Санкт-Петербургский Политехнический Университет
Институт компьютерных наук и технологий
Высшая школа программной инженерии

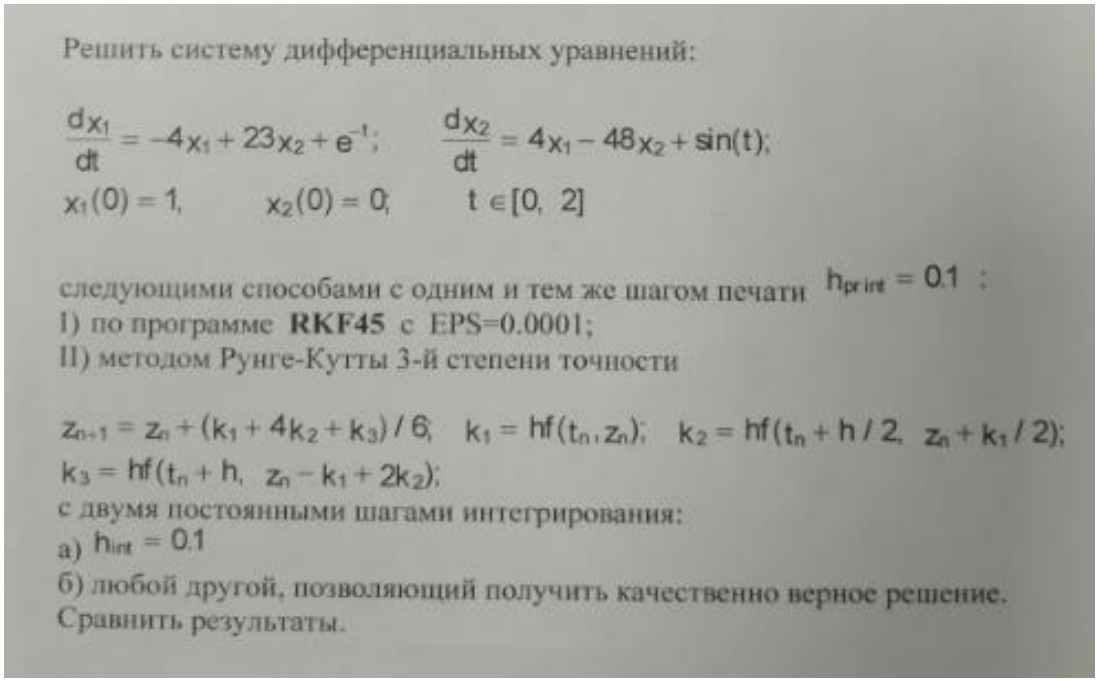
Отчёт по лабораторной работе №3

По дисциплине «Вычислительная математика»

Студент: Бабинцева К. А.
Группа: 23534/2
Преподаватель: Леонтьева Т. В.

Санкт-Петербург
2019

Постановка задачи:



Решить систему дифференциальных уравнений:

$$\frac{dx_1}{dt} = -4x_1 + 23x_2 + e^{-t}; \qquad \frac{dx_2}{dt} = 4x_1 - 48x_2 + \sin(t);$$

$$x_1(0) = 1, \qquad x_2(0) = 0; \qquad t \in [0,\ 2]$$

следующими способами с одним и тем же шагом печати $h_{print} = 0.1$:

I) по программе **RKF45** с EPS=0.0001;

II) методом Рунге-Кутты 3-й степени точности

$$z_{n+1} = z_n + (k_1 + 4k_2 + k_3)/6; \quad k_1 = hf(t_n, z_n); \quad k_2 = hf(t_n + h/2,\ z_n + k_1/2);$$

$$k_3 = hf(t_n + h,\ z_n - k_1 + 2k_2);$$

с двумя постоянными шагами интегрирования:

а) $h_{int} = 0.1$

б) любой другой, позволяющий получить качественно верное решение.

Сравнить результаты.

Ход работы:

1) Решена система дифференциальных уравнений с использованием программы RKF45 с EPS = 0.0001
2) Решена система дифференциальных уравнений методом Рунге-Кутты 3-й степени точности. Сначала с шагом h = 0.1, а затем с h = 0.03.
3) Сравнены результаты.

Результат работы программы:

RKF45:

step	x1	x2
0	1	0
0.1	0.869649	0.0750559
0.2	0.791993	0.0709935
0.3	0.725603	0.067333
0.4	0.668841	0.0643791
0.5	0.620387	0.0621164
0.6	0.579168	0.0603153
0.7	0.544099	0.05901
0.8	0.514319	0.058001
0.9	0.489007	0.0571929
1	0.467376	0.0566014
1.1	0.44879	0.0560859
1.2	0.432664	0.055549
1.3	0.418391	0.0550595
1.4	0.405527	0.0544875
1.5	0.393665	0.0537391
1.6	0.382361	0.0528934
1.7	0.37131	0.0518567
1.8	0.360238	0.0505665
1.9	0.348834	0.0491207
2	0.336926	0.0474294

(a) Решение с помощью программы RKF45

Runge Kutta method 3 degrees of accuracy:

step	x1	x2
0	1	0
0.1	0.839813	0.50368
0.2	1.11429	-2.1244
0.3	-0.776689	11.386
0.4	8.55208	-58.2318
0.5	-39.8576	300.369
0.6	209.208	-1546.89
0.7	-1074.05	7968.77
0.8	5536.11	-41048.7
0.9	-28514.7	211453
1	146889	-1.08924e+006
1.1	-756659	5.61097e+006
1.2	3.89774e+006	-2.89035e+007
1.3	-2.00782e+007	1.48889e+008
1.4	1.03428e+008	-7.66964e+008
1.5	-5.32783e+008	3.95082e+009
1.6	2.74449e+009	-2.03516e+010
1.7	-1.41376e+010	1.04836e+011
1.8	7.28261e+010	-5.40038e+011
1.9	-3.75145e+011	2.78187e+012
2	1.93247e+012	-1.43301e+013

(b) Решение, полученное методом Рунге-Кутты 3 степени точности

Рис.2: Результат работы программы при шаге h=0.1

Затем был подсчитан шаг следующим способом:

$h < \frac{2}{\lambda_{k\,max}}$; $\lambda_{k\,max}$ – максимальное по модулю собственное число.

Для матрицы:

-4 23

4 -48

Максимальное по модулю собственное значение = 48. Значит h < 0.042. Возьмем 0.03

step	x1	x2
0	1	0
0.03	0.940009	0.0613133
0.06	0.905783	0.0736919
0.09	0.87822	0.075238
0.12	0.853042	0.0744047
0.15	0.829235	0.0731301
0.18	0.806545	0.0718288
0.21	0.784884	0.0705869
0.24	0.764203	0.0694198
0.27	0.74446	0.068328
0.3	0.725616	0.0673084
0.33	0.707634	0.0663576
0.36	0.690481	0.0654722
0.39	0.67412	0.0646485
0.42	0.658519	0.0638834
0.45	0.643647	0.0631736
0.48	0.629471	0.0625158
0.51	0.615963	0.061907

(a) Решение, полученное подпрограммой RKF45

step	x1	x2
0	1	0
0.03	0.9269	0.0690041
0.06	0.895262	0.0760224
0.09	0.869651	0.0755428
0.12	0.845699	0.0742248
0.15	0.822839	0.0728794
0.18	0.800967	0.0716023
0.21	0.780041	0.0704021
0.24	0.760023	0.0692767
0.27	0.74088	0.0682228
0.3	0.722579	0.0672371
0.33	0.705087	0.0663163
0.36	0.688373	0.0654571
0.39	0.672406	0.0646564
0.42	0.657158	0.0639112
0.45	0.6426	0.0632185
0.48	0.628704	0.0625754
0.51	0.615443	0.0619791

(b) Решение, полученное методом Рунге-Кутты 3 степени точности

Рис.2: Результат работы программы при шаге h=0.03

|rkf45 - Runge Kutta Method(3)|:

step	x1	x2
0	0	0
0.03	0.0171066	0.00444651
0.06	0.0197494	0.00172779
0.09	0.0229103	0.0042466
0.12	0.0265311	0.00516297
0.15	0.0301592	0.00563701
0.18	0.0336567	0.00599262
0.21	0.0369948	0.00630921
0.24	0.0401729	0.00660559
0.27	0.0431971	0.0068865
0.3	0.0460743	0.00715351
0.33	0.0488117	0.00740745
0.36	0.0514158	0.00764899
0.39	0.0538932	0.00787871
0.42	0.0562498	0.00809716
0.45	0.0584914	0.00830489
0.48	0.0606236	0.00850237
0.51	0.0626514	0.0086901

Рис.3: Разница между решением, полученным с помощью RKF45, и решением методом Рунге-Кутты 3 степени точности

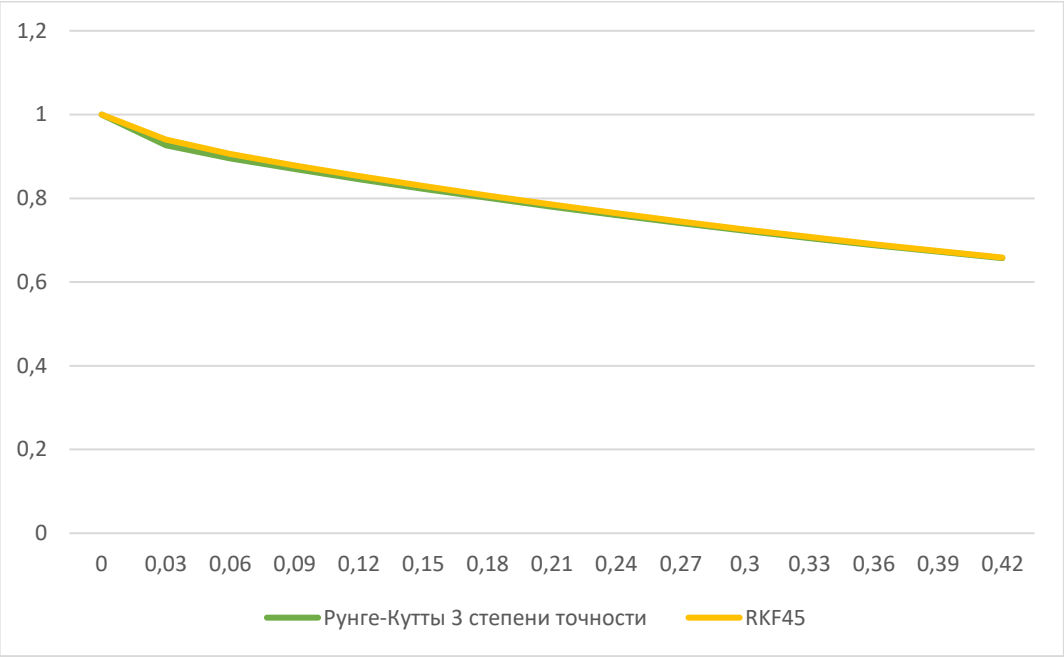


Рис. 4: Зависимость y от t для решения, полученного с помощью RKF45 и решения, полученного методом Рунге-Кутты 3 степени точности

Вывод:

Таким образом, решение, полученное с помощью подпрограммы RKF45 и используемым внутри нее методом Рунге–Кутты–Фельберга, оказалось более точным, чем решение, полученное с помощью метода Рунге-Кутты 3-й степени точности. Полученный результат объясняется тем, что первый метод обладает четвертой степенью точности, в то время как последний обладает третьей степенью точности.

Приложение

Листинг 1. main.cpp

```
#include <iostream>
#include <iomanip>
#include <cmath>
#include "rkf45.hpp"
#include "functions.hpp"

const int N = 2;
```

```
const double START = 0.0;
const double STEP = 0.1;
const double END = 2.0;

void fun(double t, double *x, double *dx)
{
  dx[0] = -4 * x[0] + 23 * x[1] + exp(-t);
  dx[1] = 4 * x[0] - 48 * x[1] + sin(t);
}

double **rkf45(int N, int points)
{
  double x[] = {1, 0};
  double t = 0.0;
  double tout;
  double relerr = 1e-4;
  double abserr = 1e-4;
  int flag;
  double work[15];
  int iwork[5];
  double **values = new double *[points];

  for (int i = 0; i < points; i++)
  {
  values[i] = new double[N];
  }

  for (int i = 0; i < points; i++)
  {
    tout = START + STEP * i;
    flag = 1;
    RKF45(fun, N, x, &t, &tout, &relerr, &abserr, &flag, work, iwork);

    for (int j = 0; j < N; ++j)
    {
      values[i][j] = x[j];
```

```
   }
  }

  return values;
}

double **runge(int N, int points)
{
  double **values = new double *[points];
  double x[] = {1, 0};
  double k1[N];
  double k2[N];
  double k3[N];
  double tout;

  for (int i = 0; i < points; i++)
  {
   values[i] = new double[N];
  }

  for (int i = 0; i < N; ++i)
  {
   values[0][i] = x[i];
  }

  for (int i = 0; i < points; i++)
  {
   tout = START + STEP * i;
   k1[0] = STEP * fun1(tout, x[0], x[1]);
   k1[1] = STEP * fun2(tout, x[0], x[1]);

   k2[0] = STEP * fun1(tout + STEP / 2, x[0] + k1[0] / 2, x[1] + k1[1] / 2);
   k2[2] = STEP * fun2(tout + STEP / 2, x[0] + k1[0] / 2, x[1] + k1[1] / 2);

   k3[0] = STEP * fun1(tout + STEP, x[0] - k1[0] + 2 * k2[0], x[1] - k1[1]
+ 2 * k2[1]);
```

```
   k3[1] = STEP * fun2(tout + STEP, x[0] - k1[0] + 2 * k2[0], x[1] - k1[1]
+ 2 * k2[1]);

   for (int j = 0; j < N; ++j)
   {
   values[i][j] = x[j];
   }

   x[0] += (k1[0] + 4 * k2[0] + k3[0]) / 6;
   x[1] += (k1[1] + 4 * k2[1] + k3[1]) / 6;
  }

  return values;
}

void print_runge(int N, int points)
{
  double **values = new double *[points];
  values = runge(N, points);

  std::cout << "Runge Kutta method 3 degrees of accuracy: " <<
std::endl
          << std::left << std::setw(10) << "step"
          << std::left << std::setw(15) << "x1"
          << std::left << std::setw(15) << "x2" << std::endl;

  for (int i = 0; i < points; i++)
  {
   std::cout << std::setw(10) << START + STEP * i;
   for (int j = 0; j < N; j++)
   {
    std::cout << std::setw(15) << values[i][j];
   }
   std::cout << std::endl;
  }
}
```

```
void print_rkf45(int N, int points)
{
  double **values = new double *[points];
  values = rkf45(N, points);

  std::cout << "RKF45: " << std::endl
          << std::left << std::setw(10) << "step"
          << std::left << std::setw(15) << "x1"
          << std::left << std::setw(15) << "x2" << std::endl;

  for (int i = 0; i < points; i++)
  {
    std::cout << std::setw(10) << START + STEP * i;
    for (int j = 0; j < N; j++)
    {
      std::cout << std::setw(15) << values[i][j];
    }
    std::cout << std::endl;
  }
}

int main()
{
  int points = (END - START) / STEP + 1;
  print_rkf45(N, points);
  print_runge(N, points);
  return 0;
}
```