Итерационные методы решения линейных систем. Вариант 11

Содержание

Решение системы методом Гаусса

Ввод данных

Введем матрицу системы

```
A = [6.687233,0.80267,-2.06459;
    0.80267,5.07816,0.48037;
    -2.06459,0.48037,4.02934]
```

```
A = 3×3
    6.6872    0.8027   -2.0646
    0.8027    5.0782    0.4804
   -2.0646    0.4804    4.0293
```

Введем столбец свободных членов

```
b = [0;1;0]
```

```
b = 3×1
     0
     1
     0
```

Найдем точное решение системы методом Гаусса, используя метод из стандарта языка Matlab

```
x_acc = A\b;
x_acc
```

```
x_acc = 3×1
   -0.0386
    0.2072
```

```
   -0.0445
```

Преобразование исходной системы

Выполним преобразование исходной системы к виду $x = H_D + g_D$, где $H_D = E - D^{-1}A,\ g_D = D^{-1}b.$ Здесь $D = (d_{ij})_{i,j=1}^n,\ d_{ii} = a_{ii},\ d_{ij} = 0\ \ при\ \ i \neq j$.

Найдем матрицу D

```
D = eye(size(A,1))
```

```
D = 3×3
     1     0     0
     0     1     0
     0     0     1
```

```
for i = 1:size(A,1)
    D(i,i)=A(i,i)
end
```

```
D = 3×3
    6.6872         0         0
         0    1.0000         0
         0         0    1.0000
D = 3×3
    6.6872         0         0
         0    5.0782         0
         0         0    1.0000
D = 3×3
    6.6872         0         0
         0    5.0782         0
         0         0    4.0293
```

```
invD = D^(-1)
```

```
invD = 3×3
    0.1495         0         0
         0    0.1969         0
         0         0    0.2482
```

Найдем $H_D,\ g_D$

```
E = eye(size(A,1))
```

```
E = 3×3
     1     0     0
     0     1     0
     0     0     1
```

```
H_D = E - invD*A
```

```
H_D = 3×3
    0.0000   -0.1200    0.3087
   -0.1581         0   -0.0946
    0.5124   -0.1192         0
```

```
g_D = invD*b
```

```
g_D = 3×1
         0
    0.1969
         0
```

Вычислим $||H_D||_\infty$

```
normOfH_D = norm(H_D,"inf")
```

```
normOfH_D = 0.6316
```

Априорная оценка k

Найдем априорную оценку того k, при котором $||x^* - x^k||_\infty < \epsilon,\ \epsilon = 0,0001$

```
eps = 0.0001
```

```
eps = 1.0000e-04
```

```
normOfg_D = norm(g_D,"inf")
```

```
normOfg_D = 0.1969
```

```
k_apr = round(log(eps*(1-normOfH_D)/normOfg_D)/log(normOfH_D)-0.499999)
```

```
k_apr = 18
```

Нахождение решения методом простой итерации

Алгоритм

В качестве начального приближения выберем вектор, все компоненты которого равны 0. Далее будем использовать формулу $x^{(k+1)} = H_D x_k + g_D$, пока условие $||x^{(k)} - x^{k-1}|| < \epsilon$ не будет выполнено.

Реализация алгоритма

Реализация находится в файле simpleIter.m

Результат работы

```
[x_curr,x_prev,k_real] = simpleIter(H_D,g_D,eps)
```

```
x_curr = 3×1
   -0.0386
    0.2072
   -0.0444
x_prev = 3×1
   -0.0385
    0.2072
   -0.0444
k_real = 10
```

Сравним априорное и фактическое значения k

```
k_real
```

```
k_real = 10
```

```
k_apr
```

```
k_apr = 18
```

Априорные и апостериорные оценки погрешности

Априорная оценка погрешности

```
xk_apr_err = normOfH_D^k_real*normOfg_D/(1-normOfH_D)
```

```
xk_apr_err = 0.0054
```

Апостериорная оценка погрешности

```
xk_apost_err = normOfH_D/(1-normOfH_D)*norm(x_curr-x_prev,"inf")
```

```
xk_apost_err = 7.9576e-05
```

Фактическая погрешность

```
xk_fact_err = norm(x_acc-x_curr)
```

```
xk_fact_err = 5.7347e-05
```

Уточнение приближения по Люстернику

Найдем спектральный радиус матрицы H

```
spectOfH_D = max(abs(eig(H_D)))
```

```
spectOfH_D = 0.4622
```

Уточним решение по Люстернику:

```
x_clar = x_prev + 1/(1-spectOfH_D) * (x_curr-x_prev)
```

```
x_clar = 3×1
   -0.0386
    0.2072
   -0.0445
```

Фактическая погрешность уточненного решения:

```
x_clar_err = norm(x_acc-x_clar,"inf")
```

```
x_clar_err = 2.7857e-06
```

Поиск решения методом Зейделя

Алгоритм

Метод Зейделя для системы $x = Hx + g$ совпадает с методом простой итерации для системы $x = H_{seid}x + g_{seid}$, где $H_{seid} = (E - H_L)^{-1}H_R,\ g_{seid} = (E - H_L)^{-1}g_D,\ H_D = H_L + H_R$

Реализация

Представим матрицу H_D в виде $H_D = H_L + H_R$, используя функцию seidelPreapration.m

```
[H_L, H_R] = seidelPreparation(H_D)
```

```
H_L = 3×3
         0         0         0
   -0.1581         0         0
    0.5124   -0.1192         0
H_R = 3×3
    0.0000   -0.1200    0.3087
         0         0   -0.0946
         0         0         0
```

Сведем задачу к методу простой итерации

```
H_seid = (E-H_L)^(-1)*H_R
```

```
H_seid = 3×3
    0.0000   -0.1200    0.3087
   -0.0000    0.0190   -0.1434
    0.0000   -0.0638    0.1753
```

```
g_seid = (E-H_L)^(-1)*g_D
```

```
g_seid = 3×1
         0
    0.1969
   -0.0235
```

Воспользуемся написанной ранее функцией

```
[x_curr_seid,x_prev_seid,k_real_seid] = simpleIter(H_seid,g_seid,eps)
```

```
x_curr_seid = 3×1
   -0.0386
    0.2072
   -0.0445
x_prev_seid = 3×1
   -0.0385
    0.2072
   -0.0444
k_real_seid = 6
```

Фактическая погрешность приближения

```
x_curr_seid_err = norm(x_acc-x_curr_seid,"inf")
```

```
x_curr_seid_err = 1.8505e-05
```

Уточнение приближения по Люстернику

Найдем спектральный радиус H_{seid}

```
spectOfH_seid = max(abs(eig(H_seid)))
```

```
spectOfH_seid = 0.2206
```

Уточним приближение по Люстернику

```
x_clar_seid = x_prev_seid + 1/(1-spectOfH_seid) * (x_curr_seid-x_prev_seid)
```

```
x_clar_seid = 3×1
   -0.0386
    0.2072
   -0.0445
```

Фактическая погрешность приближения:

```
x_clar_seid_err = norm(x_acc-x_clar_seid,"inf")
```

```
x_clar_seid_err = 5.7925e-10
```

Решение методом верхней релаксации

Алгоритм

Для системы $x = H_D x + g_D$ расчетная формула имеет следующий вид:

$$x_i^{(k)} = x_i^{(k-1)} + q\left(\sum_{j=1}^{i-1} h_{ij} x_j^{(k)} + \sum_{j=i+1}^{n} h_{ij} x_j^{(k-1)} - x_i^{(k-1)} + g_i\right) \text{ , где } q = \frac{2}{1 + \sqrt{1 - \rho^2(H_D)}}$$

Реализация

Реализация находится в функции relaxationMethod.m

Результат работы

```
[x_curr_rel,x_prev_rel,k_real_rel] = relaxationMethod(H_D,g_D,eps)
```

```
x_curr_rel = 3×1
   -0.0386
    0.2072
   -0.0445
x_prev_rel = 3×1
   -0.0386
    0.2072
   -0.0445
k_real_rel = 5
```

Найдем фактическую погрешность:

```
x_rel_err = norm(x_acc - x_curr_rel,"inf")
```

```
x_rel_err = 4.9055e-06
```

Вывод

Сравним результаты работы описанных методов

```
method = {'Метод простой итерации'; 'Метод Зейделя'; 'Метод верхней релаксации'}
```

```
method = 3×1 cell
'Метод простой итерации'
'Метод Зейделя'
'Метод верхней релаксации'
```

```
steps = [k_real;k_real_seid;k_real_rel]
```

```
steps = 3×1
    10
     6
     5
```

```
fact_error = [xk_fact_err;x_curr_seid_err;x_rel_err]
```

fact_error = 3×1

10^{-4} ×

```
    0.5735
    0.1850
    0.0491
```

```
clar_fact_err = [x_clar_err;x_clar_seid_err;0]
```

clar_fact_err = 3×1

10^{-5} ×

```
    0.2786
    0.0001
         0
```

```
t = table(method,steps,fact_error,clar_fact_err)
```

t = 3×4 table

	method	steps	fact_error	clar_fact_err
1	'Метод прос...	10	5.7347e-05	2.7857e-06
2	'Метод Зейделя'	6	1.8505e-05	5.7925e-10
3	'Метод верх...	5	4.9055e-06	0

```
t.Properties.VariableNames ={'Название метода','Количество итераций', 'Фактическая погрешность'
```

t = 3×4 table

...

	Название метода	Количество итераций	Фактическая погрешность
1	'Метод простой итерации'	10	5.7347e-05
2	'Метод Зейделя'	6	1.8505e-05
3	'Метод верхней релакс...	5	4.9055e-06

Вывод: Среди реализованных методов наименьшее число итераций требуется для Метода верхней релаксации, который, к тому же, имеет наименьшую фактическую погрешность. Наиболее медленным и наименее точным является Метод простой итерации.