

UNIVERSIDADE FEDERAL DA PARAÍBA

DISCENTE: Nathan Carlos de Macena Gomes

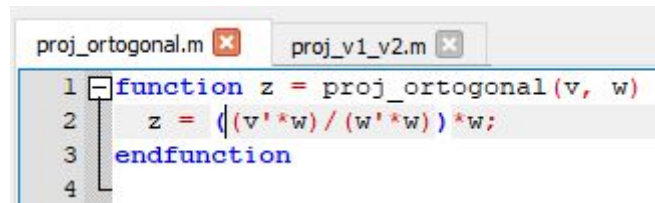
MATRÍCULA: 20180040977

DISCIPLINA: Álgebra Linear Computacional

Professor: Miguel

### QUESTÃO 1.

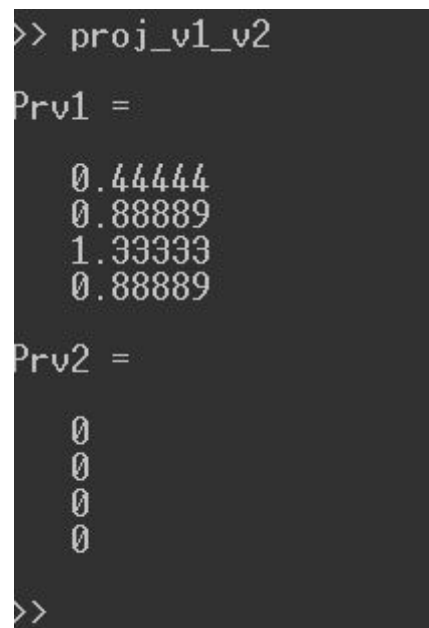
Rotina para calcular a projeção ortogonal de um vetor



```
proj_ortogonal.m x proj_v1_v2.m x
1 function z = proj_ortogonal(v, w)
2   z = ((v'*w) / (w'*w)) * w;
3 endfunction
4
```

### QUESTÃO 2.

Resultados da Projeção de w em v1 e v2



```
>> proj_v1_v2

Prv1 =

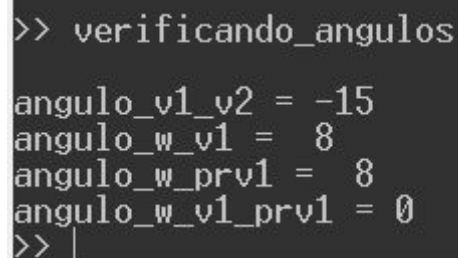
    0.44444
    0.88889
    1.33333
    0.88889

Prv2 =

     0
     0
     0
     0

>>
```

### QUESTÃO 3.



```
>> verificando_angulos

angulo_v1_v2 = -15
angulo_w_v1 = 8
angulo_w_prv1 = 8
angulo_w_v1_prv1 = 0
>>
```

Os pares:

(v1, v2) é **menor** que zero, logo, o ângulo é **maior** que 90°.

(w, v1) é **maior** que zero, logo, o ângulo é **menor** que 90°.

(w, prv1) é **maior** que zero, logo, o ângulo é **menor** que 90°.

(w, v1 - prv1) é **igual** a zero, logo, o ângulo é **igual** a 90°.

#### QUESTÃO 4.

Matriz ortogonal é quando uma matriz  $A$  tem inversa e é igual a sua transposta.

#### QUESTÃO 5.

a) Sejam  $A$  e  $B$  duas matrizes ortogonais:

$$\text{Então, } A \cdot A^+ = A^+ \cdot A = I \text{ e } B \cdot B^+ = B^+ \cdot B = I$$

$$\text{Se } A \cdot B = C$$

$$\text{temos } A^+ A B = A^+ C \Rightarrow I B = A^+ C \Rightarrow A^+ C$$

$$B^+ B = B^+ A^+ C \Rightarrow \boxed{I = (AB)^+ C}$$

Como  $C = A \cdot B$ ,  
então  $AB$  é ortogonal

$$b) M^T \cdot M = I$$

Pelo teorema de Binet

$$\det(M^T \cdot M) = \det(I)$$

$$\det(M^T) \cdot \det(M) = 1 \quad (\det M^T = \det M)$$

$$\det^2 M = 1 \Rightarrow \boxed{\det M = \pm 1}$$

c) Seja  $M$  uma matriz ortogonal cujos vetores  
colunas são  $u = (a, b)$  e  $v = (c, d)$

$$Id = M^T M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} a & c \\ b & d \end{pmatrix} = \begin{pmatrix} aa + bb & ac + bd \\ ac + bd & cc + dd \end{pmatrix}$$

$$= \begin{pmatrix} u \cdot u & u \cdot v \\ u \cdot v & v \cdot v \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Logo,  $u \cdot v = v \cdot v = 1$ ,  $v \cdot u = 0$   
 $u$  e  $v$  formam uma base ortonormal

### QUESTÃO 6.

```

decomp_qr.m  qr_a_a2.m
1 function [Q R] = decomp_qr(A)
2     [m n] = size(A);
3     R = zeros(n);
4     Q = zeros(m,n);
5     for j = 1:n
6         proj = zeros(m,1);
7
8         for i = 1:j-1
9             R(i,j) = Q(:,i)' * A(:,j);
10            proj = proj + R(i,j)*Q(:,i);
11        endfor
12        v = A(:,j) - proj;
13        R(j,j) = norm(v);
14        Q(:,j) = v / R(j,j);
15    endfor
16 endfunction

```

### QUESTÃO 7.

Para A:

```

Q =
    0.89443   -0.40825
   -0.44721    0.81650
    0.00000    0.40825

R =
    2.23607    0.00000
    0.00000    2.44949

```

Para A2:

```

Q =
    0.81650   -0.40825    0.62554    0.00000
   -0.40825    0.81650   -0.20851    0.31623
    0.00000    0.00000    0.41703    0.94868
    0.40825    0.40825    0.62554    0.00000

R =
    2.44949    0.00000    0.00000    0.00000
    0.00000    2.44949    0.00000    0.00000
    0.00000    0.00000    4.79583    0.00000
    0.00000    0.00000    0.00000    3.16228

```

## QUESTÃO 8

Cálculo dos autovalores utilizando a decomposição qr

```
autovalor_qr.m x decomp_qr.m x
1 function lambda = autovalor_qr(A)
2     epsilon = 10e-3;
3
4     [Z R] = decomp_qr(A);
5     Z = A * Z;
6
7     while(-1)
8         [Z R] = decomp_qr(Z);
9         Z = A * Z;
10        lambda = diag(Z);
11        r = sum(sum(abs(Z - diag(lambda))));
12
13        if r < epsilon
14            break;
15        endif
16    endwhile
17 endfunction
```

## QUESTÃO 9.

A matriz de Householder nada mais é que, em gráficos, dado uma reta  $z$ , achar a sua matriz de projeção  $z'$  ou seja,

$$z' = I - 2 \frac{vv^T}{v^T v}$$

## QUESTÃO 10.

a)  $u = x + y$        $u^+ = (x + y)^+$   
 $v = x - y$

$$\Rightarrow u^+ \cdot v = (x + y)^+ (x - y)$$
$$= x^+ x - \cancel{x^+ y} + \cancel{x y^+} - y^+ y$$

Se,  $|x| = |y|$

$$\Rightarrow u^+ \cdot v = 0$$

b)

$$Qu = u$$

$$\Rightarrow Qu = Qx + Qy$$

$$u = x + y$$

$$Q = I - \frac{2vv^T}{v^Tv} \Rightarrow \left( I - \frac{2vv^T}{v^Tv} \right) \cdot x = y$$

$$\text{Logo, } \begin{cases} Qx = y \\ Qy = x \end{cases} \Rightarrow \begin{cases} Qu = x + y \\ Qu = u \end{cases}$$

$$Qv = -v$$

$$v = x - y$$

$$\Rightarrow Qv = Q(x - y) \\ Qv = Qx - Qy$$

$$Q = I - \frac{2vv^T}{v^Tv} \Rightarrow \left( I - \frac{2vv^T}{v^Tv} \right) \cdot x = y$$

$$\text{Logo, } \begin{cases} Qx = y \\ Qy = x \end{cases} \Rightarrow Qv = Qx - Qy$$

$$Qv = y - x$$

$$Qv = -v$$

c)  $QQ = I$

Seja  $Q$  uma matriz simétrica, então

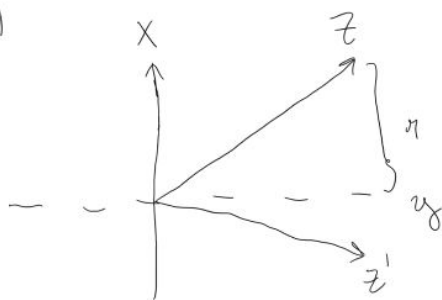
$$Q = Q^T,$$

$$Q \cdot Q^T = I$$

d)  $Q$  é uma matriz simétrica e, obrigatoriamente,

$$Q = Q^T$$

2)



$$z' = z + 2\eta$$

$$= z - 2 \operatorname{pr}_{\perp v}(z) = z - \frac{2z^T v}{v^T v} v$$

$$= z - 2v \frac{v^T z}{v^T v} = \left( I - \frac{2vv^T}{v^T v} \right) z$$

$$Q = I - \frac{2vv^T}{v^T v}$$

### QUESTÃO 11.

Gerando

		$a$	$x^T$	$1$	$Q^+$
	$A = A^T$	$x$	$A$	$0$	$Q$
$1$	$Q^+$	$Q$	$x^T$	$Q$	$1 \times 1$
$0$	$Q$	$1 \times 1$	$Q A$	$1 \times 1$	$0 \dots 0$
		$0$		$0$	$x$
		$0$		$0$	$Q A Q$

Our result,

$$\begin{bmatrix} 1 & Q \\ 0 & Q^+ \end{bmatrix} \begin{bmatrix} a & x^T \\ x & A \end{bmatrix} \begin{bmatrix} 1 & Q^+ \\ 0 & Q \end{bmatrix} = \left[ \begin{array}{c|c} a & 1 \times 1 \ 0 \dots 0 \\ \hline 1 \times 1 & Q A Q \end{array} \right]$$



## QUESTÃO 12.

```
tridiagonalizacao.m x tridiagonalizacao_c1_c2.m x
1 function T = tridiagonalizacao (A)
2     T = A;
3     [m n] = size(A);
4
5     for i = 1:n-2
6         x = T((i+1):n,i);
7         sx = size(x,1);
8         y = zeros(sx,1);
9         y(1) = norm(x);
10        w = x - y;
11        Q = eye(sx) - 2*(w * w') / (w' * w);
12        T((i+1):n,i) = y;
13        T(i, (i+1):n) = y;
14        T((i+1):n, (i+1):n) = Q * T((i+1):n, (i+1):n) * Q;
15
16    endfor
17 endfunction
18
```

## QUESTÃO 13.

Matriz tridiagonal das matrizes C1, C2 e C2:

```
tri_c1 =
```

2.00000	1.41421	0.00000
1.41421	6.50000	1.50000
0.00000	1.50000	6.50000

```
tri_c2 =
```

0.00000	2.23607	0.00000	0.00000
2.23607	0.00000	0.44721	0.00000
0.00000	0.44721	NaN	NaN
0.00000	0.00000	NaN	NaN

```
>> |
```

```
tri_c3 =
```

5.00000	1.00000	0.00000	0.00000
1.00000	3.00000	1.41421	0.00000
0.00000	1.41421	2.50000	-1.50000
0.00000	0.00000	-1.50000	4.50000

```
>> |
```

#### QUESTÃO 14.

```
poli_sturm.m ✖
1 function pw = poli_sturm(d, e, w)
2     n = size(d,1);
3     pw = zeros(n+1, 1);
4     pw(1) = 1;
5     pw2 = w - d(1);
6     for i = 3:n+1
7         pw(i) = (w - d(i-1)) * pw(i-1) - (e(i-2)^2) * pw(i-2);
8     endfor
9 endfunction
```

#### QUESTÃO 15.

```
msinal.m ✖
1 function ms = msinal(w,d,e)
2     ms = 0;
3     pw = poli_sturm(w,d,e);
4     n = size(d,1);
5     for i = 2:n+1
6         if ((pw(i) * pw(i-1)) < 0)
7             ms = ms + 1;
8         endif
9     endfor
10 endfunction
```

### QUESTÃO 16.

$$A = \begin{bmatrix} 2 & 1 & & & \\ 1 & 5 & 2 & & \\ & 2 & 4 & 2 & \\ & & 2 & 3 & 1 \\ & & & 1 & 5 \end{bmatrix}$$

$p_0 = 1$   
 $\rightarrow p_1 = \lambda - 2$   
 $\rightarrow p_2 = (\lambda - 5)p_1 - 1^2$   
 $\rightarrow p_3 = (\lambda - 4)p_2 - 2^2 p_1$   
 $\rightarrow p_4 = (\lambda - 3)p_3 - 2^2 p_2$   
 $\rightarrow p_5 = (\lambda - 5)p_4 - 1^2 p_3$

1
2
$-1 \cdot 2 - 1 = -3$
$-4 \cdot 2 = -8$
$1 \cdot (-8 - 4(-3)) = 4$
$1 \cdot 4 - 1(-8) = 12$

$\Rightarrow$  Os intervalos são  
 $[1, 3]$   
 $[2, 8]$   
 $[0, 8]$  *é o maior intervalo, portanto os outros estão lá.*  
 $[0, 6]$   
 $[4, 6]$

### QUESTÃO 17.

```

teste.m x sturm_bissecao.m x gershgorin.m x
1 function lambda = sturm(d, e)
2     [a b] = gershgorin(d, e);
3
4     lambda = bissecao(a, b, d, e);
5
6 endfunction
7

```

### QUESTÃO 18.

```
tridiag_sim.m x teste.m x
1 function x = tridiag_sim(d, e, b)
2     T = diag(d) + diag(e,1) + diag(e, -1);
3
4     A = T;
5     Ab = [A b];
6     n=size(A,2);
7
8     for i=n:-1:1
9         soma = 0;
10        for j=i+1:n
11            soma=soma+Ab(i,j)*x(j);
12        endfor
13        x(i) = (b(i) - soma) /Ab(i,i);
14    endfor
15 endfunction
16
```

### QUESTÃO 19.

```
iterac_inv.m x resolve00.m x tridiag_sim.m x
1 function x = iterac_inv(d,e,lambda)
2     tol1 = 1, tol2 = 10e-7, epsilon = 10e-6;
3     T = diag(d) + diag(e,1) + diag(e, -1);
4     A = T;
5     tal = lambda + epsilon;
6     n = size(d,1);
7
8     %escolhendo b0
9     while(1)
10        bo = rand(n,1);
11        bo = bo/abs(bo);
12        At = A - tal*eye(n);
13        y1 = resolve00(At, bo);
14        if abs(y1) > tol1
15            break
16        endif
17    end
18
19    %calculando x
20    k=1;
21    while(1)
22        At2 = A - tal*eye(n);
23        x = resolve00(At2, bo);
24        x=x/abs(x);
25
26        if abs(x- bo) < tol2
27            break
28        endif
29        bo=x;
30        k=k+1;
31    end
32 endfunction
33
```

## QUESTÃO 20.

```
autovalores_sim.m x iterac_inv.m x resolve.m x tridiagonalizacao.m x
1 function [lambda X] = autovalores_sim (A)
2
3     T = tridiagonalizacao(A)
4     [m n] = size(A);
5
6     %pegando d e e
7     d = diag(T);
8     for i = 1:n-1
9         e(i) = T(i,i+1);
10    endfor
11
12    lambda = lambda_sturm(d, e)
13    X = iterac_inv(d,e,lambda);
14 endfunction
15
```