Relatório da Aula Prática 3

Sillas Rocha da Costa

2 de maio de 2024

Questão 01

Versão 1 do método da potência que pode ser encontrado na pasta "funcs", no arquivo "potencia_v1.sci", também possui a implementação da função return_max que:

```
function [lambda, x1, k, n_erro] = metodo_potencia_v1(A, x0, epsilon, M)
    k=0;
    x0 = x0/return_max(x0);
    x1 = A*x0;
    n_erro = epsilon + 1;

    while (k <= M && n_erro >= epsilon) then
        lambda = return_max(x1);
        x1 = x1/lambda;
        n_erro = norm(x1 - x0, %inf);
        x0 = x1;
        x1 = A*x0;
        k = k+1;
    end

endfunction
```

Versão 2 do método da potência que pode ser encontrado na pasta "funcs", no arquivo "potencia_v2.sci":

```
function [lambda, x1, k, n_erro] = metodo_potencia_v2(A, x0, epsilon, M)
    k = 0;
    x0 = x0/norm(x0, 2);
    x1 = A*x0;
    n_erro = epsilon + 1;

    while (k <= M && n_erro >= epsilon) then
        lambda = x1'*x0;
        if (lambda < 0) then
            x1 = -x1;
        end
        x1 = x1/norm(x1, 2);
        n_erro = norm(x1 - x0, 2);
        x0 = x1;
        x1 = A*x0;
        k = k+1;
    end

endfunction
```

Questão 02

Implementação do método da potência deslocada com iteração inversa, código se encontra na pasta "funcs", no arquivo potencia_inv.sci, para sua implementação foi reutilizado o código da eliminação gaussiana da aula prática 1.

```
function [lambda1, x1, k, n_erro] = metodo_potencia_inv(A, x0, epsilon, alfa, M)
    k = 0;
    x0 = x0/norm(x0, 2);
    n_erro = epsilon + 1;
    A_alfa = A - alfa*eye(A);

    while (k <= M && n_erro >= epsilon) then
        x1 = Gaussian_Elimination_4(A_alfa, x0);
        x1 = x1/norm(x1, 2);
        lambda1 = x1'*A*x1;

        if (x1'*x0 < 0) then
            x1 = -x1;
        end

        n_erro = norm(x1 - x0, 2);
        x0 = x1;
        k = k+1;
    end

endfunction
```

Questão 03

Após os primeiros testes, foi perceptível que os dois primeiros métodos na iteração final não retornavam os vetores normalizados e na direção correta, por isso, foi adicionado ao final da v1 o código:

```
    x1 = x1/lambda;
    if (lambda < 0) then
        x1 = -x1;
    end
```

e ao final da v2 o código:

```
    x1 = x1/norm(x1, 2);
    if (lambda < 0) then
        x1 = -x1;
    end
```

Para que houvesse a normalização final antes do retorno dos autovetores. As alterações podem ser verificadas nos arquivos das funções.

Deste modo, testes com vetores de entradas criadas pela função rand foram testados nas seguintes matrizes simétricas, as iterações e o tempo de execução de cada versão foram explicitadas para ajudar nas comparações, onde a primeira linha são os dados da v1 e a segunda da v2, é possível visualizar que ambas possuem resultados finais de lambdas iguais, com poquíssima diferença na quantidade de iterações, sendo a principal diferença no tempo de execução, em que a v2 é executada de maneira mais veloz que a v1, os testes foram executados repetidas vezes, em que a cada teste um novo vetor aleatório era gerado, e os resultados continuaram sendo semelhantes a estes:

```
"Matriz A1:"

 2.  -1.
-1.   1.

"Iterações, tempo e autovalores do método v1 e v2 para a matriz A1: "

 0.0002667   11.   2.618034
 0.0001428   11.   2.618034

"Matriz A2:"

 2.  -1.   5.
-1.  -3.   2.
 5.   2.  -4.

"Iterações, tempo e autovalores do método v1 e v2 para a matriz A2: "

 0.0008776   48.  -7.8577288
 0.0005079   47.  -7.8577287

"Matriz A3:"

 1.   5.   9.
 5.   3.   8.
 9.   8.   7.

"Iterações, tempo e autovalores do método v1 e v2 para a matriz A3: "

 0.0002907   16.   19.022768
 0.0001722   15.   19.022768

"Matriz A4:"

 4.   2.   1.   3.
 2.   5.   3.   2.
 1.   3.   6.   1.
 3.   2.   1.   7.

"Iterações, tempo e autovalores do método v1 e v2 para a matriz A4: "

 0.0004721   28.   11.641309
 0.0002938   27.   11.641309
```

Abaixo se encontram os resultados dos autovalores encontrados pelas funções em comparação com os autovalores de maior módulo das matrizes, em ambos os casos:

```
--> [lambda11, lambda21, return_max(spec(A1))]
 ans  =

   2.618034   2.618034   2.618034

--> [lambda12, lambda22, return_max(spec(A2))]
 ans  =

  -7.8577287  -7.8577287  -7.8577287

--> [lambda13, lambda23, return_max(spec(A3))]
 ans  =

   19.022768   19.022768   19.022768

--> [lambda14, lambda24, return_max(spec(A4))]
 ans  =

   11.641309   11.641309   11.641309
```

Questão 04

Seguem os testes realizados com matrizes simétricas, onde os chutes "alphas"foram baseados em Discos de Gerschgorin, usando sempre o centro dos discos como aproximação incial:

```
 "Matriz 1:"

  12.   1.   2.
  1.   -3.   1.
  2.    1.   4.

 "Chutes, autovalores encontrados com o chute e real espectro:"

  12.   12.565479  -3.1749615
 -3.   -3.1749615   3.6094829
  4.    3.6094829   12.565479

 "Matriz 2:"

  26.  -1.   2.
 -1.    7.   1.
  2.    1.  -6.

 "Chutes, autovalores encontrados com o chute e real espectro:"

  26.   26.170293  -6.2097815
  7.    7.039489    7.039489
 -6.   -6.2097815   26.170293

 "Matriz 3:"

  2.4440625   0.4056845   0.5053718
  0.4056845   2.4693936   0.6186707
  0.5053718   0.6186707   2.9446283

 "Chutes, autovalores encontrados com o chute e real espectro:"

  2.4440625   2.1326699   2.0233077
  2.4693936   2.1326699   2.1326699
  2.9446283   3.7021069   3.7021069
```

A partir dos 2 primeiros testes, é possível visualizar que a função funciona de maneira correta para encontrar os autovalores, entretanto, o exemplo 3, gerado a partir da multiplicação matricial de uma matriz de entradas randomizadas com sua transposta, que possui dois autovalores próximos, o método dos chutes pelos centros dos discos acaba não sendo uma escolha tão eficiente, uma forma de contornar este caso seria usar as bordas dos discos, já que é garantido pelo terema, que o autovalor está entre as duas bordas de extremidades do disco, e quando existe uma intersecção de dois discos, ao usar os chutes iniciais nas bordas que não se intersectam de cada disco, com certeza chegaremos nos dois autovalores, caso contrário, deve haver apenas um igual referente aos dois discos, já que, caso sejam diferentes, um deve ser maior que o outro, logo o chute da maior borda o achará primeiro, e analogamente, o mesmo acontecerá com o outro. Assim atualizamos os chutes do seguinte modo, para fazer o primeiro chute ser a partir da borda menor do disco:

```
--> C
 C  =

   2.4440625   0.4056845   0.5053718
   0.4056845   2.4693936   0.6186707
   0.5053718   0.6186707   2.9446283

--> alfa31 = C(1,1) - C(1,2) - C(1,3);

--> alfa32 = C(2,2);

--> alfa33 = C(3,3);
```

E obtemos os resultados:

```
--> disp("Matriz 3:", C)

 "Matriz 3:"

  2.4440625   0.4056845   0.5053718
  0.4056845   2.4693936   0.6186707
  0.5053718   0.6186707   2.9446283

--> disp("Chutes, autovalores encontrados com o chute e real espectro:",

 "Chutes, autovalores encontrados com o chute e real espectro:"

  1.5330062   2.0233077   2.0233077
  2.4693936   2.1326699   2.1326699
  2.9446283   3.7021069   3.7021069
```

Assim, achamos os 3 diferentes autovalores da matriz 3.

Questão 05

Os primeiros testes foram para ver a diferença entre os tempos de execução do método da potência v1 e v2, testados para matrizes de diferentes ordens geradas com a função rand, para as ordens 3, 20, 50 e 1000, deste modo, é possível observar que a versão 2 é executada em menos tempo que a versão 1:

```
"Tempos dos métodos da potência versão 1 e 2 para as geradas matrizes aleatoriamente"

"de ordens 3, 20, 50 e 1000, primeira coluna a v1, segunda coluna a v2"

"e terceira a diferença, cada linha respectivamente a uma orden:"

 0.0003215   0.0001952   0.0001263
 0.0003897   0.0002466   0.0001431
 0.0004162   0.0002515   0.0001647
 0.0059729   0.0052423   0.0007306
```

Os próximos testes foram em relação ao método da inversa deslocada, ao gerar matrizes simétricas com o auxílio da função rand e usando os elementos da diagonal de chutes para o cálculo dos autovalores:

```
"Reultados do método da inversa comparado com o real spectro"

"Resultados da 2x2:"

 2.0517519   2.0517519
 4.0155888   4.0155888

"Resultados da 3x3:"

 1.7708669   1.7708669
 2.1552056   2.1552056
 4.0907343   4.0907343

"Resultados da 5x5:"

 1.8843589   1.8843589
 4.0494148   3.5408508
 4.2676785   4.0494148
 4.2676785   4.2676785
 4.2676785   7.7324686

"Resultados da 10x10:"

 21.870862   20.220321
 23.823647   21.870862
 25.165771   23.823647
 27.889841   25.165771
 27.889841   25.972433
 31.567274   27.889841
 31.567274   29.791267
 32.345235   31.567274
 32.345235   32.345235
 32.345235   57.687641
```

Considerações finais:

Os métodos da potência, tanto a versão 1 quanto a versão 2, demonstraram boa capacidade de convergir para os resultados esperados. Ambos os métodos foram eficazes em encontrar autovalores e autovetores das matrizes fornecidas, com desempenhos finais semelhantes, exceto pelo tempo de execução.

A versão 2 do método da potência mostrou-se mais eficiente em termos de tempo de execução, superando a versão 1 em todas as ordens de matrizes testadas. Indicando ser superior a versão 1.

Além disso, o método da inversa deslocada apresentou bons resultados em encontrar os autovalores, o principal método utilizado foi o dos chutes iniciais serem os elementos da diagonal, que muitas vezes mostrou convergir para diferentes autovalores. No entanto, observou-se que o método não é tão eficiente em casos de matrizes com autovalores próximos.

Para contornar este problema, é possível melhorar a escolha dos chutes inciais, usando a borda dos Discos de Gerschgorin por exemplo.

Em geral, os métodos apresentam ótimos resultados na identificação de autovalores e autovetores, entretanto apresentam diferentes performances na execução, e os chutes iniciais também podem influenciar fortemente os resultados finais.