Συστήματα Αυτομάτου Ελέγχου

Προσομοίωση Συστημάτων

Χρήση της εργαλειοθήκης του Matlab

Control Systems Toolbox

Διδάσκοντες : **Αν Καθηγήτρια Ο. Κοσμίδου**
Καθηγητής Ι. Μπούταλης
Εργαστήριο Σ.Α.Ε. – Δ.Π.Θ.

Προσομοίωση δυναμικών συστημάτων

- Θα αναφερθούμε σύντομα στη χρήση του προγραμματιστικού περιβάλλοντος MATLAB για την προσομοίωση απλών και σύνθετων συστημάτων

Μαθηματικά μοντέλα

Διαφορική εξίσωση

$$y^{(n)}(t)+\alpha_1 y^{(n-1)}(t)+\ldots+\alpha_n y(t)=b_1 u^{(m)}(t)+b_2 u^{(m-1)}(t)+\ldots+b_{m+1}u(t)$$

Συνάρτηση Μεταφοράς

$$H(s) = \frac{L\{y(t)\}}{L\{u(t)\}} = \frac{y(s)}{u(s)} = \frac{b_1 s^m + b_2 s^m + \ldots + b_{m+1}}{s^n + \alpha_1 s^n + \ldots + \alpha_n}$$

Πολυώνυμα ως προς s

Κρουστική απόκριση

$$h(t) = L^{-1}\{H(s)\}$$

Χειρισμός Πολυωνύμων

$$p(s) = s^3 + 3s^2 + 4$$

Χρήση των συναρτήσεων **roots** και **poly**

```
>>p=[1 3 0 4];
>>r=roots(p)

r =
 -3.3553
  0.1777+ 1.0773i
  0.1777- 1.0773i
>>p=poly(r)
p =

  1.0000    3.0000    0.0000    4.0000
```

$p(s) = s^3 + 3s^2 + 4$

Υπολογισμός των ριζών του p(s) = 0.

Σύνθεση του πολυωνύμου από τις ρίζες του.

Χειρισμός Πολυωνύμων

$$n(s) = (3s^2 + 2s + 1)(s + 4) \quad \Longrightarrow \quad n(s) = 3s^3 + 14s^2 + 9s + 4$$

Πολλαπλασιασμός με χρήση της συνάρτησης **conv**

Υπολογισμός της τιμής πολυωνύμου με την **polyval**

```
>>p=[3 2 1]; q=[1 4];
>>n=conv(p,q)
n=
   3    14    9    4

>>value=polyval(n,-5)
value =

  -66
```

Πολλαπλασιασμός των p και q..

$n(s) = 3s^3 + 14s^2 + 9s + 4$

Υπολογισμός του n(s) στο σημείο s = –5.

Χειρισμός συναρτήσεων μεταφοράς

$$y^{(4)} + 10y^{(3)} + 30y^{(2)} + 40y^{(1)} + 24 = 4u^{(2)} + 36u^{(1)} + 32$$

$$H(s) = \frac{4s^2 + 36s + 32}{s^4 + 10s^3 + 30s^2 + 40s + 24}$$

Τα μοντέλα των Γ.Χ.Α αποτελούν αντικείμενα (**objects**), ώστε ο χρήστης να χειρίζεται τα διάφορα μοντέλα των συστημάτων ως απλές οντότητες

```
% Script e1 Create G(s) as a tf object
  clear % removes all variables from the workspace
  numG = [4 36 32]  % Create numerator
  denG = [1 10 30 40 24] % … & denominator polynomials
  G = tf(numG,denG) % create TF object
  pause
  get(G)  % See properties of TF object
  pause
% Extract numerator & denominator polynomials from TF
% object')
  [nn,dd] = tfdata(G,'v')
  pause
  pzmap(G);grid
  title('Ex 1: pole-zero map from TF object‘)
```


Χειρισμός συναρτήσεων μεταφοράς

$$G(s) = \frac{4s+3}{s^2+6s+5} = \frac{4(s+0.75)}{(s+1)(s+2)}$$

Μετατροπή από TF σε ZPK μορφή

```
% Script e2 Create G(s) as ZPK object
  clear % removes all variables from the workspace
  numG = [4 36 32]  % Create numerator
  denG = [1 10 30 40 24] % … & denominator polynomials
  G = tf(numG,denG) % create TF object
  pause
% Convert G(s) into ZPK object
  GG = zpk(G)
  pause
 [zz,pp,kk] = zpkdata(GG,'v') %Extract poles and zeros from ZPK form
  pause
  [z,p,k] = zpkdata(G,'v') %Extract poles and zeros from TF form
  pause
  pzmap(GG);grid
  title('Ex 2: pole-zero map from ZP object‘)
```

Ανάλυση σε μερικά κλάσματα - κρουστική απόκριση

$$G(s) = \frac{3s+2}{2s^3+4s^2+5s+1} = \frac{1.5(s+0.67)}{(s+0.24)(s+0.88+j1.41)(s+0.88-j1.41)}$$

$$G(s) = \frac{0.37}{(s+0.24)} + \frac{-0.19-j0.55}{(s+0.88-j1.41)} + \frac{-0.19+j0.55}{(s+0.88+j1.41)}$$

```
% Script e3 partial-fraction expansion and impulse response
  clear % removes all variables from the workspace
  numG = [3 2]  % Create numerator
  denG = [2 4 5 1] % … & denominator polynomials
  G = tf(numG,denG) % create TF object
  pause
 [zG,pG,kG] = zpkdata(G,'v') %Extract poles and zeros from TF form
  pause
 [resG,polG,otherG] = residue(numG,denG) %Do partial-fraction
 % expansion to get residues at each pole
  pause
  impulse(G); grid
  %title('Ex 3 - impulse response');
```


Χρονική απόκριση μεμονωμένων πόλων

$$G(s) = \frac{k}{(s-p)} \rightarrow g(t) = ke^{pt}$$

$$G(s) = \frac{ke^{j\varphi}}{(s-(\sigma+j\omega))} + \frac{ke^{-j\varphi}}{(s-(\sigma-j\omega))} \rightarrow g(t) = 2ke^{\sigma t}\cos(\omega t+\varphi)$$

```
% Script e4 Responses due to distinct poles
  clear % removes all variables from the workspace
  numG = [3 2]  % Create numerator
  denG = [2 4 5 1] % … & denominator polynomials
  G = tf(numG,denG) % create TF object
  pause
  [resG,polG,otherG] = residue(numG,denG) %Do partial-fraction
 % expansion to get residues at each pole)
  pause
  t = [0:0.1:20];
  ycmplx = cpole2t(polG(1),resG(1),t); %response due to complex poles
  yreal = rpole2t(polG(3),resG(3),t); %response due to real pole
  ytot = ycmplx + yreal;
  plot(t,ytot,t,ycmplx,t,yreal,'--');grid
  title('Ex 4: responses: total, from complex & real poles')
  legend('total','complex poles','real pole‘)
```

Χρονική απόκριση σε βηματική είσοδο

Υπάρχουν δύο τρόποι. Στον πρώτο, η συν. Μεταφοράς τροποποιείται ώστε να ενσωματώσει το πόλο της μοναδιαίας βηματικής συνάρτησης. Στη συνέχεια υπολογίζεται η κρουστική απόκριση της νέα συν. μεταφοράς

$$G'(s) = \frac{1}{s}G(s) = \frac{3s+2}{s(2s^3+4s^2+5s+1)}$$

Στο δεύτερο τρόπο, χρησιμοποιείται η συνάρτηση **step** πάνω στην αρχική συνάρτηση μεταφοράς

```
% Script e5 Step response from G(s)
  clear % removes all variables from the workspace
  numG = [3 2]  % Create numerator
  denG = [2 4 5 1] % … & denominator polynomials
% Add pole at s=0 to G(s) to get transfer function of step response
  numstep = numG
  denstep = [denG 0]
  pause
  Gstep = tf(numstep,denstep) % create G(s)/s
  impulse(Gstep,20) % produce impulse response
  pause
  G = tf(numG,denG) % create the G(s) of initial system
  step(G,20)
  DC_gain = dcgain(G) % calculate DC gain – g(0)
```


Χρονική απόκριση σε γενική είσοδο

Χρησιμοποιείται η συνάρτηση **lsim** αφού πρώτα δημιουργήσουμε διάνυσμα με δείγματα της εισόδου σε συγκεκριμένο χρονικό διάστημα

$$G(s) = \frac{3s+2}{2s^3+4s^2+5s+1} \qquad u(t) = \begin{cases} 0 & t<0 \\ 2 & 0 \le t < 3 \\ 0.7 & t \ge 3 \end{cases}$$

```
% Script e6 Time response to piecewise constant input
  clear % removes all variables from the workspace
  G = tf([3 2],[2 4 5 1]) % create G(s)
  pause
  time = [0:0.02:10]'; % create vector with 501 time samples
  u = 2.0*(1+0*(time)); % vector of 2's with length of "time"
  for ii=min(find(time>=3.0)):length(u)
  u(ii) = 0.7;
  end
  y = lsim(G,u,time); % Compute response
  plot(time,y,time,u,'--');grid
  legend('y(t)','u(t)‘)
```

Πόλοι – μηδενικά και χρονική απόκριση

$$G(s) = \frac{(s+5)(s^2+1)}{(s+1)(s+2)^2(s+3)}$$

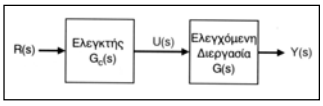
Οι πόλοι καθορίζουν την ευστάθεια, τα μηδενικά επηρεάζουν το πλάτος της χρονικής *απόκρισης στις διάφορες συχνότητες λειτουργίας*

```
% Script e7 Time response to piecewise constant input
  clear % removes all variables from the workspace
  zz = [-5; i; -i]      % generate G(s) in ZPK form
  pp = [-1; -2; -2; -3]
  kk = 1
  G = zpk(zz,pp,kk)
  pause
  t = [0:0.1:10]; % generate time vector
  impulse(G,t) % impulse response of G
  pause
  [numG,denG] = tfdata(G,'v');
  G1 = tf([1 5],denG); % remove complex zeros from numG
  impulse(G1,t)
  grid, title('impulse response with zeros factored out')
  pause
  e = sin(t); % input signal with a frequency of 1 rad/s
  y = lsim(G,e,t); % simulate the time response of G with input e
  plot(t, y), grid, title('time response to sinusoidal input')
  xlabel('time'), ylabel('amplitude')
```

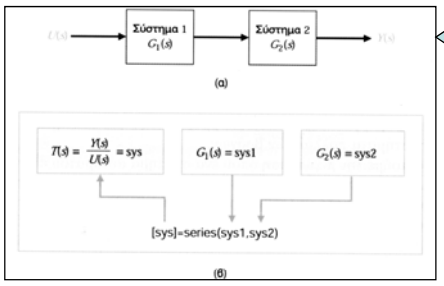
$$L\{\sin(t)\} = \frac{1}{s^2+1}$$

Δημιουργία και χειρισμός σύνθετων συστημάτων

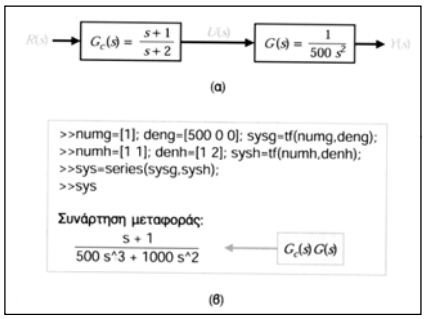
Σύνδεση συστημάτων σε σειρά



Είναι σύνηθες να συνδέουμε δύο ή περισσότερα συστήματα εν σειρά.(π.χ. τον ελεγκτή και το υπό έλεγχο σύστημα)

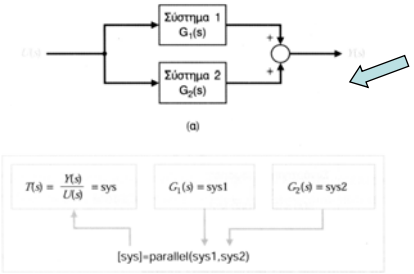


Στο MATLAB η σειριακή σύνδεση επιτυγχάνεται με τη χρήση της συνάρτησης **series**

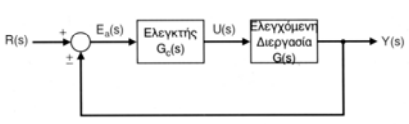


Εδώ βλέπουμε την εφαρμογή σε ένα παράδειγμα. Είναι γνωστό ότι το σύνθετο σύστημα έχει συνάρτηση μεταφοράς ίση με το γινόμενο των δύο συναρτήσεων μεταφοράς των δύο συστημάτων

Παράλληλη σύνδεση και Χρήση Ανάδρασης



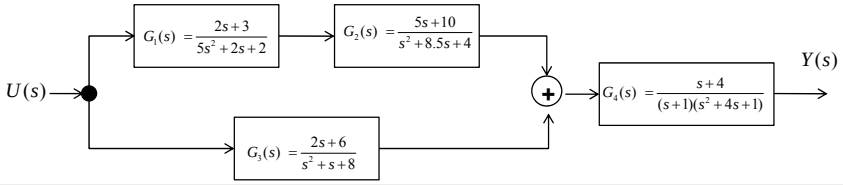
Στα λειτουργικά διαγράμματα εμφανίζονται συχνά συναρτήσεις μεταφοράς των οποίων οι αντίστοιχες μονάδες συνδέονται **παράλληλα**. Στις περιπτώσεις αυτές χρησιμοποιείται η συνάρτηση ***parallel***. Η χρήση της συνάρτησης parallel, περιγράφεται στο σχήμα. Η σύνθετη συνάρτηση μεταφοράς προκύπτει από το άθροισμα των δύο επιμέρους συναρτήσεων



Μπορούμε επίσης να εισάγουμε ένα σήμα ανάδρασης στο σύστημα ελέγχου, κλείνοντας τον βρόχο με την λεγόμενη μοναδιαία ανάδραση *(unity feedback)*, όπως φαίνεται στο σχήμα. Το σήμα $E_a(s)$ είναι το λεγόμενο σήμα σφάλματος *(error signal)*. Το σήμα *R(s)* αντιστοιχεί στην είσοδο αναφοράς *(reference input)*. Στο συγκεκριμένο σύστημα, ο ελεγκτής βρίσκεται στον ευθύ κλάδο και η συνάρτηση μεταφοράς κλειστού βρόχου είναι,

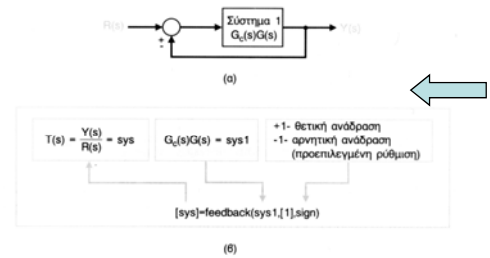
$$T(s) = \frac{G_c(s)G(s)}{1 \mp G_c(s)G(s)}$$

Παράλληλη – Σειριακή σύνδεση

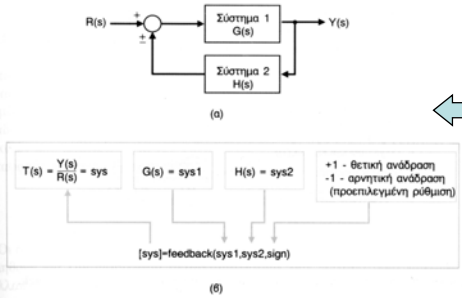



```
% Script e8 Series/parallel multi block system
  clear % removes all variables from the workspace
  G1 = tf([2 3],[5 2 2])
  G2 = tf([5 10],[1 8.5 4])
  G3 = tf([2 6],[1 1 8])
  pause
  G4 = tf([1 4],conv([1 1],[1 4 1]))
  pause
  T = G4*(G3 + G2*G1) % transfer function of the entire system
%  T = series(G4,parallel(G3, series(G2,G1))) % Alternative computation
  pause
  Tzpk = zpk(T) % Entire system in zpk form
  pause
  Tpoles = pole(T), Tzeros = zero(T)  % get the poles and zeros of T
  pause
  TgainDC = dcgain(T), step(T) % compute the DC gain and the step response
```


Ανάδραση με τη συνάρτηση Feedback

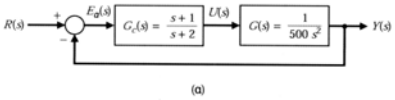


Συχνά συναντούμε την περίπτωση όπου ένα σύστημα κλειστού βρόχου περιλαμβάνει έναν κλάδο μοναδιαίας ανάδρασης, όπως φαίνεται στο σχήμα. Στις περιπτώσεις αυτές, μπορούμε να χρησιμοποιήσουμε την συνάρτηση ***feedback*** για τον υπολογισμό της αντίστοιχης συνάρτησης μεταφοράς κλειστού βρόχου, θέτοντας *H(s)=1.*



Στο διπλανό σχήμα φαίνεται η συνάρτηση feedback μαζί με την αντίστοιχη σύνθεση του συστήματος, όπου στον κλάδο ανάδρασης περιλαμβάνεται η συνάρτησηH(s). Αν παραλείψουμε την παράμετρο εισόδου "sign", τότε θεωρείται ότι έχουμε αρνητική ανάδραση

Παράδειγμα με τη συνάρτηση Feedback



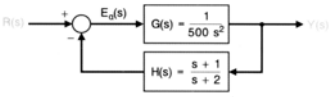
(α)

```
>>numg=[1]; deng=[500 0 0]; sys1=tf(numg,deng);
>>numc=[1 1]; denc=[1 2]; sys2=tf(numc,denc);
>>sys3=series(sys1,sys2);
>>sys=feedback(sys3,[1])
```

Transfer function:

$$\frac{s+1}{500\,s^3+1000\,s^2+s+1} \qquad \frac{Y(s)}{R(s)}=\frac{G_c(s)G(s)}{1+G_c(s)G(s)}$$

(β)



(α)

```
>>numg=[1]; deng=[500 0 0]; sys1=tf(numg,deng);
>>numh=[1 1]; denh=[1 2]; sys2=tf(numh,denh);
>>sys=feedback(sys1,sys2);
>>sys
```

Συνάρτηση μεταφοράς:

$$\frac{s+2}{500\,s^3+1000\,s^2+s+1} \qquad \frac{Y(s)}{R(s)}=\frac{G(s)}{1+G(s)H(s)}$$

(β)

Η Συνάρτηση Feedback σε Σύστημα με Κλάδο Μοναδιαίας Ανάδρασης

Έστω η διεργασία και ο αντίστοιχος ελεγκτής που παριστάνονται από τις συναρτήσεις μεταφοράς $G_1(s)$ και $G_2(s)$ αντίστοιχα, όπως δίνονται στο διπλανό σχήμα. Για να χρησιμοποιήσουμε την συνάρτηση *feedback* θα πρέπει πρώτα να εκτελέσουμε την συνάρτηση ***series***, ώστε να προκύψει το σύστημα $G_1(s)G_2(s)$ και στη συνέχεια εκτελούμε την συνάρτηση feedback για να κλείσουμε τον βρόχο. Η ακολουθία των εντολών που απαιτείται για τον σκοπό αυτό δίνεται στο σχήμα

Μια άλλη βασική δομή ενός συστήματος αυτομάτου ελέγχου, φαίνεται παρακάτω.

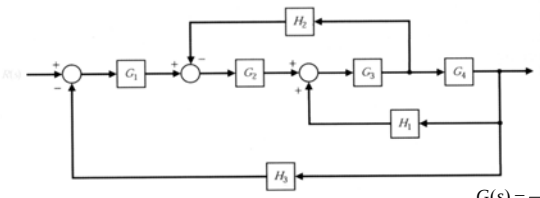
$$T(s)=\frac{G(s)}{1\mp G(s)H(s)}$$

Στην περίπτωση αυτή ή μονάδα του ελεγκτή τοποθετείται στον κλάδο της ανάδρασης.

Για τον υπολογισμό της συνάρτησης μεταφοράς κλειστού βρόχου με την μονάδα του ελεγκτή στον κλάδο της ανάδρασης, χρησιμοποιούμε την συνάρτηση feedback.

Απλοποίηση λειτουργικών διαγραμμάτων

Οι συναρτήσεις *series, parallel* και *feedback*, μπορούν να χρησιμοποιηθούν για την απλοποίηση σύνθετων λειτουργικών διαγραμμάτων. Θέλουμε να υπολογίσουμε την αντίστοιχη συνάρτηση μεταφοράς κλειστού βρόγχου του παρακάτω συστήματος.



$$G_1(s)=\frac{1}{s+10} \qquad G_2(s)=\frac{1}{s+1}$$

$$G_3(s)=\frac{s^2+1}{s^2+4s+4} \qquad G_4(s)=\frac{s+1}{s+6}$$

$$H_1(s)=\frac{s+1}{s+2} \qquad H_2(s)=2 \qquad H_3(s)=1$$

$$G(s)=\frac{s^5+4s^4+6s^3+6s^2+5s+2}{12s^6+205s^5+1066s^4+2517s^3+3128s^2+2196s+712}$$

Ακολουθούμε μια διαδικασία πέντε βημάτων:

1. Εισάγουμε τις συναρτήσεις μεταφοράς στο περιβάλλον του MATLAB.
2. Μετακινούμε την μονάδα H_2 πίσω από την μονάδα G_4.
3. Απαλείφουμε τον βρόχο $G_3G_4H_1$
4. Απαλείφουμε τον βρόχο που περιλαμβάνει την μονάδα H_2
5. Απαλείφουμε τον βρόχο που απομένει και υπολογίζουμε την $T(s)$.

```
% Script e9 Complex multi block system
Clear
G1 = tf([1],[1 10]); G2 = tf([1],[1 1]);
G3 = tf([1 0 1],[1 4 4]); G4 = tf([1 1],[1 6]);
H1 = tf([1 1],[1 2]);   H2 = tf([2],[1]);
H3 = tf([1],[1]); % up to here STEP 1
sys1 = H2/G4;  % STEP 2
sys2 = series(G3, G4);
sys3 = feedback(sys2, H1,+1);  % STEP 3
sys4 = series(G2, sys3);
sys5 = feedback(sys4, sys1); % STEP 4
sys6= series(G1, sys5);
G = feedback(sys6, H3) % STEP 5
```

Περισσότερες δυνατότητες – Χρήση βοήθειας

```
>> help control

  Control System Toolbox.
  Version 5.2 (R13)  28-Jun-2002

  General.
    ctrlpref    - Set Control System Toolbox preferences.
    ltimodels   - Detailed help on the various types of LTI
models.
    ltiprops    - Detailed help on available LTI model properties.

  Creating linear models.
    tf          - Create transfer function models.
    zpk         - Create zero/pole/gain models.
    ss, dss     - Create state-space models.
    frd         - Create a frequency response data models.
    filt        - Specify a digital filter.
    set         - Set/modify properties of LTI models.

  Data extraction.
    tfdata      - Extract numerator(s) and denominator(s).
```

… κλπ

Μπορούμε να δούμε περισσότερα για τις δυνατότητες της εργαλειοθήκης ΣΑΕ με τη χρήση της εντολής help από την γραμμή εντολών ή ανατρέχοντας στα εγχειρίδια χρήσης

```
>> help tf

 TF  Creation of transfer functions or
conversion to transfer function.

   Creation:
     SYS = TF(NUM,DEN) creates a
continuous-time transfer function SYS with
     numerator(s) NUM and denominator(s)
DEN.  The output SYS is a TF object.

     SYS = TF(NUM,DEN,TS) creates a
discrete-time transfer function with
     sample time TS (set TS=-1 if the sample
time is undetermined).
```

… κλπ

Βιβλιογραφία

1. R. Dorf and R. Bishop, “Σύγχρονα Συστήματα Αυτομάτου Ελέγχου”, Εκδόσεις Τζιόλα (11η Έκδοση στα Ελληνικά), 2009.
2. Dean Frederick and Joe Chow, “Feedback Control Problems using Matlab”, Brooks/Cole publishing, 2000.
3. B. Lurie and P. Enright, “Classical Feedback Control with Matlab”, Marcel Dekkerm, 2000.