**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут ім. Ігоря Сікорського»**

Інститут Прикладного системного аналізу

(назва факультету, інституту)

Кафедра Системного проектування

(назва кафедри)

Пояснювальна записка

до курсової роботи на тему:

“Паралельне розв’язання диференційного рівняння у частинних похідних за допомогою методів кінцевих різниць”

**Студент групи** ДА-62 Слєпцова Ольга Олегівна

(шифр групи) (прізвище, ім’я, по батькові) (підпис)

**Керівник проекту** к.т.н., викл. Яременко В.С.

(вчені ступінь та звання, прізвище, ініціали) (підпис)

Керівник: Виконавець:

Яременко В.С. ст. Слєпцова О.О.,

гр. ДА-62

Допущений до захисту Зал. книжка

№ ДА-6228

Захищено із оцінкою

Київ – 2019

Форма No У-6.01   
Затв. наказом УРСР

ННК «ІПСА» НТУУ “Київський Політехнічний Інститут ім. Ігоря Сікорського”

(назва вищого навчального закладу)

Кафедра Системного проектування

Дисципліна Паралельні обчислення

Спеціальність 6.050101

Курс 3 Група ДА-62 Семестр 6

**ЗАВДАННЯ**

**на курсову роботу студента**

Слєпцової Ольги Олегівни

(прізвище, ім’я, по батькові)

1. Тема курсової роботи

**“Паралельне розв’язання диференційного рівняння у частинних похідних за допомогою методів кінцевих різниць”**

1. Строк здачі студентом закінченого проекту (роботи) до 20.05.2019 р.

Студент Слєпцова О.О.

(підпис)

Керівник ас. Яременко В.С. (підпис)

“20” травня 2019 р.

# **ЗАВДАННЯ**

Задача № 22.

Знайти розв’язок рівняння:

, де – деякі константи.

Точний розв’язок:

, де , – деякі константи, .

# **ТЕОРЕТИЧНІ ВІДОМОСТІ**

## МЕТОД СКІНЧЕННИХ РІЗНИЦЬ

Розглянемо розв’язання лінійного одновимірного рівняння теплопровідності:

з крайовими

та початковою

умовами.

Одним зі способів чисельного розв’язання такого типу рівнянь є апроксимація всіх похідних рівняння їхніми скінченими різницями. Розділимо ділянку простору, використовуючи сітку , а часовий інтервал – сітку . Нехай крок сітки буде рівномірний і в часі, і в просторі. Тоді різницю між двома послідовними точками в просторі позначимо , а в часі – , а чисельною апроксимацією значення буде .

### ЯВНИЙ МЕТОД

Використовуючи праву різницю по часу та центральну різницю по простору для апроксимації похідних, отримаємо рекурентне рівняння:

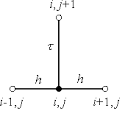


Рис. 1 – Шаблон найбільш поширеного явного методу

Тоді для значення отримуємо:

, де

Таким чином, маючи останнє рекурентне співвідношення та знаючи значення функції в момент часу , можна отримати відповідні значення в момент часу .

Відомо, що явний метод чисельно стійкий і збіжний при , а його похибка пропорційна часовому кроку та квадрату просторового кроку:

# **РОЗВ’ЯЗОК**

Виберемо наступні значення сталих коефіцієнтів:

, ,

Тоді рівняння набуває наступного вигляду:

а точний розв’язок:

де

Знайдемо початкову умову:

та граничні умови:

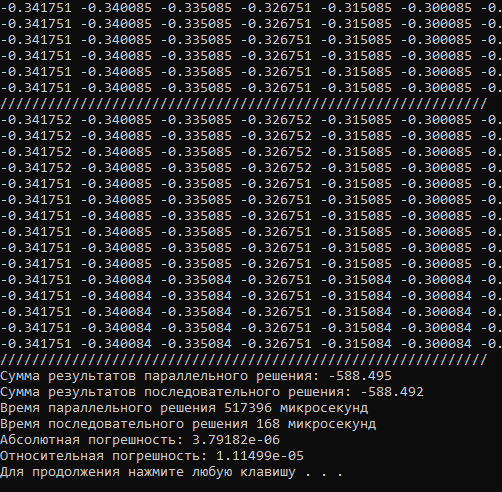
Застосуємо до рівняння неявну різницеву схему:

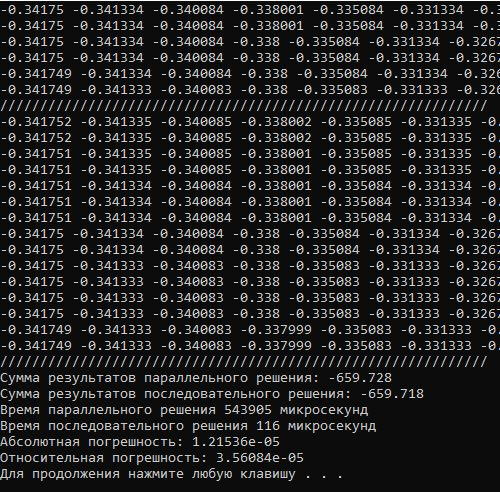
Тоді різницеве рівняння матиме наступний вигляд:

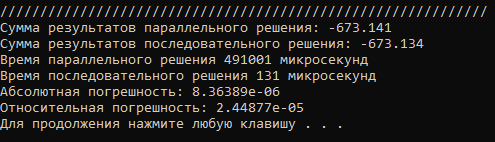
звідки

Отримаємо формулу для кожного з (n-1) внутрішніх вузлів поточного часового шару.

# **РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ**





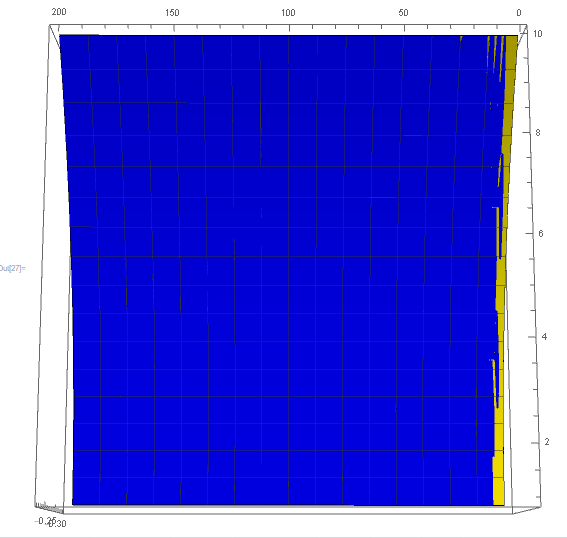


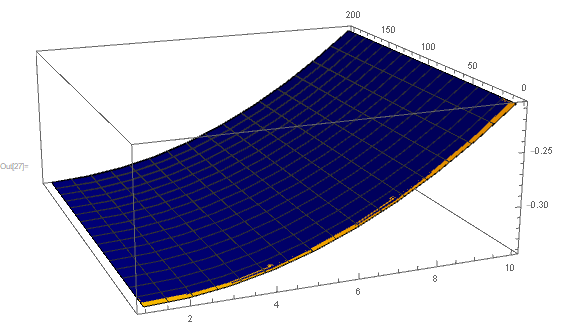
|  |  |  |
| --- | --- | --- |
|  | **Послідовний** | **Паралельний** |
|  | 168 мкс | 517396 мкс |

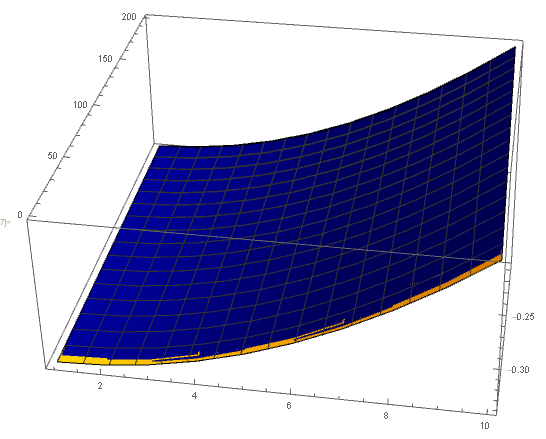
Таблиця 1 – Час виконання у мікросекундах

ПОВЕРХНЯ ЗА ЯВНИМ МЕТОДОМ

На рисунку зображено поверхню, знайдену явним кінцево-різницевим методом і поверхню точного рішення із кроками і . Поверхні мало відрізняються одна від одної, також можна спостерігати явище накопичення похибки. Графік було побудовано за допомогою пакету Wolfram Mathematica.







# **ВИСНОВОК**

В даній роботі було реалізовано паралельне рішення диференційного рівняння в частинних похідних.

Також було знайдено похибки для явного методу при різних кроках.

Було порівняно час на вирішення задачі послідовно і паралельно. Час на паралельну програму виявився більшим, ніж при послідовному пошуку рішень. На мою думку це через те, що в явному методі мало незалежних розрахунків. А також причиною є те, що для створення потоків потрібен час.

# **ДОДАТОК 1. ЛІСТИНГ ПРОГРАМИ**

// ConsoleApplication1.cpp: определяет точку входа для консольного приложения.

//

#include "stdafx.h"

#include <iostream>

#include <fstream>

#include <thread>

#include <ctime>

#include <vector>

#include <string>

typedef std::chrono::high\_resolution\_clock Clock;

using namespace std;

int A = 6;

int B = 4;

double a = 0.00008;

double b = 0.00008;

double tau = 0.0011;

double h = 0.033;

double getExact(double x, double t)

{

double F = A - 6 \* a\*t;

double result = (x\*x) / F + 6 / sqrt((F\*F\*F)) - F / 8;

return result;

}

double getApproximate(double w\_i, double w\_i\_plus, double w\_i\_minus)

{

double sigma = tau / (h\*h);

double result = w\_i + a \* sigma\*(0.25\*pow((w\_i\_plus - w\_i\_minus), 2) + w\_i \* (w\_i\_plus - 2 \* w\_i + w\_i\_minus)) + b \* tau;

return result;

};

void countValue(double x, double t, double\*\* Arr, int from, int to, int column, int i)

{

for (int j = from; j < to; j++)

{

if (i == 0 || j == 0 || j == (column - 1)) {

Arr[i][j] = getExact(x, t);

}

else {

Arr[i][j] = getApproximate(Arr[i - 1][j], Arr[i - 1][j + 1], Arr[i - 1][j - 1]);

}

x += h;

}

}

void mistake(double \*\*array1, double \*\*array2, int row, int column)

{

double absolute = -1;

int t\_i;

int t\_j;

double relative;

for (int i = 0; i < row; i++)

{

for (int j = 0; j < column; j++)

{

double temp\_val = abs(array1[i][j] - array2[i][j]);

if (temp\_val > absolute) {

absolute = temp\_val;

t\_i = i;

t\_j = j;

}

}

}

cout << "Абсолютная погрешность: " << absolute << endl;

relative = abs(absolute / array2[t\_i][t\_j]);

cout << "Относительная погрешность: " << relative << endl;

}

void printArray(double \*\*array)

{

for (int i = 0; i < 15; i++)

{

for (int j = 0; j < 15; j++)

{

cout << array[i][j] << " ";

}

cout << endl;

}

cout << "/////////////////////////////////////////////////////////////" << endl;

}

double writeArrayToFile(double \*\*array, string filename, int row, int column)

{

ofstream out;

double serialSum = 0;

out.open(filename, ios\_base::trunc);

out << "{";

for (int i = 0; i < row; i++)

{

out << "{";

for (int j = 0; j < column; j++)

{

serialSum += array[i][j];

if (j == column - 1) {

out << array[i][j];

}

else {

out << array[i][j] << ",";

}

}

out << "},";

}

out << "}";

out.close();

return serialSum;

}

int main()

{

setlocale(LC\_ALL, "Russian");

int countThread = 4;

const int row = 200, column = 10;

double \*\*Array1 = new double \*[row];

double \*\*Array2 = new double \*[row];

for (int i = 0; i < row; i++)

{

Array1[i] = new double[column];

Array2[i] = new double[column];

}

double x = 0;

double t = 0;

int from = 0;

int to = 0;

int temp = (column / countThread);

auto start1 = std::chrono::steady\_clock::now();

for (int i = 0; i < row; i++)

{

x = 0;

from = 0;

vector < thread> threadVector;

for (int j = 0; j < countThread; j++)

{

if (j == (countThread - 1))

{

to = column;

}

else

{

to = temp \* (j + 1);

}

double temp\_x = (x + h) \* from;

threadVector.emplace\_back(thread(countValue, temp\_x, t, Array1, from, to, column, i));

from = to;

}

for (int z = 0; z < threadVector.size(); z++)

{

threadVector[z].join();

}

t += tau;

}

auto end1 = std::chrono::steady\_clock::now();

printArray(Array1);

double parallelSum = writeArrayToFile(Array1, "array1.txt", row, column);

t = 0;

auto start2 = std::chrono::steady\_clock::now();

for (int i = 0; i < row; i++)

{

x = 0;

for (int j = 0; j < column; j++)

{

Array2[i][j] = getExact(x, t);

x += h;

}

t += tau;

}

auto end2 = std::chrono::steady\_clock::now();

printArray(Array2);

double serialSum = writeArrayToFile(Array2, "array2.txt", row, column);

cout << "Сумма результатов параллельного решения: " << parallelSum << endl;

cout << "Сумма результатов последовательного решения: " << serialSum << endl;

cout << "Время параллельного решения " <<

chrono::duration\_cast<std::chrono::microseconds>(end1 - start1).count() << " микросекунд" << endl;

cout << "Время последовательного решения " <<

chrono::duration\_cast<std::chrono::microseconds>(end2 - start2).count() << " микросекунд" << endl;

mistake(Array2, Array1, row, column);

for (int i = 0; i < row; i++)

delete[]Array1[i];

for (int i = 0; i < row; i++)

delete[]Array2[i];

system("pause");

return 0;

}

Посилання на GitHub:

<https://github.com/Lacertian/CourseWorkParallel>