Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Санкт-Петербургский политехнический университет Петра Великого»

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Лабораторная работа №3**

по дисциплине «Вычислительная математика»

**Вариант №21**

Выполнил

Студент группы 3530901/80004 Иванов К. А.

\_\_\_\_\_\_\_\_\_\_\_

Преподаватель Цыган В. Н.

\_\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург 2020

**Постановка задачи:**

Привести дифференциальное уравнение к системе двух дифференциальных уравнений первого порядка.

Решить на интервале 1≤ ≤ 2

Начальные условия: ,

Точное решение: √

Решить с разным шагом , и с выбранной погрешностью EPS из диапазона 0.001 – 0.00001, возьмем 0.0001.

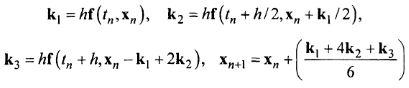
Решаем дифф.ур следующими методами:

Основной (обязательный):

1. RKF45 (из учебника С. М. Устинова, метод Рунге-Кутты-Фельберга четвертой и пятой степени).

Дополнительные (индивидуально полученная по заданию):

1. Рунге-Кутты 3-й степени точности (из учебника С. М. Устинова):



(метод 3-ей степени точности)

1. Усоверш. метод ломанных Эйлера (из учебника С. М. Устинова):

(метод 2-ой степени точности)

**Описание решения:**

1. Построение функции и графика точного решения;
2. Используя RKF45 решить на заданном интервале 1<=t<=2 и отобразить графически решение с помощью метода;
3. Используя метод Рунге-Кутты 3-й степени точности решить на интервале и отобразить графически решение с помощью метода;
4. Используя усовершенствованный метод ломаных Эйлера решить на интервале и отобразить графически решение с помощью метода;;
5. Оценить и сравнить локальные и глобальные погрешности.

Вычисления и построения функции и графиков будет выполнено при помощи двух популярных библиотек языка Python 3, SciPy и matplotlib.

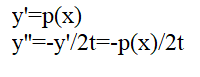
Ознакомиться с тем, как работают внутренне та или иная операция можно на соответствующих сайта с документацией по библиотекам.

SciPy:[docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.ode.html#scipy-integrate-ode](https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.ode.html#scipy-integrate-ode)

matplotlib:[matplotlib.org/api/\_as\_gen/matplotlib.pyplot.html?highlight=plot#module-matplotlib.pyplot](https://matplotlib.org/api/_as_gen/matplotlib.pyplot.html?highlight=plot#module-matplotlib.pyplot)

**Ход работы:**

Дифф.ур. вида , приведем к системе двух дифф.ур. первого порядка. Для этого выполним замену y`=p(x), получая систему вида:



Именно эту систему будем решать на интервале t∈[1,2]

Функция pick\_step – выборка точек некоторой функции, на указанном промежутке и с указанным шагом.

Программа RKF45 описана в некотором методе integrate.ode, библиотеки SciPy языка Python, которую можно получить, если установить ей параметр “dopri5”, устанавливаем требуемую по заданию погрешность(absolute tolerance) atol = 0.001.

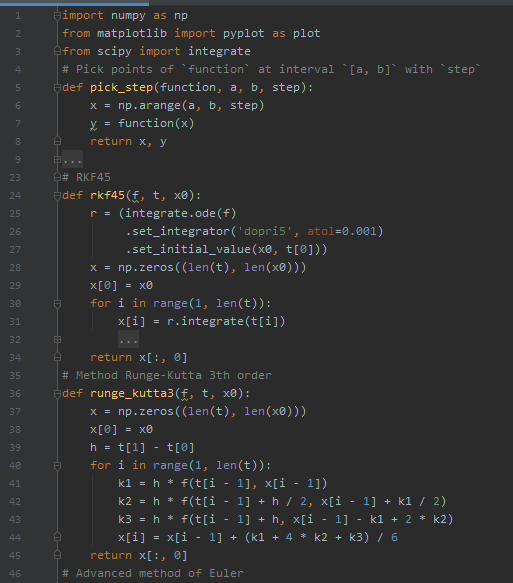
Функция runge\_kutta3 – разворачивает одноименный метод для решения.

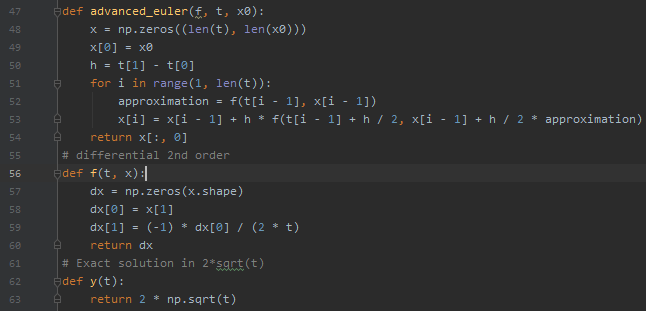
Функция advanced\_euler – разворачивает одноименный метод для решения.

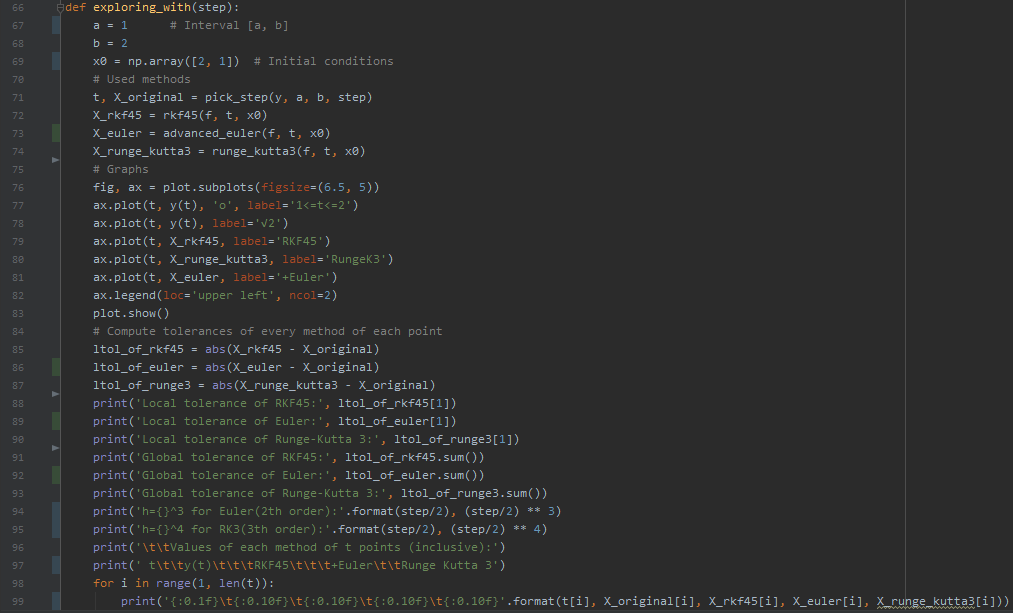
Функции f(t, x) – система уравнений, а y(t) – точное решение.

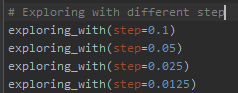
Учитывая потребность в расчете с разным шагом примем функцию exploring\_with – которая принимает разный шаг на вход, а внутри представляет из себя установку интервала, начальные условия, расчет используя нужные методы, вывод графика с точным решением и используемыми при решении методами, вычисление погрешности, и вывод их на экран, а так же вывод каждого значения конкретного метода в каждой точке на заданном интервале.

Ниже предоставлен код программы, который можно найти на [github.com:](https://github.com/b0r1ngx/ComputationalMath)

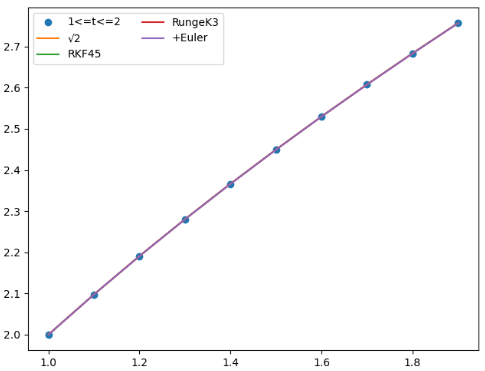
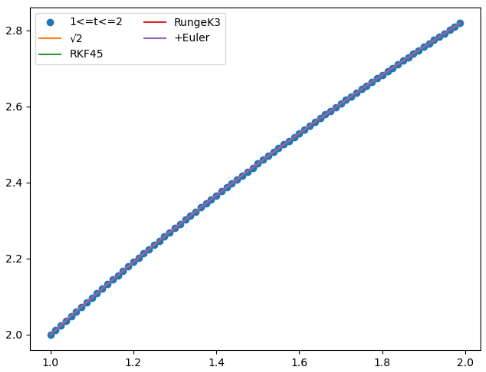






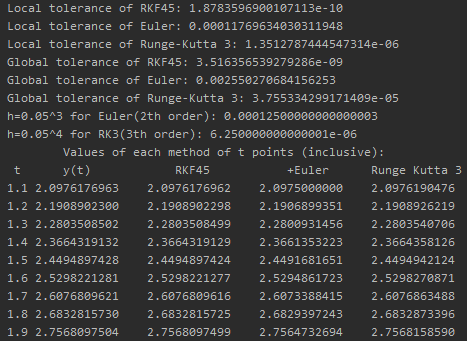


Графики получились примерно одинаковые и достаточно точные, поэтому по ним мало что можно оценить, приведу два с разным шагом. Также приведу значения функций в точках лишь для случая , чтобы сэкономить место.

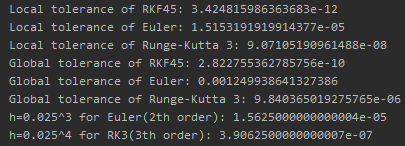
 

(h = 0.1) (h = 0.0125)

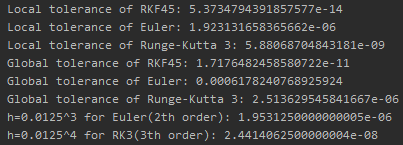
h = 0.1:



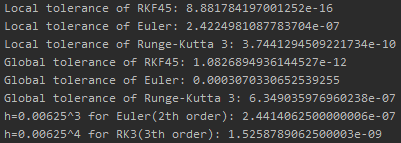
h = 0.05:



h = 0.025:



h = 0.0125:



**Выводы:**

Все методы отработали отлично, ведь по тому как они показывают себя на графике, с точностью можно сказать, что все методы справились со своей работой.

Исходя из полученных результатов можно сделать вывод, что при уменьшении шага интегрирования в 2 раза погрешности уменьшаются примерно на 1-2 порядка.

Дополнительно добавил сравнение локальных погрешностей между заданных мною методов и шага интегрирования. Вывод о зависимости можно сделать посмотрев на предоставление для каждого шага прикрепленные рисунки. Для усовершенствованного метода ломаных Эйлера, это 1 в 1 точно, для Рунге-Кутты 3 степени, это условие не выполняется на порядок, и тот метод который в моей программе и применяя их к нахождению решения данного уравнения имеет локальную погрешность на порядок меньше чем выражение (h/2)^4.

Примечательно то, что глобальная погрешность накапливается достаточно быстро, достигая весьма высоких значений по сравнению со значениями локальных погрешностей каждого из методов, даже несмотря на то, что наша функция на заданном промежутке ведет себя «плавно» при работе всех методов. Изучив работу и посмотрев на работу методов, можно сделать вывод о том, что в более выигрышном свете предстает RKF45, за ним идет Рунге-Кутта 3 степени и хуже всего справляется усоверш. метод ломаных Эйлера, ну конечно, становится понятно почему так, при анализе работы данных моделей и в целом математических функций на деле, если сравнить время работы каждого из методов то можно заметить, что именно метод Эйлера быстрее всего проводит свою работу.

Весь исходный код и файлы можно найти в моем профиле:

[github.com/b0r1ngx/ComputationalMath](https://github.com/b0r1ngx/ComputationalMath)