MATH550: Coursework 2

36850162

2024-10-25

```r
# Parameters for the Lotka-Volterra model
alpha <- 0.05          # Birth rate of rabbits
beta <- 1.2e-4         # Rate of foxes eating rabbits
gamma <- 0.04          # Death rate of foxes
initial_rabbits <- 60  # Initial population of rabbits (R1)
initial_foxes <- 30    # Initial population of foxes (F1)
time_steps <- 104      # Total time steps (104 weeks for a 2-year period)

# Initialization function to set up population vectors
initialize_population <- function(initial_rabbits, initial_foxes, time_steps) {
  rabbit_population <- numeric(time_steps)  # Vector to store rabbit population over time
  fox_population <- numeric(time_steps)     # Vector to store fox population over time
  rabbit_population[1] <- initial_rabbits   # Set initial number of rabbits
  fox_population[1] <- initial_foxes        # Set initial number of foxes
  list(rabbit_population, fox_population)   # Return vectors as a list
}

# Initialize population vectors
populations <- initialize_population(initial_rabbits, initial_foxes, time_steps)
rabbit_population <- populations[[1]]
fox_population <- populations[[2]]

# Unified Lotka-Volterra Model function with input validation
lotka_volterra_model <- function(rabbit_population, fox_population, alpha, beta, gamma, time_steps, sto
  # Input validation
  if (!is.numeric(alpha) || alpha <= 0) stop("Error: alpha (birth rate) must be a positive number.")
  if (!is.numeric(beta) || beta <= 0) stop("Error: beta (consumption rate) must be a positive number.")
  if (!is.numeric(gamma) || gamma <= 0) stop("Error: gamma (death rate) must be a positive number.")
  if (!is.numeric(rabbit_population[1]) || rabbit_population[1] < 0) stop("Error: initial_rabbits must
  if (!is.numeric(fox_population[1]) || fox_population[1] < 0) stop("Error: initial_foxes must be a non
  if (!is.numeric(time_steps) || time_steps <= 0 || round(time_steps) != time_steps) stop("Error: time_

  # Set seed for stochastic model
  if (stochastic) set.seed(17540)

  # Loop through each time step, calculating population changes
  for (t in 2:time_steps) {
    # Calculate births, deaths, and predation based on model type
    if (stochastic) {
      rabbits_born <- rbinom(1, rabbit_population[t-1], alpha)
      rabbits_eaten <- rbinom(1, rabbit_population[t-1] * fox_population[t-1], beta)
      foxes_died <- rbinom(1, fox_population[t-1], gamma)
```

```
    } else {
      rabbits_born <- alpha * rabbit_population[t-1]
      rabbits_eaten <- beta * rabbit_population[t-1] * fox_population[t-1]
      foxes_died <- gamma * fox_population[t-1]
    }

    # Update populations and ensure non-negative values
    rabbit_population[t] <- max(0, rabbit_population[t-1] + rabbits_born - rabbits_eaten)
    fox_population[t] <- max(0, fox_population[t-1] + rabbits_eaten - foxes_died)
  }

  # Return the final populations as a list
  list(rabbits = rabbit_population, foxes = fox_population)
}

# Task 1: Run the deterministic model
det_results <- lotka_volterra_model(rabbit_population, fox_population, alpha, beta, gamma, time_steps, s
det_rabbits <- det_results[[1]]
det_foxes <- det_results[[2]]

# Display the last few values of the deterministic model results
print("Deterministic Model - Last few values:")
```

```
## [1] "Deterministic Model - Last few values:"
```

```
print(tail(det_rabbits, n = 5))
```

```
## [1] 107.94989  84.03695  66.03831  52.41642  42.03523
```

```
print(tail(det_foxes, n = 5))
```

```
## [1] 2262.657 2201.461 2135.603 2067.103 1997.421
```

```
# Task 2: Run the stochastic model
sto_results <- lotka_volterra_model(rabbit_population, fox_population, alpha, beta, gamma, time_steps, s
sto_rabbits <- sto_results[[1]]
sto_foxes <- sto_results[[2]]

# Display the last few values of the stochastic model results
print("Stochastic Model - Last few values:")
```

```
## [1] "Stochastic Model - Last few values:"
```

```
print(tail(sto_rabbits, n = 5))
```

```
## [1] 481 360 260 179 128
```

```
print(tail(sto_foxes, n = 5))
```

```
## [1] 2964 3005 3001 2980 2918
```

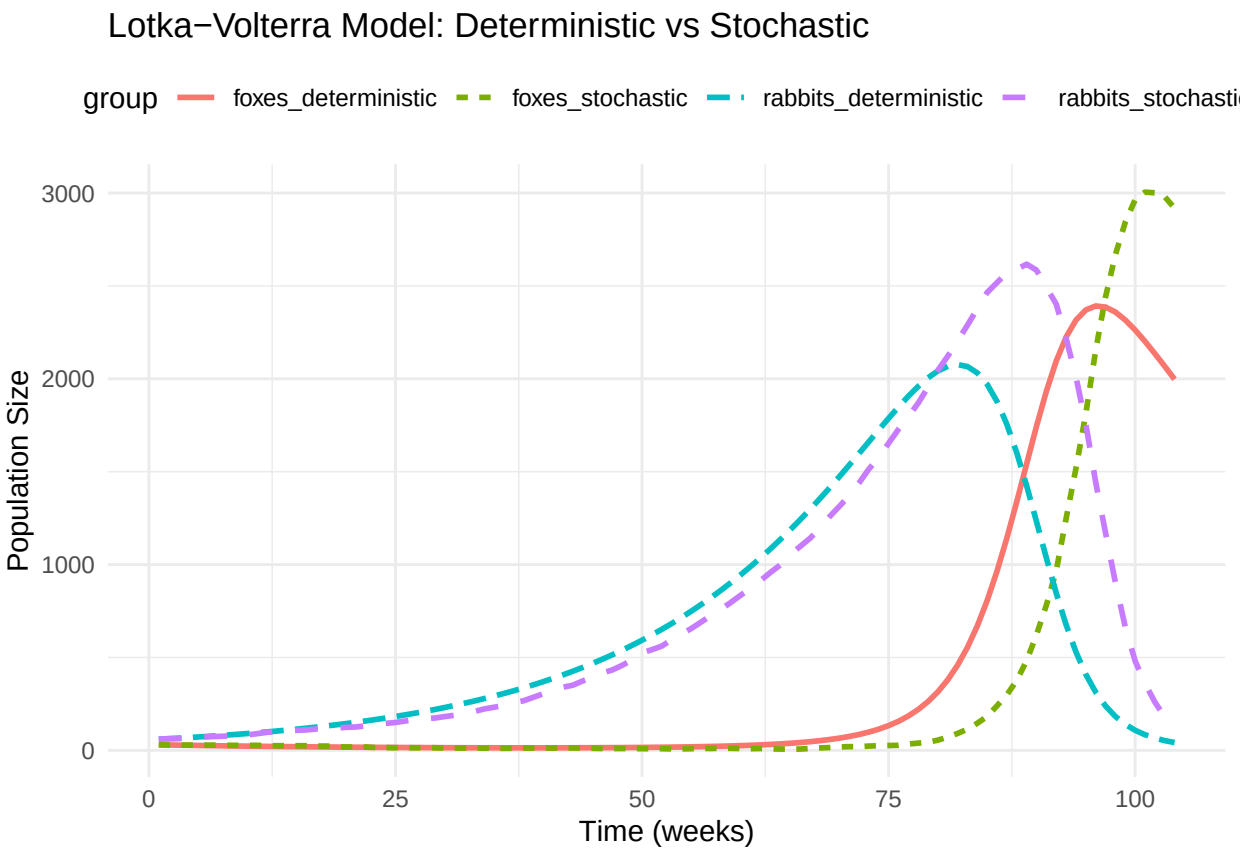
```r
# Task 3: Visualization of Results

# Create the data frame for visualization
create_LV_dataframe <- function(rabbit_det, fox_det, rabbit_sto, fox_sto, time_steps) {
  data.frame(
    time = rep(1:time_steps, 4),
    group = rep(c("rabbits_deterministic", "foxes_deterministic", "rabbits_stochastic", "foxes_stochasti
    size = c(rabbit_det, fox_det, rabbit_sto, fox_sto)
  )
}

# Prepare data for visualization
LV <- create_LV_dataframe(det_rabbits, det_foxes, sto_rabbits, sto_foxes, time_steps)

# Task 3: Plotting the Results
library(ggplot2)

ggplot(LV, aes(x = time, y = size, color = group, linetype = group)) +
  geom_line(linewidth = 1) +  # Adjust line thickness using linewidth
  labs(
    title = "Lotka-Volterra Model: Deterministic vs Stochastic",
    x = "Time (weeks)",
    y = "Population Size"
  ) +
  theme_minimal() +  # Use a minimal theme for a cleaner look
  theme(
    legend.position = "top"  # Place legend at the top for better visibility
  )
```


Lotka−Volterra Model: Deterministic vs Stochastic
group
foxes_deterministic
foxes_stochastic
rabbits_deterministic
rabbits_stochasti
Population Size
3000
2000
1000
0
0
25
50
75
100
Time (weeks)