**Лабораторна робота №7**

**ЧИСЕЛЬНЕ РІШЕННЯ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ**

Мета: набуття теоретичних знань та практичних навичок чисельного рішення звичайних ДР першого порядку

Завдання:

1. Вирішити диференціальне рівняння чисельними методами (Таблиця 2):

• Методом Ейлера;

• Методом Рунге-Кутта;

2 Побудувати точкові графіки отриманих функцій.

Короткі теоретичні відомості

Розв’язування багатьох фізичних задач описується диференціальними рівняннями, тобто рівняннями, які містять невідому функцію однієї або кількох змінних та її похідні. Порядком диференціального рівняння називають порядок старшої похідної, яка входить в рівняння.

Хай дано диференціальне рівняння першого порядку

(7.1)

Як бачимо, функція y залежить тільки від однієї змінної, тому це диференціальне рівняння називається звичайним.

Рішенням диференціального рівняння є функція *y*(*x*). Кожне диференціальне рівняння має нескінченну кількість рішень. Для виділення із усієї кількості рішень одного конкретного рішення, необхідно задати додаткові умови, кількість яких дорівнює порядку рівняння. Для рівняння першого порядку достатньо вимагати, щоб шукана функція в якійсь точці *x*0 приймала задане значення 

Коли усі додаткові умови задаються при одному значенні незалежної змінної *x*0, то кажуть, що має місто задача Коші (задача з початковими умовами). Помітимо, що для рівнянь першого порядку задача Коші отримується автоматично, оскільки маємо одну додаткову умову.

Для рівнянь другого порядку додаткових умови дві, і якщо одне з них сформульоване в точці *x*0, а друге в точці *x*1, то це буде не задача Коші, а крайова задача.

Сформулюємо задачу Коші для звичайного диференціального рівняння першого порядку: на відрізку [*a*, *b*] знайти функцію , яка задовольняє диференціальному рівнянню (7.1) і початковій умові , *х*0∈[*a,* *b*]*.*

Припустимо, що рішення диференціального рівняння існує і тільки єдине. Точку *х0*будемо обирати на лівому кінці інтервалу, тобто *х*0 = *a*.

Деякі види диференціальних рівнянь мають точне аналітичне рішення. Але у більшості випадків треба використати наближені методи.

При використанні чисельних методів рішення диференціального рівняння розшукується у формі таблиці . Для цього інтервал [*a*, *b*] розділяється на *n* рівних частин, що дає множину точок *xi*:

 ,  ,

яка називається сіткою з рівномірним кроком. Рішення розшукуються у вузлах сітки, і тому його називають сіточною функцією.

Диференціальне рівняння визначає в кожній точці (*х*, *у*) області існування значення похідної *у′*, тобто тангенсу куту нахилу дотичної до графіка шуканої функції, або, як кажуть, – задає поле напрямків. Цей геометричний зміст звичайного диференціального рівняння першого порядку лежить в основі більшості наближених чисельних методів розв’язування задачі Коші.

Розглянемо деякі найбільш прості з них.

**Метод Ейлера**

Це найбільш простий метод знаходження рішення, який забезпечує порівняно низьку точність.

Якщо відоме значення функції *y*(*x*), то значення функції в точці *x+h* при достатньо малому *h* приблизно можна знайти за допомогою відомої формули:

*.*

Для геометричної інтерпретації перепишемо цю залежність в виді

,(7.2)

де – кутовий коефіцієнт нахилу дотичної до графіка, який визначається з (7.1).

Рис.7.1

Переписав цю залежність в сіточній формі, отримаємо рекурентну формулу для визначення функції в вузлах сітки (рис. 7.1):

, (7.3)

де , , .

**Метод Рунге-Кутту**

Припустимо, що наближене значення *yi* в точці *xi* вже відомо. Для знаходження *yi+*1 = *y*(*xi+*1) (рис. 7.3), спочатку використовуємо схему Ейлера (визначимо значення *y*в точці *xi*+*h/*2*,* яке позначимо як *yi+*1/2):

,

де *,* а потім скористуємося добре відомим рівнянням

,

де .

Рис.7.3



Подивіться на рис. 7.3, що ми зробили. З точки  “стріля-ємо” з кутовим коефіцієнтом *k*1*,* отримуємо точку . Потім знову “стріляємо” із точки , але вже з уточненим коефіцієнтом, отримуємо точку .

Цей метод називають методом Рунге-Кутта. Процедуру реалізації цього методу треба скласти самостійно.

**Чотириточечна схема методу Рунге-Кутту**

Метод Рунге-Кутту має добру точність. Але на практиці використовують більш точну чотириточечну схему методу Рунге-Кутту.

Хай як i раніше , але значення кутового коефіцієнту *k* знайдемо

Рис.7.4.

як результат усереднення чотирьох кутових коефіцієнтів, які обчислюються в точках, вказаних на рис.7.4. Геометрична інтерпретація аналогічна попередньому методу:

, ,

, .

Тоді

.

**Завдання**

Використовуючи метод Ейлера та чотириточечну схему методу Рунге-Кутта, скласти таблицю наближених значень інтегралу диференціального рівняння *y*′=*f*(*x*,*y*) при початкових умовах *y*(*x*0)=*y*0 на інтервалі [*a, b*] з кроком *h*=0,1.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Хід роботи**

1. Текст програми

import tmpl from 'lodash/template';

import Chart from 'chart.js';

import './differential.sass';

import template from './differential.template.html';

import Model from './model.js';

const model = new Model();

const eyler = model.eylerDifferential().map(item => {

return { x: +item.x.toFixed(2), y: +item.y.toFixed(2)}

});

const eylerXs = eyler.map(item => item.x);

const $contentSlot = document.querySelector('.content-slot');

$contentSlot.innerHTML = tmpl(template)(model);

const $canvas = document.querySelector('.canvas');

var ctx = $canvas.getContext('2d');

var differentialChart = new Chart(ctx, {

type: 'scatter',

data: {

labels: eylerXs.map(x => `x: ${x}`),

datasets: [{

label: 'Eyler Method',

data: eyler,

backgroundColor: 'transparent',

borderColor: 'rgb(0,110,135)',

}],

},

});

import './differential.sass';

export default class Model {

a = 0.6;

b = 2.6;

h = 0.2;

first = 3.4;

eylerDifferential() {

const nodes = this.getNodes();

const res = [];

let prev = this.first;

for (let i = 1; i < nodes.length; i++) {

const next = prev + this.h \* this.primeF(nodes[i - 1], prev);

res.push({

x: nodes[i - 1],

y: next,

});

prev = next;

}

return res;

}

primeF(x, y) {

return 4.1 \* x - y \* y + 0.6;

}

getNodes() {

const steps = (this.b - this.a) / this.h + 1;

const nodes = new Array(steps);

const x0 = this.a;

for (let i = 0; i < steps; i++) {

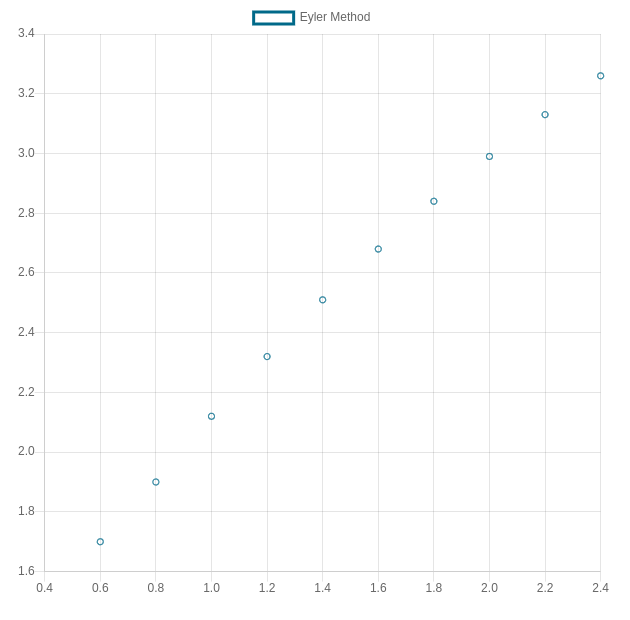
nodes[i] = x0 + i \* this.h;

}

return nodes;

}

}

1. Результат виконання:

**Висновок:** на цій лабораторній роботі набув **теоретичних знань та практичних навичок чисельного рішення звичайних ДР першого порядку**