Міністерство науки і освіти України

Національний університет “Львівська політехніка”

Інститут прикладної математики та фундаментальних наук

Кафедра прикладної математики

Звіт

про виконання лабораторної роботи №2

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

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

Львів 2017

**Мета:** навчитися чисельно знаходити розв’язок задачі Коші для системи рівнянь вкладеним методом Рунге-Кутта 3-го порядку.

**Постановка задачі**

Використовуючи мову програмування Go, написати та відлагодити програму чисельного розв’язування задачі Коші для систем ЗДР вкладеним методом Рунге-Кутта 3-го порядку при заданій точності .

|  |  |  |
| --- | --- | --- |
| Варіант 14 |  |  |

**Алгоритм чисельного розв’язування задачі Коші для систем ЗДР вкладеними**

**методами Рунге-Кутта**

1. **Ввести значення** , ,  (*N* – розмірність системи рівнянь), , , .

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** ; **вивести значення** , ;

; **go to 3**

**end;**

**else**

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

10. **End**.

**Програмний код**

package main

import ("math"

"strconv")

const t0 = 0.0

const T = 1

const eps = 0.0001

const epsM = 0.0001

const tau0 = 0.5

const y10 = 1.0

const y20 = 0.5

func y1d(t float64, y1 float64, y2 float64) float64 {

    return t + y1\*y2

}

func y2d(t float64, y1 float64, y2 float64) float64 {

    return t - y1\*y1

}

func main() {

    method := SystemRungeKuttaMethod{}

    method.Start(t0, T, eps, epsM, tau0, y10, y20)

    println("FINISHED!")

}

type SystemRungeKuttaMethod struct {

    t    float64

    T    float64

    tau  float64

    eps  float64

    epsM float64

    y1 float64

    y2 float64

    v1 float64

    v2 float64

    t1 float64

    k11, k12, k13 float64

    k21, k22, k23 float64

    w1, w2 float64

    E    float64

    tauH float64

}

func (m SystemRungeKuttaMethod) Start(t0 float64, T float64, eps float64, epsM float64, tau float64, y10 float64, y20 float64) {

    m.t = t0

    m.T = T

    m.tau = tau

    m.eps = eps

    m.epsM = epsM

    m.y1 = y10

    m.y2 = y20

    m.Step2()

    return

}

func (m SystemRungeKuttaMethod) Step2() {

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

    println("y1 = " + strconv.FormatFloat(m.y1, 'f', 8, 64))

    println("y2 = " + strconv.FormatFloat(m.y2, 'f', 8, 64))

    m.Step3()

    return

}

func (m SystemRungeKuttaMethod) Step3() {

    if math.Abs(m.T-m.t) < m.epsM {

        m.Step10()

        return

    }

    m.Step4()

    return

}

func (m SystemRungeKuttaMethod) Step4() {

    if m.t+m.tau > m.T {

        m.tau = m.T - m.t

    }

    m.Step5()

    return

}

func (m SystemRungeKuttaMethod) Step5() {

    m.v1 = m.y1

    m.v2 = m.y2

    m.t1 = m.t

    m.Step6()

    return

}

func (m SystemRungeKuttaMethod) Step6() {

    m.k11 = y1d(m.t, m.y1, m.y2)

    m.k21 = y2d(m.t, m.y1, m.y2)

    m.Step7()

    return

}

func (m SystemRungeKuttaMethod) Step7() {

    m.k12 = y1d(m.t+1\*m.tau, m.y1+1\*m.tau\*m.k11, m.y2+1\*m.tau\*m.k11)

    m.k22 = y1d(m.t+1\*m.tau, m.y1+1\*m.tau\*m.k21, m.y2+1\*m.tau\*m.k21)

    m.k13 = y1d(m.t+0.5\*m.tau, m.y1+0.25\*m.tau\*m.k11+0.25\*m.tau\*m.k12, m.y2+0.25\*m.tau\*m.k11+0.25\*m.tau\*m.k12)

    m.k23 = y1d(m.t+0.5\*m.tau, m.y1+0.25\*m.tau\*m.k21+0.25\*m.tau\*m.k22, m.y2+0.25\*m.tau\*m.k21+0.25\*m.tau\*m.k22)

    m.w1 = m.y1 + m.tau\*(0.5\*m.k11+0.5\*m.k12+0\*m.k13)

    m.w2 = m.y2 + m.tau\*(0.5\*m.k21+0.5\*m.k22+0\*m.k23)

    m.y1 = m.y1 + m.tau\*(1.0/6.0\*m.k11+1.0/6.0\*m.k12+4.0/6.0\*m.k13)

    m.y2 = m.y2 + m.tau\*(1.0/6.0\*m.k21+1.0/6.0\*m.k22+4.0/6.0\*m.k23)

    m.Step8()

    return

}

func (m SystemRungeKuttaMethod) Step8() {

    m.E = math.Max(math.Abs(m.y1-m.w1), math.Abs(m.y2-m.w2)) /

        (math.Max(1, math.Max(m.y1, m.y2)))

    m.tauH = m.tau \* math.Min(5, math.Max(0.1, 0.9\*math.Pow(m.eps/m.E, 1.0/(8+1))))

    m.Step9()

    return

}

func (m SystemRungeKuttaMethod) Step9() {

    if m.E <= m.eps {

        m.t = m.t + m.tau

        m.tau = m.tauH

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

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

        println("y1 = " + strconv.FormatFloat(m.y1, 'f', 8, 64))

        println("y2 = " + strconv.FormatFloat(m.y2, 'f', 8, 64))

        m.Step3()

        return

    } else {

        m.y1 = m.v1

        m.y2 = m.v2

        m.t = m.t1

        m.tau = m.tauH

        m.Step7()

        return

    }

}

func (m SystemRungeKuttaMethod) Step10() {

    return

}

**Результати виконання**

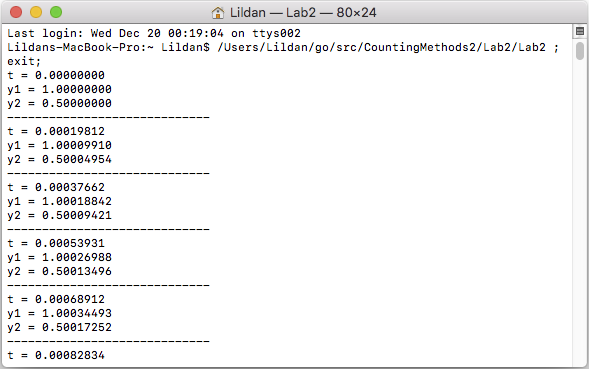


Рис 1 Початок виводу

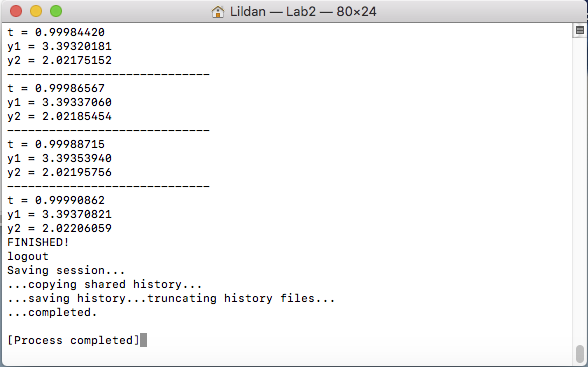


Рис 2 кінець виводу

**Висновок**

Під час виконання лабораторної роботи було написано програму для чисельного розв’язування задачі Коші систем ЗДР вкладеним методом Рунге-Кутта 3-го порядку. Для реалізації програми використано мову програмування Go.