

# Aula Prática 3 (ALN)

Raphael F. Levy

April 29, 2021

## 1 Introdução

Para a terceira Aula Prática, foram trabalhados diferentes métodos de potência, que se aplicam a matrizes  $n \times n$  que tenham um autovalor dominante  $\lambda_1$ , e o método age iterativamente para produzir uma sequência de escalares que converge para o autovalor dominante, assim como uma sequência de vetores que convergem para o vetor associado ao autovalor dominante, chamado de autovetor dominante ( $v_1$ ). Para simplificar, consideramos que, para que o método funcione de forma devida, a matriz  $A$  utilizada deve ser diagonalizável ( $D = P^{-1}AP$ ).

Os métodos foram feitos usando algoritmos “padrões” e métodos deslocados, um com iteração inversa e outro usando a iteração de Rayleigh, sendo que os últimos usam respectivamente um valor para achar o autovalor de  $A$  mais próximo sem calcular a matriz inversa, enquanto o outro usa o quociente de Rayleigh:  $\frac{x_k^T A x_k}{x_k^T x_k} = \frac{x_k + 1^T x_k}{x_k^T x_k}$

## 2 Testando o método da Potência (Questão 1)

Ambas as funções criadas para determinar o autovalor dominante da matriz  $A$  através do Método da Potência (Versão 1 e 2) estão no arquivo “Metodo\_da\_Potencia.sci”.

Testando com diferentes matrizes e vetores iniciais teremos:

```
1 --> A=[3 1; 1 3] (Autovalores: 4,2)
2 A =
3
4     3.    1.
5     1.    3.
6
7 --> x0=[14;30]
8 x0 =
9
10    14.
11    30.
12
13 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
14
```

```

15 "0 metodo convergiu com a precisao passada"
16 lambda =
17
18     4.0000000
19 x1 =
20
21     1.
22     1.
23 k =
24
25     34.
26 n_erro =
27
28     8.467D-11
29
30 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
31
32 "0 metodo convergiu com a precisao passada"
33 lambda =
34
35     4.
36 x1 =
37
38     0.7071068
39     0.7071068
40 k =
41
42     32.
43 n_erro =
44
45     8.467D-11
46 -----
47
48 --> A=[5 2; 2 5] (Autovalores: 7,3)
49 A =
50
51     5.    2.
52     2.    5.
53
54 --> x0=[1;3]
55 x0 =
56
57     1.
58     3.
59
60 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
61
62 "0 metodo convergiu com a precisao passada"
63 lambda =
64
65     7.0000000
66 x1 =
67
68     1.
69     1.
70 k =
71

```

```

72     29.
73     n_erro  =
74
75     6.631D-11
76
77 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
78
79     "0 metodo convergiu com a precisao passada"
80     lambda  =
81
82     7.
83     x1  =
84
85     0.7071068
86     0.7071068
87     k  =
88
89     27.
90     n_erro  =
91
92     7.736D-11
93
94 -----
95
96 --> A=[-1 3 -1; -3 5 -1; -3 3 1] (Autovalores: 2, 2, 1)
97     A  =
98
99     -1.    3.   -1.
100    -3.    5.   -1.
101    -3.    3.    1.
102
103 --> x0=[1;3;5]
104     x0  =
105
106     1.
107     3.
108     5.
109
110 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
111
112     "0 metodo convergiu com a precisao passada"
113     lambda  =
114
115     2.0000000
116     x1  =
117
118     0.3333333
119     0.6666667
120     1.
121     k  =
122
123     33.
124     n_erro  =
125
126     5.174D-11
127
128 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)

```

```

129
130     "0 metodo convergiu com a precisao passada"
131     lambda =
132
133         2.0000000
134     x1 =
135
136         0.2672612
137         0.5345225
138         0.8017837
139     k =
140
141         30.
142     n_erro =
143
144         8.147D-11
145
146     -----
147
148 --> A=[1 2 0; 0 3 0; 2 -4 2] (Autovalores: 3, 2, 1)
149     A =
150
151         1.    2.    0.
152         0.    3.    0.
153         2.   -4.    2.
154
155 --> x0=[3;4;5]
156     x0 =
157
158         3.
159         4.
160         5.
161
162 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
163
164     "0 metodo convergiu com a precisao passada"
165     lambda =
166
167         3.0000000
168     x1 =
169
170         -0.5
171         -0.5
172         1.
173     k =
174
175         57.
176     n_erro =
177
178         7.099D-11
179
180 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
181
182     "0 metodo convergiu com a precisao passada"
183     lambda =
184
185         3.0000000

```

```

186 x1 =
187
188     0.4082483
189     0.4082483
190    -0.8164966
191 k =
192
193     55.
194 n_erro =
195
196     6.693D-11
197
198 -----
199
200 --> A=[1 0 2 0; 2 1 1 1; 2 3 0 1; -1 1 2 2] (Observa o: CONTEM
      AUTOVALORES IMAGINARIOS!!)
201                                     (Autovalores: 4.56,
      1.89, 1 .23+0.833i, 1 .23-0.833i)
202 A =
203
204     1.    0.    2.    0.
205     2.    1.    1.    1.
206     2.    3.    0.    1.
207    -1.    1.    2.    2.
208
209 --> x0=[1;2;3;4]
210 x0 =
211
212     1.
213     2.
214     3.
215     4.
216
217 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
218
219     "0 metodo convergiu com a precisao passada"
220 lambda =
221
222     4.5594454
223 x1 =
224
225     0.5618853
226     0.8475481
227     1.
228     0.8930305
229 k =
230
231     29.
232 n_erro =
233
234     8.890D-11
235
236 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
237
238     "0 metodo convergiu com a precisao passada"
239 lambda =
240

```

```

241      4.5594454
242      x1 =
243
244      0.3339143
245      0.5036766
246      0.5942749
247      0.5307056
248      k =
249
250      26.
251      n_erro =
252
253      4.985D-11
254
255      -----
256
257      --> A=[0 1; 0 0] (Autovalor: 0) (Nao diagonalizavel)
258      A =
259
260      0.    1.
261      0.    0.
262
263      --> x0=[3;4]
264      x0 =
265
266      3.
267      4.
268
269      --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
270
271      "0 numero de iteracoes passou do maximo de iteracoes permitido"
272      lambda =
273
274      Nan
275      x1 =
276
277      Nan
278      Nan
279      k =
280
281      101.
282      n_erro =
283
284      Nan
285
286      --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
287
288      "0 numero de iteracoes passou do maximo de iteracoes permitido"
289      lambda =
290
291      Nan
292      x1 =
293
294      Nan
295      Nan
296      k =
297

```

```

298     101.
299     n_erro  =
300
301     Nan
302
303
304 -----
305
306 --> A=[1 1; -1 3] (Autovalor: 2)
307     A  =
308
309         1.    1.
310        -1.    3.
311
312 --> x0=[1;2]
313     x0  =
314
315         1.
316         2.
317
318 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
319
320 "0 numero de iteracoes passou do maximo de iteracoes permitido"
321     lambda  =
322
323         2.0196078
324     x1  =
325
326         1.9805825
327         2.0194175
328     k  =
329
330         101.
331     n_erro  =
332
333         0.0001904
334
335 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
336
337 "0 numero de iteracoes passou do maximo de iteracoes permitido"
338     lambda  =
339
340         2.0196060
341     x1  =
342
343         1.4141469
344         1.4416061
345     k  =
346
347         101.
348     n_erro  =
349
350         0.0000952
351
352 -----
353
354 --> A=[2 1; 1 2] (Autovalores: 3, 1)

```

```

355 A =
356
357     2.    1.
358     1.    2.
359
360 --> x0=[1;3]
361 x0 =
362
363     1.
364     3.
365
366 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
367
368 "0 metodo convergiu com a precisao passada"
369 lambda =
370
371     3.0000000
372 x1 =
373
374     1.
375     1.
376 k =
377
378     23.
379 n_erro =
380
381     6.373D-11
382
383 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
384
385 "0 metodo convergiu com a precisao passada"
386 lambda =
387
388     3.
389 x1 =
390
391     0.7071068
392     0.7071068
393 k =
394
395     21.
396 n_erro =
397
398     9.560D-11

```

Através de alguns testes, foi possível notar que, como esperado, sendo  $A$  diagonalizável, o método irá convergir, caso contrário, é possível que o número de iterações passe do permitido e o lambda correto não seja encontrado. É possível notar também que os métodos podem não achar os autovalores corretos para matrizes com menos autovalores que sua ordem (por exemplo, matrizes 2x2 com um autovalor, 3x3 com dois autovalores, etc).

Testando os tempos para diferentes números de iterações com variáveis de entrada constantes:

```

1 --> A=[-1 3 -1; -3 5 -1; -3 3 1]

```



```

2  A  =
3
4  -1.   3.  -1.
5  -3.   5.  -1.
6  -3.   3.   1.
7
8  --> x0=[1;3;5]
9  x0  =
10
11     1.
12     3.
13     5.
14
15  --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
      ;t1=toc()
16
17     "0 metodo convergiu com a precisao passada"
18  t1  =
19
20     0.0009175
21
22  --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),200)
      ;t2=toc()
23
24     "0 metodo convergiu com a precisao passada"
25  t2  =
26
27     0.0007797
28
29  --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),300)
      ;t3=toc()
30
31     "0 metodo convergiu com a precisao passada"
32  t3  =
33
34     0.0010204
35
36  --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),400)
      ;t4=toc()
37
38     "0 metodo convergiu com a precisao passada"
39  t4  =
40
41     0.0007966
42
43  --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),500)
      ;t5=toc()
44
45     "0 metodo convergiu com a precisao passada"
46  t5  =
47
48     0.0006009
49
50  --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),600)
      ;t6=toc()
51
52     "0 metodo convergiu com a precisao passada"

```

```

53 t6 =
54
55     0.0015199
56
57 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),700)
    ;t7=toc()
58
59 "0 metodo convergiu com a precisao passada"
60 t7 =
61
62     0.0010745
63
64 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),800)
    ;t8=toc()
65
66 "0 metodo convergiu com a precisao passada"
67 t8 =
68
69     0.0006365
70
71 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),900)
    ;t9=toc()
72
73 "0 metodo convergiu com a precisao passada"
74 t9 =
75
76     0.0009277
77
78 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10)
    ,1000);t10=toc()
79
80 "0 metodo convergiu com a precisao passada"
81 t10 =
82
83     0.0008272
84
85 -----
86
87 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
    ;t11=toc()
88
89 "0 metodo convergiu com a precisao passada"
90 t11 =
91
92     0.0005534
93
94 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),200)
    ;t12=toc()
95
96 "0 metodo convergiu com a precisao passada"
97 t12 =
98
99     0.0007731
100
101 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),300)
    ;t13=toc()
102

```

```

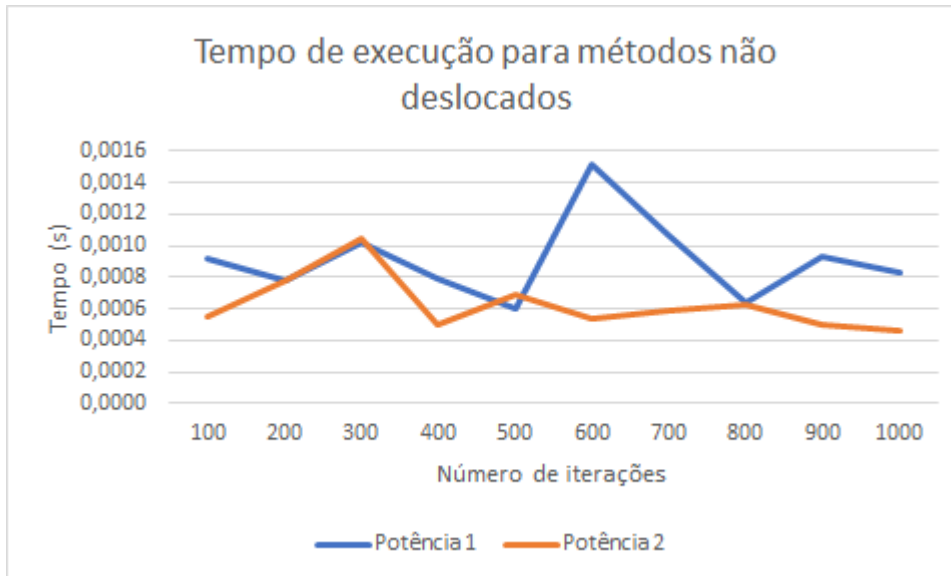
103 "0 metodo convergiu com a precisao passada"
104 t13 =
105
106 0.0010449
107
108 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),400)
      ;t14=toc()
109
110 "0 metodo convergiu com a precisao passada"
111 t14 =
112
113 0.0005043
114
115 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),500)
      ;t15=toc()
116
117 "0 metodo convergiu com a precisao passada"
118 t15 =
119
120 0.0006954
121
122 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),600)
      ;t16=toc()
123
124 "0 metodo convergiu com a precisao passada"
125 t16 =
126
127 0.0005419
128
129 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),700)
      ;t17=toc()
130
131 "0 metodo convergiu com a precisao passada"
132 t17 =
133
134 0.0005887
135
136 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),800)
      ;t18=toc()
137
138 "0 metodo convergiu com a precisao passada"
139 t18 =
140
141 0.0006297
142
143 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),900)
      ;t19=toc()
144
145 "0 metodo convergiu com a precisao passada"
146 t19 =
147
148 0.0005034
149
150 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10)
      ,1000);t20=toc()
151
152 "0 metodo convergiu com a precisao passada"

```

```

153 t20 =
154
155 0.0004615

```



Tempo de execução dos métodos não deslocados para as entradas passadas, alterando  $M$

Pela imagem, é possível ver que, mesmo que próximos, o Método da Potência 2 costuma ter tempos mais rápidos que o Método da Potência 1, já que calcular o quociente de Rayleigh é mais rápido que encontrar a coordenada de maior módulo no vetor.

### 3 Questão 4

Vamos usar novamente a matriz diagonalizável  $A = [-1, 3, -1; -3, 5, -1; -3, 3, 1]$ , cujos autovalores são  $\lambda = 1, 2, 2$ , para comparar o número de iterações necessárias e o tempo de execução. Para fazer da mesma maneira que foi feita anteriormente, usaremos  $x_0 = [1; 3; 5]$ .

```

1 --> A=[-1 3 -1; -3 5 -1; -3 3 1] (Autovalores: 2, 1)
2 A =
3
4     -1.     3.     -1.
5     -3.     5.     -1.
6     -3.     3.      1.
7
8 --> x0=[1; 3; 5]
9 x0 =
10

```

```

11     1.
12     3.
13     5.
14
15 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
16
17     "0 metodo convergiu com a precisao passada"
18     lambda =
19
20         2.00000000
21     x1 =
22
23         0.33333333
24         0.66666667
25         1.
26     k =
27
28         33.
29     n_erro =
30
31         5.174D-11
32
33 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
34
35     "0 metodo convergiu com a precisao passada"
36     lambda =
37
38         2.00000000
39     x1 =
40
41         0.2672612
42         0.5345225
43         0.8017837
44     k =
45
46         30.
47     n_erro =
48
49         8.147D-11
50
51 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
52         ; t1=toc()
53
54     "0 metodo convergiu com a precisao passada"
55     t1 =
56
57         0.0008487
58 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
59         ; t2=toc()
60
61     "0 metodo convergiu com a precisao passada"
62     t2 =
63
64         0.0004305

```

Usando essa matriz e autovetor, é possível ver que o algoritmo 1, que usa a

coordenada de maior módulo de  $x_1$  como  $\lambda$ , foi feito com mais iterações e levou mais tempo, aproximadamente o dobro, que o algoritmo 2, que usa  $\lambda$  como o quociente de Rayleigh:  $\lambda = x_0^T * x_1$ . Com isso, fica claro que usar o quociente de Rayleigh torna o processo mais simples e consequentemente rápido, já que o algoritmo não precisa encontrar a coordenada de maior módulo do autovetor a cada iteração.

## 4 Testando os métodos da Potência Deslocada (Questões 2 e 3)

Ambas as funções criadas para determinar o autovalor dominante da matriz  $A$  mais próximo do alfa passado através do Método da Potência Deslocada (Iteração Inversa e Método de Rayleigh) estão no arquivo “Metodo\_da\_Potencia\_Deslocada.sci”.

Testando com diferentes matrizes e vetores iniciais teremos:

```

1 --> A=[3 1; 1 3] (Autovalores: 4, 2)
2 A =
3
4     3.    1.
5     1.    3.
6
7 --> x0=[14;30]
8 x0 =
9
10    14.
11    30.
12
13 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
14     ,5,100)
15
16 "0 numero de iteracoes passou do maximo de iteracoes permitido"
17 lambda =
18
19     4.
20
21 x1 =
22
23     0.7071068
24     0.7071068
25
26 k =
27
28    101.
29
30 n_erro =
31
32     2.
33
34 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
35     ,4,100)
36
37 "0 numero de iteracoes passou do maximo de iteracoes permitido"
38 lambda =
39
40    Nan

```

```

36 x1 =
37
38     Nan
39     Nan
40 k =
41
42     101.
43 n_erro =
44
45     Nan
46
47 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,3,100)
48
49 "0 numero de iteracoes passou do maximo de iteracoes permitido"
50 lambda =
51
52     3.7664234
53 x1 =
54
55     0.4228855
56     0.9061831
57 k =
58
59     101.
60 n_erro =
61
62     0.6834861
63
64 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,2,100)
65
66 "0 numero de iteracoes passou do maximo de iteracoes permitido"
67 lambda =
68
69     Nan
70 x1 =
71
72     Nan
73     Nan
74 k =
75
76     101.
77 n_erro =
78
79     Nan
80
81 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,1,100)
82
83 "0 metodo convergiu com a precisao passada"
84 lambda =
85
86     2.
87 x1 =
88
89     -0.7071068

```

```

90     0.7071068
91     k  =
92
93     23.
94     n_erro  =
95
96     5.842D-11
97
98 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
    ,5,100)
99
100 "0 metodo convergiu com a precisao passada"
101 lambda  =
102
103     4.0000000
104     x1  =
105
106     -0.7071068
107     -0.7071068
108     k  =
109
110     5.
111     n_erro  =
112
113     0.
114
115 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
    ,4,100)
116
117 "0 numero de iteracoes passou do maximo de iteracoes permitido"
118 lambda  =
119
120     Nan
121     x1  =
122
123     Nan
124     Nan
125     k  =
126
127     101.
128     n_erro  =
129
130     Nan
131
132 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
    ,3,100)
133
134 "0 numero de iteracoes passou do maximo de iteracoes permitido"
135 lambda  =
136
137     Nan
138     x1  =
139
140     Nan
141     Nan
142     k  =
143

```



```

144     101.
145     n_erro  =
146
147     Nan
148
149 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,2,100)
150
151     "0 numero de iteracoes passou do maximo de iteracoes permitido"
152     lambda  =
153
154     Nan
155     x1  =
156
157     Nan
158     Nan
159     k  =
160
161     101.
162     n_erro  =
163
164     Nan
165
166 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,1,100)
167
168     "0 numero de iteracoes passou do maximo de iteracoes permitido"
169     lambda  =
170
171     Nan
172     x1  =
173
174     Nan
175     Nan
176     k  =
177
178     101.
179     n_erro  =
180
181     Nan
182
183 -----
184
185 --> A=[1 2 0; 0 3 0; 2 -4 2] (Autovalores: 3, 2, 1)
186     A  =
187
188     1.    2.    0.
189     0.    3.    0.
190     2.   -4.    2.
191
192 --> x0=[3;4;5]
193     x0  =
194
195     3.
196     4.
197     5.
198

```

```

199 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,5,100)
200
201 "0 numero de iteracoes passou do maximo de iteracoes permitido"
202 lambda =
203
204     3.
205 x1 =
206
207     0.4082483
208     0.4082483
209    -0.8164966
210 k =
211
212    101.
213 n_erro =
214
215     2.
216
217 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,4,100)
218
219 "0 numero de iteracoes passou do maximo de iteracoes permitido"
220 lambda =
221
222     3.
223 x1 =
224
225     0.4082483
226     0.4082483
227    -0.8164966
228 k =
229
230    101.
231 n_erro =
232
233     2.
234
235 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,3,100)
236
237 "0 numero de iteracoes passou do maximo de iteracoes permitido"
238 lambda =
239
240     Nan
241 x1 =
242
243     Nan
244     Nan
245     Nan
246 k =
247
248    101.
249 n_erro =
250
251     Nan
252

```

```

253 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,2,100)
254
255 "0 numero de iteracoes passou do maximo de iteracoes permitido"
256 lambda =
257     Nan
258 x1 =
259     Nan
260
261     Nan
262     Nan
263     Nan
264 k =
265
266     101.
267 n_erro =
268
269     Nan
270
271 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,1,100)
272
273 "0 numero de iteracoes passou do maximo de iteracoes permitido"
274 lambda =
275     Nan
276 x1 =
277     Nan
278
279     Nan
280     Nan
281     Nan
282 k =
283
284     101.
285 n_erro =
286
287     Nan
288
289 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,0,100)
290
291 "0 metodo convergiu com a precisao passada"
292 lambda =
293     1.0000000
294 x1 =
295     -0.4472136
296     2.776D-17
297     0.8944272
298 k =
299
300     35.
301 n_erro =
302
303     6.403D-11
304
305
306

```

```

307 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,5,100)
308
309 "0 numero de iteracoes passou do maximo de iteracoes permitido"
310 lambda =
311
312     Nan
313 x1 =
314
315     Nan
316     Nan
317     Nan
318 k =
319
320     101.
321 n_erro =
322
323     Nan
324
325 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,4,100)
326
327 "0 metodo convergiu com a precisao passada"
328 lambda =
329
330     3.00000000
331 x1 =
332
333     -0.4082483
334     -0.4082483
335     0.8164966
336 k =
337
338     7.
339 n_erro =
340
341     6.301D-14
342
343 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,3,100)
344
345 "0 numero de iteracoes passou do maximo de iteracoes permitido"
346 lambda =
347
348     Nan
349 x1 =
350
351     Nan
352     Nan
353     Nan
354 k =
355
356     101.
357 n_erro =
358
359     Nan
360

```

```

361 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,2,100)
362
363 "0 numero de iteracoes passou do maximo de iteracoes permitido"
364 lambda =
365     Nan
366 x1 =
367     Nan
368     Nan
369     Nan
370     Nan
371 k =
372     101.
373 n_erro =
374     Nan
375
376 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,1,100)
377
378 "0 numero de iteracoes passou do maximo de iteracoes permitido"
379 lambda =
380     Nan
381 x1 =
382     Nan
383     Nan
384     Nan
385     Nan
386 k =
387     101.
388 n_erro =
389     Nan
390
391 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,0,100)
392
393 "0 metodo convergiu com a precisao passada"
394 lambda =
395     1.0000000
396 x1 =
397     0.4472136
398     8.225D-23
399     -0.8944272
400 k =
401     7.
402 n_erro =
403     7.206D-11
404

```

```

415 -----
416
417 --> A=[0 2 -3 5; 2 4 1 3; -3 1 1 4; 5 3 4 2] (Autovalores: ~9.205,
418 ~3.563, ~1.295, ~-7.063)
419 A =
420     0.    2.   -3.    5.
421     2.    4.    1.    3.
422    -3.    1.    1.    4.
423     5.    3.    4.    2.
424
425 --> x0=[1;2;3;4]
426 x0 =
427
428     1.
429     2.
430     3.
431     4.
432
433 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
434 ,10,100)
435
436 "0 numero de iteracoes passou do maximo de iteracoes permitido"
437 lambda =
438     9.2050871
439 x1 =
440
441     0.405511
442     0.5832051
443     0.2445709
444     0.6600134
445 k =
446
447    101.
448 n_erro =
449
450     2.
451
452 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
453 ,9,100)
454
455 "0 metodo convergiu com a precisao passada"
456 lambda =
457     9.2050871
458 x1 =
459
460     0.405511
461     0.5832051
462     0.2445709
463     0.6600134
464 k =
465
466     8.
467 n_erro =
468

```

```

469      4.979D-11
470
471 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,4,100)
472
473      "0 numero de iteracoes passou do maximo de iteracoes permitido"
474      lambda =
475
476      3.5634895
477      x1 =
478
479      -0.5648791
480      -0.0809848
481      0.812747
482      0.117454
483      k =
484
485      101.
486      n_erro =
487
488      2.0000000
489
490 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,3,100)
491
492      "0 metodo convergiu com a precisao passada"
493      lambda =
494
495      3.5634895
496      x1 =
497
498      -0.5648791
499      -0.0809848
500      0.812747
501      0.117454
502      k =
503
504      22.
505      n_erro =
506
507      4.259D-11
508
509 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,2,100)
510
511      "0 numero de iteracoes passou do maximo de iteracoes permitido"
512      lambda =
513
514      1.2948071
515      x1 =
516
517      0.345427
518      -0.8082635
519      0.0919221
520      0.4679109
521      k =
522

```

```

523     101.
524     n_erro  =
525
526     2.
527
528 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,1,100)
529
530     "0 metodo convergiu com a precisao passada"
531     lambda  =
532
533     1.2948071
534     x1  =
535
536     0.345427
537     -0.8082635
538     0.0919221
539     0.4679109
540     k  =
541
542     13.
543     n_erro  =
544
545     1.179D-11
546
547 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,-7,100)
548
549     "0 metodo convergiu com a precisao passada"
550     lambda  =
551
552     -7.0633837
553     x1  =
554
555     -0.6302005
556     0.0048345
557     -0.5207473
558     0.5758873
559     k  =
560
561     7.
562     n_erro  =
563
564     1.691D-12
565
566 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,10,100)
567
568     "0 metodo convergiu com a precisao passada"
569     lambda  =
570
571     9.2050871
572     x1  =
573
574     0.405511
575     0.5832051
576     0.2445709

```



```

577     0.6600134
578     k   =
579
580     4.
581     n_erro   =
582
583     6.460D-12
584
585 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,9,100)
586
587     "0 metodo convergiu com a precisao passada"
588     lambda   =
589
590     9.2050871
591     x1   =
592
593     0.405511
594     0.5832051
595     0.2445709
596     0.6600134
597     k   =
598
599     4.
600     n_erro   =
601
602     1.665D-16
603
604 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,4,100)
605
606     "0 metodo convergiu com a precisao passada"
607     lambda   =
608
609     3.5634895
610     x1   =
611
612     0.5648791
613     0.0809848
614     -0.812747
615     -0.117454
616     k   =
617
618     5.
619     n_erro   =
620
621     1.119D-16
622
623 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,3,100)
624
625     "0 metodo convergiu com a precisao passada"
626     lambda   =
627
628     3.5634895
629     x1   =
630

```

```

631     0.5648791
632     0.0809848
633     -0.812747
634     -0.117454
635     k =
636
637     5.
638     n_erro =
639
640     1.127D-16
641
642 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
        ,2,100)
643
644     "0 metodo convergiu com a precisao passada"
645     lambda =
646
647     3.5634895
648     x1 =
649
650     -0.5648791
651     -0.0809848
652     0.812747
653     0.117454
654     k =
655
656     6.
657     n_erro =
658
659     0.
660
661 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
        ,1,100)
662
663     "0 metodo convergiu com a precisao passada"
664     lambda =
665
666     1.2948071
667     x1 =
668
669     -0.345427
670     0.8082635
671     -0.0919221
672     -0.4679109
673     k =
674
675     5.
676     n_erro =
677
678     6.883D-11
679
680 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
        ,-7,100)
681
682     "0 metodo convergiu com a precisao passada"
683     lambda =
684

```

```

685 -7.0633837
686 x1 =
687
688 0.6302005
689 -0.0048345
690 0.5207473
691 -0.5758873
692 k =
693
694 5.
695 n_erro =
696
697 1.118D-16

```

Usando métodos deslocados, é possível notar que não basta mais que a matriz seja diagonalizável para que o método convirja. No caso, é preciso que o alfa passado seja menor ou igual que os autovalores da matriz.

Observando os testes passados, é notável que se o alfa passado for igual ao autovvalor da matriz, os métodos irão falhar. Contudo, é perceptível que o método de Rayleigh não funciona corretamente sempre, como é possível ver no último exemplo mostrado, em que o lambda esperado era -7.0633837 e o método devolveu 3.5634895.

Testando os tempos para diferentes *alfas* com variáveis de entrada constantes:

```

1 --> A=[0 2 -3 5; 2 4 1 3; -3 1 1 4; 5 3 4 2] (Autovalores: ~9.205,
2 ~3.563, ~1.295, ~-7.063)
3
4 A =
5
6 0. 2. -3. 5.
7 2. 4. 1. 3.
8 -3. 1. 1. 4.
9 5. 3. 4. 2.
10
11 --> x0=[1;2;3;4]
12 x0 =
13
14 1.
15 2.
16 3.
17 4.
18
19 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
20 ,10^(-10),10,100);t1=toc()
21
22 "0 numero de iteracoes passou do maximo de iteracoes permitido"
23 t1 =
24
25 0.0264886
26
27 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
28 ,10^(-10),9,100);t2=toc()
29
30 "0 metodo convergiu com a precisao passada"
31 t2 =

```

```

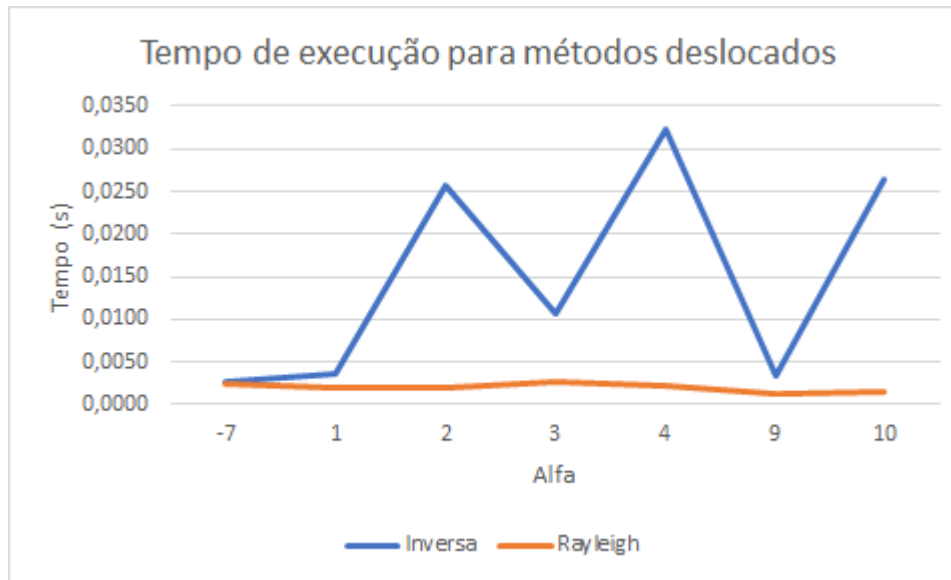
28      0.0033274
29
30
31 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),4,100);t3=toc()
32
33      "0 numero de iteracoes passou do maximo de iteracoes permitido"
34 t3 =
35
36      0.0322190
37
38 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),3,100);t4=toc()
39
40      "0 metodo convergiu com a precisao passada"
41 t4 =
42
43      0.0105523
44
45 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),2,100);t5=toc()
46
47      "0 numero de iteracoes passou do maximo de iteracoes permitido"
48 t5 =
49
50      0.0255967
51
52 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),1,100);t6=toc()
53
54      "0 metodo convergiu com a precisao passada"
55 t6 =
56
57      0.0035725
58
59 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),-7,100);t7=toc()
60
61      "0 metodo convergiu com a precisao passada"
62 t7 =
63
64      0.0026467
65
66 -----
67
68 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),10,100);t8=toc()
69
70      "0 metodo convergiu com a precisao passada"
71 t8 =
72
73      0.0014118
74
75 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),9,100);t9=toc()
76
77      "0 metodo convergiu com a precisao passada"

```

```

78 t9 =
79
80     0.0013488
81
82 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),4,100);t10=toc()
83
84     "0 metodo convergiu com a precisao passada"
85 t10 =
86
87     0.0021528
88
89 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),3,100);t11=toc()
90
91     "0 metodo convergiu com a precisao passada"
92 t11 =
93
94     0.0027536
95
96 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),2,100);t12=toc()
97
98     "0 metodo convergiu com a precisao passada"
99 t12 =
100
101     0.0018643
102
103 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),1,100);t13=toc()
104
105     "0 metodo convergiu com a precisao passada"
106 t13 =
107
108     0.0020644
109
110 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),-7,100);t14=toc()
111
112     "0 metodo convergiu com a precisao passada"
113 t14 =
114
115     0.0023858

```

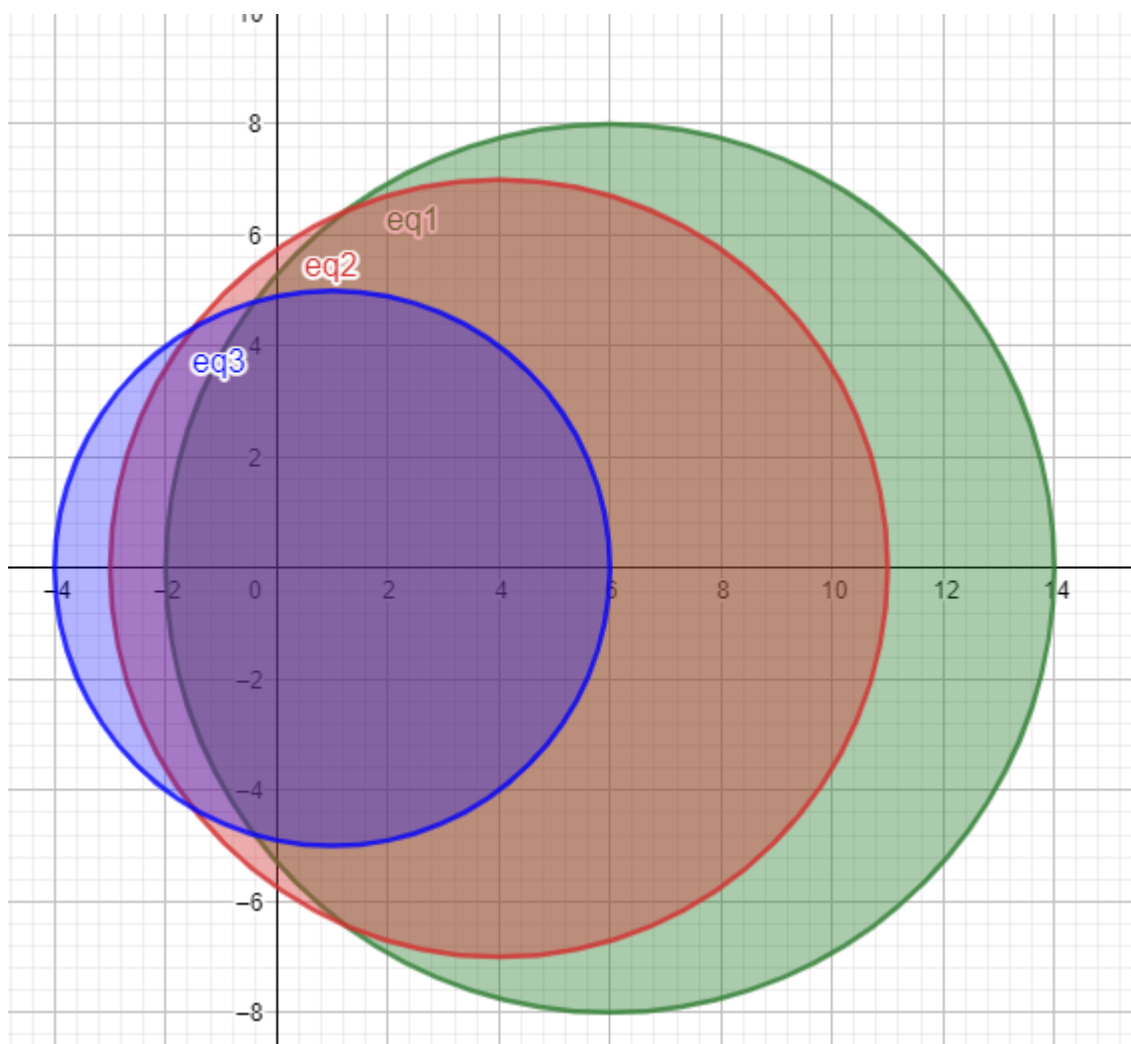


Tempo de execução dos métodos deslocados para as entradas passadas, alterando *alfa*

Pelo gráfico, é possível ver que o algoritmo de Rayleigh foi bem mais rápido que o método deslocado inverso, embora o método inverso apresente tanto picos de 0,035 segundos quanto vales de aproximadamente 0,005, se aproximando do método de Rayleigh. Isso ocorre pois o algoritmo de Rayleigh atualiza o *alfa* a cada iteração, e portanto consegue chegar mais perto do autovalor de forma mais rápida que o método deslocado inverso.

## 5 Questão 5

Para essa questão, vamos usar a matriz simétrica  $3 \times 3$   $A = [1, 2, 3; 2, 4, 5; 3, 5, 6]$  e o autovetor  $x_0 = [1; 2; 3]$ , para encontrar os discos de Gerschgorin dessa matriz, de forma a estimar a localização de seus autovalores. Pelo Teorema de Gerschgorin ( $A = [a_{ij}]_{n \times n} \Rightarrow r_i = \sum_{j \neq i} |a_{ij}|$ ), os autovalores dessa matriz estão localizados no eixo dos reais dentro de discos com centros em  $a_{ii}$ , ou seja, em 1, 4 e 6, e raios respectivamente iguais a 5, 7 e 8.



Discos de Gershgorin calculados

- Disco azul:  $|z - 6| \leq 8$
- Disco vermelho:  $|z - 4| \leq 7$
- Disco cinza:  $|z - 1| \leq 5$

```

1 --> A=[1 2 3; 2 4 5; 3 5 6]
2   A  =
3
4       1.    2.    3.
5       2.    4.    5.
6       3.    5.    6.
7
8 --> x0=[1;2;3]
9   x0  =
10
11      1.
12      2.
13      3.
14
15 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
16      ,1,100)
17
18      "0 numero de iteracoes passou do maximo de iteracoes permitido"
19      lambda  =
20
21      0.1709152
22      x1  =
23
24      -0.591009
25      0.7369762
26      -0.3279853
27      k  =
28
29      101.
30      n_erro  =
31
32      2.0000000
33 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
34      ,10^(-10),1,100); t1=toc()
35
36      "0 numero de iteracoes passou do maximo de iteracoes permitido"
37      t1  =
38
39      0.0219016
40 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
41      ,4,100)
42
43      "0 numero de iteracoes passou do maximo de iteracoes permitido"
44      lambda  =
45
46      0.1709152
47      x1  =
48
49      -0.5910089

```



```

49      0.7369763
50      -0.3279854
51      k      =
52
53      101.
54      n_erro  =
55
56      2.0000000
57
58 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),4,100); t2=toc()
59
60      "0 numero de iteracoes passou do maximo de iteracoes permitido"
61      t2      =
62
63      0.0173268
64
65 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,6,100)
66
67      "0 numero de iteracoes passou do maximo de iteracoes permitido"
68      lambda  =
69
70      11.344814
71      x1      =
72
73      0.327988
74      0.5910056
75      0.7369778
76      k      =
77
78      101.
79      n_erro  =
80
81      0.0000097
82
83 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),6,100); t3=toc()
84
85      "0 numero de iteracoes passou do maximo de iteracoes permitido"
86      t3      =
87
88      0.0178859
89
90 -----
91
92 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,1,100)
93
94      "0 metodo convergiu com a precisao passada"
95      lambda  =
96
97      11.344814
98      x1      =
99
100     0.3279853
101     0.591009

```

```

102     0.7369762
103     k   =
104
105     7.
106     n_erro   =
107
108     0.
109
110 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),1,100); t4=toc()
111
112     "0 metodo convergiu com a precisao passada"
113     t4   =
114
115     0.0016335
116
117 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,4,100)
118
119     "0 metodo convergiu com a precisao passada"
120     lambda   =
121
122     11.344814
123     x1   =
124
125     -0.3279853
126     -0.591009
127     -0.7369762
128     k   =
129
130     5.
131     n_erro   =
132
133     0.
134
135 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),4,100); t5=toc()
136
137     "0 metodo convergiu com a precisao passada"
138     t5   =
139
140     0.0014949
141
142 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,6,100)
143
144     "0 metodo convergiu com a precisao passada"
145     lambda   =
146
147     11.344814
148     x1   =
149
150     0.3279853
151     0.591009
152     0.7369762
153     k   =
154

```

```

155     5.
156     n_erro =
157
158     0.
159
160 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
      ,10^(-10),6,100); t6=toc()
161
162 "0 metodo convergiu com a precisao passada"
163 t6 =
164
165     0.0015383

```

Calculando o polinômio característico dessa matriz, chegamos a  $-\lambda^3 + 11\lambda^2 + 4\lambda - 1 = 0$ , e fazendo essa conta, achamos  $\lambda_1 = 11.344814282762078$ ,  $\lambda_2 = 0.17091518882718093$  e  $\lambda_3 = -0.5157294715892573$ . Sendo assim, é possível ver que os métodos deslocados foram boas aproximações para achar os autovalores da matriz, com exceção do autovalor negativo, que não foi encontrado. Assim, usaremos um *alfa* diferente de  $a_{ii}$  para tentar encontrar o autovalor restante:

```

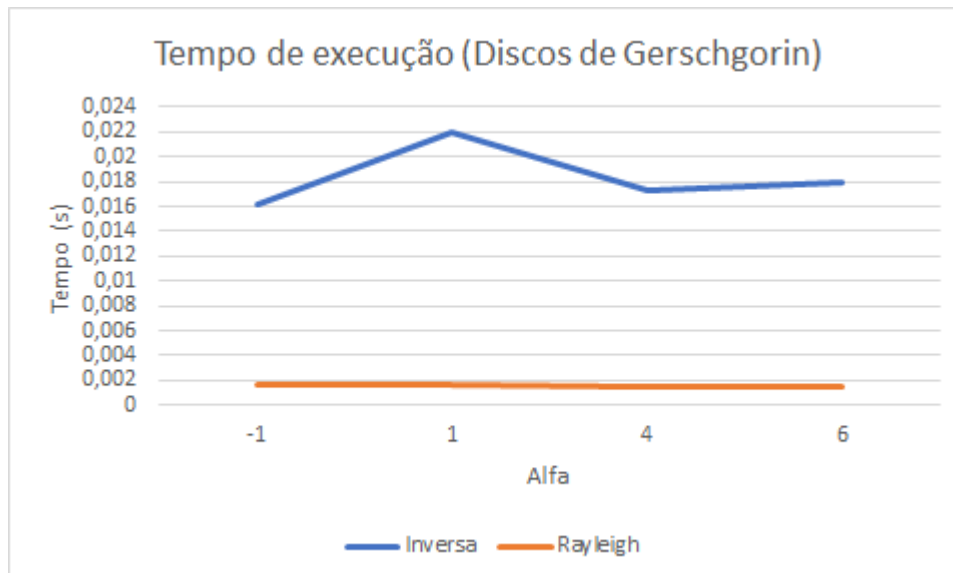
1
2 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,-1,100)
3
4 "0 numero de iteracoes passou do maximo de iteracoes permitido"
5 lambda =
6
7     -0.5157295
8 x1 =
9
10     0.7369762
11     0.3279853
12     -0.591009
13 k =
14
15     101.
16 n_erro =
17
18     2.
19
20 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
      ,10^(-10),-1,100); t7=toc()
21
22 "0 numero de iteracoes passou do maximo de iteracoes permitido"
23 t7 =
24
25     0.0162406
26
27 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,-1,100)
28
29 "0 metodo convergiu com a precisao passada"
30 lambda =
31
32     -0.5157295
33 x1 =
34

```

```

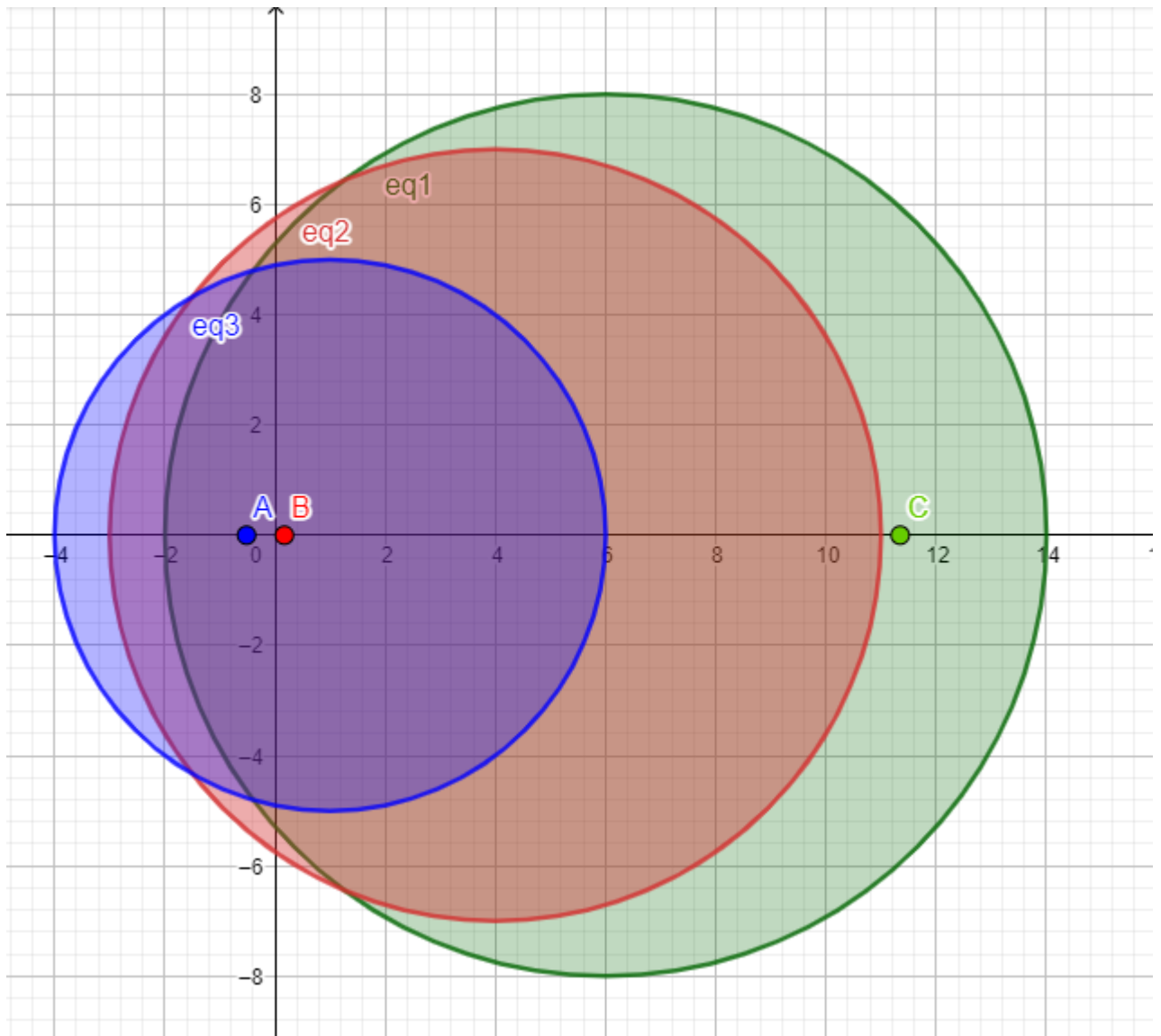
35     0.7369762
36     0.3279853
37     -0.591009
38     k =
39
40     6.
41     n_erro =
42
43     4.996D-16
44
45 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
46     ,10^(-10),-1,100); t8=toc()
47
48     "0 metodo convergiu com a precisao passada"
49     t8 =
50
51     0.0016906

```



Tempos de execução nos Discos de Gershgorin

Mesmo que o método de Rayleigh tenha devolvido o mesmo  $\lambda$  para diferentes  $\alpha$ s, o algoritmo de Rayleigh foi bem mais rápido que o método deslocado inverso, aproximadamente 10 vezes mais rápido. Isso ocorre pois o algoritmo de Rayleigh atualiza o  $\alpha$  a cada iteração, e portanto consegue chegar mais perto do autovalor de forma mais rápida que o método deslocado inverso.



Discos de Gershgorin calculados com os autovalores indicados

## 6 Testes extras

Embora sabendo que sendo diagonalizável a matriz irá convergir, o oposto não necessariamente é verdade. Por exemplo, buscando matrizes para testar, encontrei uma matriz não diagonalizável que, contrariando a minha expectativa, passou nos testes de convergência do método:

```

1 --> A=[2 1 1; 0 2 2; 0 0 3]
2   A  =
3
4     2.    1.    1.
5     0.    2.    2.
6     0.    0.    3.
7
8 --> x0=[1;2;3]
9   x0  =
10
11     1.
12     2.
13     3.

```

$$\text{Det}(A - \lambda * I) = (2 - \lambda)^2 * (3 - \lambda) = 0 \Rightarrow \lambda_1 = 3, \lambda_2 = 2, \lambda_3 = 2$$

```

1 --> D=[2 0 0; 0 2 0; 0 0 3]
2   D  =
3
4     2.    0.    0.
5     0.    2.    0.
6     0.    0.    3.

```

Para  $\lambda = 2$ :

```

1 --> v=[a;b;c]
2
3   v  =
4
5     a
6     b
7     c
8
9 --> A_linha=[0 1 1; 0 0 2; 0 0 1]
10  A_linha =
11
12     0.    1.    1.
13     0.    0.    2.
14     0.    0.    1.
15
16  A_linha*v=[0;0;0]
17
18  a=a
19  b=0
20  c=0

```

Com isso, temos que não existem  $v_1, v_2$  LI tal que  $Av_1 = 2v_1$  e  $Av_2 = 2v_2$ .

Tomando  $AP = P * [2, 0, 0; 0, 2, 0; 0, 0, 3]$ , podemos fazer  $v_1 = P.e_1 \Rightarrow Av_1 = AP.e_1 = P * [2, 0, 0; 0, 2, 0; 0, 0, 3] * [1; 0; 0] = P.2e_1 = 2v_1$  e  $v_2 = P.e_2 \Rightarrow Av_2 = AP.e_2 = P * [2, 0, 0; 0, 2, 0; 0, 0, 3] * [0; 1; 0] = P.2e_2 = 2v_2$ . Com isso, teríamos que ter  $v_1$  e  $v_2$  LI já que  $P$  é inversível, porém já havíamos provado anteriormente que não existem  $v_1$  e  $v_2$  LI, portanto  $A$  não pode ser diagonalizável. Ainda assim, é possível que  $A$  seja usado no método:

```

1 --> A=[2 1 1; 0 2 2; 0 0 3] (Autovalores: 3, 2, 2)
2   A  =

```

```

3
4     2.   1.   1.
5     0.   2.   2.
6     0.   0.   3.
7
8 --> x0=[1;2;3]
9     x0 =
10
11     1.
12     2.
13     3.
14
15 --> [lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
16
17     "0 metodo convergiu com a precisao passada"
18     lambda =
19
20     3.0000000
21     x1 =
22
23     1.
24     0.6666667
25     0.3333333
26     k =
27
28     63.
29     n_erro =
30
31     7.924D-11
32
33 --> [lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
34
35     "0 metodo convergiu com a precisao passada"
36     lambda =
37
38     3.0000000
39     x1 =
40
41     0.8017837
42     0.5345225
43     0.2672612
44     k =
45
46     60.
47     n_erro =
48
49     8.488D-11
50
51 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
52     ,0,100)
53
54     "0 numero de iteracoes passou do maximo de iteracoes permitido"
55     lambda =
56
57     1.9791757
58     x1 =

```

```

59     0.9997831
60     -0.0208288
61     3.842D-20
62     k =
63
64     101.
65     n_erro =
66
67     0.0002192
68
69 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
    ,0,100)
70
71     "0 metodo convergiu com a precisao passada"
72     lambda =
73
74     3.0000000
75     x1 =
76
77     0.8017837
78     0.5345225
79     0.2672612
80     k =
81
82     3.
83     n_erro =
84
85     2.483D-16
86
87 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
    ,1,100)
88
89     "0 numero de iteracoes passou do maximo de iteracoes permitido"
90     lambda =
91
92     1.9897970
93     x1 =
94
95     0.9999479
96     -0.0102036
97     6.037D-33
98     k =
99
100    101.
101    n_erro =
102
103    0.0001052
104
105 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
    ,1,100)
106
107     "0 metodo convergiu com a precisao passada"
108     lambda =
109
110     1.8571429
111     x1 =
112

```



```

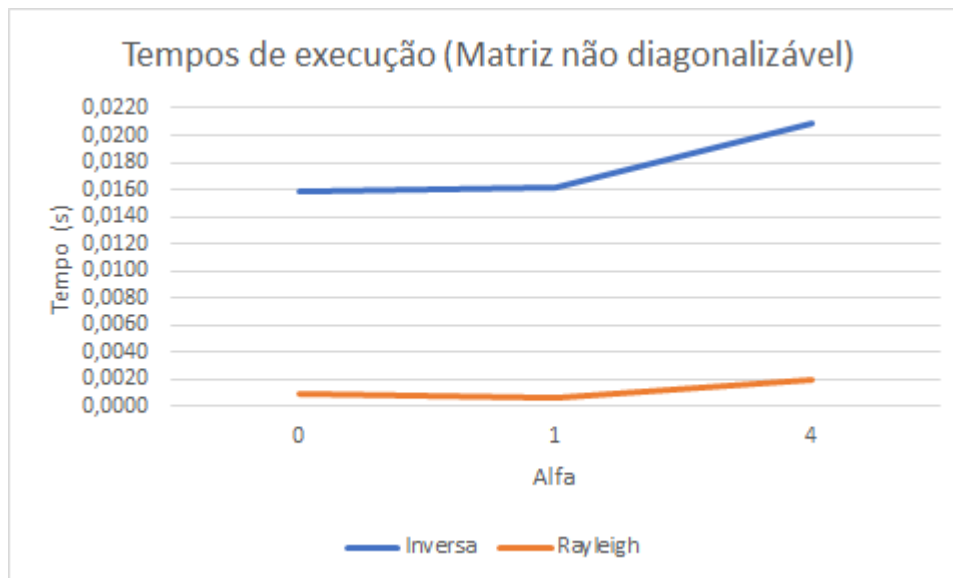
113     0.2672612
114     -0.5345225
115     0.8017837
116     k   =
117
118     1.
119     n_erro   =
120
121     2.544D-16
122
123 --> [lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0,10^(-10)
      ,4,100)
124
125 "0 numero de iteracoes passou do maximo de iteracoes permitido"
126 lambda   =
127
128     3.
129     x1   =
130
131     0.8017837
132     0.5345225
133     0.2672612
134     k   =
135
136     101.
137     n_erro   =
138
139     2.
140
141 --> [lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0,10^(-10)
      ,4,100)
142
143 "0 metodo convergiu com a precisao passada"
144 lambda   =
145
146     3.0000000
147     x1   =
148
149     -0.8017837
150     -0.5345225
151     -0.2672612
152     k   =
153
154     7.
155     n_erro   =
156
157     2.654D-13
158
159 -----
160
161 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_1(A,x0,10^(-10),100)
      ; t1=toc()
162
163 "0 metodo convergiu com a precisao passada"
164 t1   =
165
166     0.4845251

```

```

167
168 --> tic();[lambda,x1,k,n_erro]=Metodo_potencia_2(A,x0,10^(-10),100)
    ; t2=toc()
169
170 "0 metodo convergiu com a precisao passada"
171 t2 =
172
173 0.0013723
174
175 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
    ,10^(-10),4,100); t3=toc()
176
177 "0 numero de iteracoes passou do maximo de iteracoes permitido"
178 t3 =
179
180 0.0208992
181
182 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
    ,10^(-10),4,100); t4=toc()
183
184 "0 metodo convergiu com a precisao passada"
185 t4 =
186
187 0.0019363
188
189 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
    ,10^(-10),1,100); t5=toc()
190
191 "0 numero de iteracoes passou do maximo de iteracoes permitido"
192 t5 =
193
194 0.0162066
195
196 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
    ,10^(-10),1,100); t6=toc()
197
198 "0 metodo convergiu com a precisao passada"
199 t6 =
200
201 0.0006033
202
203 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_inversa(A,x0
    ,10^(-10),0,100); t7=toc()
204
205 "0 numero de iteracoes passou do maximo de iteracoes permitido"
206 t7 =
207
208 0.0159103
209
210 --> tic();[lambda,x1,k,n_erro]=Potencia_deslocada_Rayleigh(A,x0
    ,10^(-10),0,100); t8=toc()
211
212 "0 metodo convergiu com a precisao passada"
213 t8 =
214
215 0.0009824

```



Tempos de execução de métodos deslocados em matriz não diagonalizável

Embora a matriz tenha passado no teste, é notável que os métodos deslocados não devolveram sempre o  $\lambda$  esperado, especialmente para  $\lambda = 2$ . De qualquer forma, é visível que mesmo para uma matriz não diagonalizável, o Método da Potência 2 foi bem mais rápido que o 1, assim como o de Rayleigh foi bem mais rápido como o método inverso, como pode ser visto no gráfico.

## 7 Fontes de consulta

- Aulas Gravadas
- Álgebra Linear, David Poole (2ª edição)
- Álgebra Linear I - Aula 19, PUC RIO  
(<http://www.mat.puc-rio.br/cursos/MAT1200/roteiros/rotaula19.pdf>)
- Geogebra (para construção dos Discos de Gerschgorin)
- Exemplos de Matrizes 3x3 não diagonalizáveis, Matemática Universitária,  
(<https://www.youtube.com/watch?v=ae1MZN-8SIY>)
- Análise Numérica para Busca de Autovalores, Hortênsia Virginia Américo  
([https://repositorio.ufmg.br/bitstream/1843/EABA-978HQP/1/monografia\\_hortensia\\_americo.pdf](https://repositorio.ufmg.br/bitstream/1843/EABA-978HQP/1/monografia_hortensia_americo.pdf))