Lab Report 1: Population Modeling

Team Members: Joie Green, Melissa Prado, Niva Razin, Alexander Ross

EA4 Lab Certification

We certify that the members of the team named below worked on this lab using only our own ideas, possibly with help from the TAs and our professors, specifically in the writing of the report, the programming and the analysis required for each task. We did not collaborate with any members of any other team nor did we discuss this lab or ask assistance from anyone outside of our team, the TAs and our professors. If we accessed any websites in working on these labs, these websites are explicitly listed in our report (with the exception of websites used only to get information on specific Matlab programs). We understand that there will be severe consequences if this certification is violated and there is unauthorized collaboration.

1. Name NIVARAZIN Signature _____ Date 1/21/20
2. Name Alexander Ross Signature _____ Date 1/21/2020
3. Name Joie Green Signature _____ Date 1/21/2020
4. Name Melissa Prado Signature _____ Date 1/21/2020

INTRODUCTION

In this lab, we learned about population modeling. We expanded on the basic doomsday-extinction model to develop a more sensitive, realistic, and concise equation that accounts for predator-prey interaction, the crowding effect, and time dependence. We used nondimensionalization and computational tools (e.g., Matlab's *ode45* solver and *fsolve* function) to integrate these factors into the more advanced model and evaluate the effects of different parameters. In turn, we have gained valuable experience in mathematical modeling, Matlab coding, critical thinking/analysis, writing, collaboration, and project management.

TASK 1

$$\frac{d\tilde{P}}{dt} = \tilde{P}\left(-c + \frac{\tilde{P}}{\tilde{\beta}+\tilde{\gamma}\tilde{P}}\right) \quad (1)$$

The units of left side of equation (1) are population/time. In order for the equation to be balanced, the right side must also have these units. The units of c, β, and γ can be determined using dimensional analysis:

$$\frac{[population]}{[time]} = [population]\left(-c + \frac{[population]}{\tilde{\beta}+\tilde{\gamma}[population]}\right).$$

Divide both sides by population units:

$$\frac{1}{[time]} = -c + \frac{[population]}{\tilde{\beta}+\tilde{\gamma}[population]}.$$

By expanding and further manipulating the equation we get:

$$\tilde{\beta} + \tilde{\gamma}[population] + (\tilde{\beta})(c)[time] + (\tilde{\gamma})(c)[time][population] = [population][time] \quad (2)$$

In order for the left-hand side of equation (2) to evaluate to units of [population][time], the units of the parameters must be [population][time] for $\tilde{\beta}$, 1/[time] for c, and [time] for $\tilde{\gamma}$.

TASK 2

We use nondimensionalization to transform equation (1) into an equation with a single parameter, *B*:

$$\frac{dP}{dt} = P\left(-1 + \frac{P}{1+BP}\right). \quad (3)$$

To do this, we must define two variables *P* and *t*:

$$\tilde{P}(\tilde{t}) = P_{ref}P(t) \ , \ \tilde{t} = t_{ref}t.$$

We first plug these terms into equation (1) and apply the chain rule:

$$\frac{d\tilde{P}}{d\tilde{t}} = \frac{P_{ref}}{t_{ref}}\frac{dP}{dt} = P_{ref}P\left(-c + \frac{P_{ref}P}{\tilde{\beta}+\tilde{\gamma}P_{ref}P}\right).$$

Then we manipulate the equation:

$$\frac{dP}{dt} = P\left(-t_{ref}c + \frac{t_{ref}P_{ref}P}{\tilde{\beta}+\tilde{\gamma}P_{ref}P}\right) \ . \quad (4)$$

In order for the $-t_{ref}c$ term of equation (4) to equal -1, as in equation (3), we see that $t_{ref} = \frac{1}{c}$. We plug this value of t_{ref} back into (4):

$$\frac{dP}{dt} = P\left(-1 + \frac{P_{ref}P}{c\tilde{\beta}+c\tilde{\gamma}P_{ref}P}\right) .$$

We'll now focus on the second term, $\frac{P_{ref}P}{c\tilde{\beta}+c\tilde{\gamma}P_{ref}P}$. In order to get the numerator of this fraction equal to P, we set $P_{ref} = c\tilde{\beta}$. We plug this value into the fraction, then factor out $\frac{c\tilde{\beta}}{c\beta}$, which reduces to 1:

$$\frac{c\tilde{\beta}P}{c\tilde{\beta}+c^2\tilde{\gamma}\tilde{\beta}P} = \left(\frac{c\tilde{\beta}}{c\beta}\right)\frac{P}{1+c\tilde{\gamma}P} = \frac{P}{1+c\tilde{\gamma}P} .$$

Now, we can see that $B = c\tilde{\gamma}$ gives equation (3).

TASK 3

One critical point of equation (3) is $P = 0$, which signifies a population's extinction. We can verify that this is a stable critical point using the derivative test:

$$\frac{dP}{dt} = P\left(-1 + \frac{P}{1+BP}\right) = -P + \frac{P^2}{1+BP} = f(P)$$

We find the derivative of $f(P)$ using the product rule:

$$\frac{df}{dP} = -1 + (2P)\left(\frac{1}{1+BP}\right) + (P^2)\left[\frac{-B}{(1+BP)^2}\right]$$

Plugging in for $P = 0$, we get:

$$f'(0) = -1 < 0$$

Our negative result indicates that $P = 0$ is a stable critical point.

We will now find a second critical point P_+ that, by definition of a critical point, gives $f(P_+) = 0$. We find P_+ by solving for P when the term $\left[-1 + \frac{P}{1+BP}\right]$ of equation (3) is set to zero:

$$-1 + \frac{P}{1+BP} = 0 \;\rightarrow\; P = 1 + BP \;\rightarrow\; P(1-B) = 1$$

$$\rightarrow \qquad P_+ = \frac{1}{(1-B)} \tag{5}$$

In order for P_+ to be physically significant, it must be greater than zero (i.e., there cannot be a negative population). So, $(1-B)$ must also be positive:

$$1 - B > 0 \;\rightarrow\; B < 1$$

Thus, B must be less than a critical value $B_{crit} = 1$.

Now, we can determine the stability of this critical point by qualitatively performing the derivative test. As discussed above, we take $B < 1$, such as $B = 0$, and $P > 0$:

$$\frac{df(P>0,\, B=0)}{dP} = -1 + 2P\left(\frac{1}{1+(0)P}\right) + P^2\left(\frac{-(0)}{(1+(0)P)^2}\right) = -1 + 2P + 0 = -1 + 2P > 0$$

Since we know $P > 0$, we know that the derivative test results in a positive value, indicating that $P = P_+$ is an unstable critical point.

TASK 4

As determined in Task 3, B must be less than 1 in order for P_+ to be physically significant. **Figure 2** shows a plot of the solution to equation (3) given two different initial values, $P = 1.999, 2.001$. Plugging $B = 0.5$ into equation (5) gives the critical point $P_+ = 2$. We also determined that P_+ must be an unstable critical point, which is consistent with our plots. That is, the function diverges from $P = 2$ as $t \rightarrow \infty$. With an initial value less than P_+, the function approaches zero (population progresses to extinction). With an initial value greater than P_+, the function approaches infinity as t→ ∞ (doomsday scenario).

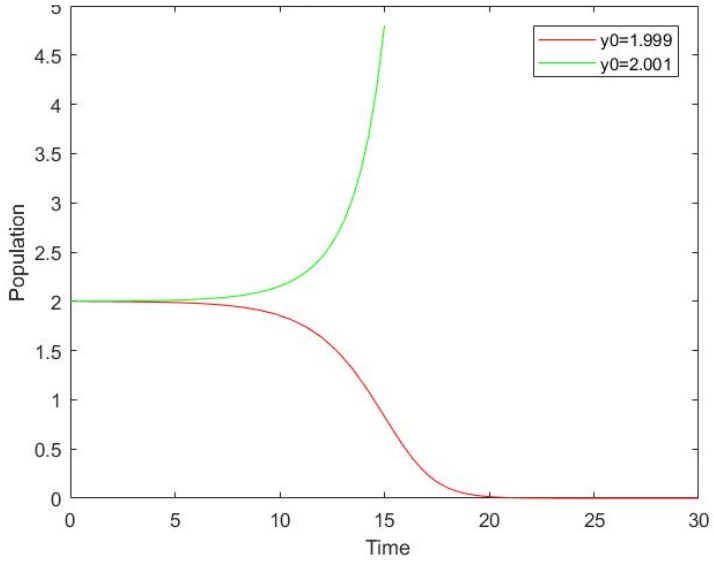


Figure 1. *Plots of population over time with two different initial conditions,* $P(0) = 1.999$ *and* $P(0) = 2.001$. *The plots exhibit starkly different, divergent behavior because their initial conditions, although only 0.002 apart, lie on opposite sides of the critical point* $P_+ = 2$.

TASK 5

Part 1

We now want to transform the equation:

$$\frac{dP}{dt} = P\left(-1 + \frac{P}{1+BP} - \varepsilon P^2\right) = PN(P) \tag{6}$$

to the form:

$$N(P) = \frac{f(P)}{(1+BP)}, \tag{7}$$

where *f(P)* is a cubic polynomial. To do this, we turn the inner quadratic equation of equation (6), $-1 + \frac{P}{1+BP} - \varepsilon P^2$, into a cubic polynomial over the common denominator $1 + BP$. We multiply the terms -1 and $-\varepsilon P^2$ by $\frac{1+BP}{1+BP}$:

$$\frac{dP}{dt} = P\left(-\frac{1+BP}{1+BP} + \frac{P}{1+BP} - \frac{\varepsilon P^2(1+BP)}{1+BP}\right)$$

$$\frac{dP}{dt} = P\left(\frac{-1-BP+P-\varepsilon P^2-\varepsilon P^2 BP}{1+BP}\right)$$

$$\frac{dP}{dt} = P\left(\frac{-1-BP+P-\varepsilon P^2-\varepsilon P^3 B}{1+BP}\right)$$

$$\frac{dP}{dt} = P\left(\frac{-\varepsilon BP^3-\varepsilon P^2-BP+P-1}{1+BP}\right)$$

Thus, we get equation (7) by setting $f(P) = -1 - BP + P - \varepsilon P^2 - \varepsilon BP^3$.

We know that the product of the roots of a polynomial is equal to the constant term (x^0) of the expanded polynomial. The constant term can thus indicate the signs of the roots. A polynomial to the *nth* term where *n* is odd will have a negative constant term if all the roots are positive (8). For example, when $n = 3$ and all three roots $x = p, q, r$ are positive, the constant term is negative:

$$(x-p)(x-q)(x-r) = x^3 - (p+q+r)x^2 + (pq+rp+rq)x - rpq \tag{8}$$

Alternatively, if just one (or all three) of the roots $x = -p, q, r$ is negative, the constant term is positive:

$$(x+p)(x-q)(x-r) = x^3 + (p-q-r)x^2 + (-pq-rp+rq)x + rpq$$

Additionally, factoring out a -1 from $f(P)$ and setting the coefficient of P^3 to 1 gives:

$$f(P) = -\left(\frac{1}{\varepsilon B} + \frac{(B-1)P}{\varepsilon B} + \frac{P^2}{B} + P^3\right),$$

and $f(P)$ is now in the form of (8). By the same logic, we see that $f(P)$ must have at least one negative root since the constant term of the expanded polynomial $\left(\frac{1}{\varepsilon B}\right)$ is positive (we have shown that B and ε are positive). Thus, $f(P)$ cannot have three positive roots.

Combining (6) and (7) gives:

$$\frac{dP}{dt} = P\left(\frac{f(P)}{1+BP}\right). \tag{9}$$

With the knowledge that B and P are positive, we see that the critical points of (9) are dictated by the roots of $f(P)$. Thus, (6) cannot have more than two critical points because $f(P)$ cannot have three positive roots.

Part 2

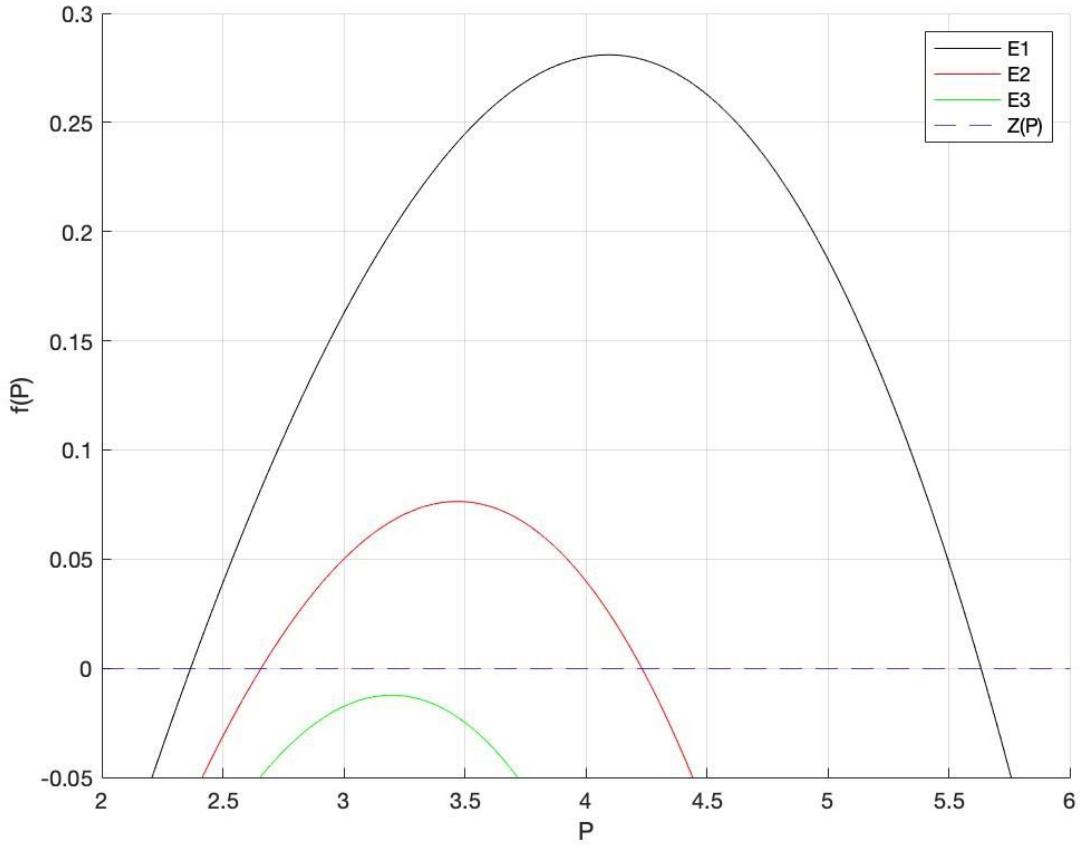
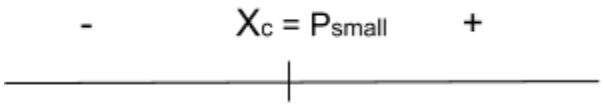


Figure 2. *f(P) versus P for three different values of* ε*, where* $E1 = 0.015$ *(black),* $E2 = 0.020$ *(red), and* $E3 = 0.023$ *(green).* $Z(P) = 0$ *is also shown in blue for reference.*

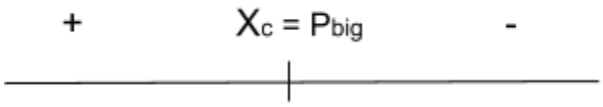
In **Figure 2**, we see two of the three critical points for $\varepsilon = E1$ and $\varepsilon = E2$ represented by the intersections of the red and the black curves, respectively, with $Z(P) = 0$. These critical points can be denoted as $P = P_{small}$ for each curve's first intersection point with $Z(P) = 0$ and $P = P_{big}$ for each curve's second intersection point with $Z(P) = 0$. The final critical point can be determined by looking at the initial differential equation (6). We can clearly see that a solution for $\frac{dP}{dt} = 0$ is $P = 0$, as the outer P term still remains in the equation. If we plug in $P = 0$, we get $\frac{dP}{dt} = 0$, making it a critical point. Thus, for $\varepsilon = \text{E1}$ and $\varepsilon = \text{E2}$, there are three physical critical points.

However, for $\varepsilon = \text{E3}$, there is only one physical critical point; its curve does not intersect $Z(P) = 0$. The only critical point for $\varepsilon = \text{E3}$ is $P = 0$.

Further, we can demonstrate the stability of these critical points by performing a sign test at the P_{small} and P_{big} intersection points. We indicate the sign of *f(P)* to the left and right of the critical points:



Above we see that $P = P_{small}$ is unstable, as its phase diagram indicates a divergence from the critical point (negative *f(P)* left of P_{small}, positive *f(P)* right of P_{small}). On the other hand, below we see that $P = P_{big}$ is stable, as its phase diagram indicates a convergence on the critical point (positive *f(P)* left of P_{big}, negative *f(P)* right of P_{big}). (Note that P_{small} and P_{big} may not always exist, as in the case of $\varepsilon = \text{E3}$.) Finally, as equation (6) shows, $\frac{dP}{dt} = PN(P)$, thus $P = 0$ is another critical point because it gives $\frac{dP}{dt} = 0$. As shown by way of the derivative test in task 3, we saw that $P = 0$ is stable.



Thus, we can see that equation (6) removes the unphysical behavior of equation (3) by making it so that the higher critical point is stable. This means that the population does not grow

infinitely as in the doomsday model but instead converges on a critical point, reflecting the crowding effect.

Part 3

To approximate P_{small}, we must first make the appropriate substitution:

$$0 = 0.5\varepsilon(P_{small})^3 - \varepsilon(P_{small})^2 - 0.5(P_{small}) + (P_{small}) - 1$$

$$0 = 0.5\varepsilon(P_+ + \nu\varepsilon)^3 - \varepsilon(P_+ + \nu\varepsilon)^2 - 0.5(P_+ + \nu\varepsilon) + (P_+ + \nu\varepsilon) - 1$$

$$0 = 0.5\varepsilon(\tfrac{1}{1-B} + \nu\varepsilon)^3 - \varepsilon(\tfrac{1}{1-B} + \nu\varepsilon)^2 - 0.5(\tfrac{1}{1-B} + \nu\varepsilon) + (\tfrac{1}{1-B} + \nu\varepsilon) - 1$$

$$0 = 0.5\varepsilon(2 + \nu\varepsilon)^3 - \varepsilon(2 + \nu\varepsilon)^2 - 0.5(2 + \nu\varepsilon) + (2 + \nu\varepsilon) - 1$$

Then, we can perform a higher order approximation based on the fact that $\varepsilon^3 << \varepsilon^2 << \varepsilon$ as $\varepsilon \to 0$. This eliminates most of the terms from the expression, and after several iterations of algebraic simplifications, we are left with:

$$-\tfrac{1}{2}(2)^3 - 2^2 + \tfrac{1}{2}\nu = 0$$

Solving this equation, we see that $\nu = 16$. Thus we can approximate P_{small} as $P_{approx_{small}} = 2 + 16\varepsilon$.

To approximate P_{big}, we must make the appropriate substitutions:

$$N(P) = (-1 + \tfrac{P}{1+BP} - \varepsilon P^2)$$

$$N(P) = (-1 + \tfrac{(\mu/\varepsilon^r)}{1+0.5(\mu/\varepsilon^r)} - \varepsilon(\mu/\varepsilon^r)^2)$$

Looking at the middle term on the right side of the above equation, we can show that as $\varepsilon \to 0$, the middle term will tend to $\frac{1}{B}$ or simply 2. This evaluation can be seen below:

$$\lim_{\varepsilon\to 0} \tfrac{(\mu/\varepsilon^r)}{1+0.5(\mu/\varepsilon^r)} = \lim_{\varepsilon\to 0} \tfrac{(\mu)}{\varepsilon^r+0.5(\mu)} = \tfrac{(\mu)}{0+0.5(\mu)} = \tfrac{1}{0.5} = 2$$

We can then solve for *r* under the condition that the last term, $\varepsilon(\mu/\varepsilon^r)^2$, must be independent of ε. This procedure can be seen below:

$$\varepsilon(\mu/\varepsilon^r)^2 = \varepsilon(\mu^2/\varepsilon^{r^2}) = \varepsilon\mu^2/\varepsilon^{r^2} \;\Rightarrow r = 1/2 \;\Rightarrow \varepsilon\mu^2/\varepsilon = \mu^2$$

Finally, we can solve for μ and determine P_{big}. This procedure is shown below:

$$\lim_{\varepsilon \to 0}(-1 + \frac{(\mu)}{\varepsilon^r + 0.5(\mu)} - \mu^2) = 0 \;\Rightarrow\; 0 = -1 + 2 - \mu^2 \;\Rightarrow\; \mu = 1$$

$$P_{approx_{big}} = (\mu/\varepsilon^r) = (1/\varepsilon^{1/2})$$

Part 4

ε	P_{small}	$P_{approx_{small}}$	P_{big}	$P_{approx_{big}}$	Trust
0.015	2.3670	2.2400	5.6330	8.1650	Y
0.020	2.6585	2.3200	4.2313	7.0711	Y
0.023	3.1983	2.3680	3.1983	6.5938	N

Table 1. *Predicted and computed roots of f(P). The final column indicates whether or not we trust the computational results.*

Based on the values of **Table 1**, we see that we cannot always trust the results of computational approximations. In this case, if we were to trust the value of our approximation, we would see that our approximation of the root for $\varepsilon = 0.023$ would have grossly overestimated P_{small} and P_{big}. This error could have had major consequences had these results been used in a real-world engineering project. Thus, the lesson to be learned is that we must always strive to critically analyze and verify the results of our computations.

TASK 6

The code from Task 4 is modified to solve equation (6). The modified code is first run with the values $\varepsilon = 0.020$, $pinit = 4.25$, $tfin = 100$, and the default tolerance of 1e-3. Based on the results of the critical point analysis in Task 5, we would expect a smooth curve, as the critical points were both shown to be stable. The figure created with these values looks suspicious as the plots are very jagged, not smooth.

Absolute error and relative error are then changed to 1e-4 and 1e-12 using the function *odeset*. The three results are then plotted on the same graph, as shown in **Figure 3**.

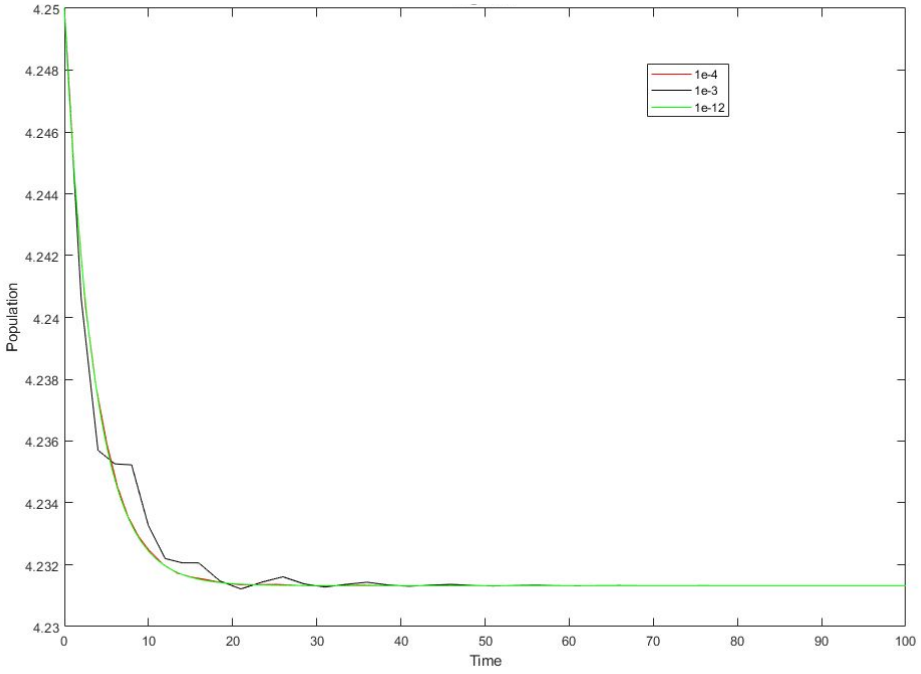


Figure 3. *Plot of the population using the three different tolerances, 1e-4 (red), 1e-3 (black), and 1e-12 (green).*

The results from using $tol = 1e-4$ is more reliable than that of $tol = 1e-3$, the default tolerance, because of the more stable curve, thereby having more accurate results.

The variable *count* is then introduced in order to keep track of how many times the *ode45* function calls *yprime.m*. The values for each tolerance are shown in **Table 3**. The final value of *count* is the number of times that the function called *yprime.m* since the function stops running once the results are within the given tolerance. The smaller the tolerance, the more times the function will run, thus resulting in a higher count.

Tolerance	**Count**
1e-3(default)	73
1e-4	79
1e-12	835

Table 2. *Computation work for various tolerances.*

We see that one should not blindly accept the results of approximation without first analyzing the results further, since the initial results are not always going to be the most accurate.

We then analyze the nonautonomous equation:

$$\varepsilon = \varepsilon_0 + epsamp * sin(\omega t),$$

where ε is dependent on time. This is done by introducing the new global variables ε_0, $epsamp$, and ω into the code used for the autonomous population model. The code is run for four values of tolerance which are all plotted on **Figure 4a.**

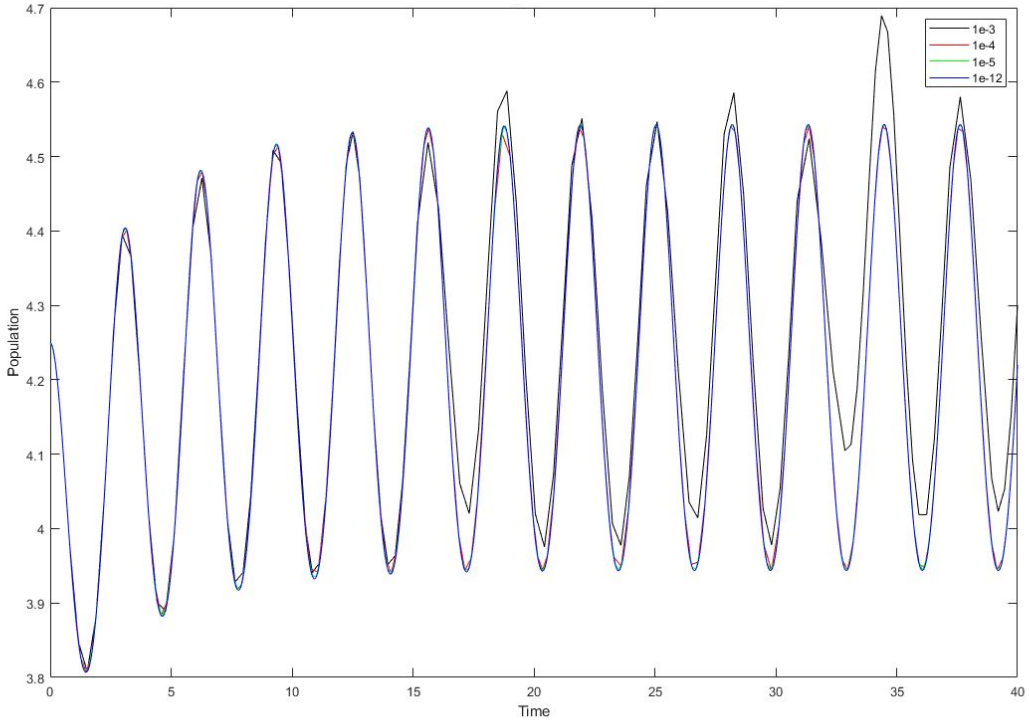


Figure 4a. *Plot of the nonautonomous model using the four different tolerances.*

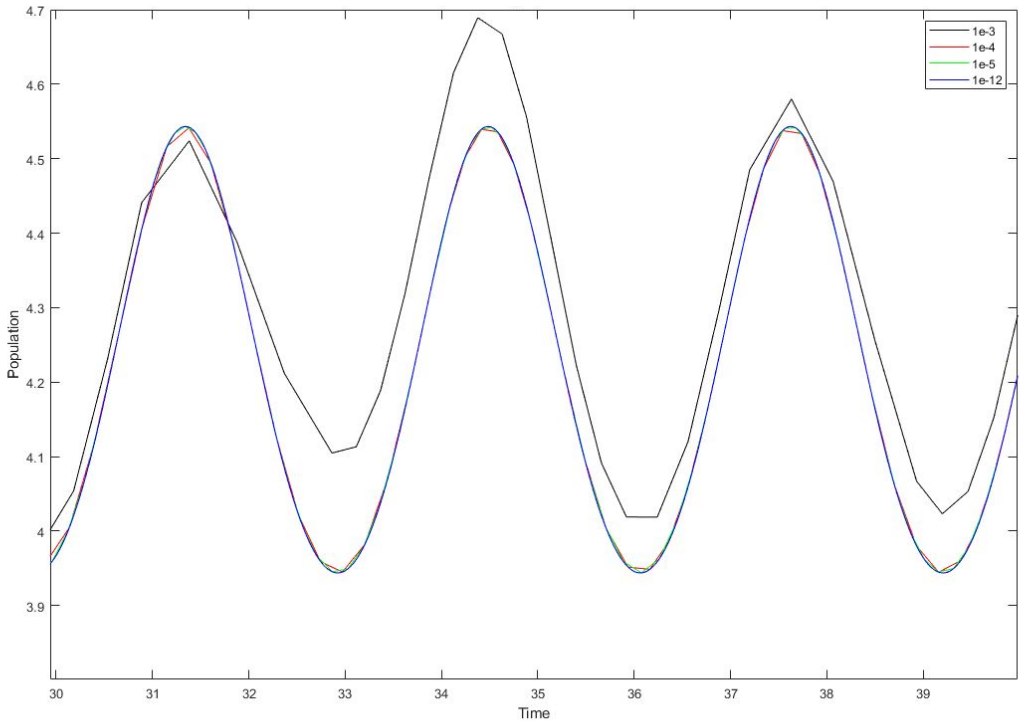


Figure 4b. *Plot of the nonautonomous model using the four different tolerances zoomed in from 30 to 40 seconds in order to better see the differences between the results.*

As seen in **Figure 4a** and **Figure 4b**, the default tolerance of 1e-3 is unacceptable because it mostly overestimates the population and is very unstable. Further, while the results using the tolerance of 1e-4 are much more accurate qualitatively than that of the default tolerance because of its increase in stability, the results are still not as stable as possible and can be improved.

CONCLUSION

We used Matlab throughout this project. In task 4 we used it to integrate, solve, and plot our ordinary differential equation (3) using the *ode45* solver. We used Matlab as a graphical and computational tool in task 5. With *fsolve*, we assessed the presence and stability of critical points through theoretical methods (i.e., approximation with the small parameter ε). We saw that we mustn't blindly trust the results of these computations in task 5, but instead scrutinize the results of our computer's approximations. We were reminded of this lesson in task 6 when we had to improve the function's performance by adjusting its default tolerance from 1e-3 to 1e-4 and1e-12.

All of this work contributed to our understanding of population models. We learned that the initial population size and type of a species can dramatically impact its growth and possibly

dictate its demise. We also took into account more nuanced ecological factors, such as predator-prey interactions and the crowding effect.

Finally, this lab afforded us the opportunity to work in a group. With this opportunity came the challenge of effective communication, equitable delegation of tasks, and time management.

Appendix A:

```
%% yprime(odefcn) for Task 4

function dPdt= odefcn(t,P,B) %odefcn is the same as yprime(it was named wrong initially)

    dPdt= P*(-1+(P/(1+(B*P))))

End

pinit=1.999 %Initial condition

tfin= [0 30]; %time interval from 0 to 30

B=.5; %global variable B

[t,P]=ode45(@(t,P)odefcn(t,P,B),tfin,pinit); %ode45 solves ode defined by odefcn(referred to as
yprime in Lab)

plot(t,P,'-r') %plots the function above

hold on %ensures both functions are plotted on the same figure

pinit= 2.001  %Second initial condition

tfin=[0 15] %time interval from 0 to 10

[t,P]=ode45(@(t,P)odefcn(t,P,B),tfin,pinit); %solves odefcn with new set of initial condition and
timer interval

plot(t,P,'-g') %plots the function above

xlabel(' Time') %defines the x axis title

ylabel('Population') %defines the y axis title

title('Population over Time') %defines the title of the figure

ln.LineWidth = 2; %changes the linewidth of functions

legend({'y0=1.999','y0=2.001'},'Location','northeast')

hold off
```

Appendix B: Code for Task 6 Part 1

Code for yprime.m

```
function dPdt= yprime(t,P,B,ep)
%Set up function formula
dPdt= P*(-1+(P/(1+(B*P)))-(ep*P^2))
End
```

Code for task6.m

```
%Set up initial conditions
pinit=4.25;
tbeg = 0;
tfin= 100;
B=.5;
ep=0.020;
tol = _;  %Set tolerance
opts = odeset('RelTol',tol,'AbsTol',tol);
[t,P]=ode45(@(t,P)yprime(t,P,B,ep),[tbeg,tfin],pinit,opts); %Run numerical method
plot(t,P); %Plot solution
title('_'); %Label graph
ylabel('Population')
xlabel('Time');
```

Appendix C: Code for Task 6 Part 2

yprimeosc.m

```
function dPdt= yprimeosc(t,P,B,ep)
%Setting global variables
global count
global ep0
global epsamp
global om
%Defining ep
ep=ep0+epsamp*sin(om*t);
dPdt= P*(-1+(P/(1+(B*P)))-(ep*P^2));%Set up function formula
end
```

task6osc.m

```
%Set up initial conditions:
pinit=4.25;
tbeg = 0;
tfin= 40;
B=.5;
ep=0.020;
global ep0
ep0=0.020;
global epsamp
```

```
epsamp=0.008;
global om
om=2;
%Set tolerance:
tol = _;
opts = odeset('RelTol',tol,'AbsTol',tol);
[t,P]=ode45(@(t,P)yprimeosc(t,P,B,ep),[tbeg,tfin],pinit,opts); %Run numerical method
plot(t,P,'b'); %Plot solution
hold on;
title('Figure _');
%Label and modify graph
ylim([3.8 4.7])
ylabel('Population')
xlabel('Time');
```

Appendix D: Code for Task 5

```
%% Task 5 -- Alexander Ross


% Question 2


% Initialize the parameters & variables:

P = 2:0.0001:6;

B = 0.5;

E1 = 0.015;

E2 = 0.02;

E3 = 0.023;


% Initialize the function f(P) for 3 different values of epsilon:

f1 = (-B.*E1.*(P).^3 - E1.*(P).^2 - B.*P + P - 1);

f2 = (-B.*E2.*(P).^3 - E2.*(P).^2 - B.*P + P - 1);

f3 = (-B.*E3.*(P).^3 - E3.*(P).^2 - B.*P + P - 1);


% Plotting the figure:

figure();

hold on

grid on

legend();

plot(P,f1,'color','k','DisplayName','E1');

plot(P,f2,'color','r','DisplayName','E2');
```

```
plot(P,f3,'color','g','DisplayName','E3');

yline(0,'--b','DisplayName','Z(P)');

axis([2 6 -0.05 0.3]);

xlabel('P');

ylabel('f(P)');

hold on


% Question 4


% Initialize the parameters & variables:

P = 2:0.0001:6;

B = 0.5;

E1 = 0.015;

E2 = 0.02;

E3 = 0.023;


% Initialize the approximate values of Psmall and Pbig:

Papproxsmall1 = 2 + 16*E1;

Papproxsmall2 = 2 + 16*E2;

Papproxsmall3 = 2 + 16*E3;

Papproxbig1 = 1/sqrt(E1);

Papproxbig2 = 1/sqrt(E2);

Papproxbig3 = 1/sqrt(E3);
```

```
% Solve for Psmall and Pbig using 'fsolve':
Ps1 = fsolve(@(x)-E1*B*x^3 - E1*x^2 + (1-B)*x -1,Papproxsmall1);
Ps2 = fsolve(@(x)-E2*B*x^3 - E2*x^2 + (1-B)*x -1,Papproxsmall2);
Ps3 = fsolve(@(x)-E3*B*x^3 - E3*x^2 + (1-B)*x -1,Papproxsmall3);
Pb1 = fsolve(@(x)-E1*B*x^3 - E1*x^2 + (1-B)*x -1,Papproxbig1);
Pb2 = fsolve(@(x)-E2*B*x^3 - E2*x^2 + (1-B)*x -1,Papproxbig2);
Pb3 = fsolve(@(x)-E3*B*x^3 - E3*x^2 + (1-B)*x -1,Papproxbig3);
```