PROJETO 4:
PROBLEMA DE AUTOVALORES E AUTOVETORES.

Me. Gubio Gomes De Lima.

07/10/2022

1 Introdução

Problema : Encontrar auto valores e auto vetores de uma matriz complexa. Objetivo especifico:

1. Mostre na tela do computador a matriz complexa original.
2. Mostre na tela do computador a matriz complexa original.
3. Mostre na tela do computador a matriz real de ordem 2n tridiagonalizada.
4. Mostre na tela do computador Autos valores e autos vetores.

Queremos resolver uma matriz complexa $A = B + iC$, onde B é a parte real e C a parte complexa. Para podemos fazer essa abordagem, convertermos a matriz nxn em uma matriz 2nx2n. No qual a dividiremos a nova matriz em 4 quadrante.

$$A_{[2n,2n]} = \begin{bmatrix} B & C \\ C & B \end{bmatrix} \tag{1}$$

Os métodos utilizados no programa foram *HouseHolder* para tridiagonalizar a matriz. Que necessita de no máximo de k-n passo onde n é a dimensão da matriz. O processo é dado por k transformação dada por :

$$A^{(k)} = [O^K]^T A^{(k-1)} O^K \tag{2}$$

onde T é a transporta, A^0 é a matrix inicial k = 1,2,...n.

Coma matriz tri-diagonalizada iremos obter os polinômios característico por meia da expressão.

$$P_n(\lambda_n) = (A_{n,n} - \lambda) * P_{n-1}(\lambda_n) - A^2_{n,n-1} P_{n-2}(\lambda) \tag{3}$$

com $P_0 = 1$, sabemos que o o autovalores podem ser encontrado fazendo $P_n(\lambda_n) = 0$, então se resume a buscar os zeros da função, neste caso podemos usar busca direta. Em especifico foi notado que esta função dificilmente passo pelo zero , logo no scripy usamos a condição dos valores estarem muito próximo de zero.

Por fim desejamos encontrar os auto vetores, para tanto desejamos resovler a seguinte expressão :

$$v^{i+1}_{new} = [A-]^{-1} v^i \tag{4}$$

Notando esse método necessita que conseguimos obter a matriz inversa, para tanto usaremos o o método de Gauss-Jordan.

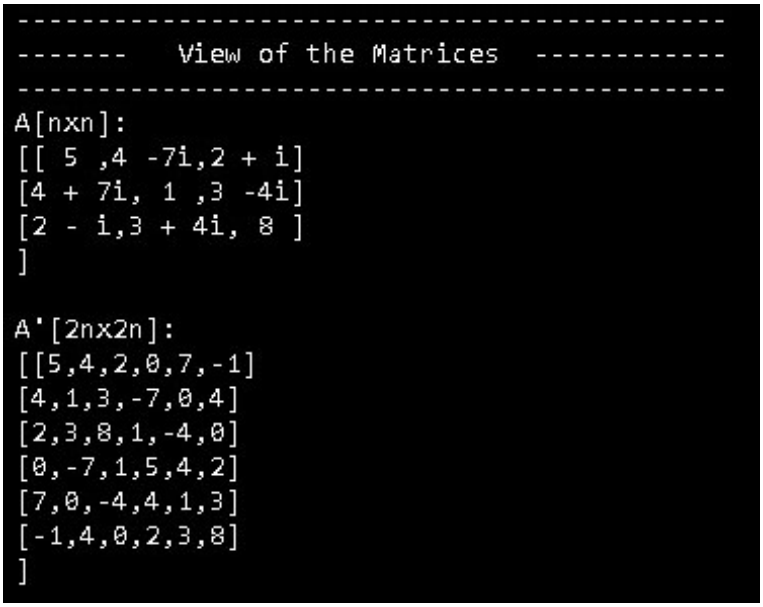
2 Descrição do programa

Com o objetivo de resolver o problemas, vamos utilizar a linguagem de programação C++ e os métodos numéricos ensinados em sala de aula.

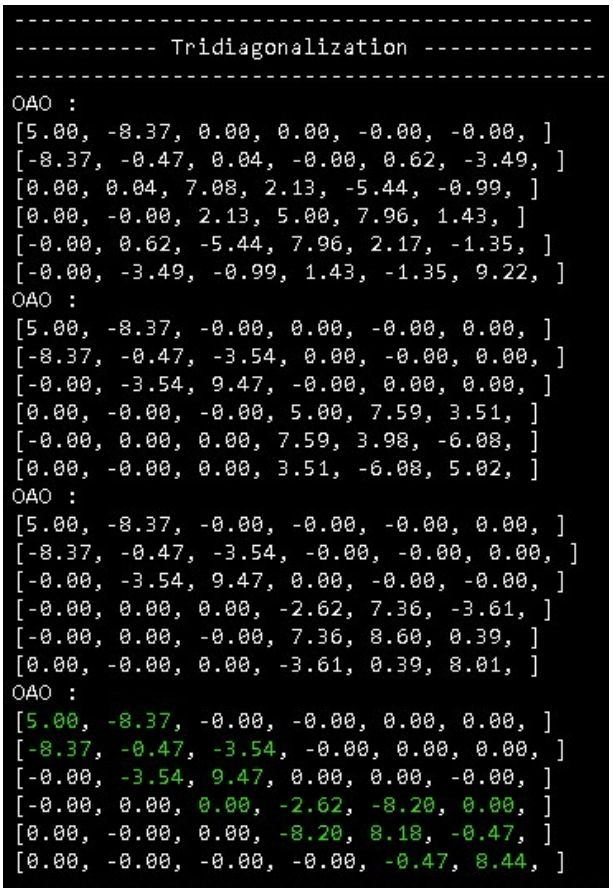
No programa criado o primeiro passo foi criar um comando para que o usuário digite sua matriz , mas antes é solicitado a dimensão dela e qual a precisão(casa decimais que sera usada em algum passo do script). Para facilitar o processo de teste foi inserido antes de tudo um comando para usar que o script use uma matriz já salva.

```
PS C:\Users\Oibug> cd "c:\Users\Oibug\Desktop\gubio\Fís
ica Computacional\Projeto 4"
PS C:\Users\Oibug\Desktop\gubio\Física Computacional\Pr
ojeto 4> & .\"matriz.exe"
Type 1 if you dont want type a matrix
1
What is the dimension of the matrix
3
What is the precision of elements of matriz
2
-----------------------------------------
```

Como mostrado acima , se o usuário digitar 1 ele usara a matriz salva, de acordo com a dimensão da matriz, e em seguida será apresentado na tela as matrizes. Como ilustrado seguir:



Em seguida iremos tri-diagonalizar a matriz A', usando o método HouseHolder. Podemos ver na imagem a seguir o resultado final de cada transformação k.



Próximo passo encontrar os auto valores com o polinômio característico. Nesse caso seria muito interessante usar o busca direta para encontrar os auto valores, somando todos os elementos da matrix, mas foi notado que as função não cruzam o zero apenas "tocam" , portanto adaptamos para que o auto valor seja encontrar quando a função estiver muito próxima de zero, iniciando do valor mínimo possível.

O primeiro passo é determinar o minimo e maximo possiveis somando cada elementos, criando dois "for" um para a coluna(j) e outro para linha(i) das matrizes, para cada valor de i e j armazenamo em uma variavel para somar com a próxima.

```
for(int j=0; j<2*n ;j++)                                    /// Determinando o máximo e mínimo do loop.
{
    for(int i=0; i<2*n ;i++)
    {
        A_maxi_mini = A_maxi_mini +abs(A_n[j][i]);                  /// Calculando o valor inicial para o
    }
}
lambda_inicial = -A_maxi_mini/10-1 ;
cout<<"Esta iniciando de "<< lambda_inicial <<" ate "<<A_maxi_mini<<"\n";
```

Obtido esses valores podemos iniciar o loop para encontra os auto valores, veja o resultado a seguir :

```
-----------------------------------------------
------ Characteristic polynomials -----------
-----------------------------------------------
Esta iniciando de -8.53 ate 75.35
Encontrei algo !
 Auto valor encontrado : -7.04
Encontrei algo !
 Auto valor encontrado : 8.40
Encontrei algo !
 Auto valor encontrado : 12.64
```

O scripty avisa sempre que encontrar um possível valor mandando a mensagem "Encontrei algo ", depois dessa mensagem ele da alguns passo para trás e diminui o passo para aumentar a precisão do valor encontrado . Com os auto valores em mãos iremos buscar os auto valores, mas lembre que precisa encontrar a matriz inversa, descrita na equação(4).

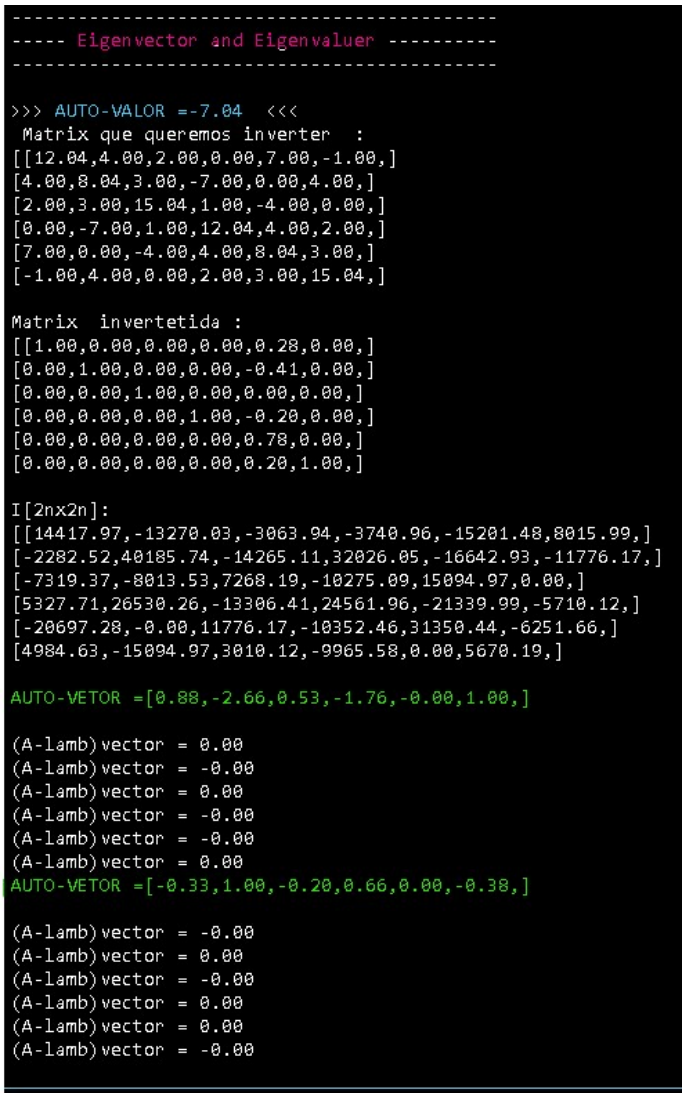
A seguir temos duas matriz, a que desejamos inverte e a matriz identidade, nosso objetivo é que a 1° matriz se transforme em uma matriz identidade, para tanto realizamos alguma manipulação. Ambas matrizes sofreram da mesma regras de manipulação, portanto o resultado da matriz identidade é matriz inversa.

```
-----------------------------------------
----- Eigenvector and Eigenvaluer ----------
-----------------------------------------

>>> AUTO-VALOR =8.40  <<<
 Matrix que queremos inverter  :
[[-3.40,4.00,2.00,0.00,7.00,-1.00,]
[4.00,-7.40,3.00,-7.00,0.00,4.00,]
[2.00,3.00,-0.40,1.00,-4.00,0.00,]
[0.00,-7.00,1.00,-3.40,4.00,2.00,]
[7.00,0.00,-4.00,4.00,-7.40,3.00,]
[-1.00,4.00,0.00,2.00,3.00,-0.40,]

Matrix  invertetida :
[[1.00,0.00,0.00,0.00,0.00,0.00,]
[0.00,1.00,0.00,0.00,0.00,0.00,]
[0.00,0.00,1.00,0.00,0.00,-0.00,]
[0.00,0.00,0.00,1.00,0.00,0.00,]
[0.00,0.00,0.00,0.00,1.00,0.00,]
[0.00,0.00,0.00,0.00,0.00,1.00,]

I[2nx2n]:
[[-14602.47,2388.65,-794.01,-0.00,-5430.15,19617.59,]
[2388.65,-2410.00,7425.11,5430.15,-0.00,-2913.58,]
[-794.01,7425.11,-26397.95,-19617.59,2913.58,0.00,]
[0.00,5430.15,-19617.59,-14602.47,2388.65,-794.01,]
[-5430.15,0.00,2913.58,2388.65,-2410.00,7425.11,]
[19617.59,-2913.58,0.00,-794.01,7425.11,-26397.95,]
```

Porm fim, usaremos essa matriz inversa encontrada , para determinar os auto vetores, para tanto precisamo da um chute inicial, de um auto vetor e em seguida a equação(4) encontrar um novo auto

vetor depois de varias tentativas o resultado irá convergir para o auto vetor da Matriz inicial para aquele autovalor, aqui estou fazendo esse processo 10.000 vezes.

Foi notado que o scripy sempre dar dois autovalores possível dependendo do chute inicial, portanto fiz um loop para ele tentar com um valor $+-[10, 10, 10, 10...]$ de tamanho n. Assim ele apresenta dois auto vetores par ao mesmo auto valor, mostrando que o auto valor é degenerado. Veja seguir :



Note que logo em seguida, é realizado o calculo da equação de auto valor, com o auto vetor encontrada a matrix inicial e o auto valor encontrada, espera-se que o resultado seja zero ou próximo disso.

A seguir podemos ver par aos outro autovalroes:

```
>>> AUTO-VALOR =8.40  <<<
 Matrix que queremos inverter  :
[[-3.40,4.00,2.00,0.00,7.00,-1.00,]
[4.00,-7.40,3.00,-7.00,0.00,4.00,]
[2.00,3.00,-0.40,1.00,-4.00,0.00,]
[0.00,-7.00,1.00,-3.40,4.00,2.00,]
[7.00,0.00,-4.00,4.00,-7.40,3.00,]
[-1.00,4.00,0.00,2.00,3.00,-0.40,]

Matrix  invertetida :
[[1.00,0.00,0.00,0.00,0.00,0.00,]
[0.00,1.00,0.00,0.00,0.00,0.00,]
[0.00,0.00,1.00,0.00,0.00,-0.00,]
[0.00,0.00,0.00,1.00,0.00,0.00,]
[0.00,0.00,0.00,0.00,1.00,0.00,]
[0.00,0.00,0.00,0.00,0.00,1.00,]

I[2nx2n]:
[[-14602.47,2388.65,-794.01,-0.00,-5430.15,19617.59,]
[2388.65,-2410.00,7425.11,5430.15,-0.00,-2913.58,]
[-794.01,7425.11,-26397.95,-19617.59,2913.58,0.00,]
[0.00,5430.15,-19617.59,-14602.47,2388.65,-794.01,]
[-5430.15,0.00,2913.58,2388.65,-2410.00,7425.11,]
[19617.59,-2913.58,0.00,-794.01,7425.11,-26397.95,]

AUTO-VETOR =[-0.03,-0.27,1.00,0.75,-0.13,0.08,]

(A-lamb)vector = 0.00
(A-lamb)vector = 0.00
(A-lamb)vector = -0.00
(A-lamb)vector = -0.00
(A-lamb)vector = 0.00
(A-lamb)vector = -0.00
AUTO-VETOR =[0.12,1.00,-3.68,-2.74,0.49,-0.31,]

(A-lamb)vector = -0.00
(A-lamb)vector = -0.00
(A-lamb)vector = 0.00
(A-lamb)vector = 0.00
(A-lamb)vector = -0.00
(A-lamb)vector = 0.00
```

```
>>> AUTO-VALOR =12.64  <<<
 Matrix que queremos inverter  :
[[-7.64,4.00,2.00,0.00,7.00,-1.00,]
[4.00,-11.64,3.00,-7.00,0.00,4.00,]
[2.00,3.00,-4.64,1.00,-4.00,0.00,]
[0.00,-7.00,1.00,-7.64,4.00,2.00,]
[7.00,0.00,-4.00,4.00,-11.64,3.00,]
[-1.00,4.00,0.00,2.00,3.00,-4.64,]

Matrix  invertetida :
[[1.00,0.00,0.00,0.00,0.00,0.00,]
[0.00,1.00,0.00,0.00,0.00,0.00,]
[0.00,0.00,1.00,0.00,0.00,0.00,]
[0.00,0.00,0.00,1.00,0.00,0.00,]
[0.00,0.00,0.00,0.00,1.00,0.00,]
[0.00,0.00,0.00,0.00,0.00,1.00,]

I[2nx2n]:
[[6518.10,4616.29,1634.82,-0.00,4824.09,5691.43,]
[4616.29,6839.55,5369.98,-4824.09,0.00,2820.75,]
[1634.82,5369.98,5379.24,-5691.43,-2820.75,0.00,]
[0.00,-4824.09,-5691.43,6518.10,4616.29,1634.82,]
[4824.09,0.00,-2820.75,4616.29,6839.55,5369.98,]
[5691.43,2820.75,0.00,1634.82,5369.98,5379.24,]

AUTO-VETOR =[nan,nan,nan,nan,nan,nan,]

(A-lamb)vector = nan
(A-lamb)vector = nan
(A-lamb)vector = nan
(A-lamb)vector = nan
(A-lamb)vector = nan
(A-lamb)vector = nan
AUTO-VETOR =[1.00,0.64,0.17,0.10,0.81,0.90,]

(A-lamb)vector = 0.00
(A-lamb)vector = 0.00
(A-lamb)vector = 0.00
(A-lamb)vector = 0.00
(A-lamb)vector = 0.00
(A-lamb)vector = 0.00
```

Note que em exceção do primeiro autovetor do auto valor 12.64, todos satisfazem a equação de auto valor. Indicando um bom resultado, mas quando comparamos com o resultado analítico temos vemos que esses resultados não estão bons .