MS211 - Atividade 2

Adair Antonio da Silva Neto

Questão 1

O sistema têm solução?

```
A = [3 -2 4;-1 2 2;4 1 1;5 -6 0]; % Matriz dos coeficientes
b = [10;5;-1;0];
```

Como a matriz dos coeficientes A tem quatro linhas e três colunas não é possível computar o determinante. Portanto, vamos observar o posto de A e o posto de $A|b$ onde b é o vetor $(10, 5, -1, 0)^T$.

Observe que podemos escalonar a matriz A obtendo então:

```
disp(rref(A));
```

```
     1     0     0
     0     1     0
     0     0     1
     0     0     0
```

Ou seja, A tem três linhas linearmente independentes. Escalonando $A|b$,

```
C = [A b]; % Matriz estendida com os coeficientes
disp(rref(C));
```

```
    1.0000         0         0   -0.7778
         0    1.0000         0   -0.6481
         0         0    1.0000    2.7593
         0         0         0         0
```

ou seja, $\text{posto}(A) = \text{posto}(A|b)$. Portanto, o sistema é consistente, i.e., admite solução.

Método de Eliminação de Gauss

```
[L, U, b, x] = gausseliminationopt(A, b, 3);
```

Onde obtemos a matriz triangular superior:

```
disp(U);
```

```
    3.0000   -2.0000    4.0000
         0    1.3333    3.3333
         0         0  -13.5000
```

O vetor b:

```
disp(b);
```

```
   10.0000
    8.3333
  -37.2500
         0
```

E o vetor solução:

```
disp(x.');
```

```
  -0.7778
  -0.6481
   2.7593
```

Questão 2

Matriz de Hankel A_4

```
% Cálculo dos coeficientes da matriz de Hankel
A = zeros(4);
for i = 1:4
    for k=(i-3):i
        if k>0
            A(i,4+k-i) = 2^k;
        else
            A(i,4+k-i) = 2^(1/(2-k));
        end
    end
end

disp(A);
```

```
    1.1892    1.2599    1.4142    2.0000
    1.2599    1.4142    2.0000    4.0000
    1.4142    2.0000    4.0000    8.0000
    2.0000    4.0000    8.0000   16.0000
```

Determinar $\overline{b}$ tal que $A_4x = \overline{b}$ tem como solução $\overline{x} = [1, \dots, 1]^t$

Fazendo A_4x obtemos b:

```
x = [1;1;1;1];
b = A*x;
disp(b);
```

```
    5.8633
    8.6741
   15.4142
   30.0000
```

Códigos para Eliminação de Gauss com e sem pivoteamento

Função auxiliar para substituição:

```
% function solution = backsubstitution(A, b, n)
% x(n) = b(n)/A(n,n);
% k = n;
% while k>1
%     k = k-1;
%     soma = b(k);
```

```
%     j = k;
%     while j<n
%         j = j+1;
%         soma = soma - A(k,j)*x(j);
%     end
%     x(k) = soma/A(k,k);
% end
% solution = x;
% end
```

Sem pivoteamento:

```
% function [L,U,b,x] = gausseliminationopt(A, b, n)
%
% % Processo de eliminação
% for k=1:n-1
%     for i = k+1:n
%         m = A(i,k) / A(k,k);
%         A(i,k) = m;
%         b(i, :) = b(i, :) - m * b(k, :);
%         for j=k+1:n
%             A(i,j) = A(i,j) - m * A(k,j);
%         end
%     end
% end
%
% L = eye(n);
% U = zeros(n);
%
% % Monta as matrizes U e L
% for i=1:n
%    for j=1:n
%        if j<i
%            L(i,j) = A(i,j);
%        else
%            U(i,j) = A(i,j);
%        end
%    end
% end
%
% x = backsubstitution(U, b, n);
%
% end
```

Com pivoteamento:

```
% function x = gausseliminationpivot(A, b, n)
%     for k=1:n-1
%
%         % Finds the pivot
%         max = abs(A(k,k));
%         maxv = k;
%         for i=k+1:n
%             if abs(A(i,k)) > max
```

```
%                 max = abs(A(i,k));
%                 maxv = i;
%             end
%         end
%
%         % Exchanges
%         mem = A(k,:);
%         A(k,:) = A(maxv,:);
%         A(maxv,:) = mem;
%         mem = b(k,:);
%         b(k,:) = b(maxv,:);
%         b(maxv,:) = mem;
%
%         % Gauss elimination procedure
%         for i = k+1:n
%             m = A(i,k) / A(k,k);
%             A(i,k) = m;
%             b(i, :) = b(i, :) - m * b(k, :);
%             for j=k+1:n
%                 A(i,j) = A(i,j) - m * A(k,j);
%             end
%         end
%     end
%     x = backsubstitution(A, b, n);
% end
```

Resolver $A_4 x = \overline{b}$ usando Eliminação de Gauss com e sem pivoteamento

Sem pivoteamento, obtemos:

```
A_opt = A;
b_opt = b;
[L,U,b_opt,x_opt] = gausseliminationopt(A_opt, b_opt, 4);
disp(x_opt);
```

```
    1.0000    1.0000    1.0000    1.0000
```

Por outro lado, utilizando pivoteamento:

```
A_pivot = A;
b_pivot = b;
x_pivot = gausseliminationpivot(A_pivot, b_pivot, 4);
disp(x_pivot);
```

```
    1.0000    1.0000    1.0000    1.0000
```

Erro relativo

Para o método sem pivoteamento:

```
error_opt = norm(x-x_opt,2)/norm(x_opt,2);
disp(error_opt);
```

```
   3.8218e-15
```

Para o método com pivoteamento:

```
error_pivot = norm(x-x_pivot,2)/norm(x_pivot,2);
disp(error_pivot);
```

```
   5.9798e-15
```

Matriz de Hankel A_{10}

```
% Cálculo dos coeficientes da matriz de Hankel
A = zeros(10);
for i = 1:10
    for k=(i-9):i
        if k>0
            A(i,10+k-i) = 2^k;
        else
            A(i,10+k-i) = 2^(1/(2-k));
        end
    end
end

disp(A);
```

```
   1.0e+03 *

  Columns 1 through 6

    0.0011    0.0011    0.0011    0.0011    0.0011    0.0011
    0.0011    0.0011    0.0011    0.0011    0.0011    0.0012
    0.0011    0.0011    0.0011    0.0011    0.0012    0.0013
    0.0011    0.0011    0.0011    0.0012    0.0013    0.0014
    0.0011    0.0011    0.0012    0.0013    0.0014    0.0020
    0.0011    0.0012    0.0013    0.0014    0.0020    0.0040
    0.0012    0.0013    0.0014    0.0020    0.0040    0.0080
    0.0013    0.0014    0.0020    0.0040    0.0080    0.0160
    0.0014    0.0020    0.0040    0.0080    0.0160    0.0320
    0.0020    0.0040    0.0080    0.0160    0.0320    0.0640

  Columns 7 through 10

    0.0012    0.0013    0.0014    0.0020
    0.0013    0.0014    0.0020    0.0040
    0.0014    0.0020    0.0040    0.0080
    0.0020    0.0040    0.0080    0.0160
    0.0040    0.0080    0.0160    0.0320
    0.0080    0.0160    0.0320    0.0640
    0.0160    0.0320    0.0640    0.1280
    0.0320    0.0640    0.1280    0.2560
    0.0640    0.1280    0.2560    0.5120
    0.1280    0.2560    0.5120    1.0240
```

Determinar $\overline{b}$ tal que $A_{10}x = \overline{b}$ tem como solução $\overline{x} = [1, \ldots, 1]^t$

Fazendo $A_{10}x$ obtemos b:

```
x = [1;1;1;1;1;1;1;1;1;1];
b = A*x;
```

```
disp(b);
```

```
  1.0e+03 *

    0.0125
    0.0154
    0.0223
    0.0372
    0.0681
    0.1310
    0.2579
    0.5127
    1.0234
    2.0460
```

Resolver $A_{10}x = \overline{b}$ usando Eliminação de Gauss com e sem pivoteamento

Sem pivoteamento, obtemos:

```
A_opt = A;
b_opt = b;
[L,U,b_opt,x_opt] = gausseliminationopt(A_opt, b_opt, 10);
disp(x_opt);
```

```
 Columns 1 through 6

   1.0000    1.0000    1.0000    1.0000    1.0000    1.0000

 Columns 7 through 10

   1.0000    1.0000    1.0000    1.0000
```

Por outro lado, utilizando pivoteamento:

```
A_pivot = A;
b_pivot = b;
x_pivot = gausseliminationpivot(A_pivot, b_pivot, 10);
disp(x_pivot);
```

```
 Columns 1 through 6

   1.0000    1.0000    1.0000    1.0000    1.0000    1.0000

 Columns 7 through 10

   1.0000    1.0000    1.0000    1.0000
```

Erro relativo

Para o método sem pivoteamento:

```
error_opt = norm(x-x_opt,2)/norm(x_opt,2);
disp(error_opt);
```

```
  2.4503e-11
```

Para o método com pivoteamento:

```
error_pivot = norm(x-x_pivot,2)/norm(x_pivot,2);
```

```
disp(error_pivot);
```

```
9.8787e-13
```

De fato, no método com pivoteamento temos um erro de $9.8787e-13$ enquanto no método sem pivoteamento temos um erro de $2.4503e-11$. Como a matriz de Hankel A_{10} tem coeficientes $m = \frac{a_{i,k}}{a_{k,k}}$ com denominadores próximos de um, as entradas da matriz crescem em módulo ao longo das iterações, o que implica em um aumento do erro.

Questão 3

Vamos considerar o sistema $A_4x = b$, com b qualquer e A_4 a matriz de Hankel de ordem quatro.

```
% Cálculo dos coeficientes da matriz de Hankel
A = zeros(4);
for i = 1:4
    for k=(i-3):i
        if k>0
            A(i,4+k-i) = 2^k;
        else
            A(i,4+k-i) = 2^(1/(2-k));
        end
    end
end

disp(A);
```

```
    1.1892    1.2599    1.4142    2.0000
    1.2599    1.4142    2.0000    4.0000
    1.4142    2.0000    4.0000    8.0000
    2.0000    4.0000    8.0000   16.0000
```

Verificando com método de Jacobi e Gauss-Seidel

Tomando

```
b = [5.863341727270689; 8.674134612267968; 15.414213562373096; 30.000000000000000];
```

com o método de Jacobi obtemos o seguinte resultado:

```
A_jacobi = A;
b_jacobi = b;
x0 = [0; 0; 0; 0];
epsilon = 10^(-4);

x_jacobi = jacobi(A_jacobi, b_jacobi, x0, epsilon);
disp(x_jacobi);
```

```
  1.0e+307 *

    4.6150
    5.0309
    2.8745
       Inf
```

Pelo método de Gauss-Seidel:

```
A_gs = A;
b_gs = b;
x0 = [0; 0; 0; 0];

x_gs = gauss_seidel(A_gs,b_gs,x0);
disp(x_gs);
```

```
Inf
NaN
NaN
NaN
```

Temos que ambos os métodos não convergem. Observa-se que a matriz A não satisfaz o critério das linhas (em nenhuma das formas), nem o critério de Sassenfeld (colocar contas!).

Como a matriz de Hankel é simétrica (i.e. $A_4 = A_4^t$) as somas que são feitas no numerador de ambos os métodos são maiores do que os denominadores, fazendo com que ao longo das iterações, os valores de x divirjam.

Código que permite inserir A e b

```
% % Matriz A
% fprintf("Vamos digitar a matriz A.\n");
% for i=1:4
%     for j=1:4
%         A(i,j)=input(sprintf('Entrada A(%d,%d): ', i, j));
%     end
% end
%
% % Vetor b
% fprintf("Agora vamos digitar o vetor b.\n");
% for i=1:4
%     b(i)=input(sprintf('Entrada %d: ', i));
% end
%
% % Resolução
% x = gauss_seidel(A,b.',[0;0;0;0]);
% disp(x);
```

Com coeficientes potências de 2 é possível que Gauss-Seidel convirja? Se sim, implementar satisfazendo $r^{(k)} < \varepsilon$

Afirmação: Não é possível construir uma matriz de Hankel de ordem quatro tal que a convergência do método de Gauss-Seidel seja garantida.

Demonstração. Por contradição considere uma matriz qualquer de Hankel de ordem quatro.

```
syms c1 c2 c3 c4 c5 c6 c7;
hankel = [c1 c2 c3 c4; c2 c3 c4 c5; c3 c4 c5 c6; c4 c5 c6 c7];
disp(hankel);
```

$$\begin{pmatrix} c_1 & c_2 & c_3 & c_4 \\ c_2 & c_3 & c_4 & c_5 \\ c_3 & c_4 & c_5 & c_6 \\ c_4 & c_5 & c_6 & c_7 \end{pmatrix}$$

E suponha que seja possível garantir a convergência do método de Gauss-Seidel. Assim, o critério de Sassenfeld deve ser satisfeito, isto é, devemos ter $\beta_i < 1$ para $i = 1, 2, 3, 4$. Explicitamente,

- $$\beta_1 = \frac{|c_2| + |c_3| + |c_4|}{|c_1|} < 1$$
- $$\beta_2 = \frac{|c_2|\beta_1 + |c_4| + |c_5|}{|c_3|} < 1$$

Observe que $\beta_3 = \dfrac{|c_3|\beta_1 + |c_4|\beta_2 + |c_6|}{|c_5|} < 1 \iff |c_5| > |c_3|\beta_1 + |c_4|\beta_2 + |c_6|$.

Como β_1 e β_2 são ambos menores do que um, temos que $|c_5| > |c_3| + |c_4| + |c_6|$.

Porém, de β_2 temos que $|c_3| > |c_5|$ e como $|c_4|$ e $|c_6|$ são ambos positivos, temos uma contradição.

Logo, o critério de Sassenfeld não é satisfeito e o método de Gauss-Seidel pode ou não convergir, como queríamos demonstrar.