Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
—
Институт компьютерных наук и кибербезопастности
Высшая школа программной инженерии

Лабораторная работа №2
по дисциплине «Вычислительная математика»

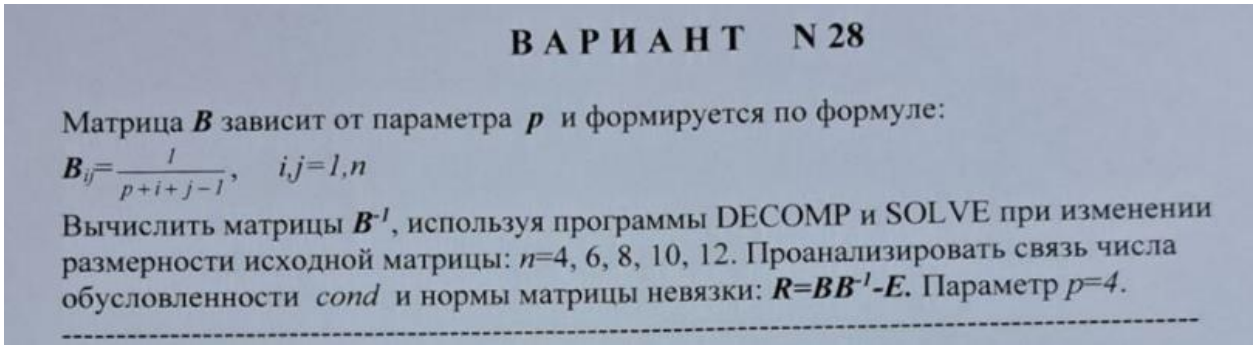
Выполнил
Студент группы 5130904/20004 Машкин А.А.

Преподаватель Устинов С.М.

Санкт-Петербург
2024

Оглавление

Задание



ВАРИАНТ N 28

Матрица $\boldsymbol{B}$ зависит от параметра p и формируется по формуле:

$$B_{ij}=\frac{1}{p+i+j-1}, \quad i,j=1,n$$

Вычислить матрицы $\boldsymbol{B^{-1}}$, используя программы DECOMP и SOLVE при изменении размерности исходной матрицы: n=4, 6, 8, 10, 12. Проанализировать связь числа обусловленности *cond* и нормы матрицы невязки: $\boldsymbol{R=BB^{-1}-E}$. Параметр p=4.

Санкт-Петербург
2024

Результаты

```
------------------------------------------------------------------------
Result for order of matrix = 4
Condition number = 156097
The norm of a matrix: 0.00000000000000022
------------------------------------------------------------------------


------------------------------------------------------------------------
Result for order of matrix = 6
Condition number = 227026846.11517279298277572
The norm of a matrix: 0.00000000000066791
------------------------------------------------------------------------


------------------------------------------------------------------------
Result for order of matrix = 8
Condition number = 293917944054.69198983907699585
The norm of a matrix: 0.00000000064642336
------------------------------------------------------------------------


------------------------------------------------------------------------
Result for order of matrix = 10
Condition number = 361460089257909.94595336914062500
The norm of a matrix: 0.00000063321658672
------------------------------------------------------------------------


------------------------------------------------------------------------
Result for order of matrix = 12
Ehd with error flag = 3
------------------------------------------------------------------------
```

Вывод

В лабораторной работе видно, что при росте числа обусловленности растет и норма матрицы невязки. Но на матрице порядка 12 программа выдает ошибку с флагом равным 3, это означает, что определитель в этом случае у нас равен нулю.

Санкт-Петербург
2024

Код программы

<DIR>/computational_mathematics/lab_2/main.cpp

```
#include <iostream>
#include <iomanip>
#include <algorithm>
#include "decomp.h"
#include "solve.h"
#include "matrix_function.h"

int main() {
  const auto default_precision{std::cout.precision()};
  static long double cond = 0;
  double det = 0;
  const int ndim = 12;
  int n = 2;
  int p = 4;
  int pivot[ndim], i, flag;
  long double B[ndim * ndim] = {0};
  long double E[ndim * ndim] = {0};
  long double result_for_multiplication[ndim * ndim] = {0};
  long double result_for_subtraction[ndim * ndim] = {0};
  long double copy_of_b[ndim * ndim] = {0};
  long double copy_of_e[ndim * ndim] = {0};
  for (int k = 0; k < 5; k++)
  {
    n += 2;
    for (int i = 0; i < ndim; i++)
    {
      for (int j = 0; j < ndim; j++)
      {
        B[i * ndim + j] = 1.0 / (p + i + j - 1);
        if (i == j)
        {
          E[i * ndim + j] = 1;
        }
        else
        {
          E[i * ndim + j] = 0;
        }
      }
    }
    if (k == 0)
    {
      std::copy(B, B + ndim * ndim, copy_of_b);
      std::copy(E, E + ndim * ndim, copy_of_e);
    }
    decomp(n, ndim, B, &cond, pivot, &flag);
    std::cout << "------------------------------------------------------------
-----------\n";
```

```
    std::cout << "Result for order of matrix = " << n << "\n";
    if (flag == 0)
    {
      for (int i = 0; i < n; i++)
      {
        solve(n, ndim, B, (E + i * ndim), pivot);
      }
      std::cout << "Condition number = " << cond << "\n" << std::fixed <<
std::setprecision(17);// << std::scientific << std::setprecision(1);

//      std::cout << "R = [\n";
      mashkin::multiply(n, ndim, copy_of_b, E, result_for_multiplication);
      mashkin::subtract(n, ndim, result_for_multiplication, copy_of_e,
result_for_subtraction);
//      mashkin::print_matrix(n, ndim, result_for_subtraction);
//      std::cout << "]\n";
      std::cout << "The norm of a matrix: ";
      std::cout <<
mashkin::calculate_the_norm_of_a_matrix(result_for_subtraction, ndim);
      std::cout << "\n";

      det = pivot[n-1];
      for (i = 0; i < n; i++)
        det = det * B[i * ndim + i];
    }
    else
    {
      std::cout << "Ehd with error flag = " << flag << "\n";
    }
    std::cout << "-------------------------------------------------------------
-----------\n\n";
  }
  return 0;
}
```

<DIR>/computational_mathematics/lab_2/solve.h

```
#ifndef LAB2_SOLVE_H
#define LAB2_SOLVE_H

int solve (int n, int ndim, long double *a, long double b[], int pivot[]);
#endif
```

<DIR>/computational_mathematics/lab_2/solve.cpp

```
#include "solve.h"

#define AINDEX(i,j) (i * ndim + j)

int solve (int n, int ndim, long double *a, long double b[], int pivot[])
{   /* --- begin function solve() --- */

    int    i, j, k, m;
```

```
    long double t;

    if (n == 1)
    {
        /* trivial */
        b[0] /= a[0];
    }
    else
    {
        /* Forward elimination: apply multipliers. */
        for (k = 0; k < n-1; k ++)
        {
            m = pivot[k];
            t = b[m]; b[m] = b[k]; b[k] = t;
            for (i = k+1; i < n; ++i) b[i] += a[AINDEX(i,k)] * t;
        }

        /* Back substitution. */
        for (k = n-1; k >= 0; --k)
        {
            t = b[k];
            for (j = k+1; j < n; ++j) t -= a[AINDEX(k,j)] * b[j];
            b[k] = t / a[AINDEX(k,k)];
        }
    }

    return(0);
}  /* --- end function solve() --- */
```

<DIR>/computational_mathematics/lab_2/matrix_function.h

```
#ifndef LAB2_MATRIX_FUNCTION_H
#define LAB2_MATRIX_FUNCTION_H

namespace mashkin
{
    void multiply(int n, int ndim, long double *a, long double *b, long
double *result);
    void subtract(int n, int ndim, long double *a, long double *b, long
double *result);
    double calculate_the_norm_of_a_matrix(long double *matrix, int ndim);
}
#endif
```

<DIR>/computational_mathematics/lab_2/matrix_function.cpp

```
#include "matrix_function.h"
#include <iomanip>
#include <iostream>

namespace mashkin {
```

```
    void multiply(int n, int ndim, long double *a, long double *b, long
double *result) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                long double temp = 0.0;
                for (int k = 0; k < n; k++) {
                    temp += a[i * ndim + k] * b[k * ndim + j];
                }
                result[i * ndim + j] = temp;
            }
        }
    }

    void subtract(int n, int ndim, long double *a, long double *b, long
double *result)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                result[i * ndim + j] = a[i * ndim + j] - b[i * ndim + j];
            }
        }
    }

    double calculate_the_norm_of_a_matrix(long double *matrix, int ndim)
    {
        double result = 0.0;
        for (int i = 0; i < ndim; i++)
        {
            double var = 0.0;
            for (int j = 0; j < ndim; j++)
            {
                var += matrix[i * ndim + j];
            }
            if (var > result)
            {
                result = var;
            }
        }
        return result;
    }
}
```