Modelling Biological Systems

BIOS13 **Student:** Chandrashekar CR, ch1131ch-s@student.lu.se

Lecturer: Mikael Pnotarp, mikael.pontarp@biol.lu.se

> **Problem 2: The dynamics of a spatially structured population (8p)**
>
> First consider a single population that follows the dynamics,
>
> $$\frac{dn_1}{dt} = (r_1 - kn_1^2)n_1$$
>
> where:
>
> - n_1 is the population size,
> - t is time,
> - r_1 is the intrinsic growth rate,
> - k represents the strength of density dependence,
> - All parameters are positive, i.e., larger than zero.
>
> a) What is the (non-trivial) equilibrium population size? (1p)

2a) To find the (non trivial) equilibrium population size, we will equate the rate of change of population size to zero. This is because we know that at equilibrium there will be no change in the population size.

$$\frac{dn_1}{dt} = 0 \Rightarrow (r_1 - kn_1^2)n_1 = 0 \tag{1}$$

$$\begin{gathered}\text{Trivial Solution:}\\ n_1 = 0\end{gathered} \tag{2}$$

$$\begin{gathered}\text{Non-Trivial Solution:}\\ (r_1 - kn_1^2) = 0\end{gathered} \tag{3}$$

$$\begin{gathered}\text{Therefore:}\\ n_1 = n_{eq} = \pm\sqrt{\frac{r_1}{k}}\end{gathered} \tag{4}$$

Since n_{eq} (population size at equilibrium) represents a population size, it must be non-negative and therefore we discard the negative root. Thus, the non-trivial equilibrium population size is $n_{eq} = \sqrt{\frac{r_1}{k}}$. The equilibrium size n_{eq} is determined by the intrinsic growth rate r_1 and the strength of density dependence k.

- `r` **(Intrinsic Growth Rate):** This represents the population's potential for growth under ideal conditions (unlimited resources, no competition). A higher `r` means the population can grow faster.
- `k` **(Strength of Density Dependence):** This represents how strongly the population growth rate is reduced as the population density increases. A higher `k` means that the population's growth is more sensitive to crowding and competition.

- $\sqrt{\frac{r}{k}}$**:** This formula indicates that the equilibrium population size is determined by the ratio of the intrinsic growth rate (`r`) to the strength of density dependence (`k`). Let's break down the implications:
  - **Higher `r`, Higher `n`:** If the intrinsic growth rate (`r`) is high, the population has a greater capacity to grow. Therefore, it can reach a larger equilibrium size before density-dependent factors (represented by `k`) become strong enough to balance the growth.
  - **Higher `k`, Lower `n`:** If the strength of density dependence (`k`) is high, even a relatively small increase in population size will significantly reduce the growth rate due to increased competition for resources or other limiting factors. This results in a lower equilibrium population size.

> **Problem 2: The dynamics of a spatially structured population (8p)**
>
> b) Show that it is a stable equilibrium. (1p)

2b) To demonstrate the stability of the equilibrium, the behavior of the system is investigated in the vicinity of the equilibrium point. Specifically, it is examined whether the population converges towards the equilibrium over time. This was achieved by analyzing the derivative of the ordinary differential equation (ODE) that governs the dynamics of the population. The stability of the equilibrium is confirmed if the derivative evaluated at the equilibrium point is negative. This indicates that a small perturbation from the equilibrium will result in a restoring force that drives the population back towards the equilibrium point.

$$\frac{dn_1}{dt} = f(n_1) = (r_1 - kn_1^2)n_1 \tag{5}$$

$$f'(n_1) = \frac{d((r_1 - kn_1^2)n_1)}{dn_1} = \frac{d(r_1 n_1)}{dn_1} - \frac{d(kn_1^3)}{dn_1} \tag{6}$$

$$f'(n_1) = r_1 - 3kn_1^2 \tag{7}$$

We know that $n_{eq} = \sqrt{\frac{r_1}{k}}$. Substituting it in the above equation in place of n_1, we get: (8)

$$f'(n_{eq}) = r_1 - 3k\frac{r_1}{k} \tag{9}$$

$$f'(n_{eq}) = -2r_1 \tag{10}$$

We know that r_1, the intrinsic growth factor is always greater than zero. Therefore, the derivative of $f'(n_{eq})$ is always less than zero. This indicates that the equilibrium $n_{eq} = \sqrt{\frac{r_1}{k}}$ is stable.

> **Problem 2: The dynamics of a spatially structured population (8p)**
>
> c) Write a script in R that plots $\frac{dn}{dt}$ as a function of n. Choose the parameter values yourself.

2c) This script visualizes density-dependent population growth by plotting the rate of population change ($\frac{dn}{dt}$) against population size (n). It performs the following steps:

1. **Clearing and Loading:** Clears the workspace and loads the ggplot2 library (installing it if necessary).
2. **Parameter Initialization:** Defines the intrinsic growth rate (r) and the strength of density dependence (k).
3. **Population Size Range:** Creates a sequence of population sizes (n) from 0 to 20.
4. **Population Dynamics Equation:** Defines a function `pop_eqn(n)` that calculates dn/dt based on the equation incorporating r and k.
5. **Calculating dn/dt:** Calculates dn/dt for each population size in the defined range.
6. **Creating a Data Frame:** Creates a data frame popdata with columns for n and dn/dt.
7. **Generating the Plot:** Uses ggplot2 to plot dn/dt against n, adding a horizontal line at y = 0 to represent equilibrium points. The plot is labeled and formatted for clarity. Saves the plot as a high-resolution image and displays it in the R console.

```r
# This script plots the rate of change of population as a function of population size n.

# Clear environment
rm(list = ls())

# Load necessary libraries
if(!require(ggplot2)){install.packages("ggplot2")}
library(ggplot2)

# Parameter Initialization
r = 1     # Intrinsic growth rate (rate of growth when resources are unlimited)
k = 0.02  # Strength of density dependence (how much the growth rate is affected by
    population density)

# Population size range (n)
# Creates a sequence of population sizes from 0 to 20 with increments of 0.1.
n = seq(0, 20, by = 0.1)

# Population dynamics equation (defining a function is a good practice for reusability)
pop_eqn = function(n) {
  # This equation describes the rate of change of population (dn/dt) as a function of
      population size (n).
  return(r - (k * n^2)) * n
}

# Rate of change of population (dn/dt)
dndt = pop_eqn(n)

# Create a data frame for plotting (ggplot2 works best with data frames)
popdata = data.frame(n = n, dndt = dndt)
# Create a data frame for eqb line
eqb_line = data.frame(yintercept = 0)

# Create the plot using ggplot2
pop_growth_dynamics = ggplot(data = popdata, aes(x = n, y = dndt)) +
  geom_line(aes(color = "Population Dynamics"), linewidth = 1.2) + # Plots the line
      representing the population dynamics
  geom_hline(data = eqb_line, aes(yintercept = yintercept, color = "Equilibrium Points"),
      linetype = "dashed", linewidth = 1) + # Adds a horizontal red dashed line at y = 0.
      This line represents the equilibrium points where the population is not changing
      (dn/dt = 0).
  scale_color_manual(
    name = "Legend",
    values = c("Population Dynamics" = "blue",
               "Equilibrium Points" = "red")
  ) +
  labs(
    title = "Population Growth Dynamics",
    x = "Population Size (n)",
    y = "Rate of Change (dn/dt)",
    caption = paste("Model: dn/dt = (r - k * n^2) * n\n", # Using paste for better caption
        formatting
                    "Parameters: r =", r, ", k =", k) # Include parameter values in caption
                        for clarity
  ) +
  theme_classic() + # Use a clean theme
  theme(
    plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 18), # Center
        and style the title
    axis.title = element_text(face = "bold", size = 16), # Style axis titles
    axis.text = element_text(size = 14), # Style axis text
    legend.text = element_text(size = 12), # Style legend text
    plot.caption = element_text(hjust = 0, size = 12), # Style caption
    panel.grid.major = element_blank(), # Remove major grid lines for a cleaner look
    panel.grid.minor = element_blank() # Remove minor grid lines
  )

# Save the plot (using relative path and specifying units and DPI is good practice)
ggsave(plot = pop_growth_dynamics, filename =
    "./scripts/final_exam/exam_plots/population_dynamics.png", width = 200, height = 150,
    units = "mm", dpi = 600) # Added height parameter to ggsave

# Print the plot to the console (useful for interactive sessions)
print(pop_growth_dynamics)
```

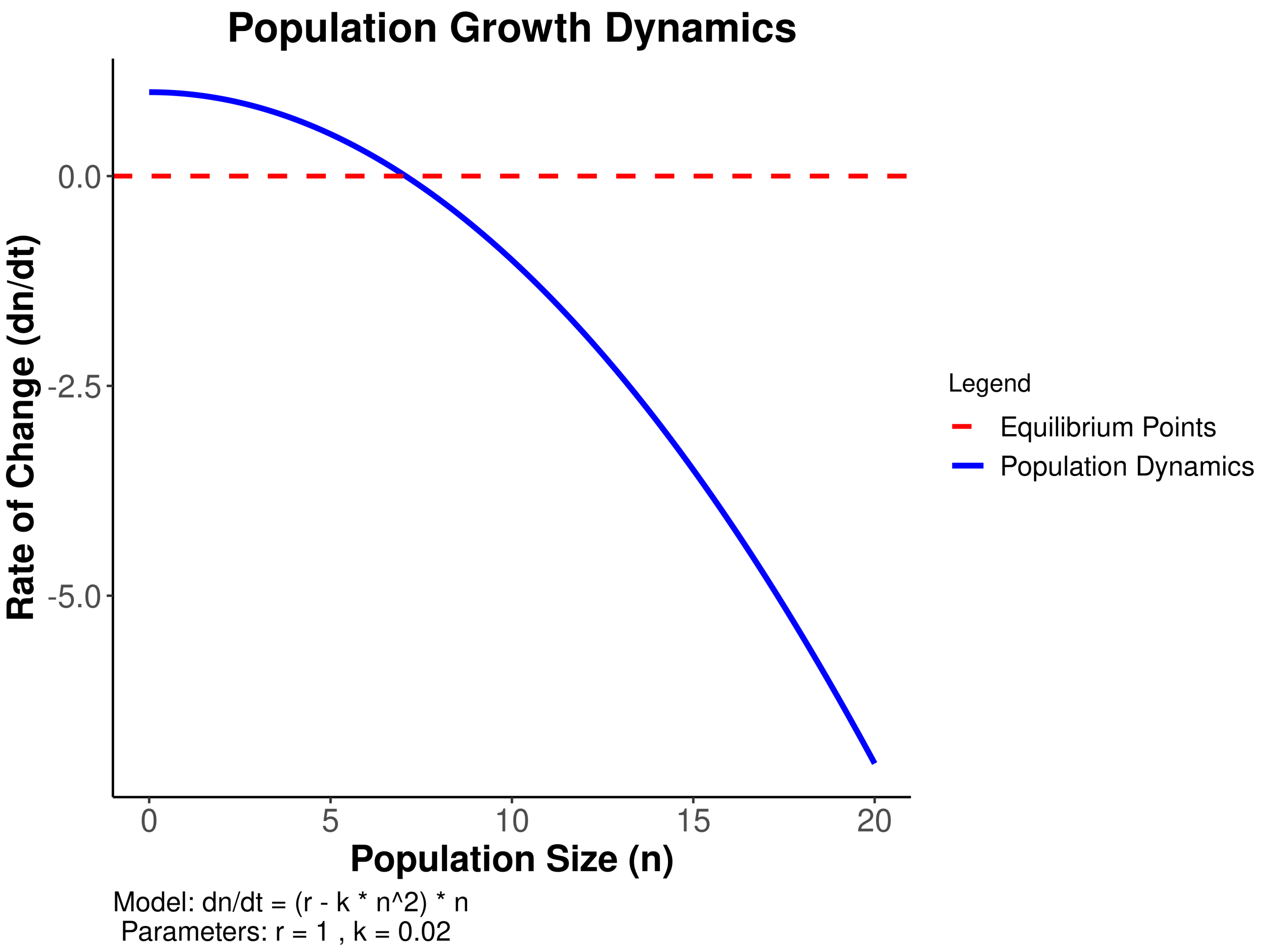


Figure 1: Population growth dynamics showing the rate of change against population size.

Now assume there is a second population of the same species that has the dynamics

$$\frac{dn_2}{dt} = (r_2 - kn_2^2)n_2 \tag{11}$$

Further assume that there is one-direction dispersal (one can think of drift down a stream or in prevailing wind direction) from population 1 to population 2, such that a proportion m of population 1 disperses to population 2 per unit unit. In other words, individuals disperse from population 1 to population 2 at a rate mn_1 per unit time unit.

> **Problem 2: The dynamics of a spatially structured population (8p)**
>
> d) Write down the new equation for the dynamics of n_1 and n_2, given the migration described above. (1p)

2d) To incorporate one-directional dispersal from population n_1 to population n_2, we modify the equations for the dynamics of n_1 and n_2 to include the migration term mn_1.

For n_1: The population n_1 loses individuals at a rate mn_1. Therefore the equation becomes:

$$\begin{gathered} \frac{dn_1}{dt} = (r_1 - kn_1^2)n_1 - mn_1 \\ m > 0\text{: Proportion of individuals dispersing from } n_1 \text{ to } n_2 \text{ per unit time.} \end{gathered} \tag{12}$$

For n_2: The population n_2 gains individuals from n_1 at a rate mn_1. The equation for n_2 becomes:

$$\begin{gathered} \frac{dn_2}{dt} = (r_2 - kn_2^2)n_2 + mn_1 \\ m > 0 : \text{ Proportion of individuals dispersing from } n_1 \text{ to } n_2 \text{ per unit time.} \end{gathered} \tag{13}$$

> **Problem 2: The dynamics of a spatially structured population (8p)**
>
> e) What is the new equilibrium of population 1? (1p)

2e) To find the new equilibrium for population n_1 under one-directional dispersal, we need to solve the modified equation for n_1 when $\frac{dn_1}{dt} = 0$. The equation is as follows:

$$\begin{gathered} \frac{dn_1}{dt} = (r_1 - kn_1^2)n_1 - mn_1 = 0 \\ = n_1[(r_1 - kn_1^2) - m] = 0 \\ \text{Trivial Solution: } n_1 = 0 \\ \text{This is when the population is zero, and does not make any sense in our analysis.} \\ \text{Non-Trivial Solution: } (r_1 - kn_1^2) - m = 0 \\ r_1 - kn_1^2 = m \\ kn_1^2 = r_1 - m \end{gathered} \tag{14}$$

$$n_1 = n_{new_{eq}} = \pm\sqrt{\frac{r_1 - m}{k}}$$
$$\text{We will ignore the negative root} \tag{15}$$
$$n_{new_{eq}} = \sqrt{\frac{r_1 - m}{k}}$$

Some important points about this equilibrium is, $r_1 > m$. If the dispersal rate m is too high relative to the intrinsic growth rate r_1, the equilibrium population n_1 goes closer to extinction. (See next question (2f))

> **Problem 2: The dynamics of a spatially structured population (8p)**
>
> f) Population 1 loses a lot of individuals through this one-directional dispersal. Above what value of m does population 1 go extinct? (1p)

2f) The domain of the square root function is only defined for non-negative values. The new equilibrium population size is given by $n_{new_{eq}}$ is given by:

$$n_{new_{eq}} = \sqrt{\frac{r_1 - m}{k}}$$
$$\text{For } n_{new_{eq}} \text{ to be real and valid, the term inside the square root, } \sqrt{\frac{r_1 - m}{k}} \text{ must be non-negative:} \tag{16}$$
$$\sqrt{\frac{r_1 - m}{k}} \geq 0$$

Since $k > 0$ (a carrying capacity is always positive), the inequality simplifies to:

$$r_1 - m \geq 0 \implies r_1 \geq m \tag{17}$$

1. **Case** $r_1 - m = 0$: If $r_1 = m$, the equilibrium population size becomes:

   $$n_{new_{eq}} = \sqrt{\frac{0}{k}} = 0 \tag{18}$$

   A population size of zero biologically corresponds to extinction. This, when $r_1 = m$, Population 1 goes extinct.

2. **Case** $r_1 - m < 0$: If $r_1 < m$, the term $r_1 - m$ becomes negative making $\sqrt{\frac{r_1 - m}{k}}$ undefined in the real number domain. Biologically, this means that Population 1 cannot sustain itself, as a negative or imaginary population size has no biological meaning. Therefore, Population 1 is extinct.

3. **Case** $r_1 - m > 0$: If $r_1 > m$, the population size $n_{new_{eq}}$ is positive and biologically valid. In this case, Population 1 can persist.

Therefore, population 1 goes extinct when $m \geq r_1$. The critical threshold is $m = r_1$, beyond which the one-directional dispersal rate m causes the population size to drop to zero or become, undefined, resulting in extinction (i.e **Case 2** and**Case 1**).

> **Problem 2: The dynamics of a spatially structured population (8p)**
>
> g) Write an R script that simulate the coupled dynamics of n_1 and n_2 and plots them as function of time. (2p)

2g) This script models the dynamics of two interacting populations, considering intrinsic growth rates, density dependence, and one-way dispersal. The results are visualized using a clean and customizable plot created with `ggplot2`. Compared to single-population models, this approach incorporates interactions and provides a detailed visualization of their combined dynamics. The script is structured as follows:

1. **Setting Up the Environment**

   (a) **Clearing the Workspace:** All objects from the R workspace were removed to prevent conflicts with existing variables.

   (b) **Loading Necessary Libraries:** The script checked for and installed any missing libraries (`ggplot2`, `deSolve`, and `tidyr`) before loading them. These libraries were essential for solving differential equations, managing data, and creating visualizations.

2. **Defining the Coupled Dynamics Function**
   The function `coupled_dynamics` modeled the interactions between two populations over time. The function parameters were:

   (a) `t:` Current time point in the simulation.

   (b) `state:` A list containing the population sizes of `n1` and `n2`.

   (c) `params:` A list of model parameters.

3. **Extracting Parameters and State Variables**
   The function retrieved specific parameters from the `params` list:

   (a) `r1` and `r2:` Intrinsic growth rates of populations 1 and 2.

   (b) `k:` Strength of density dependence, regulating population growth based on density.

   (c) `m:` Dispersal rate from population 1 to population 2.

   Current population sizes (`n1` and `n2`) were extracted from the `state` list. Two differential equations were defined to calculate the rate of change for each population:

   - `dn1dt:` The change in population 1 considered intrinsic growth, density dependence, and loss due to dispersal to population 2.
   - `dn2dt:` The change in population 2 included intrinsic growth, density dependence, and gain from dispersal from population 1.

   The function returned a list of the calculated rates of change (`dn1dt` and `dn2dt`).

4. **Setting Model Parameters and Initial Conditions**
   The following parameters were set:

   - `r1 = 1:` Growth rate of population 1.
   - `r2 = 0.6:` Growth rate of population 2.
   - `k = 0.02:` Density dependence.

    - `m = 0.1`: Dispersal rate from population 1 to 2.

    Initial population sizes were set as:

    - `n1 = 10`: Population size of species 1.
    - `n2 = 3`: Population size of species 2.

5. **Simulating Population Dynamics**
   The `ode` function from the `deSolve` package was used to solve the differential equations numerically. The output was a matrix containing the population sizes at each time point.

6. **Preparing Data for Plotting**

   (a) **Converting to a Data Frame:** The simulation output matrix was converted into a data frame for easier manipulation.

   (b) **Reshaping Data for Plotting:** The `pivot_longer` function transformed the data into a ”long” format suitable for `ggplot2`. Population size columns (`n1` and `n2`) were combined into a single `Size` column, and a new `Population` column identified the corresponding population.

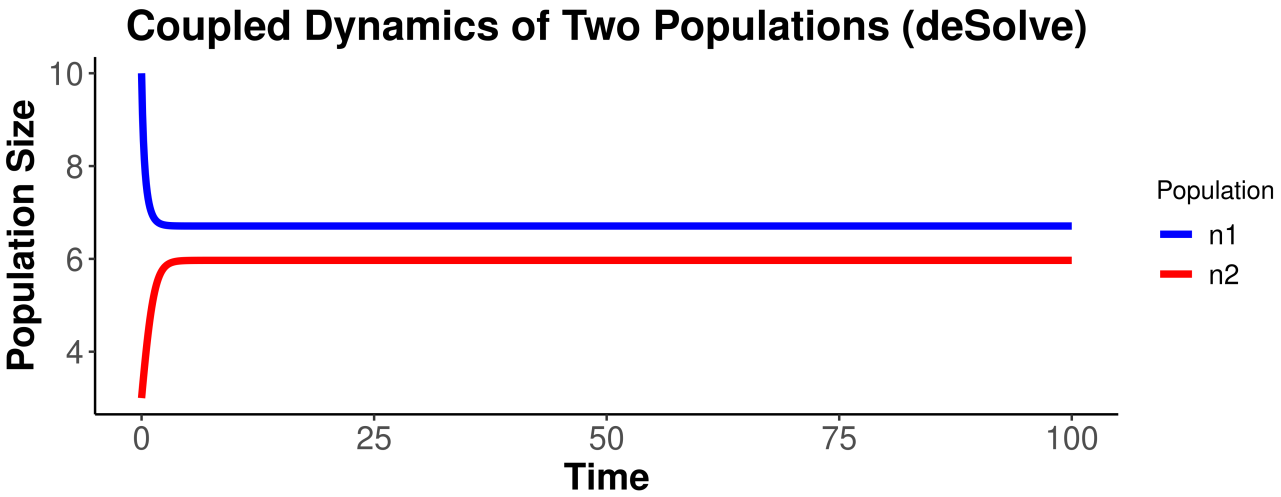
7. **Creating the Plot**
   The plot was created using `ggplot2`:

   - The `time` variable was mapped to the x-axis and the population size (`Size`) to the y-axis.
   - The `Population` variable determined the line color.
   - Colors were customized using `scale_color_manual` (`blue` for `n1` and `red` for `n2`).
   - Titles, axis labels, and captions were added using `labs`. The caption detailed the model equations, parameters, and initial conditions.
   - `theme_classic` and additional theme settings were used for a clean layout and aesthetic improvements.

8. **Saving and Displaying the Plot**
   The plot was saved as a high-resolution PNG file (`200 mm` wide, `600 dpi`) using `ggsave`. It was also displayed in the R console using `print(coupled_dynamics_plot)`.

```r
# This script simulates the coupled dynamics of n1 and n2 and plots them as function of time.

# Clear the environment
rm(list = ls())

# Import necessary libraries
# Check if libraries are installed, install if not
if (!require(ggplot2)) {
  install.packages("ggplot2")
}
if (!require(deSolve)) {
  install.packages("deSolve")
}
if (!require(tidyr)) {
  install.packages("tidyr")
}
```

```
library(deSolve) # Load package for solving differential equations
library(ggplot2) # Load package for creating plots
library(tidyr)  # Load package for data manipulation

# Define a function for the coupled population dynamics
coupled_dynamics = function(t, state, params) {
  # Unpack the parameters from the 'params' list
  r1 = params["r1"] # Intrinsic growth of population 1
  r2 = params["r2"] # Intrinsic growth of population 2
  k = params["k"] # Strength of density dependence
  m = params["m"] # One directional dispersal rate from population 1 to population 2
      (proportion)

  # Access state variables from the 'state' list
  n1 = state["n1"]
  n2 = state["n2"]

  # Define the population dynamics equations
  # dn1dt: rate of change of population 1
  # dn2dt: rate of change of population 2
  dn1dt = (r1 - k * n1^2) * n1 - m * n1
  dn2dt = (r2 - k * n2^2) * n2 + m * n1

  # Return a list containing the rates of change
  return(list(c(dn1dt, dn2dt)))
}

# Set the parameters for the model
params = c(r1 = 1, r2 = 0.6, k = 0.02, m = 0.1)

# Define the initial population sizes for each population
initial_state = c(n1 = 10, n2 = 3)

# Define the time steps for the simulation
timevec = seq(0, 100, by = 0.1) # Sequence from 0 to 100 with steps of 0.1

# Solve the differential equations using the 'ode' function from deSolve
output = ode(y = initial_state, times = timevec, func = coupled_dynamics, parms = params)

# Convert the output from the 'ode' function to a data frame
output_df = as.data.frame(output)

# Reshape the data frame for easier plotting with ggplot2
# tidyr's pivot_longer creates separate columns for n1 and n2
population_data = pivot_longer(output_df, cols = c("n1", "n2"), names_to = "Population",
    values_to = "Size")

# Create the plot using ggplot2
coupled_dynamics_plot = ggplot(data = population_data, aes(x = time, y = Size, color =
    Population)) +
  scale_color_manual(values = c("n1" = "blue", "n2" = "red")) + # Set colors for each
      population
  geom_line(linewidth = 1.5) +                            # Plot lines for each population
  labs(
    title = "Coupled Dynamics of Two Populations (deSolve)",
    x = "Time",
    y = "Population Size",
    caption = paste(
      "Model: dn1/dt = (r1 - k*n1^2)n1 - m*n1",
      "\ndn2/dt = (r2 - k*n2^2)*n2 + m*n1",
      "\nParameters: r1 = 1, r2 = 0.6, k = 0.02, m = 0.1",
      "\nn1 =10, n2 = 3",
      sep = "\n"
    )  # Combine model equations, parameters, and initial conditions in caption
  ) +
  theme_classic() +                                         # Use a classic theme
  theme(
    plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 18), # Center
        and style title
    axis.title = element_text(face = "bold", size = 16),                  # Style axis titles
    plot.caption = element_text(hjust = 0, size = 12),                 # Style caption
    legend.text = element_text(size=12),                # Hide minor grid lines
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.text = element_text(size=14)
  )


# Save the plot as a high resolution image
ggsave(plot = coupled_dynamics_plot,filename =
    "./scripts/final_exam/exam_plots/coupled_dynamics.png", width = 200, units = "mm" ,
    dpi=600)

# Display the plot
print(coupled_dynamics_plot)
```



Model: dn1/dt = (r1 - k*n1^2)n1 - m*n1

dn2/dt = (r2 - k*n2^2)*n2 + m*n1

Parameters: r1 = 1, r2 = 0.6, k = 0.02, m = 0.1

n1 =10, n2 = 3

Figure 2: Coupled dynamics of two population with respect to time.