

Eliminação Gaussiana com Pivotamento


Prof. Americo Cunha

Universidade do Estado do Rio de Janeiro – UERJ

americo.cunha@uerj.br

www.americocunha.org



 @AmericoCunhaJr



@AmericoCunhaJr



@AmericoCunhaJr



Experimento Computacional 1

$$\begin{bmatrix} 10^{-16} & 10 \\ 5 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 10 + 10^{-16} \\ -1 \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [1e-16 10; 5 -6]
>> b = [10+1e-16; -1]
>> x = gauss(A,b)
```

Experimento Computacional 1

$$\begin{bmatrix} 10^{-16} & 10 \\ 5 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 10 + 10^{-16} \\ -1 \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [1e-16 10; 5 -6]
>> b = [10+1e-16; -1]
>> x = gauss(A,b)
```

x =

0
1



Experimento Computacional 1

$$\begin{bmatrix} 10^{-16} & 10 \\ 5 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 10 + 10^{-16} \\ -1 \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [1e-16 10; 5 -6]
>> b = [10+1e-16; -1]
>> x = gauss(A,b)
```

x =

0

1

**Esse exemplo mostra que o
algoritmo da eliminação gaussiana é instável!**



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ 5 x_1 - 6 x_2 = -1 \end{cases}$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ 5 x_1 - 6 x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ 5 x_1 - 6 x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1 \approx -\frac{5}{\varepsilon} L_1$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ 5 x_1 - 6 x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1 \approx -\frac{5}{\varepsilon} L_1$$



$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ - (50/\varepsilon) x_2 = -(50/\varepsilon + 5) \end{cases}$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ 5 x_1 - 6 x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1 \approx -\frac{5}{\varepsilon} L_1$$



$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ - (50/\varepsilon) x_2 = -(50/\varepsilon + 5) \end{cases}$$

$$x_2 = \frac{-(50/\varepsilon + 5)}{-(50/\varepsilon)}$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10x_2 = 10 + \varepsilon \\ 5x_1 - 6x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1 \approx -\frac{5}{\varepsilon} L_1$$



$$\begin{cases} \varepsilon x_1 + 10x_2 = 10 + \varepsilon \\ - (50/\varepsilon)x_2 = -(50/\varepsilon + 5) \end{cases}$$

$$x_2 = \frac{-(50/\varepsilon + 5)}{-(50/\varepsilon)}$$

$$= 1 + \varepsilon/10$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10x_2 = 10 + \varepsilon \\ 5x_1 - 6x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1 \approx -\frac{5}{\varepsilon} L_1$$



$$\begin{cases} \varepsilon x_1 + 10x_2 = 10 + \varepsilon \\ - (50/\varepsilon)x_2 = -(50/\varepsilon + 5) \end{cases}$$

$$x_2 = \frac{-(50/\varepsilon + 5)}{-(50/\varepsilon)}$$

$$= 1 + \varepsilon/10$$

$$x_1 = \frac{10 + \varepsilon - 10(1 + \varepsilon/10)}{\varepsilon}$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10x_2 = 10 + \varepsilon \\ 5x_1 - 6x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1 \approx -\frac{5}{\varepsilon} L_1$$



$$\begin{cases} \varepsilon x_1 + 10x_2 = 10 + \varepsilon \\ - (50/\varepsilon)x_2 = -(50/\varepsilon + 5) \end{cases}$$

$$x_2 = \frac{-(50/\varepsilon + 5)}{-(50/\varepsilon)}$$

$$= 1 + \varepsilon/10$$

$$x_1 = \frac{10 + \varepsilon - 10(1 + \varepsilon/10)}{\varepsilon}$$

$$= 0$$



Por que a eliminação gaussiana erra esse cálculo?

$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ 5 x_1 - 6 x_2 = -1 \end{cases} \quad L_2 \leftarrow L_2 - \frac{5}{\varepsilon} L_1 \approx -\frac{5}{\varepsilon} L_1$$



$$\begin{cases} \varepsilon x_1 + 10 x_2 = 10 + \varepsilon \\ - (50/\varepsilon) x_2 = -(50/\varepsilon + 5) \end{cases}$$

$$x_2 = \frac{-(50/\varepsilon + 5)}{-(50/\varepsilon)}$$

$$= 1 + \varepsilon/10$$

$$x_1 = \frac{10 + \varepsilon - 10(1 + \varepsilon/10)}{\varepsilon}$$

$$= 0$$

Como esse problema pode ser remediado?



Experimento Computacional 2

$$\begin{bmatrix} 5 & -6 \\ 10^{-16} & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 10 + 10^{-16} \end{bmatrix} \longrightarrow \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
> > A = [5 -6; 1e-16 10]
> > b = [-1; 10+1e-16]
> > x = gauss(A,b)
```



Experimento Computacional 2

$$\begin{bmatrix} 5 & -6 \\ 10^{-16} & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 10 + 10^{-16} \end{bmatrix} \longrightarrow \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [5 -6; 1e-16 10]  
>> b = [-1; 10+1e-16]  
>> x = gauss(A,b)
```

x =

**1
1**



Experimento Computacional 2

$$\begin{bmatrix} 5 & -6 \\ 10^{-16} & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 10 + 10^{-16} \end{bmatrix} \longrightarrow \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [5 -6; 1e-16 10]  
>> b = [-1; 10+1e-16]  
>> x = gauss(A,b)
```

x =

**1
1**

Em geral, pivôs pequenos podem ser evitados trocando a ordem das equações!



Estratégias de pivotamento

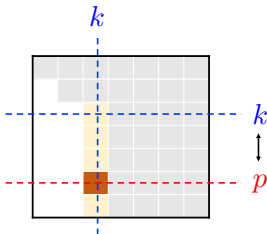
A cada estágio k do algoritmo, antes da eliminação:

Pivotamento Parcial:

1. escolha o menor p , tal que

$$|a_{pk}^{(k)}| = \max_{k \leq i \leq n} |a_{ik}^{(k)}|$$

2. troque linhas k/p

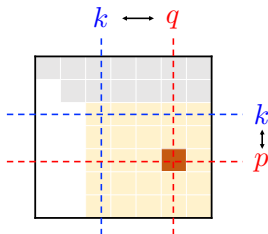


Pivotamento Completo:

1. escolha os menores p e q , tal que

$$|a_{pq}^{(k)}| = \max_{k \leq i, j \leq n} |a_{ij}^{(k)}|$$

2. troque linhas k/p e colunas k/q



Algoritmo p/ eliminação gaussiana c/ pivotamento parcial

Input: A , b

```
1: Compute the length of  $b$ 
2: for  $k=1:n-1$  do
3:   Partial pivoting process
4:   for  $i=k+1:n$  do
5:      $l_{ik} = -a_{ik}/a_{kk}$ 
6:     for  $j=1:n$  do
7:        $a_{ij} = a_{ij} + l_{ik} a_{kj}$ 
8:     end for
9:      $b_i = b_i + l_{ik} b_k$ 
10:  end for
11: end for
12: Compute  $x$  with backward substitution
13: return
```

Output: x

Esse algoritmo tem uma implementação pedagógica da eliminação gaussiana com pivotamento, não é o mais eficiente do ponto de vista computacional.



Implementação em GNU Octave

```
1 function [x,A,b] = gaussPivotP(A,b)
2   n = length(b);
3   for k=1:n-1
4     [A,b] = pivoting(A,b,n,k);
5     for i=k+1:n
6       Lik = -A(i,k)/A(k,k);
7       for j=1:n
8         A(i,j) = A(i,j) + Lik*A(k,j);
9       end
10      b(i) = b(i) + Lik*b(k);
11    end
12  end
13  x = backsub(A,b);
14 end
```



Implementação em GNU Octave

```
1 function [A,b] = pivoting(A,b,n,k)
2     pivot = abs(A(k,k));
3     row   = k;
4     for i=k+1:n
5         if abs(A(i,k)) > pivot
6             pivot = abs(A(i,k));
7             row   = i;
8         end
9     end
10    for j=k:n
11        swap      = A(row,j);
12        A(row,j) = A(k,j);
13        A(k,j)   = swap;
14    end
15    swap      = b(row);
16    b(row)   = b(k);
17    b(k)     = swap;
18 end
```



Experimento Computacional 3

$$\begin{bmatrix} 10^{-16} & 10 \\ 5 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 10 + 10^{-16} \\ -1 \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [1e-16 10; 5 -6]
```

```
>> b = [10+1e-16; -1]
```

```
>> x = gaussPivotP(A,b)
```



Experimento Computacional 3

$$\begin{bmatrix} 10^{-16} & 10 \\ 5 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 10 + 10^{-16} \\ -1 \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [1e-16 10; 5 -6]
```

```
>> b = [10+1e-16; -1]
```

```
>> x = gaussPivotP(A,b)
```

x =

1
1



Experimento Computacional 3

$$\begin{bmatrix} 10^{-16} & 10 \\ 5 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 10 + 10^{-16} \\ -1 \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



```
>> A = [1e-16 10; 5 -6]
>> b = [10+1e-16; -1]
>> x = gaussPivotP(A,b)
```

x =

1
1

O pivotamento aumenta a estabilidade do algoritmo!



Quando não é preciso pivotar?

- Matriz Diagonal Dominante (DD)

$$|a_{ii}| \geq \sum_{\substack{i=1 \\ i \neq j}}^n |a_{ij}|, \quad i = 1, \dots, n$$

- Matriz Simétrica Positiva Definida (SPD)

$A = A^T$ e $\mathbf{x}^T A \mathbf{x} > 0$, para qualquer vetor não nulo $\mathbf{x} \in \mathbb{R}^n$

$$A = \begin{bmatrix} 4 & 1 \\ 1 & 3 \end{bmatrix} \quad B = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 5 & 1 \\ -1 & 1 & 3 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 2 \\ 0 & -3 \end{bmatrix}$$



Quando não é preciso pivotar?

- Matriz Diagonal Dominante (DD)

$$|a_{ii}| \geq \sum_{\substack{i=1 \\ i \neq j}}^n |a_{ij}|, \quad i = 1, \dots, n$$

- Matriz Simétrica Positiva Definida (SPD)

$A = A^T$ e $\mathbf{x}^T A \mathbf{x} > 0$, para qualquer vetor não nulo $\mathbf{x} \in \mathbb{R}^n$

$$A = \begin{bmatrix} 4 & 1 \\ 1 & 3 \end{bmatrix} \quad B = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 5 & 1 \\ -1 & 1 & 3 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 2 \\ 0 & -3 \end{bmatrix}$$

A é **SPD**; B é **DD**; C não é **SPD** nem **DD**.



Qual o custo computacional adicional do pivotamento?

Estratégias de Pivotamento:

$$\text{flops}(\textbf{Pivotamento Parcial}) \sim n^2$$

$$\text{flops}(\textbf{Pivotamento Completo}) \sim n^3$$

Eliminação Gaussiana com Pivotamento:

$$\text{flops}(\textbf{Gauss Pivotamento Parcial}) \sim \frac{2}{3} n^3$$

$$\text{flops}(\textbf{Gauss Pivotamento Completo}) \sim \frac{5}{3} n^3$$




Fatos sobre a eliminação gaussiana

1. A eliminação gaussiana pode ser instável, caso a matriz do sistema tenha pivôs pequenos;
2. Estratégias de pivotamento aumentam a estabilidade do algoritmo da eliminação gaussiana;
3. Tipicamente o pivotamento parcial é suficiente para garantir a estabilidade do algoritmo. Mas existem casos (raros) onde só o pivotamento completo a garante estabilidade.




Para pensar em casa ...

Exercício computacional:

Pense num algoritmo eficiente (em termos de processamento e uso de memória) para implementar a eliminação gaussiana c/ pivotamento parcial. Implemente esse algoritmo no ambiente GNU Octave. 

Exercício computacional:

Pense num algoritmo eficiente (em termos de processamento e uso de memória) para implementar a eliminação gaussiana c/ pivotamento completo. Implemente esse algoritmo no ambiente GNU Octave. 



Como citar esse material?

A. Cunha, *Eliminação Gaussiana com Pivotamento*,
Universidade do Estado do Rio de Janeiro – UERJ, 2021.



 @AmericoCunhaJr



@AmericoCunhaJr



@AmericoCunhaJr

Essas notas de aula podem ser compartilhadas nos termos da licença Creative Commons BY-NC-ND 3.0, com propósitos exclusivamente educacionais.

