## Национальный исследовательский университет «МЭИ»

## (Московский Энергетический Институт)

**Кафедра математического и компьютерного моделирования**

## Численные методы

## Лабораторная работа №7:

**«ЧИСЛЕННОЕ РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ СИСТЕМ ОДУ»**

*Выполнил:* Солонин Е. В. А-14-19

*Преподаватель:* Амосова О. А.

Вариант 41

2021 Москва

**Постановка задачи:**

**Задача 7.1.** Дана задача Коши для системы двух обыкновенных дифференциальных уравнений



ПОРЯДОК РЕШЕНИЯ ЗАДАЧИ:

1. Модифицировать программу решения задачи по явному методу Эйлера из лабораторной работы 6 для решения системы дифференциальных уравнений. Найти приближенное решение задачи Коши с шагом *h*=0.01. на отрезке [0,1]. Оценить величину погрешности по правилу Рунге.
2. Модифицировать программу решения задачи по индивидуальному варианту из лабораторной работы 6 для решения системы дифференциальных уравнений. Найти приближенное решение задачи Коши с шагом *h*=0.01 на отрезке [0,1]. Оценить величину погрешности по правилу Рунге.
3. На одном чертеже построить графики первой компоненты *u(t*) найденного обоими методами решения, а на другом - графики второй компоненты *v(t*) найденного обоими методами решения.
4. Сравнить полученные результаты.

|  |  |
| --- | --- |
| N |  |
| 7.1.41 |  |

**Теоретический материал:**

Метод Эйлера:

Усовершенствованный метод Эйлера:

**Код программы** *(пояснения к коду красным цветом)***:**

**import** numpy  
**import** matplotlib.pyplot **as** plt  
  
*# задача Коши***def** dy(y, t):  
 u, v = y  
 du = t \* (u \*\* 2) \* numpy.sqrt(v)  
 dv = numpy.sqrt(u) + numpy.sqrt(v)  
 **return** [du, dv]  
  
*# формула метода Эйлера***def** Euler(y, t, h):  
 u, v = y  
 y0, y1 = dy(y, t)  
 euler1, euler2 = u + h \* y0, v + h \* y1  
 **return** [euler1, euler2]  
  
*# формула усовершенствованного метода Эйлера***def** Euler\_modified(y, t, h):  
 u, v = y  
 y0, y1 = dy(y, t)  
 y\_support0, y\_support1 = dy([u + h / 2 \* y0, v + h / 2 \* y1], t + h / 2)  
 eulerMod1, eulerMod2 = u + h \* y\_support0, v + h \* y\_support1  
 **return** [eulerMod1, eulerMod2]  
  
*# расчет числа точек N для шага h*get\_n = **lambda** h: int(numpy.ceil((t\_n - t\_0) / h))  
  
*# получить массив решений выбранным методом с шагом h для двух компонент u и v***def** get\_y\_data(h, Method):  
 n = get\_n(h)  
 t\_data = numpy.linspace(t\_0, t\_n, n + 1)  
 y\_data = [[1, 1]]  
 **for** i **in** range(1, len(t\_data)):  
 y\_data.append(Method(y\_data[i - 1], t\_data[i - 1], h))  
 y\_data = numpy.array(y\_data)  
 u, v = y\_data[:, 0], y\_data[:, 1]  
 **return** [u, v]  
  
*# максимальное значение погрешности по правилу Рунге***def** Runge(y\_data\_h, y\_data\_2h):  
 r\_data = [0]  
 **for** i **in** range(1, len(y\_data\_2h)):  
 r\_data.append(numpy.abs((y\_data\_h[2 \* i] - y\_data\_2h[i])))  
 **return** max(r\_data)  
  
*# функция, осуществляющая: поиск решения выбранным методом;  
# оценку погрешности по правилу Рунге***def** solve(Method, methodName):  
 u\_h, v\_h = get\_y\_data(h, Method)  
 u\_2h, v\_2h = get\_y\_data(2 \* h, Method)  
  
 max\_r\_u = Runge(u\_h, u\_2h)  
 max\_r\_v = Runge(v\_h, v\_2h)  
  
 print(**"\n"**, methodName, **":"**)  
 print(**"Max Runge error for u(t): "**, max\_r\_u)  
 print(**"Max Runge error for v(t): "**, max\_r\_v)  
 **return** [u\_h, v\_h]  
  
*# построение графиков для двух методов одной компоненты решения***def** show\_graph(data\_euler, data\_eulerMod, title):  
 n = get\_n(h)  
 t\_data = numpy.linspace(t\_0, t\_n, n + 1)  
 fig, ax = plt.subplots()  
 plt.plot(t\_data, data\_euler, label=**"Euler"**)  
 plt.plot(t\_data, data\_eulerMod, label=**"Euler modified"**)  
 ax.set\_title(title)  
 plt.legend()  
 plt.show()  
  
t\_0 = 0  
t\_n = 1  
h = 0.01  
  
u\_euler, v\_euler = solve(Euler, **"Euler"**)  
u\_eulerMod, v\_eulerMod = solve(Euler\_modified, **"Euler modified"**)  
  
show\_graph(u\_euler, u\_eulerMod, **"u(t)"**)  
show\_graph(v\_euler, v\_eulerMod, **"v(t)"**)

**Результаты выполнения:**

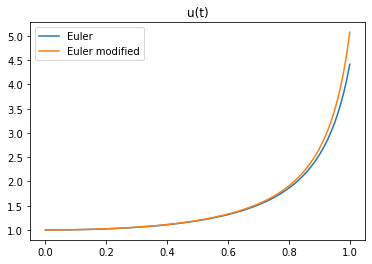
Euler :

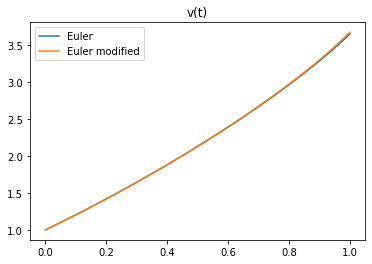
Max Runge error for u(t): 0.454982637144246

Max Runge error for v(t): 0.021519478844226114

Euler modified :

Max Runge error for u(t): 0.061646522345664145

****Max Runge error for v(t): 0.0012717212391528676

****

**Анализ результатов:**

**Постановка задачи:**

**Задача 7.2.** Дана задача Коши для двух систем обыкновенных дифференциальных уравнений с постоянными коэффициентами

 ,

 ,

где  и  – заданные матрицы,  - заданные векторы. Исследовать поведение решения систем уравнений

ПОРЯДОК РЕШЕНИЯ ЗАДАЧИ:

1. 1. Используя встроенную функцию пакета NUMPY для нахождения собственных чисел матриц *A* и *B*, найти коэффициенты жесткости обеих систем. Установить какая задача является жесткой.
2. Численно решить обе задачи на отрезке [0,1] c шагом *h*=0.01 явным методом Эйлера. Определить, для какой из задач явный метод неустойчив при данном шаге *h*. Построить графики компонент полученного решения.
3. Численно решить обе задачи на отрезке [0,1] c шагом *h*=0.01 по индивидуальному варианту из лабораторной работы 6. Определить, для какой из задач метод неустойчив при данном шаге *h*. *h*=0.01. Построить графики компонент полученного решения.
4. Для жесткой задачи экспериментально подобрать шаг *h*, при котором графики компонент решения, полученного по явному методу Эйлера, визуально совпадают с графиками компонент решения, полученного по неявному методу с шагом *h*=0.01. Сравнить найденное значение шага с теоретическим значением шага, при котором явный метод Эйлера для жестких задач должен быть устойчивым.
5. Сравнить полученные результаты.

**Теоретический материал:**

Явный метод Эйлера:

Неявный метод Эйлера:

Усовершенствованный метод Эйлера:

**Код программы** *(пояснения к коду зеленым цветом)***:**

**Результаты выполнения:**

**Анализ результатов:**