ESS 575: Discrete Logistic Lab

Team England

31 August, 2022

Team England:

- Caroline Blommel
- Carolyn Coyle
- Bryn Crosby
- George Woolsey

cblommel@mail.colostate.edu, carolynm@mail.colostate.edu, brcrosby@rams.colostate.edu, george.woolsey@colostate.edu

Objective

See the course website for a full lab description. Important R concepts and methods utilized include:

1. writing functions
2. creating data structures
3. looping
4. plotting

Question 1

$$x_{t+1} = \lambda x_t(1 - x_t) \tag{1}$$

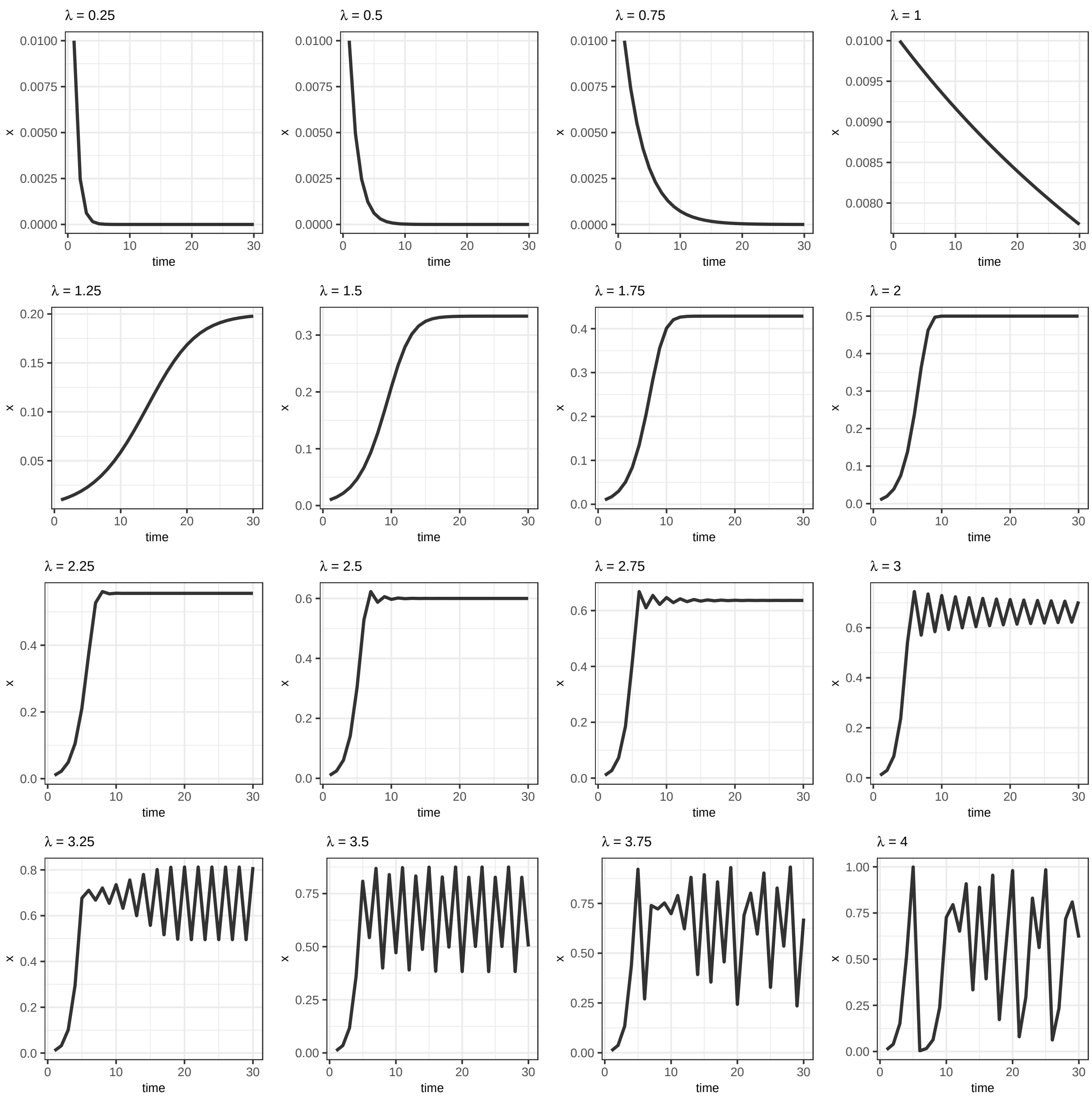
Where:

- λ is the per capita rate of population growth
- x_t is the population size at time t.

Write an R function for equation 1. Use your function to simulate how population size changes over time in response to variation in the parameter λ. Set up a model experiment with an outer for loop controlling the value of λ to range from .25 to 4.0 in steps of .25. Create an inner loop varying time from 2 to 30 in steps of 1. Assume that the initial condition (i.e., the value of x at time = 1) for the population size is .01. Create a plot of x as a function of time for each value of λ. You should display your plots with 4 x 4 panels, one panel for each of your simulations. Give each plot a title showing the value of λ. (Hint, convert the numeric value of λ to a character value using z = as.character(lambda) and use main = z as an option in the plot statement). What can you conclude about the effect of λ on the dynamics? Use your panel plot to illustrate the points in your discussion.For an engaging time waster, figure out how to put a symbolic λ in the title of your plots. So, you will need to combine a symbolic λ with a numeric value that changes with each plots title.

```r
# lambda
lambda <- seq(0.25, 4, 0.25)

# time
t <- seq(1, 30, 1)

# x
x <- numeric(length(t))
x[1] <- 0.01

# set up function
fn_x_t <- function(x, t, lambda) {
  x_t <- lambda * x[t-1] * (1 - x[t-1])
  return(x_t)
}

# set up plot grid
plts <- list()

#loop through lambda values
for (my_l in 1:length(lambda)) {
  # assign values of x by t
  for (my_t in 2:length(t)) {
    x[my_t] <- fn_x_t(x, my_t, lambda[my_l])
  }

  # set up temp data
  dta <- data.frame(
    t
    , x
  )
  # plot
  plts[[my_l]] <-
    ggplot(dta, aes(x = t, y = x)) +
      geom_line(
        size = 1
        , alpha = 0.8
      ) +
      labs(
        title = bquote(paste(
            lambda
            , " = "
            , .(lambda[my_l])
          ))
      ) +
      xlab("time") +
      ylab("x") +
      theme_bw() +
      theme(
```

```
        legend.position="none"
        , plot.title = element_text(size = 9)
        , axis.text =  element_text(size = 8)
        , axis.title =  element_text(size = 8)
      )
}

# pass plots to grid extra package
do.call(gridExtra::grid.arrange, plts)
```



Short Answers

What can you conclude about the effect of λ on the dynamics?

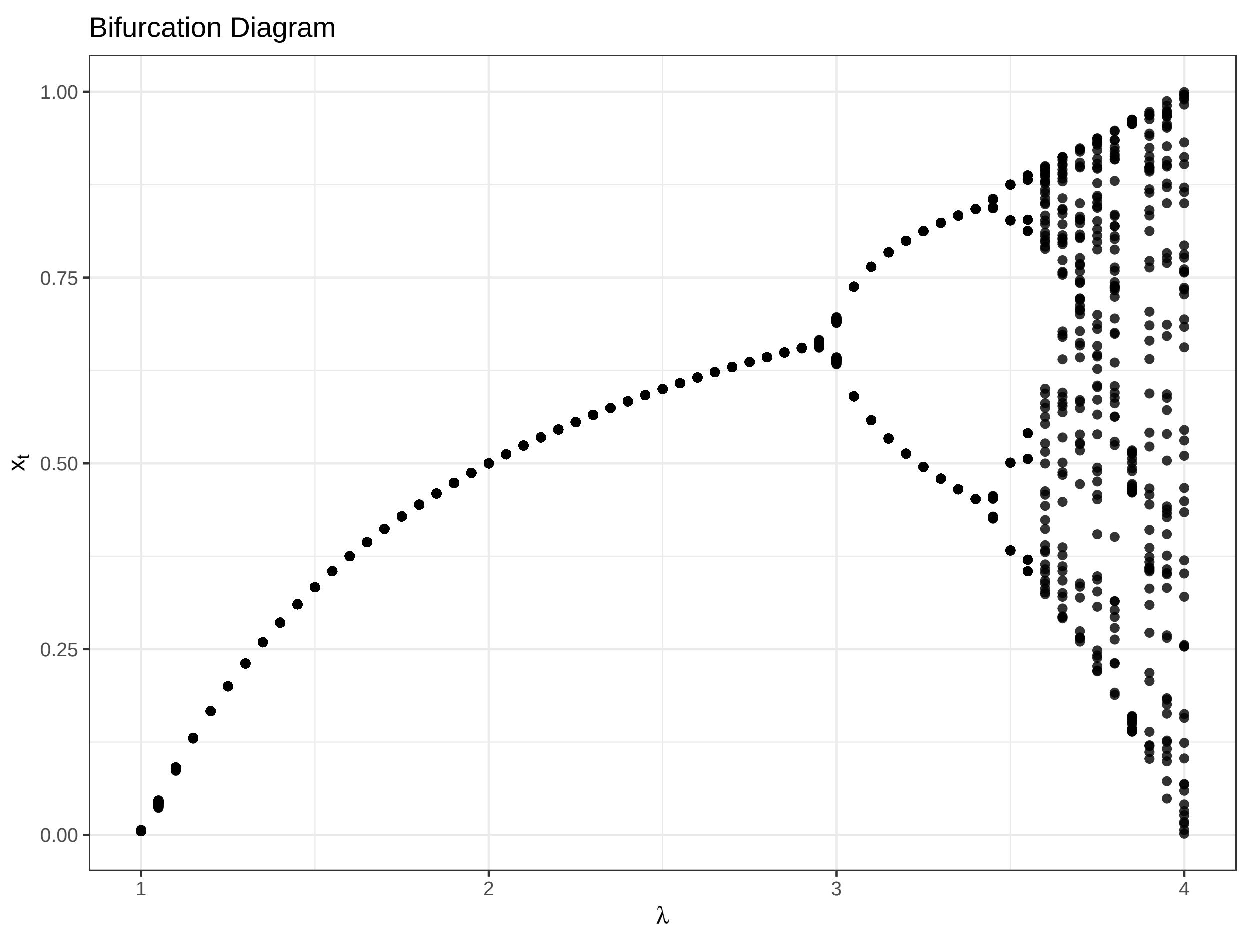
As shown in the output above, we can conclude that when lambda is less than 1, the impact of lambda on population size results in a negative logistic relationship. Alternatively, for values of lambda greater than 1 but close to 1, the impact of lambda on population size results in a positive logistic relationship. However, as lambda approaches positive infinity, the impact of lambda is random over time.

Question 2

Now set up a model experiment where you vary λ in steps of .05 from λ=1 to λ=4. Run the model for 100 time steps and save the values of $x[t]$ for the last 50 time steps (t=51–100) to a matrix where column one contains the value of λ and column two contains the value of $x[t]$. (Discard the values of $x+t$ for $t < 51$). Plot the values of x at $t = 51$ to 100 as a function of λ. The trick to this problem is thinking about how to store your data relative to the iteration that is going on (which is what makes it a good problem). Think about it —- you need a vector that keeps track of $x[t]$ as t goes from 1 to 100. You also need a matrix that consists of groups of rows of data, where each group consists of 50 rows of each value of λ in one column and the values of $x[t]$ at time = 51-100 in the other column. The full "stack" of these rows is the matrix. As a hint, you could implement this by creating a "counter" within your loops (say, `j = j + 1`) that you use to index the rows of the array used to store the values of λ and $x[t]$. At the end of your model experiment, that is, after all of the looping is complete, the counter j will equal the number of rows in the data matrix. Alternatively (and I think better) you might use the `rbind()` function without needing a counter. For example, consider the following code illustrating how to build a matrix of data by accumulating rows one at a time:

```
rm(list=ls())
# lambda
lambda <- seq(1, 4, 0.05)

# time
t <- seq(1, 100, 1)

# x
x <- numeric(length(t))
x[1] <- 0.01

# set up function
fn_x_t <- function(x, t, lambda) {
  x_t <- lambda * x[t-1] * (1 - x[t-1])
  return(x_t)
}

# set up empty vector
  vect_lamda <- c()
  vect_xt <- c()
  vect_t <- c()

#loop through lambda values
for (my_l in 1:length(lambda)) {

  # assign values of x by t
  for (my_t in 2:length(t)) {
    x[my_t] <- fn_x_t(x, my_t, lambda[my_l])
  }

  # add vectors together
```

```
  vect_lamda <- c(vect_lamda, rep(lambda[my_l], each=length(t[51:100])))
  vect_xt <- c(vect_xt, x[51:100])
  vect_t <- c(vect_t, t[51:100])
}
# put vectors together
fnl_matrix <- matrix(c(vect_lamda, vect_xt, vect_t), ncol = 3)
fnl_data <- data.frame(vect_lamda, vect_xt, vect_t)

# plot
  ggplot(fnl_data, aes(x = vect_lamda, y = vect_xt)) +
    geom_point(
      alpha = 0.8
    ) +
    labs(
      title = "Bifurcation Diagram"
    ) +
    xlab(expression(lambda)) +
    ylab(expression(x["t"])) +
    theme_bw() +
    theme(
      legend.position="none"
    )
```



Short Answers

Interpret what you see in this diagram.

The bifurcation diagram above shows that for lambda values greater than one but less than 3 the population size tends to increase over time. For values of lambda near 3 and above, the influence of lambda on population size is random. That is, population size either increases or decreases without an identifiable pattern for lambda values at 3 and above.