Міністерство науки і освіти України

Національний університет “Львівська політехніка”

Інститут прикладної математики та фундаментальних наук

Кафедра прикладної математики

Звіт

про виконання лабораторної роботи №1

з курсу “Чисельні методи, частина 2”   
на тему  
“Однокрокові методи чисельного розв’язування задачі Коші для звичайних диференціальних рівнянь”

Виконав:  
студент групи ПМ-42  
Сватюк Д.Р.  
Перевірила:  
Візнович О.В.

Львів 2017

**Мета:** навчитися аналітично і чисельно розв'язувати задачі Коші.

**Постановка задачі**

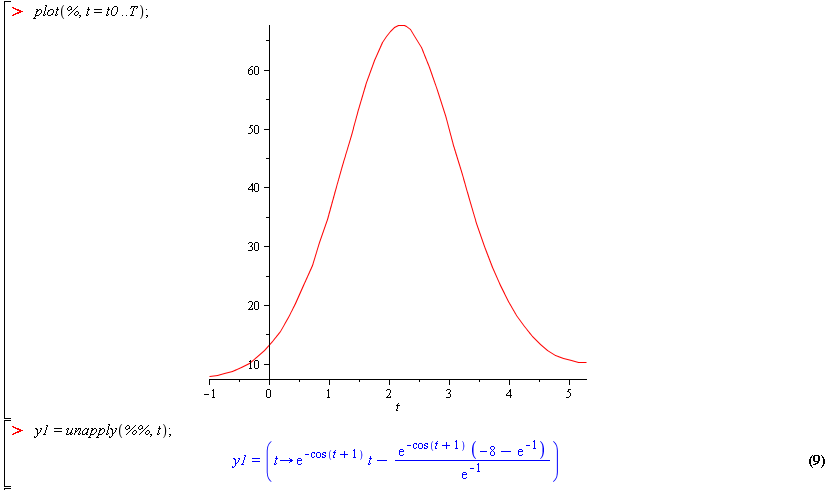
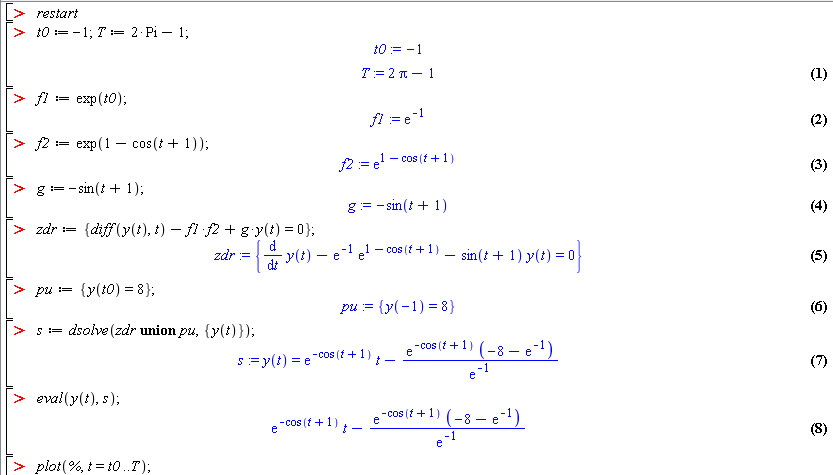
Використовуючи мову програмування Go, написати та відлагодити програму чисельного розв’язування задачі Коші для ЗДР вкладеним методом Рунге-Кутта 3-го порядку з автоматичним вибором кроку при заданій точності  та . Знайти аналітичний розв'зок цієї ж задачі з використанням середовища Maple.





|  |  |  |
| --- | --- | --- |
| № |  |  |
| 14 |  |  |

**Аналітичний розв’язок задачі**



**1454**

**Алгоритм чисельного розв’язування задачі Коші вкладеними**

**методами Рунге-Кутта**

1. **Ввести значення** , , , , , .

2. **Ініціалізувати змінні** ; ; ; .

**Вивести значення** ; ; ;.

3. **If**  **then go to** 10.

4. **If**  **then** .

5. ; .

6. **Обчислити** .

7. **Обчислити**



8. **Обчислити** , .

9. **If**  **then**

**begin** ; **обчислити точний розв’язок** ; ;

**вивести значення** , , ;;

**if**  **then** ;

**go to 3 end**

**else** **begin** ; ; ; **go to** 7 **end**.

10. **Вивести норму похибки** .

11. **End**.

**Програмний код**

package main

import (

"math"

"strconv"

)

const t0 = -1.0

const T = 2\*math.Pi - 1

const y0 = 8.0

const eps = 0.01

const tau0 = 0.2

const epsM = 0.0001

func main() {

method := RungeKuttaMethod{}

method.Start(t0, T, y0, eps, tau0, epsM)

println("FINISHED!")

}

// u'(t) = f(t,u) = (e^t0)\*(e^(1-cos(t+1))) + sin(t+1)\*u

func f(t float64, u float64) float64 {

return (1.0/math.E)\*math.Pow(math.E, (1-math.Cos(t+1))) + math.Sin(t+1)\*u

}

// Analytic solution of this differential equation found in Maple

func u(t float64) float64 {

return math.Pow(math.E, -math.Cos(t+1))\*t - math.E\*(math.Pow(math.E, -math.Cos(t+1))\*(-8.0-1.0/math.E))

}

type RungeKuttaMethod struct {

t          float64

T          float64

y          float64

tau        float64

eMax       float64

v          float64

t1         float64

w          float64

k1, k2, k3 float64

E          float64

tauH       float64

epsM       float64

}

func (m RungeKuttaMethod) Start(t0 float64, T float64, y0 float64, eps float64, tau0 float64, epsM float64) {

m.Step1(t0, T, y0, eps, tau0, epsM)

return

}

func (m RungeKuttaMethod) Step1(t0 float64, T float64, y0 float64, eps float64, tau0 float64, epsM float64) {

m.t = t0

m.T = T

m.y = y0

m.tau = tau0

m.eMax = 0

m.epsM = epsM

m.Step2()

return

}

func (m RungeKuttaMethod) Step2() {

println("t = " + strconv.FormatFloat(m.t, 'f', 8, 64))

println("y = " + strconv.FormatFloat(m.y, 'f', 8, 64))

println("u(t) = " + strconv.FormatFloat(u(m.t), 'f', 8, 64))

println("|y - u(t)| = " + strconv.FormatFloat(math.Abs(m.y-u(m.t)), 'f', 8, 64))

m.Step3()

return

}

func (m RungeKuttaMethod) Step3() {

if math.Abs(m.T-m.t) < m.epsM || m.t > m.T {

m.Step10()

return

} else {

m.Step4()

}

return

}

func (m RungeKuttaMethod) Step4() {

if m.t+m.tau > m.T {

m.tau = m.T - m.t

}

m.Step5()

return

}

func (m RungeKuttaMethod) Step5() {

m.v = m.y

m.t1 = m.t

m.Step6()

return

}

func (m RungeKuttaMethod) Step6() {

m.k1 = f(m.t, m.y)

m.Step7()

return

}

func (m RungeKuttaMethod) Step7() {

m.k2 = f(m.t+1\*m.tau, m.y+m.tau\*1\*m.k1)

m.k3 = f(m.t+0.5\*m.tau, m.y+m.tau\*0.25\*m.k1+m.tau\*0.25\*m.k2)

m.w = m.y + m.tau\*(0.5\*m.k1+0.5\*m.k2+0\*m.k3)

m.y = m.y + m.tau\*(1.0/6.0\*m.k1+1.0/6.0\*m.k2+4.0/6.0\*m.k3)

m.Step8()

return

}

func (m RungeKuttaMethod) Step8() {

m.E = math.Abs(m.y-m.w) / math.Max(1.0, math.Abs(m.y))

m.tauH = m.tau \* math.Min(5.0, math.Max(0.1, 0.9\*math.Pow(eps/m.E, 1.0/(1.0+8.0))))

m.Step9()

return

}

func (m RungeKuttaMethod) Step9() {

if m.E <= eps {

if m.tau > 0 {

m.t = m.t + m.tau

} else {

m.Step10()

return

}

un := u(m.t)

m.tau = m.tauH

println("-----------------------------------")

println("t = " + strconv.FormatFloat(m.t, 'f', 20, 64))

println("y = " + strconv.FormatFloat(m.y, 'f', 20, 64))

println("u(t) = " + strconv.FormatFloat(un, 'f', 20, 64))

println("|y - u(t)| = " + strconv.FormatFloat(math.Abs(m.y-un), 'f', 20, 64))

if m.eMax < math.Abs(m.y-un) {

m.eMax = math.Abs(m.y - un)

}

m.Step3()

return

}

m.y = m.v

m.t = m.t1

m.tau = m.tauH

m.Step7()

return

}

func (m RungeKuttaMethod) Step10() {

println("Maximal error = " + strconv.FormatFloat(m.eMax, 'f', 8, 64))

return

}

**Результати виконання**

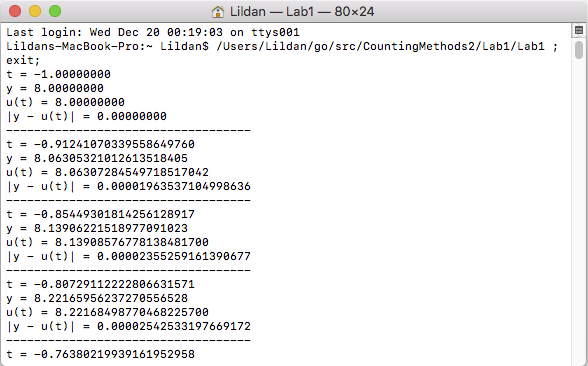
****

Рис Початок виводу

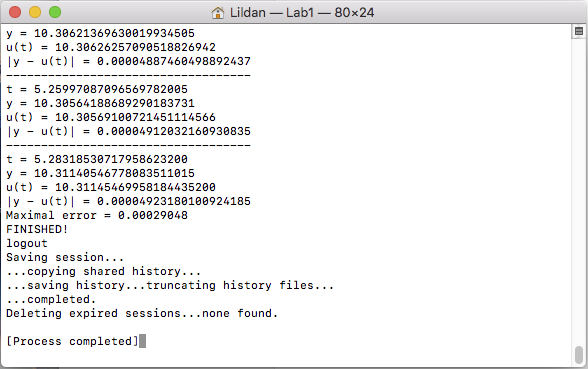


Рис 2 Кінець виводу

**Висновок**

Під час виконання лабораторної роботи було набуто навички аналітичного розв'язування задачі Коші у середовищі Maple.

Також було написано програму для чисельного розв’язування задачі Коші вкладеним методом Рунге-Кутта 3-го порядку. Для реалізації програми використано мову програмування Go.