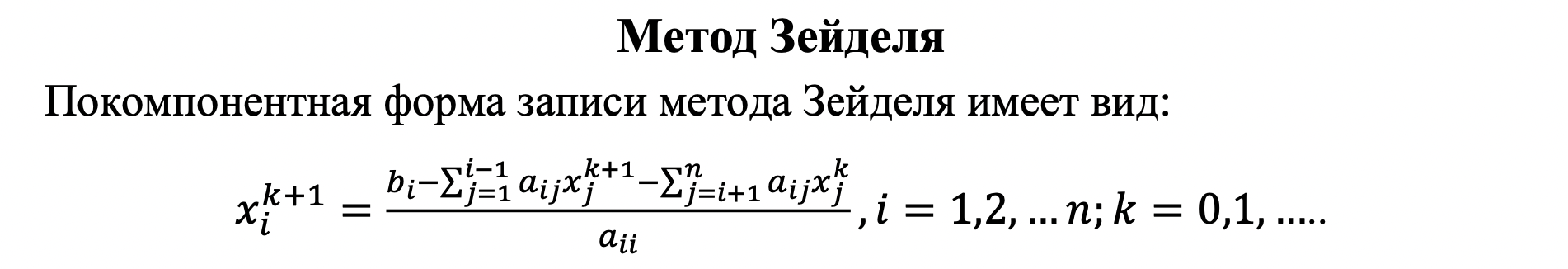
Work 15 Кириленко Константин 351





Условие сходимости:

Чтобы методы сходились, требуется выполнение некоторых условий. Для Якоби должно быть диагональное преобладание. Для Зейделя - спектральный радиус меньше единицы, диагонального преобладание так же достаточно.

// Function for random initialization of the matrix

// and the vector elements

// Function for random initialization of the matrix

// and the vector elements

void RandomDataInitialization(long double\*\* pMatrix, long double\* pVector,

int Size) {

long double s;

int i, j; // Loop variables

srand(unsigned(clock()));

for (i = 0; i < Size; i++) {

pVector[i] = rand() / double(1000);

for (j = 0; j < Size; j++) {

pMatrix[i ][ j] = rand() / double(1000);

}

}

for (i = 0; i < Size; i++) {

s = 0;

pVector[i] = rand() / double(1000);

for (j = 0; j < Size; j++) {

if (i != j)

s += pMatrix[i ][j];

}

pMatrix[i ][i] = rand() / double(1000) + s;

}

}

#define e 0.00000000001

void ZeidelResultCalculation(long double\*\* pMatrix, long double\* pVector,

long double\* pResult, int Size, int NProc, int ProcId) {

double\* g = new double[Size];

double\* d = new double[Size];

int t = 0;

//#pragma omp parallel for

for (int i = 0; i < Size; i++) { pResult[i] = 0; g[i] = 0; }

bool flag = false;

double sum;

double sum1;

do {

//cout << t << endl;

//t++;

flag = false;

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

g[i] = pVector[i];

sum = 0;

sum1 = 0;

int n1 = Size / NProc;

int n2 = (ProcId + 1) \* n1;

if (NProc == ProcId + 1) {

n2 = Size;

}

int st = ProcId \* n1;

for (int j = st; j < n2; j++) {

if (i != j) {

sum += (pMatrix[i][j] \* g[j]);

}

}

MPI\_Reduce(&sum, &sum1, 1, MPI\_DOUBLE, MPI\_SUM, 0, MPI\_COMM\_WORLD);

MPI\_Bcast(&sum1, 1, MPI\_DOUBLE, 0, MPI\_COMM\_WORLD);

g[i] -= sum1;

g[i] = (1.0 \* g[i]) / pMatrix[i][ i];

if (fabs(g[i] - pResult[i]) >= e) flag = true;

pResult[i] = g[i];

}

} while (flag);

delete[] g;

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Порядок системы | Последовательный алгоритм | Параллельный алгоритм | |
| Время | Ускорение |
| 1 | 10 | 0.000546 | 0.003389 | 0.1611 |
| 2 | 100 | 0.003050 | 0.010387 | 0.2936 |
| 3 | 500 | 0.098860 | 0.026965 | 3.66623 |
| 4 | 1000 | 0.171142 | 0.116945 | 1.4634008 |
| 5 | 1500 | 0.432129 | 0.171327 | 2.522247 |
| 6 | 2000 | 0.642627 | 0.344131 | 1.87639 |
| 7 | 2500 | 1.129990 | 0.567905 | 1.98975 |
| 8 | 3000 | 1.429655 | 0.675155 | 2.1175211 |

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Порядок системы | Ускорение алгоритма Гаусса | Ускорение алгоритма  Зейделя |
| 1 | 10 | 0.39996 | 0.1611 |
| 2 | 100 | 1.193 | 0.2936 |
| 3 | 500 | 7.971 | 3.66623 |
| 4 | 1000 | 2.6894 | 1.4634008 |
| 5 | 1500 | 2.9523 | 2.522247 |
| 6 | 2000 | 3.0113 | 1.87639 |
| 7 | 2500 | 2.816 | 1.98975 |
| 8 | 3000 | 2.8422 | 2.1175211 |

Ускорение алгоритма Зейделя немного меньше. Максимум также достигается на средних значениях размерности. Однако общая скорость выполнения и точность намного выше.