AULA PRÁTICA 1
ALGORITMOS DE RESOLUÇÃO DE SISTEMAS LINEARES

ALGORITMO DE ELIMINAÇÃO GAUSSIANA

A função Scilab a seguir, implementa o algoritmo de eliminação Gaussiana para resolver um sistema quadrado Ax = b com A invertível, obtendo também a decomposição LU da matriz A. Nesta versão, supomos que os elementos diagonais (pivôs) da matriz dos coeficientes ao longo do processo são sempre não nulos.

```
/////////////////////////////////////////////////////////////////////
//Variáveis de saída:
//x: solução do sistema Ax=b (assumimos que tal solução existe).
//C: Seja A=LU a decomposição LU de A.
//Então C(i,j)=L(i,j) para i>j e C(i,j)=U(i,j) para j>=i.
/////////////////////////////////////////////////////////////////////
function [x, C]=Gaussian_Elimination_1(A, b)
C=[A,b];
[n]=size(C,1);

for j=1:(n-1)
   //O pivô está na posição (j,j)
   for i=(j+1):n
 //O elemento C(i,j) é o elemento na posição (i,j) of L na decomposição LU de A
      C(i,j)=C(i,j)/C(j,j);
      //Linha i ← Linha i - C(i,j)*Linha j
//Somente os elementos da diagonal ou acima dela são computados
//(aqueles que compõem a matrix U)
      C(i,j+1:n+1)=C(i,j+1:n+1)-C(i,j)*C(j,j+1:n+1);
   end
end

x=zeros(n,1);

// Calcula x, sendo Ux=C(1:n,n+1)

x(n)=C(n,n+1)/C(n,n);
for i=n-1:-1:1
   x(i)=(C(i,n+1)-C(i,i:n)*x(i:n))/C(i,i);
end

C=C(1:n,1:n);

endfunction
```

1) Teste a função dada usando algumas matrizes quadradas A e respectivos vetores b. Use exemplos dos quais você saiba a resposta para verificar se a função realmente está funcionando corretamente.
2) Agora teste com a matriz
   A1=[1 -2 5 0; 2 -4 1 3; -1 1 0 2; 0 3 3 1] e com o vetor b1=[1;0;0;0].
3) Modifique a função dada trocando linhas quando no início da iteração j o elemento na posição (j,j) é nulo. Chame esta nova função de Gaussian_Elimination_2 e teste-a com a matriz A1 e o vetor b1 dados.
   Agora teste-a com a matriz A2=[0 10^{-20} 1; 10^{-20} 1 1; 1 2 1] e o vetor b2=[1; 0; 0].
4) Modifique a função do item 3 para escolher o maior pivô em módulo quando no início da iteração j o elemento na posição (j,j) é nulo. Chame esta nova função de Gaussian_Elimination_3 e teste-a com a matriz A2 e o vetor b2 dados. Agora com a matriz A3=[10^{-20} 10^{-20} 1; 10^{-20} 1 1; 1 2 1] e o vetor b3=b2.
5) Modifique a função do item 4 para escolher sempre o maior pivô em módulo no início da iteração j independente do elemento na posição (j,j) ser nulo ou não. Nessa função, retorne também a matriz de permutação P utilizada. Chame esta nova função de Gaussian_Elimination_4 e teste-a com a matriz A3 e o vetor b3 dados.
6) Uma vez que você tem a decomposição LU de uma matriz quadrada A de ordem *n* (ou de PA, sendo P uma matriz permutação) a resolução de um sistema linear Ax=b pode ser obtida mais rapidamente usando a decomposição LU já feita, em vez de fazer todo o escalonamento de novo. Escreva uma função Scilab de nome Resolve_com_LU, que receba como variáveis de entrada uma matriz C com a decomposição LU de A (ou de PA, conforme matriz retornada pelas funções anteriores) e uma matriz B de ordem *n* x *m* e retorne uma matriz X, com a mesma ordem de B, cujas colunas sejam as soluções dos sistemas lineares $Ax_i=b_i$, $1 \leq i \leq m$.
   Observação: talvez você ache necessário passar outra(s) variável(is) de entrada para essa função.
   Teste a sua função com a matriz A1 dada anteriormente e com a matriz B1=[2 4 -1 5 ;0 1 0 3 ; 2 2 -1 1 ; 0 1 1 5]. Teste também com a matriz A2 dada anteriormente e com a matriz B2=[1 1 2; 1 -1 0; 1 0 1]. Finalmente, teste com a matriz A3 dada anteriormente e com a matriz B3=B2.

Sugestão de Leitura:

1) Investigação, páginas de 83 a 87, livro Álgebra Linear – David Poole – 4ªedição.
2) Fatoração LU – páginas de 180 a 189, livro Álgebra Linear – David Poole – 4ªedição.