Population consisting of juveniles (𝐽) and adults (𝐴). The juveniles mature to become adults at a rate *g* but also die at a rate *μ*J. The adults reproduce at a rate *b* and die at a rate 𝜇0 (1+𝑐A), where *c* is a positive constant representing density dependence of the mortality

*3a) Where is the non-trivial equilibrium? (1p)*

I solved question 3a by solving equation dJ/dt and dA/dt mathematically by rearranging these combination of equations together. An equilibrium is defined as ‘no change’. In this case when dJ/dt or dA/dt = 0

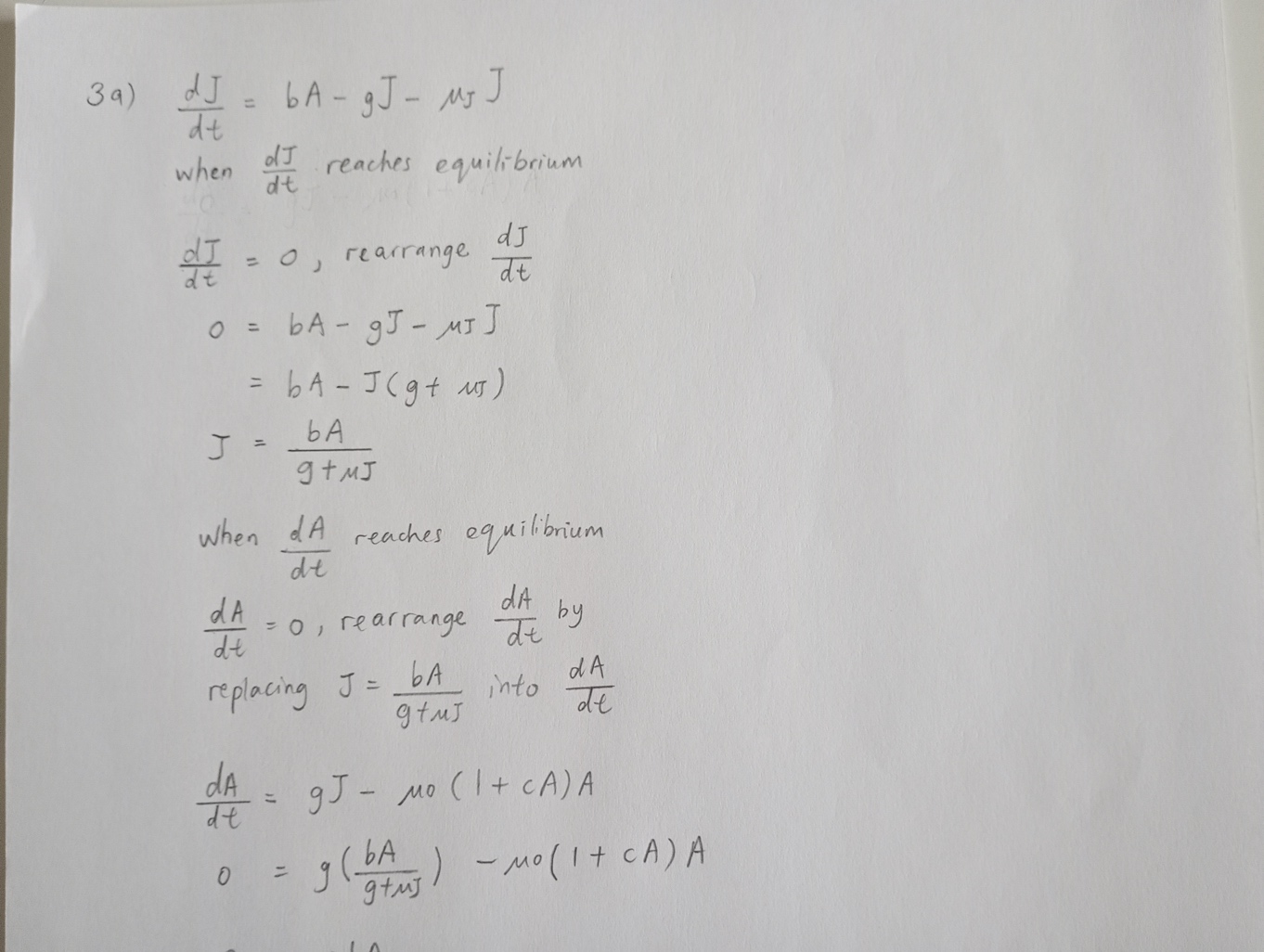
**

Figure 1. Answer for question 3a, see more on the next page.

*A white paper with writing on it

Description automatically generated*

Figure 2. Answer for question 3a, see more on below.

*A white paper with writing on it

Description automatically generated*

Figure 3. Answer for question 3a, the trivial solution for A is 0, see more on the next page.

*A paper with writing on it

Description automatically generated*

Figure 4. Answer for question 3a.

The non-trivial equilibrium for J and A are

A close up of a math problem

Description automatically generatedA close up of a paper

Description automatically generated

*3b) Calculate the Jacobian matrix of that equilibrium! (1p)*

I used the equilibrium value of J and A (derived from 3a). I carried out partial derivative method on differential equation of dJ/dt and dA/dt to fit the component into the Jacobian matrix.

A white paper with writing on it

Description automatically generated

Figure 5. Answer for question 3b.

*3c) How would you use the Jacobian to evaluate the stability properties of the equilibrium (don’t do it!)? (1p)*

To determine the stability of an equilibrium in multidimensional dynamic system, I calculate the Jacobian matrix of that equilibrium. I used the result from the Jacobian matrix (see figure from question 3b). The trace of the Jacobian (T) is negative, indicating that the sum of the diagonal elements of the matrix is negative. The determinant of the Jacobian (D) is negative, indicating the product of the diagonal elements.

The eigenvalues (λ) are negative, indicating Equilibrium A∗ and J\* are stable because all eigenvalues of Jacobians have negative real part. See below,

A white paper with writing on it

Description automatically generated

Figure 6. Answer for question 3c.

*3d) Write an R script that plots the isoclines of the system! (2p)*

First I need expressions for the isoclines, which are the points in phase space where one or the other of the state variables has zero growth.

I solved question 3d by coding a function that includes J and A reached equilibrium to draw J and A isoclines. The x-axis represents the population of adults.

The y-axis represents the population of juveniles.

The blue dashed line represents the J\_isocline, indicating where the juvenile population remains constant concerning changes in the adult population.

The red dashed line represents the A\_isocline, indicating where the adult population remains constant concerning changes in the juvenile population.

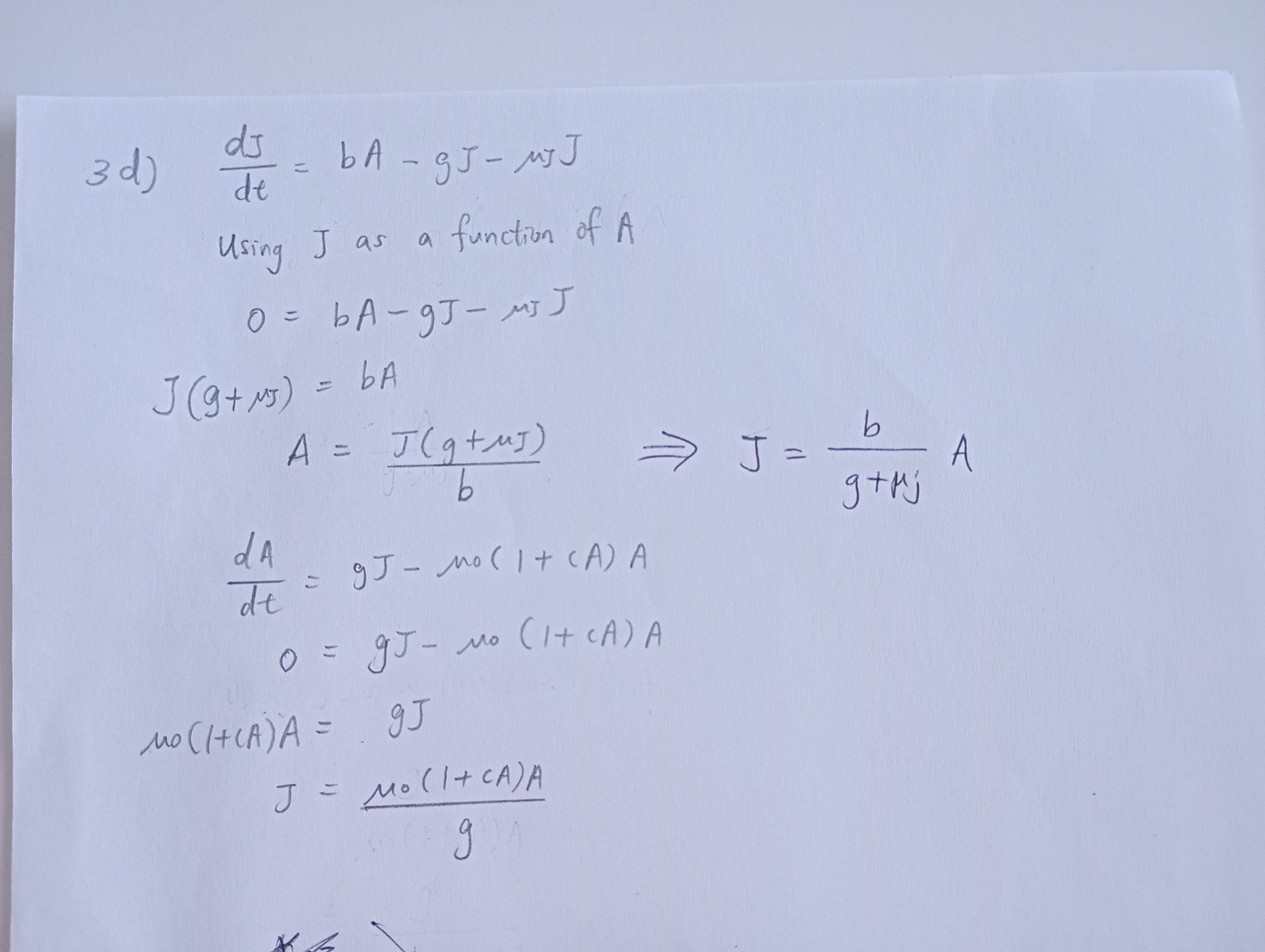
This plot helps visualize the equilibrium points and relationships between juvenile and adult populations in a given ecological or population dynamics model.

Figure 7. Answer for question 3d.

R script and answer for question 3d (Figure 7) and 3eii (Figure 8).

**Doc.name: BIOS13\_examQ3d3eii\_HooiMin.R**

rm(list = ls())

# Define system parameters first

pars <- list(b=1, g=0.5, mu\_j=0.3, mu\_0=0.2, c=0.01)

# Plotting J isocline as a function of A:

A <- seq(0, 500) # a range of A-values

J <- pars$b \* A / (pars$g + pars$mu\_j)

plot(A, J, type="l", lty=5, col='blue', xlab='Adult (A)', ylab='Juvenile (J)',

ylim=c(0, 500), xlim=c(0, 500))

# Plotting A isocline

J <- pars$mu\_0 \* (1 + pars$c \* A) \* A / pars$g

lines(A, J, type="l", lty=5, col='red')

# Next, use ode to calculate a trajectory:

library(deSolve)

# The set of differential equations:

AJfunc <- function(t, AJ, pars) {

A <- AJ[1] # A is the first element of AJ

J <- AJ[2] # J is the second element of AJ

dAdt <- pars$g \* J - pars$mu\_0 \* A - pars$mu\_0 \* pars$c \* A^2

dJdt <- pars$b \* A - pars$g \* J - pars$mu\_j \* J

return(list(c(dAdt, dJdt)))

}

time\_vec <- seq(0, 500, 0.1) # Reducing the time range

AJ0 <- c(A = 5, J = 150) # Adjusted initial values for A and J

results <- ode(func = AJfunc, times = time\_vec, y = AJ0, parms = pars)

# Plot the result in phase space using the results data:

lines(results[, "A"], results[, "J"], type='l', col='black')

legend('bottomright', legend=c('J isocline', 'A isocline', 'Trajectory'),

col=c('blue', 'red', 'black'), lty=c(5, 5, 1))

A graph of a person with a line

Description automatically generated with medium confidence

Figure 8. J and A isoclines in the phase plane together with the trajectory.

*3e) Write another script that simulates the differential equations and plots the result in two ways: i) as A and J vs. time and ii) in the phase plane together with the isoclines. (2p) (you may use the same parameter values as above, and an initial condition of your own choice)*

I solved question 3e by coding into two R script, in which 3ei ( R script run as A and J vs. time) and 3eii (R script run as in the phase plane together with the isoclines, Figure 8).

I created an ode function to solve the differential equation.

R script and the answer for question 3ei.

**Doc.name: BIOS13\_exam\_Q3ei\_HooiMin.R**

rm(list = ls())

library(deSolve)

# set up a vector of time-points for the output:

timevec <- seq( 0, 100, by = 1)

# Define parameters

P <- list(b = 1, g = 0.5, mu\_j = 0.3, mu\_0 = 0.2, c = 0.01)

ja0 <- c(j = 150, a = 5) # initial population sizes

# define the function:

JA\_sys <- function(t, ja, P) {

# extract vector content:

j <- ja[1] # firt element of the vector

a <- ja[2] # second element of the vector

# calculate the age-structured rates:

dJdt <- P$b \* a - P$g \* j - P$mu\_j \* j

dAdt <- P$g \* j - P$mu\_0 \* (1 + P$c \* a) \* a

return(list(c(dJdt, dAdt))) # the result as a vector in a list

}

# call the ode function to solve the differential equation:

out <- ode(y = ja0, func = JA\_sys, times = timevec, parms = P)

time <- out[,'time'] # extract the time vector

j <- out[,'j'] # extract the juvenile population size vector

a <- out[,'a'] # extract the adult population size vector

# first population sizes over time:

plot( time, j , type='l', col='blue' , ylab='Population size', ylim=c(0,300))

lines( time, a, type='l', col='red')

legend('bottomright', legend=c('juvenile','adult'), col=c('blue','red'), lty=1)

A graph of a number of children

Description automatically generated with medium confidence

Figure 9. Population size vs time

*3f) Extend the model to include cannibalism, i.e. that adults feed on the juveniles (a common behaviour among many fish, reptiles, and other groups). Motivate all model extensions! Feel free to make extra assumptions, if necessary. The Lotka-Volterra predator-prey equations may be a source of inspiration. (3p)*

I created an ode function to solve the differential equation. I added the term s.a.j is the rate

at which adult predators kill juvenile and the corresponding increase of fitness of adult predators is proportional to this term by the constant gamma ε (0, 1) (Robert Marik and Lenka Pribylova, 2006).

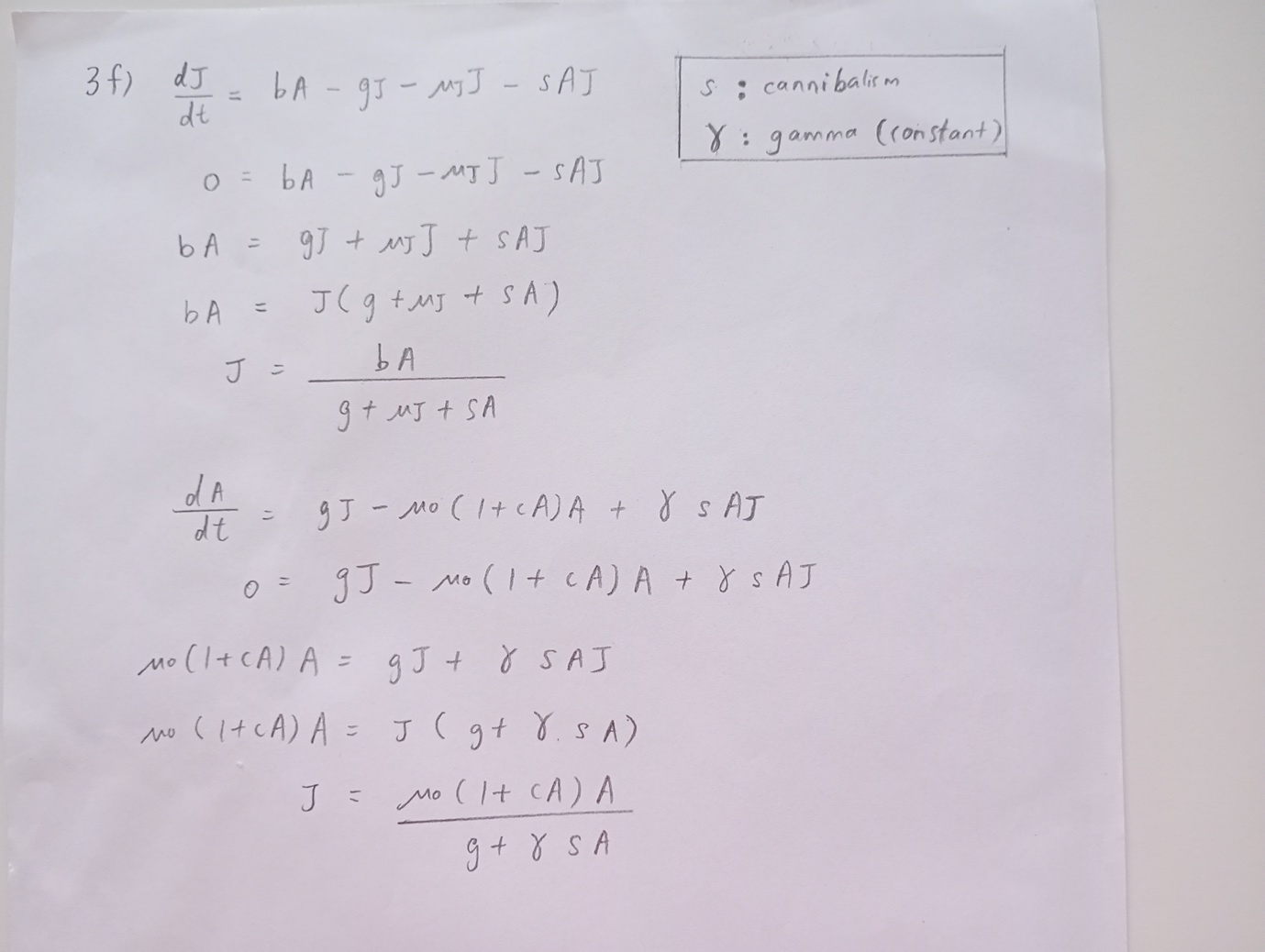


Figure 10. Answer for question 3f.

R script and here are the answers for question 3f. I created two separate R script, to check the population size over time and the isoclines in the phase space after including cannibalism.

R script (the isoclines in the phase space after including cannibalism)

**Doc.name: BIOS13\_examQ3f\_iso\_HooiMin.R**

rm(list = ls())

# Define system parameters first

pars <- list(b=1, g=0.5, mu\_j=0.3, mu\_0=0.2, c=0.01, s = 0.001, gamma = 0.001)

# Plotting J isocline, using A as a function of J:

A <- seq(0, 500) # a range of A-values

J <- pars$b \* A / (pars$g + pars$mu\_j + pars$s \* A)

plot(A, J, type="l", lty=5, col='blue', xlab='Adult (A)', ylab='Juvenile (J)',

ylim=c(0, 500), xlim=c(0, 500))

# Plotting A isocline

J <- pars$mu\_0 \* (1 + pars$c \* A) \* A / (pars$g + pars$gamma \* pars$s \* A)

lines(A, J, type="l", lty=5, col='red')

# Next, use ode to calculate a trajectory:

library(deSolve)

# The set of differential equations:

AJfunc <- function(t, AJ, pars) {

A <- AJ[1] # A is the first element of AJ

J <- AJ[2] # J is the second element of AJ

dAdt <- pars$g \* J - pars$mu\_0 \* A - pars$mu\_0 \* pars$c \* A^2 + pars$gamma \*

pars$s \* A \* J

dJdt <- pars$b \* A - pars$g \* J - pars$mu\_j \* J - pars$s \* A \* J

return(list(c(dAdt, dJdt)))

}

time\_vec <- seq(0, 500, 0.1) # Reducing the time range

AJ0 <- c(A = 5, J = 150) # Adjusted initial values for A and J

results <- ode(func = AJfunc, times = time\_vec, y = AJ0, parms = pars)

# Plot the result in phase space using the results data:

lines(results[, "A"], results[, "J"], type='l', col='black')

legend('bottomright', legend=c('J isocline', 'A isocline', 'Trajectory'),

col=c('blue', 'red', 'black'), lty=c(5, 5, 1))

A graph of a person with a line

Description automatically generated with medium confidence

Figure 11. J and A isoclines in the phase plane together with the trajectory after including cannibalism

R script (population size over time after including cannibalism)

**Doc.name: BIOS13\_examQ3f\_HooiMin.R**

rm(list = ls())

library(deSolve)

# Define the function with cannibalism

JA\_sys\_with\_cannibalism <- function(t, ja, P) {

# Extract vector content

j <- ja[1] # First element of the vector

a <- ja[2] # Second element of the vector

# # Calculate the age-structured rates with cannibalism

dJdt <- P$b \* a - P$g \* j - P$mu\_j \* j - P$s \* a \* j

dAdt <- P$g \* j - P$mu\_0 \* (1 + P$c \* a) \* a + P$gamma \* P$s \* a \* j

return(list(c(dJdt, dAdt))) # The result as a vector in a list

}

# Define parameters including cannibalism rate

P\_with\_cannibalism <- list(b = 1, g = 0.5, mu\_j = 0.3, mu\_0 = 0.2, c = 0.01,

s = 0.001, gamma = 0.001)

# Set initial population sizes

ja0 <- c(j = 150, a = 5)

# Time vector

timevec <- seq(0, 100, by = 1)

# Solve the differential equation with cannibalism

out\_with\_cannibalism <- ode(y = ja0, func = JA\_sys\_with\_cannibalism,

times = timevec, parms = P\_with\_cannibalism)

time <- out\_with\_cannibalism[,'time'] # Extract the time vector

j <- out\_with\_cannibalism[,'j'] # Extract the juvenile population size vector

a <- out\_with\_cannibalism[,'a'] # Extract the adult population size vector

# Plot the results

plot(time, j , type='l', col='blue' , ylab='population size', ylim=c(0,200))

lines(time, a, type='l', col='red')

legend('center', legend=c('juvenile','adult'), col=c('blue','red'), lty=1)

A graph with numbers and lines

Description automatically generated

Figure 12. Population size vs time after including cannibalism