Departament de Disseny i Programació de Sistemes Electrònics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Anàlisi de circuits amb Octave

Enginyeria de Sistemes TIC (iTIC)
EPSEM - UPC

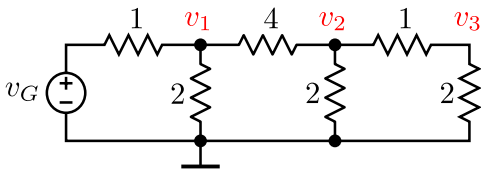
Pere Palà
Rosa Giralt

Setembre de 2012

En aquest document expliquem com podem ajudar-nos amb Octave per analitzar tant circuits resistius com dinàmics. Octave *no* és un programa d'anàlisi de circuits, però es pot fer servir per als càlculs que cal dur a terme. Octave o Matlab, la versió no lliure en què s'inspira, són **les** eines matemàtiques que fan servir els enginyers electrònics i/o de telecomunicació.

1 Plantejament del problema

Comencem pels circuits resistius. Considerem un circuit elemental, com el que es representa a continuació:



Tenim tres tensions nodals desconegudes, v_1, v_2 i v_3. Per tant, plantegem 3 KCLs i obtenim el sistema d'equacions

$$\begin{bmatrix} \frac{1}{1}+\frac{1}{2}+\frac{1}{4} & -\frac{1}{4} & 0 \\ -\frac{1}{4} & \frac{1}{4}+\frac{1}{2}+\frac{1}{1} & -\frac{1}{1} \\ 0 & -\frac{1}{1} & \frac{1}{1}+\frac{1}{2} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} v_g \\ 0 \\ 0 \end{bmatrix}, \tag{1}$$

que podem expressar matricialment de forma compacta com

$$\mathbf{T}\mathbf{v} = \mathbf{w} \tag{2}$$

Per resoldre aquest sistema d'equacions, podem ajudar-nos de `octave`, que és l'objectiu d'aquest document.

2 Primers passos amb Octave

En primer lloc, cridem `octave`. Està als repositoris de linux però també hi ha versió per windows.

```
xxx@yyy-u:~$ octave
GNU Octave, version 3.0.5
Copyright (C) 2008 John W. Eaton and others.
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE.  For details, type 'warranty'.
...
octave:1>
```

En primer lloc, creem la matriu $\mathbf{T}$, especificant cadascun del seus elements:

```
octave:1> T=[1+1/2+1/4 -1/4 0; -1/4 1/4+1/2+1 -1; 0 -1 1+1/2]
T =
   1.75000  -0.25000   0.00000
  -0.25000   1.75000  -1.00000
   0.00000  -1.00000   1.50000
```

Observeu que les files es separen amb espais en blanc (també es podria fer amb coma ",") i les columnes amb punt i coma ";".

A continuació entrem el vector de dades $\mathbf{w}$ (el terme de la dreta de l'equació (1)):

```
octave:2> w=[1; 0; 0]
w =
   1
   0
   0
```

Observareu que hem posat 1 en comptes de v_g. En un sistema lineal la sortida és proporcional a l'entrada. Si coneixem la sortida per una entrada 1, la sortida per una entrada 2 serà el doble i la sortida per una entrada v_g serà v_g vegades més gran (o petita).

Ara, per resoldre el sistema d'equacions, és a dir obtenir el vector $\mathbf{v}$, (aquí comença la màgia de l'Octave) amb una simple instrucció n'hi ha prou:

```
octave:3> v=T\w
v =
   0.590909
   0.136364
   0.090909
```

Aquesta expressió és equivalent a haver calculat explícitament la inversa de la matriu $\mathbf{T}$ i multiplicar-la pel vector $\mathbf{w}$, com es fa matemàticament. Això s'escriuria així:

```
octave:3> v=inv(T)*w
v =
   0.590909
   0.136364
   0.090909
```

No obstant, per problemes de més dimensió és més eficaç la primera notació, que només demana resoldre el sistema d'equacions i no necessàriament calcular la matriu inversa.

Ara tenim totes les tensions nodals al vector $\mathbf{v}$. Si volem accedir a una de les variables, per exemple la tensió de sortida al node 3, v_3:

```
octave:4> v(3)
ans =  0.090909
```

Aquest valor és la sortida per $v_g = 1$. En general, la sortida serà $v_3 = 0.90900 v_g$.

El resultat ens queda a la variable *ans*, que podem aprofitar per qualsevol altra cosa. I amb això, ja podem calcular qualsevol cosa. Per exemple, el corrent que circula per la resistència de 4Ω val:

```
octave:5> i_4=(v(1)-v(2))/4
i_4 =  0.11364
```

Aquest valor, novament, és la sortida per $v_g = 1$. Si volem l'expressió general, sols cal tenir en compte la proporcionalitat amb v_g, de forma que $i_4 = 0.11364 v_g$.

Qualsevol altra variable es calcularia de forma inmediata a partir de les variables generadores calculades.

3 Càlculs freqüents en circuits dinàmics

En `octave`, podem representar un polinomi mitjançant un vector. El conveni és que els vectors comencen pel terme de major grau. Així, el polinomi

$$P(s) = s^4 + 3s^3 + 4s^2 + 5s + 1 \tag{3}$$

pot ser descrit mitjançant un vector, de la forma:

```
octave:1> P=[1 3 4 5 1]
P =
   1   3   4   5   1
```

Per calcular les arrels d'aquest polinomi, tenim l'ordre `roots`:

```
octave:2> roots(P)
ans =
  -2.12074 + 0.00000i
  -0.32072 + 1.37110i
  -0.32072 - 1.37110i
  -0.23781 + 0.00000i
```

Observem que aquest polinomi té dues arrels reals negatives i un parell d'arrels complexes conjugades.

Per saber el valor d'un polinomi en un punt qualsevol (real o complexe) podem fer servir l'ordre `polyval`. Per exemple, per avaluar el valor del polinomi anterior a $s = 2+2j$, escriurem

```
octave:3> polyval(P,2+2*j)
ans = -101 +  90i
```

El resultat d'aquestes ordres queda emmagatzemat per defecte a la variable `ans`, però ho podem assignar a qualsevol variable que ens interessi guardar per operacions posteriors.

Per exemple, per comprovar que la primera de les arrels que hem calculat és, efectivament, una arrel, podem fer

```
octave:7> r=roots(P);
octave:8> polyval(P,r(1))
ans =  1.2434e-14
```

Però també podem comprovar totes les arrels simultàniament. Donat que `octave` és una eina orientada a vectors, la majoria de funcions operen sobre vectors igual que sobre escalars:

```
octave:9> polyval(P,r)
ans =
  1.2434e-14 + 0.0000e+00i
  -8.6597e-15 + 1.3101e-14i
  -8.6597e-15 - 1.3101e-14i
  0.0000e+00 + 0.0000e+00i
```

Veiem, doncs, que les arrels s'han calculat amb una precisió de l'ordre de 10^{-14}.

De cara a la presentació dels resultats, podem jugar amb la comanda `format`. Si busqueu informació sobre la comanda, `doc format`, trobareu informació al respecte. De moment, podeu provar què passa amb les ordres següents

```
octave:1> pi
ans =  3.1416
octave:2> format short
octave:3> pi
ans =  3.1416
octave:> format short e
octave:4> pi
ans =  3.1416e+00
octave:5> format long e
octave:6> pi
ans =  3.14159265358979e+00
octave:7> format bank
octave:8> pi
ans = 3.14
```

4 Descomposició en fraccions simples

Considerem una funció $F(s)$ com la següent

$$F(s) = \frac{s^2 + 4.5s + 2}{s^4 + 5s^3 + 10s^2 + 10s + 4} \tag{4}$$

Després d'emplenar els vectors `num` i `den` adequadament, fem servir la comanda `residue` per calcular els anomenats *residus* corresponents als pols de $F(s)$:

```
octave:1> [r,p,k,e]=residue(num,den)
r =
  1.5000e+00 - 6.9389e-16i
  9.9920e-16 - 1.2500e+00i
  9.2635e-16 + 1.2500e+00i
  -1.5000e+00 + 3.0866e-17i
p =
  -2.00000 + 0.00000i
  -1.00000 + 1.00000i
  -1.00000 - 1.00000i
  -1.00000 + 0.00000i
k = [](0x0)
e =
   1
   1
   1
   1
```

És a dir, el resultat ens està dient que

$$F(s) = \frac{s^2 + 4.5s + 2}{s^4 + 5s^3 + 10s^2 + 10s + 4} = \frac{1.5}{s+2} + \frac{-1.25j}{s+1-j} + \frac{1.25j}{s+1+j} + \frac{-1.5}{s+1} \tag{5}$$

En aquesta forma, la transformada inversa és molt més senzilla de realitzar. El vector `e` conté informació sobre la multiplicitat del pol corresponent. Podem comprovar-ho amb un exemple senzill, on el denominador és $(s+1)^2 = s^2 + 2s + 1$:

$$F(s) = \frac{1}{s^2 + 2s + 1} \tag{6}$$

```
octave:1> [r,p,k,e]=residue(1,[1 2 1])
r =
   0
   1
p =
  -1
  -1
k = [](0x0)
e =
   1
   2
```

El resultat ens està dient que la descomposició en fraccions simples és, com ja sabem:

$$F(s) = \frac{1}{s^2 + 2s + 1} = \frac{0}{(s+1)^1} + \frac{1}{(s+1)^2} \tag{7}$$

Finalment, cal esmentar el vector `k`, format per la divisió dels polinomis, quan el grau del numerador és igual o superior al del denominador. Per exemple, si

$$F(s) = \frac{s+2}{s+1} \tag{8}$$

aleshores $F(s)$ es pot escriure com

$$F(s) = 1 + \frac{1}{s+1} \tag{9}$$

Si provem de fer la descomposició en fraccions simples mitjançant `octave`, obtindrem el següent resultat

```
octave:1> [r,p,k,e]=residue([1 2],[1 1])
r =  1
p = -1
k =  1
e =  1
```

Si ens hi fixem atentament, veurem que aquest és precisament el resultat de la equació 9. En general, `k` és un vector que representa el polinomi resultant de fer la divisió. En general, pels circuits *ben comportats*, el grau del numerador és inferior o, com a molt, igual al denominador. Per tant, com a molt tindrem una constant (que tindrà com a transformada inversa de Laplace una funció delta de Dirac.

5 Gràfic de la transformada inversa de Laplace

Si un sistema té una funció de xarxa $H(s)$, la sortida, en el domini transformat de Laplace, vindrà donada per $V_O(s) = H(s)V_I(s)$. Per tornar al domini del temps, caldrà calcular la transformada inversa de $V_O(s)$. Si calculem la transformada inversa de $H(s)$, això equival a calcular la sortida quan $V_I(s) = 1$, és a dir, quan $v_I(t) = \delta(t)$. Per tant, la transformada inversa de $H(s)$ s'anomena resposta impulsional, perquè és la resposta a l'impuls unitari.

Quan vulguem visualitzar una transformada inversa, podem fer servir la funció `impulse`, que calcula la resposta impulsional d'un sistema, és a dir, la resposta quan l'entrada val 1 al domini s. La forma d'emprar aquesta funció, a partir d'un numerador i un denominador donats és

```
octave:12> impulse(tf(num,den))
```

on `tf` és necessari per convertir el format "quocient de polinomis" en el format adequat.

Si afegim

```
octave:13> [y,t]=impulse(tf(num,den));
```

tindrem els vectors `t` i `y` disponibles. El ";" al final, fa que el resultat no es mostri a la pantalla. Per exemple, ara podem calcular el màxim de `y`

```
octave:14> max(y)
ans =  0.43562
```

i trobar l'instant de temps en què es produeix:

```
octave:15> t(y==max(y))
ans =  0.85714
```

6 Gràfic de qualsevol cosa

Sovint tenim la necessitat de fer una representació gràfica d'un senyal. Suposem, per exemple, que volem representar el senyal

$$v(t) = 4e^{-5t}\cos\left(2\pi 5t + \frac{\pi}{8}\right) \tag{10}$$

Octave treballa amb vectors. Així que el primer pas serà crear un vector que contingui els valors de l'eix de temps t on volem avaluar la funció. Per exemple, si tenim intenció de visualitzar un senyal en l'interval $[0 \cdots 10]$ aleshores podem escriure

```
octave:1> t=linspace(0,10,5)
t =
    0.00000    2.50000    5.00000    7.50000   10.00000
```

Així hem creat un vector t que conté 5 valors equiespaiats entre 0 i 10.

Octave sap operar amb vectors i fa extensions interessants de les operacions tradicionals amb vectors. Per exemple, sap què vol dir $\sin(t)$, quan t és un vector:

```
octave:2> sin(t)
ans =
   0.00000   0.59847  -0.95892   0.93800  -0.54402
```

És a dir, $\sin(t)$ retorna un vector de 5 elements, que conté $\sin(0)$, $\sin(2.5)$, etc. fins $\sin(10)$.

El mateix passa amb $\exp(-t)$, l'exponencial e^{-t}:

```
octave:3> exp(-t)
ans =
   1.0000e+00   8.2085e-02   6.7379e-03   5.5308e-04   4.5400e-05
```

Per fer operacions entre variables que, en realitat, són vectors, simplement cal recordar com funcionen les operacions entre vectors. Així, la suma i la resta de vectors funcionen igual que quan treballem amb escalars. Però la cosa canvia parlant del producte i del quocient!

Si a i b són vectors, no té sentit la operació $a \times b$, a no ser que un sigui un vector fila i l'altre un vector columna i el resultat és el que manen les regles de l'àlgebra –i probablement no són el que busquem:

Per fer la gràfica de $\exp(-t) \times \sin(t)$, probablement volem aconseguir que els vectors resultants de cada operador siguin *multiplicats element a element* entre sí: el primer pel primer, el segon pel segon, etc.

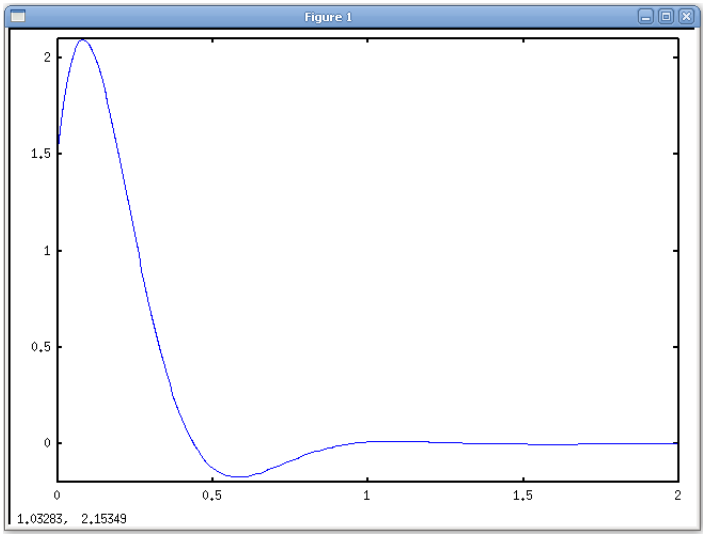
Per fer-ho hem d'emprar la notació `.*` (punt i asterisc), tal com segueix

```
octave:4> s=sin(t);
octave:5> e=exp(-t);
octave:6> s*e
error: operator *: nonconformant arguments (op1 is 1x5, op2 is 1x5)
error: evaluating binary operator '*' near line 6, column 2
octave:6> s.*e
ans =
   0.000000   0.049126  -0.006461   0.000519  -0.000025
```

Aquí veieu que el primer intent de l'ordre #6 ha fracassat perquè les dimensions no concorden. El resultat final, és un vector format per $\sin(0) * e^{0}$, $\sin(2.5) * e^{-2.5}$, etc. fins $\sin(10) * e^{-10}$.

Una vegada obtingut el resultat i desat en una variable, ja podem representar-ho mitjançat l'ordre `plot`. L'exemple proposat a l'inici, representat no amb 5 punts sinó amb 201 punts, quedaria com segueix:

```
octave:1> t=linspace(0,2,201);
octave:2> y=4*exp(-5*t).*sin(2*pi*t+pi/8);
octave:3> plot(t,y)
```



Observeu que hem creat un eix de temps entre 0 i 2, tenint en compte la dinàmica esperada del circuit.

Una vegada tenim la gràfica, podem moure el cursor pel damunt i investigar, de forma aproximada, els valors dels màxims, mínims, passos per zero, etc. Clickant amb el botó central podem deixar caure punts concrets sobre la gràfica

Experimenteu també què passa amb l'ordre `grid`.