Thinh Nguyen

Dr Olenick

Computational Physics

12/7/2022

Using the 4th order Runge-Kutta algorithm to analyse the Lotka-Volterra equation on predator and prey populations model.

Abstract: In this research project we take a look at the Lotka-Volterra model, a population prediction model. This model predicts the dynamics of biological systems in which two species interact, one is the predator and one is the prey. We used the Runge-Kutta 4th order algorithm to simulate this model. The Runge-Kutta 4th order algorithm is the most efficient method to solve ordinary differential equations. We used this algorithm to simulate two populations (tigers and deers) based on the Lotka-Volterra model, and traced their behaviour over time. We also optimised the parameters that go with the model to get results that are as close to real data as possible.

Introduction: The Lotka-Volterra equations, also known as predator-prey equations, are a pair of first order nonlinear differential equations, used to describe the dynamics of biological systems, where 2 species interact, 1 as the predator and 1 as the prey. The population changes according to the pair of equations:

$$\frac{dx}{dt} = \alpha x - \beta xy,$$
$$\frac{dy}{dt} = \delta xy - \gamma y,$$

There are 6 parameters that come with the model: α, β, x, y, δ and γ. The populations of each species are denoted using parameters x and y, with x is the population of the prey, and y is the population of the predator. The rates in which each species reproduces and dies are denoted with parameters α, β, δ and γ, with α is the natural birth rate of the prey and β is the death rate of the predator in the presence of the predator. Δ is the birth rate of the predator in the presence of the prey and γ is the natural death rate of the predator.

Results: We have successfully simulated the model using the Runge-Kutta 4th order and obtained various graphs with different changes in the parameters. With the hypothetical predator and prey system of tiger and deer in the same environment, starting with 100 deers and 40 tiger, $\alpha = 0.5$, $\beta = 0.01$, $\delta = 0.01$ and $\gamma = 0.5$, we've received this graph:

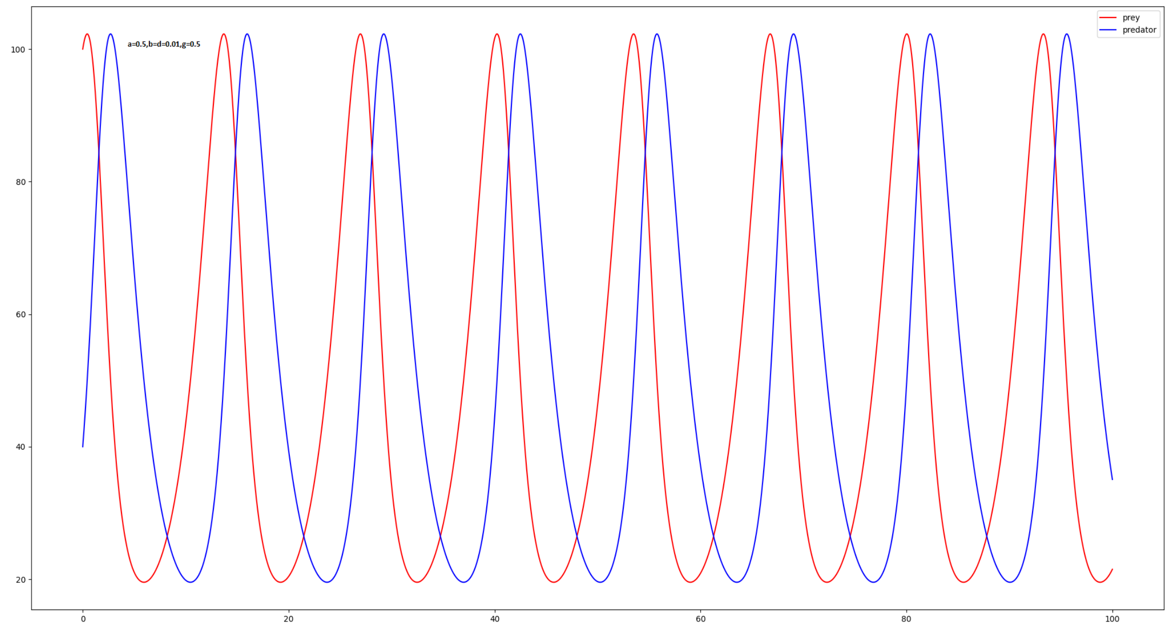

Figure 1. Lotka-Volterra model simulated over 100 days with deer initial population = 100 and tiger initial population = 40

At the beginning, the population of tigers shot up almost exponentially, because the number of deers were greatly outnumbered the number of tigers. This corresponds to the term δxy which increases the number of tigers based on the number of deers, because the tigers' only food source is the deers available. This also explains why at the same time the tiger population increases, the deer population also greatly decreases, yielding the βxy term, which negatively affects the number deers based on how many tigers there are. The terms γy and αx yields the rates at which the tigers die due to the scarcity of food and competition, and at the same time, the rates at which the deers reproduce due to the lack of tigers. Then as the number of deers increases, we go back to the beginning, the tiger population increases right after, making a cycle of both population increasing and decreasing. In fact, with any difference in population, the graph will always follow the same pattern of oscillating up and down and both reach the same maximum capacity. Figure 2 below, for example:

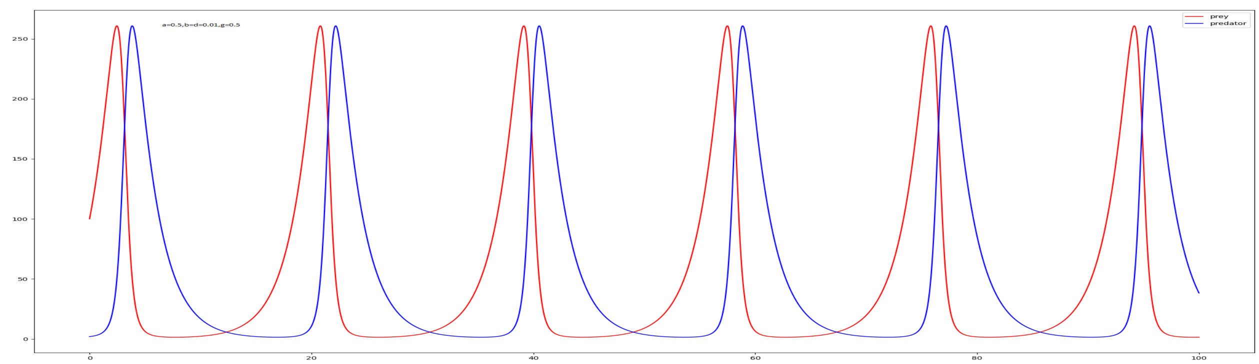

Figure 2.1. Lotka-Volterra model simulated with tiger initial population = 2

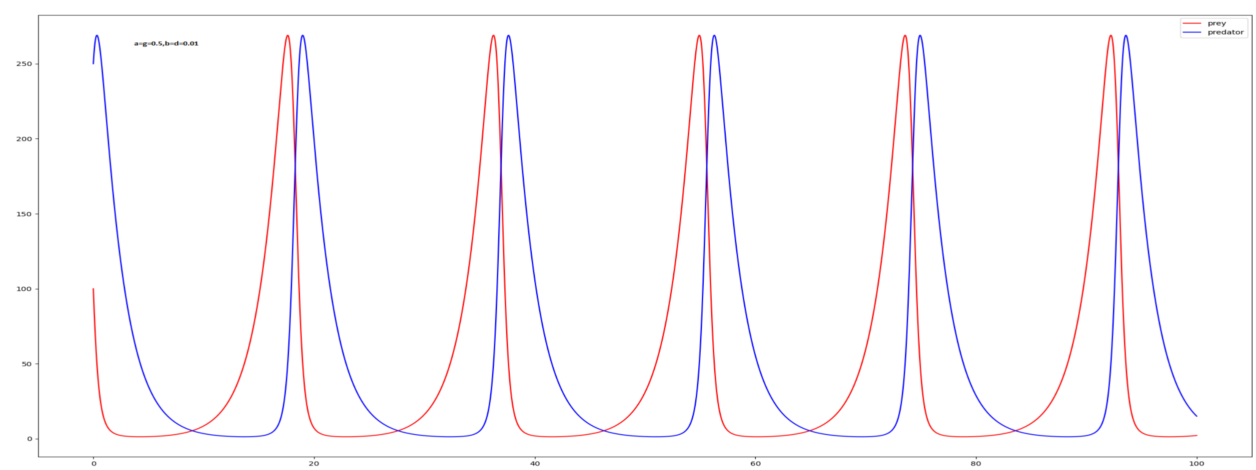

Figure 2.2. Lotka-Volterra model simulated with tiger initial population = 250

As mentioned above, the graphs are almost identical in behaviour. After the initial difference, the two graphs, once again, follow an identical pattern.

We then now changed some other parameters to see if there is any change in the graph behaviour. This time, we will change the rate at which the deers reproduce in the absence of tigers α. Instead of α = 0.5, we will simulate the model using α = 1 and α = 1.5. Keep in mind with these changes in parameters, the initial populations for the deers and tigers will always stay consistent at 100 and 40, respectively, and the time simulated will always be 100 days. Figure 3 below are the results:

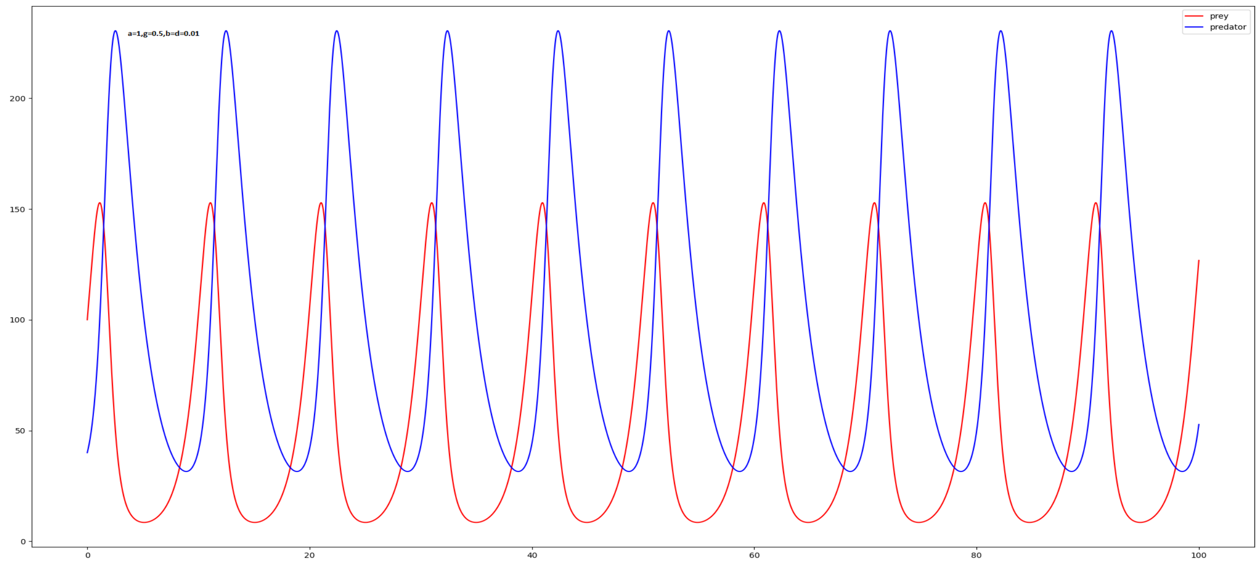

Figure 3.1. Lotka-Volterra model simulated with α = 1

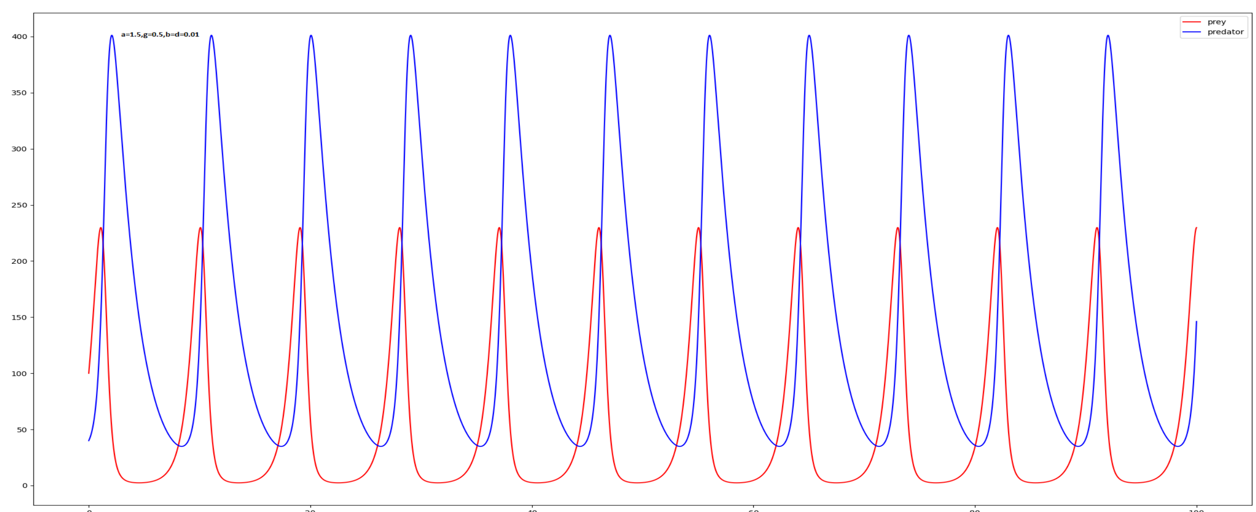

Figure 3.2. Lotka-Volterra model simulated with α = 1.5

From these graphs, it can be seen that with a bigger value of α, we have the deer maximum population higher as well. Since there are now more deers to eat, the tiger population also reaches a higher maximum capacity. Follow up is to change the rate at which the tiger population decreases due to natural causes, γ. We now have γ = 0.7 and γ = 0.8, and Figure 4 below shows the results:

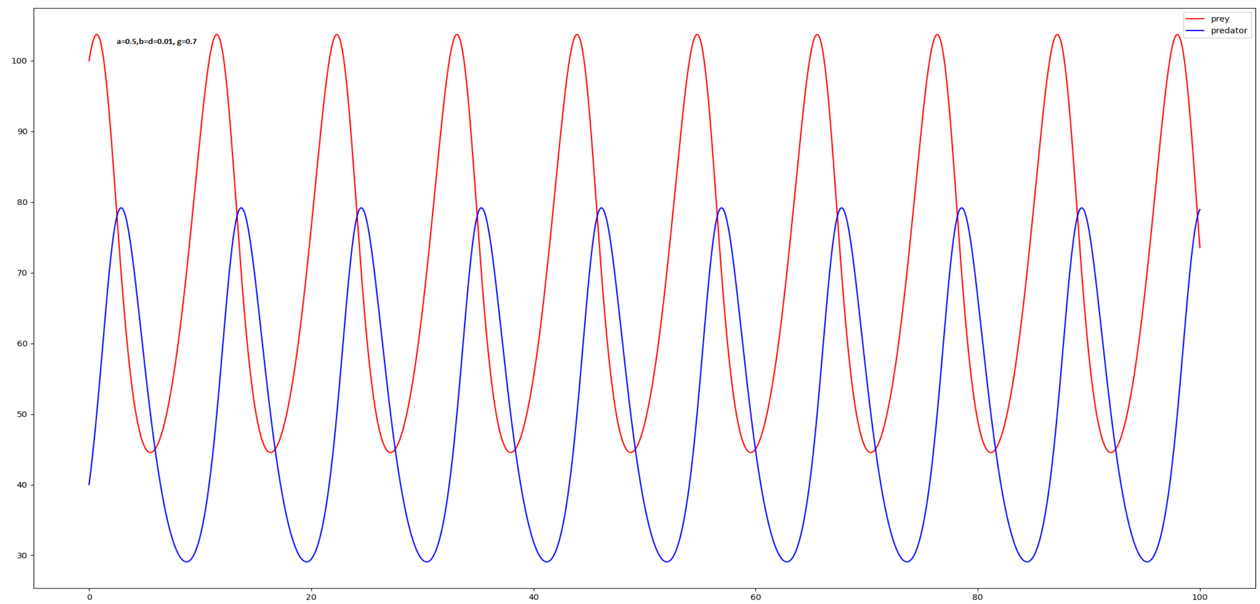

Figure 4.1. Lotka-Volterra model simulated with γ = 0.7

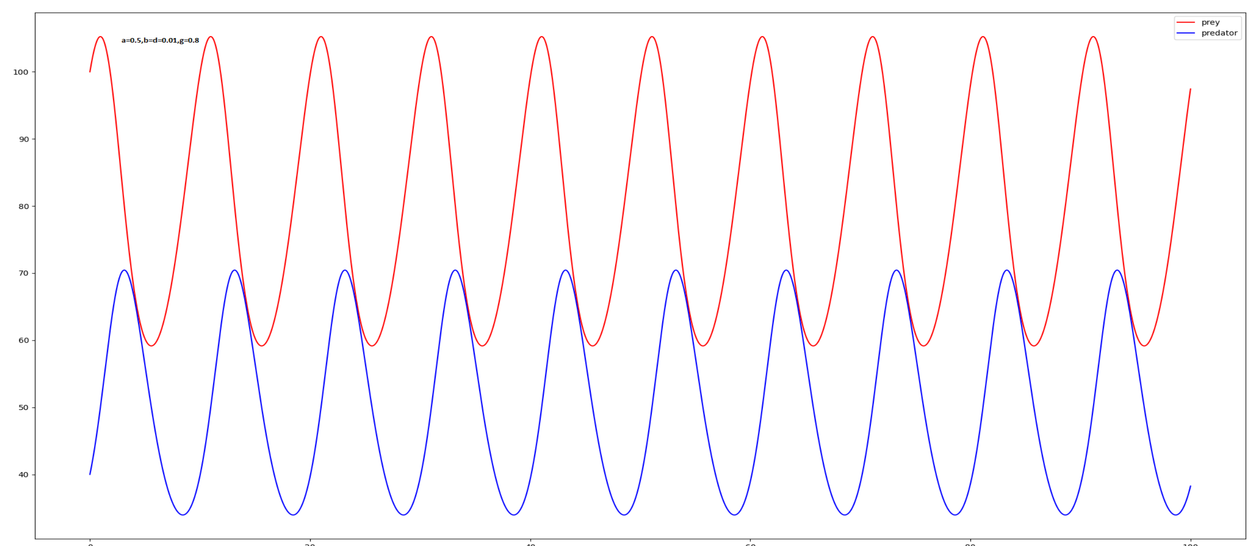

Figure 4.2. Lotka-Volterra model simulated with γ = 0.8

The graphs of the deer and tiger populations are now “separated”—the higher the γ parameter, the bigger the “separation” is. This is due to the fact that the death rate of the tigers is now higher, leading to the maximum population of the tiger being substantially lower than before. The bigger the γ parameter is, the lower the maximum capacity of the tigers is going to be.

Next up, we increase the β parameter—the death rate of the deers due to natural causes and the tigers. Instead of $\beta = 0.01$, we now have $\beta = 0.02$ and $\beta = 0.04$. Figure 5 below shows the results:

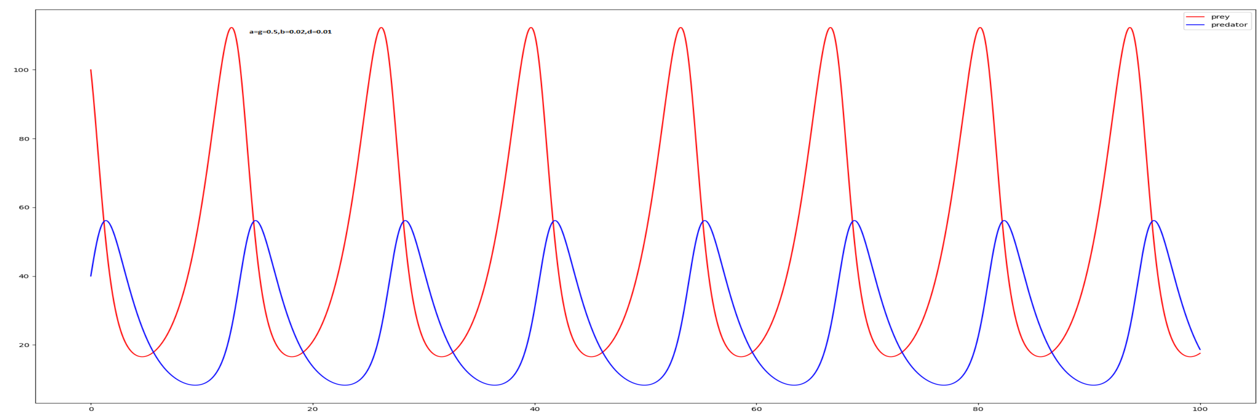

Figure 5.1. Lotka-Volterra model simulated with $\beta = 0.04$

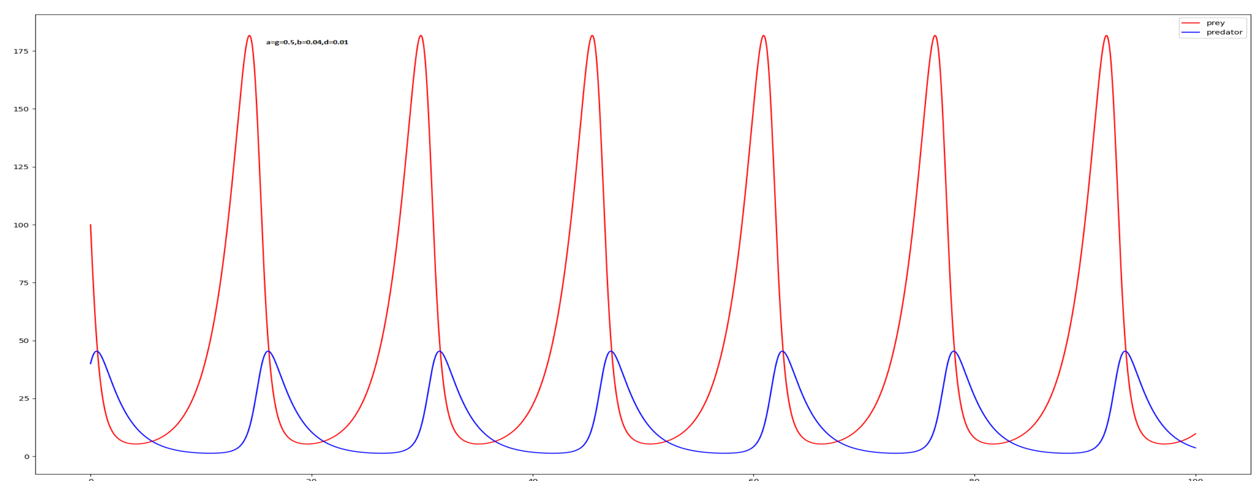

Figure 5.2. Lotka-Volterra model with $\beta = 0.02$

From the graphs, we can see that with a higher value for β, it takes less tigers to hunt the deers, because of the term βxy decreasing the population of the deers. It's also worth noting that the higher β is, the lower the maximum capacity of the deers is also going to be, and this makes sense, because β is the deers' death rate.

Now, we change the parameter δ. We are now going to increase δ from δ = 0.01 to δ = 0.02 and δ = 0.04. Figure 6 below shows the results:

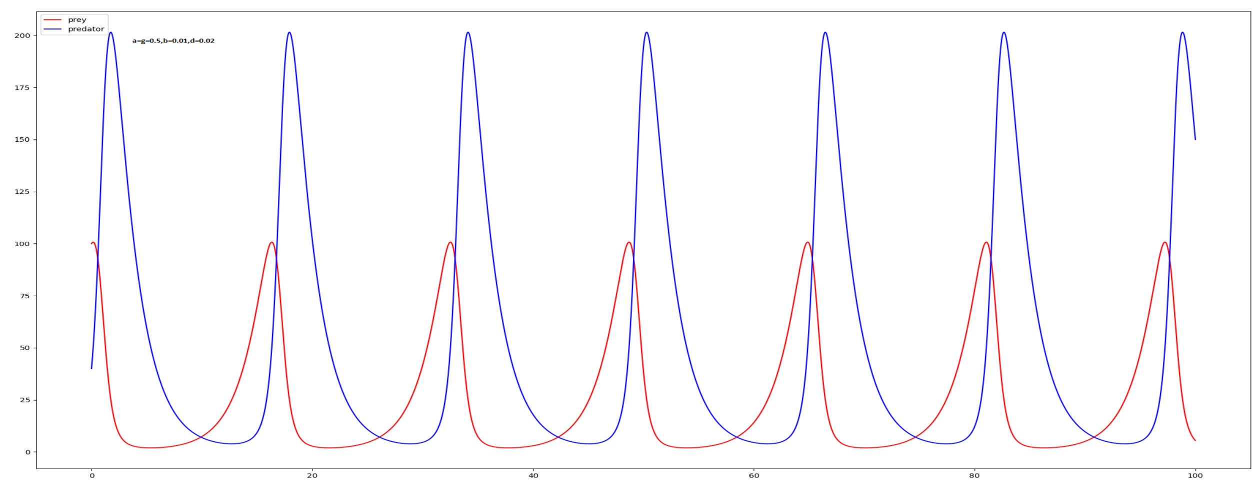

Figure 6.1. Lotka-Volterra simulated model with δ = 0.02

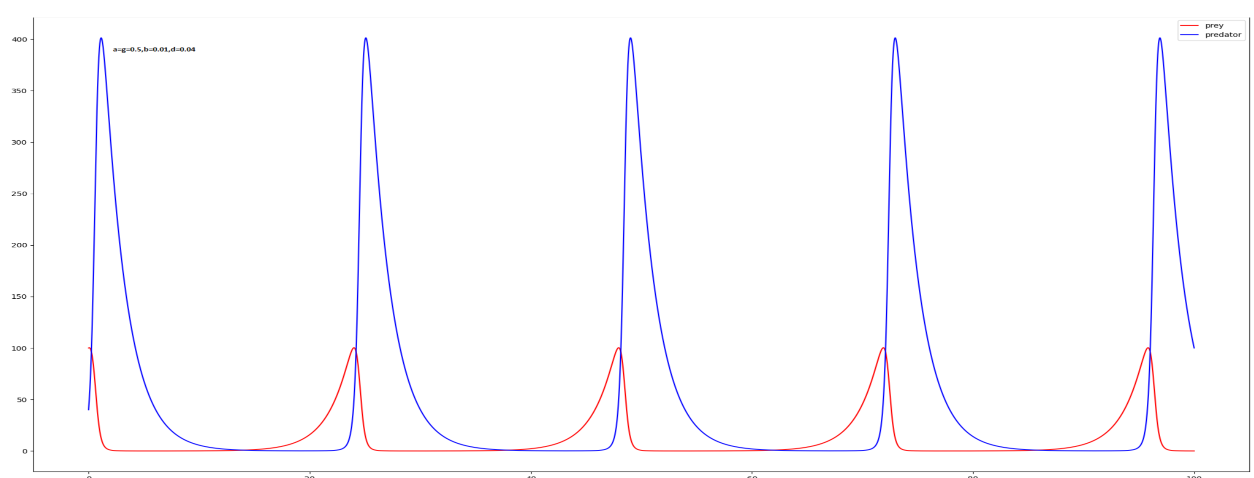

Figure 6.2. Lotka-Volterra simulated model with δ = 0.04

With δ increasing, it takes less deers to increase the population of tigers, yielding the δxy term which positively affects the population of tigers.

The following demonstrates what happens when we simultaneously change α and γ. Figure 7 below shows the changes when we increase both γ and α to 1 instead of 0.5 and also decreases both of them to 0.25:

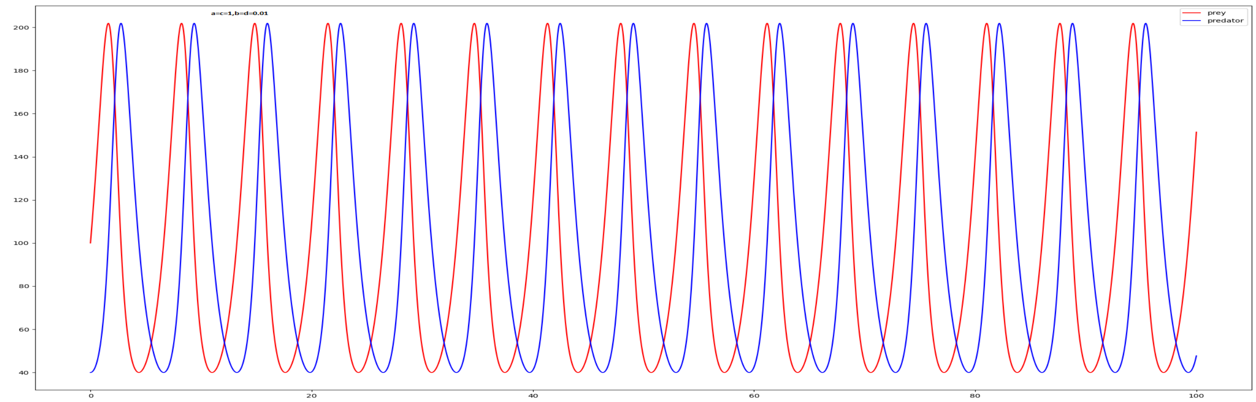

Figure 7.1. Lotka-Volterra model simulated with α = γ = 1

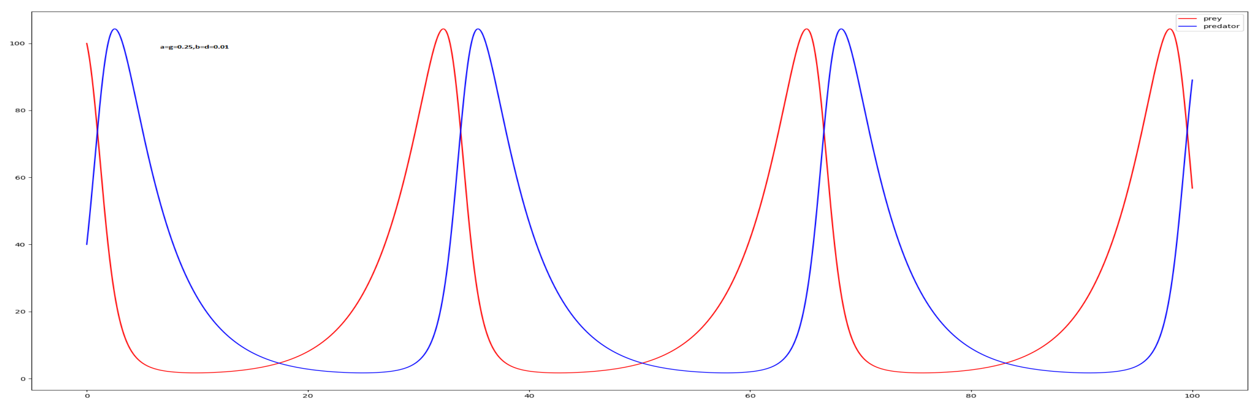

Figure 7.2. Lotka-Volterra model simulated with α = γ = 0.25

Here we have an interesting effect on the population cycle when we change both the α and γ parameters. Increasing α and γ also increases the period in which both populations rise and drop. The opposite is also true, and the answer for this phenomenon is that since we increase the birth rate of the deers and at the same time increase the death rate of the tigers, more deers are being consumed by the tigers and also more deers are being born due to the absence of tigers, and that is why we get such fast population growth and decay period. The opposite is also true when we decrease the birth rate of the deer population and decrease the death rate of the tiger population.

The next thing to look at is what will happen if we change both α and β. Figure 8 below shows us the effects:

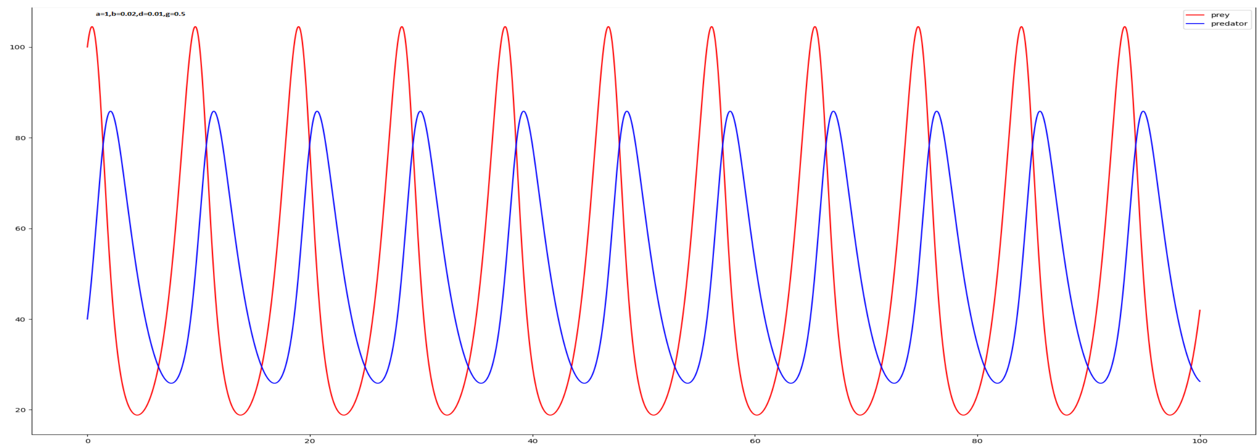

Figure 8.1. Lotka-Volterra simulated model with α = 1, β = 0.02

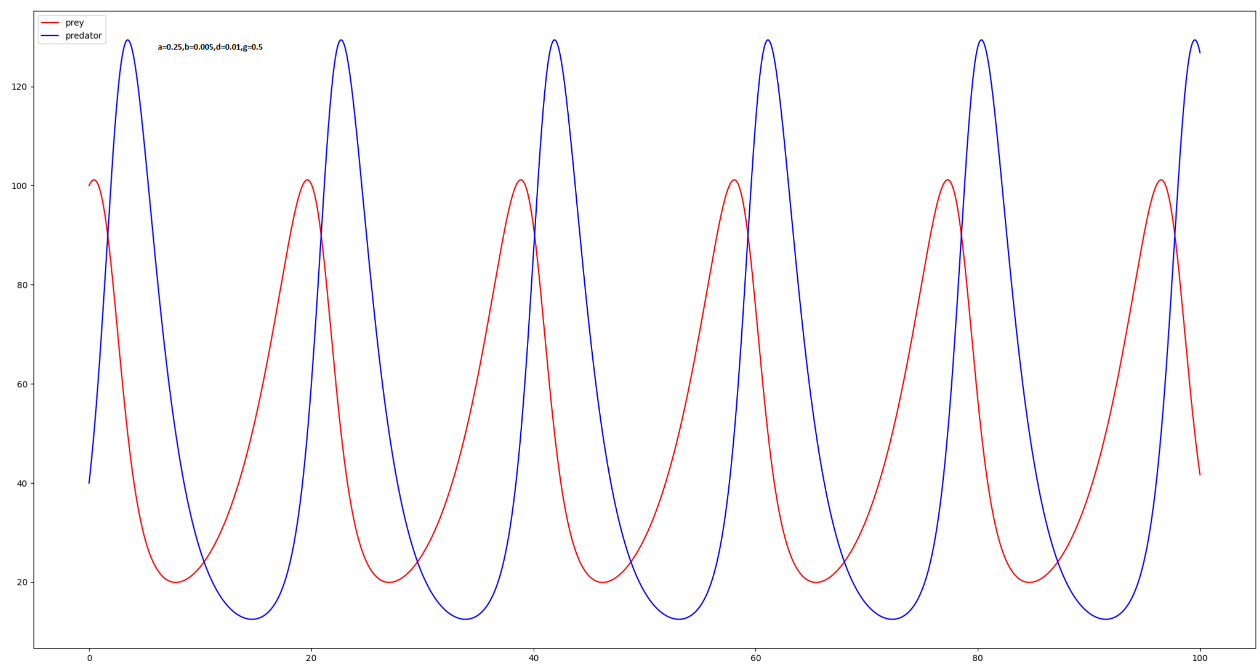

Figure 8.2. Lotka-Volterra simulated model with α = 0.25, β = 0.005

Here, increasing both α and β gives us a higher maximum capacity for the deer population, and decreasing both α and β gives us, logically, much lower increases in the deer population's max capacity. Logical, since α positively affects the birth rate of the deer population, and β negatively affects the deer population. It is also worth noting that the death rate has a bigger effect than the birth rate for the deer population, because the population's decay also is being affected by the amount of predators in the environment, and the opposite is true for the tiger population—more deers, more tigers, less deers, less tigers. This is shown in figure 9, where we change both δ and γ, increasing both of them and then decreasing both of those parameters:

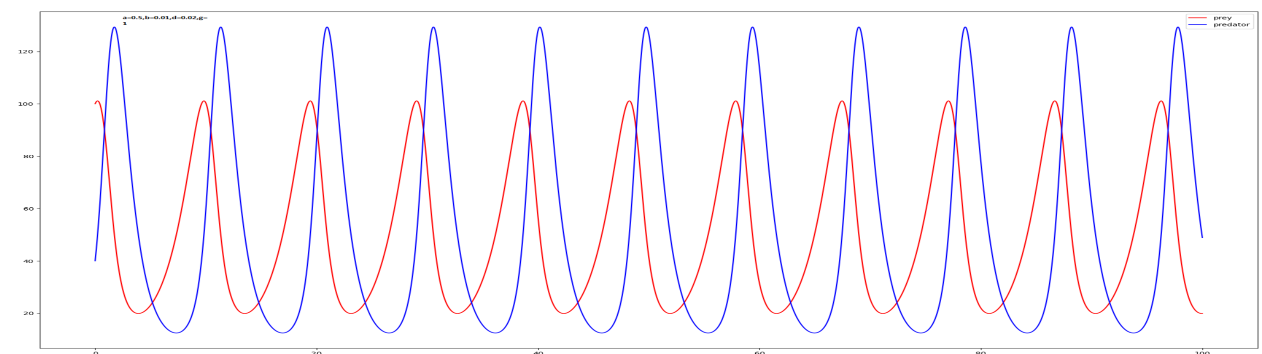

Figure 9.1. Lotka-Volterra model simulated with δ = 0.02 and γ = 1

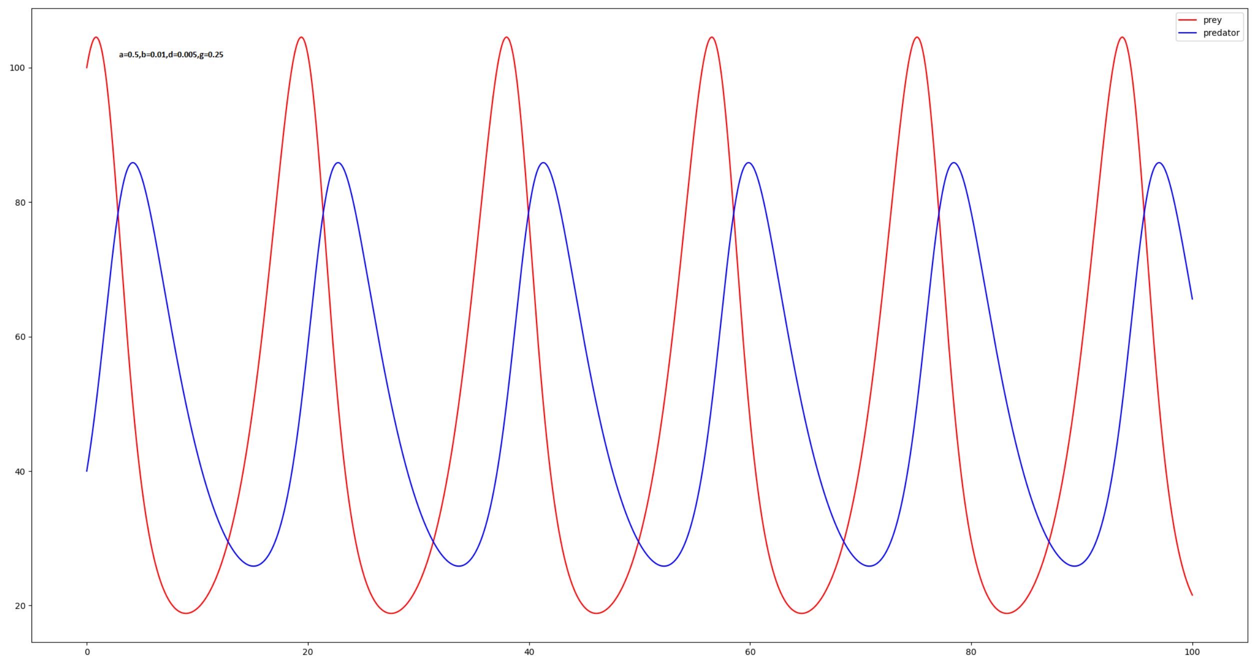

Figure 9.2. Lotka-Volterra model simulated with δ = 0.005 and γ = 0.25

Lastly, Figure 10 demonstrates the phase space diagram for this model.

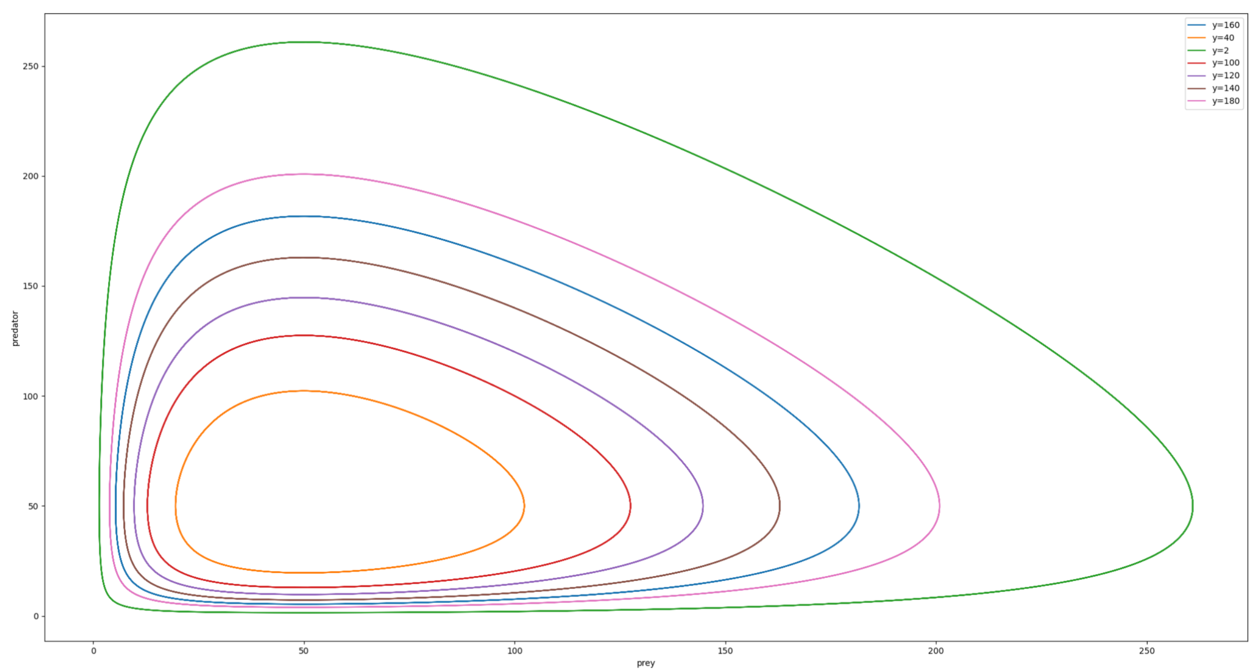

Figure 10. Phase diagram for the Lotka-Volterra model with different initial predator populations

The phase diagram for the model creates a loop because the graphs of the populations over time resembles a sine graph with fixed periods and fixed amplitude.

Discussion: The Lotka-Volterra model is the simplest model of predator-prey interactions, therefore, it does not take into account realistic events bound to happen when two species interact such as the predator stalking, chasing and the time it takes for each species to actually make contact, and thus, it does not produce a very accurate depiction of the realistic dynamics. However, it is still a pretty useful model to predict the dynamics of the population for each species because it does a perfect job in predicting the dynamics of the population between two species with perfect conditions: every event is instantaneous and no natural calamities such as diseases. The graph shown in Figure 11 even though is definitely not the same as the graphs predicting the population trends using the Lotka-Volterra model, it still follows the same dynamics pattern as the graphs using the Lotka-Volterra model.

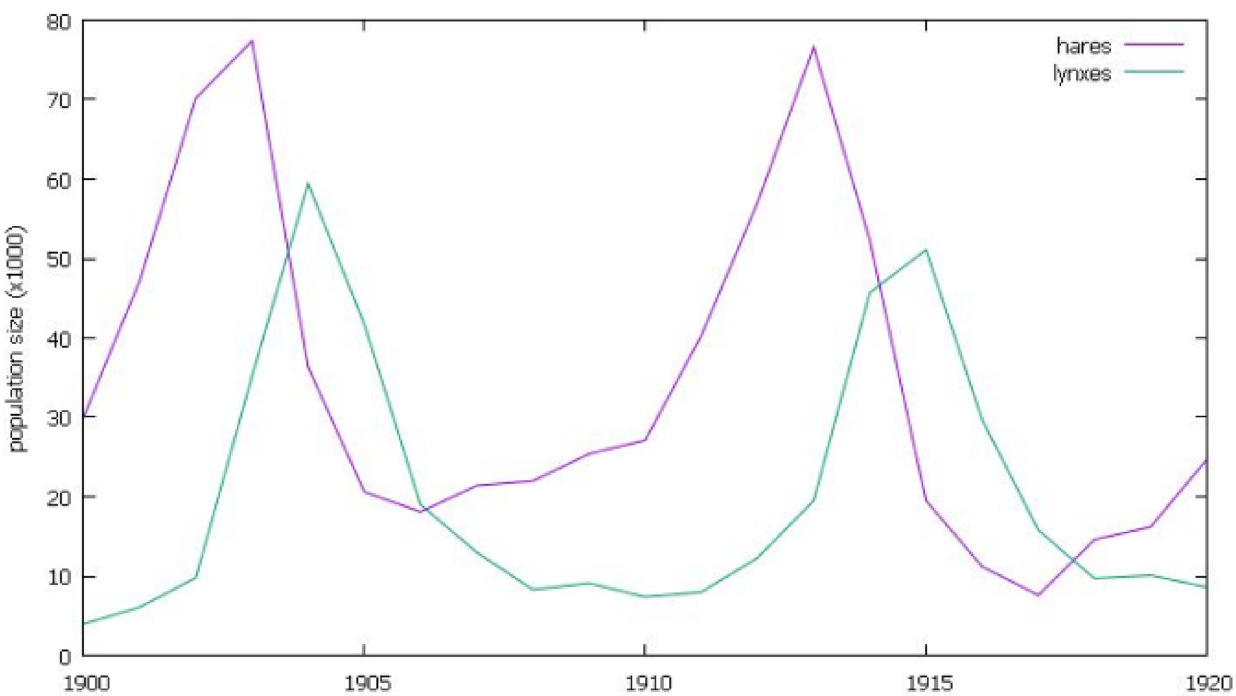

Figure 11. Actual population graph of lynx and snowshoe hares from the Hudson Bay Company, Canada over the period 1900-1920 (Gilpin, 1973).

Conclusion: In this project we have demonstrated that complex processes and phenomena in nature can be modelled using simple mathematical models. Along with these simple mathematical models, using appropriate parameters for such models also yields success in studying these complex systems and phenomena. In this project, we looked at two populations in which fit the predator and prey model (tigers and deers), and by using the Lotka-Volterra predator-prey model with the Runge-Kutta 4th order algorithm, we have successfully simulated the model, which was confirmed by comparing the simulated model to the actual obtained data from the Hudson Bay Company. Even though the two models are not entirely similar, they still follow the same graph pattern consistently and therefore, it would still beneficial for us to use the Lotka-Volterra model to predict a predator and prey system's population size, under consideration that the environment they live in is perfect for their reproduction and their contacts are consistent, instantaneous and happen frequently.

References:

(1). Bing Liu, Yujuan Zhang, Lansun Chen, "The dynamical behaviours of a Lotka–Volterra predator–prey model concerning integrated pest management", *Nonlinear Analysis: Real World Applications, Volume 6, Issue 2,* 2005

(2). Kasim Hussain, Fudziah Ismail, Norazak Senu, "Runge-Kutta Type Methods for Directly Solving Special Fourth-Order Ordinary Differential Equations", *Mathematical Problems in Engineering*, vol. 2015, Article ID 893763, 11 pages, 2015. https://doi.org/10.1155/2015/893763

(3). Kunis, F., & Dimitrov, M. (2020). Investigating Lotka-Volterra model using computer simulation. Open Schools Journal for Open Science, 3(10).

Appendix: This is the snippet of the code used to simulate the model and generate different graphs.

```
#lotka_volterra.py
#using RK4 to solve the lotka_volterra equations and plot
#2 populations:

import numpy as np
import matplotlib.pyplot as plt
#init parameters

alpha = 0.5; #deer birth rate (prey)
beta = 0.01; #deer death rate (prey)
x = 100; #deer population (prey)
y = 40; #tiger population (predator)
delta = 0.01; #tiger birth rate (predator)
gamma = 0.5; #tiger death rate (predator)
t = 0; #time
h = 0.01; #time step
pc = []
pc.append(x)
Pc = []
Pc.append(y)
tc = []
tc.append(t)
#prey population function
def prey(x, y):
    return alpha*x - beta*x*y

#predator population function
def predator(x, y):
    return delta*x*y - gamma*y


while t <= 100:


    k1p = prey(x, y)*h
    k1P = predator(x, y)*h

    k2p = prey(x + k1p/2, y + k1P/2)*h
    k2P = predator(x + k1p/2, y + k1P/2)*h

    k3p = prey(x + k2p/2, y + k2P/2)*h
    k3P = predator(x + k2p/2, y + k2P/2)*h

    k4p = prey(x+ k3p, y+ k3P)*h
    k4P = predator(x+ k3p, y+ k3P)*h

    x += (k1p + 2*k2p + 2*k3p + k4p)/6
    y += (k1P + 2*k2P + 2*k3P + k4P)/6
    pc = np.append(pc, x)
    Pc = np.append(Pc, y)
    t+=h
    tc= np.append(tc, t)
```