Lab 3

Sawyer Balint

Fall 2024; Marine Semester Block 3

Contents

1 Introduction

This document is available at https://github.com/sjbalint/BI521/tree/main/scripts/labs

```
#import packages
library(tidyverse) #for data wrangling
library(here) #for filepath management
library(ggsci) #for colors
library(scales) #for log axis breaks

#custom graphing theme
#including geoms to reduce repetition
theme <- list(
  theme_classic(),
  scale_color_jama(),
  scale_fill_jama(),
  theme(legend.position="right"),
  labs(x="Year",y="Population")
)

logtheme <- list(
    scale_y_log10(labels = trans_format(log10, math_format(10^.x))),
    annotation_logticks(sides="l")
)
```

2 Part I

2.1 Task 1

Write a custom function to calculate the value of $\frac{dN}{dt} = rN(t)(1 - N(t)/k) - EN(t)$. The function should take as arguments the current stock-size N, level of fishing effort E and model parameters r, k.

```
#function with some default parameters
BM_1 <- function(N=0.25, r=2.5, K=1, E=0.5){

  #calculate dN_dt
  dN_dt <- r * N * (1-(N/K))-(E*N)

  return(dN_dt)

}
```

Use the custom R function to plot the growth rate dN/dt for a few different sets of values for parameters r and K. How do the parameters affect the shape?

```
#empty list to store results
result.list <- list()

#nested for loops
#range of r values
for (r in c(1,2,3)){

  #range of K values
  for (K in c(1,2)){

    for (N in seq(0,2,0.01)){


      #calculate dN/dt
      dN_dt <- BM_1(N=N, r=r, K=K, E=0)

      #dataframe to store results
      df <- data.frame(dN_dt=dN_dt, N=N, r=as.character(r), K=as.character(K))

      #store results
      result.list <- append(result.list, list(df))

    } #N
  } #k
} #r

#compile results
result.df <- bind_rows(result.list) %>%
  filter(dN_dt>=0)

#plot
ggplot(result.df, aes(N, dN_dt, color=r, linetype=K))+
```

```
  theme+
  geom_line()+
  scale_y_continuous(expand=expansion(mult=c(0,0.05)))+
  labs(x="N", y="dN/dt")
```

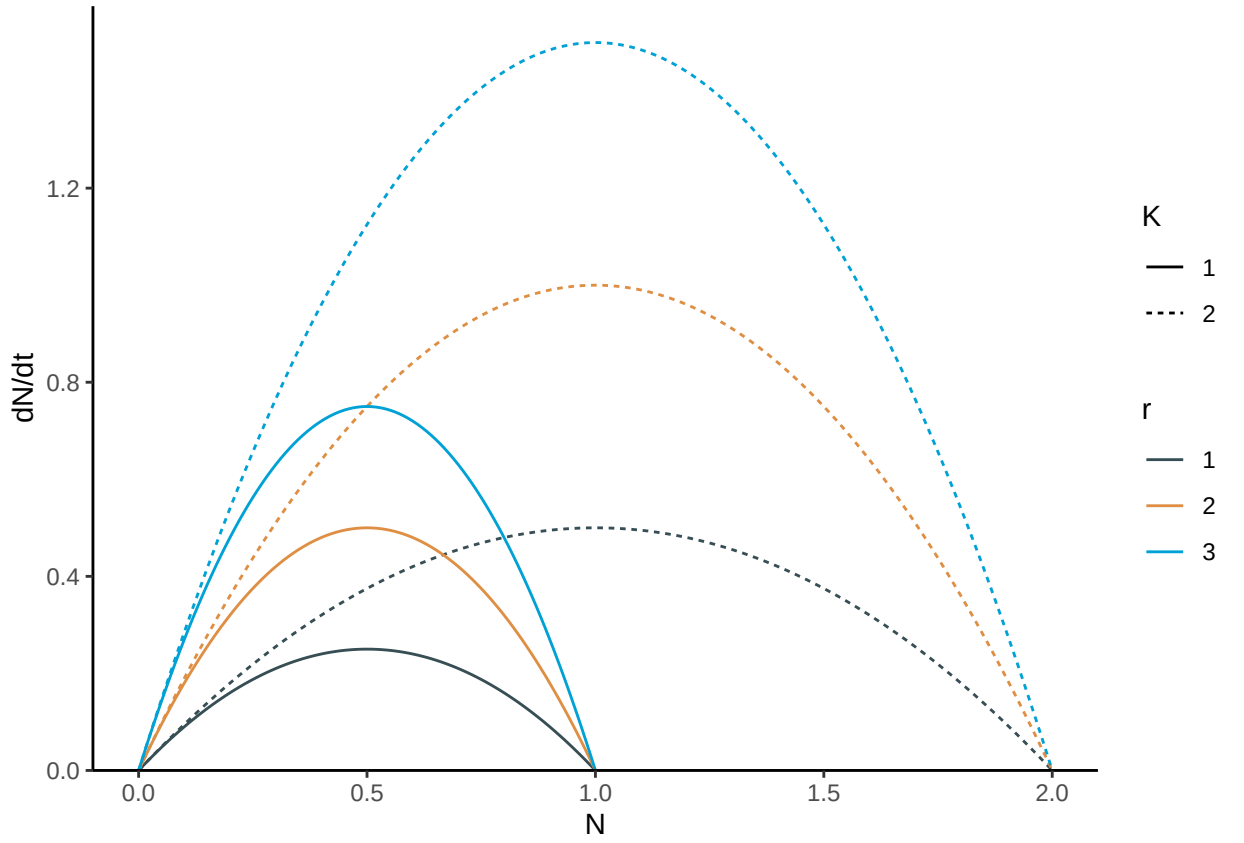

Fig. 1: *dN/dt* increases with both r and K, and the maximum *dN/dt* occurs when $N = 1/2K$

2.2 Task 2

Rewrite the `for` loop above to simulate the population dynamics but use the custom function to get the rate of change of the population.

```
#empty list to store results
result.list <- list()
#duration of iteration
dt <- 0.01

#nested for loops
#range of r values
for (r in c(1,2,3)){

  #range of K values
  for (K in c(1,2)){
```

```
    #reset N0
    N <- 0.25

    #iterate over 10 years (if dt is in units of years)
    for (t in c(1:(10/dt))){

      #calculate N
      N <- N + dt*BM_1(N=N, r=r, K=K, E=0)

      #dataframe to store results
      df <- data.frame(t=t*dt, N=N, r=as.character(r), K=as.character(K))

      #store results
      result.list <- append(result.list, list(df))

    } #time
  } #k
} #r

#compile results
result.df <- bind_rows(result.list)

#plot
ggplot(result.df, aes(t, N, color=r, linetype=K))+
  theme+
  geom_line()
```

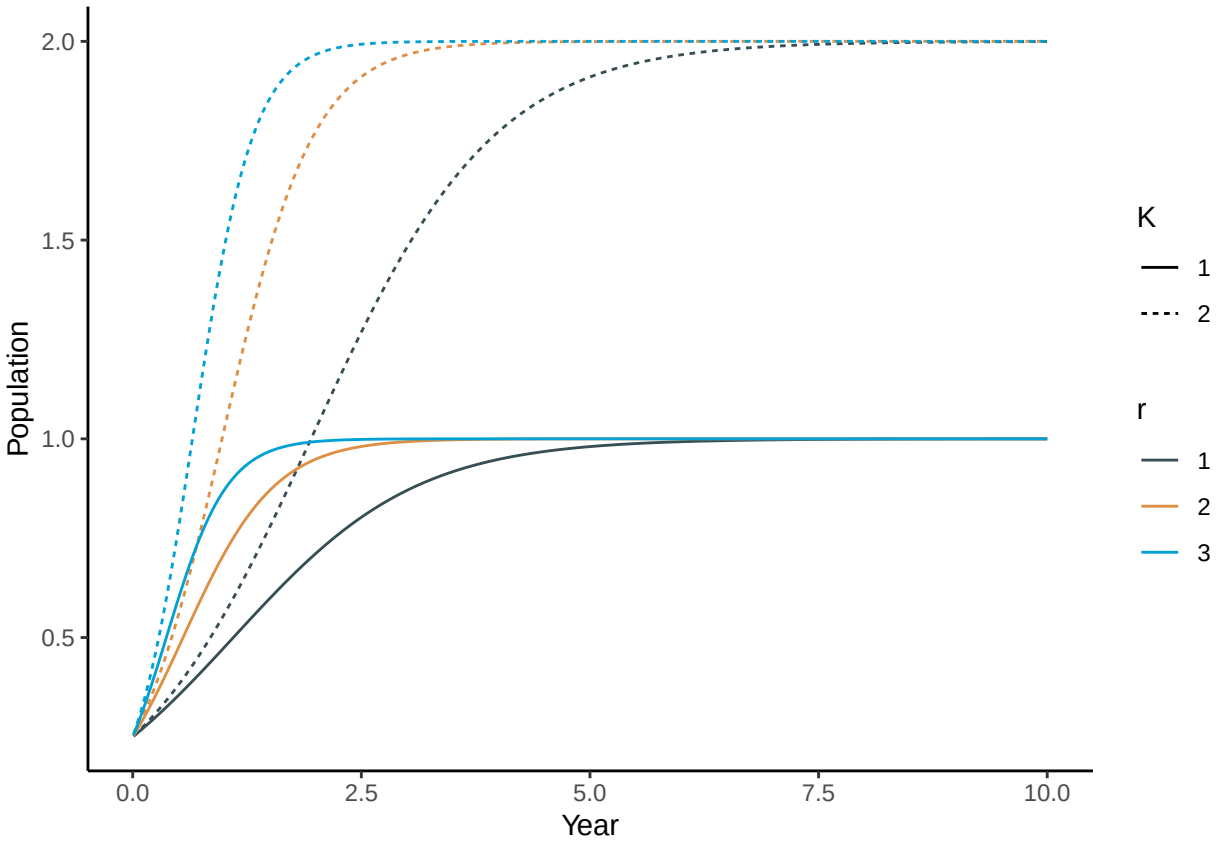

Fig. 2: Population asymptotically approaches K, and a larger K results in a larger population at equilibrum. r influences the "steepness" of the population curve: at higher replacement, the return from the purtubation is faster.

2.3 Task 3

```
#make a function with some default parameters
BM_2 <- function(N0=0.25, r=2.5, K=1, E=0.5, dt=0.01, duration=10){

  #empty list to store results
  result.list <- list()

  #set initial population
  N <- N0

  #iterate
  for (t in c(1:(duration/dt))){

      #calculate N
      N <- N + dt*BM_1(N=N, r=r, K=K, E=E)

      #dataframe to store results
      df <- data.frame(t=t*dt, N=N)
```

```
      #store results
      result.list <- append(result.list, list(df))

  }

  #compile
  result.df <- bind_rows(result.list)

  return(result.df)
}


result.list <- list()

for (E in c(0.2, 0.5, 0.7)){

  df <- BM_2(E=E) %>%
    mutate(E=as.character(E))

  result.list <- append(result.list, list(df))

}

result.df <- bind_rows(result.list)

#plot
ggplot(result.df, aes(t, N, color=E))+
  theme+
  geom_line()
```

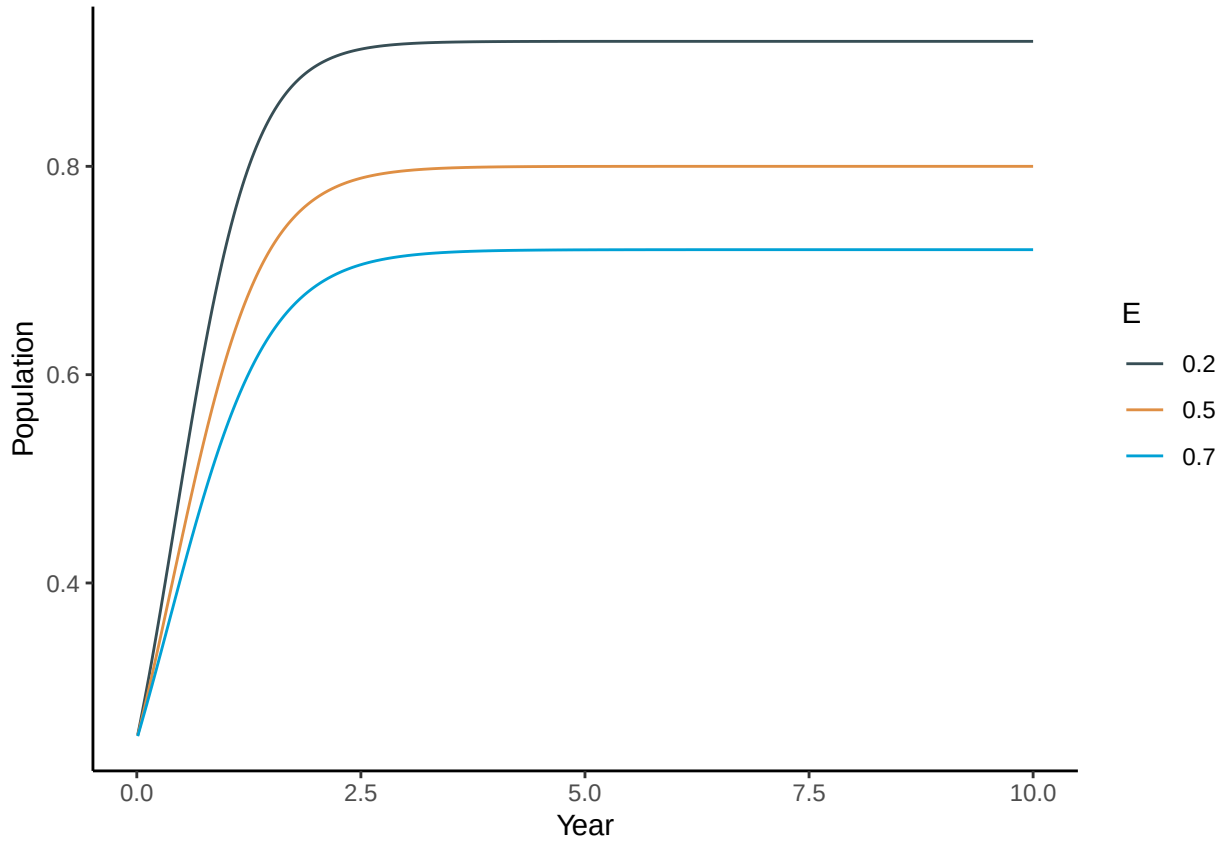

Fig. 3: E impacts the equilibrium population size, with larger E corresponding to a lower population size and slower recovery.

2.4 Task 4

```
#function to fine equilibrium population level
equil <- function(K=1,E=0.5,r=2.5){
  K * (1 - (E/r))
}

#function for single perturbation
crt <- function(dN=0.2, r=2.5, K=1, E=0.2, index=FALSE){

  N_equil <- equil(K,E,r)
  N0 <- N_equil - dN
  v_N <- BM_2(N0=N0, r=r, K=K, E=E)
  return_index <- min(which(abs(v_N - N_equil) < dN / exp(1)))
  return_time <- v_N[return_index,"t"]

  #I want to be able to decide whether I get the index or time step
  #if dt = 0.01, then time step = 1000 * index
  if (index){
     return(return_index)
  } else {
```

```
    return(return_time)
  }
}

#plot of population over time
plot.df <- BM_2(E=0.2)

#find return time
#this is why we need index... because time step is decimal
point <- c(crt(E=0.2),plot.df[crt(E=0.2, index=TRUE),"N"])

#make the plot
ggplot(plot.df, aes(t, N))+
  theme+
  geom_line()+
  geom_point(x=point[1], y=point[2], shape=23, size=4, fill="darkred")+
  geom_hline(yintercept=equil(1,0.2,2.5), linetype="dashed")
```

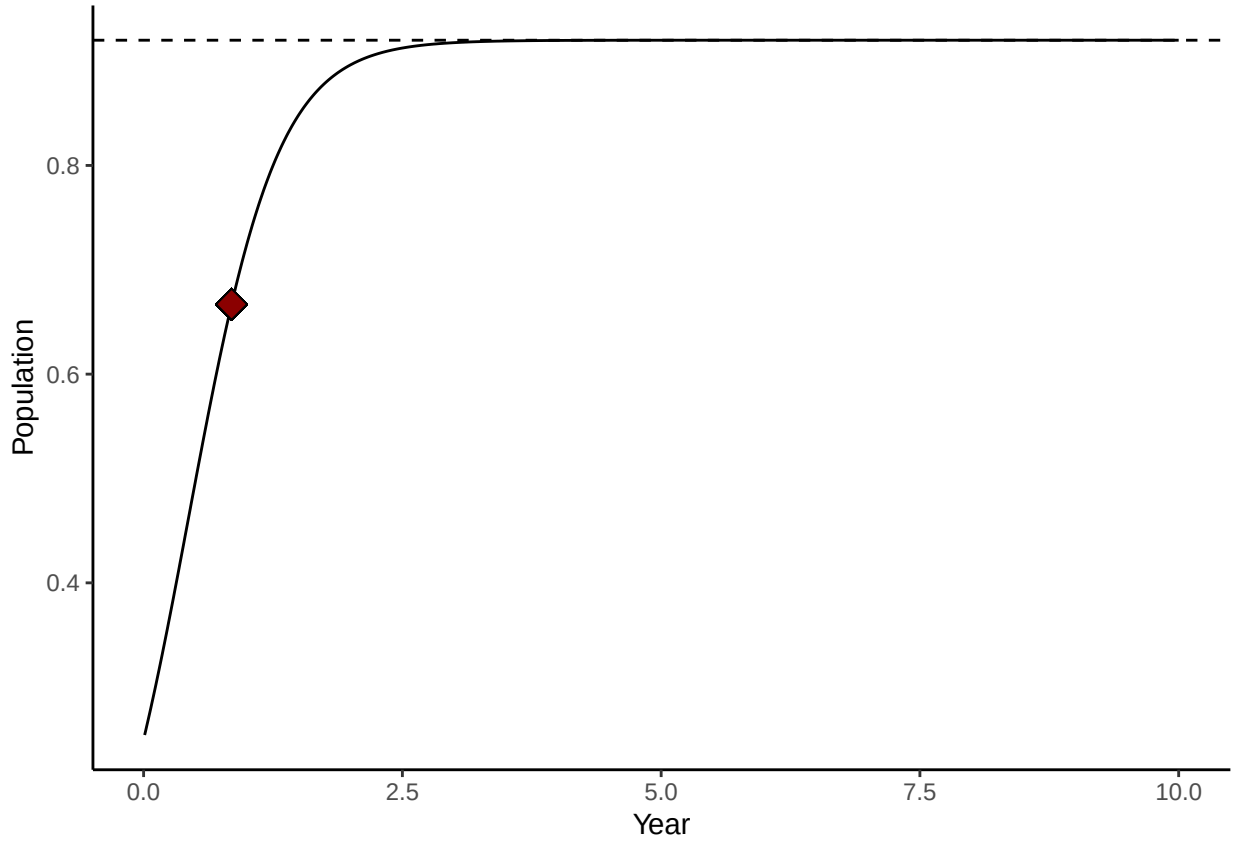

Fig. 4: Plot of return time with $E = 0.2$.