Групповой проект хищник-жертва

Программная реализация проекта

Беличева Д. М., Демидова Е. А.,
Самигуллин Э. А., Смирнов-Мальцев Е. Д.

Содержание

Список иллюстраций

1 Цель работы

- Программная реализация проекта хищник-жертва.

2 Задачи

- Описать функции для решения ОДУ в Octave
- Построить график зависимости числа хищниов от числа жертв
- Построить графики зависимости числа видов от времени
- Найти стационарное состояние системы

3 Теоретическое введение

В Octave нет метода Эйлера, однако есть методы Рунге-Кутты[1].

`ode23(@f, interval, X0, options)`, `ode45(@f, interval, X0, options)` — функции решений обыкновенных нежёстких дифференциальных уравнений (или систем) методом Рунге-Кутты 2-3-го и 4-5-го порядка точности соответственно.

Функции решают систему дифференциальных уравнений, автоматически подбирая шаг для достижения необходимой точности. Входными параметрами этих функций являются:

- `f` – вектор-функция для вычисления правой части дифференциального уравнения или системы;
- `interval` – массив из двух чисел, определяющий интервал интегрирования дифференциального уравнения или системы;
- `X0` – вектор начальных условий системы дифференциальных систем;
- `option` – параметры управления ходом решения дифференциального уравнения или системы.

При решении дифференциальных уравнений необходимо определить следующие параметры:

- `RelTol` – относительная точность решения, значение по умолчанию 10–3;
- `AbsTol` – абсолютная точность решения, значение по умолчанию 10–3;
- `InitialStep` – начальное значение шага изменения независимой переменной, значение по умолчанию 0.025;

- `MaxStep` – максимальное значение шага изменения независимой переменной, значение по умолчанию 0.025.

Все функции возвращают:

- массив T - координат узлов сетки, в которых ищется решение;
- матрицу X, i-й столбец которой является значением вектор-функции решения в узле Ti.

4 Рунге-Кутта второго и третьего порядка

1. Реализация алгоритма Рунге-Кутта второго и третьего порядка с параметрами по умолчанию(рис. 4.1).

```
function dx=f(t, x)
  a = 0.2; % коэффициент естественной смертности хищников
  b = 0.05; % коэффициент естественного прироста жертв
  c = 0.5; % коэффициент увеличения числа хищников
  d = 0.02; % коэффициент смертности жертв
  dx(1) = -a*x(1) + b*x(1)*x(2);
  dx(2) = c*x(2) - d*x(1)*x(2);
endfunction

A(:,1) =  21:0.5:25;
A(:,2) = 1:0.5:5;

for i = 1:size(A(:,1))
    [T M] = ode23 (@f, [0 50],  A(i,:));
    X = M(:, 1);
    Y = M(:, 2);
    plot(X, Y);
    hold on;
```

```
end

[T M] = ode23 (@f, [0 30],  [25 4]);
X = M(:, 1);
Y = M(:, 2);
plot(X, Y, '+',"linewidth", 3);
```

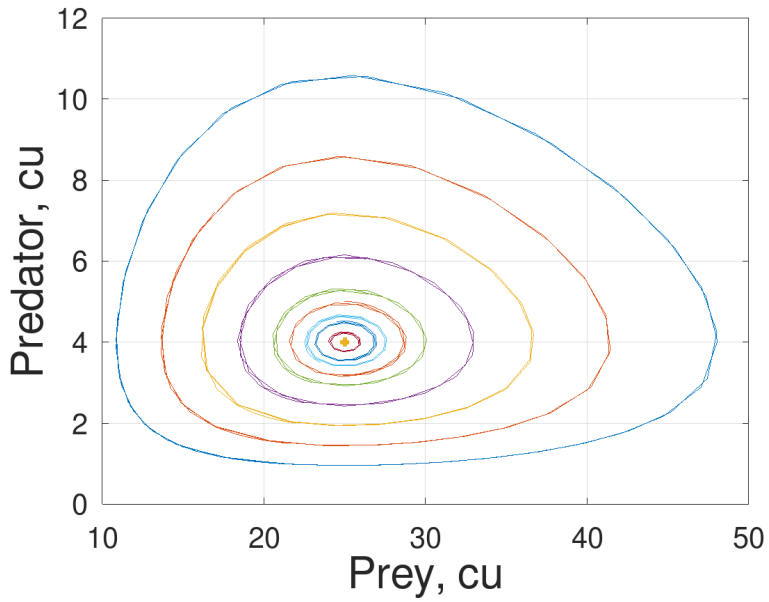

Рис. 4.1: Фазовый портрет Рунге-Кутта второго и третьего порядка

2. Реализация алгоритма Рунге-Кутта второго и третьего порядка с максимальным шагом в 0.1 (рис. 4.2).

```
clear; clf;
function dx=f(t, x)
  a = 0.2; % коэффициент естественной смертности хищников
  b = 0.05; % коэффициент естественного прироста жертв
  c = 0.5; % коэффициент увеличения числа хищников
  d = 0.02; % коэффициент смертности жертв
  dx(1) = -a*x(1) + b*x(1)*x(2);
```

```
  dx(2) = c*x(2) - d*x(1)*x(2);
endfunction

A(:,1) =  21:0.5:25;
A(:,2) = 1:0.5:5;
opt = odeset ("MaxStep", 0.1);
for i = 1:size(A(:,1))
[T M] = ode23 (@f, [0 50],  A(i,:),opt);
X = M(:, 1);
Y = M(:, 2);
plot(X, Y);
hold on;
end

[T M] = ode23 (@f, [0 50],  [25 4]);
X = M(:, 1);
Y = M(:, 2);
plot(X, Y, '+',"linewidth", 3);
grid on;
set(gca, "FontSize",20)

xlabel('Prey, cu', "fontsize",30);
ylabel('Predator, cu', "fontsize",30)
```

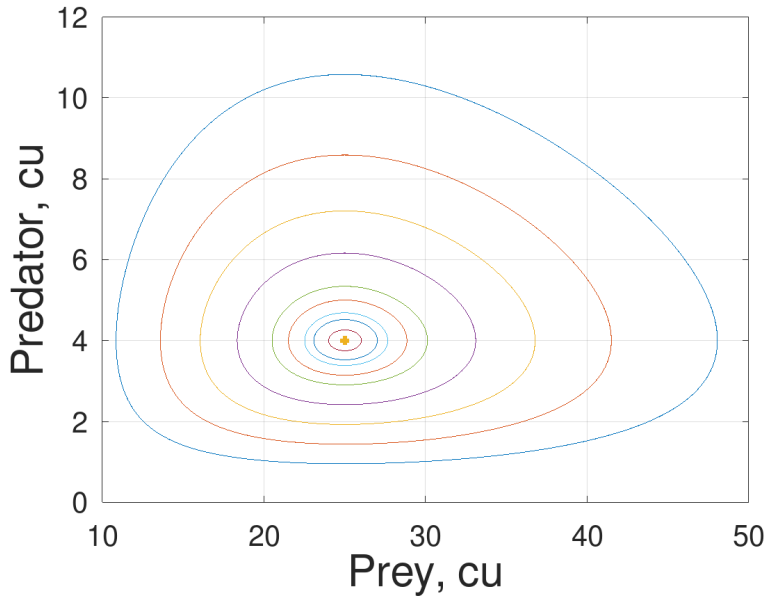

Рис. 4.2: Фазовый портрет Рунге-Кутта второго и третьего порядка с шагом в 0.1

3. Построение зависимости видов от времени модели хищник-жертва с использованием Рунге-Кутта второго и третьего порядка (рис. 4.3).

```
function dx=f(t, x)
  a = 0.2; % коэффициент естественной смертности хищников
  b = 0.05; % коэффициент естественного прироста жертв
  c = 0.5; % коэффициент увеличения числа хищников
  d = 0.02; % коэффициент смертности жертв
  dx(1) = -a*x(1) + b*x(1)*x(2);
  dx(2) = c*x(2) - d*x(1)*x(2);
endfunction


A(:,1) =  21:0.5:25;
A(:,2) = 1:0.5:5;


[T M] = ode23 (@f, [0 100],  [21 1]);
```

```
X = M(:, 1);
Y = M(:, 2);
plot(T,X,T, Y);
hold on;

grid on;
set(gca, "FontSize",20)
legend("Prey", "Predator")
xlabel('Time, cu', "fontsize",30);
ylabel('Animals, cu', "fontsize",30)
```

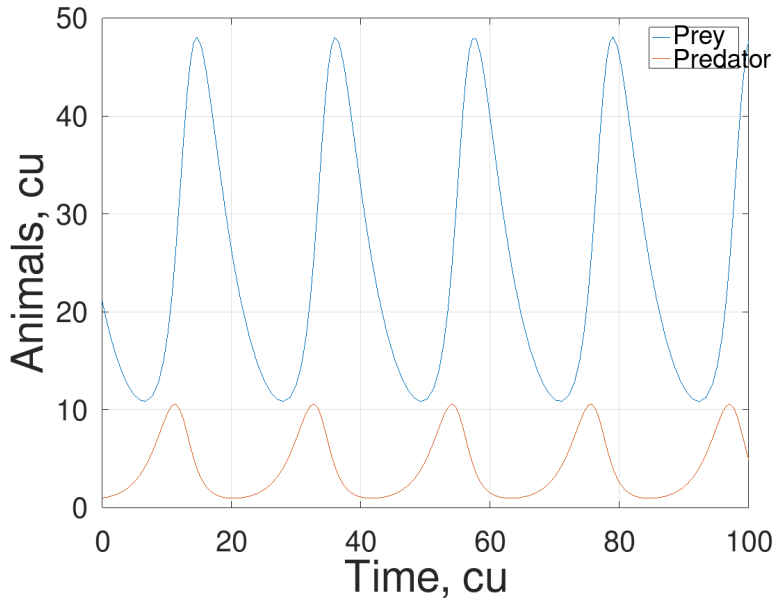

Рис. 4.3: Зависимость хищника-жертвы от времени

5 Рунге-Кутта четвертого и пятого порядка

1. Реализация алгоритма Рунге-Кутта четвертого и пятого порядка с параметрами по умолчанию(рис. 5.1).

```
function dx=f(t, x)
  a = 0.2; % коэффициент естественной смертности хищников
  b = 0.05; % коэффициент естественного прироста жертв
  c = 0.5; % коэффициент увеличения числа хищников
  d = 0.02; % коэффициент смертности жертв
  dx(1) = -a*x(1) + b*x(1)*x(2);
  dx(2) = c*x(2) - d*x(1)*x(2);
endfunction


A(:,1) =  21:0.5:25;
A(:,2) = 1:0.5:5;
for i = 1:size(A(:,1))
    [T M] = ode45 (@f, [0 50],  A(i,:));
    X = M(:, 1);
    Y = M(:, 2);
    plot(X, Y);
    hold on;
end
```

```
[T M] = ode45 (@f, [0 50],  [25 4]);
X = M(:, 1);
Y = M(:, 2);
plot(X, Y, '+',"linewidth", 3);
grid on;
set(gca, "FontSize",20)

xlabel('Prey, cu', "fontsize",30);
ylabel('Predator, cu', "fontsize",30)
```

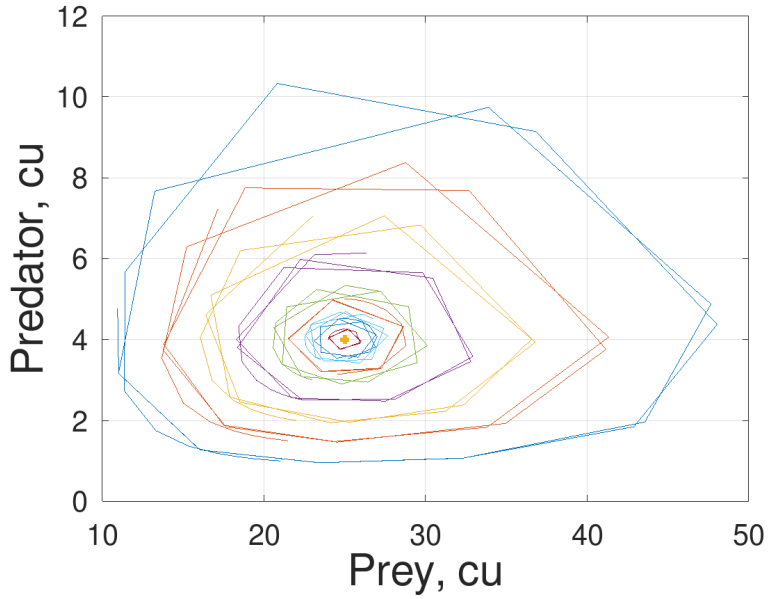

Рис. 5.1: Фазовый портрет Рунге-Кутта четвертого и пятого порядка

2. Реализация алгоритма Рунге-Кутта четвертого и пятого порядка с максимальным шагом 0.1 (рис. 5.2).

```
function dx=f(t, x)
  a = 0.2; % коэффициент естественной смертности хищников
  b = 0.05; % коэффициент естественного прироста жертв
```

```
  c = 0.5; % коэффициент увеличения числа хищников
  d = 0.02; % коэффициент смертности жертв
  dx(1) = -a*x(1) + b*x(1)*x(2);
  dx(2) = c*x(2) - d*x(1)*x(2);
endfunction

A(:,1) =  21:0.5:25;
A(:,2) = 1:0.5:5;
opt = odeset ("MaxStep", 0.1);
for i = 1:size(A(:,1))
    [T M] = ode45 (@f, [0 50],  A(i,:),opt);
    X = M(:, 1);
    Y = M(:, 2);
    plot(X, Y);
    hold on;
end

[T M] = ode45 (@f, [0 50],  [25 4]);
X = M(:, 1);
Y = M(:, 2);
plot(X, Y, '+',"linewidth", 3);
grid on;
set(gca, "FontSize",20)

xlabel('Prey, cu', "fontsize",30);
ylabel('Predator, cu', "fontsize",30)
```

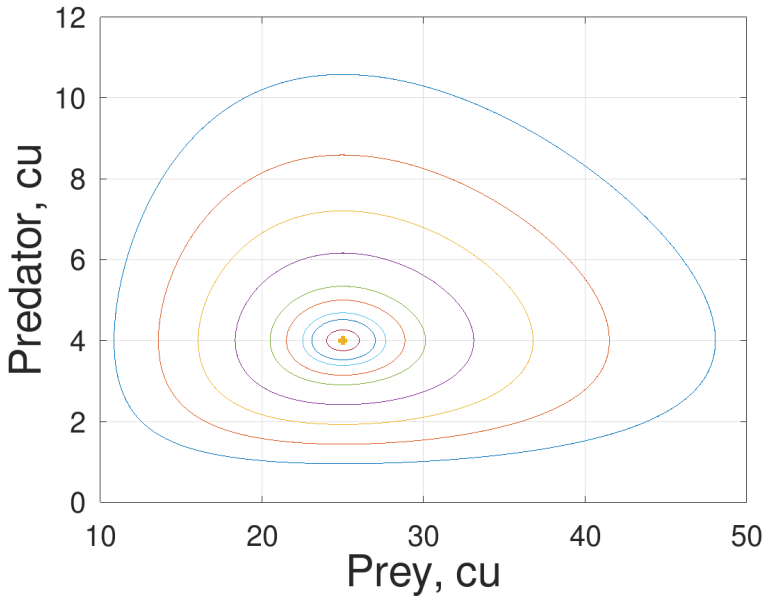

Рис. 5.2: Фазовый портрет Рунге-Кутта четвертого и пятого порядка с шагом в 0.1

3. Построение зависимости видов от времени модели хищник-жертва с использованием Рунге-Кутта четвертого и пятого порядка (рис. 5.3).

```
function dx=f(t, x)
  a = 0.2; % коэффициент естественной смертности хищников
  b = 0.05; % коэффициент естественного прироста жертв
  c = 0.5; % коэффициент увеличения числа хищников
  d = 0.02; % коэффициент смертности жертв
  dx(1) = -a*x(1) + b*x(1)*x(2);
  dx(2) = c*x(2) - d*x(1)*x(2);
endfunction


A(:,1) =  21:0.5:25;
A(:,2) = 1:0.5:5;
par = odeset("MaxStep", 0.1)
```

```
    [T M] = ode45 (@f, [0 100],  [21 1], par);
    X = M(:, 1);
    Y = M(:, 2);
    plot(T,X,T, Y);
    hold on;

grid on;
set(gca, "FontSize",20)
legend("Prey", "Predator")
xlabel('Time, cu', "fontsize",30);
ylabel('Animals, cu', "fontsize",30)
```

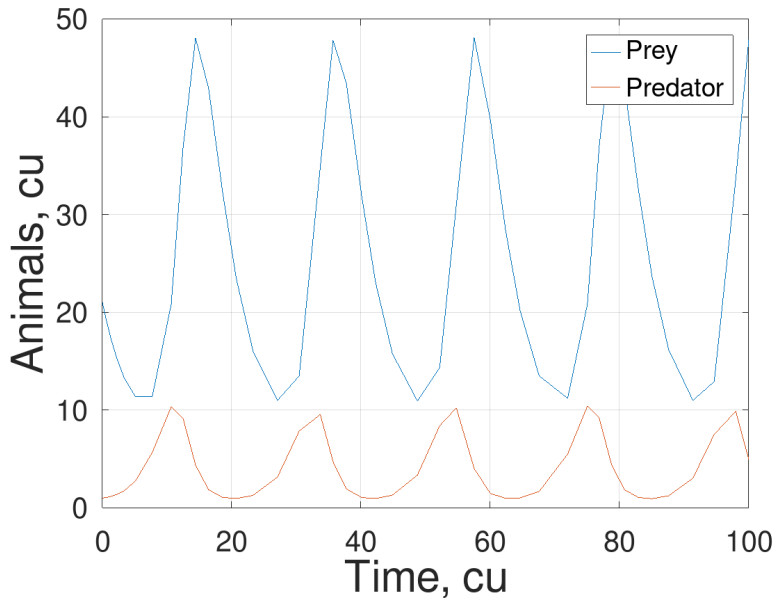

Рис. 5.3: Зависимость хищника-жертвы от времени

6 Вывод

В результате работы была выполнена программная реализация проекта, а именно были построены графики зависимости видов друг от друга, от времени и найдено стационарное состояние системы с помощью методов Рунге_Кутты 2-3-го и 4-5-го порядка точности.

Список литературы

1. GNU Octave Documentation [Электронный ресурс]. Free Software Foundation, 2023. URL: https://docs.octave.org/v4.2.0/Matlab_002dcompatible-solvers.html.