**Лабораторная работа №1**

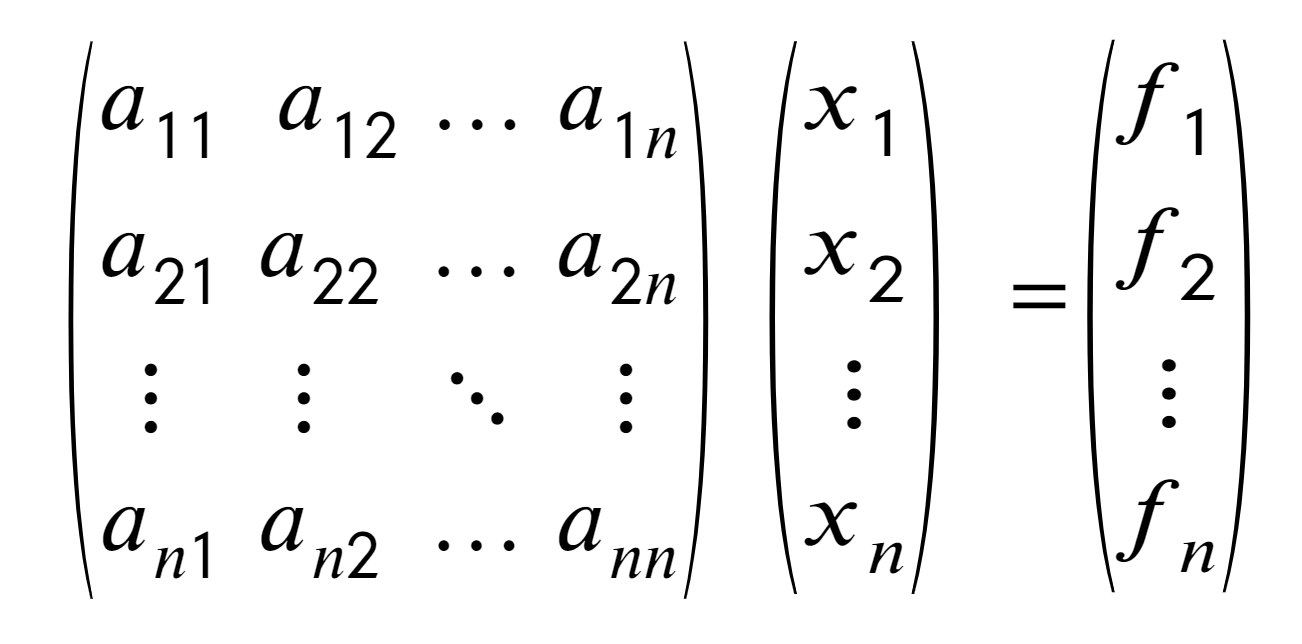
«Метод Гаусса»

Выполнил Бахар Артём,2 курс 4 группа

**1) Постановка задачи**

Необходимо найти решение системы линейных алгебраических

уравнений вида Ах=f, где A- квадратная матрица n-ого порядка, х и f – столбцы размеров n×1.



Предполагается, что det A=|A|≠0. Тогда решение системы существует

и оно единственно.

Нам необходимо найти:

a) Приближенное решение x\*;

b) Максимум-норма невязки ||Ax\* - f||∞;

c) Максимум-норма погрешности ||x-x\*||∞

d) Найти определитель матрицы

e) Найти обратную матрицу A-1(построчно) и матрицу A-1A(построчно)

**2) Алгоритм решения**

Решение системы линейных алгебраических уравнений будет

найдено методом Гаусса с выбором главного элемента по столбцу.

Метод Гаусса содержит две совокупности операций, условно

названных прямой ход и обратный ход.

**Прямой ход:**

Прямой ход метода Гаусса заключается в исключении элементов,

расположенных ниже элементов, соответствующих главной

диагонали матрицы А. При этом матрица А с помощью элементарных

преобразований приводится к нижне треугольной, а расширенная

матрица системы – к трапециевидной.

Выбор главного элемента по столбцу заключается в том, чтобы

на k-ом шаге переставить строки матрицы так, чтобы наибольший по

модулю элемент при xk попал на главную диагональ, а затем выбрать

его в качестве главного элемента .

Далее исключим переменную из всех уравнений, начиная с

(k+1)-ого. Для этого вычтем получившуюся после перестановки k-ую

строку из остальных строк, домножив её на величину, равную

отношению k-ого элемента каждой из этих строк к k-ому элементу

первой строки, обнуляя тем самым столбец под ним.

**На k-ом шаге:**

**Обратный ход:**

Вычисляем ответ из последнего уравнения как

С помощью полученного значения находим *xn-1* из предпоследнего

уравнения, и так далее, находим *x1* из первого уравнения.

**Вычисление определителей и обращения:** Так как мы сводим

матрицу А к треугольному виду при помощи элементарных

преобразований, то:

*где m - кол-во перестановок строк/столбцов*

*aii - главные элементы метода Гаусса*

Нахождение обратной эквивалентно решению уравнения:

AX=E <=> X=A^(-1)

**3) Листинг программы**

*Инициализация данных*

std::cout << "Matrix A:\n\n";

double matrixA[10][10];

for (int i = 0; i < 10; i++)

{

for (int k = 0; k < 10; k++)

{

matrixA[i][k] = double(((rand() % 2000-1000)))/100;

}

}

for (int i = 0; i < 10; i++)

{

for (int k = 0; k < 10; k++)

{

std::cout << matrixA[i][k] << " ";

}

std::cout << "\n";

}

std::cout << "\n\nvector x:\n\n";

double matrixX[10];

for (int i = 0; i < 10; i++)

{

matrixX[i] = double((rand() % 2000 - 1000)) / 100;

}

for (int i = 0; i < 10; i++)

{

std::cout << matrixX[i] << "\n";

}

std::cout << "\nMatrix Ax(f):\n";

double matrixF[10];

for (int i = 0; i < 10; i++)

{

double temp = 0;

for (int k = 0; k < 10; k++)

{

temp += matrixA[i][k] \* matrixX[k];

}

matrixF[i] = temp;

}

for (int i = 0; i < 10; i++)

{

std::cout << matrixF[i] << "\n";

}

Метод Гаусса

Прямой ход:

//Gaussian method with selection of the main element by column

double matrixA2[10][10];

for (int i = 0; i < 10; i++)

{

for (int k = 0; k < 10; k++)

{

matrixA2[i][k] = matrixA[i][k]; //copy matrixA

}

}

double matrixF2[10];

for (int i = 0; i < 10; i++)

{

matrixF2[i] = matrixF[i];//copy matrixF

}

int swaps=0;//number of swaps

for (int k = 0; k < 10; k++)

{

int j = k;

for (int i = k+1; i < 10; i++)

{

double max = matrixA2[k][k];

if (abs(max) < abs(matrixA2[i][k])) { //the main of the column

max = matrixA2[i][k];

j = i;

}

}

//swap strings

if (j != k)

{

swaps++;

double temp = matrixF2[k];

matrixF2[k] = matrixF2[j];

matrixF2[j] = temp;

for (int i = k; i < 10; i++)

{

temp = matrixA2[k][i];

matrixA2[k][i] = matrixA2[j][i];

matrixA2[j][i] = temp;

}

}

for (int i = k+1; i < 10; i++)

{

double l = matrixA2[i][k] / matrixA2[k][k];

matrixF2[i] = matrixF2[i] - l \* matrixF2[k];

for (int j = k+1; j < 10; j++)

{

matrixA2[i][j] = matrixA2[i][j] - (l \* matrixA2[k][j]);

}

}

}

Обратный ход

//reverse motion

double matrixX2[10];

matrixX2[9] = matrixF2[9] / matrixA2[9][9];

for (int i = 8; i>=0 ; i--)

{

double gauss=0;

int k = 10 - i;

for (int j = 0; j < k-1; j++)

{

gauss += matrixA2[i][9-j] \* matrixX2[9-j];

}

matrixX2[i] = (matrixF2[i] - gauss) / matrixA2[i][i];

}

std::cout << "\n\n";

std::cout.setf(std::ios::fixed);

std::cout.precision(20); //0 - число символов после точки

for (int i = 0; i < 10; i++)

{

std::cout << matrixX2[i]<<" ";

}

Сравнение сгенерированного вектора f и полученного путем перемножения Ax\*

/\*a\*x`:\*/

double matrixF3[10];

for (int i = 0; i < 10; i++)

{

double temp = 0;

for (int k = 0; k < 10; k++)

{

temp += matrixA[i][k] \* matrixX2[k];

}

matrixF3[i] = temp;

}

Максимум-норма

//Ax`-f:

for (int i = 0; i < 10; i++)

{

matrixF3[i] = matrixF3[i] - matrixF[i];

}

double matrixnorm = abs(matrixF3[0]);

for (int i = 1; i < 10; i++)

{

if (matrixnorm < abs(matrixF3[i])) matrixnorm = matrixF3[i];

}

double matrixX3[10];

//x-x`

for (int i = 0; i < 10; i++)

{

matrixX3[i] = matrixX[i] - matrixX2[i];

}

double xnorm=abs(matrixX3[0]);

for (int i = 1; i < 10; i++)

{

if (xnorm < abs(matrixX3[i])) xnorm = matrixX3[i];

}

Нахождение определителя матрицы A

double detA=1;

for (int i = 0; i < 10; i++)

{

detA \*= matrixA2[i][i];

}

detA \*= pow(-1, swaps);

std::cout << "\n\n" << detA<<"\n\n";

Нахождение обратной матрицы для матрицы A

//A^-1:

double matrixAs[10][10] = { 0 };

for (int i = 0; i < 10; i++)

{

matrixAs[i][i] = 1;

}

double matrixAc[10][10];

for (int i = 0; i < 10; i++)

{

for (int k = 0; k < 10; k++)

{

matrixAc[i][k] = matrixA[i][k];

}

}

for (int j = 0; j < 9; j++)

{

for (int i = j; i < 9; i++)

{

double l = matrixA[i + 1][j] / matrixA[j][j];

for (int k = j; k < 10; k++)

{

matrixA[i + 1][k] -= l \* matrixA[j][k];

}

for (int k = 0; k < 10; k++)

{

matrixAs[i + 1][k] -= l \* matrixAs[j][k];

}

}

}

//reverse motion

for (int j = 9; j > 0; j--)

{

double l;

for (int i = j; i >0; i--)

{

l = matrixA[i - 1][j] / matrixA[j][j];

matrixA[i - 1][j] -= l \* matrixA[j][j];

for (int k = 0; k <10; k++)

{

matrixAs[i-1][k] -= l \* matrixAs[j][k];

}

}

}

for (int i = 0; i < 10; i++)

{

for (int k = 0; k < 10; k++)

{

matrixAs[i][k] /= matrixA[i][i];

}

}

std::cout << "\ninverse matrix:\n\n";

for (int i = 0; i < 10; i++)

{

for (int k = 0; k < 10; k++)

{

std::cout << matrixAs[i][k] << " ";

}

std::cout << "\n";

}

//A^-1\*A:

std::cout << "\ninverse matrix\*matrix:\n\n";

double matrixAresult[10][10];

for (int i = 0; i < 10; i++)

{

double temp;

for (int j = 0; j < 10; j++)

{

temp = 0;

for (int k = 0; k < 10; k++)

{

temp += matrixAs[i][k] \* matrixAc[k][j];

}

matrixAresult[i][j] = temp;

}

}

std::cout << "\n\n";

for (int i = 0; i < 10; i++)

{

for (int k = 0; k < 10; k++)

{

std::cout << matrixAresult[i][k] << " ";

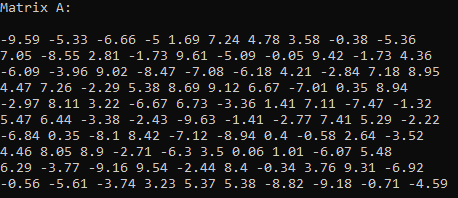
}

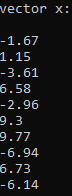
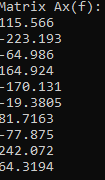
std::cout << "\n";

}

4) **Результаты**

Матрица A:

****

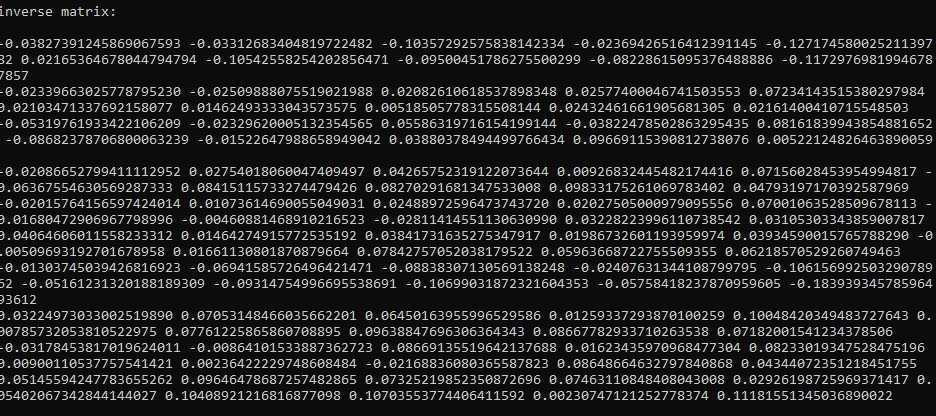
Вектор x: Вектор Ax(f):

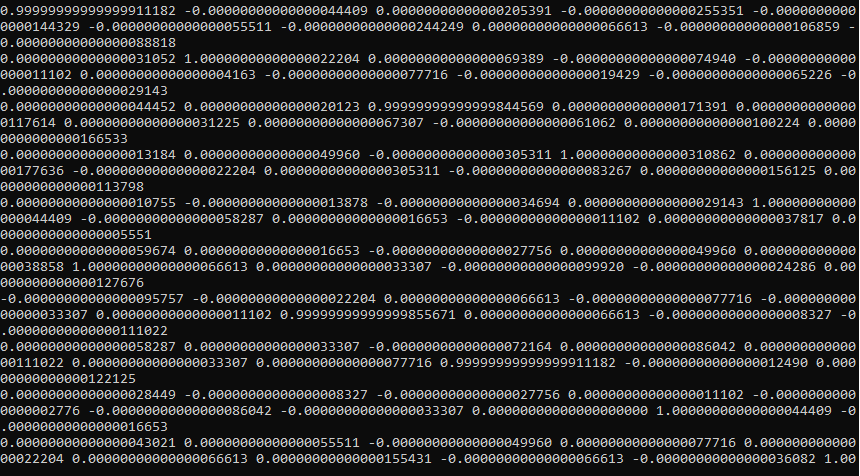
Вектор х\*

****

detA= 94904021896.89067077636718750000

Обратная матрица матрицы А и их произведение:

****

****

Максимум-норма невязки = 0.00000000000008526513

Максимум-норма погрешности= 0.00000000000001598721

**5) Вывод**

В действительности метод Гаусса с выбором главного элемента по столбцу дает довольно-таки точное решение системы линейных уравнение. Это следует из полученных в ходе выполнения лабораторной работы значения максимум-нормы невязки и максимум-нормы погрешности.