Differential Equations Model in R

Frank Zhong

2024-11-28

Differential Equations Model in R

Differential equations are mathematical tools used to model relationships between changing quantities and their rates of change. They are fundamental in describing dynamic systems across various fields, including physics, biology, economics, and engineering. A differential equation expresses how a dependent variable changes concerning an independent variable, often time. By solving these equations, one can predict the behavior of systems ranging from the growth of populations to the motion of celestial bodies or the spread of diseases.

In modeling, differential equations help capture the essence of complex phenomena by representing continuous processes in terms of simple relationships. For instance, in epidemiology, they describe the dynamics of infection transmission, while in physics, they govern laws like Newton's and Maxwell's equations. Whether ordinary (ODEs) or partial (PDEs), differential equations form a bridge between theoretical principles and real-world applications, making them invaluable for prediction.

In this article, we will focus on four classic differential equations models: the Malthusian Population Model, the Logistic Growth Model, the Lotka-Volterra Predator-Prey Model and the Lotka-Volterra Species Competition Model.

We library all the packages required.

```r
library(deSolve) # for solving ODEs
library(ggplot2)
library(tidyverse)
```

Malthusian Population Model

The **Malthusian Model** describes exponential population growth, where the population increases at a constant rate without any limiting factors. It assumes unlimited resources and no competition, resulting in a constant growth rate. The model is governed by the differential equation:

$$\frac{dP}{dt} = rP$$

Where:
P is the population size;

r is the intrinsic growth rate;
$\frac{dP}{dt}$ is the rate of change in population.

This model leads to exponential growth:

$$P(t) = P_0 e^{rt}$$

Where P_0 is the initial population.

Logistic Growth Model

The **Logistic Growth Model** extends the Malthusian model by introducing a carrying capacity K, reflecting limited resources. As the population grows, the growth rate slows down and approaches the carrying capacity. The differential equation is:

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$

Where:
P is the population size;
r is the intrinsic growth rate;
K is the carrying capacity.

The solution leads to an S-shaped curve, where the population grows rapidly at first and then slows as it approaches K.

Lotka-Volterra Predator-Prey Model

The **Lotka-Volterra Predator-Prey Model** describes the dynamics between predator and prey species. It assumes that the prey population grows exponentially in the absence of predators, while the predator population depends on the availability of prey for food. The system of differential equations is:

$$\frac{dP}{dt} = \alpha P - \beta PQ$$

$$\frac{dQ}{dt} = \delta PQ - \gamma Q$$

Where:
P is the prey population;
Q is the predator population;
α is the growth rate of the prey;
β is the predation rate;
δ is the rate of predator reproduction per prey eaten;
γ is the death rate of predators.

These equations describe oscillatory dynamics, where the prey population increases, followed by an increase in predators, which then leads to a decline in prey, followed by a decline in predators, and the cycle repeats.

Lotka-Volterra Species Competition Model

The **Lotka-Volterra Species Competition Model** describes the interactions between two species competing for limited resources. Each species has its own growth rate, and the competition for resources affects the growth rates of both species. The system of equations is:

$$\frac{dN_1}{dt} = r_1 N_1 \left(1 - \frac{N_1 + \alpha_{12} N_2}{K_1}\right)$$

$$\frac{dN_2}{dt} = r_2 N_2 \left(1 - \frac{N_2 + \alpha_{21} N_1}{K_2}\right)$$

Where:
N_1 and N_2 are the populations of the two species;
r_1 and r_2 are the intrinsic growth rates; K_1 and K_2 are the carrying capacities of each species;
α_{12} is the competition coefficient representing the effect of species 2 on the growth of species 1;
α_{21} is the competition coefficient representing the effect of species 1 on the growth of species 2.

This model can lead to competitive exclusion (one species out-competing the other), coexistence, or oscillatory dynamics depending on the values of the coefficients.

Example 1: U.S. Population (1790-2000)

The following table presents the population of the United States in millions, based on decennial census data from 1790 to 2000.

Year	Population (millions)	Year	Population (millions)
1790	3.9	1900	76.0
1800	5.3	1910	92.0
1810	7.2	1920	106.5
1820	9.6	1930	123.2
1830	12.9	1940	131.7
1840	17.1	1950	150.7
1850	23.2	1960	179.3
1860	31.4	1970	204.0

Year	Population (millions)	Year	Population (millions)
1870	38.6	1980	226.5
1880	50.2	1990	251.4
1890	62.9	2000	281.4

For this population, we use the Malthusian Population Model and the Logistic Growth Model to fit and make predictions. Firstly, input this data and construct the models.

```r
population_data <- data.frame(time = seq(0, 210, by = 10),
observed = c(3.9, 5.3, 7.2, 9.6, 12.9, 17.1, 23.2, 31.4, 38.6, 50.2, 62.
9, 76.0, 92.0, 106.5, 123.2, 131.7, 150.7, 179.3, 204.0, 226.5, 251.4,
281.4))

initial_population <- population_data$observed[1]

### Malthusian Population Model
# Define the Malthusian model as a function for the ODE solver
malthusian_model <- function(time, state, parameters) {
  with(as.list(c(state, parameters)), {
    dP <- r * P  # dP/dt = r * P
    list(c(dP))
  })
}

# Objective function for Malthusian model to minimize SSR
malthusian_objective_ode <- function(params, data) {
  r <- params[1]
  initial_state <- c(P = data$observed[1])  # Initial population size

  # Solve the Malthusian ODE
  times <- data$time
  solution <- ode(y = initial_state, times = times, func = malthusian_m
odel, parms = list(r = r))

  # Calculate the residual sum of squares (RSS)
  predicted <- solution[, "P"]
  observed <- data$observed
  ssr <- sum((observed - predicted)^2)

  return(ssr)
}

# Estimate the growth rate r for the Malthusian model using optim
malthusian_fit_ode <- optim(par = c(r = 0.02), fn = malthusian_objectiv
e_ode, data = population_data)
```

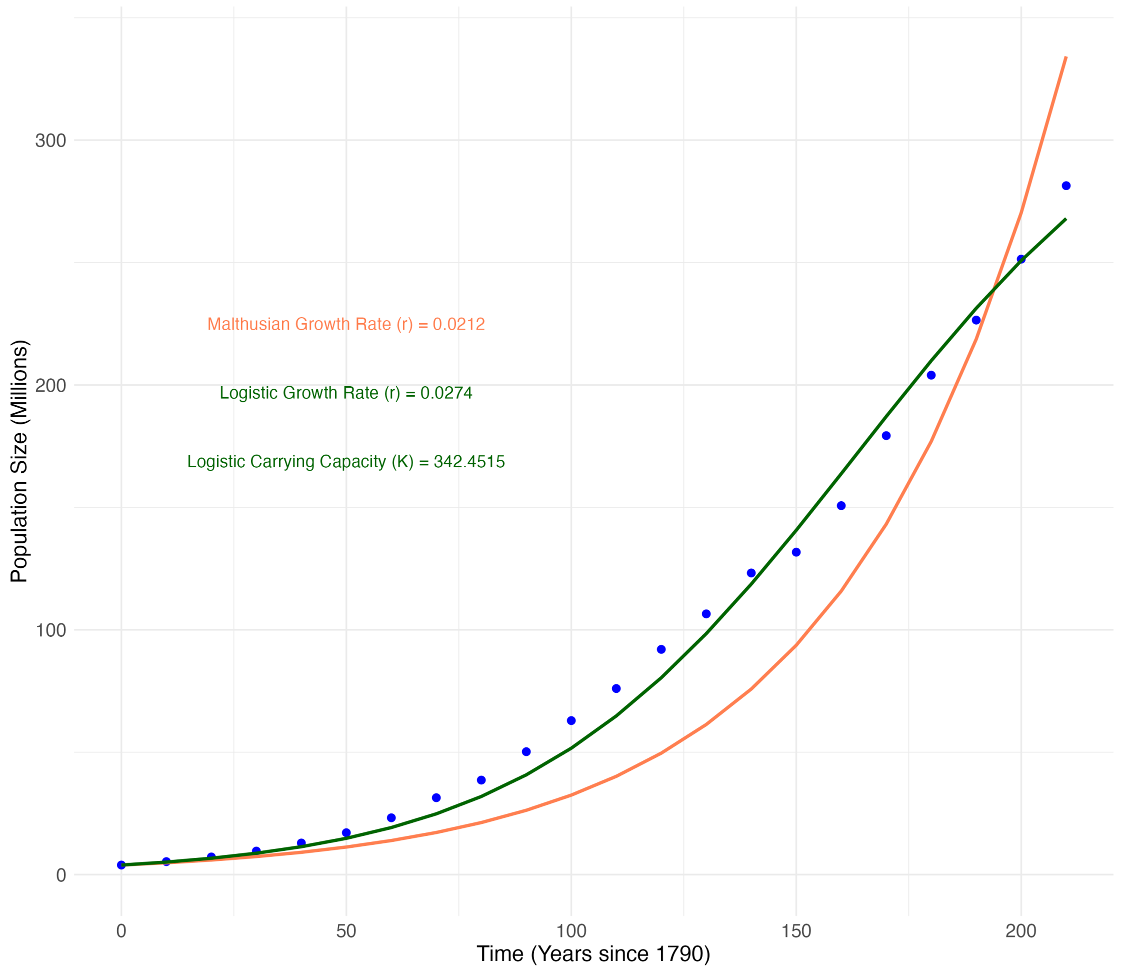
```
## Warning in optim(par = c(r = 0.02), fn = malthusian_objective_ode, d
ata = population_data): one-dimensional optimization by Nelder-Mead is
unreliable:
## use "Brent" or optimize() directly
```

```r
r_malthusian <- malthusian_fit_ode$par["r"]

# Solve the Malthusian ODE using the estimated parameter
malthusian_solution <- ode(y = c(P = initial_population), times = popul
ation_data$time, func = malthusian_model, parms = list(r = r_malthusia
n))

### Logistic Growth Model
# Define the Logistic model as a function for the ODE solver
logistic_model <- function(time, state, parameters) {
  with(as.list(c(state, parameters)), {
    dP <- r * P * (1 - P / K)  # dP/dt = r * P * (1 - P / K)
    list(c(dP))
  })
}

# Objective function for Logistic model to minimize SSR
logistic_objective_ode <- function(params, data) {
  r <- params[1]
  K <- params[2]
  initial_state <- c(P = data$observed[1])  # Initial population size

  # Solve the Logistic ODE
  times <- data$time
  solution <- ode(y = initial_state, times = times, func = logistic_mod
el, parms = list(r = r, K = K))

  # Calculate the residual sum of squares (RSS)
  predicted <- solution[, "P"]
  observed <- data$observed
  ssr <- sum((observed - predicted)^2)

  return(ssr)
}

# Initial guess for logistic model parameters
initial_guess_logistic <- c(r = 0.02, K = max(population_data$observed)
 * 1.2)

# Estimate the growth rate r and carrying capacity K for the Logistic m
odel using optim
logistic_fit_ode <- optim(par = initial_guess_logistic, fn = logistic_o
bjective_ode, data = population_data)
r_logistic <- logistic_fit_ode$par["r"]
```

```
K_logistic <- logistic_fit_ode$par["K"]

# Solve the Logistic ODE using the estimated parameters
logistic_solution <- ode(y = c(P = initial_population), times = populat
ion_data$time, func = logistic_model, parms = list(r = r_logistic, K =
K_logistic))
```

Then we can visualise the results.

```
# Create a data frame for plotting
malthusian_predicted <- data.frame(time = malthusian_solution[, "time"],
 predicted = malthusian_solution[, "P"])
logistic_predicted <- data.frame(time = logistic_solution[, "time"], pr
edicted = logistic_solution[, "P"])

rm(list=ls()[!ls() %in% c("malthusian_predicted", "logistic_predicted",
 "r_malthusian", "r_logistic", "K_logistic", "population_data", "malthu
sian_model", "logistic_model")])

# Plot the observed and model predictions
population.model.plot <- ggplot() +
  geom_point(data = population_data, aes(x = time, y = observed), color
 = "blue", size = 2) +
  geom_line(data = malthusian_predicted, aes(x = time, y = predicted),
color = "coral", linewidth = 1, linetype = "solid") +
  geom_line(data = logistic_predicted, aes(x = time, y = predicted), co
lor = "darkgreen", linewidth = 1, linetype = "solid") +
  labs(title = "",
       x = "Time (Years since 1790)", y = "Population Size (Millions)")
 +
  theme_minimal() +
  scale_y_continuous(limits = c(0, max(population_data$observed) * 1.2))
 +
  theme(legend.position = "bottom") +
  annotate("text", x = 50, y = max(population_data$observed) * 0.8,
           label = paste("Malthusian Growth Rate (r) =", round(r_malthu
sian, 4)), color = "coral") +
  annotate("text", x = 50, y = max(population_data$observed) * 0.7,
           label = paste("Logistic Growth Rate (r) =", round(r_logistic,
 4)), color = "darkgreen") +
  annotate("text", x = 50, y = max(population_data$observed) * 0.6,
           label = paste("Logistic Carrying Capacity (K) =", round(K_lo
gistic, 4)), color = "darkgreen") +
  theme(
    plot.title = element_text(size = 16),      # Title size
    axis.title.x = element_text(size = 14),    # X-axis title size
    axis.title.y = element_text(size = 14),    # Y-axis title size
    axis.text.x = element_text(size = 12),     # X-axis labels size
    axis.text.y = element_text(size = 12)      # Y-axis labels size
  )
```



Note that the Logistic Growth Model provides a much better fit than the Malthusian Population Model. The model fits are as follows:

Model	Parameter	Estimate
Malthusian Population Model	Growth rate r	0.0212
Logistic Growth Model	Growth rate r	0.0274
	Carrying capacity K	342.452

And we can also obtain the numeric prediction values for the next 30 years.

```
predict.time <- seq(0, 240, by = 10) # predict 30 more years
initial.population <- c(P = population_data$observed[1])

malthusian.prediction <- ode(y = initial.population, times = predict.ti
me, func = malthusian_model, parms = r_malthusian)
malthusian.prediction.df <- as.data.frame(malthusian.prediction)

logistic.prediction <- ode(y = initial.population, times = predict.time,
 func = logistic_model, parms = c(r_logistic, K_logistic))
```

```
logistic.prediction.df <- as.data.frame(logistic.prediction)

rm(list=ls()[!ls() %in% c("malthusian.prediction.df", "logistic.predict
ion.df", "population_data")])

prediction <- population_data %>%
  full_join(malthusian.prediction.df, by = "time") %>%
  full_join(logistic.prediction.df, by = "time") %>%
  rename(observed = observed,
         malthusian = P.x,
         logistic = P.y)
```

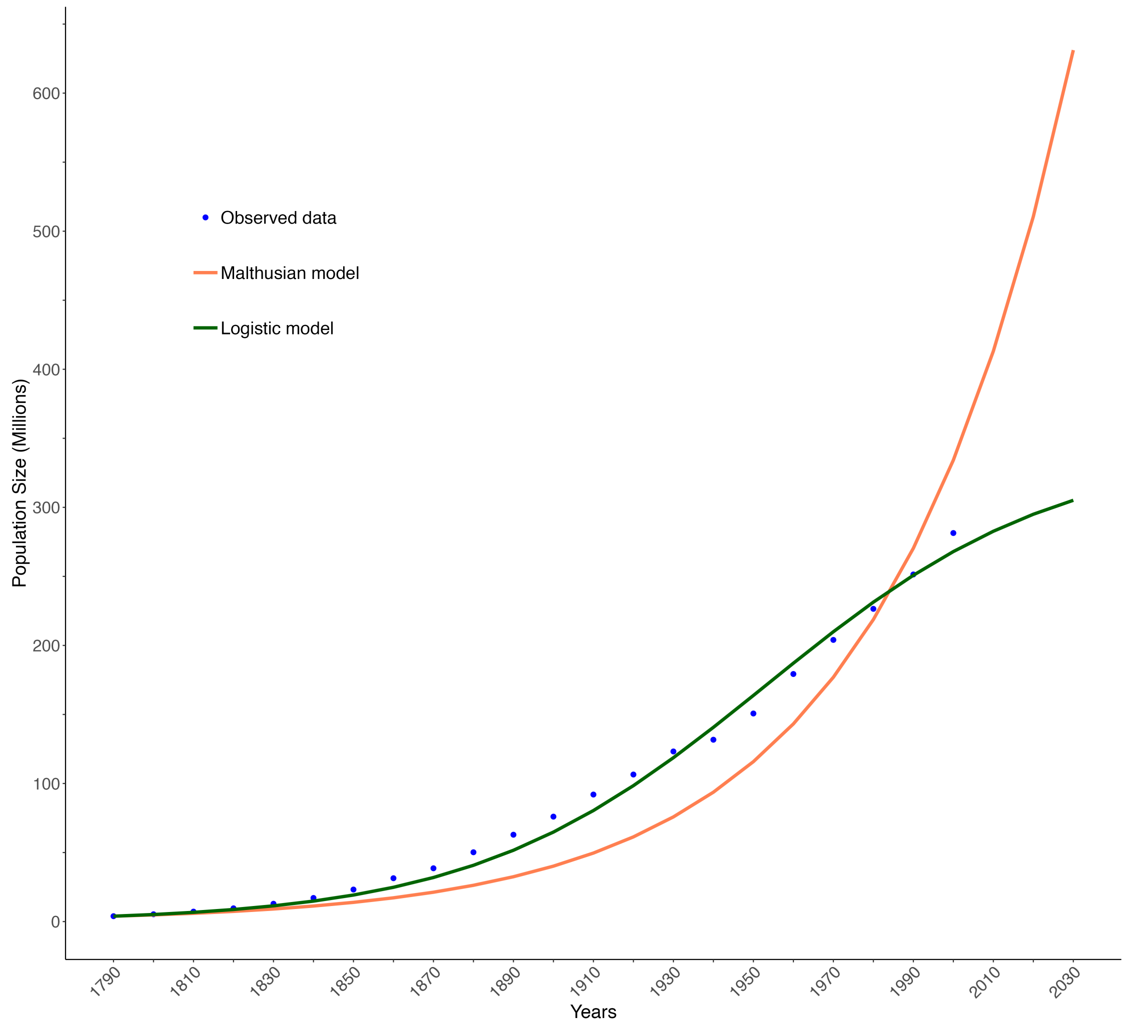
The predictions as summarised as follows:

Years	Malthusian Population Model prediction	Logistic Growth Model prediction
2010	413.1	282.7
2020	510.6	295.0
2030	631.2	305.1

Similarly, plot the predictions.

```
prediction.real <- prediction %>% mutate(time = 1790 + prediction$time)

population.prediction.plot <- ggplot() +
  geom_point(data = population_data, aes(x = time, y = observed), color
 = "blue", size = 2) +
  geom_line(data = prediction, aes(x = time, y = malthusian), color = "
coral", linewidth = 1, linetype = "solid") +
  geom_line(data = prediction, aes(x = time, y = logistic), color = "da
rkgreen", linewidth = 1, linetype = "solid") +
  labs(title = "",
       x = "Years since 1970", y = "Population Size (Millions)") +
  theme_minimal() +
  scale_y_continuous(limits = c(0, max(population_data$observed) * 1.2))
 +
  theme(legend.position = "bottom") +
  scale_x_continuous(breaks = seq(0, 240, by = 20)) +
  scale_y_continuous(breaks = seq(0, 650, by = 100)) +
  theme(
    axis.title.x = element_text(size = 18),
    axis.title.y = element_text(size = 18),
    axis.text.x = element_text(size = 16),
    axis.text.y = element_text(size = 16)
    )

## Scale for y is already present.
## Adding another scale for y, which will replace the existing scale.
```

```
population.prediction.plot <- population.prediction.plot + theme(axis.t
ext.x = element_text(angle = 45, hjust = 1))

xs <- 1810
ys <- 400
population.prediction.plot <- population.prediction.plot +
  annotate("text", x = xs+6.8, y = ys+110, label = "Observed data", col
or = "black", size = 6, hjust = 0) +
  annotate("point", x = xs+3, y = ys+110, color = "blue", size = 2) +
  annotate("text", x = xs+6.8, y = ys+70, label = "Malthusian model", c
olor = "black", size = 6, hjust = 0) +
  annotate("segment", x = xs, xend = xs+6, y = ys+70, yend = ys+70, col
or = "coral", linewidth = 1.5, linetype = "solid") +
  annotate("text", x = xs+6.8, y = ys+30, label = "Logistic model", col
or = "black", size = 6, hjust = 0) +
  annotate("segment", x = xs, xend = xs+6, y = ys+30, yend = ys+30, col
or = "darkgreen", linewidth = 1.5, linetype = "solid")
```



Example 2: Two Lotka-Volterra Models Combined

We are given a situation where three species, A, B, and C, interact in an ecosystem. Species A and B have a predator-prey relationship, while species A and C have a competitive relationship. We aim to construct a system of differential equations to model the population sizes of these species.

Let N_A, N_B, and N_C represent the population sizes of species A, B, and C, respectively. The system of differential equations is as follows:

1. **Species A (Prey and Competitor)**:

   The rate of change of species A depends on its intrinsic growth, the predation pressure from species B, and competition with species C:

   $$\frac{dN_A}{dt} = r_A N_A \left(1 - \frac{N_A}{K_A}\right) - \alpha N_A N_B - \beta N_A N_C$$

   r_A: Intrinsic growth rate of species A.
   K_A: Carrying capacity of species A.
   α: Predation rate coefficient between species A and B.
   β: Competition coefficient between species A and C.

2. **Species B (Predator)**:

   The rate of change of species B depends on the predation of species A and its own mortality rate:

   $$\frac{dN_B}{dt} = \delta N_A N_B - \gamma N_B$$

   δ: Conversion efficiency of consumed species A into species B growth.
   γ: Mortality rate of species B.

3. **Species C (Competitor)**:

   The rate of change of species C is affected by competition with species A:

   $$\frac{dN_C}{dt} = \mu N_C \left(1 - \frac{N_C}{K_C}\right) - \beta N_A N_C$$

   μ: Intrinsic growth rate of species C.
   K_C: Carrying capacity of species C.
   β: Competition coefficient between species A and species C.

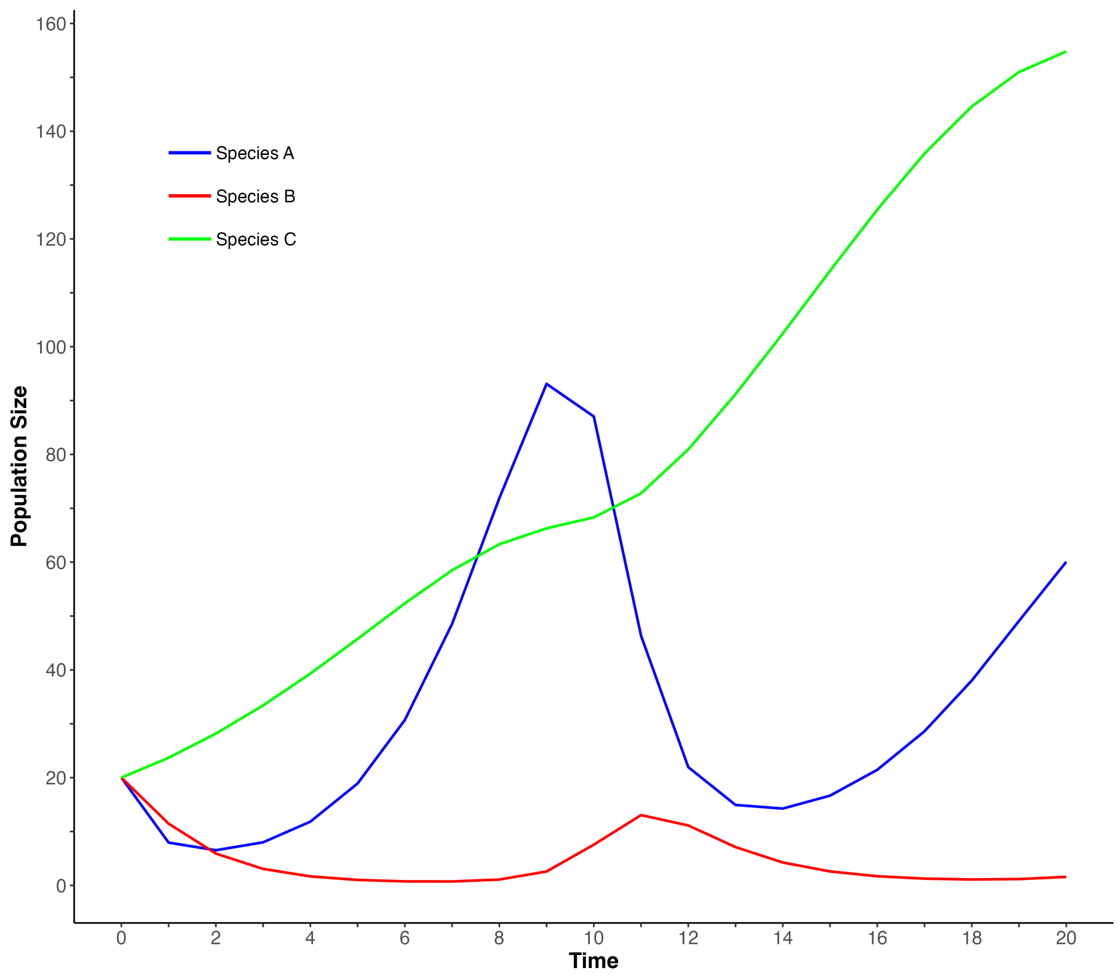
Explanation:

Predator-Prey Interaction: Species A (prey) is consumed by species B (predator), which affects their respective population growths.
Competition: Species A and C compete for the same resources, affecting their growth rates.
Growth Rates: r_A, δ, μ, and their respective carrying capacities K_A and K_C determine the ecosystem dynamics.

Then we can implement this model. The model parameters are given as:

Parameter	Description	Value
r_1	Intrinsic growth rate of species A	1
λ_1	Interaction coefficient between species A and B	0.1
σ_1	Competition coefficient between species A and C	0.5
N_1	Carrying capacity of species A	500
r_2	Intrinsic growth rate of species B	0.5
λ_2	Interaction coefficient between species A and B	0.02
r_3	Intrinsic growth rate of species C	0.2
σ_3	Competition coefficient between species A and C	2
N_3	Carrying capacity of species C	300
e	Environmental factor affecting all species	0.3

```r
# Define the Lotka-Volterra + Species Competition Model
model <- function(time, state, parameters) {
  with(as.list(c(state, parameters)), {
    dx1 <- x1 * (r1 - lamda1 * x2 - e) - r1 * x1 * (x1 / N1 + sigma1 * 
x3 / N3)
    dx2 <- x2 * (-r2 + lamda2 * x1 - e)
    dx3 <- r3 * x3 * (1 - x3 / N3 - sigma3 * x1 / N3)
    list(c(dx1, dx2, dx3))
  })
}

# Set the initial conditions, time and parameters
predict.time <- seq(0, 20, by = 1)
initial.condition <- c(x1 = 20, x2 = 20, x3 = 20)
parameters <- c(r1 = 1,
                lamda1 = 0.1,
                sigma1 = 0.5,
                N1 = 500,
                r2 = 0.5,
                lamda2 = 0.02,
                r3 = 0.2,
                sigma3 = 2,
                N3 = 300,
                e = 0.3)

# Solve for the ODEs
volterra.sc <- ode(y = initial.condition, times = predict.time, func =
model, parms = parameters)
volterra.sc.df <- as.data.frame(volterra.sc)

rm(list=ls()[!ls() %in% c("volterra.sc.df")])

# Visualize the results
prediction.plot <- ggplot() +
  geom_line(data = volterra.sc.df, aes(x = time, y = x1), color = "blue
", linewidth = 0.8, linetype = "solid") +
  geom_line(data = volterra.sc.df, aes(x = time, y = x2), color = "red",
 linewidth = 0.8, linetype = "solid") +
  geom_line(data = volterra.sc.df, aes(x = time, y = x3), color = "gree
n", linewidth = 0.8, linetype = "solid") +
  labs(title = "",
       x = "Time", y = "Population Size") +
  theme_classic() +
  scale_x_continuous(breaks = seq(0, 20, by = 1), labels = ifelse(seq(0,
 20, by = 1) %% 2 == 0, seq(0, 20, by = 1), "")) +
  scale_y_continuous(breaks = seq(0, 160, by = 10), labels = ifelse(seq
(0, 160, by = 10) %% 20 == 0, seq(0, 160, by = 10), "")) +
  theme(
    axis.title.x = element_text(size = 14, face = "bold"),
    axis.title.y = element_text(size = 14, face = "bold"),
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12)
  )
xs <- 1
ys <- 120
prediction.plot <- prediction.plot +
  annotate("text", x = xs+1, y = ys+16, label = "Species A", color = "b
lack", size = 4, hjust = 0) +
  annotate("segment", x = xs, xend = xs+0.9, y = ys+16, yend = ys+16, c
olor = "blue", linewidth = 1, linetype = "solid") +
  annotate("text", x = xs+1, y = ys+8, label = "Species B", color = "bl
ack", size = 4, hjust = 0) +
  annotate("segment", x = xs, xend = xs+0.9, y = ys+8, yend = ys+8, col
or = "red", linewidth = 1, linetype = "solid") +
  annotate("text", x = xs+1, y = ys, label = "Species C", color = "blac
k", size = 4, hjust = 0) +
  annotate("segment", x = xs, xend = xs+0.9, y = ys, yend = ys, color =
 "green", linewidth = 1, linetype = "solid")
```



Species A
Species B
Species C
Population Size
Time
0
20
40
60
80
100
120
140
160
0
2
4
6
8
10
12
14
16
18
20