Resum d'ordres i procediments Octave a Circuits i Sistemes Lineals

Orestes Mas

12 de febrer de 2023

1 Ús bàsic

Octave és un programari lliure de matemàtica numèrica[1]. Podem utilitzar Octave simplement com a calculadora, escrivint directament les expressions matemàtiques que volem calcular, amb característiques molt útils en una carrera d'enginyeria:

- Permet operar amb **nombres complexos** de forma natural.
- Té integrades les funcions més habituals: trigonomètriques, exponencials, etc.
- Treballa de forma nativa amb **matrius**, polinomis i estructures més complexes.
- Permet assignar nombres a **variables** i escriure **programes**.

Per exemple (les línies que comencen amb «#» són comentaris explicatius que no s'executen):

```
# sin^2(x) + cos^2(x) = 1 per qualsevol x
>> sin(20)**2 + cos(20)**2
ans =  1

# Expressem 1/√2 en decibels
>> Guany = 20*log10(1/sqrt(2))
Guany = -3.0103

# Complex de mòdul 2 i fase 60° (=π/3)
>> Z = 2*exp(j*pi/3)
Z =  1.0000 + 1.7321i

# Calculem el mòdul (abs) i la fase (angle)
>> abs(Z)          # Mòdul (= valor absolut)
ans =  2
>> angle(Z)*180/pi # Fase (convertida a graus)
ans =  60.000
```

[1] És a dir, amb altes capacitats de processar nombres. Val a dir que, aplicant algunes extensions, també té certes capacitats per manipular expressions simbòliques, però hi ha altres programes més adaptats a aquesta funció

Noteu que si no especifiquem una variable on assignar els resultats, aquests sempre van a parar a la variable «ans», que se sobreescriu en cada nova operació. Així mateix, si acabem cada ordre amb un punt i coma «;», els resultats no es mostren en pantalla.

Al final d'aquest document teniu un full resum amb les ordres i operacions més usuals en Octave. A banda de les funcions integrades, podem afegir funcionalitat sobre la marxa carregant una gran varietat de paquets addicionals especialitzats en àrees concretes de coneixement, com ara processament del senyal, teoria de control, economia, algorismes genètics, xarxes neuronals, etc.

2 Vectors i matrius

Els vectors i matrius s'especifiquen com un conjunt de nombres separats per espais o comes (si estan en la mateixa fila) o bé separats per punt i coma si estan en files diferents.

La manipulació de vectors i matrius és una de les característiques principals i més potents d'Octave, permetent l'obtenció de resultats complexos de forma simple i ràpida gràcies a la utilització d'algorismes molt optimitzats:

```
>> b = [4; 9; 2]; # Vector columna
>> A = [ 3 4 5;
         1 3 1;
         3 5 9 ]; # Matriu 3x3
>> x = A \ b  # Resol el sistema Ax = b
x =
  -1.5000
   4.0000
  -1.5000
```

Doneu un cop d'ull al full resum i a la documentació general per aprendre més ordres de creació i manipulació de vectors i matrius.

2.1 Generació de valors equiespaiats

Un dels usos dels vectors és la de servir com a dades d'abscisses i ordenades en el dibuix de gràfics. Per a crear un conjunt de nombres equiespaiats usem les funcions `linspace` i `logspace`. La primera s'utilitza principalment en eixos de temps, on normalment els instants de temps estan (equi)espaiats linealment, i la segona s'usa principalment en eixos de freqüència, on normalment l'espaiat és logarítmic.

La funció `logspace` té la sintaxi[2] següent: `logspace(<ei>,<ef>[,N])`, que crea un vector fila amb N valors espaiats logarítmicament[3] que van de 10^{ei} fins a 10^{ef}. Els noms «ei» i «ef» signifiquen, per tant, «exponent inicial» i «exponent final».

Per exemple, si volem crear un vector amb 500 punts de freqüències que vagi de 10 Hz a 100 kHz farem:

```
>> f = logspace(1,5,500);
```

3 Polinomis

3.1 Definició

Un polinomi del tipus

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots a_1 x + a_0$$

es descriu amb un vector format pels seus coeficients: `p = [`a_n, a_{n-1}, $\cdots$, a_1, a_0`]`

Per exemple, suposem que tenim la funció de xarxa

$$H_1(s) = \frac{5000}{s + 1000} \tag{1}$$

que és la corresponent a un filtre passabaix de primer ordre amb $\omega = 1000$ rad/s i $H_{màx} = 5$, en Octave definirem el seus polinomis numerador i denominador de la forma següent:

```
>> num = [5000];
>> den = [1, 1000];
```

[2] Usem la mateixa notació del full resum

[3] El **quocient** entre 2 elements consecutius és sempre el mateix

3.2 Avaluació

Per tal de trobar el valor que pren un polinomi $p(x)$ en un punt $x = x_0$ usem la funció `polyval(`p,x_0`)`. El punt d'avaluació pot ser un nombre complex, de manera que per avaluar la funció de xarxa de l'equació (1) en el punt $s = 500\text{j}$ farem:

```
>> H = polyval(num,500j)/polyval(den,500j)
ans =  4 - 2i
```

Aquest resultat és el mateix que hauríem obtingut manualment fent:

$$H_1(500\text{j}) = \frac{5000}{500\text{j} + 1000} = \frac{10}{2 + \text{j}}$$
$$= \frac{10 \times (2 - \text{j})}{5} = 4 - 2\text{j}$$

També podem avaluar en un *conjunt de punts*, passant a la funció `polyval` un vector o llista de valors en lloc d'un sol valor. En aquest cas cal tenir en compte que si la nostra expressió involucra quocients de vectors hem d'usar l'operador «`./`» (punt-barra) en lloc de la barra sola per fer la divisió **element a element**:

```
>> w = [500; 1000; 2000];
>> H = polyval(num,j*w)./polyval(den,j*w)
ans =

4.0000 - 2.0000i
2.5000 - 2.5000i
1.0000 - 2.0000i
```

3.3 Arrels de polinomis

Un polinomi de grau N té sempre N arrels en el cos dels nombres complexos. Per tal de calcular-les amb Octave usarem la funció `roots`, passant el polinomi com a paràmetre. Per exemple, pel polinomi $p(x) = x^3 + 21x^2 + 115x + 375$ les arrels són $x = -15$ i $x = -3 \pm 4\text{j}$:

```
>> p = [1 21 115 375];
>> roots(p)
ans =

  -15.0000 +       0i
   -3.0000 +  4.0000i
   -3.0000 -  4.0000i
```

3.4 Producte de polinomis

El coeficient del terme x^n del producte de 2 polinomis $f(x) \cdot g(x)$ és $\sum_{i+j=n} f_i\, g_j$, on f_i i g_j són els coeficients dels termes de grau i i j dels polinomis $f(x)$ i $g(x)$, respectivament.

Si apliquem aquest algorisme a tots els coeficients del polinomi producte, l'operació resultant es coneix amb el nom de *convolució* i Octave la implementa amb el nom `conv`.

Per exemple, $(x^2 - 6x + 25)(x + 2) = x^3 - 4x^2 + 13x + 50$:

```
>> p = conv([1 -6 25],[1 2])
p =

    1   -4   13   50
```

4 Resposta temporal de circuits i sistemes lineals

Hi ha diverses funcions d'Octave que ens poden ajudar a determinar la resposta temporal de circuits o sistemes lineals. Unes serveixen per descriure i manipular (associar-los en cascada, en paral·lel...) sistemes, i altres serveixen per calcular la seva resposta, en part o totalment.

4.1 Descripció de sistemes lineals

A l'assignatura de CSL descrivim i operem amb els sistemes lineals en el **domini transformat de Laplace**.

Descripció polinòmica Les funcions transformades que utilitzarem a CSL seran sempre quocients de polinomis, per la qual cosa la manera més simple de descriure-les en Octave és mitjançant els polinomis numerador i denominador.

Descripció sistèmica La descripció anterior no serveix per sistemes complexos com, per exemple, sistemes amb múltiples entrades i sortides. En aquests casos cal introduir el sistema d'una manera diferent, que no tractarem aquí.

4.2 Determinació de zeros i pols

Els zeros i els pols d'una funció transformada $F(s)$ no són sinó les arrels dels polinomis numerador i denominador d'aquesta funció, respectivament. Per tant, per determinar-los usarem la funció **roots** com s'ha comentat a la secció 3.3.

Per exemple, els pols i zeros del circuit descrit per la funció de xarxa $H(s) = \frac{s^2+16}{s^2+7s+10}$ els trobarem fent:

```
>> num = [1 0 16];
>> den = [1 7 10];
>> roots(num)
ans =

   0 + 4i
   0 - 4i

>> roots(den)
ans =

  -5
  -2
```

4.3 Descomposició en fraccions parcials

Una de les operacions més habituals en el determinació de la resposta temporal de sistemes lineals és el càlcul de la descomposició en fraccions parcials de la resposta. Per ajudar a trobar-la, Octave disposa d'una funció molt útil anomenada `residue`. Els seus paràmetres són:

- Com a **entrades** li hem d'especificar els polinomis numerador i denominador de la funció transformada a descomposar.
- Com a **sortides** retorna 4 vectors:
  1. Llista de residus de cada fracció parcial.
  2. Llista de pols associats a cada fracció parcial.
  3. Polinomi resultant en cas de fraccions impròpies.
  4. Multiplicitat dels pols anteriors.

Per exemple, la resposta al graó d'un circuit descrit per $H(s) = \frac{s-30}{s+10}$ és:

$$R(s) = H(s) \cdot \frac{1}{s} = \frac{s-30}{s(s+10)} = \frac{s-30}{s^2+10s}$$

La seva descomposició en fraccions parcials es pot trobar fent:

```
>> num = [1 -30];
>> den = [1 10 0];
>> [r,p,k,m] = residue(num,den)
r =

  -3
   4

p =

    0
  -10

k = [](0x0)
m =

   1
   1
```

Aquest resultat subministra tota la informació necessària per plantejar la descomposició en fraccions simples i trobar la transformada inversa:

$$R(s) = \frac{3}{s} + \frac{4}{(s+10)} \tag{2}$$

$$r(t) = 3\,\mathrm{u(t)} + 4\,e^{-10\,t}\,\mathrm{u(t)} \tag{3}$$

Posem un altre exemple, aquesta vegada més complicat. Volem invertir la funció transformada:

$$R(s) = \frac{10s^3 + 72s^2 + 16s + 50}{s^4 + 8s^3 + 25s^2}$$

Amb Octave farem:

```
>> num = [10 72 16 50];
>> den = [1 8 25 0 0];
>> [r,p,k,m]=residue(num,den)
r =
   5 - 5i
   5 + 5i
   0 + 0i
   2 + 0i

p =
  -4 + 3i
  -4 - 3i
   0 + 0i
   0 + 0i

k = [](0x0)
m =
   1
   1
   1
   2
```

Tenim, per tant, 4 pols: 2 d'ells són complexos (conjugats) amb residus també complexos i dos més que són reals, els quals de fet són el mateix (un pol a l'origen) que té multiplicitat 2.

Pel pas següent és convenient expressar els residus $r_{\{1,2\}} = 5 \mp 5\mathrm{j}$ en forma polar:

```
>> abs(r(1)), angle(r(1))*180/pi
ans = 7.0711  # Això és 5·√2
ans = -45     # Això és π/4 en radians
```

Per tant $r_{\{1,2\}} = 5\sqrt{2}\angle{\mp 45°}$. Amb tota aquesta informació, la descomposició en fraccions simples és:

$$R(s) = \frac{5\sqrt{2}\angle{-45°}}{s-(-4+3\mathrm{j})} + \frac{5\sqrt{2}\angle{+45°}}{s-(-4-3\mathrm{j})} + \frac{0}{s} + \frac{2}{s^2}$$

i d'aquí la transformació inversa és immediata, terme a terme, tenint en compte que els dos primers termes els agrupem per tal d'obtenir funcions temporals que tinguin coeficients reals:

$$r(t) = \left[10\sqrt{2}\cdot e^{-4\,t}\cos\left(3\,t - \frac{\pi}{4}\right) + 2\,t\right]\mathrm{u}(t)$$

5 Obtenció de corbes de resposta freqüencial

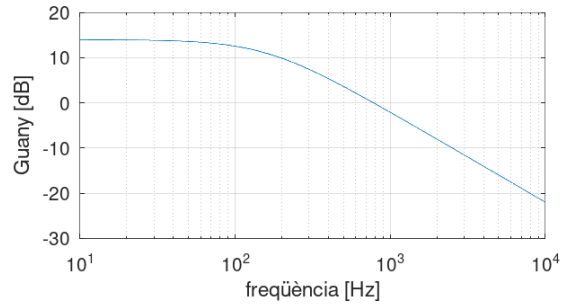
Donada una funció de xarxa $H(s)$, les corbes de resposta freqüencial són la representació gràfica de les funcions $20\log\left(|H(\mathrm{j}\omega)|\right)$ i $\measuredangle H(\mathrm{j}\omega)$ en funció de la freqüència, usant una escala logarítmica a l'eix d'abcisses.

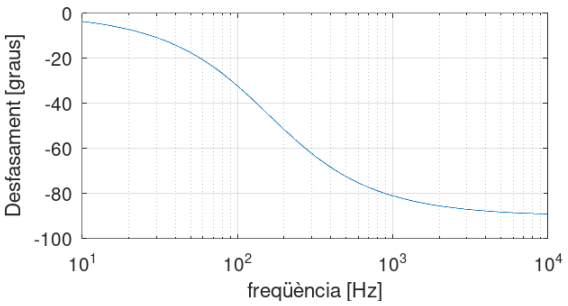
Així, per pintar aquestes corbes seguirem les passes següents:

1. Definirem un vector de freqüències, generalment amb espaiat logarítmic.
2. Avaluarem la funció de xarxa en cadascuna de les freqüències anteriors usant la funció `polyval` vista a la secció anterior.
3. Calcularem els vectors de guany i desfasament.
4. Pintarem les corbes usant la funció `semilogx`.

Per exemple, suposem que volem pintar el diagrama de Bode de la funció de xarxa (1) entre 10 Hz i 10 kHz. Ho podem fer usant el codi següent:

```
>> num = 5000; den = [1 1000];
>> f = logspace(1,4,500);
>> w = 2*pi*f;
>> H = polyval(num,j*w)./polyval(den,j*w);
>> Guany = 20*log10(abs(H));
>> Desfasament = angle(H)*180/pi;
>> semilogx(f,Guany),grid
>> xlabel('freqüència [Hz]')
>> ylabel('Guany [dB]')
>> figure()
>> semilogx(f,Desfasament),grid
>> xlabel('freqüència [Hz]')
>> ylabel('Desfasament [graus]')
```



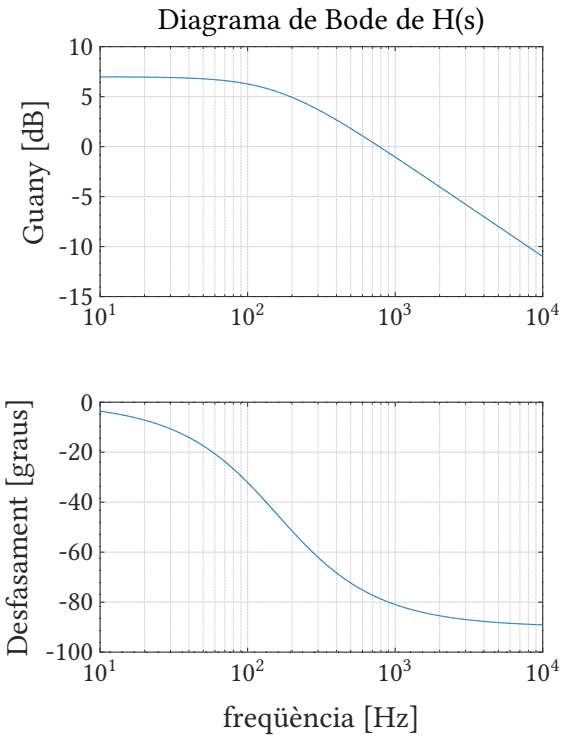


Una vegada tingueu les gràfiques en pantalla podeu usar el menú «Fitxer -> Desa com a...» per desar la figura en un format (PNG per exemple) que després pugueu incorporar al vostre document.

NOTA: En algunes versions d'Octave la impressió del gràfic falla si teniu **caràcters accentuats** en el text.

També pot ser útil emprar la funció «`subplot`» per posar les corbes d'amplificació i desfasament en una sola figura:

```
# 2 files, 1 columna, seleccionem el 1r.
>> subplot(2,1,1)
>> semilogx(f,Guany),grid
>> title('Diagrama de Bode de H(s)')
>> ylabel('Guany [dB]')
# 2 files, 1 columna, seleccionem el 2n.
>> subplot(2,1,2)
>> semilogx(f,Desfasament),grid
>> xlabel('freqüència [Hz]')
>> ylabel('Desfasament [graus]')
```



MATLAB/GNU Octave quick reference sheet

Last revision: February 21, 2013

By Anders Damsgaard Christensen,
anders.damsgaard@geo.au.dk, http://cs.au.dk/~adc.

The < > symbols denote required arguments, [ ] args. are optional. The bracketing symbols should not be written.

General session control

`whos`	List all defined variables
`clear`	Delete all defined variables
`clc`	Clear home screen
`edit <file>[.m]`	edit file, create if it doesn't already exist
`save '<filename>'`	Save all variables to `<filename>.mat`
`load '<filename>'`	Load variables from `<filename>.mat`
`help <command>`	Quick help on `command`
`doc <command>`	Extensive help on `command`

Variables

When assigning variables, the values will be displayed. This can be suppresed by adding the suffix ;

`ans`	Value of last calculation
`x = 1`	Define variable x to be a scalar with value 1
`x = y`	Set x equal to y
`v = [1,2,3]`	Define variable v to be a row vector with values $\begin{pmatrix}1&2&3\end{pmatrix}$
`v = [1;2;3]`	Define variable v to be a column vector
`M = [1,2,3;4,5,6]`	Define variable M to be a matrix with values $\begin{pmatrix}1&2&3\\4&5&6\end{pmatrix}$
`v = <s>[:st]:<e>`	Create a row vector with values from `s` to `e` with a step size of `st`
`v = linspace(<s>,<e>[,st])`	Create a row vector with values from `s` to `e` with `st` intermediate values
`A = zeros(<N>[,M])`	Create a $N \times M$ matrix with values 0
`A = ones(<N>[,M])`	Create a $N \times M$ matrix with values 1
`A = rand(<N>[,M])`	Create a $N \times M$ matrix with uniformly distr. values in $[0, 1[$
`A = randn(<N>[,M])`	Create a $N \times M$ matrix with normal (Gaussian) distr. values with $\mu = 0, \sigma = 1$

Arithmetic and standard functions

In general, the number of elements returned equal the dimensions of the input variables. `a*b` $= ab$, `a/b` $= \frac{a}{b}$, `a**b` $= a^b$, `a%b`: remainder, `sqrt(a)` $= \sqrt{a}$, `a**(1/b)` $= \sqrt[b]{a}$, `abs(a)` $= |a|$, `log(a,b)` $= \log_b(a)$ `sin(a)` $= \sin(a)$,
`M.*N`: element-wise multiplication of two vectors/matrices
`M*N`: multiplication of two vectors/matrices
`A(:)`: show matrix as vector
`A'`: Transpose of vector/matrix
`C=[A;B]`: Concentrate two vectors/matrices
`size(A)`: Dimensions of vector/matrix
`sum(A)`: Column sum of vector/matrix
`inv(A)`: Inverse of matrix
`det(A)`: Determinant of matrix
`A\b`: For a matrix A and col. vector b find solution x to $Ax = b$
Constants: `pi` $= \pi$, `e` $= e$, `i` $= i$, `inf` $= \infty$

Vector/matrix slicing

In the following, `n` and `m` can be single values or vectors.

`v(<n>)`	The n-th value of vector v
`v(1,<n>)`	The 1st to n-th value of vector v
`v(<n>,end)`	The n-th value to the end of vector v
`M(<n>,<m>)`	The n, m-th value of matrix M
`M(<n>,:)`	The n-th row of matrix M
`M(<n>,:)`	The n-th row of matrix M
`I = find(X > 2)`	Find indexes in X where the value is greater than 2

Plotting and visualization

In the following, `n` and `m` can be single values or vectors.

`figure`	Create new figure window
`plot(x,y)`	Plot vector y as a function of x with a line
`plot(x,y,'*')`	Plot vector y as a function of x with points
`plot(x,y,'*-')`	Plot vector y as a function of x with a line and points
`semilogx(x,y)`	Plot vector y as a function of x, with x on a log scale
`semilogy(x,y)`	Plot vector y as a function of x, with y on a log scale
`loglog(x,y)`	Plot vector y as a function of x, on a loglog scale
`hist(x)`	Plot a histogram of values in x
`grid`	Show numeric grid in the plot background
`axes equal`	Set a 1:1 aspec ratio on the plot axes
`title('bla')`	Set a plot title
`xlabel('bla')`	Set x-axis label

Custom functions

elations: `==`, `~=`, `>`, `<`, `<=`, `>=`
Conditional structures:
`if expr ...[elseif ...] [else ...] end`
Iteration structures: `for var=expr ... end`
Function syntax:

```
function [out1, ...] = name (par1, ...)
...
end
```

MATLAB reference manual

`http://www.mathworks.se/help/matlab/index.html`

GNU Octave reference manual

`https://www.gnu.org/software/octave/doc/interpreter/`

MATLAB/GNU Octave wikibook

`https://en.wikibooks.org/wiki/MATLAB_Programming/GNU_Octave`

Introduction to MATLAB

`www.mathworks.com/moler/intro.pdf`