Stochastic Process

Chen Li

5/10/2020

(1) Stochastic simulations of population growth

Matlab code stochasticBirth.m to simulate cell population growth, with a birth rate $\lambda = 0.4$. Starting with a step size $\Delta t = 0.001$ and an initial population size of $N(0) = 1$, modify the code to run 10,000 stochastic simulations (realizations). Calculate the distribution of population sizes (i.e., the number of realizations with $N = 1$, $N = 2$, ... individuals) at time $t = 5$.

(a) the folowing is a histgram of distribution of population from 10000 trails at t=5, and the red curve is Pn(5)

Code for (a) and (b)

```
trials = 10000; %number of realizations
N = zeros(tsteps,trials); %population matrix
N(1,:) = N0; %initialize population

for j=1:trials
    for i=1:tsteps-1
        pop = N(i,j); %get current pop.
        randVec = rand(pop,1); %vector of N rand #s on (0,1)
        cellBorn = length(find(randVec<lambda*dt)); %number of cells that divided
        N(i+1,j) = N(i,j) + cellBorn; %recurrence eq.
    end
end

%maximum and minimum populations attained from simulations
minPop = min(N(end,:));
maxPop = max(N(end,:));

endCount = (minPop-0.5):(maxPop+0.5); %vector of possible outcomes (for histogram)

P = zeros(maxPop,1); %initalize probability mass function P_n(t)

%Specify P_n(t). You need to multiply P_n(t) by 'trials' in order to compare
%it with the histogram.
for n=1:maxPop
    P(n) = exp(-lambda*5)*(1-exp(-lambda*5))^(n-1);
end

%figures (don't forget to add axis labels!)
figure(1)
```

```
plot(tvec, N(:,1:5), 'LineWidth', 2) %time course for five trials

%you don't need a legend for this figure
figure(2)
histogram(N(end,:), endCount) %histogram of final cell counts
hold on
plot(1:maxPop, P * 10000, 'LineWidth', 3) %probability mass function

figure(3)
plot(tvec, mean(N'), 'LineWidth', 2) %plot average population size versus time
```

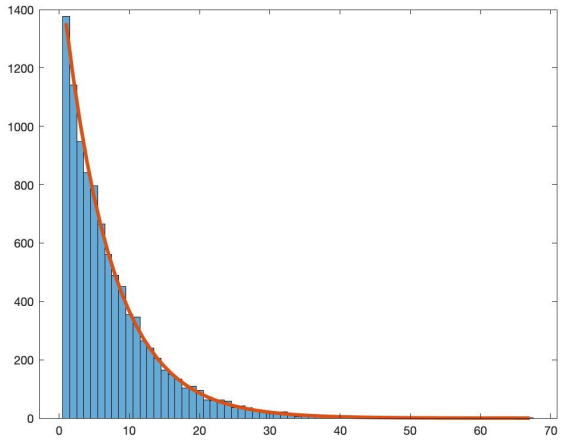

Figure 1: Stimulation of dt=0.001

(b) stimulation when dt = 0.005, we can see the histgram still fit the curve well, with slightly more deviation.

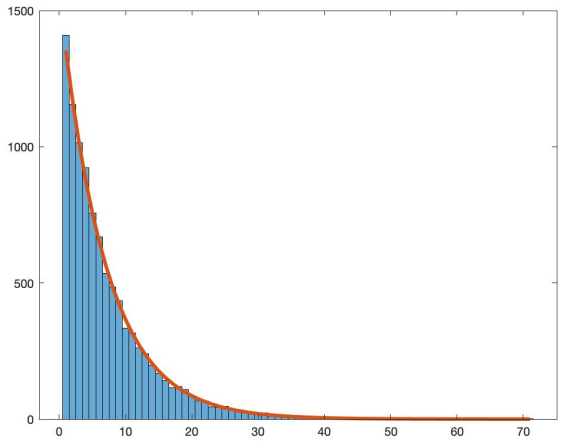

Figure 2: Stimulation of dt=0.005

stimulation when dt = 1, when dt is so huge, the histgram doesn't fit the curve at all. We get Pn by discard the higher order term of dt, this is only valid when dt is much smaller than 1.

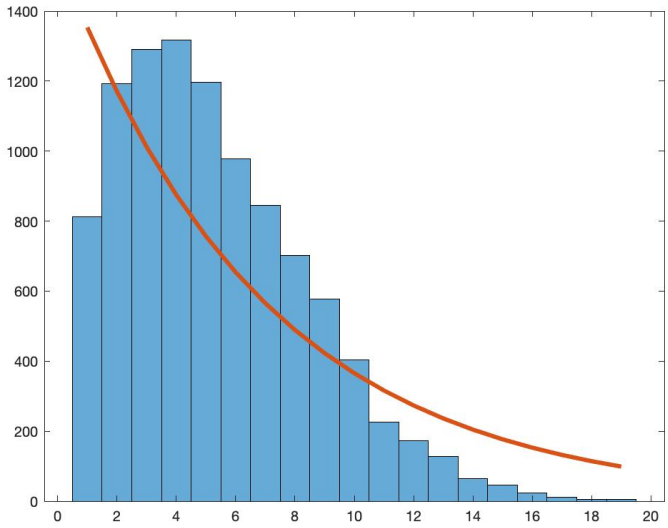

Figure 3: Stimulation of dt=1

(c) Inlcuding death rate

Now consider what happens if, in addition to cells dividing, the cells in the population may also die, at rate μ. That is, in each interval [t, t + Δt], the likelihood of a given cell dying is $\mu\Delta t$. Explain in words how you would need to modify your stochastic simulation algorithm to include cell death.

For each inner for loop, I need to use rand(pop,1) to generate another random vector, and for the value that less than $\mu\Delta t$ represent a death of individual. The pop need to substract the number of stimulated death. When Δt is small, the time span is only enough for one action to happen, so a cell is not going to dividing and dying within the same time span.

(d) stochastic simulation from part(c)

Modified code:

```
for j=1:trials
    for i=1:tsteps-1
        pop = N(i,j); %get current pop.
        randVec = rand(pop,1); %vector of N rand #s on (0,1)
        cellBorn = length(find(randVec<lambda*dt)); %number of cells that divided
        randVec2 = rand(pop,1);
        cellDead = length(find(randVec2<mu*dt));
        N(i+1,j) = N(i,j) + cellBorn - cellDead; %recurrence eq.
    end
end
```

Average population size against time

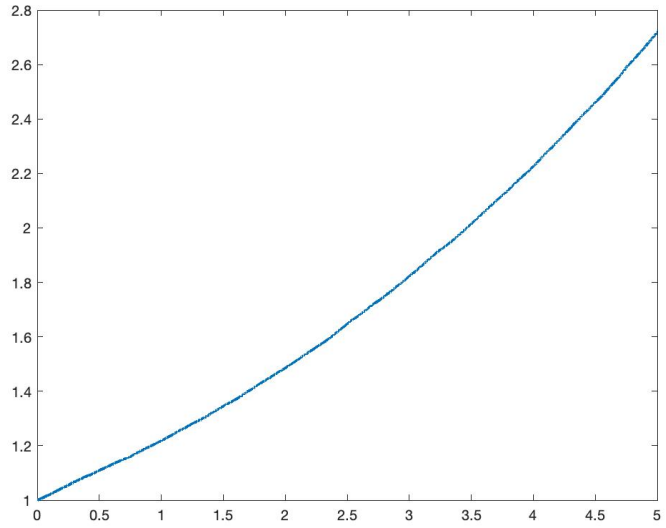

Figure 4: Average population size against time

Below is the plot of log average population against time. We can see the graph is pretty linearal, meaning the original population is exponential growth with time. the slope is around 0.2, which is the rate of growth.

Part II Stochastic simulations of a death process

We will now build a stochastic model for how a population of bacteria is depleted by anantibiotic. Assume that at time t = 0, there are precisely N bacteria in the population. Once the antibiotic is added, the bacteria stop dividing (that is, $\lambda = 0$), and start to die with a mortality rate μ. That is, in time Δt, the probability of a given cell dying is $\mu\Delta t$. We want to recalculate the probability mass function Pn(t), which represents the probability that there are n cells at time t.

(a) initial condition:

$$P_n(0) = \begin{cases} 0 & n \neq N \\ 1 & n = N \end{cases}$$

(b) derive differential equation

$$P_n(t + \Delta t) = \delta_{n+1} P_{n+1}(t) + \gamma_n P_n(t)$$

where δ_{n+1} represent the probability that exactly one death occur among n+1 individuals and γ_n represent the probabllity that no death occur among n individuals. Easy to derive:

$$\gamma_n = (1 - \mu\Delta t)^n \approx 1 - \mu n \Delta t$$

$$\delta_{n+1} \approx 1 - (1 - \mu\Delta t)^{n+1} \approx \mu(n + 1)\Delta t$$

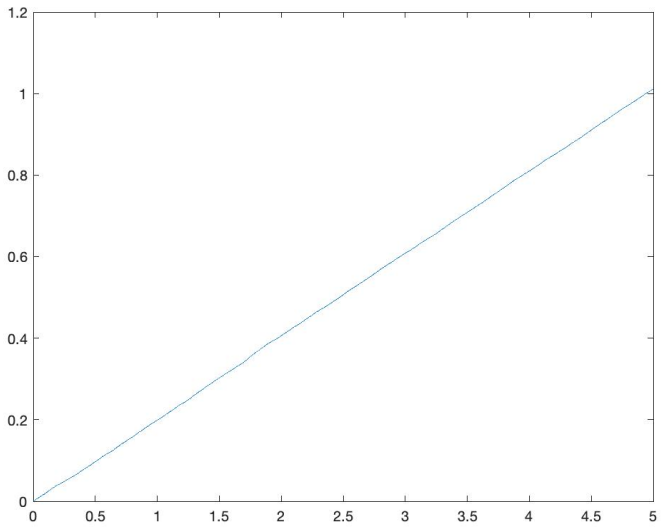

Figure 5: Logarithmic scales average population size against time

Thus

$$P_n(t+\Delta t) = \mu(n+1)\Delta t P_{n+1}(t) + (1-\mu n\Delta t)P_n(t)$$

$$\frac{P_n(t+\Delta t) - P_n(t)}{\Delta t} = \mu(n+1)P_{n+1}(t) - \mu n P_n(t)$$

So as $t \to 0$

$$\frac{dP_n}{dt} = \mu(n+1)P_{n+1}(t) - \mu n P_n(t)$$

When n = N, $P_{N+1} = 0$**,** $\frac{dP_N}{dt} = -\mu N P_N(t)$

(c) plug in initial condition

$$\frac{dP_N}{dt} = -\mu N P_N(t)$$

$$P_N(t) = Ce^{-\mu Nt}, P_N(0) = 1$$

$$P_N(t) = e^{-\mu Nt}$$

Now, plug in $P_N(t)$ to find $P_{N-1}(t)$

$$\frac{dP_{N-1}}{dt} = \mu N P_N(t) - \mu(N-1)P_{N-1}(t)$$

$$\frac{dP_{N-1}}{dt} + \mu(N-1)P_{N-1}(t) = \mu N e^{-\mu Nt}$$

$$\frac{dP_{N-1}}{dt}e^{\mu(N-1)t} + \mu(N-1)e^{\mu(N-1)t}P_{N-1}(t) = \mu N e^{-\mu Nt}e^{\mu(N-1)t}$$

$$e^{\mu(N-1)t}P_{N-1}(t) = \int \mu N e^{-\mu Nt}e^{\mu(N-1)t}dt = \int \mu N e^{-\mu t}dt$$

$$e^{\mu(N-1)t}P_{N-1}(t) = -Ne^{-\mu t} + C$$

$$P_{N-1}(t) = -Ne^{-\mu Nt} + Ce^{-\mu(N-1)t}, P_{N-1}(0) = 0$$

$$P_{N-1}(t) = Ne^{-\mu Nt}(e^{\mu t} - 1)$$

Next, plug in $P_{N-1}(t)$ to find $P_{N-2}(t)$

$$\frac{dP_{N-2}}{dt} = \mu(N-1)P_{N-1}(t) - \mu(N-2)P_{N-2}(t)$$

$$\frac{dP_{N-2}}{dt} + \mu(N-2)P_{N-2}(t) = \mu(N-1)Ne^{-\mu Nt}(e^{\mu t} - 1)$$

$$\frac{dP_{N-2}}{dt}e^{\mu(N-2)t} + \mu(N-2)e^{\mu(N-2)t}P_{N-2}(t) = \mu(N-1)Ne^{-\mu Nt}(e^{\mu t} - 1)e^{\mu(N-2)t}$$

$$e^{\mu(N-2)t}P_{N-2}(t) = N(N-1)(\frac{1}{2}e^{-2\mu t} - e^{-\mu t}) + C$$

$$P_{N-2}(t) = N(N-1)(\frac{1}{2}e^{-N\mu t} - e^{-(N-1)\mu t}) + Ce^{-(N-2)\mu t}, P_{N-2}(0) = 0$$

$$P_{N-2}(t) = N(N-1)(\frac{1}{2}e^{-N\mu t} - e^{-(N-1)\mu t}) + \frac{1}{2}N(N-1)e^{-(N-2)\mu t}$$

$$P_{N-2}(t) = N(N-1)(\frac{1}{2}e^{-N\mu t} - e^{-(N-1)\mu t} + \frac{1}{2}e^{-(N-2)\mu t})$$

Now we have three functions, they all fit with the following fomula:

$$P_n(t) = \binom{N}{n}\sum_{k=0}^{N-n}\binom{N-n}{k}(-1)^{k+N-n}e^{(k-N)\mu t}$$

.

(d)Examine the equation with matlab

```
% Code for build up Pn
syms k
for n=1:maxPop
    P(n) = nchoosek(N0,n) * symsum(nchoosek(N0-n,k)*(-1)^(k+N0-n)*exp((k-N0)*mu*T),k,0,N0-n);
end
 ...
%plot
histogram(N(end,:), endCount) %histogram of final cell counts
hold on
plot(1:maxPop, P * 1000, 'LineWidth', 3) %probability mass function
```

Figure6, The curve fits hisgram at T=10,N0=20,dt=0.001,μ=0.1

(e) Suppose that at time t, the average number of cells in the population is $\bar{n}(t)$. Between time t and t + Δt, how many cells, on average, will die? Base on the infomation, derive ODE for $\bar{n}(t)$

We know the death rate is μ, from which we can derive following equation:

$$\bar{n}(t + \Delta t) = \bar{n}(t) - \mu\bar{n}\Delta t$$

Organize a bit

$$\frac{\bar{n}(t + \Delta t) - \bar{n}(t)}{\Delta t} = -\mu\bar{n}$$

As $\Delta t \to 0$, we get the ODE:

$$\frac{d\bar{n}}{dt} = -\mu\bar{n}$$

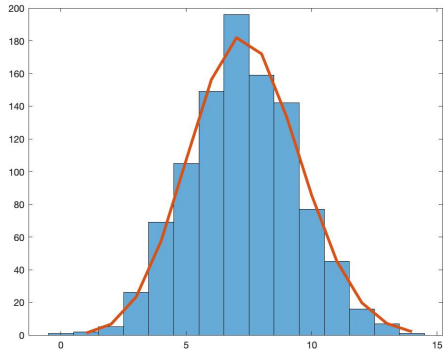

Figure 6: Histgram for anitbiotic model

(f) Find the solution of the differential equation and show that it accords with stimulation.

This is first order homogeneous differential equation, with initial condition $\bar{n}(0) = 20$

$$\frac{d\bar{n}}{\bar{n}} = -\mu dt$$

$$\log(\bar{n}) = -\mu t$$

$$\bar{n} = Ce^{-\mu t}, 20 = Ce^{-\mu t} \Rightarrow C = 20$$

$$\bar{n} = 20e^{-\mu t}$$

```
bar = zeros(tsteps,1);
for i = 1:tsteps
    bar(i) = 20*exp(-mu*tvec(i));
end

figure(3)
plot(tvec, mean(N'), 'LineWidth', 2) %plot average population size versus time
hold on
plot(tvec, bar, 'LineWidth', 2)
```

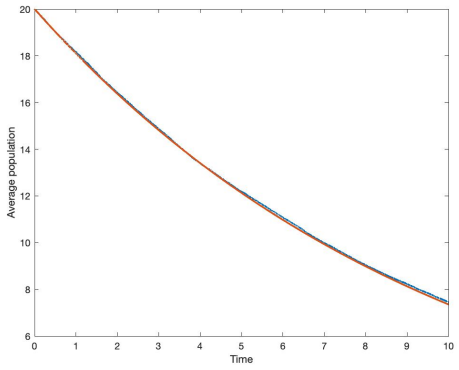

Figure 7: Average population against time

As we can see, the result of stimulation overlap with the curve derived above

(g) Run a stochastic simulation for a long enough time for the bacteria population to reach zero.

I adjust T to 100, and the blue line, stimulation, reaches 0 at around 80, however the red line, equation curve, never reaches 0, which is just a fact of exponential decay. Apparently, stochastic model is more realiztic when population is small. The reason is when the sample size is small, the randomness can be very large. Deterministic model only counts for average behavior. Therefore, stochastic model is more reliztic as it take randomness into account

Part III Population genetics

The Moran process is a model that describes how the diversity of populations changes with time. Consider a population of N = 10 cells. These cells have some form of diversity (i.e., different genes): to keep the biology simple, let's imagine that initially half of the cells are colored red and the other half are colored green. At each time step, exactly one cell divides. We pick this cell at random, and make a copy of it (if the original cell is red, then its child will also be red, etc.). Because of overcrowding, the overall number of cells must remain constant (i.e., equal to N). Each time a cell divides, one must be pushed out of the population. So pick one of the original N cells and remove it from the population (in particular, the cell that is removed could be the same or different as the one that divides).

(a) Do you expect the average number of red cells (initially 5) to change over time? Explain why or why not.

R(k+1)=R(k)+C(k), where R(k) is number of red cell at k-reproduction, C(k) is 0, 1 or -1, representing the change of R at k-reproduction. Expected value for C is $1/3\times1+1/3\times(-1)+0=0$. So average of R is no going to change

(b) simulate a single Moran process for a population of N = 10 cells.

```
%% This script simulates a Moran Process for N cells %%
clear; close all;
% Choose number of trials to run
numTrials = 1;
% Number of cells in population
N = 10;
% Initialize number of Red cells
init = 5;
% Initialize data cell to store Red cell data
Data{numTrials}=[];
% Initialize a vector to store time it takes to become homogeneous
tmax = zeros(numTrials,1);
% Repeat the process numTrials times
for trial = 1:numTrials
    N_Red(1) = init; %start with initial number for red cells
    i = 1; %start counter at 1
    % Track cell divisions until cells are homogeneous
    % (Note: we can do this while loop without worrying about it being
    % infinite, because we know we will have either 0 or N red cells in
    % finite time.)
    while N_Red(i) ~= 0 && N_Red(i) ~= N
        randNum1 = rand; randNum2 = rand;
```

```
        % Red cell divides and pushes out green cell
        if randNum1 < N_Red(i)/N && randNum2 > N_Red(i)/N
            N_Red(i+1) = N_Red(i)+1;
        % Green cell divides and pushes out red cell
        elseif randNum1 > N_Red(i)/N && randNum2 < N_Red(i)/N
            N_Red(i+1) = N_Red(i)-1;
        % Red cell divides and pushes out red; or green divides and pushes
        % out green. Both result in no change.
        else
            N_Red(i+1) = N_Red(i);
        end
        i = i+1; %increase counter
    end
    %store number of steps it takes to become homogeneous
    tmax(trial)=i-1;
    %store time series data in Data
    Data{trial}=N_Red;
    clear N_Red i
end

% Plot all the time series data on one figure
for j = 1:numTrials
    figure(1)
    title(['Moran Process for Cell Division, Trials = ', num2str(numTrials)])
    ylabel('Number of Red Cells')
    xlabel('Time Step')
    plot(0:tmax(j),Data{j})
    hold on
end
hold off

%print mean time to homogenize
mean(tmax)
```

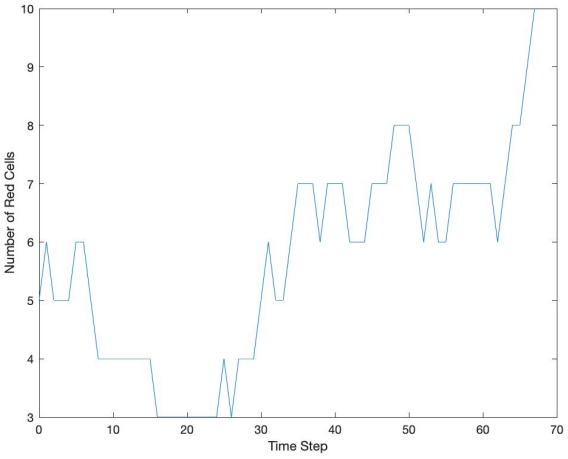

Figure 8: Population of Red cell in one stimulation

10 cell all become red in 67 generation

(c) Simulate 100 populations of N = 10 cells.

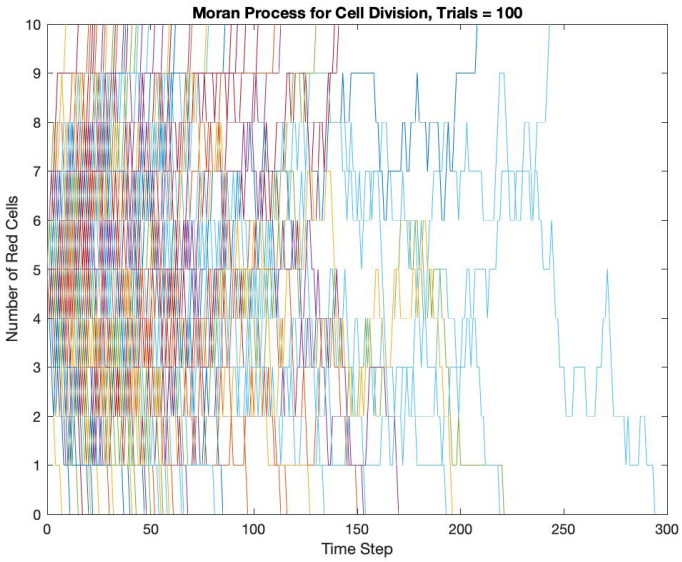

Figure 9: Population of Red cell in 100 stimulation

All the group honogenized within 300 generations

(d) Reconcile answer from part (c) with answer from part (a)

In part a, we were talking that the expected value for R is not going to change. As for part c, even number of red cell changed in every single group, the average number of red cell still remains the same The fundamental of all the population eventual homogenize is that the probability of R = 10 or R = 0 is non-zero, which is gauranteed to happen given enough try. Once R = 10 or R = 0 there is only one type of cell left, and the pmf of C become P(C=0)=1. So the populatin will stay homogenized

(e) Use simulations with different values of N to show that the average time for the population to become homogeneous is proportional to N^2 . Calculate the average time for the population to become homogeneous for each N by running 100 stochastic simulations. Plot these data in a way that makes it clear that the time to become homogeneous is proportional to N^2.

The average slope is 0.8137

```
% Code used for stimulation and find slope
tmean = zeros(50,1);
slope = zeros(50,1);
for N = 2:2:100
    init = N/2;
    for trial = 1:numTrials
        N_Red(1) = init; %start with initial number for red cells
        i = 1; %start counter at 1
        % Track cell divisions until cells are homogeneous
        % (Note: we can do this while loop without worrying about it being
```

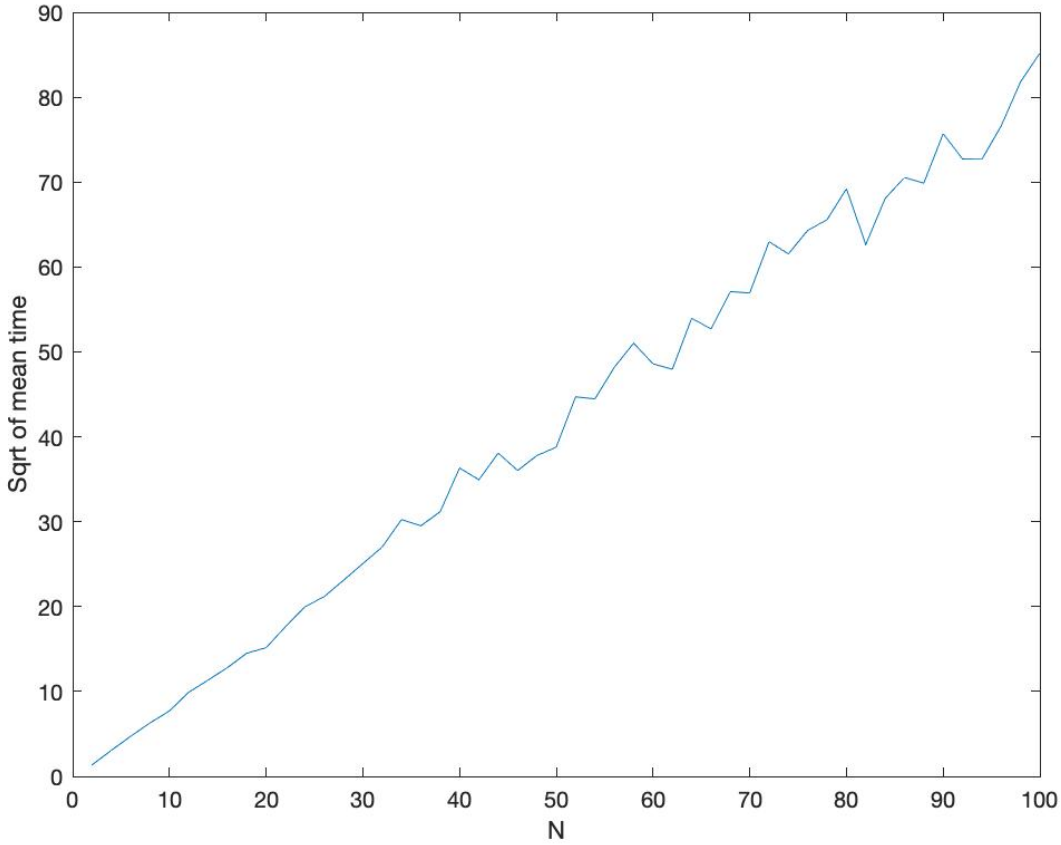

Figure 10: Square root of average time of 100 stimulation against corresponding total cell numbers

```
        % infinite, because we know we will have either 0 or N red cells in
        % finite time.)
        while N_Red(i) ~= 0 && N_Red(i) ~= N
            randNum1 = rand; randNum2 = rand;
            % Red cell divides and pushes out green cell
            if randNum1 < N_Red(i)/N && randNum2 > N_Red(i)/N
                N_Red(i+1) = N_Red(i)+1;
            % Green cell divides and pushes out red cell
            elseif randNum1 > N_Red(i)/N && randNum2 < N_Red(i)/N
                N_Red(i+1) = N_Red(i)-1;
            % Red cell divides and pushes out red; or green divides and pushes
            % out green. Both result in no change.
            else
                N_Red(i+1) = N_Red(i);
            end
            i = i+1; %increase counter
        end
        %store number of steps it takes to become homogeneous
        tmax(trial)=i-1;
        %store time series data in Data
        Data{trial}=N_Red;
        clear N_Red i
    end
    tmean(N)=sqrt(mean(tmax));
    slope(N)=tmean(N)/N;
end

figure(1)
plot(2:2:100,tmean(2:2:100))
ylabel('Sqrt of mean time')
xlabel('N')
sum(slope)/50
```