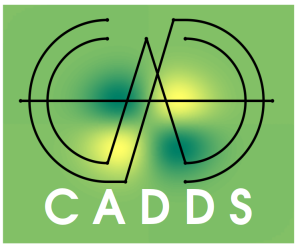
Manual de Laboratorio:

CÓMPUTO MATRICIAL BÁSICO CON GNU OCTAVE



Prof. Dr. Fredy Vides
Scientific Computing Innovation Center, UNAH &
Centre for Analysis of Data-Driven Systems
E-mail: fredy.vides@unah.edu.hn

ÍNDICE

OBJETIVOS

1. Aplicar GNU Octave en la realización de operaciones elementales del cómputo matricial.

1. CÓMPUTO MATRICIAL BÁSICO

1.1. Operaciones y funciones matriciales. Consideremos las siguientes matrices:

$$A = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \tag{1.1}$$

$$B = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{1.2}$$

$$C = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{1.3}$$

$$D = \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{1.4}$$

$$v = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{1.5}$$

Los comandos necesarios para ingresar las matrices anteriores en Octave/MATLAB son los siguientes:

```
>> A=[2,-1,0;-1,2,-1;0,-1,2]
A =

   2  -1   0
  -1   2  -1
   0  -1   2

>> B=[0,-1,1;1,0,0;0,1,0];
>> C=[0,0,1;1,0,0;0,1,0];
>> D=[0,-1;1,0;0,1];
>> v=[0;1;0];
```

1.1.1. Operaciones con matrices. Mostraremos a continuación los procedimientos computacionales correspondientes a algunas operaciones algebraicas fundamentales.

- $R_1 = A + B$:

```
>> R1=A+B
R1 =

   2  -2   1
   0   2  -1
   0   0   2
```

- $R_2 = 7A$:

```
>> R2=7*A
R2 =

   14   -7    0
   -7   14   -7
    0   -7   14
```

- $R_3 = B^\top$

```
>> R3=B.'
R3 =

    0   1   0
   -1   0   1
```

```
    1    0    0
```

Importante: El símbolo (') en la expresión `R3=B.'` debe ingresarse seleccionando en el teclado la tecla que corresponde al apóstrofo.

- $R_4 = (B + iC)^*$

```
>> R4=(B+i*C)'
R4 =

   0 - 0i   1 - 1i   0 - 0i
  -1 - 0i   0 - 0i   1 - 1i
   1 - 1i   0 - 0i   0 - 0i
```

- $R_5 = BC$

```
>> R5=B*C
R5=B*C
R5 =

  -1    1    0
   0    0    1
   1    0    0
```

- $R_6 = \mathbf{1}_3$

```
>> R6=eye(3)
R6 =

Diagonal Matrix

   1    0    0
   0    1    0
   0    0    1
```

- $R_7 = a_{12}$

```
>> R7=A(1,2)
R7 = -1
```

- $R_8 = [a_{12}\ a_{13}]$

```
>> R8=A(1,2:3)
R8 =

  -1    0
```

- $R_9 = A_2 = [a_{21}\ a_{22}\ a_{33}]$

```
>> R9=A(2,:)
R9 =

  -1    2   -1
```

- $R_{10} = \det(A)$

```
>> R10=det(A)
R10 =  4.0000
```

- $R_{11} = A^{-1}$

```
>> R11=inv(A)
R11 =

   0.75000   0.50000   0.25000
   0.50000   1.00000   0.50000
   0.25000   0.50000   0.75000
```

- $R_{12} = \mathrm{tr}(-2D^*D)$

```
>> R12=trace(-2*D'*D)
R12 = -6
```

- $R_{13} = \langle A^3, v \rangle$

```
>> R13=A(:,3)'*v
R13 = -1
```

- $R_{14} = A \circ B = [a_{ij}b_{ij}]$

```
>> R14=A.*B
R14 =

    0    1    0
   -1    0   -0
    0   -1    0
```

- $R_{15} = A \otimes B = [a_{ij}b_{kl}]$

```
>> R15=kron(A,B)
R15 =

    0   -2    2   -0    1   -1    0   -0    0
    2    0    0   -1   -0   -0    0    0    0
    0    2    0   -0   -1   -0    0    0    0
   -0    1   -1    0   -2    2   -0    1   -1
   -1   -0   -0    2    0    0   -1   -0   -0
   -0   -1   -0    0    2    0   -0   -1   -0
    0   -0    0   -0    1   -1    0   -2    2
    0    0    0   -1   -0   -0    2    0    0
    0    0    0   -0   -1   -0    0    2    0
```

- $R_{16} = AB^{-1}$

```
>> R16=A/B
R16 =

   -0    2   -1
   -1   -1    1
    2    0    1
```

- $R_{17} = B^{-1}A$

```
>> R17=B\A
R17 =

   -1    2   -1
   -0   -1    2
```

```
   2  -2   2
```

- Descomposición espectral (en autovalores) $A = P\Lambda P^{-1}$:

```
>> [P,Lambda]=eig(A);
>> A-P*Lambda/P
ans =

   6.6613e-16   2.2204e-16  -1.1102e-16
  -4.4409e-16  -4.4409e-16   0.0000e+00
  -1.8645e-16   4.4409e-16  -4.4409e-16
```

- Descomposición DVS en valores singulares $A = U\Sigma V^*$:

```
>> [U,Sigma,V]=svd(A);
>> A-U*Sigma*V'
ans =

   4.4409e-16   2.2204e-16   6.9797e-17
   2.2204e-16   6.6613e-16   8.8818e-16
  -6.7008e-17  -6.6613e-16  -4.4409e-16
```

- Definir una matriz cero de 4×3:

```
>> Z=zeros(4,3)
Z =

   0   0   0
   0   0   0
   0   0   0
   0   0   0
```

- Definir una matriz de unos de 3×5:

```
>> O=ones(3,5)
O =

   1   1   1   1   1
   1   1   1   1   1
   1   1   1   1   1
```

- Consideremos nuevamente la matriz A anteriormente ingresada:
  - Extraer la diagonal principal (diagonal 0) de A:

```
>> D0=diag(A)
D0 =

   2
   2
   2
```

  - Definir una matriz diagonal $D_A \in \mathbb{C}^{3\times 3}$ con la misma diagonal 0 que A:

```
>> DA=diag(diag(A))
DA =

Diagonal Matrix
```

```
   2   0   0
   0   2   0
   0   0   2
```

- Extraer las diagonales $1, -1, -2$ de A:

```
>> D1=diag(A,1)
D1 =

  -1
  -1

>> D_1=diag(A,-1)
D_1 =

  -1
  -1

>> D_2=diag(A,-2)
D_2 = 0
```

- Definir una matriz $K \in \mathbb{C}^{4\times 4}$ cuya diagonal 2 es igual a la diagonal -1 de A y todas sus demás entradas son cero:

```
>> K=diag(diag(A,-1),2)
K =

   0   0  -1   0
   0   0   0  -1
   0   0   0   0
   0   0   0   0
```

Recomendación: Para obtener más información sobre el funcionamiento de cualquier comando estándar de Octave/MATLAB basta escribir:

```
>> help nombre_del_comando
```

en la ventana de comandos.

1.1.2. Descomposiciones y funciones matriciales. Consideremos la representación vectorial del polinomio $p(z) = z^7 - 1$ en términos de sus coeficientes $\mathbf{p} = [p_7\ p_6\ \cdots\ p_1\ p_0] = [1\,0\,0\,0\,0\,0\,0\ -1]$, podemos ingresarlo en el sistema como:

```
>> p=[1,zeros(1,6),-1]
p =

   1   0   0   0   0   0   0  -1
```

Para calcular las raíces de $p(z)$ podemos utlizar la secuencia de comandos:

```
>> r=roots (p)
r =

  -0.90097 + 0.43388i
  -0.90097 - 0.43388i
```

```
  -0.22252 + 0.97493i
  -0.22252 - 0.97493i
   1.00000 + 0.00000i
   0.62349 + 0.78183i
   0.62349 - 0.78183i
```

Es posible evaluar p en un elemento cualquiera de su dominio utilizando el comando `polyval`, en particular la expresión:

```
>> polyval(p,-1)
ans = -2
```

permite calcular el valor $p(-1)$. La evaluación de un polinomio puede llevarse a cabo también en un rango de valores, por ejemplo, la expresión:

```
>> t=-1:1/4:1;
>> polyval(p,t)
ans =

 Columns 1 through 8:

  -2.00000  -1.13348  -1.00781  -1.00006  -1.00000  -0.99994  -0.99219  -0.86652

 Column 9:

   0.00000
```

calcula los valores $p(x_k)$ correspondientes a la partición $1 = t_0 < t_1 < \cdots < t_9 = 1$ del intervalo $[0, 1]$, donde $t_k = -1 + k * h$, $h = 1/4$.
Podemos ahora calcular la matriz compañera $C(p) \in \mathbb{C}^{7\times 7}$ de $p(z)$.

```
>> Cp=fliplr(flipud(compan(p)))
Cp =

   0   1   0   0   0   0   0
   0   0   1   0   0   0   0
   0   0   0   1   0   0   0
   0   0   0   0   1   0   0
   0   0   0   0   0   1   0
   0   0   0   0   0   0   1
   1  -0  -0  -0  -0  -0  -0
```

La teoría de matrices compañeras de polinomios nos asegura que dada una raíz r de $p(z)$, $r \in \sigma(C(p)^\top)$ (r es un autovalor de $C(p)^\top$) con autovector correspondiente $v_r = [1\, r\, r^2 \cdots r^{n-1}]^\top$, es posible verificar esto en particular para r_1, con las siguiente secuencia de comandos:

```
>> Cpt=Cp.';
>> v1=r(1).^(0:6)';
>> norm(Cpt*v1-r(1)*v1)
ans =    4.7018e-15
```

donde `norm(v)` es el comando utilizado para calcular el valor $\|v\|$ para $v \in \mathbb{C}^{n\times m}$. Podemos ahora calcular las matrices de diagonalización P y Q de $C(p)^\top$ y $C(p)$, respectivamente, utlizando matrices de Vandermonde con la siguiente secuencia de comandos:

```
>> P=fliplr(vander (r).');
>> Q=P.';
>> norm(diag(diag(P'*Cpt*P))-P'*Cpt*P)
ans =    1.4378e-14
>> norm(diag(diag(Q*Cp*Q'))-Q*Cp*Q')
ans =    1.4414e-14
```

De forma alternativa podemos calcular las matrices de diagonalización P y Q, los valores propios de $C(p)^\top$ y $C(p)$ utilizando las siguientes secuencias de comandos basadas en el comando `eig(A)` de Octave/MATLAB que calcula la descomposición espectral de $A \in \mathbb{C}^{n\times n}$:

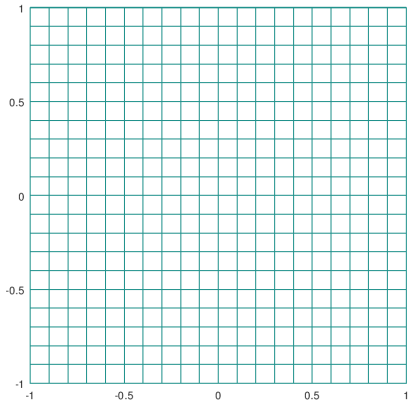
```
>> [Q,l]=eig(Cp);
>> [P,lt]=eig(Cpt);
>> [Q,l]=eig(Cp);
>> norm(diag(diag(P'*Cpt*P))-P'*Cpt*P)
ans =    2.6413e-15
>> norm(diag(diag(Q'*Cp*Q))-Q'*Cp*Q)
ans =    2.6672e-15
```

1.1.3. Lemniscatas y Pseudo-espectros. Consideremos nuevamente el polinomio de la sección anterior, una vez ingresado en el sistema con los comandos previos, podemos calcular las Lemniscatas de $p(z)$ para localizar sus raíces computacionalmente utilizando el siguiente procedimiento.

Cálculo de Lemniscata de p(z):

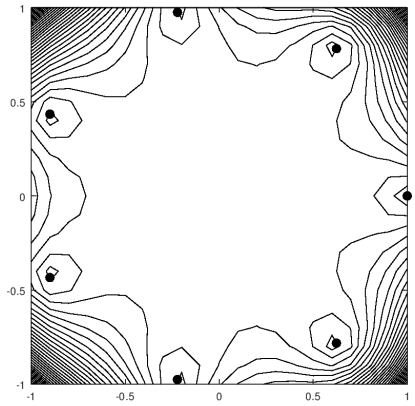
- Creación de malla inicial de la región $[-1, 1]^2$ determinada por el Teorema de Gerschgorin aplicado a $C(p)^\top$:

```
>> [x,y]=meshgrid (-1:2/20:1);
>> mesh(x,y,zeros(size(x)));
>> view (2)
>> axis square
```



- Cálculo de Lemniscatas de $p(z)$ en la malla inicial:

```
>> p=[1,zeros(1,6),-1];
>> r=roots (p);
>> contour(x,y,abs(polyval(p,x+i*y)),32,'k')
>> hold on
>> plot(r,'k.','markersize',25)
>> axis square
```

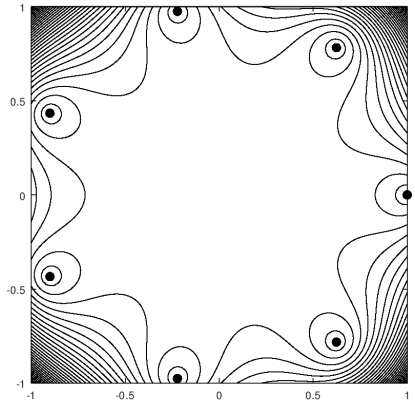


- Refinamiento de malla incial:

```
>> [x,y]=meshgrid (-1:2/100:1);
```

- Cálculo de Lemniscatas de $p(z)$ en la malla refinada:

```
>> close all
>> contour(x,y,abs(polyval(p,x+i*y)),32,'k')
>> hold on
>> plot(r,'k.','markersize',25)
>> axis square
```



El cálculo de Lemniscatas es importante para estudiar la estabilidad de sistemas dinámicos y de control discretos.
Definición: Dado $\varepsilon \geq 0$, se define el ε-Pseudo-espectro $\Lambda_\varepsilon(A)$ de $A \in \mathbb{C}^{n\times n}$ como el conjunto:

$$\Lambda_\varepsilon(A) = \left\{\lambda \in \mathbb{C} | \exists x \in \mathbb{C}^n \backslash \{0\}, E \in \mathbb{C}^{n\times n} : (A+E)x = \lambda x, \|E\| \leq \varepsilon\right\}$$

Podemos además calcular el Pseudo-espectro de $C(p)$ utilizando la siguiente secuencia de comandos basada en el comando `svd(A)` que calcula la descomposición en valores sigulares de A.

Cálculo de Pseudo-espectro de $C(p)$:

- Cálculo de malla de $[-1,1]^2$:

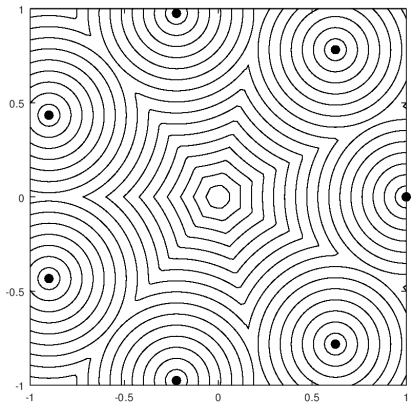
```
>> [x,y]=meshgrid (-1:2/100:1);
```

- Definición de la función `pspectra`:

```
>> pspectra=@(X)min(svd(X));
```

- Cómputo del Pseudo-espectro de $C(p)$:

```
>> Cp=fliplr(flipud(compan(p)));
>> N=size(x,1);
>> for k=1:N,for j=1:N,Z(k,j)=pspectra (Cp-(x(k,j)+i*y(k,j))*eye(7));
> end;end
>> contour(x,y,Z,16,'k')
>> l=eig(Cp);
>> hold on;
>> plot(l,'k.','markersize',25)
>> axis square
```



Al igual que el cómputo de Lemiscatas, el cómputo de Pseudo-espectros es importante para estudiar el compartamiento de sistemas dinámicos y de control discretos, importantes en la simulación numérica de procesos en ingeniería y ciencias.

1.2. Sistemas de Ecuaciones Lineales. En esta sección trabajaremos con algunas herramientas básicas de los programas orientados a matrices Matlab y Octave. Consideremos las siguientes matrices:

$$A = \begin{bmatrix} a & b & & e & b \\ b & a & & & \\ & & c & d & \\ e & & d & c & \\ b & & & & a \end{bmatrix} \tag{1.6}$$

$$f = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} \tag{1.7}$$

Donde supondremos que $a, b, c, d, e, b_1, ..., b_4 \sim N(0, 1)$. Tenemos que para ingresar las matrices anteriores en Matlab/Octave basta escribir:

```
>> a=randn;
>> b=randn;
>> c=randn;
>> d=randn;
>> e=randn;
>> A=[a b 0 e b;b a 0 0 0;0 0 c d 0;e 0 d c 0;b 0 0 0 a]
A =
-0.90411  -0.45550   0.00000  -1.63458  -0.45550
-0.45550  -0.90411   0.00000   0.00000   0.00000
 0.00000   0.00000   0.66103   0.25664   0.00000
-1.63458   0.00000   0.25664   0.66103   0.00000
-0.45550   0.00000   0.00000   0.00000  -0.90411
>> f=randn(5,1)
f =
  -1.49668
  -0.18249
  -0.44852
  -0.59657
   0.28638
>>
```

Para resolver el sistema de ecuaciones lineales

$$Ax = f$$

donde A y b estan dados por (1.6) y (1.7) respectivamente, basta con ejecutar la secuencia de comandos.

```
>> x=A\f
x =
   0.533964
  -0.067173
  -0.989978
   0.802238
  -0.585767
```

1.3. Normas Vectoriales y Matriciales. Consideremos un vector $x \in \mathbb{R}^7$ y una matriz $A \in \mathbb{R}^{7\times 7}$ generados al azar mediante la siguiente secuencia de comandos:

```
>> x=ceil(10*randn(7,1))
x =

     4
    12
    -3
     1
     4
   -10
   -10
>> A=floor(10*randn(7,7))
A =

   -24     2     8   -14   -12    16   -15
   -23   -12    -4    15    -8    -7    10
    -2     3    -2   -15     1     9    -3
    -6    18    -3     3    27     1    -5
    13     0     5    -4     7    13     8
    13     9    -1     9    14     8     1
   -18    -2   -20   -18    -4     4   -10
```

Las siguientes secuencias de comandos pueden utilizarse para calcular las normas $\|x\|_1$, $\|x\|_2$ y $\|x\|_\infty$, respectivamente:

```
>> norm(x,1)
ans =  44
>> norm(x,2)
ans =  19.647
>> norm(x,inf)
ans =  12
```

Para calcular las normas $\|A\|_1$, $\|A\|_2$ y $\|A\|_\infty$, respectivamente, pueden utilizarse las siguientes secuencias de comandos:

```
>> norm(A,1)
ans =  99
>> norm(A,2)
ans =  52.279
>> norm(A,inf)
ans =  91
```

2. Programación Lineal

Considerando el problema modelo del comando `glpk` de GNU Octave que consiste en un problema de optimización de la forma:

$$\min_{x \in \mathcal{S}} f(x) = c^\top x,$$
$$\mathcal{S} = \left\{ y \in \mathbb{R}^3 \;\middle|\; \begin{array}{l} Ay = b \\ y \geq d \end{array} \right\}$$

donde:

$$c = \begin{bmatrix} 10 \\ 6 \\ 4 \end{bmatrix}, A = \begin{bmatrix} 1 & 1 & 1 \\ 10 & 4 & 5 \\ 2 & 2 & 6 \end{bmatrix}, b = \begin{bmatrix} 100 \\ 600 \\ 300 \end{bmatrix}, d = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

La siguiente secuencia de comandos puede utilizarse para resolver el problema.

```
>> c = [10, 6, 4]';
>> A = [ 1, 1, 1;10, 4, 5;2, 2, 6];
>> b = [100, 600, 300]';
>> lb = [0, 0, 0]';
>> ub = [];
>> ctype = "UUU";
>> vartype = "CCC";
>> s = -1;
>> param.msglev = 1;
>> param.itlim = 100;
>>[xmin, fmin, status, extra] = ...
> glpk (c, A, b, lb, ub, ctype, vartype, s, param);
```

La solución aproximada $x \in \mathcal{S}$ que minimiza $f(x)$ puede visualizarse en la ventana de comandos utilizando la secuencia de comandos:

```
>> xmin
xmin =

   33.33333
   66.66667
    0.00000
```

Para resolver el problema derivado:

$$\min_{x \in \mathcal{S} \cap \mathbb{Z}^3} f(x) = c^\top x,$$

es suficiente modificar la variable `vartype` utilizando la siguiente secuencia de comandos:

```
>> vartype = "III";
```

la solución aproximada x del problema puede calcularse y visualizarse utilizando la siguiente secuencia de comandos:

```
[xmin, fmin, status, extra] = ...
> glpk (c, A, b, lb, ub, ctype, vartype, s, param);
Long-step dual simplex will be used
octave:16> xmin
xmin =
```

```
33
67
 0
```

REFERENCIAS

[1] J. W. Eaton (2020). GNU Octave (version 5.2.0) Documentation. https://octave.org/doc/v5.2.0/

[2] R. L. Burden, D. J. Faires, A. M. Burden. (2017). Análisis Numérico. 10a Ed. Cengage Learning Editores.

[3] Quarteroni A., Saleri F., Gervasio P. (2014). Scientific computing with MATLAB and Octave (Textbook).

[4] D. G. Luenberger, Y. Ye. (2016). Linear and Nonlinear Programming. 4a Ed. Springer International Publishing Switzerland.