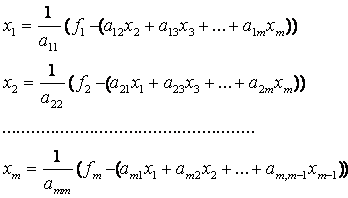
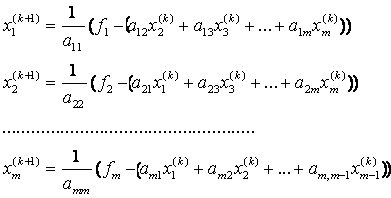
Санкт-Петербургский национальный исследовательский университет  
 информационных технологий, механики и оптики  
Кафедра информатики и прикладной математики

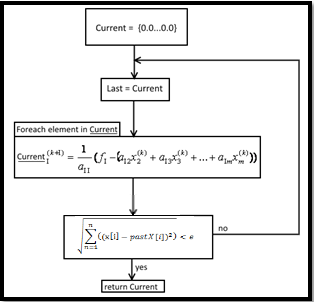
Вычислительная математика  
Лабораторная работа №1 **Метод Гаусса-Зейделя**

Выполнил: Шкаруба Н.Е.  
Проверил: Петрова М.  
группа: P3218  
год: 2015

**Метод Гаусса-Зейделя**

1. **Описание метода**

http://pers.narod.ru/study/methods/2.files/image037.gifМетод используется для решения систем линейных уравнений, которая в матричном виде записывается как: aX = b, где a – матрица коэффициентов, б = матрица результатовПредположим, что диагональные элементы матриц A исходной системы не равны 0 (aii ≠ 0, i = 1, 2, …, n). Разрешим первое уравнение системы относительно x1, второе относительно x2 и т.д. Получим следующую эквивалентную систему, записанную в скалярном виде (1)  
(1)(2)  
Теперь, задав нулевое приближение ( Xi(0) = 0) , по рекуррентным соотношениям можем выполнять итерационный процесс, до необходимой точности, а именно в виде (2)   
Условие сходимости в математическом виде:  
и хотя бы для одного i неравенство строгое. Другими словами, модули диагональных коэффициентов в каждом уравнении системы больше суммы модулей недиагональных коэффициентов (свободные члены не рассматриваются).

1. **Блок схема**  
   
2. **Исходный код**

bool dDominance(float\*\* arr, size\_t size) {

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

double sum = 0;

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

if (j != i)

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

if (abs(arr[i][i]) > sum)

return true;

}

return false;

}

bool converge(float\* first, float\* second, size\_t size, float precision) {

float norm = 0;

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

norm += (first[i] - second[i]) \* (first[i] - second[i]);

return (sqrt(norm) < precision);

}

float\* GaussSeidel(float\*\* a, float\* b, size\_t size, float precision) {

// aX = b

float\* X = new float[size];

float\* pastX = new float[size];

do {

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

pastX[i] = X[i];

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

double var = 0;

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

var += a[i][j] \* X[j];

for (int j = i + 1; j < size; j++)

var += a[i][j] \* pastX[j];

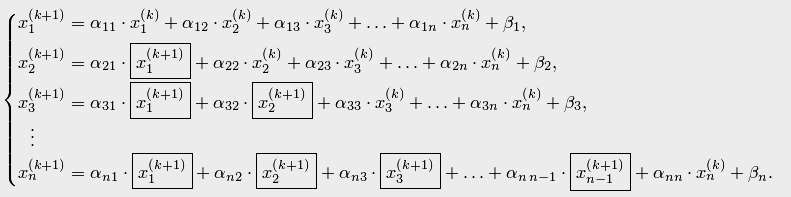
X[i] = (b[i] - var) / a[i][i];

}

} while (!converge(X, pastX, size, precision));

return X;

}

**Суть**: 

1. **Пример работы программы**:  
   a = {{8, 4, 2}, {3, 5, 1}, {3, -2, 10}};  
   b = {10, 5, 4}  
   precision = 0.001  
   result **X** = {1.037694557436856, 0.34617439021949198, 0.15800937201227178}
2. **Вывод**: