|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 1 | | |
| по дисциплине «Численные методы» | | |
| **ПРЯМЫЕ МЕТОДЫ РЕШЕНИЯ СЛАУ** | | |
|  | | |
|  |  |  |
| Группа: | ПМ-05 |
| Вариант: | 8 |
|  | чАГОЧКИНА ДАРЬЯ |
|  | КРИВЕЦКИЙ АНДРЕЙ |
| Преподаватели: | патрушев илья игоревич |
|  |  | задорожный александр геннадьевич |
| Новосибирск, 2024 | | |

1. **Цель**

Разработать программу решения СЛАУ методом с хранением матрицы в профильном формате. Исследовать накопление погрешности и ее зависимость от числа обусловленности. Сравнить реализованный метод по точности получаемого решения и количеству действий с методом Гаусса.

1. **Анализ задания**

* ***Решение СЛАУ методом разложения матрицы***

**Входные данные:** симметричная, положительно определенная матрица , вектор правых частей .

**Выходные данные:** вектор неизвестных .

Для хранения квадратной матрицы размерности в профильном формате используется следующая структура данных:

1) Вещественный массив ***di*** для хранения последовательно диагональных элементов матрицы.

2) Вещественный массив ***al*** для хранения внедиагональных элементов матрицы.

3) Целочисленный массив ***ia*** для хранения информации о профиле. Элемент ***ia(k)*** равен индексу, с которого начинаются элементы *k* -й строки (столбца) в массиве ***al****.*

Пусть дана СЛАУ вида . Тогда матрицу можно представить в виде:

Подставляя в , получим:

Обозначим:

Подставляя в , получим:

Решение СЛАУ сводится к 3 основным этапам:

1. Разбить матрицу А на элементы матриц ;
2. Решить систему с нижней треугольной матрицей (прямой ход);
3. Решить систему с верхней треугольной матрицей (обратный ход).

* ***Решение СЛАУ методом Гаусса (модификация)***

**Входные данные:** произвольная матрица , вектор правых частей .

Для хранения квадратной матрицы размерности в плотном формате используется n-мерный массив.

**Выходные данные:** вектор неизвестных .

Пусть дана СЛАУ вида . Ее решение состоит из двух этапов.

***Прямой ход, или последовательное исключение:***

1. Разделим все числа каждой строки на наибольшее число строки.
2. Среди элементов первого столбца матрицы выбираем максимальный элемент и перемещаем его на крайнее верхнее положение перестановкой строк.
3. Затем вычитаем получившуюся после перестановки первую строку из остальных строк, умноженных на некоторый коэффициент, где:

Умножение -й строки на число:

Вычитание -й строки из -й строки:

,

1. Получаем новую систему уравнений, в которой заменены соответствующие коэффициенты.
2. После того, как указанные преобразования были совершены, первую строку и первый столбец мысленно вычёркиваем и продолжаем указанный процесс для всех последующих уравнений пока не останется уравнение с одной неизвестной.

***Обратная подстановка, или обратный ход:***

Подставим полученное на предыдущем шаге значение переменной в предыдущие уравнения по формуле:

, ,

* ***Вычисление наибольшего/наименьшего собственных значений и числа обусловленности Тодда***

**Входные данные:** симметричная матрица , произвольный вектор малое число

, число для начального сравнения.

**Выходные данные:** наибольшее собственное число , число обусловленности Тодда

.

1. Вычислим скаляры , норму и вектор .
2. Вычислим (итерация нормированного вектора).
3. Вычислим , (скалярные произведения), , (приближение к нормированному собственному вектору), (приближение к собственному числу ).
4. Повторим шаги 2-3, пока .
5. Получим искомое максимальное собственное число матрицы A.
6. Найдем – максимальное собственное число матрицы A – .
7. Найдем – минимальное собственное число матрицы A.
8. Получим число обусловленности Тодда .
9. **Текст программы**

#include <iostream>

#include <fstream>

#include <string>

#define DF1

#ifdef DF1

typedef float real\_dp, real\_sum;

#endif

#ifdef DF2

typedef double real\_dp, real\_sum;

#endif

#ifdef DF3

typedef float real\_dp;

typedef double real\_sum;

#endif

using namespace std;

class profile\_matrix

{

public:

void input(string nm);

void calc\_chol();

void forward\_iteration();

void backward\_iteration();

void output(string nm);

void mul\_mat\_vec();

void set\_Hilbert(int k);

real\_dp calc\_cond();

real\_dp\* al = NULL, \* di = NULL, \* b = NULL, \* x = NULL;

int\* ia = NULL;

int n = 0;

private:

real\_dp scalar\_product(real\_dp\* a, real\_dp\* b);

real\_dp get\_max\_eig();

string name;

};

class dense\_matrix

{

public:

real\_dp\*\* a = NULL;

real\_dp\* b = NULL, \* x = NULL;

int n;

void profile2dense(profile\_matrix\* m);

void input\_dense(string name);

void gauss(bool change\_elem);

void output(string name);

};

int main()

{

profile\_matrix\* m = new profile\_matrix();

dense\_matrix\* d = new dense\_matrix();

m->input("in.txt");

m->calc\_chol();

m->forward\_iteration();

m->backward\_iteration();

m->output("out.txt");

}

void profile\_matrix::input(string nm)

{

name = nm;

ifstream fin(name);

fin >> n;

di = new real\_dp[n];

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

fin >> di[i];

ia = new int[n + 1];

for (int i = 0; i < n + 1; i++)

{

fin >> ia[i];

ia[i]--;

}

al = new real\_dp[ia[n]];

for (int i = 0; i < ia[n]; i++)

fin >> al[i];

b = new real\_dp[n];

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

fin >> b[i];

x = new real\_dp[n];

for (int i = 0; i > n; i++)

x[i] = real\_dp(i);

}

void profile\_matrix::set\_Hilbert(int k) // матрица Гильберта

{

n = k;

di = new real\_dp[k];

al = new real\_dp[(k \* k - k) / 2];

x = new real\_dp[k];

b = new real\_dp[k];

ia = new int[k + 1];

ia[0] = 0;

int c = 0;

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

{

ia[i + 1] = ia[i] + i;

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

{

al[c] = 1.0 / (i + j + 1);

c++;

}

di[i] = 1.0 / (2 \* i + 1);

}

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

x[i] = i + 1;

mul\_mat\_vec();

}

void profile\_matrix::mul\_mat\_vec() // умножение матрицы на вектор

{

for (int i = 0; i < n; i++) b[i] = 0;

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

{

int i0 = ia[i];

int i1 = ia[i + 1];

int j = i - (i1 - i0);

for (int m = i0; m < i1; m++, j++)

{

b[j] += al[m] \* x[i];

b[i] += al[m] \* x[j];

}

b[i] += di[i] \* x[i];

}

}

real\_dp profile\_matrix::scalar\_product(real\_dp\* a, real\_dp\* b) // скалярное произведение

{

real\_sum res = 0.0;

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

res += a[i] \* b[i];

return res;

}

real\_dp profile\_matrix::get\_max\_eig() // получение максимального собственного числа матрицы

{

real\_dp eps = 1e-5;

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

b[i] = 1.0;

real\_dp lam0 = 1.0;

real\_dp lam1 = 0.0;

real\_dp s = scalar\_product(b, b);

real\_dp norm = sqrt(s);

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

x[i] = b[i] / norm;

real\_dp t;

while (abs(lam1 - lam0) > eps)

{

lam0 = lam1;

mul\_mat\_vec();

s = scalar\_product(b, b);

t = scalar\_product(b, x);

norm = sqrt(s);

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

x[i] = b[i] / norm;

lam1 = s / t;

}

return lam1;

}

real\_dp profile\_matrix::calc\_cond() // нахождение числа обусловленности Тодда

{

real\_dp l\_max = get\_max\_eig();

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

di[i] -= l\_max;

real\_dp l\_min = l\_max + get\_max\_eig();

//input(name);

set\_Hilbert(n);

return l\_max / l\_min;

}

void profile\_matrix::calc\_chol() // разложение Холецкого

{

real\_sum sum\_diag = 0.0, sum\_outdiag = 0.0;

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

{

int i0 = ia[i];

int i1 = ia[i + 1];

int j = i - (i1 - i0);

for (int m = i0; m < i1; m++, j++)

{

int j0 = ia[j];

int j1 = ia[j + 1];

int ni = i0;

int nj = j0;

int kei = m - i0;

int kej = j1 - j0;

kei - kej > 0 ? ni += kei - kej : nj += kej - kei;

sum\_outdiag = 0.0;

for (int mi = ni, mj = nj; mj < j1; mi++, mj++)

sum\_outdiag += al[mi] \* al[mj];

al[m] = (al[m] - sum\_outdiag) / di[j];

sum\_diag += al[m] \* al[m];

}

di[i] = sqrt(di[i] - sum\_diag);

sum\_diag = 0.0;

}

}

void profile\_matrix::forward\_iteration() // прямой ход

{

// Решаем систему Ly = b, L - нижнетреугольная

// Результат записывается в b

real\_sum sum;

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

{

int j = i - (ia[i + 1] - ia[i]);

sum = 0.0;

for (int k = ia[i]; k < ia[i + 1]; k++, j++)

sum += b[j] \* al[k];

b[i] = (b[i] - sum) / di[i];

}

}

void profile\_matrix::backward\_iteration() // обратный ход

{

// Решаем систему L'x = y, L' - верхнетреугольная

// Результат записывается в b

int j = 0;

real\_sum sum = 0;

for (int i = n - 1; i >= 0; i--)

{

j = ia[i + 1] - ia[i];

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

{

j = ia[k + 1] - ia[k];

if (k - j <= i)

sum += al[ia[k] + i + j - k] \* b[k];

}

b[i] = (b[i] - sum) / di[i];

sum = 0;

}

}

void profile\_matrix::output(string nm)

{

ofstream fout(nm);

string s = typeid(x[0]).name();

fout << fixed;

if (s == "float")

fout.precision(7);

else

fout.precision(15);

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

fout << b[i] << endl;

}

void dense\_matrix::profile2dense(profile\_matrix\* m) // перевод профильной матрицы в плотный формат

{

n = m->n;

a = new real\_dp \* [n];

b = new real\_dp[n];

x = new real\_dp[n];

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

{

a[i] = new real\_dp[n];

b[i] = m->b[i];

}

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

{

int j = i - (m->ia[i + 1] - m->ia[i]);

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

{

a[i][k] = 0.0;

a[k][i] = 0.0;

}

for (int k = j, l = m->ia[i]; k < i; k++, l++)

{

a[i][k] = m->al[l];

a[k][i] = m->al[l];

}

a[i][i] = m->di[i];

}

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

{

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

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

cout << endl;

}

}

void dense\_matrix::input\_dense(string name)

{

ifstream fin(name);

fin >> n;

a = new real\_dp \* [n];

b = new real\_dp[n];

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

a[i] = new real\_dp[n];

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

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

fin >> a[i][j];

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

fin >> b[i];

}

void dense\_matrix::gauss(bool change\_elem) // метод Гаусса

{

if (change\_elem)

{

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

{

real\_dp max = 0.0;

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

if (a[k][i] > max)

max = a[k][i];

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

a[k][i] /= max;

b[k] /= max;

}

}

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

{

if (change\_elem)

{

real\_dp max = 0.0;

int ind\_m = 0;

for (int m = k; m < n; m++)

if (abs(a[m][k]) > max)

{

max = a[m][k];

ind\_m = m;

}

real\_dp tmp = b[k];

b[k] = b[ind\_m];

b[ind\_m] = tmp;

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

{

tmp = a[k][j];

a[k][j] = a[ind\_m][j];

a[ind\_m][j] = tmp;

}

}

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

{

float dif = a[i][k] / a[k][k];

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

a[i][j] -= dif \* a[k][j];

b[i] -= dif \* b[k];

}

}

for (int k = n - 1; k >= 0; k--)

{

real\_sum sum = 0.0;

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

sum += a[k][j] \* x[j];

x[k] = (b[k] - sum) / a[k][k];

}

}

void dense\_matrix::output(string name)

{

ofstream fout(name);

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

fout << x[i] << " ";

}

1. **Тестирование программы**

***Часть 1. Набор тестов для проверки правильности работы программы:***

*Ввод программы:*

n = 7

di = 1 2 3 4 5 6 7

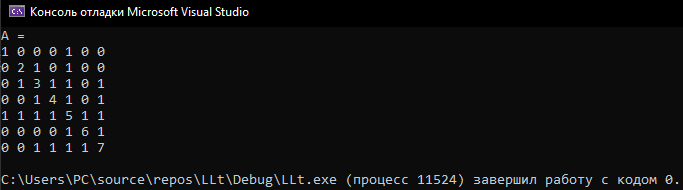
ia = 1 1 1 2 3 7 8 12

al = 1 1 1 1 1 1 1 1 1 1 1

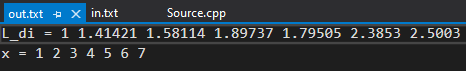
b = 6 12 27 31 48 48 67

*Вывод программы (тестирование в типе float):*

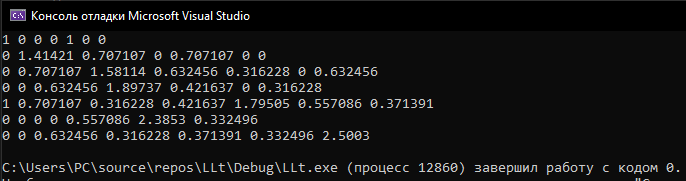
***Перевод матрицы из профильного формата в плотный***



***Решение СЛАУ методом разложения матрицы***



***Вид матрицы L в плотном формате***



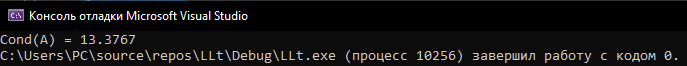
***Решение СЛАУ методом Гаусса (модификация)***



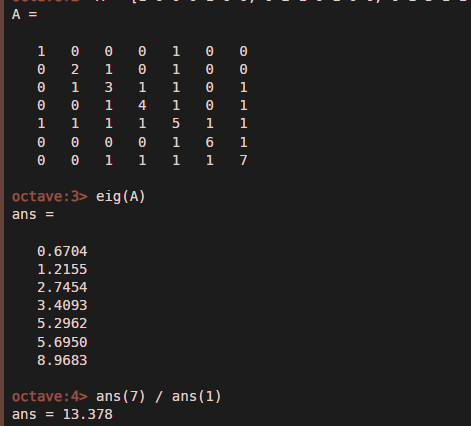
***Решение СЛАУ методом Гаусса***



***Нахождение числа обусловленности Тодда***



***Проверка собственных чисел матрицы А(в среде Octave):***



***Часть 2.***

*Исходная матрица А:*













1. **Оценка влияния увеличения числа обусловленности на точность решения**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **k, cond(A)** | **Одинарная точность** | | **Двойная точность** | | **Скалярное произведение** | |
|  | **x\* –** |  | **x\* –** |  | **x\* –** |
| 0 | **1,0000E+00** | **-6,7000E-06** | **1,0000E+00** | **-1,3989E-14** | **1,0000E+00** | **-1,2400E-05** |
| **2,0000E+00** | **-6,0000E-06** | **2,0000E+00** | **-1,9984E-14** | **2,0000E+00** | **-1,1200E-05** |
| **3,0000E+00** | **-6,0000E-06** | **3,0000E+00** | **-1,9984E-14** | **3,0000E+00** | **-1,1900E-05** |
| **4,0000E+00** | **-5,7000E-06** | **4,0000E+00** | **-1,9984E-14** | **4,0000E+00** | **-1,0000E-05** |
| **5,0000E+00** | **-6,2000E-06** | **5,0000E+00** | **-2,0428E-14** | **5,0000E+00** | **-1,1900E-05** |
| **6,0000E+00** | **-6,7000E-06** | **6,0000E+00** | **-2,0428E-14** | **6,0000E+00** | **-1,2400E-05** |
| **7,0000E+00** | **-7,2000E-06** | **7,0000E+00** | **-2,0428E-14** | **7,0000E+00** | **-1,1900E-05** |
| **8,0000E+00** | **-5,7000E-06** | **8,0000E+00** | **-2,0428E-14** | **8,0000E+00** | **-1,2400E-05** |
| **9,0000E+00** | **-6,7000E-06** | **9,0000E+00** | **-1,9540E-14** | **9,0000E+00** | **-1,2400E-05** |
| **1,0000E+01** | **-7,6000E-06** | **1,0000E+01** | **-1,9540E-14** | **1,0000E+01** | **-1,2400E-05** |
| 1 | **8,5847E-01** | **1,4153E-01** | **8,5847E-01** | **-1,4153E-01** | **8,5846E-01** | **1,4154E-01** |
| **1,9507E+00** | **4,9303E-02** | **1,9507E+00** | **-4,9306E-02** | **1,9507E+00** | **4,9310E-02** |
| **2,9478E+00** | **5,2176E-02** | **2,9478E+00** | **-5,2179E-02** | **2,9478E+00** | **5,2184E-02** |
| **3,9571E+00** | **4,2921E-02** | **3,9571E+00** | **-4,2923E-02** | **3,9571E+00** | **4,2927E-02** |
| **4,9500E+00** | **5,0034E-02** | **4,9500E+00** | **-5,0038E-02** | **4,9500E+00** | **5,0042E-02** |
| **5,9443E+00** | **5,5685E-02** | **5,9443E+00** | **-5,5688E-02** | **5,9443E+00** | **5,5693E-02** |
| **6,9509E+00** | **4,9093E-02** | **6,9509E+00** | **-4,9096E-02** | **6,9509E+00** | **4,9100E-02** |
| **7,9483E+00** | **5,1680E-02** | **7,9483E+00** | **-5,1683E-02** | **7,9483E+00** | **5,1688E-02** |
| **8,9478E+00** | **5,2195E-02** | **8,9478E+00** | **-5,2198E-02** | **8,9478E+00** | **5,2203E-02** |
| **9,9483E+00** | **5,1741E-02** | **9,9483E+00** | **-5,1744E-02** | **9,9483E+00** | **5,1749E-02** |
| 2 | **9,8378E-01** | **1,6217E-02** | **9,8378E-01** | **-1,6220E-02** | **9,8377E-01** | **1,6226E-02** |
| **1,9944E+00** | **5,6475E-03** | **1,9943E+00** | **-5,6503E-03** | **1,9943E+00** | **5,6566E-03** |
| **2,9940E+00** | **5,9769E-03** | **2,9940E+00** | **-5,9796E-03** | **2,9940E+00** | **5,9862E-03** |
| **3,9951E+00** | **4,9157E-03** | **3,9951E+00** | **-4,9189E-03** | **3,9951E+00** | **4,9243E-03** |
| **4,9943E+00** | **5,7316E-03** | **4,9943E+00** | **-5,7342E-03** | **4,9943E+00** | **5,7402E-03** |
| **5,9936E+00** | **6,3787E-03** | **5,9936E+00** | **-6,3817E-03** | **5,9936E+00** | **6,3882E-03** |
| **6,9944E+00** | **5,6233E-03** | **6,9944E+00** | **-5,6263E-03** | **6,9944E+00** | **5,6324E-03** |
| **7,9941E+00** | **5,9190E-03** | **7,9941E+00** | **-5,9227E-03** | **7,9941E+00** | **5,9299E-03** |
| **8,9940E+00** | **5,9786E-03** | **8,9940E+00** | **-5,9818E-03** | **8,9940E+00** | **5,9881E-03** |
| **9,9941E+00** | **5,9271E-03** | **9,9941E+00** | **-5,9298E-03** | **9,9941E+00** | **5,9366E-03** |
| 3 | **9,9835E-01** | **1,6482E-03** | **9,9835E-01** | **-1,6460E-03** | **9,9835E-01** | **1,6501E-03** |
| **1,9994E+00** | **5,7580E-04** | **1,9994E+00** | **-5,7340E-04** | **1,9994E+00** | **5,7720E-04** |
| **2,9994E+00** | **6,0890E-04** | **2,9994E+00** | **-6,0682E-04** | **2,9994E+00** | **6,1080E-04** |
| **3,9995E+00** | **5,0070E-04** | **3,9995E+00** | **-4,9918E-04** | **3,9995E+00** | **5,0230E-04** |
| **4,9994E+00** | **5,8410E-04** | **4,9994E+00** | **-5,8192E-04** | **4,9994E+00** | **5,8560E-04** |
| **5,9994E+00** | **6,4990E-04** | **5,9994E+00** | **-6,4763E-04** | **5,9993E+00** | **6,5180E-04** |
| **6,9994E+00** | **5,7320E-04** | **6,9994E+00** | **-5,7096E-04** | **6,9994E+00** | **5,7460E-04** |
| **7,9994E+00** | **6,0370E-04** | **7,9994E+00** | **-6,0105E-04** | **7,9994E+00** | **6,0560E-04** |
| **8,9994E+00** | **6,0940E-04** | **8,9994E+00** | **-6,0704E-04** | **8,9994E+00** | **6,1130E-04** |
| **9,9994E+00** | **6,0460E-04** | **9,9994E+00** | **-6,0176E-04** | **9,9994E+00** | **6,0560E-04** |
| 4 | **9,9983E-01** | **1,7490E-04** | **9,9984E-01** | **-1,6484E-04** | **9,9985E-01** | **1,5440E-04** |
| **1,9999E+00** | **6,6900E-05** | **1,9999E+00** | **-5,7425E-05** | **2,0000E+00** | **4,8500E-05** |
| **2,9999E+00** | **7,0800E-05** | **2,9999E+00** | **-6,0772E-05** | **2,9999E+00** | **5,1000E-05** |
| **3,9999E+00** | **5,8200E-05** | **4,0000E+00** | **-4,9992E-05** | **4,0000E+00** | **4,2000E-05** |
| **4,9999E+00** | **6,8200E-05** | **4,9999E+00** | **-5,8278E-05** | **5,0000E+00** | **4,8600E-05** |
| **5,9999E+00** | **7,5300E-05** | **5,9999E+00** | **-6,4859E-05** | **5,9999E+00** | **5,4800E-05** |
| **6,9999E+00** | **6,7200E-05** | **6,9999E+00** | **-5,7181E-05** | **7,0000E+00** | **4,8200E-05** |
| **7,9999E+00** | **7,0600E-05** | **7,9999E+00** | **-6,0194E-05** | **7,9999E+00** | **5,0100E-05** |
| **8,9999E+00** | **7,1500E-05** | **8,9999E+00** | **-6,0794E-05** | **8,9999E+00** | **5,0500E-05** |
| **9,9999E+00** | **7,0600E-05** | **9,9999E+00** | **-6,0265E-05** | **9,9999E+00** | **5,0500E-05** |
| 5 | **9,9999E-01** | **9,3000E-06** | **9,9998E-01** | **-1,6487E-05** | **9,9998E-01** | **2,0300E-05** |
| **2,0000E+00** | **-1,2000E-06** | **2,0000E+00** | **-5,7434E-06** | **2,0000E+00** | **9,3000E-06** |
| **3,0000E+00** | **-1,2000E-06** | **3,0000E+00** | **-6,0781E-06** | **3,0000E+00** | **9,5000E-06** |
| **4,0000E+00** | **-1,4000E-06** | **4,0000E+00** | **-4,9999E-06** | **4,0000E+00** | **7,6000E-06** |
| **5,0000E+00** | **-1,4000E-06** | **5,0000E+00** | **-5,8287E-06** | **5,0000E+00** | **8,6000E-06** |
| **6,0000E+00** | **-1,0000E-06** | **6,0000E+00** | **-6,4868E-06** | **6,0000E+00** | **1,0000E-05** |
| **7,0000E+00** | **-2,4000E-06** | **7,0000E+00** | **-5,7190E-06** | **7,0000E+00** | **8,1000E-06** |
| **8,0000E+00** | **-1,9000E-06** | **8,0000E+00** | **-6,0203E-06** | **8,0000E+00** | **9,1000E-06** |
| **9,0000E+00** | **-1,0000E-06** | **9,0000E+00** | **-6,0803E-06** | **9,0000E+00** | **9,5000E-06** |
| **1,0000E+01** | **-1,9000E-06** | **1,0000E+01** | **-6,0274E-06** | **1,0000E+01** | **9,5000E-06** |
| 6,  134,366 | **9,9999E-01** | **1,4300E-05** | **1,0000E+00** | **-1,6487E-06** | **1,0000E+00** | **-5,0000E-07** |
| **2,0000E+00** | **1,2300E-05** | **2,0000E+00** | **-5,7435E-07** | **2,0000E+00** | **-1,0000E-06** |
| **3,0000E+00** | **1,3600E-05** | **3,0000E+00** | **-6,0782E-07** | **3,0000E+00** | **-1,2000E-06** |
| **4,0000E+00** | **1,0500E-05** | **4,0000E+00** | **-5,0000E-07** | **4,0000E+00** | **-1,4000E-06** |
| **5,0000E+00** | **1,2900E-05** | **5,0000E+00** | **-5,8288E-07** | **5,0000E+00** | **-1,9000E-06** |
| **6,0000E+00** | **1,3400E-05** | **6,0000E+00** | **-6,4869E-07** | **6,0000E+00** | **-1,4000E-06** |
| **7,0000E+00** | **1,2400E-05** | **7,0000E+00** | **-5,7191E-07** | **7,0000E+00** | **-1,9000E-06** |
| **8,0000E+00** | **1,3400E-05** | **8,0000E+00** | **-6,0204E-07** | **8,0000E+00** | **-1,0000E-06** |
| **9,0000E+00** | **1,2400E-05** | **9,0000E+00** | **-6,0804E-07** | **9,0000E+00** | **-1,9000E-06** |
| **1,0000E+01** | **1,3400E-05** | **1,0000E+01** | **-6,0275E-07** | **1,0000E+01** | **-1,9000E-06** |
| 7,  134,367 | **1,0000E+00** | **-6,7000E-06** | **1,0000E+00** | **-1,6487E-07** | **1,0000E+00** | **-1,2400E-05** |
| **2,0000E+00** | **-6,0000E-06** | **2,0000E+00** | **-5,7435E-08** | **2,0000E+00** | **-1,1200E-05** |
| **3,0000E+00** | **-6,0000E-06** | **3,0000E+00** | **-6,0782E-08** | **3,0000E+00** | **-1,1900E-05** |
| **4,0000E+00** | **-5,7000E-06** | **4,0000E+00** | **-5,0000E-08** | **4,0000E+00** | **-1,0000E-05** |
| **5,0000E+00** | **-6,2000E-06** | **5,0000E+00** | **-5,8288E-08** | **5,0000E+00** | **-1,1900E-05** |
| **6,0000E+00** | **-6,7000E-06** | **6,0000E+00** | **-6,4869E-08** | **6,0000E+00** | **-1,2400E-05** |
| **7,0000E+00** | **-7,2000E-06** | **7,0000E+00** | **-5,7191E-08** | **7,0000E+00** | **-1,1900E-05** |
| **8,0000E+00** | **-5,7000E-06** | **8,0000E+00** | **-6,0204E-08** | **8,0000E+00** | **-1,2400E-05** |
| **9,0000E+00** | **-6,7000E-06** | **9,0000E+00** | **-6,0804E-08** | **9,0000E+00** | **-1,2400E-05** |
| **1,0000E+01** | **-7,6000E-06** | **1,0000E+01** | **-6,0275E-08** | **1,0000E+01** | **-1,2400E-05** |
| 8,  134,367 | **1,0000E+00** | **-6,7000E-06** | **1,0000E+00** | **-1,6487E-08** | **1,0000E+00** | **-1,2400E-05** |
| **2,0000E+00** | **-6,0000E-06** | **2,0000E+00** | **-5,7435E-09** | **2,0000E+00** | **-1,1200E-05** |
| **3,0000E+00** | **-6,0000E-06** | **3,0000E+00** | **-6,0782E-09** | **3,0000E+00** | **-1,1900E-05** |
| **4,0000E+00** | **-5,7000E-06** | **4,0000E+00** | **-5,0000E-09** | **4,0000E+00** | **-1,0000E-05** |
| **5,0000E+00** | **-6,2000E-06** | **5,0000E+00** | **-5,8288E-09** | **5,0000E+00** | **-1,1900E-05** |
| **6,0000E+00** | **-6,7000E-06** | **6,0000E+00** | **-6,4870E-09** | **6,0000E+00** | **-1,2400E-05** |
| **7,0000E+00** | **-7,2000E-06** | **7,0000E+00** | **-5,7191E-09** | **7,0000E+00** | **-1,1900E-05** |
| **8,0000E+00** | **-5,7000E-06** | **8,0000E+00** | **-6,0204E-09** | **8,0000E+00** | **-1,2400E-05** |
| **9,0000E+00** | **-6,7000E-06** | **9,0000E+00** | **-6,0804E-09** | **9,0000E+00** | **-1,2400E-05** |
| **1,0000E+01** | **-7,6000E-06** | **1,0000E+01** | **-6,0275E-09** | **1,0000E+01** | **-1,2400E-05** |
| 9 |  |  | **1,0000E+00** | **-1,6487E-09** |  |  |
|  |  | **2,0000E+00** | **-5,7435E-10** |  |  |
|  |  | **3,0000E+00** | **-6,0782E-10** |  |  |
|  |  | **4,0000E+00** | **-5,0000E-10** |  |  |
|  |  | **5,0000E+00** | **-5,8288E-10** |  |  |
|  |  | **6,0000E+00** | **-6,4870E-10** |  |  |
|  |  | **7,0000E+00** | **-5,7191E-10** |  |  |
|  |  | **8,0000E+00** | **-6,0204E-10** |  |  |
|  |  | **9,0000E+00** | **-6,0804E-10** |  |  |
|  |  | **1,0000E+01** | **-6,0275E-10** |  |  |
| 10 |  |  | **1,0000E+00** | **-1,6490E-10** |  |  |
|  |  | **2,0000E+00** | **-5,7470E-11** |  |  |
|  |  | **3,0000E+00** | **-6,0820E-11** |  |  |
|  |  | **4,0000E+00** | **-5,0030E-11** |  |  |
|  |  | **5,0000E+00** | **-5,8320E-11** |  |  |
|  |  | **6,0000E+00** | **-6,4910E-11** |  |  |
|  |  | **7,0000E+00** | **-5,7230E-11** |  |  |
|  |  | **8,0000E+00** | **-6,0240E-11** |  |  |
|  |  | **9,0000E+00** | **-6,0840E-11** |  |  |
|  |  | **1,0000E+01** | **-6,0311E-11** |  |  |
| 11 |  |  | **1,0000E+00** | **-1,6504E-11** |  |  |
|  |  | **2,0000E+00** | **-5,7601E-12** |  |  |
|  |  | **3,0000E+00** | **-6,1000E-12** |  |  |
|  |  | **4,0000E+00** | **-5,0200E-12** |  |  |
|  |  | **5,0000E+00** | **-5,8504E-12** |  |  |
|  |  | **6,0000E+00** | **-6,5103E-12** |  |  |
|  |  | **7,0000E+00** | **-5,7403E-12** |  |  |
|  |  | **8,0000E+00** | **-6,0396E-12** |  |  |
|  |  | **9,0000E+00** | **-6,1000E-12** |  |  |
|  |  | **1,0000E+01** | **-6,0503E-12** |  |  |
| 12 |  |  | **1,0000E+00** | **-1,6590E-12** |  |  |
|  |  | **2,0000E+00** | **-5,8997E-13** |  |  |
|  |  | **3,0000E+00** | **-6,1995E-13** |  |  |
|  |  | **4,0000E+00** | **-5,0981E-13** |  |  |
|  |  | **5,0000E+00** | **-6,0041E-13** |  |  |
|  |  | **6,0000E+00** | **-6,5992E-13** |  |  |
|  |  | **7,0000E+00** | **-5,7998E-13** |  |  |
|  |  | **8,0000E+00** | **-6,1995E-13** |  |  |
|  |  | **9,0000E+00** | **-6,1995E-13** |  |  |
|  |  | **1,0000E+01** | **-6,1995E-13** |  |  |
| 13 |  |  | **1,0000E+00** | **-1,5798E-13** |  |  |
|  |  | **2,0000E+00** | **-5,9952E-14** |  |  |
|  |  | **3,0000E+00** | **-5,9952E-14** |  |  |
|  |  | **4,0000E+00** | **-5,0182E-14** |  |  |
|  |  | **5,0000E+00** | **-6,0396E-14** |  |  |
|  |  | **6,0000E+00** | **-6,0396E-14** |  |  |
|  |  | **7,0000E+00** | **-4,9738E-14** |  |  |
|  |  | **8,0000E+00** | **-6,0396E-14** |  |  |
|  |  | **9,0000E+00** | **-6,0396E-14** |  |  |
|  |  | **1,0000E+01** | **-6,0396E-14** |  |  |
| 14 |  |  | **1,0000E+00** | **-4,7962E-14** |  |  |
|  |  | **2,0000E+00** | **-3,9968E-14** |  |  |
|  |  | **3,0000E+00** | **-3,9968E-14** |  |  |
|  |  | **4,0000E+00** | **-3,0198E-14** |  |  |
|  |  | **5,0000E+00** | **-3,9968E-14** |  |  |
|  |  | **6,0000E+00** | **-3,9968E-14** |  |  |
|  |  | **7,0000E+00** | **-3,9968E-14** |  |  |
|  |  | **8,0000E+00** | **-3,9968E-14** |  |  |
|  |  | **9,0000E+00** | **-4,0856E-14** |  |  |
|  |  | **1,0000E+01** | **-4,0856E-14** |  |  |
| 15 |  |  | **1,0000E+00** | **-1,5987E-14** |  |  |
|  |  | **2,0000E+00** | **-1,9984E-14** |  |  |
|  |  | **3,0000E+00** | **-1,9984E-14** |  |  |
|  |  | **4,0000E+00** | **-1,9984E-14** |  |  |
|  |  | **5,0000E+00** | **-2,0428E-14** |  |  |
|  |  | **6,0000E+00** | **-2,0428E-14** |  |  |
|  |  | **7,0000E+00** | **-2,0428E-14** |  |  |
|  |  | **8,0000E+00** | **-2,0428E-14** |  |  |
|  |  | **9,0000E+00** | **-1,9540E-14** |  |  |
|  |  | **1,0000E+01** | **-1,9540E-14** |  |  |

1. **Оценка влияния числа обусловленности на точность решения на матрицах Гильберта**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **k, cond(A)** | **Одинарная точность** | | **Двойная точность** | | **Скалярное произведение** | |
|  | **x\* –** |  | **x\* –** |  | **x\* –** |
| 2,  19,2815 | **1,0000E+00** | **4,0000E-07** | **1,0000E+00** | **0,0000E+00** | **1,0000E+00** | **4,0000E-07** |
| **2,0000E+00** | **-7,0000E-07** | **2,0000E+00** | **9,9920E-15** | **2,0000E+00** | **-7,0000E-07** |
| 3,  515,085 | **1,0000E+00** | **-1,2000E-06** | **1,0000E+00** | **0,0000E+00** | **1,0000E+00** | **-5,0000E-07** |
| **2,0000E+00** | **9,7000E-06** | **2,0000E+00** | **4,9960E-14** | **2,0000E+00** | **4,8000E-06** |
| **3,0000E+00** | **-1,0500E-05** | **3,0000E+00** | **-3,9968E-14** | **3,0000E+00** | **-5,5000E-06** |
| 4,  227,304 | **1,0000E+00** | **-2,8600E-05** | **1,0000E+00** | **-2,9976E-14** | **1,0000E+00** | **-2,2900E-05** |
| **1,9997E+00** | **3,2150E-04** | **2,0000E+00** | **3,7992E-13** | **1,9997E+00** | **2,6040E-04** |
| **3,0008E+00** | **-7,7200E-04** | **3,0000E+00** | **-9,2992E-13** | **3,0006E+00** | **-6,2700E-04** |
| **3,9995E+00** | **5,0020E-04** | **4,0000E+00** | **6,3016E-13** | **3,9996E+00** | **4,0700E-04** |
| 5,  145,796 | **1,0001E+00** | **-1,1490E-04** | **1,0000E+00** | **-6,9944E-14** | **9,9991E-01** | **9,2300E-05** |
| **1,9976E+00** | **2,3817E-03** | **2,0000E+00** | **1,7999E-12** | **2,0016E+00** | **-1,5869E-03** |
| **3,0109E+00** | **-1,0858E-02** | **3,0000E+00** | **-8,7201E-12** | **2,9935E+00** | **6,4898E-03** |
| **3,9830E+00** | **1,6985E-02** | **4,0000E+00** | **1,4230E-11** | **4,0094E+00** | **-9,4452E-03** |
| **5,0085E+00** | **-8,5135E-03** | **5,0000E+00** | **-7,3399E-12** | **4,9955E+00** | **4,4961E-03** |
| 6,  1447,28 | **9,9817E-01** | **1,8306E-03** | **1,0000E+00** | **-2,3399E-12** | **9,9799E-01** | **2,0142E-03** |
| **2,0536E+00** | **-5,3568E-02** | **2,0000E+00** | **6,6460E-11** | **2,0578E+00** | **-5,7771E-02** |
| **2,6310E+00** | **3,6897E-01** | **3,0000E+00** | **-4,4802E-10** | **2,6074E+00** | **3,9257E-01** |
| **4,9725E+00** | **-9,7248E-01** | **4,0000E+00** | **1,1626E-09** | **5,0246E+00** | **-1,0246E+00** |
| **3,9157E+00** | **1,0843E+00** | **5,0000E+00** | **-1,2810E-09** | **3,8660E+00** | **1,1340E+00** |
| **6,4306E+00** | **-4,3062E-01** | **6,0000E+00** | **5,0410E-10** | **6,4478E+00** | **-4,4779E-01** |
| 7,  1213,83 | **9,9434E-01** | **5,6567E-03** | **1,0000E+00** | **3,9122E-11** | **9,9276E-01** | **7,2376E-03** |
| **2,2136E+00** | **-2,1355E-01** | **2,0000E+00** | **-1,5364E-09** | **2,2813E+00** | **-2,8128E-01** |
| **1,0251E+00** | **1,9749E+00** | **3,0000E+00** | **1,4641E-08** | **3,3553E-01** | **2,6645E+00** |
| **1,1438E+01** | **-7,4383E+00** | **4,0000E+00** | **-5,6473E-08** | **1,4236E+01** | **-1,0236E+01** |
| **-8,2956E+00** | **1,3296E+01** | **5,0000E+00** | **1,0292E-07** | **-1,3601E+01** | **1,8601E+01** |
| **1,7253E+01** | **-1,1253E+01** | **6,0000E+00** | **-8,8521E-08** | **2,1962E+01** | **-1,5962E+01** |
| **3,3684E+00** | **3,6316E+00** | **7,0000E+00** | **2,8958E-08** | **1,7877E+00** | **5,2123E+00** |
| 8,  986,932 |  |  | **1,0000E+00** | **1,3348E-10** |  |  |
|  |  | **2,0000E+00** | **-7,0506E-09** |  |  |
|  |  | **3,0000E+00** | **9,1276E-08** |  |  |
|  |  | **4,0000E+00** | **-4,9132E-07** |  |  |
|  |  | **5,0000E+00** | **1,3182E-06** |  |  |
|  |  | **6,0000E+00** | **-1,8609E-06** |  |  |
|  |  | **7,0000E+00** | **1,3223E-06** |  |  |
|  |  | **8,0000E+00** | **-3,7269E-07** |  |  |
|  |  | **1,0000E+00** | **8,6548E-10** |  |  |
| 9,  815,223 |  |  | **2,0000E+00** | **-6,0356E-08** |  |  |
|  |  | **3,0000E+00** | **1,0285E-06** |  |  |
|  |  | **4,0000E+00** | **-7,3765E-06** |  |  |
|  |  | **5,0000E+00** | **2,7149E-05** |  |  |
|  |  | **6,0001E+00** | **-5,5579E-05** |  |  |
|  |  | **6,9999E+00** | **6,3960E-05** |  |  |
|  |  | **8,0000E+00** | **-3,8697E-05** |  |  |
|  |  | **9,0000E+00** | **9,5739E-06** |  |  |
|  |  | **1,0000E+00** | **4,2290E-09** |  |  |
| 10,  686,991 |  |  | **2,0000E+00** | **-3,7433E-07** |  |  |
|  |  | **3,0000E+00** | **8,1119E-06** |  |  |
|  |  | **4,0001E+00** | **-7,4706E-05** |  |  |
|  |  | **4,9996E+00** | **3,5993E-04** |  |  |
|  |  | **6,0010E+00** | **-9,9739E-04** |  |  |
|  |  | **6,9984E+00** | **1,6471E-03** |  |  |
|  |  | **8,0016E+00** | **-1,6004E-03** |  |  |
|  |  | **8,9992E+00** | **8,4402E-04** |  |  |
|  |  | **1,0000E+01** | **-1,8634E-04** |  |  |
|  |  | **1,0000E+00** | **9,2921E-08** |  |  |
| 11,  589,756 |  |  | **2,0000E+00** | **-1,0009E-05** |  |  |
|  |  | **2,9997E+00** | **2,6576E-04** |  |  |
|  |  | **4,0030E+00** | **-3,0291E-03** |  |  |
|  |  | **4,9817E+00** | **1,8341E-02** |  |  |
|  |  | **6,0654E+00** | **-6,5384E-02** |  |  |
|  |  | **6,8559E+00** | **1,4408E-01** |  |  |
|  |  | **8,1985E+00** | **-1,9848E-01** |  |  |
|  |  | **8,8336E+00** | **1,6639E-01** |  |  |
|  |  | **1,0078E+01** | **-7,7613E-02** |  |  |
|  |  | **1,0985E+01** | **1,5443E-02** |  |  |
|  |  | **1,0000E+00** | **-4,1444E-07** |  |  |
| 12,  517,131 |  |  | **1,9999E+00** | **5,2636E-05** |  |  |
|  |  | **3,0017E+00** | **-1,6578E-03** |  |  |
|  |  | **3,9774E+00** | **2,2612E-02** |  |  |
|  |  | **5,1659E+00** | **-1,6590E-01** |  |  |
|  |  | **5,2706E+00** | **7,2937E-01** |  |  |
|  |  | **9,0332E+00** | **-2,0332E+00** |  |  |
|  |  | **4,3185E+00** | **3,6815E+00** |  |  |
|  |  | **1,3317E+01** | **-4,3172E+00** |  |  |
|  |  | **6,8376E+00** | **3,1624E+00** |  |  |
|  |  | **1,2315E+01** | **-1,3150E+00** |  |  |
|  |  | **1,1763E+01** | **2,3695E-01** |  |  |
| **1,0000E+00** | **4,0000E-07** | **1,0000E+00** | **0,0000E+00** | **1,0000E+00** | **4,0000E-07** |

1. **Метод Гаусса**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Одинарная точность** | | **Двойная точность** | | **Скалярное произведение** | |
|  | **x\* –** |  | **x\* –** |  | **x\* –** |
| **1,00002** | **-2E-05** | **1** | **0** | **1,00002** | **-2E-05** |
| **2,00002** | **-2E-05** | **2** | **0** | **2,00001** | **-1E-05** |
| **3,00002** | **-2E-05** | **3** | **0** | **3,00002** | **-2E-05** |
| **4,00001** | **-1E-05** | **4** | **0** | **4,00001** | **-1E-05** |
| **5,00002** | **-2E-05** | **5** | **0** | **5,00002** | **-2E-05** |
| **6,00002** | **-2E-05** | **6** | **0** | **6,00002** | **-2E-05** |
| **7,00002** | **-2E-05** | **7** | **0** | **7,00002** | **-2E-05** |
| **8,00002** | **-2E-05** | **8** | **0** | **8,00002** | **-2E-05** |
| **9,00002** | **-2E-05** | **9** | **0** | **9,00002** | **-2E-05** |
| **10** | **0** | **10** | **0** | **10** | **0** |

1. **Выводы**