Contents

Задание данных

```
clc
clear
format long                          % задаём формат на вывод
A = readmatrix('basic_matrix.txt')   % считываем матрицу из файла
[n,m] = size(A);                     % узнаём размер матрицы для реализации алгоритма

fileID = fopen('basic_data.txt');    % открываем файл
a = str2num(fgetl(fileID))  % считываем 1-ую и преобразуем строку из файла в числа
b = str2num(fgetl(fileID))  % считываем 2-ую и преобразуем строку из файла в числа
c = str2num(fgetl(fileID))  % считываем 3-ую и преобразуем строку из файла в числа
d = str2num(fgetl(fileID))  % считываем 4-ую и преобразуем строку из файла в числа
fclose(fileID);
[m,n] = size(b);            % n - число элементов вектора b
```

```
A =

    50     1     0     0     0     0    10
    -1    90    -1     0     0     0    -9
     0     1   125    -1     0     0    12
     0     0     1   110     0     0    11
     0     0     0    -1    85     1     9
     0     0     0     0     1    70     8


a =

    -1     1     1    -1     1


b =

    50    90   125   110    85    70


c =

     1     1    -1     0     1


d =

    10    -9    12    11     9     8
```

Провека достаточного условия применимости метода

```
for i = 1:n
    if i == 1
        if abs(b(i)) < abs(c(i))% Индивидуальное условие для первой строки
            error('Не выполнено достаточное условие применимости метода');
```

```
        end
    end

    if (1 < i && i < n)
        if abs(b(i)) < abs(a(i-1)) + abs(c(i))
            error('Не выполнено достаточное условие применимости метода');
        end
    end

    if i == n
        if abs(b(i)) < abs(a(i-1))% Индивидуальное условие для последней строки
            error('Не выполнено достаточное условие применимости метода');
        end
    end
end
```

Прямая прогонка

```
alfa = zeros(1,n-1); % Изначально заполняем нулями вектор
                     % коэффициентов альфа с заданным размером (n-1)
beta = zeros(1,n);   % Изначально заполняем нулями вектор
                     % коэффициентов бета с заданным размером (n)
for i = 1:n
    if i == 1        % Индивидуальный расчёт для коэффициентов первой строки
        gamma = b(i);
        alfa(i) = -c(i)/gamma;
        beta(i) = d(i)/gamma;
    end

    if (1 < i && i < n)
        gamma = b(i)+a(i-1)*alfa(i-1);
        alfa(i) = -c(i)/gamma;
        beta(i) = (d(i)-a(i-1)*beta(i-1))/gamma;
    end

    if i == n        % Индивидуальный расчёт для коэффициентов последней строки
        gamma = b(i)+a(i-1)*alfa(i-1);
        beta(i) = (d(i)-a(i-1)*beta(i-1))/gamma;
    end
end
```

Обратная прогонка

```
x = zeros(1,n);
for i = n:-1:1
    if i == n % Индивидуальный расчёт вектора решений для последней строки
        x(n) = beta(n);
    end

    if i < n
        x(i) = beta(i) + alfa(i)*x(i+1);
    end
end
```

Вычисление правой части на основе полученного решения

```
d_calculate = zeros(1,n);
for i = 1:n
    if i == 1 % Индивидуальный расчёт вектора правой части для первой строки
        d_calculate(i) = b(i)*x(1)+c(i)*x(2);
    end

    if (1 < i && i < n)
        d_calculate(i) = a(i-1)*x(i-1)+b(i)*x(i)+c(i)*x(i+1);
    end

    if i == n % Индивидуальный расчёт вектора правой части для последней строки
        d_calculate(i) = a(i-1)*x(i-1)+b(i)*x(i);
```

```
    end
end
```

Вектор невязки и его первая и инфинити норма

```
r = d_calculate - d      % вычисляем вектор невязки правых частей
r_norm_1 = FirstNorm(r)  % первая норма для вектора невязки
r_norm_inf = InfNorm(r)  % инфинити норма для вектора невязки
```

```
r =

   1.0e-14 *

  Columns 1 through 3

                  0                   0   0.177635683940025

  Columns 4 through 6

                  0  -0.177635683940025                   0
```

Экспериментальная оценка устойчивости

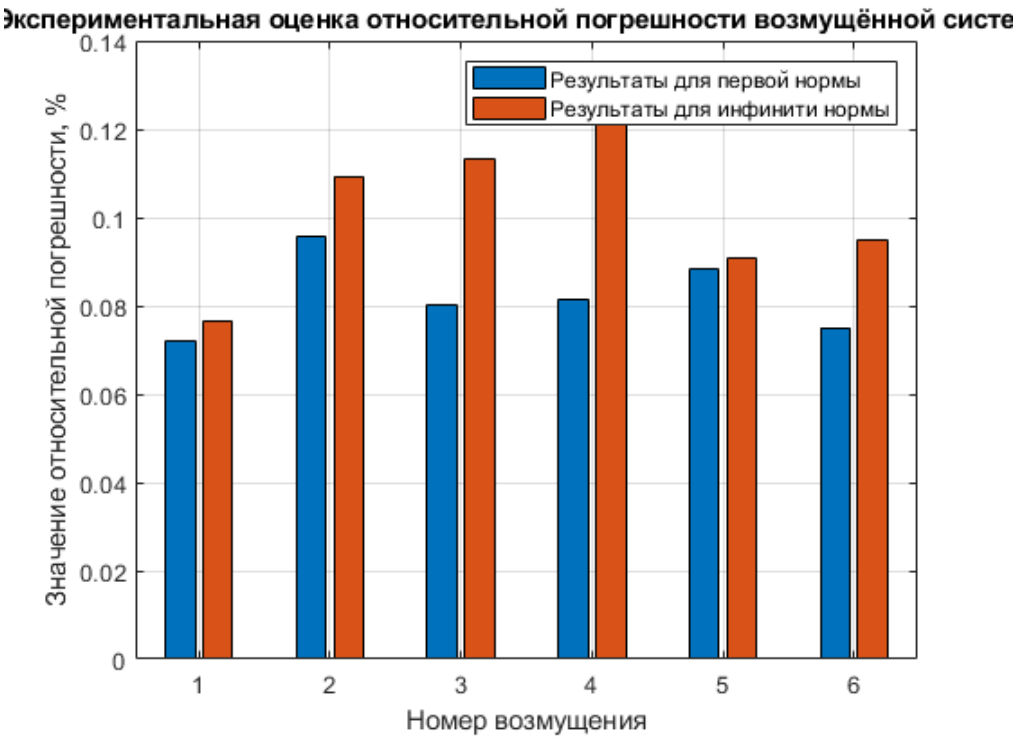
```
sigma = 0.01;            % задаём сигму для ошибки
X = (1:n)';

FirstEx = x-Errors(sigma);
SecondEx = x-Errors(sigma);
ThirdEx = x-Errors(sigma);
FourthEx = x-Errors(sigma);
FifthEx = x-Errors(sigma);
SixthEx = x-Errors(sigma);

SigmaFirstNorm = [FirstNorm(FirstEx)*100/FirstNorm(x),FirstNorm(SecondEx)*100/FirstNorm(x),...
    FirstNorm(ThirdEx)*100/FirstNorm(x),FirstNorm(FourthEx)*100/FirstNorm(x),...
    FirstNorm(FifthEx)*100/FirstNorm(x),FirstNorm(SixthEx)*100/FirstNorm(x)]

SigmaInfNorm = [InfNorm(FirstEx)*100/InfNorm(x),InfNorm(SecondEx)*100/InfNorm(x),InfNorm(ThirdEx)*100/InfNorm(x),...
    InfNorm(FourthEx)*100/InfNorm(x),InfNorm(FifthEx)*100/InfNorm(x),InfNorm(SixthEx)*100/InfNorm(x)]

bar(X, [SigmaFirstNorm',SigmaInfNorm'])                                  % строим столбчатую гистограмму
legend('Результаты для первой нормы','Результаты для инфинити нормы')    % устанавливаем легенду
grid on
title('Экспериментальная оценка относительной погрешности возмущённой системы')
xlabel('Номер возмущения')
ylabel('Значение относительной погрешности, %')
axis auto
set(gca, 'XTick',-1:1:n) % устанавливаем шаг сетки для оси OX
```

Функции для расчёта норм

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

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

```
            count = abs(x(i));
        end
    end
    norm = count;
end
```

```
r_norm_1 =

     3.552713678800501e-15


r_norm_inf =

     1.776356839400250e-15



SigmaFirstNorm =

  Columns 1 through 3

   0.071850083029202   0.095587981071800   0.080076620380946

  Columns 4 through 6

   0.081338533969040   0.088233213827857   0.075043905783388

SigmaInfNorm =

  Columns 1 through 3

   0.076679703412432   0.109345173603413   0.113426439299720

  Columns 4 through 6

   0.134056225061677   0.090792618784417   0.095001360617956
```



Функция для оценки устойчивости

```
function x_error = Errors(sigma)

    fileID = fopen('basic_data.txt');    % открываем файл
    a = str2num(fgetl(fileID));          % считываем 1-ую и преобразуем строку из файла в числа
    b = str2num(fgetl(fileID));          % считываем 2-ую и преобразуем строку из файла в числа
    c = str2num(fgetl(fileID));          % считываем 3-ую и преобразуем строку из файла в числа
    d = str2num(fgetl(fileID));          % считываем 4-ую и преобразуем строку из файла в числа
    fclose(fileID);
    [m,n] = size(b);                     % n - число элементов вектора b

    mu = 0;
    for i = 1:n
        d(i) = d(i) + random('Normal',mu,sigma);
    end

    alfa = zeros(1,n-1);
    beta = zeros(1,n);
    for i = 1:n
        if i == 1
            gamma = b(i);
            alfa(i) = -c(i)/gamma;
            beta(i) = d(i)/gamma;
        end

        if (1 < i && i < n)
            gamma = b(i)+a(i-1)*alfa(i-1);
            alfa(i) = -c(i)/gamma;
            beta(i) = (d(i)-a(i-1)*beta(i-1))/gamma;
        end

        if i == n
            gamma = b(i)+a(i-1)*alfa(i-1);
            beta(i) = (d(i)-a(i-1)*beta(i-1))/gamma;
        end
    end

    x_error = zeros(1,n);
    for i = n:-1:1
        if i == n
            x_error(n) = beta(n);
        end

        if i < n
            x_error(i) = beta(i) + alfa(i)*x_error(i+1);
        end
    end

end
```

Published with MATLAB® R2020a