Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій



**ЗВІТ**

**Про виконання лабораторної роботи № 6**

«Розв'язування перевизначених систем лінійних алгебраїчних рівнянь»

**з дисципліни «Чисельні методи»**

**Лектор:**

доцент кафедри ПЗ

Мельник Н.Б.

**Виконав:**

студ. групи ПЗ-15

Марущак А. С.

**Прийняв:**

асистент кафедри ПЗ

Гарматій Г.Ю

«\_\_\_» \_\_\_\_\_\_ 2022 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2022

**Тема роботи:** Розв'язування перевизначених систем лінійних алгебраїчних рівнянь

**Мета роботи:** ознайомлення на практиці з методами розв’язування перевизначених систем лінійних алгебраїчних рівнянь.

**Теоретичні відомості**

**Метод найменших квадратів для розв’язування перевизначених СЛАР**

Розглянемо систему лінійних алгебраїчних рівнянь, у якій кількість рівнянь є більшою за кількість невідомих

У загальному випадку ця система рівнянь є несумісною. Якщо із даної системи вибрати m рівнянь та розв’язати їх, то отриманий розв’язок не буде задовольняти всі рівняння цієї системи. Тому вчинимо інакше: знайдемо розв’язок системи наближено, але щоб він задовольняв усі рівняння системи, а саме

Розв’язок системи будемо знаходити з використанням умови мінімізації суми квадратів відхилень, тобто з умов мінімізації функції

і вимагатимемо, щоб виконувалась умова

Проведемо деякі перетворення над системою, використовуючи цю умову. Розглянемо функцію

Необхідною умовою мінімуму функції від багатьох змінних є рівність нулеві її частинних похідних. Використаємо цей факт і продиференціюємо цю функцію за змінними (i =1,m). У результаті отримаємо

Прирівнявши вирази до нуля, отримаємо нормальну систему рівнянь

в якій кількість рівнянь системи дорівнює кількості невідомих. Нормальні системи лінійних алгебраїчних рівнянь характеризуються тим, що матриці їх коефіцієнтів завжди є симетричними, а діагональні елементи - додатніми. Цю систему розв'язують довільними прямими або ітераційними методами. Якщо матриця коефіцієнтів системи рівнянь є додатньо визначеною (визначник матриці є більшим за нуль), то рекомендують для її розв’язування використовувати метод квадратного кореня. Запишемо систему лінійних алгебраїчних рівнянь у матричному вигляді

А=

де A - матриця коефіцієнтів системи розмірності m×n, X – матриця-стовпець невідомих розмірності m ×1, B - матриця-стовпець вільних членів системи розмірності m×1.

Це матричне рівняння помножимо на транспоновану матрицю до матриці A. У результаті отримаємо матричне рівняння

де N – матриця коефіцієнтів нормальної системи

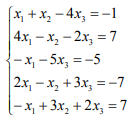
C -стовпець вільних членів

Розв’язавши нормальну систему лінійних алгебраїчних рівнянь, отримаємо її точний розв’язок (якщо використано прямі методи) або наближений розв’язок (якщо використано ітераційні методи). Отриманий розв’язок буде наближеним для СЛАР.

**Індивідуальне завдання**

1. Ознайомитися з теоретичним матеріалом.
2. Скласти програму розв’язування перевизначеної системи лінійних алгебраїчних рівнянь методом найменших квадратів. Отриману відповідну систему розв’язати методом квадратного кореня.

**Варіант завдання**



**Хід роботи**

Проведемо деякі обчилення вручну, щоб потім мати змогу порівняти

результат виконання програми з дійсним.

Випишемо матрицю коефіцієнтів системи А, і вектор-стовпець вільних членів В:

Тоді наше рівняння можна записати у вигляді

,

де – вектор-стовпець невідомих. Знайдемо матрицю коефіцієнтів відповідної нормальної системи рівнянь

та матрицю-стовпець вільних членів

Тоді систему можна записати у вигляді

Оскільки , то подану систему розв’яжемо методом квадратних коренів.

Нехай , тоді

Перевіримо:

;

Отже, розклад було здійснено правильно.

Тоді подальший розв’язок аналогічний до розв’язання системи методом LU-розкладу, де замість – матриця . Маємо , тоді

або

Звідки дістаємо, що . Тоді, з рівняння , яке записується у вигляді

Маємо, що .

Наступна мета – реалізувати подані методи програмно і переконатися в достовірності результату.

Для подальшої роботи я буду використовувати структуру даних Matrix, яка відповідає матриці заданого розміру, як зрозуміло з назви. Серед її функціональних можливостей – множення та додавання матриць, пошук транспонованої, та детермінанту, розклад розширеної матриці системи на матрицю коефіцієнтів та вектор стовпець вільних членів. Покажемо це у коді:

public static Matrix operator\*(Matrix a, Matrix b)

{

if (a.ColumnsCount != b.RowsCount)

throw new ArgumentException("Cannot multiply matrices of these sizes.");

Matrix result = new(a.RowsCount, b.ColumnsCount);

for (int i = 0; i < result.RowsCount; i++)

for (int j = 0; j < result.ColumnsCount; j++)

result[i, j] = a.Row(i) \* b.Column(j);

return result;

}

public static Matrix operator+(Matrix a, Matrix b)

{

if (a.ColumnsCount != b.ColumnsCount || a.RowsCount != b.RowsCount)

throw new ArgumentException("Cannot add matrices of different sizes.");

Matrix result = new(a.RowsCount, b.ColumnsCount);

for (int i = 0; i < result.RowsCount; i++)

for (int j = 0; j < result.ColumnsCount; j++)

result[i, j] = a[i, j] + b[i, j];

return result;

}

public Matrix Transponed()

{

Matrix result = new(ColumnsCount, RowsCount);

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

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

result[j, i] = this[i, j];

return result;

}

public void Deconstruct(out Matrix A, out Matrix B)

{

A = new Matrix(RowsCount, ColumnsCount - 1);

B = new Matrix(RowsCount, 1);

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

{

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

{

if (j == ColumnsCount - 1)

B[i, 0] = this[i, j];

else A[i, j] = this[i, j];

}

}

}

public double Determinant()

{

if (ColumnsCount != RowsCount)

throw new InvalidOperationException("Cannot find determinant for non-square matrices.");

if (ColumnsCount == 1)

return this[0, 0];

double result = 0;

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

{

result += Math.Pow(-1, i) \* this[0, i] \* this.WithoutRow(0).WithoutColumn(i).Determinant();

}

return result;

}

І основна програма буде виглядати наступним чином:

using MatrixLib;

Matrix HoletskyMethod(Matrix A, Matrix B)

{

if(A.Determinant() <= 0)

throw new ArgumentException(nameof(A));

Matrix X = new(B.RowsCount, 1), Y = new(B.RowsCount, 1);

Matrix L = new(A.RowsCount, A.ColumnsCount);

// Шукаємо матрицю L

for (int i = 0; i < L.RowsCount; i++)

{

for (int j = 0; j < L.ColumnsCount; j++)

{

if (i == j)

{

double s = 0;

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

{

s += L[i, k] \* L[i, k];

}

L[i, j] = Math.Sqrt(A[i, j] - s);

}

else if (j < i)

{

double s = 0;

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

{

s += L[i, k] \* L[j, k];

}

L[i, j] = (1 / L[j, j]) \* (A[i, j] - s);

}

}

}

Console.WriteLine($"Отримуємо матрицю L: \n{L}");

Console.WriteLine($"Добуток L \* L\_T: \n{L \* L.Transponed()}");

// Шукаємо вектор-стовпець Y

for (int i = 0; i < L.RowsCount; i++)

{

double s = 0;

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

{

s += L[i, j] \* Y[j,0];

}

Y[i, 0] = (B[i, 0] - s) / L[i, i];

}

// Шукаємо вектор-стовпець X

for (int i = L.RowsCount - 1; i >= 0; i--)

{

double s = 0;

for (int j = i + 1; j < L.ColumnsCount; j++)

{

s += L.Transponed()[i, j] \* X[j, 0];

}

X[i, 0] = (Y[i, 0] - s) / L.Transponed()[i, i];

}

//Повертаємо результат

return X;

}

//Зчитуємо матрицю з файлу

Matrix systemMatrix = Matrix.FromFile("input.txt");

//Розкладаємо розширену матрицю на матрицю коефіцієнтів і стовпець вільних членів

(Matrix A, Matrix B) = systemMatrix;

//Шукаємо нормальну матрицю системи

Matrix N = A.Transponed() \* A;

Console.WriteLine($"Нормальна матриця системи: \n{N}");

//Шукаємо новий стовпець вільних членів

Matrix C = A.Transponed() \* B;

Console.WriteLine($"Новий стовпець вiльних членiв: \n{C}");

Matrix X = HoletskyMethod(N, C);

Console.WriteLine($"Вiдповiдь: \n{X}");

Тоді, з умови дістанемо розширенну матрицю системи

Подамо цю матрицю на вхід програми і подивимось на результат.

**Результат** виконання програми:

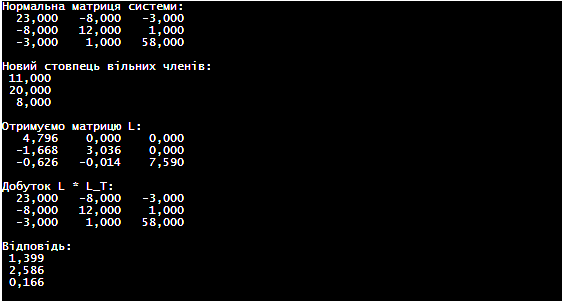


Рис 6.1 Результат виконання програми.

**Аналіз результатів:**

Як бачимо, результати обчислень на комп’ютері повністю співпали з результатами обчислень, здійснених вручну. Також, співпали матриці. Тому, можемо судити про достовірність як і ручних обчислень, так і програми, складеної на основі вивченого матеріалу.

Так як пряма підстановка результату в початкову систему дасть мало інформації, то аналітично доведемо, що отриманий роз’язок є точкою мінімуму для функції похибки. Запишемо похибку кожного рядка:

Тоді функція похибки .

Градієнт функції – напрямок найшвидшого росту. Тоді очевидно, що для мінімізації функції треба рухатись в напрямку, протилежному градієнту в точці, тобто за напрямком вектора . І запишемо наступний алгоритм:

1. Вибрати довільну точку A.
2. Знайти градієнт функції в точці А.
3. Перемістити точку А за вектором , де – коефіцієнт, який ми обираємо самі і який показує швидкість наближення.
4. Повторювати кроки 2,3 допоки відстань між попереднім і теперішнім положенням точки не стане менше за точність .

Знайдемо частинні похідні:

І тоді на основі цього алгоритму напишемо коротку програму, вибравши за початкове наближення точку з координатами (0, 0, 0):

double f(double x, double y, double z) => 23\*x\*x - 16\*x\*y - 6\*x\*z - 22\*x + 12\*y\*y + 2\*y\*z - 40\*y + 58\*z\*z - 16\*z + 173;

double dfdx(double x, double y, double z) => 46 \* x - 16 \* y - 6 \* z - 22;

double dfdy(double x, double y, double z) => -16\*x + 24\*y + 2\*z - 40;

double dfdz(double x, double y, double z) => -6 \* x + 2 \* y + 116 \* z - 16;

Vector grad(double x, double y, double z) => new Vector(

dfdx(x, y, z),

dfdy(x, y, z),

dfdz(x, y, z)

);

const double step = 0.01;

Point A = new(0, 0, 0), A\_prev;

do

{

A\_prev = (A.Clone() as Point)!;

double x = A.X;

double y = A.Y;

double z = A.Z;

A = A\_prev.MovedBy(-grad(x,y,z) \* step);

Console.WriteLine(A);

} while (A.DistanceTo(A\_prev) > 5e-5);

І тоді отримуємо наступну послідовність наближень:

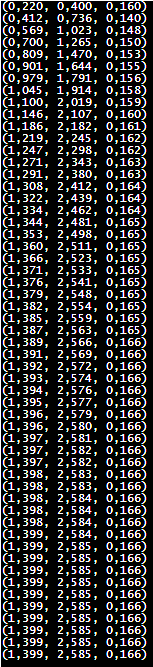


Рис 6.2 Результат виконання градієнтного спуску.

Як бачимо, методом градієтного спуску ми втрапили в ту саму точку, що і розв’язуючи нормальну систему. Тому це точка локального мінімуму. І щоб завершити доведення, треба довести, що цей мінімум є і глобальним.

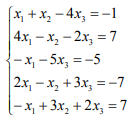
Матриця Гесе цієї функції: =

Оскільки , то за критерієм Сильвестра ця матриця – додатньо визначена. А це означає, що функція опукла, що геометрично неможливо уявити у випадку функції 3 аргументів, проте, формально кажучи . Це і означає, що функція має один мінімум і він, до того ж, глобальний. Отже, наше доведення завершене і результат підтверджений.

**Висновок:**

Виконуючи цю лабораторну роботу ми ознайомились на практиці з методами розв’язування перевизначених систем лінійних алгебраїчних рівнянь.

За допомогою цих знань ми реалізували програму, що знайшла оптимальний розв’язок наступної системи:



Перед цим, ми провели деякі обчислення вручну, і проміжні результати обчислень повністю аналогічні до результату виконання програми.

Ми отримали вектор-стовпець дійсних чисел, що відображає найоптимальніший розв' язок цієї системи:

і аналітично довели, що цей результат є вірним.