Perfilamento: Decomposição LU

FLÁVIO HENRIQUE LOPES BARBOSA
JAYSA KEYLLA SIQUEIRA BARBOSA
JOSE AUGUSTO AGRIPINO DE OLIVEIRA

Tópicos

ALGORITMO

PERFILAMENTO

RESULTADOS

GARGALOS

ALGORITMO

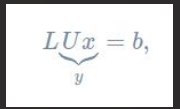
Decomposição LU (Lower Upper)

Um dos motivos para introduzir a decomposição LU é que ela fornece uma maneira eficiente de calcular a matriz inversa, a qual tem muitas aplicações na engenharia; ela também fornece um meio de avaliar o condicionamento do sistema.

A decomposição pode ser dividida em dois passos:

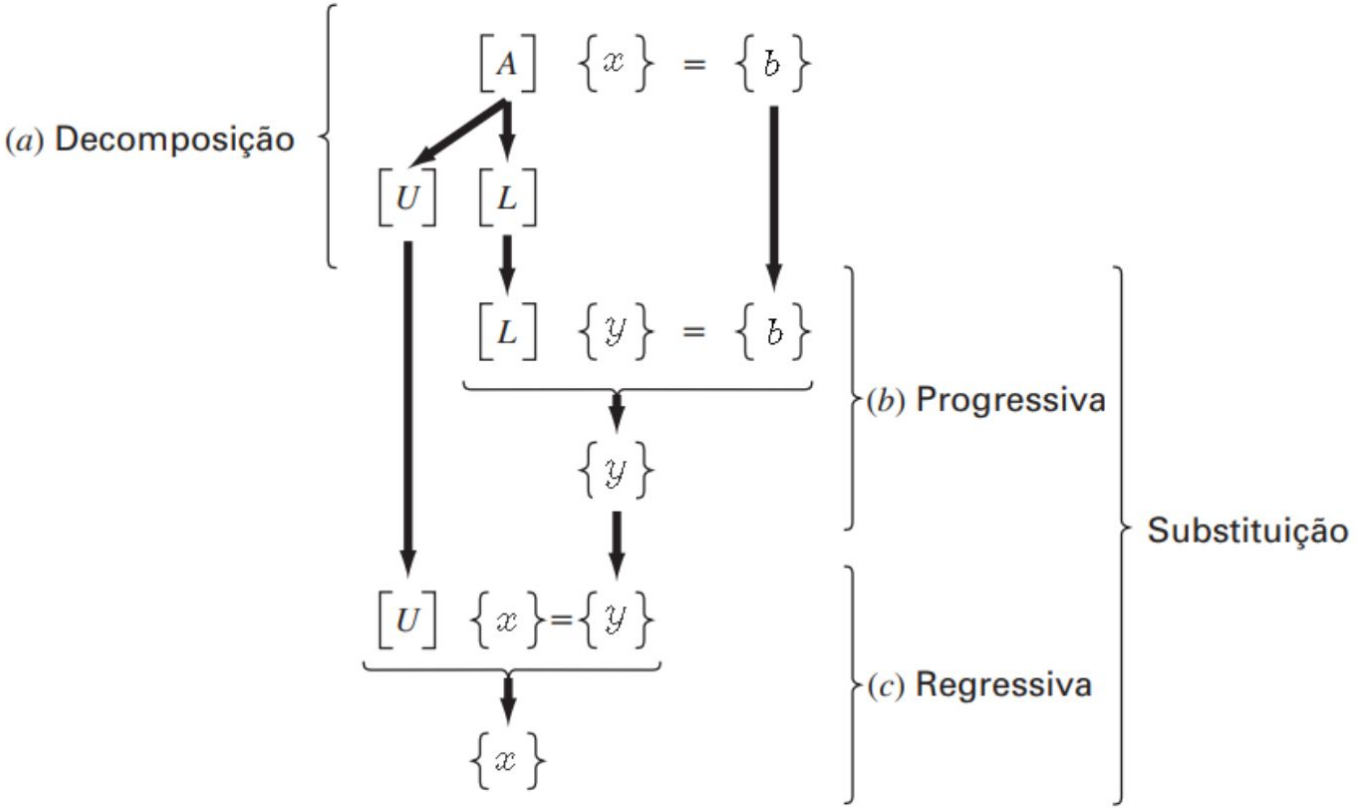
1 – Passo de decomposição: a matriz **A** é fatorada em duas matrizes triangulares, uma inferior L com elementos da diagonal principal iguais a 1, e uma superior **U** , onde, realizando a multiplicação **L×U**, obtemos a matriz **A**.

2 – Resolução do sistema: **L** e **U** são usadas para determinar a solução do sistema, **x**, através do processo: **Ax = b**
Se **A = LU**, então **LUx = y**. Defina um vetor de incógnitas auxiliar, **y**:



Ou seja, **Ux = y.** Logo, **Ly = b .**

Observe que no sistema acima, **L** é uma matriz triangular inferior, **b** é a matriz de termos independentes do sistema original e **y** é o vetor de incógnitas auxiliar. Resolvendo **Ly=b** usando substituição progressiva, podemos usar a relação **Ux=y** para encontrar xx por substituição regressiva, já que **U** é triangular superior.


(a) Decomposição
[A] {x} = {b}
[U] [L]
[L] {y} = {b}
{y}
(b) Progressiva
[U] {x}={y}
{x}
(c) Regressiva
Substituição

ALGORITMO

$$LU = \begin{pmatrix} 1 & & & & \\ \ell_{21} & 1 & & \bigcirc & \\ \ell_{31} & \ell_{32} & 1 & & \\ \dots & \dots & \dots & \ddots & \\ \ell_{n1} & \ell_{n2} & \ell_{n3} & \dots & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ & u_{22} & u_{23} & \dots & u_{2n} \\ & & u_{33} & \dots & u_{3n} \\ & \bigcirc & & \ddots & \vdots \\ & & & & u_{nn} \end{pmatrix}$$

PERFILAMENTO

- Matriz com ordem 1000;
- Matriz com ordem 2000;
- Matriz com ordem 3500;
- Matriz com ordem 5500;
- Matriz com ordem 7000;

RESULTADOS

Matriz de ordem 1000

% time	Cumulative seconds	Self seconds	Calls	Self s/call	Total s/call	name
100	1.84	1.84	1	1.84	1.84	gauss
0.00	1.84	0.00	2	0.00	0.00	preencheMatriz

RESULTADOS

Matriz de ordem 2000

% time	Cumulative seconds	Self seconds	Calls	Self s/call	Total s/call	name
99.79	14.35	14.35	1	14.35	14.37	gauss
0.21	14.38	0.03	2	0.01	0.01	preencheMatriz

RESULTADOS

Matriz de ordem 3500

% time	Cumulative seconds	Self seconds	Calls	Self s/call	Total s/call	name
99.88	69.37	69.37	1	69.37	69.41	gauss
0.12	69.45	0.08	2	0.04	0.04	preencheMatriz

RESULTADOS

Matriz de ordem 5500

% time	Cumulative seconds	Self seconds	Calls	Self s/call	Total s/call	name
99.96	294.44	294.44	1	294.44	294.50	gauss
0.04	294.56	0.12	2	0.06	0.06	preencheMatriz

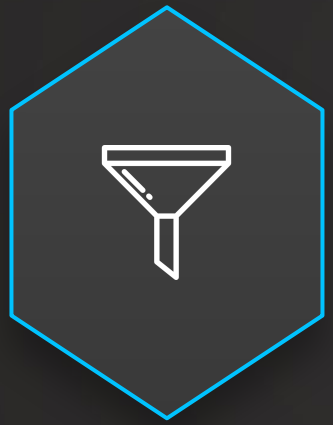
RESULTADOS

Matriz de ordem 7000

% time	Cumulative seconds	Self seconds	Calls	Self s/call	Total s/call	name
99.95	569.65	569.65	1	569.65	569.77	gauss
0.05	569.89	0.24	2	0.12	0.12	preencheMatriz

GARGALOS

-Método Gauss



```
int i,j;
float *M = (float *)calloc(n*n, sizeof(float));
float *aux = (float *)malloc(n*n*sizeof(float));
preencheMatriz(aux);
//Calculando os multiplicadores da Gauss
//Movimenta a coluna
for(j=0;j<n;j++){
    //Movimenta a linha
    for(i=j+1;i<n;i++){
        if(aux[(i*n)+j]!=0)
         {
            M[(i*n)+j]=aux[(i*n)+j]/aux[(j*n)+i];
            //multiplico a linha do multiplicar por -(multiplicador)
            int c=0;

            //Movimenta a coluna
            for(c=j;c<n;c++){
                aux[(i*n)+c]=aux[(i*n)+c]+aux[(j*n)+c]*(-1*(M[(i*n)+j]));
            }
        }
        //system("pause");
    }//Todos os multiplicadores da coluna j estão definidos

}
printf("\n\n*********RESULTADO FINAL**********\n");
for(i=0;i<n;i++) M[(i*n)+i]=1;
```