Laboratório de Matemática Computacional II

Aula 5

Melissa Weber Mendonça
Universidade Federal de Santa Catarina

2011

Anteriormente...

Vimos que, no MATLAB, podemos resolver um sistema linear

$$Ax = b$$

através do comando

```
>> x = A\b
```

Exemplo: Resolva o sistema

$$\begin{pmatrix} 2 & 1 & 3 \\ 2 & 6 & 8 \\ 6 & 8 & 18 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}$$

```
>> A = [2 1 3;2 6 8; 6 8 18]
>> b = [1;3;5]
>> x = A\b
```

Sistema triangular inferior

$$\begin{pmatrix} 2 & 0 & 0 \\ 2 & 6 & 0 \\ 6 & 8 & 18 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}$$

$$2x_1 = 1 \Rightarrow x_1 = 1/2$$
$$2x_1 + 6x_2 = 3 \Rightarrow x_2 = (3 - 2x_1)/6$$
$$6x_1 + 8x_2 + 18x_3 = 5 \Rightarrow x_3 = (5 - 6x_1 - 8x_2)/18$$

Sistema triangular inferior

Em geral,

$$x_i = \frac{(b_i - (a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{i,i-1}x_{i-1}))}{a_{ii}}$$

Para $i = 1$ até n, faça

$$x_i = \frac{\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j\right)}{a_{ii}}$$

sistematriangularinferior.m

Sistema triangular superior

$$\begin{pmatrix} 2 & 1 & 3 \\ 0 & 6 & 8 \\ 0 & 0 & 18 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}$$

$$18x_3 = 5 \Rightarrow x_3 = 5/18$$

$$6x_2 + 8x_3 = 3 \Rightarrow x_2 = (3 - 8x_3)/6$$

$$2x_1 + x_2 + 3x_3 = 1 \Rightarrow x_1 = (1 - x_2 - 3x_3)/2$$

Sistema triangular superior

Em geral,

$$x_i = \frac{(b_i - (a_{i,i+1}x_{i+1} + a_{i,i+2}x_{i+2} + \ldots + a_{in}x_n))}{a_{ii}}$$

Para $i = n$ até 1, faça

$$x_i = \frac{\left(b_i - \displaystyle\sum_{j=i+1}^{n} a_{ij}x_j\right)}{a_{ii}}$$

[sistematriangularsuperior.m]()

O processo de escalonamento visa transformar uma matriz qualquer $A \in \mathbb{R}^{n\times n}$ em uma matriz triangular superior:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & \cdots & 0 & a_{nn} \end{pmatrix} \to U = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & u_{nn} \end{pmatrix}.$$

Para fazer isso, efetuamos operações nas linhas da matriz A.

Como zerar os elementos abaixo de a_{ii} (pivô)?

Para $k = i + 1$ até n,

$$\text{(linha } k\text{) - } \frac{a_{ki}}{a_{ii}}\text{(linha } i\text{)}$$

Eliminação Gaussiana

Escreva um programa que transforme uma matriz quadrada

$$A \in \mathbb{R}^{n \times n}$$

em uma matriz triangular superior U usando a eliminação Gaussiana.

`eliminacaogaussiana.m`

Eliminação Gaussiana - Sistemas

Considere o sistema

$$Ax = b, A \in \mathbb{R}^{n\times n}, x, b \in \mathbb{R}^n.$$

Escreva um programa que resolva o sistema acima usando eliminação Gaussiana.

```
for j = k+1:n
    multiplicador = U(j,k)/U(k,k);
    U(j,:) = U(j,:) - multiplicador*U(k,:);
end
```

sistemaporeliminacao.m

Fatoração $A = LU$

Sabemos que, se uma matriz for inversível, ela pode ser decomposta em

$$A = LU$$

onde L é uma matriz triangular inferior (com diagonal 1) e U é uma matriz triagular superior.

Como encontrar L e U através da eliminação gaussiana?

Sabemos que:

```
for j = k+1:n
    multiplicador = U(j,k)/U(k,k);
    L(j,k) = multiplicador;
    U(j,:) = U(j,:) - multiplicador*U(k,:);
end
```

fatoracaolu.m

Troca de linhas

Se encontrarmos um pivô nulo durante a eliminação, ainda podemos tentar trocar as linhas de lugar e continuar a eliminação:

Exemplo:

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 3 \\ 2 & 5 & 8 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 2 \\ 0 & 3 & 6 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 \\ 0 & 3 & 6 \\ 0 & 0 & 2 \end{pmatrix}$$

Uma matriz de permutação é uma P, permutação da identidade: neste caso,

$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Troca de linhas

Exemplo:

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 3 \\ 2 & 5 & 8 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 2 \\ 0 & 3 & 6 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 \\ 0 & 3 & 6 \\ 0 & 0 & 2 \end{pmatrix}$$

Uma matriz de permutação é uma P, permutação da identidade: neste caso,

$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Com isso, podemos determinar uma fatoração $PA = LU$, em que P acumula todas as trocas de linhas (k) necessárias à eliminação:

$$P = P_k P_{k-1} \dots P_2 P_1$$

`palu.m`

Para testar: use os comandos abaixo para gerar um sistema que tem solução $(1, 1, \ldots, 1) \in \mathbb{R}^n$.

```
>> A = rand(m,n);
>> b = A*ones(n,1);
```

Estruturas de dados Heterogêneas

Muitas vezes, gostaríamos de armazenar dados da seguinte forma:

Título	Núm. Páginas	Datas de Empréstimo e Devolução
"Álgebra Linear"	205	12/08, 15/08
"Cálculo"	346	10/09, 12/09
"Geometria"	123	04/08, 05/09
"Topologia"	253	01/08, 04/09

Porém, estes dados são de natureza *heterogênea*: misturamos texto (string), números e intervalos. Como armazenar isso em uma só tabela no MATLAB?

Estrutura *Cell*

No MATLAB, podemos fazer o seguinte:

```
>> tabela = { 'Algebra Linear', 205, [1208, 1508];
>>             'Calculo', 346, [1009,1209];
>>             'Geometria', 123, [0408,0509];
>>             'Topologia', 253, [0108,0409] }
```

A célula funciona como uma matriz, mas aqui os índices são dados sempre entre chaves: `{}`.

Comandos

Para ver o que está armazenado na variável `tabela`, `basta usarmos o comando`

```
>> celldisp(tabela)
```

`Para verificar o tamanho de uma célula, usamos o comando`

```
>> size(tabela)
```

`Para criar uma célula vazia com` m `por` n `elementos, usamos o comando`

```
>> tabela = cell(m,n)
```

`Para apagar um elemento da célula, podemos usar a notação de matriz vazia:`

```
>> tabela{1,2} = []
```

Exercício

Escreva um programa que receba uma lista de nomes e datas de nascimento, e um mês. Seu programa deve dizer quais pessoas fazem aniversário naquele mês.

Exemplo:

Nome	Mês
'Joao'	'setembro'
'Maria'	'janeiro'
'Roberta'	'julho'
'Danilo'	'dezembro'

`aniversarios.m`

Exercício

Escreva um programa que receba uma lista de matrizes, e suas duas dimensões (m e n). No final, diga quais matrizes podem ser multiplicadas e em que ordem.

`podemmultiplicar.m`