Министерство образования и науки РФ

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа программной инженерии

Отчёт по лабораторной работе №3

по дисциплине «Вычислительная математика»

Выполнил студент гр. 3530904/00001 Корж В.Д.

Руководитель Устинов С.М.

Оглавление

[Постановка задачи 3](#__RefHeading___Toc1147_75006022)

[Код программы 4](#__RefHeading___Toc1149_75006022)

[Результат работы программы 7](#__RefHeading___Toc2736_1577260916)

[Вывод 10](#__RefHeading___Toc1153_75006022)

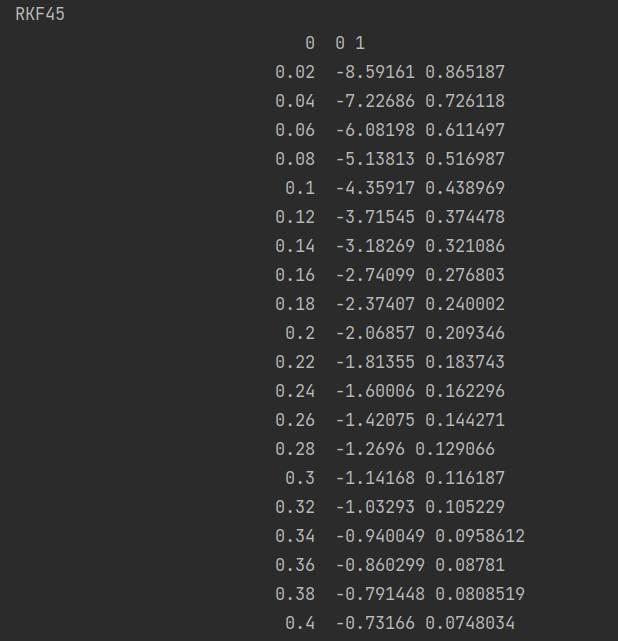
# Постановка задачи

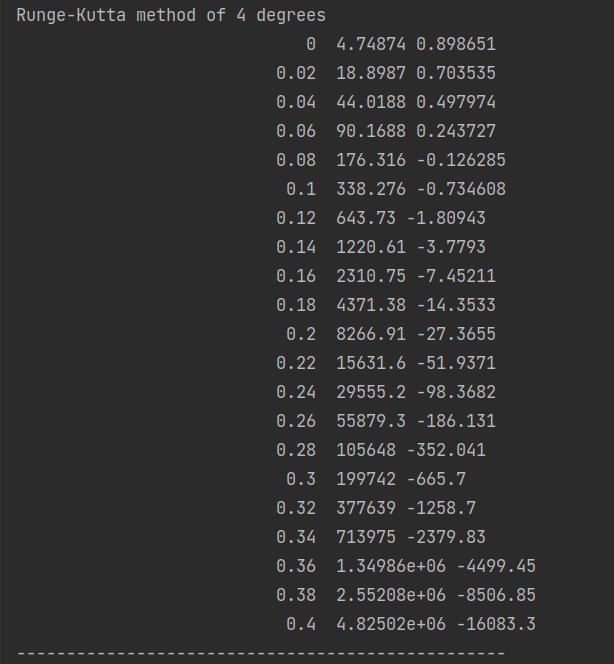
# 

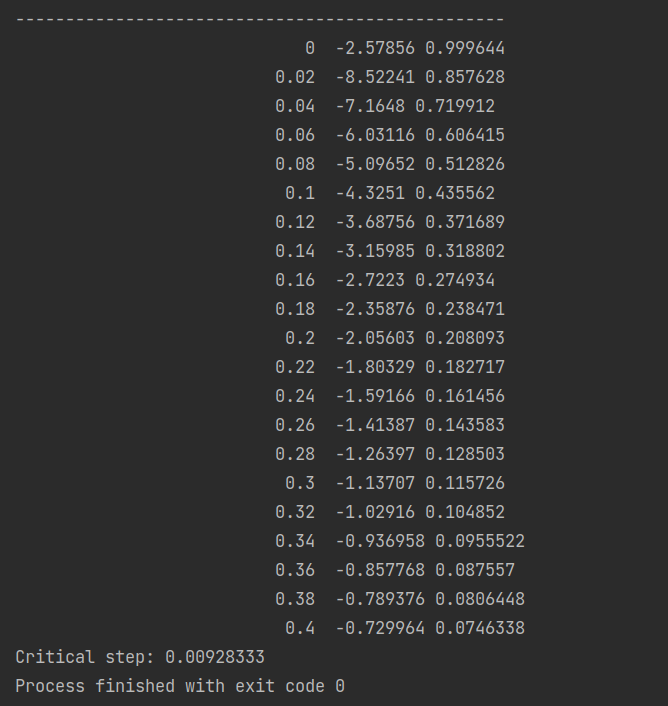
# Код программы

#include <iostream>  
#include <cmath>  
#include <iomanip>  
#include <FORSYTHE.H>  
  
double calculateCriticalStep() {  
 int x1, x2;  
 int a = 1;  
 int b = 309;  
 int c = 2690;  
 int d = b \* b - 4 \* a \* c;  
 if(d >= 0) {  
 x1 = ( -1\*b + sqrt(b\*b - 4\*a\*c) ) / (2 \* a);  
 x2 = ( -1\*b - sqrt(b\*b - 4\*a\*c) ) / (2 \* a);  
 }  
 else {  
 return -1;  
 }  
 return 2.785 / std::max(std::abs(x1), std::abs(x2));  
}  
  
void function(double t, double\* x, double\* dx) {  
 dx[0] = -310 \* x[0] - 3000 \* x[1] + 1 / (10 \* t \* t + 1);  
 dx[1] = x[0] + exp(-2 \* t);  
}  
  
void rungeKutta4(void (\*f)(double t, double\* x, double\* dx), double\* x, double t, double h) {  
 double k1[2], k2[2], k3[2], k4[2];  
 double z[2];  
 double result[2] = {0, 0};  
  
 z[0] = x[0];  
 z[1] = x[1];  
 function(t, z, result);  
 k1[0] = h \* result[0];  
 k1[1] = h \* result[1];  
  
 z[0] = x[0] + k1[0] / 3;  
 z[1] = x[1] + k1[1] / 3;  
 function(t + h / 3, z, result);  
 k2[0] = h \* result[0];  
 k2[1] = h \* result[1];  
  
 z[0] = x[0] + (-k1[0] / 3) + k2[0];  
 z[1] = x[1] + (-k1[1] / 3) + k2[1];  
 function(t + 2 \* h / 3, z, result);  
 k3[0] = h \* result[0];  
 k3[1] = h \* result[1];  
  
 z[0] = x[0] + k1[0] - k2[0] + k3[0];  
 z[1] = x[1] + k1[1] - k2[1] + k3[1];  
 function(t + h, z, result);  
 k4[0] = h \* result[0];  
 k4[1] = h \* result[1];  
  
 x[0] = x[0] + (k1[0] + 3 \* k2[0] + 3 \* k3[0] + k4[0]) / 8;  
 x[1] = x[1] + (k1[1] + 3 \* k2[1] + 3 \* k3[1] + k4[1]) / 8;  
}  
  
int main() {  
 double t = 0.0;  
 double tOut = 0.0;  
 double rungeH = 0.01;  
 double X[2] = {0.0, 1.0}; // x1 = 0; x2 = 1  
 double work[3 + 6 \* 2];  
 double relerr = 0.0000001;  
 double abserr = 0.0000001;  
 int flag = 1;  
  
 std::cout << "RKF45\n";  
 for (int i = 0; i <= 0.4 / rungeH; i++)  
 {  
 RKF45(function, 2, X, t, tOut, relerr, abserr, work, flag);  
 if (int temp = std::round(tOut \* 100); temp % 2 == 0) {  
 std::cout << std::setw(30) << tOut << '\t' << X[0] << ' ' << X[1] << '\n';  
 }  
 tOut += rungeH;  
 }  
  
 X[0] = 0.0;  
 X[1] = 1.0;  
 std::cout << "Runge-Kutta method of 4 degrees\n";  
 tOut = 0.0;  
 for (int i = 0; i <= 0.4 / rungeH; i++) {  
 rungeKutta4(function, X, tOut - rungeH, rungeH);  
 if (int temp = std::round(tOut \* 100); temp % 2 == 0) {  
 std::cout << std::setw(30) << tOut << '\t' << X[0] << ' ' << X[1] << '\n';  
 }  
 tOut += rungeH;  
 }  
  
 X[0] = 0.0;  
 X[1] = 1.0;  
 std::cout << "-------------------------------------------------\n";  
 tOut = 0.0;  
 rungeH = 0.001;  
 for (int i = 0; i <= 0.4 / rungeH; i++) {  
 rungeKutta4(function, X, tOut - rungeH, rungeH);  
 if (int temp = std::round(tOut \* 1000); temp % 20 == 0) {  
 std::cout << std::setw(30) << tOut << '\t' << X[0] << ' ' << X[1] << '\n';  
 }  
 tOut += rungeH;  
 }  
  
 std::cout << "Critical step: " << calculateCriticalStep();  
}

# Результат работы программы







# Вывод

По полученным результатам можно сделать следующий вывод. Метод Рунге-Кутты 4-ой степени точности требует маленький шаг, чтобы результаты исходных вычислений были точными, при большом шаге, результаты будут не точные. Данный факт мы можем заметить при сопоставление результатов данного метода с результатами RKF45 при таком же шаге. Полученные результаты будут отличаться. Полученный критический шаг (h = 0.00928333) позволит нам вывести более корректные результаты.