Contents


```
% Задание данных
clc
clear
format short                         % задаём формат на вывод
M = readmatrix('basic_matrix2.txt')  % считываем матрицу из файла
x_real = readmatrix('X_real.txt');   % считываем матрицу из файла
[n,m] = size(M);                     % узнаём размер матрицы для реализации алгоритма

A = M(1:n,1:n);
b = M(1:n,end);
```

```
M =

   1.0e+03 *

  -0.0524         0   -0.0006    0.0047   -1.3092
   0.0001    0.0324    0.0091    0.0005    0.2241
        0    0.0059   -0.1750    0.0024    0.0974
  -0.0050   -0.0024    0.0019   -0.0762   -7.8006
```

Реализация метода Якоби

```
x0 = zeros(1,n)';
x = zeros(1,n)';
beta = zeros(n,n);
c = zeros(1,n)';
iter = 0;

for i = 1:n
    c(i) = b(i)/A(i,i);
    for j = 1:n
        if i ~= j
            beta(i,j) = -A(i,j)/A(i,i);
        else
            beta(i,j) = 0;
        end
    end
end

beta
c

if FirstNormMatrix(beta) >= 1
    error('Не выполнено достаточное условие применимости метода');
end

x0 = [134120;25212;74050;-252120];    % вектор первого приближения
epsulon = 0.001;        % задание точности найденного решения
epsilon1 = ((1-FirstNormMatrix(beta))/FirstNormMatrix(beta))*epsulon;
```

```
flag = true;
while flag
    iter = iter + 1;
    x = beta*x0+c;
    if FirstNormVec(x-x0) < epsilon1
        flag = false;
    else
        x0 = x;
    end
end

x
iter

delta_x_Y = FirstNormVec(x_real - x);      % абсолютная погрешность для первой нормы
sigma_x_Y = delta_x_Y/FirstNormVec(x)        % относительная погрешность для первой нормы
```

```
beta =

         0         0   -0.0109    0.0903
   -0.0037         0   -0.2793   -0.0151
         0    0.0336         0    0.0139
   -0.0657   -0.0319    0.0245         0


c =

   24.9842
    6.9176
   -0.5566
  102.3703
```

Реализация метода Зейделя

```
B1 = beta;  %нижняя треугольная матрица
B2 = beta;  %верхняя треугольная матрица
x0 = zeros(1,n)';
x = zeros(1,n)';
iter = 0;

if FirstNormMatrix(beta) >= 1
    error('Не выполнено достаточное условие применимости метода');
end

for i = 1:n
    for j = i:n
        B1(i,j) = 0;
    end
end

for i = 1:n
    for j = 1:i
        B2(i,j) = 0;
    end
end

beta
B1
B2
```

```
x0 = [20;12;50;-20]; % вектор первого приближения
epsulon = 0.001;       % задание точности найденного решения
epsilon2 = ((1-FirstNormMatrix(B1))/FirstNormMatrix(B2))*epsulon;

flag = true;
while flag
    iter = iter + 1;
    for i = 1:n
        x(i) = B1(i,:)*x+B2(i,:)*x0 + c(i);
    end
    if FirstNormVec(x-x0) < epsilon2
        flag = false;
    else
        x0 = x;
    end
end

x
iter

delta_x_Z = FirstNormVec(x_real - x);     % абсолютная погрешность для первой нормы
sigma_x_Z = delta_x_Z/FirstNormVec(x)       % относительная погрешность для первой нормы
```

```
beta =

         0         0   -0.0109    0.0903
   -0.0037         0   -0.2793   -0.0151
         0    0.0336         0    0.0139
   -0.0657   -0.0319    0.0245         0


B1 =

         0         0         0         0
   -0.0037         0         0         0
         0    0.0336         0         0
   -0.0657   -0.0319    0.0245         0


B2 =

         0         0   -0.0109    0.0903
         0         0   -0.2793   -0.0151
         0         0         0    0.0139
         0         0         0         0


x =

   34.0000
    5.0000
    1.0000
  100.0000


iter =

     5


sigma_x_Z =

   3.2101e-08
```

Реализация метода Гаусса

```
x = A\b

delta_x_G = FirstNormVec(x_real - x);      % абсолютная погрешность для первой нормы
sigma_x_G = delta_x_G/FirstNormVec(x)        % относительная погрешность для первой нормы
```

```
x =

   34.0000
    5.0000
    1.0000
  100.0000


sigma_x_G =

   1.0944e-16
```

Оценка выполнения методов

```
k = 10000;
timeYakobi = zeros(1,k)';
timeZeydelya = zeros(1,k)';
timeGauss = zeros(1,k)';

for z = 1:k
    x0 = [5;5;5;5];
    %x0 = [20;12;50;-20];
    x = zeros(1,n)';

    tic
    flag = true;
    while flag
        x = beta*x0+c;
        if FirstNormVec(x-x0) < epsilon1
            flag = false;
        else
            x0 = x;
        end
    end
    timeYakobi(z) = toc;

    x0 = [5;5;5;5];
    %x0 = [20;12;50;-20];
    x = zeros(1,n)';

    tic
    flag = true;
    while flag
        for i = 1:n
            x(i) = B1(i,:)*x+B2(i,:)*x0 + c(i);
        end
        if FirstNormVec(x-x0) < epsilon2
            flag = false;
        else
            x0 = x;
        end
    end
```

```
    timeZeydelya(z) = toc;

    x0 = [5;5;5;5];
    %x0 = [20;12;50;-20];
    x = zeros(1,n)';

    tic
    x = A\b;
    timeGauss(z) = toc;
end

timeYakobiMean = mean(timeYakobi)
timeZeydelyaMean = mean(timeZeydelya)
timeGaussMean =  mean(timeGauss)

X = 1;
bar(X, [timeYakobiMean,timeZeydelyaMean,timeGaussMean]);
legend('Результаты для метода Якоби','Результаты для метода Зейделя',...
    'Результаты для метода Гаусса')   % устанавливаем легенду
grid on
title('Экспериментальная оценка времени работы алгоритмов')
xlabel('Методы')
ylabel('Время выполнения')
axis auto
```

Функции для расчёта норм

```
function norm = FirstNormVec(x) %функция расчёта первой нормы
    sum = 0;
    [n,m] = size(x);
    for i = 1:n
        sum = sum + abs(x(i));
    end
    norm = sum;
end

function norm = InfNormVec(x)   %функция расчёта инфинити нормы
    count = 0;
    [n,m] = size(x);
    for i = 1:n
        if count < abs(x(i))
           count = abs(x(i));
        end
    end
    norm = count;
end

function norm = FirstNormMatrix(X) %функция расчёта первой нормы матрицы
    max = 0;
    [n,m] = size(X);
    for i = 1:m
        if FirstNormVec(X(1:n,i)) > max
            max = FirstNormVec(X(1:n,m));
        end
    end
    norm = max;
end

function norm = InfNormMatrix(X) %функция расчёта инфинити нормы матрицы
    max = 0;
    [n,m] = size(X);
    for i = 1:n
        if FirstNormVec(X(i,1:m)') > max
            max = FirstNormVec(X(i,1:m)');
```

```
        end
    end
    norm = max;
end
```

```
x =

   33.9999
    5.0000
    0.9999
  100.0002


iter =

     9


sigma_x_Y =

   2.2898e-06
```

Published with MATLAB® R2020a