

Laboratory 3

Solving Continuous Time Single State Variable Models

Introduction

This laboratory exercise will provide you with experience in constructing, solving, and analyzing a single compartment (state-variable) model. This laboratory will focus on the exponential growth model. Through this exercise, you will learn to encode and solve the discrete and continuous time version of the model. To solve the continuous time version, you will use compare the exact analytical solution as well as two numerical approximation techniques: the Euler method and *lsoda*. In addition, you will further develop your scientific programming skills in R.

Please report your findings in the form of a short report in which you present evidence that you accomplished each of the modeling tasks listed. Please make sure to address any questions asked and include any figures or tables necessary. Your short report will be due at our next lab meeting.

Discrete vs. Continuous Time Equations

According to Gurney and Nisbet (1988), “A dynamic model is a mathematical statement of the rules governing change”. We can then distinguish two types of mathematical statements: an update rule that uses discrete time and a differential equation that uses continuous time.

An **update rule** describes “the relationship between the current and future state of the system”. Let N_t be the current state of the system and $N_{t+\Delta t}$ be the state after the discrete interval of time Δt . We can then write the generic update rule as

$$N_{t+\Delta t} = f(N_t). \quad (1)$$

This type of equation is referred to as a **difference equation**. Equations of this form are particularly suited for projecting the state of the system at regular intervals and are often used to model species that reproduce with non-overlapping generations and simultaneous reproduction, like many semelparous species. Mature adults of these organisms reproduce once and then die. For example, the checkerspot butterfly *Euphudrias editha* breeds once per year. Adults fly for a short period of time, lay their eggs near April 1, and die (Hastings 1997).

A **differential equation** specifies the rate of change of a state variable N with respect to a second variable like time t and represents a continuous change. The equations are formally defined as

$$\frac{dN}{dt} = \lim_{\Delta t \rightarrow 0} \frac{N_{t+\Delta t} - N_t}{\Delta t} \quad (2)$$

Exponential Growth Population Model

In class, we discussed both the difference equation form and differential equation for of the exponential population growth model. The difference model assumes that there are non-overlapping generations, while the continuous differential model assumes births and deaths happen all the time.

Discrete Difference Model

Recall that we can predict the future population size with the following discrete time difference model.

$$N_{t+1} = N_t + r_d N_t \quad (3)$$

Where N_t is the population size now, N_{t+1} is the population size one time step into the future, and r_d is the discrete growth factor. When we factor like terms we get

$$N_{t+1} = N_t(1 + r_d)$$

and then we can let $1 + r_d = \lambda$ so that

$$N_{t+1} = \lambda N_t.$$

Note $\lambda \geq 0$ by definition.

Gotelli (2008, p. 12) showed that we can use this equation recursively to predict future population size some number of time steps into the future using

$$N_t = \lambda^t N_0 \quad (4)$$

where N_0 is the initial population size and t is the selected future time step.

Continuous Differential Model

$$\frac{dN}{dt} = rN \quad (5)$$

$$= (b - d)N \quad (6)$$

Where N is the population density (number of individuals per area), $r = (b - d)$ is the intrinsic rate of growth, b is the population specific birth rate, and d is the population specific death rate.

When considering differential equations like equation (2), Gurney and Nisbet (1998) suggest that “A model that uses derivatives is essentially an update rule model with very small time steps”. Differential equations are a bit more complicated to implement and use, but are often worth the investment. For example, they are better for modelling species with overlapping generations or variables that are affected continuously.

As Gotelli (2001) notes, the model in equation (2) tells us the instantaneous rate of population change, but it does not directly let us project the size of the population at some time in the future. To do this, we must integrate the model. We can use the rules of integral calculus to find that the exact analytical solution for the exponential growth model is

$$N_t = N_0 e^{rt}, \quad (7)$$

where N_t is the population density at time t , r is the intrinsic growth rate, and N_0 is the initial population density. Given N_0 , r , and t , we can now predict the future population size (making all the assumptions embedded in our model).

Numerical Approximation: Euler and lsoda Algorithms

In most cases, our models will not have an exact analytical integral solution. Therefore, we will need to use numerical approximation techniques to estimate the solutions. The reading introduced two methods: the Euler technique and the Runge-Kutta 2 technique. These are useful for learning about numerical approximation techniques, but we will usually use a more sophisticated algorithm called *lsoda*.

Euler Technique

Euler's technique is the simplest numerical method for approximating the solution to a differential equation. It is quick and easy, but it typically generates the greatest amount of error.

The method was described in detail in the reading. A quick summary of the technique is that we estimate the population size at a fixed time step from now Δt using the current population size N_t . We can summarize this as

$$N(t + \Delta t) = N(t) + (r * N(t)) * \Delta t \quad (8)$$

We would expect the relative error to decrease as Δt decreases, and of course this is the exact solution when $\Delta t \rightarrow 0$.

Error We can estimate the error between the exact solution and our estimated solution using the root mean sum of squared predicted (RMSEP) to compare fits. This is calculated as

$$RMSEP = \sqrt{\frac{1}{n} \sum_{i=1}^n (Np_i - Na_i)^2} \quad (9)$$

where Np_i and Na_i are the predicted and actual or true population values at i , and n is the number of predicted points.

Using lsoda

Most computer packages like R and MATLAB have more sophisticated ODE solvers built into the language. R uses a routine called *lsoda* that is included in the “deSolve” package. Once a package is installed on the machine, you will need to load it by typing “library(package-name)”.

Tasks

Task 3.1: Discrete Time Model Projections

Task 3.1.1 Write a script that uses equation (4) to project the future population size. Hint: this can be done with either a for-loop or more efficiently using vectorization.

Task 3.1.2 Show the effects of changing N_0 and λ . As the cognitive task is to compare the output of several models, we want to plot multiple solutions on the same graph. Create two plots. In the first plot, let $N_0 = 20$ and plot the solutions when λ takes values of $\{0, 0.5, 1, 1.5, 2\}$. In a second plot, let $\lambda = 1.25$ and plot the solutions when N_0 takes values of $\{1, 10, 50\}$. Let $t = \{0, 0.5, 1, 1.5, 2, 2.5, \dots, 25\}$.

Task 3.2: Diagram the continuous time model

Create a diagram of the continuous time exponential growth model using the Forrester symbols that we discussed in class. Make sure to clearly label each part.

Task 3.3: Plot Population Projections

Write an R script that uses the exact solution of the differential equation (equation (7)) to project the future population sizes as time goes from 0 to 100. Start with $N_0 = 5$ and $r = 0.1$. Plot the solutions when r is -0.1 , 0.1 , and 1 on the same graph. What happens when you change N_0 ?

Task 3.4: Numerical Approximation using Euler and lsoda

In this task, you will use two numerical approximation techniques to estimate the change in population. You will first use the Euler solution with a different time steps and then you will use an “industrial strength” algorithm called *lsoda*. Given the exact solution you coded in Task 4.3, you can then compare the amount of error the different algorithms generate.

The R package *deSolve* has an ODE solver function called *ode* that will easily allow us to implement the different algorithms (and many more!). The set up to use this function is a bit tricky, so you will start your work from an existing set of programs. After making sure that the *deSolve* package is installed on your computer, please download the following two files from the course website: [exponential.r](#) and [exponential-run.r](#).

Task 3.4.1: Numerical integration with Euler

First, read through the two programs and try to understand how they works. The “*exponential.r*” file contains a new function that encodes the model. The “*exponential-run.r*” file contains the code that defines the initial conditions, model parameters, and then calls the *ode* function, which requires the model function. Notice that the program sources the model function file. The run file also has the code to plot an start to analyze the results.

Second, execute the program using the default parameters. What happens? Notice that the *ode* function is initially set up to use the “*method=euler*” integration algorithm with a time step of $dt = 1$.

Next, change the model parameters to investigate how they alter the program function. Please make any changes you find interesting and useful to learn about the program. Finally, try alternative values of the time step (dt) of 1, 0.1, and 0.01, and compare the approximated population trajectories to the analytically exact solution using RMSEP. What happens as dt increases? Why? What would you expect to happen in a function with dynamics like the logistic equation? Please plot all 4 solutions on the same figure.

Task 3.4.2: Numerical Approximation using lsoda

For our final task, we will use the *lsoda* routine. This is actually the default algorithm for *ode*, so to implement it, we can remove the *method* and *hini* function parameters from the *ode* call. Using the same initial model parameterization as in Task 4.4.1, visually compare the *lsoda* solution to the exact solution and calculate the RMSEP. How does this compare to the euler solution?

References

- Ellner, S. P., and J. Guckenheimer. 2006 . An introduction to R for dynamic models in biology.
<http://www.cam.cornell.edu/~dmb/DynamicModelsLabsInR.pdf>
- Gurney, W. S. C., and R. M. Nisbet. 1998. Ecological dynamics. Oxford University Press, New York.
- Hastings, A. 1997. Population biology: concepts and models. Springer, New York