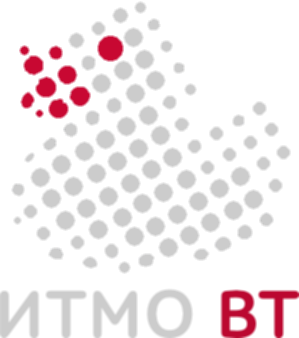
Санкт-Петербургский Национальный Исследовательский Университет Информационных Технологий, Механики и Оптики

ФКТиУ

Лабораторная работа №1

по дисциплине

«Методы и средства программной инженерии»



Выполнил: Студенты группы P3210

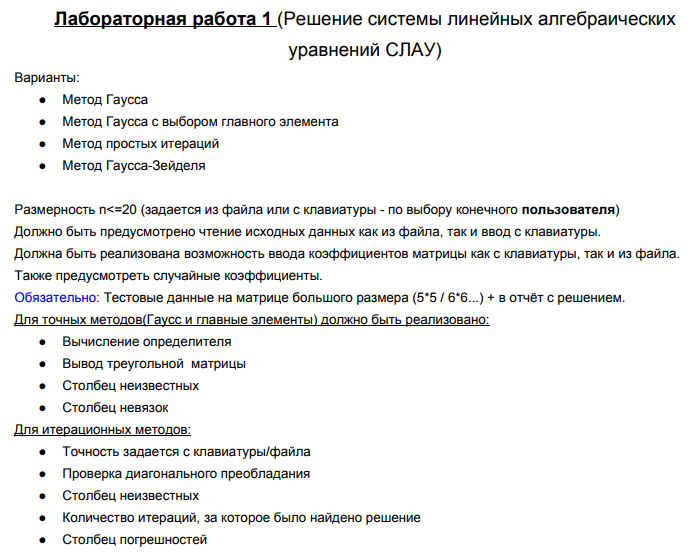
Почикалин Владислав и Чарный Никита

Преподаватель: Письмак Алексей Евгеньевич

Санкт-Петербург

2020 г.

**Задание (Метод простых итераций)**



**Листинг**

using namespace std;

typedef vector<double> d\_vector;

//матрица аргументов

vector <d\_vector> a;

vector <vector <bool>> diagonal;

//x с предыдущей итерации, текущие x, ...

d\_vector prev\_x, x, b;

float epsilon = 0.000000000001;

int iteration = 0;

struct Answer {

d\_vector x, err;

int iterations;

};

void iterate() {

iteration++;

prev\_x = x;

for (size\_t i = 0; i < a.size(); i++) {

double sum = 0;

for (size\_t j = 0; j < a[i].size(); j++) {

sum += a[i][j] \* prev\_x[j];

}

x[i] = sum + b[i];

}

}

bool checkEpsilon() {

if (iteration < 1) return true;

//находим наибольшую разницу между результатами за последнюю итерацию

double cmax = abs(x[0] - prev\_x[0]);

for (size\_t i = 1; i < x.size(); i++) {

cmax = max(cmax, abs(x[i] - prev\_x[i]));

}

//сверяемся с необходимой точностью

return cmax > epsilon;

}

bool checkDiagonal() {

float normal = 0;

for (size\_t j = 0; j < a[0].size(); j++) {

float sum = 0;

for (size\_t i = 0; i < a.size(); i++) {

sum += abs(a[i][j]);

}

normal = max(normal, sum);

}

return normal < 1;

}

void makeDiagonal(vector <d\_vector> &src\_1, d\_vector &src\_2) {

//заполняем bool матрицу, проверяя подходит ли элемент под главную диагональ

//(выполняется ли при этом диагональное преобладание)

int leng = src\_1.size();

vector <vector<bool>> m(leng, vector<bool>(leng, false));

for (size\_t i = 0; i < leng; i++) {

float sum = 0;

for (size\_t j = 0; j < leng; j++) {

sum += abs(src\_1[i][j]);

}

for (size\_t j = 0; j < leng; j++) {

if (2 \* abs(src\_1[i][j]) > sum) {

m[i][j] = true;

}

}

}

//пытаемя перестроить матрицу, если это возможно

vector <d\_vector> patch\_1(m.size());

d\_vector patch\_2(m.size());

for (size\_t i = 0; i < min(m.size(), m[0].size()); i++) {

bool found = 0;

for (size\_t j = 0; j < min(m.size(), m[0].size()); j++) {

if (m[j][i]) {

found = 1;

patch\_1[i] = src\_1[j];

patch\_2[i] = src\_2[j];

break;

}

}

// если не удалось хоть в одном столбце, условие не выполняется

if (!found){

printf("!\_\_(Нет диагонального преобладания)\_\_! \n");

printf(" => результат будет очень не точным :) \n");

return;

}

}

//если удалось достичь диагонального преобладания, вызвращаем измененную матрицу

src\_1 = patch\_1;

src\_2 = patch\_2;

}

void printSLAU(vector <d\_vector> v, d\_vector a)

{

printf("----------------------------\n");

for (int i = 0; i < v.size(); i++) {

for (int j = 0; j < v[i].size(); j++) {

printf("%.2lf \* x%i + ", v[i][j], j + 1);

}

printf("= %.2lf \n", a[i]);

}

printf("----------------------------\n");

}

Answer solve(vector <d\_vector> src\_1, d\_vector src\_2) {

//проверяем на диагональное преобладание (условие сходимости)

//если это возможно, пытаемся перестроить матрицу

makeDiagonal(src\_1, src\_2);

printSLAU(src\_1, src\_2);

a.resize(src\_1.size());

double normal\_a = 0, normal\_b = 0;

for (size\_t i = 0; i < src\_1.size(); i++) {

a[i].resize(src\_1[i].size());

for (size\_t j = 0; j < src\_1[i].size(); j++) {

if (i == j) {

a[i][i] = 0;

}

else {

a[i][j] = -src\_1[i][j] / src\_1[i][i];

}

}

}

b.resize(src\_2.size());

for (size\_t i = 0; i < src\_2.size(); i++) {

b[i] = src\_2[i] / src\_1[i][i];

}

printSLAU(a, b);

//первоначальное приближение

x = b;

//основной цикл

while (checkDiagonal() && checkEpsilon()) {

iterate();

}

//формируем ответ

Answer a;

a.x = x;

a.iterations = iteration + 1;

a.err.assign(x.size(), abs(x[0] - prev\_x[0]));

return a;

}

int main(int argc, char \*\*argv) {

system("chcp 1251");

bool flag = 1;

char checkRandom;

srand(time(0));

cout << "Желаете ли задать рандомные аргументы? (введите '+') - ";

cin >> checkRandom;

if (checkRandom == '+')

printf("Аргументы будут заданы случайным образом \n");

if (argc == 2 && checkRandom != '+'){

flag = 0;

if (!freopen(argv[1], "r", stdin))

printf("Невозможно открыть файл");

else

printf("Аргументы будут считаны из файла - %s \n", argv[1]);

}

if (flag) printf("Введите точность:\n");

cin >> epsilon;

int n;

if (flag) printf("Введите размер матрицы:\n");

cin >> n;

vector <d\_vector> src\_1(n);

d\_vector src\_2(n);

if (flag && checkRandom != '+'){

printf("Введите коэффициенты перед неизвестными и свободные члены.");

printf("Требуется ввести %i строк по %i значений, разделенных пробелом : \n", n, n+1);

}

if (checkRandom != '+')

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

src\_1[i].resize(n);

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

cin >> src\_1[i][j];

}

cin >> src\_2[i];

} else {

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

src\_1[i].resize(n);

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

src\_1[i][j] = (float)(rand() % 10000) / 100;

}

src\_2[i] = (float)(rand() % 10000) / 100;

}

}

Answer ans = solve(src\_1, src\_2);

printf("Решение (Кол-во итераций: %i):\n", ans.iterations);

for (size\_t i = 0; i < ans.x.size(); i++) {

printf("x%lu = %lf ± %lf\n", i + 1, ans.x[i], ans.err[i]);

}

\_getch();

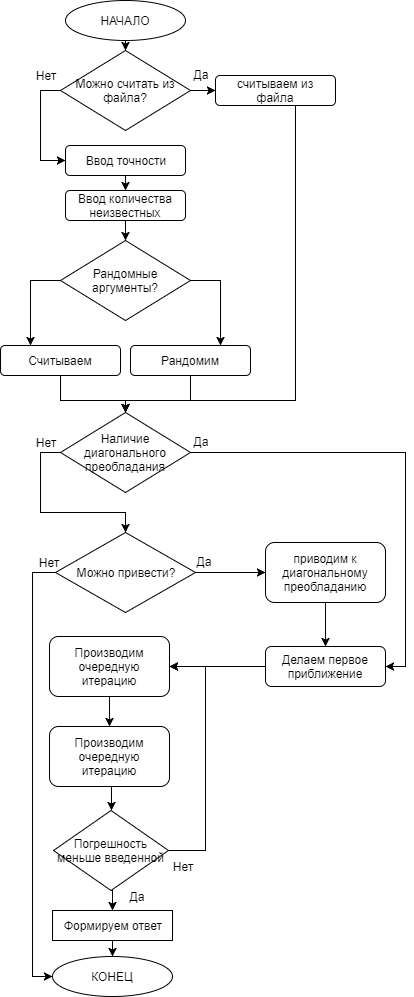
return 0;

}

**Тестовые данные**

|  |  |
| --- | --- |
| **Тест 1)**  0.000000000001  3  10 2 -1 5  -2 -6 -1 24.42  1 -3 12 46  **Решение программы:**  Решение (Кол-во итераций: 22):  x1 = 1.753939 ± 0.000000  x2 = -5.058408 ± 0.000000  x3 = 2.422570 ± 0.000000  **Правильное решение:** | **Тест 2)**  0.0000000000000001  5  100 1 2 3 4 5  6 200 7 8 9 10  11 12 300 13 14 15  16 17 18 400 19 20  21 22 23 24 500 25  **Решение программы:**  Решение (Кол-во итераций: 20):  x1 = 0.045751 ± 0.000000  x2 = 0.043537 ± 0.000000  x3 = 0.042777 ± 0.000000  x4 = 0.042392 ± 0.000000  x5 = 0.042160 ± 0.000000  **Правильное решение:** |
| **Тест 3)**  0.0000000000000001  5  100 1 2 3 4 5  11 12 300 13 14 15  6 200 7 8 9 10  21 22 23 24 500 25  16 17 18 400 19 20  **Решение программы:**  Решение (Кол-во итераций: 20):  x1 = 0.045751 ± 0.000000  x2 = 0.043537 ± 0.000000  x3 = 0.042777 ± 0.000000  x4 = 0.042392 ± 0.000000  x5 = 0.042160 ± 0.000000  **Правильное решение:** | **Тест 4)**  0.000001  6  15 1 -6 2 1 4 0  1 40 3 -8 4 3 2  2 7 62 4 -7 3 1  1 3 1 48 3 -5 1  4 7 5 5 99 6 4  1 1 1 1 1 -7 -8  **Решение программы:**  Решение (Кол-во итераций: 11):  x1 = -0.330164 ± 0.000000  x2 = 0.008909 ± 0.000000  x3 = -0.039430 ± 0.000000  x4 = 0.144736 ± 0.000000  x5 = -0.019431 ± 0.000000  x6 = 1.109232 ± 0.000000  **Правильное решение:** |

**Блок-схема**

****

**Вывод**

В ходе выполнения работы был изучен метод решения СЛАУ - простыми итерациями. А также разобраны методы Гауса, Гауса с выбором главного и метод Гауса-Зейделя.