Trabalho de Métodos Numéricos.

Gabriel Perez Claro Santana Mizuno, turma TA1 - 11/13h

25 de Junho de 2019

Descrevendo método da Fatoração LU

O procedimento da fatoração LU consiste em transformar um matriz $\mathbf{A}$ em um produto de duas matrizes $\mathbf{L}$ e $\mathbf{U}$, onde $\mathbf{L}$ é uma matriz Triangular Inferior e $\mathbf{U}$ uma matriz Triangular Superior. Estas matrizes podem ser obtidas usando o método da Eliminação Gaussiana com Estratégia de Pivoteamento Parcial, com as devidas alterações, pois nesse caso podemos ter que trocar linhas.

A partir de agora, quando for mencionado o método de Eliminação Gaussiana sempre será usado a estratégia do pivoteamento.

Considere o sistema linear $\mathbf{A_1}x = b$ e sejam as matrizes $\mathbf{L}$ e $\mathbf{U}$ obtidos pelo processo da Eliminação de Gaussiana. A matriz $\mathbf{A_1}$ é a matriz $\mathbf{A}$ com as linhas permutadas, isto é, $\mathbf{A_1} = \mathbf{PA}$, onde $\mathbf{P} = \mathbf{P_k} \cdots \mathbf{P_1}$ para $k \in \{n, \cdots, 1\}$ e b_1 é obtida a partir da permutação das mesma linhas, ou seja, $b_1 = \mathbf{P}b$.

O sistema linear $\mathbf{A_1}x = b_1$ é equivalente ao original e, se $\mathbf{A_1} = \mathbf{LU}$, teremos;

$$\mathbf{A_1}x = b \Rightarrow \mathbf{PA}x = \mathbf{P}b \Rightarrow \mathbf{LU}x = \mathbf{P}b$$

Resolvendo os seguintes sistemas triangulares, na seguinte ordem:

1. $\mathbf{L}y = \mathbf{P}b$
2. $\mathbf{U}x = y$

Onde x será solução do nosso sistema original. Porém, nesse trabalho só acharemos a matriz inversa de $\mathbf{A}$, ou seja, $\mathbf{A^{-1}}$

Achando a Inversa

Para achar a matriz inversa de $\mathbf{A}$ resolveremos n sistemas nxn, onde $b = e_i$ para todo $i \in \{1, \cdots, n\}$ e e_i é um vetor com 1 na posição i e 0 nas outras posições. Assim, será obtido $x_1, \cdots, x_n$ vetores que serão usado como vetores coluna de $\mathbf{A^{-1}}$, ou seja, $\mathbf{A^{-1}} = [x_1, \cdots, x_n]$.

Resumindo, para $i = 1$ resolver os seguintes sistema ; $\mathbf{L}y = \mathbf{P}e_1$ e $\mathbf{U}x_1 = y$

Resumindo, para $i = 2$ resolver os seguintes sistema ; $\mathbf{L}y = \mathbf{P}e_2$ e $\mathbf{U}x_2 = y$

Repetir ate achar $x_1, \cdots, x_n$ e fazer $\mathbf{A^{-1}} = [x_1, \cdots, x_n]$

Resultado

$$\text{Para } \mathbf{A} = \begin{pmatrix} 4 & -1 & 0 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 \\ -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 0 & -1 & 4 \end{pmatrix}, \text{ temos } \mathbf{A^{-1}} = \begin{pmatrix} 0.2948 & 0.0932 & 0.0282 & 0.0861 & 0.0497 & 0.0195 \\ 0.0932 & 0.3230 & 0.0932 & 0.0497 & 0.1056 & 0.0497 \\ 0.0282 & 0.0932 & 0.2948 & 0.0195 & 0.0497 & 0.0861 \\ 0.0861 & 0.0497 & 0.0195 & 0.2948 & 0.0932 & 0.0282 \\ 0.0497 & 0.1056 & 0.0497 & 0.0932 & 0.3230 & 0.0932 \\ 0.0195 & 0.0497 & 0.0861 & 0.0282 & 0.0932 & 0.2948 \end{pmatrix}$$

Matlab Código

```
%% PivotParcfunc.m
% Este programa acha o pivo de um coluna.

function [A,posicao] = PivotParcfunc(A,col,numlin)
    maior=abs(A(col:col));
    posicao=col;
    for ii=col+1:numlin
        if abs(A(ii,col))>maior
            maior=abs(A(ii,col));
            posicao=ii;
        end
    end
    auxLin=A(col,:);
    A(col,:)=A(posicao,:);
    A(posicao,:)=auxLin;
end
```

Código:

```
% Esse programa escalona matrizes usando o metodo de Gauss-Jacobi sem a
% utlizadao do pivotamento parcial
%% ElimiGaussfunc.m

function [A] = ElimiGaussfunc(MatAvelha,coluna,numtotcol,numtotlin)
    for jj=coluna+1:numtotlin
        m=MatAvelha(jj,coluna)/MatAvelha(coluna,coluna);
        for kk=coluna:numtotcol
            MatAvelha(jj,kk)=MatAvelha(jj,kk)-m*MatAvelha(coluna,kk);
        end
        MatAvelha(jj,coluna)=m;
    end
    A=MatAvelha;
end
```

Código:

```
% Esse programa troca duas linhas de uma determinada coluna de uma matriz
% trocar.m
function [A] = trocar(A,posicao1,posicao2)
    auxLin=A(posicao2,:);
    A(posicao2,:)=A(posicao1,:);
    A(posicao1,:)=auxLin;
end
```

Código:

```
%% LU.m
function [L,U,P] = LU(A)
[numlin,numcol] = size(A);

Aoriginal=A;
P = eye(numlin); Poriginal = P;

for ii=1:numlin
    [A,posicao] = PivotParcfunc(A,ii,numcol);
    [A] = ElimiGaussfunc(A,ii,numcol,numlin);
    P(:,:,ii)=trocar(Poriginal,posicao,ii);
end

% Estou dividindo a matriz em L com os m e U com a matriz superior como
% resultado do pivoteamento parcial
L = tril(A,-1)+diag(diag(ones(numlin)));
U = triu(A);
end
```

Código:

```
%% solinferior.m
function [solucao] = solinferior(A,b)

[numlin, numcol]=size(A);
X=zeros(numlin,1); % so preenchendo a matriz X poderia ser qualquer coisa
X(1,1) = b(1,1)/A(1,1);

for ii=2:numlin
    soma=0;
    for jj=1:(ii-1)
        soma = soma+A(ii,jj)*X(jj);
    end
    X(ii,1) = (b(ii,1)-soma)/A(ii,ii);
end
solucao = X;
end
```

Código:

```
%% solsuperior.m
function [solucao] = solsuperior(A,b)

[numlin, numcol]=size(A);
X=zeros(numlin,1); % so preenchendo a matriz X poderia ser qualquer coisa
X(numlin,1)=b(numlin,1)/A(numlin,1);


for ii=numlin:-1:1
    soma=0;
    for jj=ii:(numcol-1)
        soma = soma+A(ii,jj+1)*X(jj+1);
    end
    X(ii,1) = (b(ii,1)-soma)/A(ii,ii);
end
solucao = X;
end
```

Código:

```
%% inversa.m
function [Ainv] = inversa(A)

    [numlin, numcol]=size(A);
    Ainv = zeros(numlin,numcol); % Criando um matriz de zeros q vai se tronar a inversa A
    [L,U,P] = LU(A); % Usando o programa LU.m
    Pn = eye(numlin,numcol);  % Array com as matrizes de permutacao

    for ii=numlin:-1:1
        Pn = Pn*P(:,:,ii); % Guardado essas matrizes
    end

    for ii = 1:numlin
        [y] = solinferior(L,Pn(:,ii)); % Usando o programa solinferior.m
        [Ainv(:,ii)] = solsuperior(U,y); % Usando o programa solsuperior.m
    end

end
```

Código: