МIНIСТЕРСТВО ОСВIТИ I НАУКИ УКРАЇНИ

НАЦIОНАЛЬНИЙ ТЕХНIЧНИЙ УНIВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛIТЕХНIЧНИЙ IНСТИТУТ»

Кафедра прикладної математики

Звіт

з лабораторної роботи №3

з дисципліни «Алгоритми і системи комп’ютерної математики-2.

Програмні засоби»

на тему:

«Розв’язання звичайних диференційних рівнянь вищих порядків та систем диференційних рівнянь»

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-63 | *Старший викладач Бай Ю.П.* |
| *Артеменко Я. К.* |  |

KM-6301.docx

KM-6301.py

KM-6301.m

Київ — 2020

# **ЗМІСТ**

[1 ВСТУП 2](#_Toc36147735)

[2 ОСНОВНА ЧАСТИНА 3](#_Toc36147736)

[2.1 Постановка задачі 3](#_Toc36147737)

[2.2 Описання методу 3](#_Toc36147738)

[2.3 Порядок виконання роботи 4](#_Toc36147739)

[2.4 Контрольні запитання 5](#_Toc36147740)

[2.5 Опис програмних засобів 6](#_Toc36147741)

[3 ВИСНОВКИ 7](#_Toc36147742)

[4 СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 8](#_Toc36147743)

[ДОДАТКИ 9](#_Toc36147744)

[Додаток А (код програми GNU Octave) 9](#_Toc36147745)

[Додаток Б (код програми Python) 11](#_Toc36147746)

[Додаток B (результати виконання ) 12](#_Toc36147747)

[Додаток Г (порівняння результатів) 16](#_Toc36147748)

# **1 ВСТУП**

Метою даної лабораторної роботи є розробка програмного забезпечення, за допомогою якого можна буде розв’язувати звичайні диференційні рівняння вищих порядків та систем диференційних рівнянь. Програмне забезпечення організовується за допомогою мови Python та програмного пакету математичних обчислень GNU Octave. Програмне забезпечення необхідно реалізувати в двох варіантах: з використанням вбудованих функцій бібліотек Python та GNU Octave, а також власноруч реалізувати метод.

# **2 ОСНОВНА ЧАСТИНА**

# **2.1 Постановка задачі**

За допомогою Python та GNU Octave розв’язати задану задачу Коші для звичайного диференційного рівняння другого порядку.

Варіант №1:

Розв’язати поставлену задачу згідно варіанту потрібно методом Рунге-Кутти третього порядку.

# **2.2 Описання методу**

Методи Рунге-Кутти – важлива група [чисельних методів](https://uk.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%B5%D0%BB%D1%8C%D0%BD%D1%96_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8) розв’язування (систем) звичайних [диференціальних рівнянь](https://uk.wikipedia.org/wiki/%D0%94%D0%B8%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D1%96%D0%B0%D0%BB%D1%8C%D0%BD%D1%96_%D1%80%D1%96%D0%B2%D0%BD%D1%8F%D0%BD%D0%BD%D1%8F). Названі на честь німецьких математиків [Карла Рунге](https://uk.wikipedia.org/wiki/%D0%9A%D0%B0%D1%80%D0%BB_%D0%A0%D1%83%D0%BD%D0%B3%D0%B5) і [Мартіна Кутти](https://uk.wikipedia.org/w/index.php?title=%D0%9C%D0%B0%D1%80%D1%82%D1%96%D0%BD_%D0%9A%D1%83%D1%82%D1%82%D0%B0&action=edit&redlink=1), які відкрили ці методи. Це найбільш поширений метод розв'язання задачі Коші.

Для методу Рунге-Кутти 3-го порядку значення невідомої функції в точці {\displaystyle \ x\_{n+1}}xn+1 обчислюється відносно значення в попередній точці {\displaystyle \ x\_{n}}xn за формулою:

де {\displaystyle \ h}h — крок інтегрування, а коефіцієнти {\displaystyle {\textbf {k}}\_{n}}kn розраховуються таким чином:

# **2.3 Порядок виконання роботи**

1. Записуємо вхідні дані відповідно умови варіанту, тобто саме диференціальне рівняння, початкові умови, інтервал та крок.
2. Понижується степінь диференціального рівняння для того, щоб звести диференціальне рівняння другого порядку до системи двох диференціальних рівнянь 1-го порядку.
3. Далі проводиться обчислення по формулам відповідно до методу.
4. Будуються графіки розв’язків, отриманих за допомогою бібліотечних функцій та власноруч розроблених програм.

Представлену вище послідовність дій необхідно реалізувати у вигляді програмного забезпечення двома способами: використовуючи мову Python, а також за допомогою програмного пакету математичних обчислень GNU Octave.

# **2.4 Контрольні запитання**

1. Що таке диференціальні рівняння?

Диференційні рівняння — [рівняння](https://uk.wikipedia.org/wiki/%D0%A0%D1%96%D0%B2%D0%BD%D1%8F%D0%BD%D0%BD%D1%8F), що встановлюють залежність між незалежними змінними, числами (параметрами), невідомими [функціями](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D1%8F_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) та їхніми [похідними](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D1%85%D1%96%D0%B4%D0%BD%D0%B0). Невідома функція може бути як скалярною, так і векторною.

1. Яке рівняння називається диференціальним рівнянням n-го порядку?

Диференціальним рівнянням n-го порядку називається рівняння вигляду:

або, якщо воно розв’язне відносно старшої похідної:

де функція визначена і неперервна в деякій області зміни своїх аргументів.

1. Що називається розв’язком диференціального рівняння n-го порядку?

Розв’язком диференціального рівняння називається будь-яка n разів неперервно-диференційовна функція , яка при підстановці в це рівняння обертає його в тотожність:

1. За якою формулою значення невідомої функції в точці {\displaystyle \ x\_{n+1}}(n+1)?
2. За якими формулами знаходяться коефіцієнти {\displaystyle {\textbf {k}}\_{n}}kn розраховуються таким чином:

# **2.5 Опис програмних засобів**

В процесі реалізації поставленої задачі були використані наступні вбудовані функції бібліотек numpy (np), scipy, matplotlib мови Python:

1. np.linespace - повертає одновимірний масив із зазначеної кількості елементів, значення яких рівномірно розподілені усередині заданого інтервалу;
2. plt.plot – використовується для побудови графіків;
3. round() – використовується для округлення значень;
4. plt.show() – використовується для того, щоб вивести графік на екран;

В процесі реалізації поставленої задачі були використані наступні вбудовані функції GNU Octave:

1. odeset() – використовується для того, щоб задати крок;
2. plot – використовується для того, щоб виконати побудову графіку функції;
3. ode45 – використовується для вирішення диференціальних рівнянь та є вбудованою.

# **3 ВИСНОВКИ**

В даній лабораторній роботі було розроблено програмне забезпечення, що реалізує вирішення диференціальних рівнянь вищого порядку методом Рунге-Кутти третього порядку. В результаті виконання лабораторної роботи було створено дві програми з використанням мови Python, а також за допомогою програмного пакету математичних обчислень GNU Octave. В кожному з цих середовищ розробки метод був реалізований як з використанням вбудованих функцій, так і за допомогою самостійно написаного алгоритму. Також було побудовано графіки розв’язків, отриманих за допомогою бібліотечних функцій та власноруч розроблених програм.

# **4 СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. Диференціальні рівняння. Навчальний посібник для інженерних спеціальностей [Електронний ресурс]: навч. посіб. для студ. спеціальності 131 «Прикладна меїханіка»/ КПІ ім. Ігоря Сікорського ; уклад.: І. М. Копась. – Електронні текстові данні (1 файл: 2504 Кбайт). – Київ : КПІ ім. Ігоря Сікорського, 2018. – 126 с.
2. Введение в Octave для инженеров и математиков: / Е.Р.Алексеев, О.В.Чеснокова — М.: ALT Linux, 2012. — 368с.: ил. — (Библиотека ALT Linux).

# **ДОДАТКИ**

# **Додаток А (код програми GNU Octave)**

h = 0.2

a = 2

b = 4

y0 = 1

z0 = -2

function output = my\_func(t, y)

y;

t;

output = [y(2); 2\*y(2) - y(1)];

endfunction

function[y,t]=RK(a,b,h,y0,z0)

n = (b-a)/h;

y(1,:) = [y0, z0];

for i = 1:n+1

t(i) = a + (i-1)\*h;

end

for i = 2:n+1

y(i - 1,:);

K1 = my\_func(t(i - 1), y(i - 1,:));

K2 = my\_func(t(i - 1) + h/2, y(i - 1,:)+ (h/2\*K1)');

K3 = my\_func(t(i - 1) + h, y(i - 1,:) - (h\*K1 + 2\*h\*K2)');

y(i,:) = [y(i - 1, 1) + h/6\*(K1(1) + 4\*K2(1) + K3(1)), y(i - 1, 2) + h/6\*(K1(2) + 4\*K2(2) + K3(2))];

end

end

par=odeset ('InitialStep' ,h , 'MaxStep' ,h) ;

[t\_1, y\_1] = ode45("@my\_func", [a, b], [y0, z0], par);

[y\_2,t\_2] = RK(a, b, h, y0, z0);

disp('Вбудована функція:')

disp(' [a; b]: y z ')

[t\_1 y\_1]

disp('Власна функція:')

disp(' [a; b]: y z ')

[t\_1 y\_2]

plot(t\_1, y\_1(:,1), "-o", t\_2, y\_2(:,1), "-o"), title('y');

legend("orange - Власна функція, \nblue -Вбудована функція");

grid on;

figure;

plot(t\_1, y\_1(:,2), "-o", t\_2, y\_2(:,2), "-o"), title('y\_dot');

legend("orange - Власна функція, \nblue -Вбудована функція");

grid on;

# **Додаток Б (код програми Python)**

import numpy as np  
from matplotlib.pylab import plt  
import unittest  
from scipy.integrate import odeint  
import numpy as np  
  
  
def my\_func(x, y, y1):  
 return -y + 2\*y1   
  
a = 2  
b = 4  
  
x0 = 2  
y0 = 1  
  
y1 = -2  
  
h = 0.2  
  
def model(x,t):  
 y = x[0]  
 dy = x[1]  
 K = 0  
 xdot = [[],[]]  
 xdot[0] = dy  
 xdot[1] = my\_func(K, y, dy)  
 return xdot  
  
time = np.linspace(a, b, 11)  
z2 = odeint(model,[y0,y1],time)  
print('Вбудована функція')  
print(' y', ' '\*13, 'z')  
print(z2)  
# plt.plot(z1['time'],z1['y'],'r-')  
# plt.plot(z1['time'],z1['dy'],'b--')  
plt.plot(time,z2[:,0],'g:')  
plt.plot(time,z2[:,1],'k-.')  
plt.legend(['y (Python)','dy/dt (Python)'])  
# plt.show()  
  
def getXY(x0, y1, a, b, h):  
  
 # numbers = [int, float]  
 #  
 # if type(x0) not in numbers or type(y1) not in numbers or type(a) not in numbers or type(b) not in numbers or type(h) not in numbers:  
 # raise TypeError('All input data must be numbers')  
 #  
 # if x0 < a or x0 > b:  
 # raise ValueError('x0 must be in bounds')  
 #  
 # if h > b - a:  
 # raise ValueError('step can`t be more than segment')  
  
 x\_list = [x0]  
 y\_sec\_der = [my\_func(x0, y0, y1)]  
 y\_derr = [y1]  
 y\_res\_list = [y0]  
  
 for i in range(100):  
 if (x\_list[-1] > b):  
 break  
  
 x\_list.append(x\_list[i] + h)  
  
 k0=h\*y\_derr[i]  
 l0=h\*my\_func(x\_list[i], y\_res\_list[i], y\_derr[i])  
 k1=h\*(y\_derr[i] + l0/2)  
 l1=h\*my\_func(x\_list[i]+h/2, y\_res\_list[i]+k0/2, y\_derr[i])  
 k2=h\*(y\_derr[i] + l1/2)  
 l2=h\*my\_func(x\_list[i]+h/2, y\_res\_list[i]+k1/2, y\_derr[i])  
 k3=h\*(y\_derr[i] + l2/2-l0)  
 l3=h\*my\_func(x\_list[i]+h, y\_res\_list[i]+k2-k0, y\_derr[i])  
  
  
 y\_res\_list.append(y\_res\_list[i] + 1/6 \* (k0+4\*k2+k3))  
 y\_derr.append(y\_derr[i] + 1/6 \* (l0+4\*l2+l3))  
  
 return x\_list, y\_derr, y\_res\_list  
  
print('Власна функція')  
print('x', ' ' \* 15 ,'y', ' '\*15, 'z')  
x, y1, y = getXY(x0, y1, a, b, h)  
  
for i in range(len(x)):  
 print(round(x[i], 5), ' ' \* 15, round(y[i], 5), ' ' \* 15, round(y1[i], 5))  
plt.plot(x, y, label='Runge–Kutta method')  
plt.plot(x,y1)  
plt.show()

# **Додаток B (результати виконання )**

Octave:

Вбудована функція:

[a; b]: y z

ans =

2.00000 1.00000 -2.00000

2.20000 0.48856 -3.17565

2.40000 -0.29837 -4.77384

2.60000 -1.45770 -6.92405

2.80000 -3.11576 -9.79238

3.00000 -5.43657 -13.59141

3.20000 -8.63231 -18.59266

3.40000 -12.97665 -25.14225

3.60000 -18.82153 -33.68063

3.80000 -26.61846 -44.76741

4.00000 -36.94530 -59.11247

Власна функція:

[a; b]: y z

ans =

2.00000 1.00000 -2.00000

2.20000 0.64400 -2.93200

2.40000 0.11443 -4.14816

2.60000 -0.64223 -5.72324

2.80000 -1.69367 -7.75023

3.00000 -3.12519 -10.34461

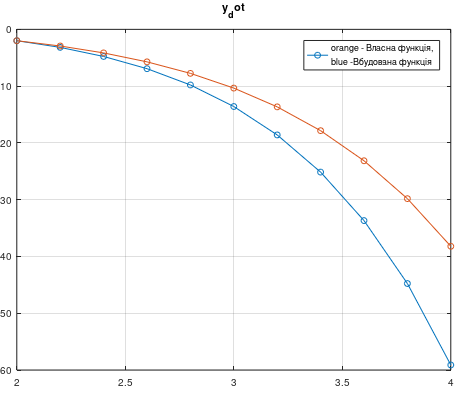
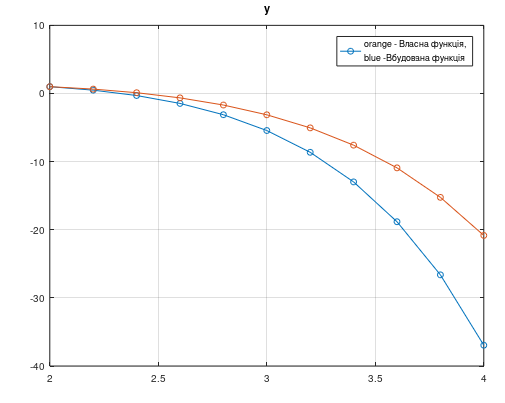
3.20000 -5.04397 -13.64952

3.40000 -7.58436 -17.84218

3.60000 -10.91432 -23.14164

3.80000 -15.24339 -29.81836

4.00000 -20.83248 -38.20584



Python:

Вбудована функція

y z

[[ 1. -2. ]

[ 0.48856109 -3.1756472 ]

[ -0.29836498 -4.77383909]

[ -1.45769511 -6.92405153]

[ -3.1157574 -9.79238021]

[ -5.4365638 -13.59140932]

[ -8.63230417 -18.59265498]

[-12.97664012 -25.14224007]

[-18.82152352 -33.68062086]

[-26.61844931 -44.76739179]

[-36.94528104 -59.11244944]]Власна функція

x y z

2 1 -2

2.2 0.5535 -2.96347

2.4 -0.09884 -4.20728

2.6 -1.01617 -5.79797

2.8 -2.27154 -7.81585

3.0 -3.95484 -10.35747

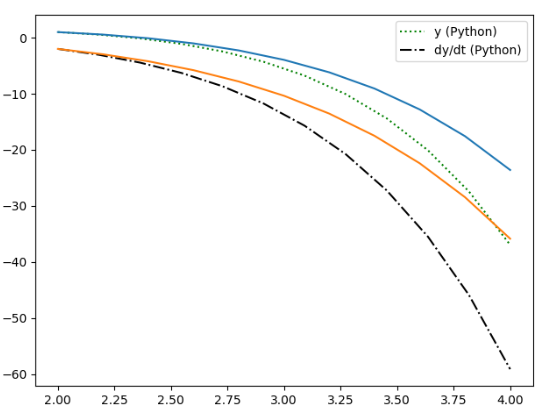
3.2 -6.17611 -13.53856

3.4 -9.06958 -17.49734

3.6 -12.79827 -22.39838

3.8 -17.55953 -28.43694

4.0 -23.59136 -35.84391



# **Додаток Г (порівняння результатів)**

Таблиця 1 – Порівняння отриманих значень «y» вбудованою функцією.

|  |  |
| --- | --- |
| y – Python | y – Octave |
| 1 | 1 |
| 0.48856109 | 0.48856 |
| -0.29836498 | -0.29837 |
| -1.45769511 | -1.45770 |
| -3.1157574 | -3.11576 |
| -5.4365638 | -5.43657 |
| -8.63230417 | -8.63231 |
| -12.97664012 | -12.97665 |
| -18.82152352 | -18.82153 |
| -26.61844931 | -26.61846 |
| -36.94528104 | -36.94530 |

Таблиця 2 – Порівняння отриманих значень «y» власною функцією.

|  |  |
| --- | --- |
| y – Python | y – Octave |
| 1 | 1 |
| 0.5535 | 0.64400 |
| -0.09884 | 0.11443 |
| -1.01617 | -0.64223 |
| -2.27154 | -1.69367 |
| -3.95484 | -3.12519 |
| -6.17611 | -5.04397 |
| -9.06958 | -7.58436 |
| -12.79827 | -10.91432 |
| -17.55953 | -15.24339 |
| -23.59136 | -20.83248 |

Таблиця 3 – Порівняння отриманих значень «z» вбудованою функцією.

|  |  |
| --- | --- |
| z – Python | z – Octave |
| -2 | -2 |
| -3.1756472 | -3.17565 |
| -4.77383909 | -4.77384 |
| -6.92405153 | -6.92405 |
| -9.79238021 | -9.79238 |
| -13.59140932 | -13.59141 |
| -18.59265498 | -18.59266 |
| -25.14224007 | -25.14225 |
| -33.68062086 | -33.68063 |
| -44.76739179 | -44.76741 |
| -59.11244944 | -59.11247 |

Таблиця 4 – Порівняння отриманих значень «z» власною функцією.

|  |  |
| --- | --- |
| z – Python | z – Octave |
| -2 | -2 |
| -2.96347 | -2.93200 |
| -4.20728 | -4.14816 |
| -5.79797 | -5.72324 |
| -7.81585 | -7.75023 |
| -10.35747 | -10.34461 |
| -13.53856 | -13.64952 |
| -17.49734 | -17.84218 |
| -22.39838 | -23.14164 |
| -28.43694 | -29.81836 |
| -35.84391 | -38.20584 |