

# Aula Prática 1 (ALN)

Raphael F. Levy

March 21, 2021

## 1 Análise da função *Gaussian\_Elimination\_1*

Fazendo alguns testes com a função *Gaussian\_Elimination\_1*, temos:

```
1
2 --> A = [1 2 3; 6 4 1; 2 3 5]
3 A =
4
5     1.     2.     3.
6     6.     4.     1.
7     2.     3.     5.
8
9 --> b = [1; 2; 3]
10 b =
11
12     1.
13     2.
14     3.
15
16 --> [x,C]=Gaussian_Elimination_1(A,b)
17 x =
18
19     1.6666667
20    -2.3333333
21     1.3333333
22 C =
23
24     1.     2.     3.
25     6.    -8.   -17.
26     2.    0.125    1.125
27
28 --> A\b
29 ans =
30
31     1.6666667
32    -2.3333333
33     1.3333333
34
35 "OK!"
36 -----
37 --> A = [2 3 -1; -4 0 2; 1 1 3]
38 A =
39
```

```

40      2.    3.   -1.
41     -4.    0.    2.
42      1.    1.    3.
43
44 --> b = [6; -1; 0]
45 b =
46
47      6.
48     -1.
49      0.
50
51 --> [x,C]=Gaussian_Elimination_1(A,b)
52 x =
53
54     -0.047619
55      1.8333333
56     -0.5952381
57 C =
58
59      2.    3.    -1.
60     -2.    6.     0.
61      0.5  -0.0833333  3.5
62
63 --> A\b
64 ans =
65
66     -0.047619
67      1.8333333
68     -0.5952381
69
70 "OK!"
71 -----
72 --> A = [2 0; 1 -1]
73 A =
74
75      2.    0.
76      1.   -1.
77
78 --> b = [2; -1]
79 b =
80
81      2.
82     -1.
83
84 --> [x,C]=Gaussian_Elimination_1(A,b)
85 x =
86
87      1.
88      2.
89 C =
90
91      2.    0.
92      0.5  -1.
93
94 --> A\b
95 ans =
96

```

```

97     1.
98     2.
99
100 "OK!"
101 -----
102 --> A = [1 2; 2 1]
103 A =
104
105     1.    2.
106     2.    1.
107
108 --> b = [1; 0]
109 b =
110
111     1.
112     0.
113
114 --> [x,C]=Gaussian_Elimination_1(A,b)
115 x =
116
117     -0.3333333
118     0.6666667
119 C =
120
121     1.    2.
122     2.   -3.
123
124 --> A\b
125 ans =
126
127     -0.3333333
128     0.6666667
129
130 "OK!"
131 -----
132 --> A = [3 -2; 6 -4]
133 A =
134
135     3.   -2.
136     6.   -4.
137
138 --> b = [9; 18]
139 b =
140
141     9.
142    18.
143
144 --> [x,C]=Gaussian_Elimination_1(A,b)
145 x =
146
147     Nan
148     Nan
149 C =
150
151     3.   -2.
152     2.    0.
153

```

```

154 --> A\b
155 ans =
156
157     3.
158     0.
159
160 "Erro: Deu problema porque o pivoteamento da matriz eliminou a
    linha 2"
161 -----
162 --> A = [2 0; 4 0]
163 A =
164
165     2.    0.
166     4.    0.
167
168 --> b = [2; -1]
169 b =
170
171     2.
172    -1.
173
174 --> [x,C]=Gaussian_Elimination_1(A,b)
175 x =
176
177     Nan
178    -Inf
179 C =
180
181     2.    0.
182     2.    0.
183
184 --> A\b
185 ans =
186
187    -3.831D-17
188     0.

```

Erro: A multiplicação  $Ax = b$  leva a um resultado impossível, já que  $2x = 2$  e simultaneamente  $4x = -1$ , e não é possível que isso seja válido para um único  $x$ . Como a segunda coluna é composta por zeros,  $y$  pode tomar qualquer valor

## 2 Testando a função *Gaussian\_Elimination\_1* com $A1$ e $b1$

Usando a função inicial dada, temos o seguinte resultado para as entradas  $A1$  e  $b1$ :

```

1 --> A1 = [1 -2 5 0; 2 -4 1 3; -1 1 0 2; 0 3 3 1]
2 A1 =
3
4     1.   -2.    5.    0.
5     2.   -4.    1.    3.
6    -1.    1.    0.    2.
7     0.    3.    3.    1.
8

```

```

9 --> b1 = [1; 0; 0; 0]
10 b1 =
11
12     1.
13     0.
14     0.
15     0.
16
17 --> [x,C]=Gaussian_Elimination_1(A1,b1)
18 x =
19
20     Nan
21     Nan
22     Nan
23     Nan
24 C =
25
26     1.    -2.     5.     0.
27     2.     0.    -9.     3.
28    -1.   -Inf   -Inf    Inf
29     0.    Inf     Nan     Nan
30
31 --> A1\b1
32 ans =
33
34    -0.3247863
35    -0.1709402
36     0.1965812
37    -0.0769231

```

Erro: Fazendo o pivoteamento da matriz  $A1$ , é possível ver que o pivô que deveria "aparecer" na segunda linha será 0, e a função *Gaussian\_Elimination\_1* não consegue trabalhar com um pivô 0.

### 3 Testando a *Gaussian\_Elimination\_2* com $A1$ e $b1$ e $A2$ e $b2$

Utilizando a função *Gaussian\_Elimination\_2* desenvolvida para trocar as linhas da matriz caso o elemento na posição  $(j,j)$  seja nulo, temos:

```

1
2 --> A1=[1 -2 5 0; 2 -4 1 3; -1 1 0 2; 0 3 3 1]
3 A1 =
4
5     1.    -2.     5.     0.
6     2.    -4.     1.     3.
7    -1.     1.     0.     2.
8     0.     3.     3.     1.
9
10 --> b1=[1; 0; 0; 0]
11 b1 =
12
13     1.
14     0.
15     0.

```

```

16      0.
17
18 --> [x,C]=Gaussian_Elimination_2(A1,b1)
19 x   =
20
21      -0.3247863
22      -0.1709402
23      0.1965812
24      -0.0769231
25 C   =
26
27      1.   -2.   5.   0.
28      -1.  -1.   5.   2.
29      2.   0.  -9.   3.
30      0.  -3.  -2.  13.
31
32 --> A1\b1
33 ans  =
34
35      -0.3247863
36      -0.1709402
37      0.1965812
38      -0.0769231
39
40 "OK!"
41 -----
42 --> A2=[0 10^-20 1; 10^-20 1 1; 1 2 1]
43 A2   =
44
45      0.          1.000D-20   1.
46      1.000D-20   1.          1.
47      1.          2.          1.
48
49 --> b2=[1; 0; 0]
50 b2   =
51
52      1.
53      0.
54      0.
55
56 --> [x,C]=Gaussian_Elimination_2(A2,b2)
57 x   =
58
59      -1.000D+20
60      0.
61      1.
62 C   =
63
64      1.000D-20   1.          1.
65      0.          1.000D-20   1.
66      1.000D+20  -1.000D+40   1.000D+40
67
68 --> A2\b2
69 ans  =
70
71      1.
72      -1.

```

73

1.

Erro: A função não devolveu o resultado esperado porque haverá uma troca entre a primeira coluna e a segunda, dado que  $A_{2,1}$  é 0, porém o pivô substituto não é o maior encontrado na matriz, e é com ele que a eliminação gausseana é corretamente feita, portanto o resultado foi diferente do esperado.

## 4 Testando a *Gaussian\_Elimination\_3* com $A_2$ e $b_2$ e $A_3$ e $b_3$

Utilizando a função *Gaussian\_Elimination\_3* desenvolvida para trocar as linhas da matriz caso o elemento na posição  $(j,j)$  seja nulo, temos:

```

1
2 --> A2=[0 10^-20 1; 10^-20 1 1; 1 2 1]
3 A2 =
4
5      0.          1.000D-20    1.
6      1.000D-20    1.          1.
7      1.           2.          1.
8
9 --> b2=[1; 0; 0]
10 b2 =
11
12      1.
13      0.
14      0.
15
16 --> [x,C]=Gaussian_Elimination_3(A2,b2)
17 x =
18
19      1.
20     -1.
21      1.
22 C =
23
24      1.           2.          1.
25      1.000D-20    1.          1.
26      0.           1.000D-20    1.
27
28
29 --> A2\b2
30 ans =
31
32      1.
33     -1.
34      1.
35
36 "Ok!"
37 -----
38 --> A3=[10^-20 10^-20 1; 10^-20 1 1; 1 2 1]
39 A3 =
40
41      1.000D-20    1.000D-20    1.

```

```

42      1.000D-20      1.      1.
43      1.      2.      1.
44
45 --> b3=b2
46 b3 =
47
48      1.
49      0.
50      0.
51
52 --> [x,C]=Gaussian_Elimination_3(A3,b3)
53 x =
54
55      0.
56     -1.
57      1.
58 C =
59
60      1.000D-20      1.000D-20      1.
61      1.      1.      0.
62      1.000D+20      1.      -1.000D+20
63
64 --> A3\b3
65 ans =
66
67      1.
68     -1.
69      1.

```

Erro: Apesar de ter pivôs não nulos, nenhum dos dois primeiros é o maior em módulo, já que  $10^{-20} < 1$ . Com isso, a *Gaussian\_Elimination\_3* não fez trocas de linhas já que não há um pivô nulo, e como o maior pivô está na última linha, a eliminação não foi feita corretamente.

## 5 Testando a *Gaussian\_Elimination\_4* com $A3$ e $b3$

Utilizando a função *Gaussian\_Elimination\_4* desenvolvida para sempre escolher a linha com o maior pivô, temos:

```

1
2 --> A3=[10^-20 10^-20 1; 10^-20 1 1; 1 2 1]
3 A3 =
4
5      1.000D-20      1.000D-20      1.
6      1.000D-20      1.      1.
7      1.      2.      1.
8
9 --> b3=b2
10 b3 =
11
12      1.
13      0.
14      0.

```



```

15
16 --> [x,C,P]=Gaussian_Elimination_4(A3,b3)
17 x  =
18
19     1.
20    -1.
21     1.
22 C  =
23
24     1.          2.          1.
25    1.000D-20    1.          1.
26    1.000D-20   -1.000D-20    1.
27 P  =
28
29     0.    0.    1.
30     0.    1.    0.
31     1.    0.    0.
32
33 "Ok!"

```

## 6 Testando a função *Resolve\_com\_LU*

Infelizmente a função *Resolve\_com\_LU* não funcionou corretamente para que os testes pedidos pudessem ser feitos, visto que ela está apenas retornando a matriz B passada.

```

1
2 --> A3=[10^-20 10^-20 1; 10^-20 1 1; 1 2 1]
3 A3  =
4
5     1.000D-20    1.000D-20    1.
6     1.000D-20    1.          1.
7     1.          2.          1.
8
9 --> b3=b2
10 b3  =
11
12     1.
13     0.
14     0.
15
16 --> [x,C1,P1]=Gaussian_Elimination_4(A3,b3)
17 x  =
18
19     1.
20    -1.
21     1.
22 C1  =
23
24     1.          2.          1.
25    1.000D-20    1.          1.
26    1.000D-20   -1.000D-20    1.
27 P1  =
28
29     0.    0.    1.

```

```

30      0.    1.    0.
31      1.    0.    0.
32
33 --> B1 = [2 4 -1 5; 0 1 0 3; 2 2 -1 1; 0 1 1 5]
34 B1 =
35
36      2.    4.   -1.    5.
37      0.    1.    0.    3.
38      2.    2.   -1.    1.
39      0.    1.    1.    5.
40
41 --> [X]=Resolve_com_LU(C1,B1,P1)
42 X =
43
44      2.    4.   -1.    5.
45      0.    1.    0.    3.
46      2.    2.   -1.    1.
47      0.    1.    1.    5.
48 -----
49 --> A2=[0 10^-20 1; 10^-20 1 1; 1 2 1]
50 A2 =
51
52      0      1.000D-20      1.
53      1.000D-20      1.          1.
54      1.          2.          1.
55
56 --> b2=[1; 0; 0]
57 b2 =
58
59      1.
60      0.
61      0.
62
63 --> [x,C2,P2]=Gaussian_Elimination_4(A2,b2)
64 x =
65
66      1.
67     -1.
68      1.
69 C2 =
70
71      1.          2.          1.
72      1.000D-20      1.          1.
73      0.          1.000D-20      1.
74 P2 =
75
76      0.    0.    1.
77      0.    1.    0.
78      1.    0.    0.
79
80 --> B2=[1 1 2; 1 -1 0; 1 0 1]
81 B2 =
82
83      1.    1.    2.
84      1.   -1.    0.
85      1.    0.    1.
86

```

```
87 --> [X]=Resolve_com_LU(Cb,B2,P2)
88 X =
89
90      1.      1.      2.
91      1.     -1.      0.
92      1.      0.      1.
```