Final Exam - Question Set 1

Trilokinath Modi

March 23, 2021

1 Q1

Given,

$$\begin{aligned} b\delta t &= \textbf{Prob}(\text{one individual gives birth in time } \delta t) \\ p &= \textbf{Prob}(\text{The birth will be 1 new individual}) \\ 1-p &= \textbf{Prob}(\text{The birth will be 2 new individuals}) \end{aligned}$$

1.1 Q1a

Let,

$$Q_n(t) = \textbf{Prob}(\text{Population} = n \text{ at time} = t)$$

Then the possible scenarios that frame the master equation are shown in equation set 1.

$$\begin{aligned} n-1 \quad &\xrightarrow{p} \quad n \quad \xrightarrow{p} \quad n+1 \\ n-1 \quad &\xrightarrow{p} \quad n \quad \xrightarrow{1-p} \quad n+2 \\ n-2 \quad &\xrightarrow{1-p} \quad n \quad \xrightarrow{p} \quad n+1 \\ n-2 \quad &\xrightarrow{1-p} \quad n \quad \xrightarrow{1-p} \quad n+2 \end{aligned} \tag{1}$$

Note that, the 4 scenarios mentioned in equation 1 simply means that the population is going from $n-1$ or $n-2$ to n, followed by n to $n+1$ or $n+2$. Using this the resulting master equation can be written as shown in equation 2

$$\begin{aligned} Q_n(t+\delta t) = Q_n(t) + b\delta t\{&p(n-1)Q_{n-1}(t) + (1-p)(n-2)Q_{n-2}(t) \\ &- pnQ_n(t) - (1-p)nQ_n(t)\} \end{aligned} \tag{2}$$

$$\implies Q_n(t+\delta t) = Q_n(t) + b\delta t\left\{p(n-1)Q_{n-1}(t) + (1-p)(n-2)Q_{n-2}(t) - nQ_n(t)\right\} \tag{3}$$

The equation 3 can then be written as equation 4.

$$\frac{Q_n(t+\delta t) - Q_n(t)}{\delta t} = b\left\{p(n-1)Q_{n-1}(t) + (1-p)(n-2)Q_{n-2}(t) - nQ_n(t)\right\} \tag{4}$$

Taking limit $\delta t \to 0$, we get equation 5.

$$\frac{dQ_n(t)}{dt} = b\left\{p(n-1)Q_{n-1}(t) + (1-p)(n-2)Q_{n-2}(t) - nQ_n(t)\right\} \tag{5}$$

Multiplying equation 5 by n and sum over n, we get equation 6.

$$\frac{1}{b}\frac{d}{dt}\sum_{n=0}^{\infty} nQ_n(t) = p\sum_{n=0}^{\infty} n(n-1)Q_{n-1}(t) + (1-p)\sum_{n=0}^{\infty} n(n-2)Q_{n-2}(t) - \sum_{n=0}^{\infty} nnQ_n(t) \tag{6}$$

In equaiton 6, Changing variables from $n = n_1 + 1$ in first sum and $n = n_2 + 2$ in second term, we get equation 7.

$$\frac{1}{b}\frac{d}{dt}\sum_{n=0}^{\infty} nQ_n(t) = p\sum_{n_1=-1}^{\infty} (n_1+1)(n_1)Q_{n_1}(t) + (1-p)\sum_{n_2=-2}^{\infty} (n_2+2)(n_2)Q_{n_2}(t) - \sum_{n=0}^{\infty} n^2Q_n(t) \tag{7}$$

Since the probability for negative population is 0, the limits in summation can be changed to $0 \to \infty$ due to presence of probability terms $Q_{n1}(t), Q_{n2}(t)$. The equation 7 is rewritten as 8.

$$\frac{1}{b}\frac{d}{dt}\sum_{n=0}^{\infty} nQ_n(t) = p\sum_{n=0}^{\infty}(n+1)(n)Q_n(t) + (1-p)\sum_{n=0}^{\infty}(n+2)(n)Q_n(t) - \sum_{n=0}^{\infty} n^2Q_n(t) \tag{8}$$

$$\begin{aligned} \implies \frac{1}{b}\frac{d}{dt}\sum_{n=0}^{\infty} nQ_n(t) &= p\sum_{n=0}^{\infty} n^2Q_n(t) + p\sum_{n=0}^{\infty} nQ_n(t) \\ &\quad - p\sum_{n=0}^{\infty} n^2Q_n(t) - p\sum_{n=0}^{\infty} 2nQ_n(t) \\ &\quad + \sum_{n=0}^{\infty} n^2Q_n(t) - \sum_{n=0}^{\infty} n^2Q_n(t) + \sum_{n=0}^{\infty} 2nQ_n(t) \end{aligned}$$

$$\implies \frac{1}{b}\frac{d}{dt}\sum_{n=0}^{\infty} nQ_n(t) = (2-p)\sum_{n=0}^{\infty} nQ_n(t) \tag{9}$$

The average population size n at time t is given by $\langle n(t)\rangle = \sum_{n=0}^{\infty} nQ_n(t)$. Using this in equation 9, we get equation 10.

$$\frac{d\langle n(t)\rangle}{dt} = b(2-p)\langle n(t)\rangle \tag{10}$$

$$\implies \langle n(t)\rangle = Ce^{b(2-p)t} \tag{11}$$

At $t = 0$, $C = n_0$, where n_0 is initial population size. Hence, the expectation value of the population size is written in equation 12.

$$\langle n(t)\rangle = n_0e^{b(2-p)t} \tag{12}$$

1.2 Q1b

Gillespie algorithm is used to simulate the birth process. Given(and let) parameters are,

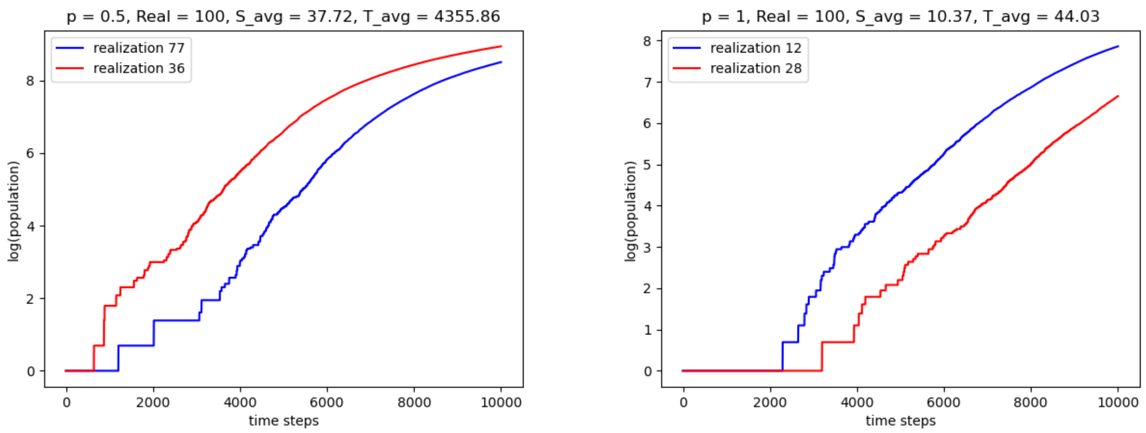
$$
\begin{aligned}
b &= 0.1 \\
p &\in \{0.5, 1\} \\
N_0 &= 1 \\
dt &= 0.01
\end{aligned}
$$

1.2.1 i

Plots for $p \in \{0.5, 1\}$ is shown in figure 1. Formulas for computing stochastic average and theoretical average is given in equation 13, 14. Note, that in both equations dt doesn't imply derivative, it implies *time* and t implies *time step*. And $N_{(i,j)}$ = population in realization i, time step j.

$$s_{avg} = \frac{\sum_{i=1}^{100} \frac{\sum_{j=1}^{10000} N_{(i,j)}}{\sum_{t=0}^{10000} tdt}}{\text{Total Realizations}} \tag{13}$$

$$t_{avg} = \frac{\sum_{t=0}^{20000} N_0 e^{b(2-p)tdt}}{\sum_{t=0}^{20000} tdt} \tag{14}$$



(a) (b)

Figure 1: Plots (a)(b) portrays birth process for the given model with y-axis as $log(population)$ and x-axis as *timesteps*. The time step in each iteration is taken as 0.01. Hence, a value in x-axis, e.g. 10000 implies 10000 time steps of 0.01 time. The plot contains two realization of birth process. The realizations are randomly chosen from 100 realizations. The plot title contains value of p, maximum realization as *Real*, stochastic average as s_{avg} and theoretical average as t_{avg}.

1.2.2 ii

The individual realizations doesn't agree with the theory for small population sizes. In the experiments involved in section 1.2.1, the initial population sizes is small. An explanation for disagreement can be made by standard deviation. As we observed in equation 12, that on average the stochastic model follows deterministic Malthus growth model. A known expression for standard deviation of stochastic $N(t)$ is $\sqrt{N_0}e^{rt}\sqrt{1-e^{rt}}$. If we let $r = b(2-p)$, we get the expression 15.

$$\mathrm{std}(N(t)) = \sqrt{N_0}e^{b(2-p)t}\sqrt{1-e^{b(2-p)t}} \tag{15}$$

The relative standard deviation then becomes,

$$\frac{\mathrm{std}(N(t))}{\langle N(t)\rangle} = \frac{\sqrt{N_0}e^{b(2-p)t}\sqrt{1-e^{b(2-p)t}}}{N_0 e^{b(2-p)t}} \tag{16}$$

For large t, we can rewrite the equation 16 as equation 17.

$$\frac{\mathrm{std}(N(t))}{\langle N(t)\rangle} \approx \frac{1}{\sqrt{N_0}} \tag{17}$$

Hence, for any realization of stochastic $N(t)$, the relative standard deviation is inversely proportional to N_0, implying larger the initial population size, smaller the deviation will be. This explains the large deviations in small population in the experiments conducted in 1.2.1.

1.2.3 iii

As t increases, the standard deviation increases in the order of $\langle N(t)\rangle$ since $\sqrt{N_0} = 1$. Hence, as the number of timesteps increased, the standard deviation between theoretical expected and simulated increases exponentially in $b(2-p)t$. The rate $b(2-p)t = 0.1(2-p)t$ and clearly for $t > \frac{0.1}{2-p}$ the deviation goes extremely large. Hence, the average population doesn't agree with the theory which can also be seen from the generated values. The deviation in the case of $p = 0.5$ is even larger than $p = 1$ because of the factor $2-p$. This can also be supported intuitively by claiming that the population in case of $p = 0.5$ has a chance of increasing by 2 individuals in every time step per individual. This makes the theoretical predictions even larger.

1.3 Q1c

Given, per capita growth rate $r = b(1 - \frac{N}{K})$.
A starting point for the deterministic dynamics can be rate of change of population over time is equal to per capita growth rate times population size. In the form of equation, this can be written as shown in equation 18.

$$\frac{dN(t)}{dt} = b\left(1 - \frac{N(t)}{K}\right)N(t) \tag{18}$$

The expression in equation 18 can be rewritten as shown in equation 19. In below expressions, aliasing $N(t)$ as N.

$$\begin{aligned}
&\frac{dN}{dt} = b\left(1 - \frac{N}{K}\right)N \\
\implies &\frac{dN}{dt} = b\left(\frac{KN - N^2}{K}\right) \\
\implies &\frac{dN}{N^2 - KN} = -\frac{b}{K}dt \\
\implies &\frac{dN}{N^2 - KN} = -\frac{b}{K}dt \\
\implies &\frac{dN}{\left(N - \frac{K}{2}\right)^2 - \frac{K^2}{4}} = -\frac{b}{K}dt
\end{aligned} \tag{19}$$

Integrating equation 19 both sides indefinitely, we get equation 20.

$$\begin{aligned}
&\int \frac{dN}{\left(N - \frac{K}{2}\right)^2 - \frac{K^2}{4}} = \int -\frac{b}{K}dt \\
\implies &\frac{1}{2\frac{K}{2}} \log\left|\frac{N - \frac{K}{2} - \frac{K}{2}}{N - \frac{K}{2} + \frac{K}{2}}\right| = -\frac{b}{K}t + C \\
\implies &\frac{1}{K}\log\left|\frac{N-K}{N}\right| = -\frac{b}{K}t + C
\end{aligned} \tag{20}$$

Note that, the general integral form $\int \frac{dx}{x^2 - a^2} = \frac{1}{2a} log\left|\frac{x-a}{x+a}\right|$ is used in reaching equaiton 20. Assuming $CK = C$ and cancelling out K on both sides of the equation 20, we can rewrite equation 20 as 21.

$$\begin{aligned}
&\frac{1}{K}\log\left|\frac{N-K}{N}\right| = -\frac{b}{K}t + C \\
\implies &\log\left|\frac{N-K}{N}\right| = -bt + C \\
\implies &\begin{cases} \log\left(\frac{N-K}{N}\right) = -bt + C, & \text{if } N > K \\ \log\left(\frac{K-N}{N}\right) = -bt + C, & \text{if } N < K \end{cases} \\
\implies &\begin{cases} 1 - \frac{K}{N} = C\exp(-bt), & \text{if } N > K \\ \frac{K}{N} - 1 = C\exp(-bt), & \text{if } N < K \end{cases} \\
\implies &\begin{cases} N = \frac{K}{1 - C\exp(-bt)}, & \text{if } N > K \\ N = \frac{K}{1 + C\exp(-bt)}, & \text{if } N < K \end{cases}
\end{aligned} \tag{21}$$

Removing the aliasing $N(t) = N$, we get an expression for $N(t)$. The constant can be evaluated at $N(0)$. The equation 21 can then be rewritten as 22.

$$\begin{aligned}
&\begin{cases} N(t) = \frac{K}{1-C\exp(-bt)}, & \text{if } N > K \\ N(t) = \frac{K}{1+C\exp(-bt)}, & \text{if } N < K \end{cases} \\
&\implies \begin{cases} N_0 = \frac{K}{1-C}, & \text{if } N > K \\ N_0 = \frac{K}{1+C}, & \text{if } N < K \end{cases} \\
&\implies \begin{cases} C = \frac{N_0-K}{N_0}, & \text{if } N > K \\ C = \frac{K-N_0}{N_0}, & \text{if } N < K \end{cases} \\
&\text{But if } N_0 < K, \text{ then } C = \frac{N_0 - K}{N_0} \text{for } N < K \text{ and,} \\
&C = \frac{K - N_0}{N_0} \text{ for } K > N \text{ to get positive } N(t) \\
\implies &\begin{cases} N(t) = \frac{KN_0\exp(bt)}{N_0\exp(bt)-(K-N_0)}, & \text{if } N > K \\ N(t) = \frac{KN_0\exp(bt)}{N_0\exp(bt)-(N_0-K)}, & \text{if } N < K \end{cases} \\
\implies &\begin{cases} N(t) = \frac{KN_0\exp(bt)}{-K+N_0(\exp(bt)+1)}, & \text{if } N > K \\ N(t) = \frac{KN_0\exp(bt)}{K+N_0(\exp(bt)-1)}, & \text{if } N < K \end{cases}
\end{aligned} \tag{22}$$

Rewriting expression 22 in larger font in equation 23.

$$\begin{cases} N(t) = \frac{KN_0e^{bt}}{-K+N_0(e^{bt}+1)}, & \text{if } N > K \\ \\ N(t) = \frac{KN_0e^{bt}}{K+N_0(e^{bt}-1)}, & \text{if } N < K \end{cases} \tag{23}$$

It can be noted that as $t \to \infty$, $N \to K$ in both cases of equation 23. Hence, considering $N_0 < K$, in realistic approach, the equation can be written as equation 24.

$$N(t) = \frac{KN_0e^{bt}}{K + N_0(e^{bt} - 1)} \tag{24}$$

Equation 24 is the required deterministic dynamics.

1.4 Q1d

Given set of parameters,

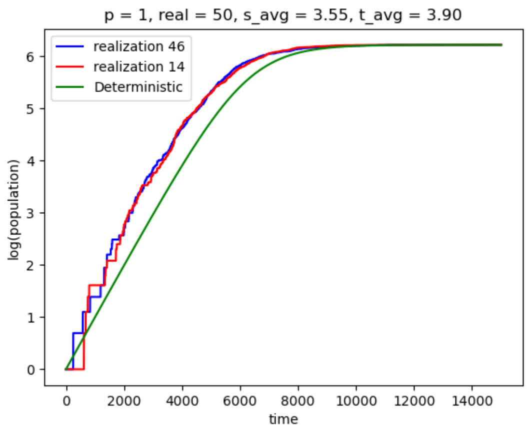
$$\begin{aligned}
b &= 0.1 \\
K &= 500 \\
N_0 &\in \{1, 10, 100\} \\
p &= 1
\end{aligned}$$

1.4.1 i

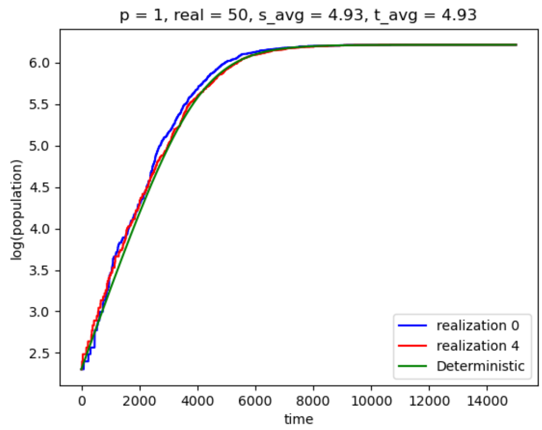
Plots for simulations beginning with initial population $N_0 \in \{1, 10, 100\}$ is shown in figure 2. Formulas for computing stochastic average and theoretical average is given in equation 27, 28. Note, that in both equations dt doesn't imply derivative, it implies *time* and t implies *time step*. And $N_{(i,j)}$ = population in realization i, time step j.

$$s_{avg} = \frac{\sum_{i=1}^{50} \frac{\sum_{j=1}^{20000} N_{(i,j)}}{\sum_{t=0}^{20000} tdt}}{\text{Total Realizations}} \tag{25}$$

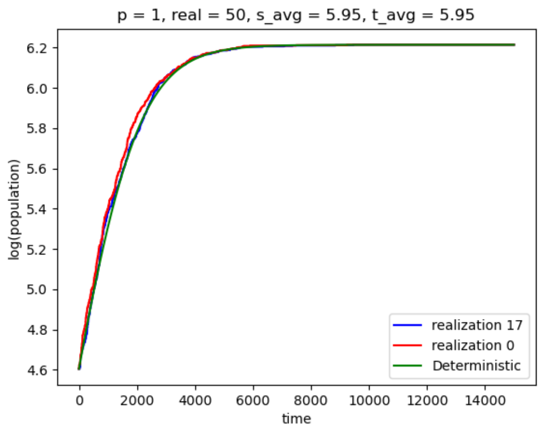
$$t_{avg} = \frac{\sum_{t=0}^{20000} \frac{KN_0 e^{btdt}}{K+N_0(e^{btdt}-1)}}{\sum_{t=0}^{20000} tdt} \tag{26}$$



(a)



(b)



(c)

Figure 2: Plots (a)(b)(c) portrays birth process for the given model with y-axis as $log(population)$ and x-axis as *timesteps*. The time step in each iteration is taken as 0.01. Hence, a value in x-axis, e.g. 10000 implies 10000 time steps of 0.01 time. The plot contains two realization of stochastic population dynamics and deterministic population growth. The realizations are randomly chosen from 50 realizations. The plot title contains value of p, maximum realization as $Real$, stochastic average as s_{avg} and theoretical average as t_{avg}.

1.4.2 ii

The individual realizations does agree for large initial population size. For small initial population sizes the individual realization is follows the trend of deterministic solution but is often not collinear until the population has reached close to non-zero steady state population. In general, the stochastic simulations agree more and more as initial population size is increased. With little increase in initial population size, there is great agreement between stochastic simulations and deterministic ones. This statement can be supported by the observations in plot 2(a)(b)(c). It can also be noted that, at large initial population(but $< K$) the system is close to stable steady state.

1.4.3 iii

The average population is almost collinear with deterministic solution in the limit of $t \to 0$ for case $N_0 = 10, N_) = 100$. The plot in figure 2(a) shows a small difference in average population of stochastic simulation as compared to deterministic solution which is due to initial stochastic fluctuations at small initial population size.

A master equation can for this model can be written as 28 governed by equation 27

$$
\begin{aligned}
n-1 \quad &\xrightarrow{b} \quad n \quad \xrightarrow{b} \quad n+1 \\
n-1 \quad &\xleftarrow{\frac{bN}{K}} \quad n \quad \xleftarrow{\frac{bN}{K}} \quad n+1
\end{aligned}
\tag{27}
$$

$$
\frac{dQ_n t}{dt} = b\big(n(n-1)Q_{n-1}(t) - n^2 Q_n t + n\frac{(n+1)}{K}(n+1) - \frac{n}{K}n^2 Q_n(t)\big) \tag{28}
$$

Following similar steps as in Q1a, we get with an exception that here $n \to n_1+1$ and $n \to n_2-1$ are used for positive terms in right side of master equation 28.

$$
\begin{aligned}
&\frac{1}{b}\frac{d}{dt}\sum_{n=0}^{\infty} nQ_n(t) = \sum_{n=0}^{\infty} nQ_n(t) - \frac{1}{K}\sum_{n=0}^{\infty} n^2 Q_n(t) \\
&\implies \frac{d\langle n(t)\rangle}{\langle n^2(t)\rangle - K\langle n(t)\rangle} = -\frac{b}{K}dt
\end{aligned}
\tag{29}
$$

The equation 29 is the exact form of deterministic equation comparing with 19. Since, this equation contains second moment it is expected that the standard deviation will be low. And hence, this explains that the stochastic model will result similar trends and population averages as compared to deterministic solutions.

Q1b

```
import numpy as np
import matplotlib.pyplot as plt
import os
MAX_REALIZATIONS = 100
MAX_ITER = 10000

dt = 0.01
N_0 = 1
b = 0.1
p = [0.5, 1]
class Gillespie():
    def __init__(self, p, dt, N_0, b):
        self.p = p
        self.b = b
        self.N = [N_0]
        self.t = [0]
        self.dt = dt
        self.pb = self.dt * self.b
        self.find_t()
        self.compute_avg()

    def find_t(self):
        while True:
            rand_expo = int(np.min(np.floor(np.random.exponential(scale=1/self.
                ↪ pb, size=self.N[-1]))))
            self.N += [self.N[-1]] * rand_expo
            rand_p = np.random.random()
            if rand_p <= self.p:
                self.N.append(self.N[-1] + 1)
            else:
                self.N.append(self.N[-1] + 2)
            if len(self.N) > MAX_ITER:
                break

    def compute_avg(self):
        self.avg = sum(self.N)/np.sum(dt * np.arange(len(self.N) - 1))


for iP in p:
    populations = list()
    average = list()

    for iRealization in range(MAX_REALIZATIONS):
        g1 = Gillespie(iP, dt, N_0, b)
        populations.append(g1.N)
        average.append(g1.avg)

    prac_average = (sum(average))/MAX_REALIZATIONS
```

```
    theor = list()
    for t in range(MAX_ITER):
        theor.append(N_0 * np.exp(b * (2 - iP) * t * dt))
    tt = dt * np.arange(MAX_ITER)
    # theory_average = N_0 * np.exp(b * (2 - iP) * MAX_ITER * dt)
    theory_average = sum(theor)/np.sum(tt)

    two_rand_realization = np.random.randint(0, MAX_REALIZATIONS, 2)
    cwd = os.getcwd()
    figNameP = cwd + "\\EXAM_Q1b_p_" + str(iP*10) + ".png"
    plt.figure()
    plt.xlabel("time steps")
    plt.ylabel("log(population)")
    plt.title("p = {0}, Real = {1}, S_avg = {2:.2f}, T_avg = {3:.2f}".format(
        ↪ iP, MAX_REALIZATIONS, prac_average, theory_average))
    plt.plot(np.log(np.array(populations[two_rand_realization[0]])), color="
        ↪ blue", label=f"realization {two_rand_realization[0]}")
    plt.plot(np.log(np.array(populations[two_rand_realization[1]])), color="
        ↪ red", label=f"realization {two_rand_realization[1]}")
    plt.legend()
    # plt.show()
    plt.savefig(figNameP)

# plt.figure()
# theor = list()
# for t in range(MAX_ITER):
# theor.append(N_0 * np.exp(b * (2 - p[0]) * t * dt))
# print(sum(theor)/(dt * MAX_ITER * (MAX_ITER + 1)/2))
# plt.plot(np.array(theor))
# plt.show()
```

Q1d

```
import numpy as np
import matplotlib.pyplot as plt
import os
MAX_REALIZATIONS = 1
MAX_ITER = 50000

dt = 0.01
b = 0.1
p = 1
K = 500

class Q1d():
    def __init__(self, p, N_0):
        self.b = 0.1
        self.p = p
        self.n = [N_0]
        self.dt = 0.01
        self.K = 500
        self.p_bn = self.b * self.dt
        self.find_t()
        self.compute_avg()

    def find_t(self):
        for iIter in range(MAX_ITER):
            self.tempPop = 0
            for iPop in range(self.n[-1]):
                self.rand_prob = np.random.random()
                if self.rand_prob <= self.p_bn * (1 - (self.n[-1]/self.K)):
                    if np.random.random() <= self.p:
                        self.tempPop += 1
            self.n.append(self.n[-1] + self.tempPop)

    def compute_avg(self):
        self.avg = sum(self.n) / np.sum(dt * np.arange(len(self.n) - 1))


for iN in [1, 10, 100]:

    populations = list()
    average = list()

    for iRealization in range(MAX_REALIZATIONS):
        g1 = Q1d(p, iN)
        populations.append(g1.n)
        average.append(g1.avg)

    prac_average = (sum(average)) / MAX_REALIZATIONS

    theor = list()
```

```python
for t in range(MAX_ITER):
    num = iN * K * np.exp(b * t * dt)
    den = iN * (np.exp(b * t * dt) - 1) + K
    theor.append(num/den)
theory_average = sum(theor) / (dt * MAX_ITER * (MAX_ITER + 1) / 2)

two_rand_realization = np.random.randint(0, MAX_REALIZATIONS, 2)

cwd = os.getcwd()
figNameP = cwd + "\\EXAM_Q1d_N_" + str(iN) + ".png"
plt.figure()
plt.xlabel("time")
plt.ylabel("log(population)")
plt.title("p = {0}, real = {1}, s_avg = {2:.2f}, t_avg = {3:.2f}".format(p
    ↪ , MAX_REALIZATIONS, prac_average, theory_average))
plt.plot(np.log(np.array(populations[two_rand_realization[0]])), color="
    ↪ blue", label=f"realization {two_rand_realization[0]}")
plt.plot(np.log(np.array(populations[two_rand_realization[1]])), color="
    ↪ red", label=f"realization {two_rand_realization[1]}")
plt.plot(np.log(np.array(theor)), color="green", label="Deterministic")
plt.legend()
plt.show()
# plt.savefig(figNameP)
```