# Universidade de São Paulo

# EP 1 - MAP3121

## Decomposição LU para Matrizes Tridiagonais


| Nome | NUSP |
|---|---|
| Diogo Vaccaro | 8803195 |
| Otávio Henrique Monteiro | 10774159 |


1 de Maio de 2022

# Conteúdo

# Introdução

O primeiro exercício programa proposto pela equipe de Métodos Numéricos e Aplicações diz respeito a um tipo de decomposição de matrizes, pelo método LU, i.e. decompondo uma matriz triangularizável em uma matriz triangular inferior e uma matriz triangular superior.

# Decomposição LU

Sendo assim, para uma matriz diagonalizável **A** temos a decomposição em matriz **L** e **U** que recuperam a matriz original com uma multiplicação de matrizes, e.g. para n=3**:**

$$A = L \cdot U = \begin{pmatrix} 1 & 0 & 0 \\ L_{2,1} & 1 & 0 \\ L_{3,1} & L_{3,2} & 1 \end{pmatrix} \cdot \begin{pmatrix} U_{1,1} & U_{1,2} & U_{1,3} \\ 0 & U_{2,2} & U_{2,3} \\ 0 & 0 & U_{3,3} \end{pmatrix} \tag{1}$$

Assim, fica evidente que os elementos da matriz A podem ser encontrados por uma relação do tipo:

$$A_{ij} = \sum_{k=1}^{\min\{i,j\}} L_{ik} U_{kj}. \tag{2}$$

$$A_{3,3} = L_{3,1} \cdot U_{1,3} \ + L_{3,2} \cdot U_{2,3} \ + 1 \cdot U_{3,3} \tag{3}$$

Assim, podemos resolver qualquer sistema linear $Ax = b$ resolvendo-se um sistema triangular inferior (usando L) e outro triangular superior (usando B), como segue:

$$Sistema\colon Ax = b$$
$$Decomposição\colon A = LU$$
$$Então\colon LUx = b$$
$$Chamamos\colon y = Ux$$

$$i.\ Resolve\ o\ sistema\ triangular\ inferior\ Ly = b$$
$$ii.\ Resolve\ o\ sistema\ triangular\ superior\ Ux = y$$

As equações a seguir, que mostram uma forma de calcular os coeficientes de forma diferente, podem ser encontradas rearranjando as equações desenvolvidas diretamente pela multiplicação matricial:

$$U_{ij} = A_{ij} - \sum_{k=1}^{i-1} L_{ik} U_{kj}, \quad j = i, \cdots, n \tag{4}$$

$$L_{ji} = \left( A_{ji} - \sum_{k=1}^{i-1} L_{jk} U_{ki} \right) / U_{ii}, \quad j = i+1, \cdots, n \tag{5}$$

$$U_{3,3} = A_{3,3} - \left( L_{3,1} \cdot U_{1,3} + L_{3,2} \cdot U_{2,3} \right) \tag{6}$$

## Matrizes Tridiagonais

Em especial, estudaram-se as matrizes tridiagonais, cujos elementos não são nulos apenas nas diagonais principal e secundárias acima e abaixo da principal.

$$A = \begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{bmatrix} \tag{7}$$

Esse tipo de matriz pode ser entendida diretamente pelo conjunto de diagonais, que facilita o armazenamento e cálculos a serem realizados. Seja A uma matriz tridiagonal triangularizável pelo método de Eliminação de Gauss sem trocas de linhas. Podemos notar que os termos da matriz A que são dados da forma $c_i$ , não são alterados durante o escalonamento. Isso acontece porque durante o processo de anular os termos $a_{i+1}$ abaixo dos pivôs $b_i$ toda a linha do termo $a_{i+1}$ é subtraída da linha do pivô multiplicada pelo termo $a_{i+1}$. No entanto, todos os termos imediatamente acima dos $c_i$ são nulos. Portanto, os termos $c_i$ não são alterados. Ou seja, na matriz U, triangular superior do escalonamento, os termos $U_{i,i+1}$ são iguais a $c_i$.

Além desse fato acima, os únicos multiplicadores que podem ser não nulos são os $L_{i+1,i}$ já que para cada pivô $b_i$ apenas a linha abaixo possui um número a ser anulado $a_{i+1}$ que resultará justamente no multiplicador $L_{i+1}$.

O primeiro termo de U, $U_{11}$ , é o termo $b_1$ diretamente, pois já é um termo pivô. Para os demais termos, fazendo o escalonamento, temos que:

$$L_{i+1,i} = a_{i+1} / u_i \text{ (multiplicador)}$$
$$U_{ii} = b_i - L_{i,i-1} * c_i \text{ (próximo pivô)}$$

Assim, a matriz tridiagonal pode ser guardada em 3 vetores a, b e c que contém as diagonais principal e secundárias.

## Matrizes Tridiagonais cíclicas

Ademais, o método desenvolvido para esse tipo de matriz pode ser expandido facilmente para sistemas tridiagonais cíclicos, que adicionam um elemento a cada diagonal secundária.

$$A = \begin{bmatrix} b_1 & c_1 & & & a_1 \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ c_n & & & a_n & b_n \end{bmatrix}$$

Sua solução é dada por:

$$x_n = \frac{d_n - c_n \tilde{y}_1 - a_n \tilde{y}_{n-1}}{b_n - c_n \tilde{z}_1 - a_n \tilde{z}_{n-1}} \quad e \quad \tilde{x} = \tilde{y} - x_n \tilde{z}$$

onde T é a submatriz principal de A (n-1)x(n-1); y~ é a solução do sistema linear Ty~ = d~ e z~ é a solução do sistema linear Tz~=v, onde v = ($a_1$,0,..,0,$c_{n-1}$); e d~=d (n-1).

Foi dado como objetivo do exercício programa desenvolver um método capaz de decompor uma matriz tridiagonal A *n* x *n* e resolver um sistema tridiagonal cíclico, que será utilizado em exercícios futuros.

# Desenvolvimento

Este exercício programa foi desenvolvido em C++, seguindo as especificações detalhadas no enunciado fornecido. Foi criado um documento *LEIAME.txt* contendo uma explicação breve de compilação, interação, funcionamento e saídas do programa.

O programa interage com o usuário pedindo a forma de entrada, que pode ser matricial ou diretamente com os parâmetros das diagonais e vetor *d*. No primeiro caso são extraídos estes parâmetros e no segundo a matriz é reconstruída, para facilitar os cálculos de comparação de resultados ao final da execução. Dessa forma o programa é capaz de se adaptar ao tipo de entrada de trabalhos posteriores.

Após a entrada de dados e extração de parâmetros é realizada a decomposição LU seguindo-se o algoritmo encontrado com base no Método de eliminação de Gauss:

```
u[1]=b[1];
for(int i=2;i<=n;i++){
   l[i]=a[i]/u[i-1];
   u[i]=b[i]-l[i]*c[i-1];
}
```

Achamos então a solução, por meio da seguinte combinação:

$$Ax = d \Rightarrow LUx = d$$
$$L(Ux) = d \Rightarrow Ly = d$$
$$Ux = y$$

Dessa forma, somos capazes de encontrar os valores de solução Ax=d. Esses valores de x encontrados são então usados na multiplicação matricial com a matriz A original e comparados com o valor esperado (d).

Foi fornecido um exemplo de sistema linear tridiagonal cíclico para testes, com funções para determinar os parâmetros das diagonais e resultado. Foram realizados testes para diferentes n, em especial para n=20.

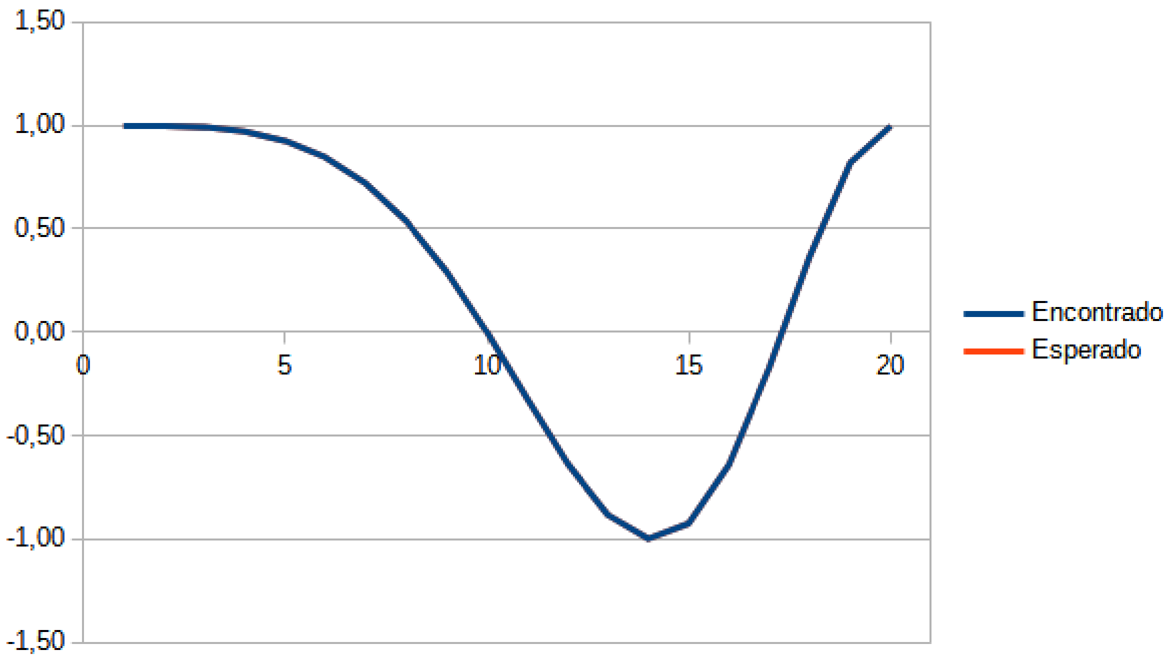
```
//Parametros até n-1
for (int i = 1; i < n; i++)
{
    float aux = (float) i;
    a[i] = (2*aux - 1)/(4*aux);
    b[i] = 2;
    c[i] = 1 - a[i];
    d[i] = cos((2*PI*i*i)/(n*n));
}
//Parametros em i=n
a[n] = (2*n - 1)/(2*n);
b[n] = 2;
c[n] = 1 - a[n];
d[n] = cos(2*PI);
```

# Saídas para n=20

Os parâmetros e resultados, gerados durante a execução do programa, para o exemplo desse sistema linear tridiagonal cíclico de dimensões n=20 estão também no arquivo *output.txt*.

O vetor solução encontrado é:
x = { +0,33032 ; +0,33370 ; +0,33082 ; +0,32459 ; +0,31054 ; +0,28498 ; +0,24376 ; +0,18349 ; +0,10274 ; +0,00361 ; -0,10670 ; -0,21473 ; -0,30114 ; -0,34331 ; -0,32098 ; -0,22451 ; -0,06386 ; +0,12581 ; +0,28714 ; +0,35589 }

O seguinte gráfico mostra a comparação entre o resultado obtido pela multiplicação de **A** com os valores encontrados de ***x*** e o resultado esperado ***d***:




Repara-se que no gráfico as curvas estão sobrescritas, já que os resultados numéricos são idênticos.

# Referências

[1] Equipe de Métodos Numéricos. Decomposição LU para Matrizes Tridiagonais - MAP3121, 2022.

# Anexos

## 1. Resultados n=20 (utilizados para o gráfico)

| d[i] | Encontrado | Esperado | Erro |
|---|---|---|---|
| 1 | 0,999877 | 0,999877 | 5,96E-08 |
| 2 | 0,998027 | 0,998027 | 0 |
| 3 | 0,990024 | 0,990024 | 5,96E-08 |
| 4 | 0,968583 | 0,968583 | 0 |
| 5 | 0,923879 | 0,923880 | 5,96E-08 |
| 6 | 0,844328 | 0,844328 | 0 |
| 7 | 0,718126 | 0,718126 | 5,96E-08 |
| 8 | 0,535827 | 0,535827 | 0 |
| 9 | 0,294040 | 0,294040 | 2,98E-08 |
| 10 | 0,000000 | 0,000000 | 3,73E-09 |
| 11 | -0,323917 | -0,323917 | 2,98E-08 |
| 12 | -0,637424 | -0,637424 | 0 |
| 13 | -0,883766 | -0,883766 | 0 |
| 14 | -0,998027 | -0,998027 | 0 |
| 15 | -0,923880 | -0,923880 | 5,96E-08 |
| 16 | -0,637424 | -0,637424 | 5,96E-08 |

| | | | |
|---|---|---|---|
| 17 | -0,171929 | -0,171929 | 1,49E-08 |
| 18 | 0,368125 | 0,368125 | 0 |
| 19 | 0,818150 | 0,818150 | 0 |
| 20 | 1,000000 | 1,000000 | 0 |

## 2. Código

```
#include<stdio.h>
#include <iostream>
#include <cmath>

const double PI = 3.141592653589793238463;
#define MAX 100

using namespace std;

// algoritmo para decomposi��o LU de uma matriz tridiagonal A nxn

void solucaoLU(float x[MAX],float y[MAX], float d[MAX],float l[MAX], float u[MAX], float c[MAX], int n);
// Solu��o da equa��o matricial LUx = d

void decomposicaoLU(float a[MAX],float b[MAX],float c[MAX],float l[MAX],float u[MAX], int n);
// decompoe a matriz tridiagonal A nxn em LU

void imprimir_vetor(float V[MAX],int n,int p);
// imprime o vetor V de tamanho n; p==0 na posicao horizontal e p==1 na posicao vertical

void imprimir_matriz(float A[MAX][MAX],int l, int c);
// imprime a matriz A de tamanho lxc

void erro_solucao(float A[MAX][MAX],float x[MAX],float d[MAX],int n);
// compara o resultado A.X com D

int main(){
    // INICIALIZAR VARI�VEIS
    bool ciclica;
    int opcao;
    int n;
    float A[MAX][MAX];
    float a[MAX];
    float b[MAX];
    float c[MAX];
    float d[MAX];
    float l[MAX];
    float u[MAX];
    float y[MAX];
    float x[MAX];
    float z[MAX];
```

```
cout<< "-- Bem vindo ao EP1 --"<< endl;
cout<< "1 - Inserir a matriz tridiagonal" << endl;
cout<< "2 - Gerar a matriz do enunciado (ciclica)" << endl;
cout<< "Digite o numero de uma das opcoes acima: ";
cin>>opcao;

cout<< "Digite o numero n de linhas e colunas da matriz Anxn: ";
cin>> n;

switch (opcao){
   case 1: // Inserir a matriz
         //Entrando com a matriz inteira
      cout<<"Entre com os valores linha a linha: "<< endl;
      for (int i = 1; i <= n; i++){
         cout<<"Entre individualmente com os valores da linha "<<i<<" : "<< endl;
         for (int j = 1; j <= n; j++){
            cin >> A[i][j];
         }

         //Diagonais a cada linha
         a[i] = A[i][i-1];
         b[i] = A[i][i];
         c[i] = A[i][i+1];
      }

      //Ciclica?
      cout << "A matriz e' ciclica? [S/N] :";
      char cicl;
      cin >> cicl;
      if (cicl == 'S' || cicl == 's'){ciclica = true;}
      else{ciclica = false;}

      if (ciclica){
         a[1] = A[1][n];
         c[n] = A[n][1];
      }
      else{
         a[1] = 0;
         c[n] = 0;
         //Para calculo final
         A[1][n] = 0;
         A[n][1] = 0;
      }

      //Entrada de d
      cout << "Digite os valores do termo d: "<<endl;
      cin.clear();
      fflush(stdin);
      for (int i = 1; i <= n; i++){
         cin >> d[i];
      }
      break;

   case 2: // Gerar a matriz
      ciclica = true;
      //Parametros até n-1
         for (int i = 1; i < n; i++)
         {
            float aux = (float) i;
            a[i] = (2*aux - 1)/(4*aux);
            b[i] = 2;
            c[i] = 1 - a[i];
```

```
            d[i] = cos((2*PI*i*i)/(n*n));
        }
        //Parametros em i=n
        float aux = (float) n;
         a[n] = (2*aux - 1)/(2*aux);
        b[n] = 2;
        c[n] = 1 - a[n];
        d[n] = cos(2*PI);

        // construir a matriz A
        for (int i = 1; i <=n; i++){
            for (int j = 1; j<=n; j++){
                if(i==j+1){
                    A[i][j]=a[i];
                } else if(i==j){
                    A[i][j]=b[i];
                } else if(i+1==j){
                    A[i][j]=c[i];
                } else if(i==1 && j==n){
                    A[i][j]=a[i];
                } else if(i==n && j==1){
                    A[i][j]=c[n];
                }
            }
        }
        break;
    default: cout<< "escolha invalida";
}

//Imprimir a matriz

cout << endl << "A matriz inserida eh a seguinte:" << endl << endl;;
imprimir_matriz(A,n,n);
//Mostra das diagonais
cout << endl << "Os parametros utilizados sao: "<< endl;
cout << "a[] = " << endl;
imprimir_vetor(a,n,0);
cout << "b[] = " << endl;
imprimir_vetor(b,n,0);
cout << "c[] = " << endl;
imprimir_vetor(c,n,0);
cout << "d[] = " << endl;
imprimir_vetor(d,n,1);

// SOLUCAO DO EP

decomposicaoLU(a,b,c,l,u,n);
if (ciclica){ // resolver matriz ciclica
    float dn = d[n];
    float aux[MAX];
    float v[MAX];
    for(int i=1;i<=n;i++){
        v[i]=0;
    }
    v[1]=a[1];
    v[n-1]=c[n-1];

    // Obter y e z solucoes das equacoes Ty=d ; Tz=v
    solucaoLU(y,aux,d,l,u,c,n-1);
    solucaoLU(z,aux,v,l,u,c,n-1);
    // Solucao Xn
    x[n]= (d[n]-c[n]*y[1]-a[n]*y[n-1])/(b[n]-c[n]*z[1]-a[n]*z[n-1]);
    for (int i=n-1;i>0;i--)
```

```
        x[i]=y[i]-x[n]*z[i];


    }else // resolver matriz tridiagonal nao ciclica
        solucaoLU(x,y,d,l,u,c,n);

    // Imprimir resposta
    cout << "\nO vetor solucao da matriz A eh:" << endl;
    imprimir_vetor(x,n,1);

    //Calcular resultado e avaliar se bate
    erro_solucao(A,x,d,n);

    //Finalizar
    cout << "\n\nDigite algum caractere para finalizar.\n";
    char end;
    cin >> end;
    return 1;
}

void decomposicaoLU(float a[MAX],float b[MAX],float c[MAX],float l[MAX],float u[MAX],int n){
    u[1]=b[1];
    for(int i=2;i<=n;i++){
        l[i]=a[i]/u[i-1];
        u[i]=b[i]-l[i]*c[i-1];
    }
}

void solucaoLU(float x[MAX],float y[MAX], float d[MAX],float l[MAX], float u[MAX], float c[MAX],int n){
    // Ly = d:
    y[1]=d[1];
    for (int i=2;i<=n;i++)
        y[i]=d[i]-l[i]*y[i-1];
    // Ux = y;
    x[n]=y[n]/u[n];
    for (int i=n-1;i>0;i--)
        x[i]=(y[i]-c[i]*x[i+1])/u[i];
}

void imprimir_vetor(float V[MAX],int n,int p){
    if (p==0){ // horizontal
        cout << "| ";
        for(int i=1;i<=n;i++){
            printf("%+7.5f ",V[i]);
            if(i!=n)cout << " ";
        };
        cout << " |" << endl;
    } else{// p==1 -> vertical
        for (int i=1;i<=n;i++) {
            cout <<" | ";
            printf("%+7.5f ",V[i]);
            cout <<"| " << endl;
        }
    }
}

void imprimir_matriz(float A[MAX][MAX],int l, int c){
        for (int i=1;i<=l;i++) {
            if(i==l/2)
                cout << " Matriz A[] = | ";
            else cout << "              | ";
            for (int j=1;j<=c;j++) {
                printf("%+7.5f ",A[i][j]);
```

```
        }
        cout << "|" << endl;
    }
}

void erro_solucao(float A[MAX][MAX],float x[MAX],float d[MAX],int n){
    //Calcular resultado e avaliar se bate
    float solucao[MAX];
    float erro[MAX];
    cout<<endl<<endl<<"Comparacao entre solucao encontrada e esperada:"<<endl;
    for (int i = 1; i <= n; i++) {
        solucao[i] = 0;
        for (int j = 1; j <= n; j++) {
            solucao[i] += A[i][j]*x[j];
        }
        if (d[i]>solucao[i]){
            erro[i]=d[i]-solucao[i];
        } else erro[i]=solucao[i]-d[i];
        cout<<"d"<<i<<": encontrada="<<solucao[i]<<"; esperada="<<d[i]<<"; erro="<<erro[i] <<".\n";
    }
}
```