

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ

Кафедра прикладной математики и искусственного интеллекта
Направление подготовки - «Прикладная математика»

КУРСОВАЯ РАБОТА

по дисциплине: «Численные методы»

Система типа хищник-жертва. Модель Вольтерра.

Студент 2 курса

группы 09-221

«___» _____ 2024 г.

М.А. Саитов

Научный руководитель

ассистент б.с.

«___» _____ 2024 г.

О.В. Глазырина

Казань, 2024 год

Содержание

1	Цель работы	3
2	Теоретические основы выполнения работы	3
3	Метод Рунге-Кутты	5
4	Задание	6
5	Проверка правильности вывода исходных уравнений	7
6	Решение тестовой задачи	8
7	Модель Вольтерра	10
8	Заключение	13
9	Список литературы	14
10	Листинг программы	15

1 Цель работы

Целью работы является исследование модели взаимодействия «Хищник — Жертва», выявление зависимостей поведения модели в зависимости от значения параметров, описывающих систему, а так же применение метода Рунге-Кутты при решении сопутствующих систем дифференциальных уравнений.

2 Теоретические основы выполнения работы

Модель Вольтерра используется для представления взаимодействия двух типов. При моделировании системы «Хищник — Жертва» будем учитывать следующие ограничения:

1. Жертва может найти достаточно пищи для пропитания
2. При каждой встрече с хищником последний убивает жертву
3. Норма рождаемости жертв x_b , нормы естественной смертности жертв x_d и хищников c являются постоянными, $a = x_b - x_d > 0$.
4. Число случаев, когда хищник убивает жертву, зависит от вероятности их встречи и, следовательно, пропорционально произведению xy .

В результате встреч с жертвами число хищников увеличивается.

Обозначим количества хищников и жертв в момент времени t через $y = y(t)$ и $x = x(t)$ соответственно. Тогда с учетом сделанных допущений, модель можно записать в виде:

$$\begin{aligned}\frac{dx}{dt} &= ax - bxy \\ \frac{dy}{dt} &= -cy + dxy\end{aligned}\tag{1}$$

где коэффициенты b, d - коэффициенты убийства жертв и воспроизводства хищников, a - описывает рождаемость жертв, c - естественную смерть жертв.

Произведя в системе (1) замены:

$$X = \left(\frac{d}{a}\right)x, \quad Y = \left(\frac{b}{a}\right)y, \quad \tau = at, \quad \sigma = \frac{c}{a}$$

уравнение (1) преобразуется к виду:

$$\begin{aligned}\frac{dX}{d\tau} &= X - XY \\ \frac{dY}{d\tau} &= -\sigma Y + XY\end{aligned}\tag{2}$$

с начальными условиями:

$$X(0) = X_0, \quad Y(0) = Y_0\tag{3}$$

описывающими количество особей каждого из вида.

3 Метод Рунге-Кутты

Для решения систем дифференциальных уравнений будем использовать метод Рунге – Кутты 4-го порядка точности:

$$\begin{aligned}k_1 &= f(t_n, y_n), \\k_2 &= f\left(t_n + \frac{h}{3}, y_n + \frac{hk_1}{3}\right), \\k_3 &= f\left(t_n + \frac{2h}{3}, y_n - \frac{hk_1}{3} + hk_2\right), \\k_4 &= f(t_n + h, y_n + h(k_1 - k_2 + k_3)), \\y_{n+1} &= y_n + \frac{h(k_1 + 3k_2 + 3k_3 + k_4)}{8}\end{aligned}\tag{4}$$

где $y'(t, y) = k(t, y)$. Задано начальное условие $y(0) = y_0$. Этот метод является методом с постоянным шагом h на отрезке $[a, b]$, значения решения вычисляются в $n = \frac{b-a}{h}$ точках.

Используя приведенный выше метод решить тестовую задачу:

$$y_1' = \frac{y_1}{2 + 2t} - 2ty_2, \quad y_2' = \frac{y_2}{2 + 2t} + 2ty_1,\tag{5}$$

на отрезке $[0, 2]$.

4 Задание

В ходе выполнения работы необходимо:

1. Проверить правильность вывода исходных уравнений (1), уравнений в безразмерном виде (2) и тестового решения (5)
2. Найти стационарные решения (состояния равновесия) системы (2).
3. Написать процедуру интегрирования задачи Коши для системы из n обыкновенных дифференциальных уравнений по формулам (4) на произвольном отрезке $[a, b]$ с постоянным шагом h .
4. Для тестовой задачи (5) построить графики зависимости максимальной погрешности решения e и $\frac{e}{h^4}$ от выбранного шага h .
Пояснить результаты расчетов.
5. Для двух наборов начальных условий (3) в окрестности состояния равновесия и нескольких значений параметра σ рассчитать динамику популяции. Привести графики наиболее характерных решений в координатах (X, Z) и дать их интерпретацию.

5 Проверка правильности вывода исходных уравнений

Для перехода от уравнения системы (1) к уравнению системы (2) мы введем новые переменные и новые параметры, а затем проведем соответствующие замены и преобразования.

Введем следующие замены:

$$X = \left(\frac{d}{a}\right) x, \quad Y = \left(\frac{b}{a}\right) y, \quad \tau = at, \quad \sigma = \frac{c}{a}$$

Пойдем методом от обратного и выведем уравнения системы (1) через уравнения системы (2):

1. Подставим в левую часть первого уравнения (2) наши замены получим:

$$\frac{dX}{d\tau} = \frac{d\left(\frac{d}{a}\right)x}{d(at)} = \frac{d}{a^2} \frac{dx}{dt}$$

2. Подставим в правую часть наши замены получим:

$$X - XY = \left(\frac{d}{a}\right)x - \left(1 - \left(\frac{b}{a}\right)y\right)$$

3. Соединив два уравнения получим:

$$\frac{d}{a^2} \frac{dx}{dt} = \left(\frac{d}{a}\right)x - \left(1 - \left(\frac{b}{a}\right)y\right)$$

4. Разделив обе части уравнения на $\frac{d}{a^2}$ получим:

$$\frac{dx}{dt} = ax - bxy$$

Проделав те же самые шаги ко второму уравнению системы (2), мы придем ко второму уравнению системы (1).

6 Решение тестовой задачи

Тестовый вариант задачи будет решен методом Рунге - Кутты 4-го порядка точности. Дана система уравнений (5) с известным точным решением:

$$y_1 = \cos(t^2)\sqrt{1+t}, \quad y_2 = \sin(t^2)\sqrt{1+t} \quad (6)$$

Необходимо решить систему с начальными условиями $y_1(0) = 0, y_2(0) = 1$ на отрезке $[0; 2]$. Перед вычислением коэффициентов стоит заметить, что $y'_1 = y'_1(y_1, y_2, t)$ и $y'_2 = y'_2(y_1, y_2, t)$, тогда:

$$\begin{aligned} k_{11} &= y'_1(t(i), y_1(i), y_2(i)) \\ k_{12} &= y'_2(t(i), y_1(i), y_2(i)) \\ k_{21} &= y'_1(t(i) + \frac{h}{3}, y_1(i) + \frac{h}{3}k_{11}, y_2(i) + \frac{h}{3}k_{12}) \\ k_{22} &= y'_2(t(i) + \frac{h}{3}, y_1(i) + \frac{h}{3}k_{11}, y_2(i) + \frac{h}{3}k_{12}) \\ k_{31} &= y'_1(t(i) + \frac{2h}{3}, y_1(i) - \frac{h}{3}k_{11} + hk_{21}, y_2(i) - \frac{h}{3}k_{12} + hk_{22}) \\ k_{32} &= y'_2(t(i) + \frac{2h}{3}, y_1(i) - \frac{h}{3}k_{11} + hk_{21}, y_2(i) - \frac{h}{3}k_{12} + hk_{22}) \\ k_{41} &= y'_1(t[i] + h, y_1(i) + h(k_{11} - k_{21} + k_{31}), y_2(i) + h(k_{12} - k_{22} + k_{32})) \\ k_{42} &= y'_2(t[i] + h, y_1(i) + h(k_{11} - k_{21} + k_{31}), y_2(i) + h(k_{12} - k_{22} + k_{32})) \\ y_1(i+1) &= y_1(i) + \frac{h}{8}(k_{11} + 3k_{21} + 3k_{31} + k_{41}) \\ y_2(i+1) &= y_2(i) + \frac{h}{8}(k_{12} + 3k_{22} + 3k_{32} + k_{42}) \end{aligned} \quad (7)$$

где $i = 1, \dots, \frac{b-a}{h} - 1$

Получим график данных функций:

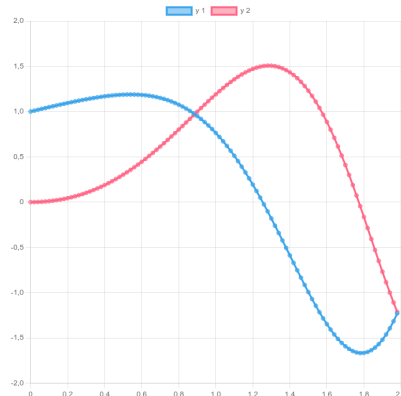


Рис.1 — График двух функций, полученных через метод Рунге - Кутты.

Решив таким образом систему, составим графики зависимости максимальной ошибки решения от шага h^4 :

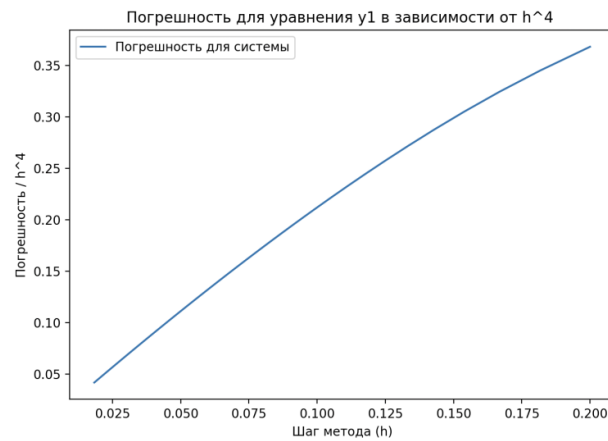


Рис.2 — Зависимость точности решения от шага h^4 .

Исходя из вида графиков, описывающих зависимость точности решения, можно сделать вывод о зависимости точности решения от шага разбиения.

7 Модель Вольтерра

Теперь мы можем, используя правило (4) решить систему (2). Выберем начальные условия $X_0 = 3, Y_0 = 0.5$, начальное число жертв и хищников, соответственно. Исходя из вида системы, отметим, что динамика популяции зависит от параметра $\sigma = \frac{c}{a}$ - отношения смертности хищников к приросту жертв. Выбрав $\sigma = 1$, т.е. установив равное отношение прироста жертв к смертности хищников получим график описывающий размеры популяций жертв и хищников с течением времени (Рис.3). Размер популяции жертв в начальный момент превышает размер популяции хищников, из-за чего вероятность встречи представителей двух видов возрастает, хищник убивает жертву. С уменьшением количества жертв, количество встреч с хищниками убывает. Когда хищники не охотятся, их популяция начинает вымирать. Уменьшение популяции хищников уменьшает вероятность встреч с жертвами, что для последних, при наличии бесконечного числа пищи является причиной резкого увеличения числа представителей вида. Резкое увеличение числа жертв предполагает возрастание числа хищников. Система приходит в циклическое, устойчивое состояние. На графике по горизонтали – время, по вертикали – популяция.

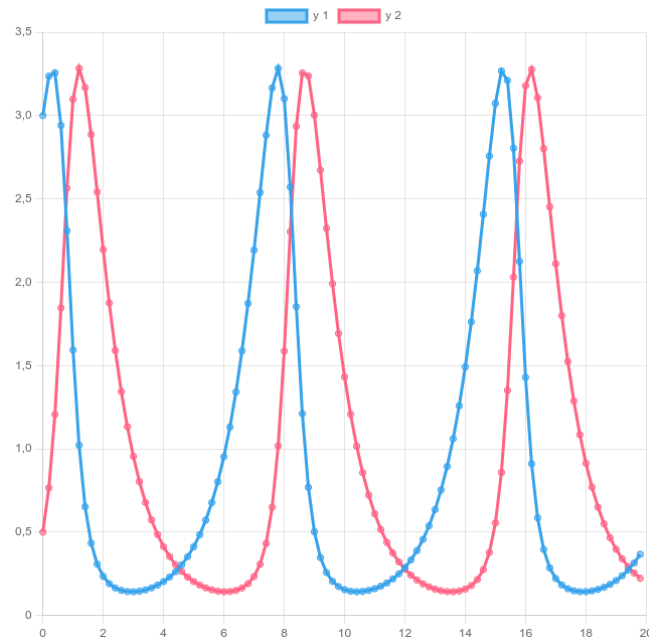


Рис.3 — Размер популяций жертв (синий) и хищников (красный) с течением времени.

Теперь выберем $\sigma = 3$, что определит трехкратное увеличение убыли хищников к росту популяций жертв. Поведение популяций видов изменилось, численность популяций изменяется не так сильно, однако, за тот же промежуток времени происходит большее число "колебаний" числа представителей вида (Рис.4).

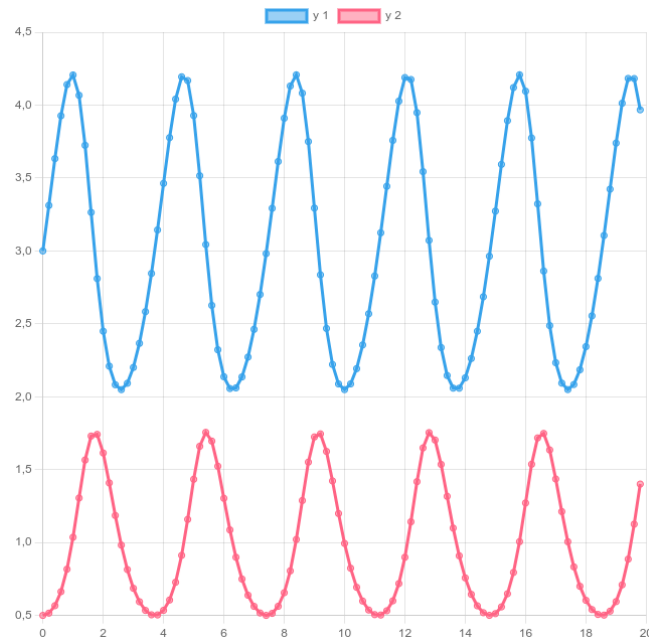


Рис.4 — Изменение популяций видов при $\sigma = 3$.

Применим те же параметры для системы с начальными условиями $X_0 = 1, Y_0 = 4$, что соответствует ситуации, когда в начальный момент число хищников в 4 раза превышает количество жертв. Тенденция стабилизации системы сохраняется, однако при значении $\sigma = 3$ популяция хищников убывает сильнее из-за малого количества доступной изначально пищи и значения коэффициента.

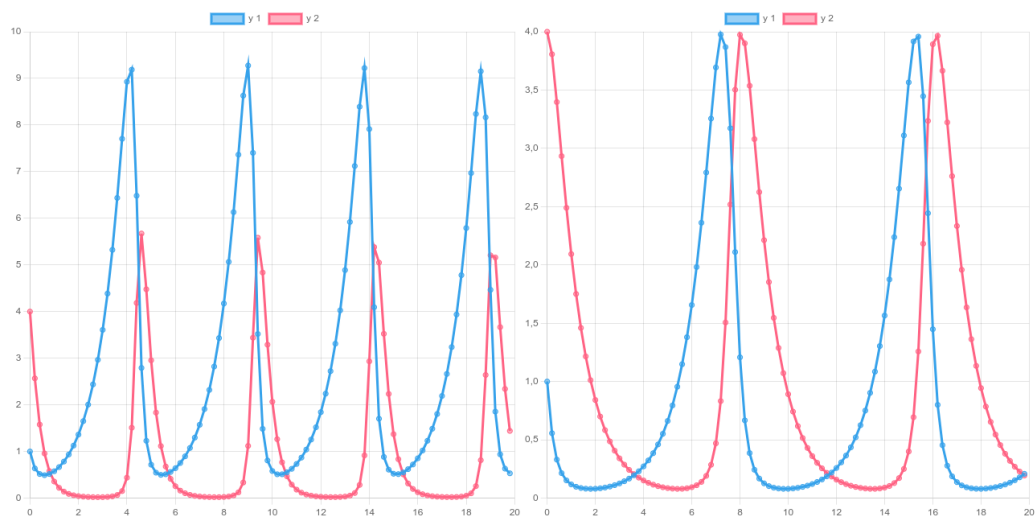


Рис.5 — Вид популяции при других начальных условиях.

Представим графики в координатах (X, Y) для двух рассмотренных начальных условий (Рис. 6).

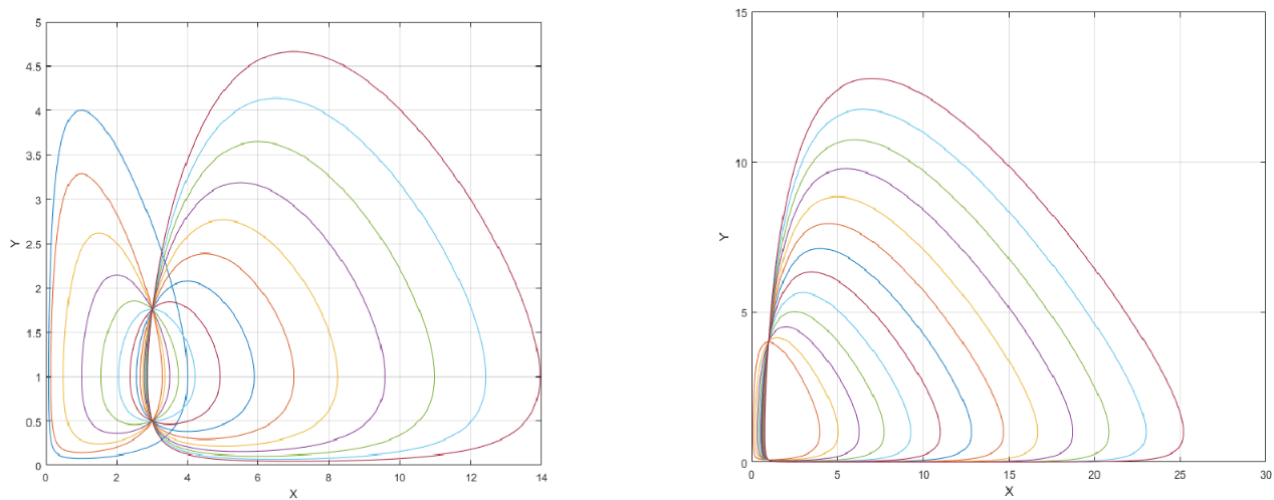


Рис.6 — Графики (X, Y) в зависимости от параметра σ .

8 Заключение

Модель Вольтерра может быть использована для моделирования системы типа «Хищник–жертва». Наблюдая за моделью можно выявить основные закономерности поведения популяций. В ходе работы были рассмотрены модели с различными значениями начального размера популяций, коэффициентов определяющих отношение убыли одного вида к приросту другого. При выполнении задачи был использован метод Рунге–Кутты для решения систем дифференциальных уравнений. Метод позволяет решать уравнения с достаточной степенью точности, достаточной для корректного моделирования системы.

9 Список литературы

1. Глазырина Л.Л., Карчевский М.М. Численные методы: учебное пособие. — Казань: Казан. ун-т, 2012. — 122
2. Глазырина Л.Л.. Практикум по курсу «Численные методы». Решение систем линейных уравнений: учеб. пособие. — Казань: Изд-во Казан. ун-та, 2017. — 52 с.
3. Дж. Ортега, У. Пул. Введение в численные методы решения дифференци- альных уравнений. — М.: Наука, 1986
4. Самарский А.А., Гулин А.В. Численные методы. — М.: Наука, 1989.
5. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы. — М.: БИНОМ. Лаб. знаний, 2007.
6. Глазырина Л. Л., Карчевский М. М. Введение в численные методы. — Казань, КГУ, 2012

10 Листинг программы

```
1 #include <iostream>
2 #include <vector>
3 #include <cmath>
4 #include <fstream>
5 #include <iomanip>
6 #include <algorithm>
7
8 #define a 0.0
9 #define b 20.0
10 #define sigma 1.0
11
12 double y1_test(double t, double y1, double y2) {
13     return (y1 / (2 + 2 * t)) - 2 * t * y2;
14 }
15
16 double y2_test(double t, double y1, double y2) {
17     return (y2 / (2 + 2 * t)) + 2 * t * y1;
18 }
19
20 double y1_dynamic(double y1, double y2) {
21     return y1 - y1*y2;
22 }
23
24 double y2_dynamic(double y1, double y2) {
25     return -sigma*y2 + y1*y2;
26 }
27 void solve_test(double h, double x0, double y0, std::vector<double>& t,
28     std::vector<double>& y1, std::vector<double>& y2) {
29     int n = static_cast<int>((b - a) / h + 1);
30     t = std::vector<double>(n);
31     y1 = std::vector<double>(n);
32     y2 = std::vector<double>(n);
33     t[0] = double(a);
34     y1[0] = x0;
```

```

34     y2[0] = y0;
35
36     for (int i = 0; i < n - 1; i++) {
37         t[i + 1] = t[i] + h;
38         double k11 = y1_test(t[i], y1[i], y2[i]);
39         double k12 = y2_test(t[i], y1[i], y2[i]);
40
41         double k21 = y1_test(t[i] + h / 3.0, y1[i] + h * k11 / 3, y2[i]
42             + h * k12 / 3.0);
43         double k22 = y2_test(t[i] + h / 3.0, y1[i] + h * k11 / 3, y2[i]
44             + h * k12 / 3.0);
45
46         double k31 = y1_test(t[i] + 2.0 * h / 3.0, y1[i] - h * k11 /
47             3.0 + h * k21, y2[i] - h * k12 / 3.0 + h * k22);
48         double k32 = y2_test(t[i] + 2.0 * h / 3.0, y1[i] - h * k11 /
49             3.0 + h * k21, y2[i] - h * k12 / 3.0 + h * k22);
50
51         double k41 = y1_test(t[i] + h, y1[i] + h * (k11 - k21 + k31),
52             y2[i] + h * (k12 - k22 + k32));
53         double k42 = y2_test(t[i] + h, y1[i] + h * (k11 - k21 + k31),
54             y2[i] + h * (k12 - k22 + k32));
55
56         y1[i + 1] = y1[i] + h * (k11 + 3.0 * k21 + 3.0 * k31 + k41) /
57             8.0;
58         y2[i + 1] = y2[i] + h * (k12 + 3.0 * k22 + 3.0 * k32 + k42) /
59             8.0;
60     }
61 }
62
63 void solve_dynamic(double h, double x0, double y0, std::vector<double>&
64     t, std::vector<double>& y1, std::vector<double>& y2) {
65     int n = static_cast<int>((b - a) / h + 1);
66     t = std::vector<double>(n);
67     y1 = std::vector<double>(n);
68     y2 = std::vector<double>(n);
69     t[0] = double(a);

```



```

61     y1[0] = x0;
62     y2[0] = y0;
63
64     for (int i = 0; i < n - 1; i++) {
65         t[i + 1] = t[i] + h;
66         double k11 = y1_dynamic(y1[i], y2[i]);
67         double k12 = y2_dynamic(y1[i], y2[i]);
68
69         double k21 = y1_dynamic(y1[i] + h * k11 / 3, y2[i] + h * k12 /
70                                 3);
71         double k22 = y2_dynamic(y1[i] + h * k11 / 3, y2[i] + h * k12 /
72                                 3);
73
74         double k31 = y1_dynamic(y1[i] - h * k11 / 3 + h * k21, y2[i] -
75                                 h * k12 / 3 + h * k22);
76         double k32 = y2_dynamic(y1[i] - h * k11 / 3 + h * k21, y2[i] -
77                                 h * k12 / 3 + h * k22);
78
79         double k41 = y1_dynamic(y1[i] + h * (k11 - k21 + k31), y2[i] +
80                                 h * (k12 - k22 + k32));
81         double k42 = y2_dynamic(y1[i] + h * (k11 - k21 + k31), y2[i] +
82                                 h * (k12 - k22 + k32));
83
84         y1[i + 1] = y1[i] + h * (k11 + 3.0 * k21 + 3.0 * k31 + k41) /
85                                 8.0;
86         y2[i + 1] = y2[i] + h * (k12 + 3.0 * k22 + 3.0 * k32 + k42) /
87                                 8.0;
88     }
89 }
90
91 void exact(double h, std::vector<double>& y1, std::vector<double>& y2)
92 {
93     int n = static_cast<int>((b - a) / h + 1);
94     y1.resize(n);
95     y2.resize(n);
96     for (int i = 0; i < n; ++i) {
97         double t = a + i * h;

```

```

88     y1[i] = std::cos(t * t) * std::sqrt(1 + t);
89     y2[i] = std::sin(t * t) * std::sqrt(1 + t);
90 }
91 }
92
93 int main(){
94     double x0 = 1;
95     double y0 = 0;
96     int n = 36;
97     std::vector<double> N(n);
98     std::vector<double> e(n);
99     std::vector<double> h(n);
100    std::vector<double> e4(n);
101    for(int i = 0; i < 25; i++){
102        N[i] = (i+1);
103    }
104    for(int i = 25; i < n; i++){
105        N[i] = (N[i-1] + 25);
106    }
107    for(int i = 0; i < n; i++){
108        h[i] = ((b-a)/N[i]);
109    }
110    for(int i = 0; i < n; i++){
111        std::vector<double> t(n);
112        std::vector<double> y1(n);
113        std::vector<double> y2(n);
114        solve_test(h[i], x0, y0, t, y1, y2);
115        std::vector<double> y1ex(n);
116        std::vector<double> y2ex(n);
117        exact(h[i], y1ex, y2ex);
118
119        std::vector<double> y1_error(n);
120        std::vector<double> y2_error(n);
121        double max_y1_error = -1000;
122        double max_y2_error = -1000;
123        for(int j = 0; j < n; j++){

```

```

124         y1_error[j] = (abs(y1[j] - y1ex[j]));
125         y2_error[j] = (abs(y2[j] - y2ex[j]));
126         if(y1_error[j] > max_y1_error){
127             max_y1_error = y1_error[j];
128         }
129         if(y2_error[j] > max_y2_error){
130             max_y2_error = y2_error[j];
131         }
132     }
133     e[i] = (std::max(max_y1_error, max_y2_error));
134 }
135
136 for(int i = 0; i < n; i++){
137     double h4 = pow(h[i], 4);
138     e4[i] = (e[i]/h4);
139 }
140
141 return 0;
142 }

```