Solutions for Laboratory 3
Solving Continuous Time Models: Single State Variable Models—Exponential Growth

Bio 535, Fall 2011

Introduction

The primary objective of this laboratory was to build your experience with implementing, running, and exploring single state variable models. We used numerical approximation techniques to approximate the exact integral solution of continuous time models of exponential and logistic growth. We also found chaotic behavior in the Ricker model, a discrete time analogue of the logistic model. In the course of this laboratory we developed our R programming skills and learned techniques for exploring the behavior of simulation models.

Apart of this laboratory was (1) to read and understand someone else's R scripts, (2) modify an existing script, and then (3) to write your own. All three steps are good practice for real world quantitative and computational science. Reading and interpreting existing programs is useful not only to understand and evaluate someone else's science, but it is also a good way to learn how to program. Success in modifying a program provides evidence that you understand how it works. Finally, constructing your own programs to solve a new task requires a deeper mastery of the concepts.

Now that you have some experience, we can discuss both the modeling results and provide some programming tips and tricks in R . To further facilitate your learning, I have included a copy of most of my R scripts in the Appendices.

Task 3.1: Discrete Time Model Projections

Task 3.1.1

For this task, you were to program the solutions to the discrete time exponential growth model. This equation is

$$N_t = \lambda^t N_0.$$

The code for my solution is in Appendix . In this script, I show how to solve this problem using both a for-loop and by vectorizing the problem.

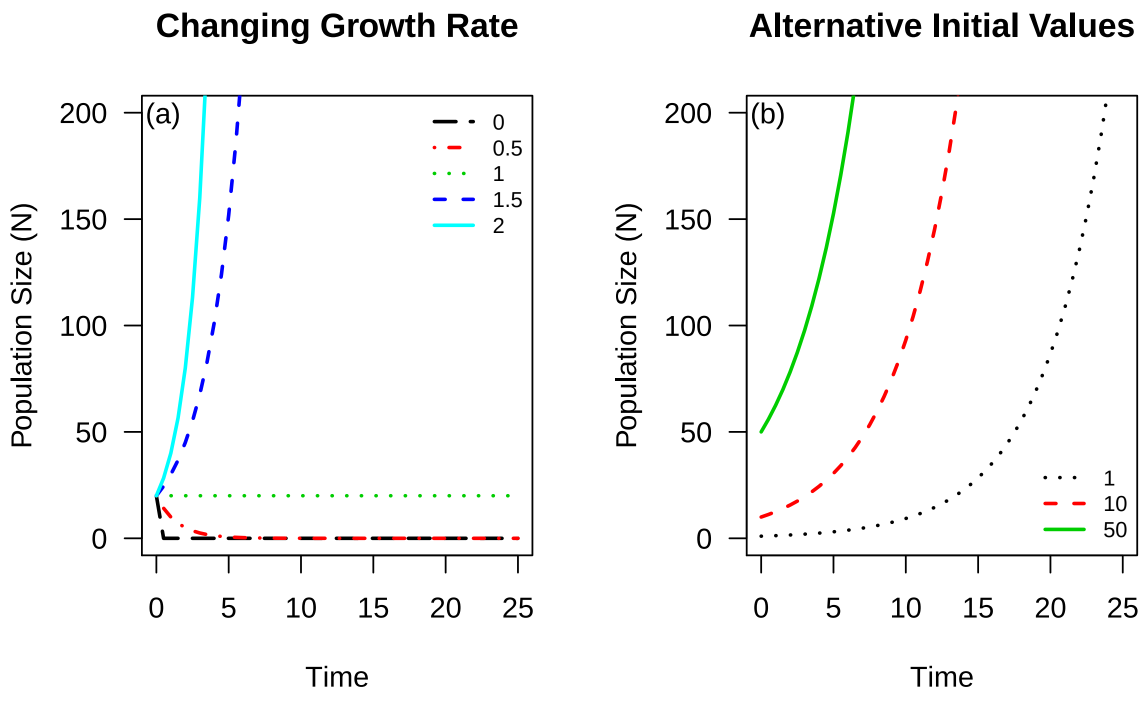

Figure 1: Solution to discrete time model showing the effects of changing (a) different growth rates and (b) different initial conditions.

Task 3.1.2

This task builds on the previous by having you investigate the effects of changing the parameter values and the initial conditions. This is a form of *sensitivity analysis*, which is a technique to investigate the dynamic behavior of a model and how the model parameters influence it. Here we performed a *one-at-a-time* (OAT) sensitivity analysis. The resultant behaviors indicate that λ changes how fast the population grows ($\lambda > 1$) or declines ($\lambda < 1$) and how the initial condition influences the outcome (Figure 1).

Task 3.2: Diagram the model

When we diagram a model, using any visual vocabulary, we want to help ourselves and the reader understand the system and how we have abstracted it. A good diagram will also hint at or explain the structure of the quantitative model.

Figure 2 is a diagram for the exponential growth population model using the Forrester diagramming symbols. Notice that there is one representation for each part of the model. For example, each state variable and parameter is represented only once. Information arrows shows when (but not how) elements relate to specific processes.

If we rewrite the exponential growth model as follows, it may be easier to see its corre-

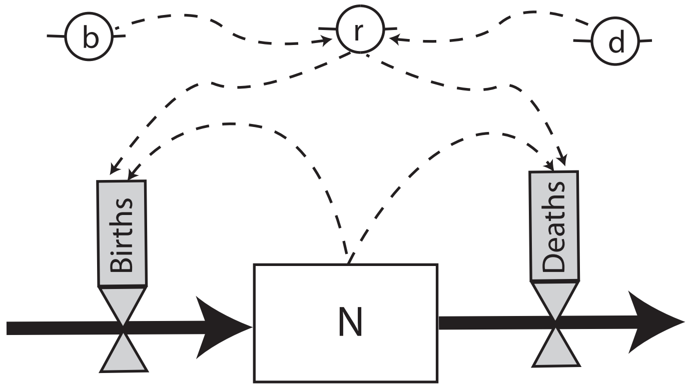

Figure 2: Forrester diagram of the exponential growth model.

spondence to the diagram.

$$\begin{aligned} \frac{dN}{dt} &= rN \qquad (1) \\ &= (b-d)N \qquad (2) \\ &= \underbrace{bN}_{\text{Births}} - \underbrace{dN}_{\text{Deaths}} \qquad (3) \end{aligned}$$

Now the input–output structure of the model is more obvious. Births add to the population and Deaths subtract from the population. It is also clear that the model assumes that births and deaths are effected identically by the population density.

Task 3.3: Population projections

For this task, you used the exact integral solution of the exponential growth model ($N_t = N_0 e^{rt}$) to project the future size of the population. You also used a *one-at-a-time* sensitivity analysis to determine the effect of changing the two parameters in this model (N_o and r) on the population projections. Figure 3 illustrates example solutions.

Task 3.4: Numerical approximations

In this task you were to compare the exact solution of the exponential growth model with numerical approximations using the Euler and lsoda methods. For the Euler method, we used three values of $\Delta t = 1, 0.1, 0.01$. Figure 4 shows these comparisons.

As Δt decreases the numerical approximation becomes more accurate. This is evident in both the plot of the solutions (Figure 4) and in the estimated RMSEP (Table 1).

It is interesting to note that these errors would increase if we increased the time span or our simulation. This is in part because we are working with a monotonically increasing func-

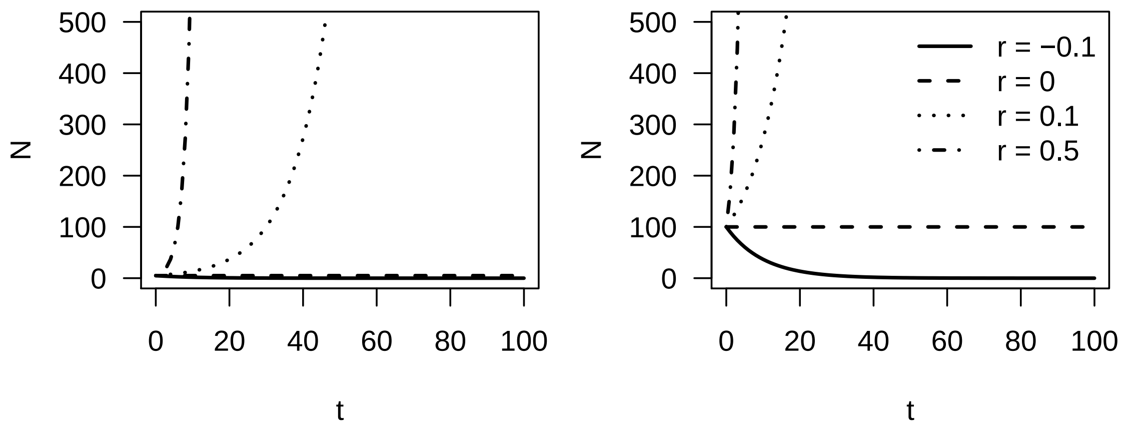

Figure 3: Effect of varying r on the exact solutions of the exponential model. Left panel shows when $N_0 = 5$ and right panel shows when $N_0 = 100$

Table 1: Error between the analytical solution and three Euler method numerical approximations of the exponential growth model as well as the lsoda calculation

dt	**RMSEP**
1	9294.745
0.1	1193.684
0.01	122.6739
lsoda	0.0228

tion. Numerical error becomes more interesting when we are working with non-monotonic functions like logistic growth. Thus, the true utility of lsoda is not realized in this exercise.

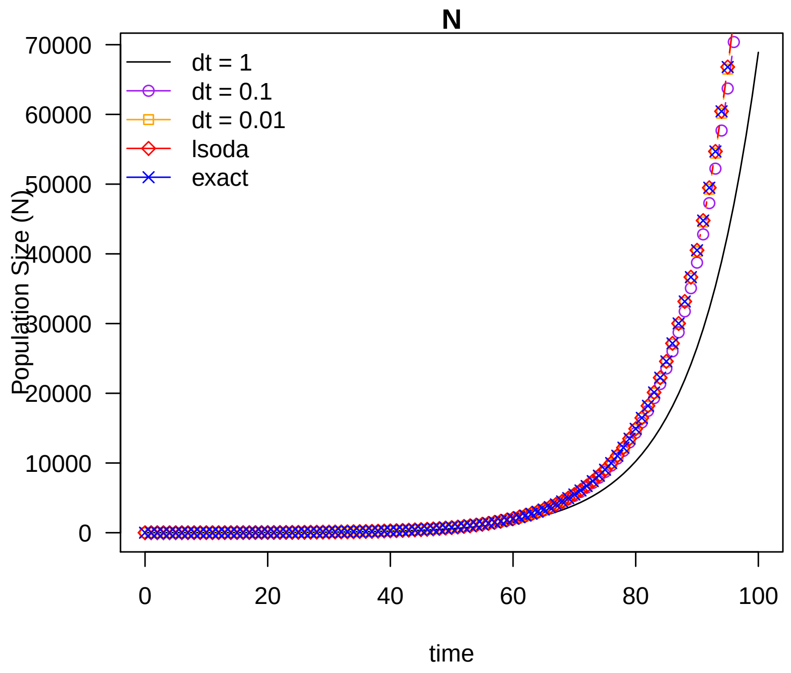

Figure 4: Analytical, Euler, and lsoda method numerical approximations of the exponential growth model.

Appendix: R Scripts

Discrete Time Solution

This script shows two ways of solving the discrete time problem. One uses a for-loop, the other vectorizes the problem. I then use the *sapply* command to neatly apply the growth function to different values of λ or N_o. Alternatively, you could have used another for-loop (as shown).

```
# Difference Equation
# Exponential Growth
# Borrett
# Sept 13, 2011
# BIOL534, Lab 3, Task 1
# ----------------------

# parameters
times=seq(0,25,by=0.5)     # time span for solution
lambda = 1.1               # geometric growth rate
NO = 5                     # initial population size

# --- Two Alternative Solutions ---
# Solution 1: Vectorized
N=NO*lambda^times
```

```
plot(times,N)

# Solution 2: For-Loop
N2=0               # neet to initialize the populationvector
for(i in times){
  N2[i+1] = N0*lambda^i
}
plot(times,N2)
# ---------------------------

##############
# Task 3.1.1
##############
N0 = 20                     # initial condiion
lambda = c(0,0.5,1,1.5,2)   # set of growth rates
times=seq(0,25,by=0.5)      # time span
#
# using sapply for solutions
N2 = sapply(lambda, function(lambda1) N0*lambda1^times)

# you could alternatively use a for-loop to solve this problem
N2a = matrix(rep(0,length(times)*length(lambda)),
  nrow=length(times))  # intialize population container
#
for(i in 1:length(lambda)){
  N2a[,i]=N0*lambda[i]^times
}

##############
# Task 3.1.2
##############
N0 = c(1,10,50)   # initial population densities
lambda = 1.25     # discrete growth rate
times=seq(0,25,by=0.5)    # solution time
#
# using sapply function for solutions
N3 = sapply(N0, function(N0) N0*lambda^times)

# -- FIGURE --
fn="../figures/fig-t31.pdf" # file name
pdf(fn, height=4.5,width=7)   # opens PDF file to write figure
opar <- par(las=1,
            mfrow=c(1,2))   # create a figure with two plots
                            # (1 row, 2 columns)
# - first plot
matplot(times,N2,
        ylim=c(0,200),
        type="l",lty=5:1,lwd=2,
        main="Changing Growth Rate",
        ylab="Population Size (N)",
        xlab="Time"
        )
mtext("(a)",side=3,line=-1,adj=0.01)   # add panel label
legend("topright",legend=c("0","0.5","1","1.5","2"),
       cex=0.75,    # changes legend text size
       col=1:5,     # changes plot colors (by number)
       lty=5:1,     # sets line type
       lwd=2,
       bty="n") # turns off legend box
# - second plot
matplot(times,N3,
        ylim=c(0,200),type="l",
        lty=3:1,lwd=2,
        main="Alternative Initial Values",
        ylab="Population Size (N)",
        xlab="Time"
        )
mtext("(b)",side=3,line=-1,adj=0.01)
legend("bottomright",
       legend=c("1","10","50"),
       cex=0.75,   # change text size
       lty=3:1,
       col=1:3,
       lwd=2,
       bty="n") # turns off legend box
#
rm(opar)
dev.off()  # close PDF
cmd <- paste("open",fn) # create a system command to open the figure
system(cmd)             # send the command to operating system (Mac only)
```

Analytical Solution to the Exponential Population Model: A

Here is an initial R script that you could use to solve the problem.

```
# Solving Continuous Time Equations
# Biol534, Lab 3, Task 3.3
# Stuart Borrett
# Sept. 9, 2010
# ----------------------------------

# model parameters
No = 100;    # initial population size
t = 0:100;   # time span of evaluation
```

```
N=rep(0,101) # initialize the population size vector
line.type = 1 # inital line type

for(r in c(-0.1,0,0.1,0.5)){
  N=No*exp(r*t);     # analytical solution of the exponential growth model

  if(r==-0.1){
    plot(t,N,type="l",lty=line.type, lwd=2,
         ylim=c(0,1000))
  } else {
    points(t,N,type="l",lty=line.type, lwd = 2)
  }
  line.type=line.type+1;     # incremental counter to change the line type
}

# add legend
legend("topright",
       legend = c("r = -0.1", "r = 0", "r = 0.1", "r = 0.5"),
       col="black",
       lwd = 2, lty = c(1:4),
       bty="n",
     )
```

Analytical Solution to the Exponential Population Model: B

Here is a fancier R script that performs all of the required analysis at once and creates a PDF figure.

```
# Solving Continuous Time Equations
# Biol534, Lab 3, Task 3.3
# Stuart Borrett
# Sept. 9, 2010
# ---------------------------------

# model parameters
No=c(5,100); # inital population size
r = c(-0.1, 0, 0.1, 0.5); # intrinsic growth rate
t = 0:100;   # time span of evaluation

# model setup / program parameters
N=rep(0,101) # initialize the population size vector

# start creating pdf of plot
fn <- "../figures/t33b.pdf"
pdf(file=fn, height=3,width=7)      # open the PDF file
opar <- par(las=1,mfcol=c(1,2),
            mar=c(4,4,1,1),oma=c(1,1,1,1))  # change plot parameters
```

```
# model projection loops
for(i in 1:2){            # vary inital population size

  for(j in 1:4){  # vary intrinsic growth rate

    N=No[i]*exp(r[j]*t);     # analytical solution of the exponential growth model

    if(r[j]==-0.1){
      plot(t,N,type="l",lty=j, lwd=2,
           ylim=c(0,500))
    } else {
      points(t,N,type="l",lty=j, lwd = 2)
    }

  }
}

# add a legend to the plot
legend("topright",
       legend = c("r = -0.1", "r = 0", "r = 0.1", "r = 0.5"),
       col="black",
       lwd = 2, lty = c(1:4),
       bty="n",
      )


dev.off()   # close the PDF file
rm(opar)

# on a mac, the next line will open the PDF created
cmd <- paste("open",fn)
system(cmd)
```

Comparing Numerical Approximations

Run File

```
# exponential growth - run file
# Borrett 13 Sept 2011
# - Solution to Lab 4, Task 4.4
# deSolve version
# ------------------------------

# install required library
library(deSolve)

# load model
source("expgrowth-model.r")
```

```
# set model parameters  (using list data type)
parameters <- c(b = 0.5, # specific birth rate
                d = 0.4); # specific death rate
state <- c(N = 5)          # initial state
times <- seq(0,100, by=1);  # time vector

## ---------------------------------------------
## NUMERICAL APPROXIMATIONS
out=list()        # make out a list (initialize variable)
dt=c(1,0.1,0.01)  # integration steps
#
## EULER SOLUTIONS -- numerical approximations
for(i in 1:3){    # solve model with different time steps.
  out[[i]] <- ode(y=state,
                 times=times,
                 func=expgrowth,
                 parms=parameters,
                 method="euler",hini=dt[i])
  }
## LSODA SOLUTIONS -- numerical approximations
out.lsoda <- ode(y=state,times=times, func=expgrowth, parms=parameters)  # integrate the model

## Find exact solution
Na <- as.list(state)$N[1] * exp( (as.list(parameters)$b-as.list(parameters)$d)*times);

## Calculate Root Mean Square Error for each solution
rmsep=list()
for(i in 1:4){
  if(i <4){
    rmsep[[i]] <- sqrt( sum( (Na - out[[i]][,2] )^2)/length(Na) );  # Euler errors
  } else {
    rmsep[[i]] <- sqrt( sum( (Na - out.lsoda[,2])^2)/length(Na));   # lsoda error
  }
}

# -- PLOT -- #
fn <- "../figures/t34-compare.pdf"
pdf(fn,height=5,width=6)
opar <- par(las=1,mar=c(4,4,1,1),oma=c(1,1,1,1) )
#
plot(out[[1]])
#
points(out[[2]],pch=21,type="b",col="purple")
points(out[[3]],pch=22,type="b",col="orange")
points(out.lsoda,pch=23,type="b",col="red")
points(times,Na,lty = 2, col="blue",pch=4)
```

```
mtext("Population Size (N)",side=2,line=3,las=3)
#
legend("topleft",pch=c(NA,21,22,23,4),lwd=1,legend=c("dt = 1","dt = 0.1","dt = 0.01", "lsoda",
       col=c("black","purple","orange","red","blue"),
       bty="n")
#
dev.off()  # close PDF file
cmd <- paste("open",fn)
system(cmd)
```

Model File

```
# exponential growth - model
# Borrett, 20 Oct 2010
# deSove version
# ----------------------------

expgrowth=function(t, state, parameters){
  with(as.list(c(state,parameters)), {
    # auxiliary parmeters
    r = b - d;

    # rate of change
    dN = r*N;

    list(c(dN))
  }) # end with(as.list ...
}
```