and have not acquired immunity yet (that is, their immune status = 0). If their distance to the vaccination center is under the radius of vaccination, they will be vaccinated and subsequently get immune, and their immune status will be assigned to 1. After that, the number of vaccinated people will be increased by one person until it reaches the vaccination capacity.

After running, the function will return the days needed to reach no infection in the population after vaccination.

**The codes are in the file virus\_simulation\_2024\_partg.m from line 75 to line 103**

%% Now, we simulate the vaccination process

%Randomly assign the position of vaccination center

vaccinationCenter = ceil(rand(1,2) \* sizeEnvironment);

vaccinationRadius = 8;

%Set vaccination capacity

vaccinationCapacity = 50;

%Set the first value of the number of people who are vaccinated at the start

vaccinationNumber = 0;

%Find people who haven't been infected and haven't obtained immunity yet as

%potentially vaccinated people

potentialVac = find(and(people(:,5)==0,people(:,6)==0));

%Simulate the vaccination process

for l = 1: length(potentialVac)

%If the number of vaccinated people reaches the vaccination capacity, the

%vaccination will be stopped

if vaccinationNumber == vaccinationCapacity

break;

end

%Calculate the distance between the potentially vaccinated people and

%vaccination center

distancesToCenter(l) = sqrt((people(potentialVac(l),1) - vaccinationCenter(1)).^2 + (people(potentialVac(l),2) - vaccinationCenter(2)).^2);

%If the distance between a potentially vaccinated person and vaccination

%center is under 8, this person will be vaccinated and get immune

if distancesToCenter(l) < vaccinationRadius

people(potentialVac(l),6)=1;

%The number of vaccinated people is increased by 1

vaccinationNumber=vaccinationNumber+1;

end

end

**Reflection section**

While doing this assignment, I encountered many errors that prevented me from running the codes further. To address these problems, I almost used the website MathWork.com (<https://nl.mathworks.com/> ) to search for similar questions someone had asked before and found the solution through their discussion. This assignment provided me with the real-world experience of running code for a project where I could make many mistakes, and the most important is that I can repair them on my own to complete my task.

Through this assignment, I have learned numerous new functions. For example, the *remap* and *not* functions are used in line 92 to reshape the size of a matrix and find its logical matrix, respectively. Another new function I learned is the *randperm* function, which is used to randomly generate an array's indices. Moreover, I have had the chance to practice more with the *find* function and *for* loop to simulate a biological event, so now I am able to familiarize myself with them.

The most challenging question I have is how to assign the age randomly to people in the population based on the age distribution of the Netherlands population. My codes looked very complicated, and I still cannot find the simpler one. Additionally, by tackling the final question, I know that the code given by AI is not always true and comprehensive. This may be due to the lack of input information about the question; therefore, the code made by AI is very general. I warned myself to be careful when writing my code using AI. Although it can assist me in finding out the functions to solve tasks I am struggling with, AI still cannot generate the whole code for complicated work.