1^o Εργαστήριο στα Συστήματα Ελέγχου

Ευάγγελος Κατσούπης 2017030077
Απόστολος Γιουμερτάκης 2017030142
Α. Ραφαήλ Ελληνιτάκης 2017030118
Κωνσταντίνος Βούλγαρης 2017030125

Ομάδα 37

20 Μαρτίου 2021

Κεφάλαιο 1

Α΄ Μέρος Άσκησης

1.1 Matlab και PID Control Systems.

Με τις συναρτήσεις και τα εργαλεία που μας προσφέρει η Matlab, μπορούμε εύκολα να μοντελοποιήσουμε και να μικρορυθμίσουμε τα συστήματα ελέγχου μας μέσω γραφικών απεικονίσεων και έτοιμων εργαλείων. Μερικά από αυτά είναι το PID Tuner, που χρησιμοποιήσαμε για να τελειοποιήσουμε τα βάρη των συστημάτων ελέγχου μας, η συνάρτηση **feedback()**που ορίζει τον βρόγχο ανάδρασης μεταξύ εξόδου και εισόδου του κυκλώματος μας, της **pidstd()**που μας βοηθάει να ορίσουμε ένα control systemμε τις μεταβλητές που του δίνονται, την **impulse()**, την **step()**κ.α.

Τα παραπάνω αποτελούν απαραίτητο εργαλείο για την σχεδίαση και τελειοποίηση ενός control system πριν χρησιμοποιηθεί στην πράξη.

1.2 Κώδικες Matlab

```
time = 0:0.01:3.5;
% a1.Trasfer Functions
% H1
num = 1;
den = [0.3 1];
H1 = tf(num, den)
rH1 = lsim(H1,time,time);

H2 = H1^2
rH2 = lsim(H2,time,time);
```

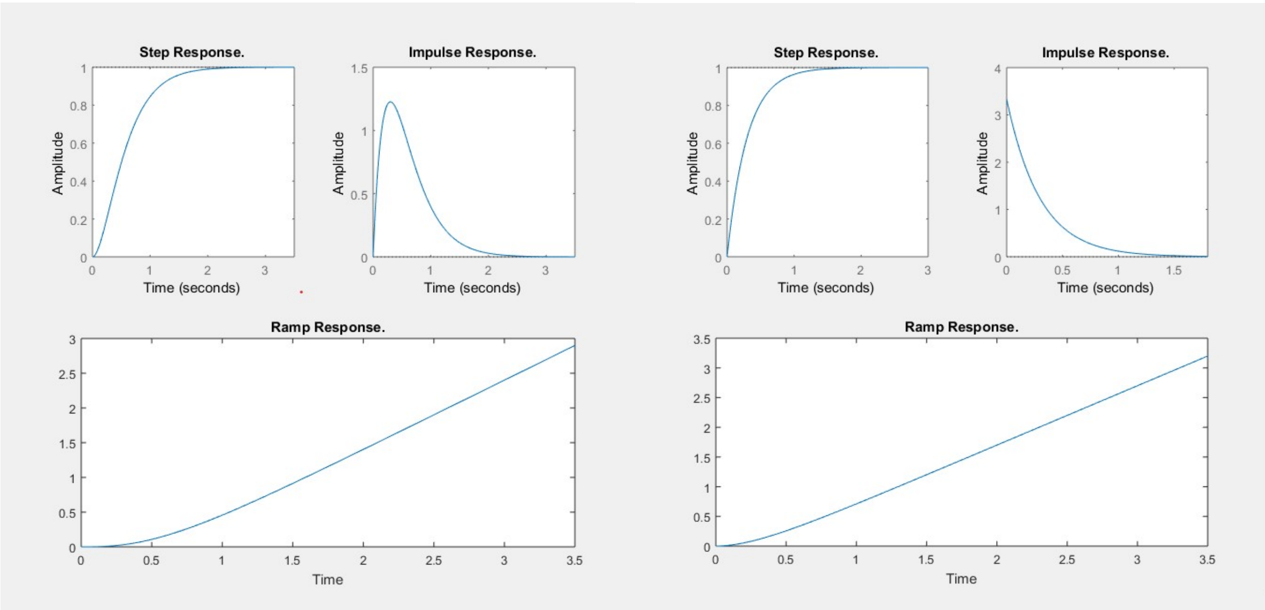
Σχήμα 1.1: Ο ορισμός των συναρτήσεων μεταφοράς και των αποκρίσεων τους.

Για να ορίσουμε την συνάρτηση μεταφοράς που μας δίνεται από την εκφώνηση, και σύμφωνα με την εικόνα 1.1, χρησιμοποιούμε την μέθοδο **tf()** της Matlab και της δίνουμε ώς όρισμα πίνακες για τον αριθμητή και τον παρονομαστή, που αντιστοιχούν σε κάποια τάξη πολυωνύμου και τους συντελεστές της. Παρατηρούμε ότι η δεύτερη συνάρτηση μεταφοράς είναι η πρώτη αλλά υψωμένη στο τετράγωνο, οπότε δεν την ορίζουμε από την αρχή.

Χρησιμοποιούμε το **lsim()** για να μπορέσουμε αργότερα να προβάλλουμε την γραφική παράσταση της απόκρισης ράμπας του κάθε συστήματος, όπως φαίνεται στην εικόνα 1.2, και έπειτα με την χρήση της subplot() ορίζουμε τα παράθυρα με τις γραφικές παραστάσεις, και μέσω της **step()** παίρνουμε την **βηματική απόκριση**. Έπειτα, για την **κρουστική απόκριση**, δίνουμε την συνάρτηση μεταφοράς μας στην μέθοδο **impulse()** η οποία μας προβάλλει την γραφική απεικόνιση της κρουστικής απόκρισης στο αναδυόμενο παράθυρο. Τέλος, για την **απόκριση ράμπας** χρησιμοποιήσαμε πιο πάνω όπως αναφέραμε την **lsim()**, το αποτέλεσμα της οποίας προβάλλουμε στο παράθυρο γραφικών. Η ακριβώς αντίστοιχη διαδικασία ακολουθήθηκε και για την δεύτερη συνάρτηση μεταφοράς.

```
figure('Name','Tranfer function H1');
subplot(2,2,1);
step(H1);          % The step response
title('Step Response.')
xlabel('Time');  % x axis label
subplot(2,2,2);
impulse(H1);       % The impulse response
title('Impulse Response.')
xlabel('Time');
hold on;
subplot(2,2,[3,4]);
plot(time,rH1);  % ramp response
title('Ramp Response.')
xlabel('Time');
```

Σχήμα 1.2: Δημιουργία παραθύρου προβολής της βηματικής απόκρισης του H1.



(α′) Η κρουστική, η βηματική και η απόκριση ράμπας του H2.

(β′) Η κρουστική, η βηματική και η απόκριση ράμπας του H1.

Σχήμα 1.3

1.3 Γραφικές Παραστάσεις

Για τις γραφικές παραστάσεις χρησιμοποιήθηκε αντίστοιχη διαδικασία με την προαναφερθείσα στην ενότητα 1.2, με χρήση της **tf()** για ορισμό των συναρτήσεων μεταφοράς, της **subplot()**για άνοιγμα παραθύρων πολλαπλών γραφικών και της **step()** για αναπαράσταση της βηματικής απόκρισης. Ονομάσαμε τις συναρτήσεις μεταφοράς με διαδοχικά ονόματα **h1,h1,h3...h12**για εύκολη αναγνώριση και ανάγνωση. Για απλούστευση της διαδικασίας, όπως φαίνεται στην εικόνα 1.4 για να κατασκευάσουμε τον σύνθετο παρονομαστή, ορίζουμε δύο διαφορετικές συναρτήσεις και υψώνουμε και πολλαπλασιάζουμε αντίστοιχα, όπως παρακάτω:

$$\frac{1}{0.5 * s + 1} \frac{1}{(0.1 * s + 1)^3} = \frac{1}{(0.5s + 1)(0.1s + 1)^3}$$

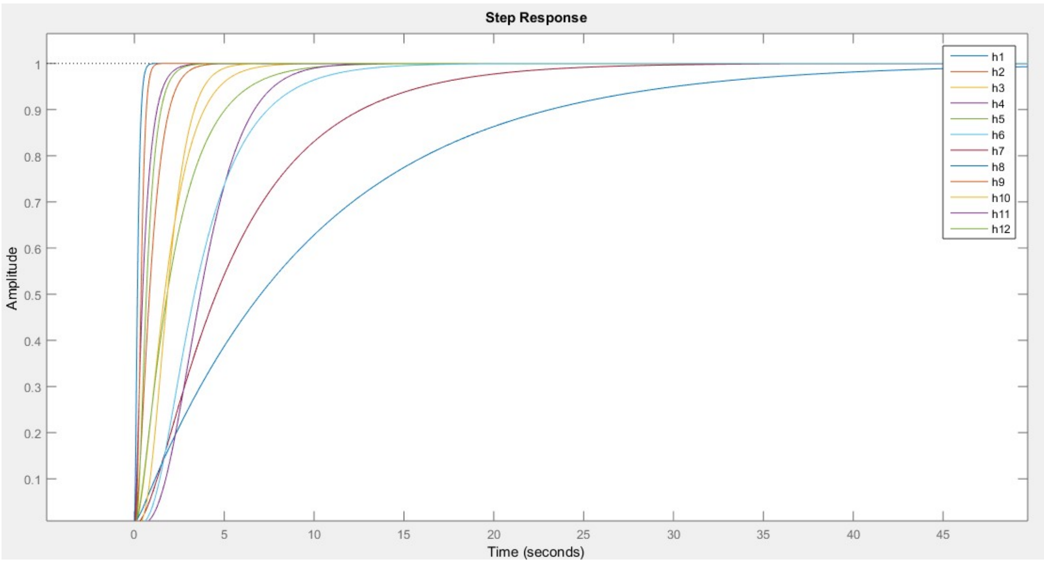
Παρατηρούμε στην γραφική αναπαράσταση όλων των συναρτήσεων μεταφοράς, στο σχήμα 1.6, τις βηματικές αποκρίσεις τους συγκριτικά μεταξύ τους. Επίσης οι τιμές των Ks, Tu και Tg βρίσκονται στον πίνακα 1.1.

```
den = [0.1 1];
den1 = [0.5 1];
h11 = tf(num,den)^3 * tf(num,den1);
```

Σχήμα 1.4: Ορισμός της συνάρτησης μεταφοράς H11.

```
subplot(4,3,11);
step(h11);
legend('h11');
hold on;
```

Σχήμα 1.5: Γραφική αναπαράσταση της συνάρτησης μεταφοράς του H11.



Σχήμα 1.6: Γραφική απεικόνιση της βηματικής απόκρισης των συναρτήσεων h1-h12.

Ks	Tu	Tg
0.99987	0	0.31776
0.99952	0.096709	0.81512
0.99952	0.032236	0.27171
0.99709	0.15	1.36
0.99787	0.29	2.72
0.99783	0.055262	0.74604
0.99783	0.22105	2.9842
0.99783	0.55262	7.4604
0.99987	0.46052	10.592
0.99991	0.15	0.44
0.99991	0.72	2.23
0.99894	1.43	4.46
0.99852	0.21184	0.9026
0.99852	1.0592	4.5131

Table 1.1: Οι τιμές των Ks, Tu και Tg όλων των συναρτήσεων μεταφοράς του πρώτου ερωτήματος με την συνάρτηση paramCalc().

Κεφάλαιο 2

Β' Μέρος Άσκησης

2.1 Υπολογισμός Συνάρτησης Μεταφοράς

Για να ξεκινήσουμε την υλοποίηση του μοντέλου ελεγκτή, ορίζουμε αρχικά το σύστημα ανοικτού βρόγχου τρίτης τάξεως που μας δίνεται στην περιγραφή της άσκησης, με όρους:

$$\mathbf{Ks{=}1.0}, T_1\mathbf{=2.0}, T_2\mathbf{=2.0}, T_3\mathbf{=2.0}.$$

Απο τον ορισμό των συναρτήσεων μεταφοράς στην Matlab, προκύπτει η συνάρτηση μεταφοράς:

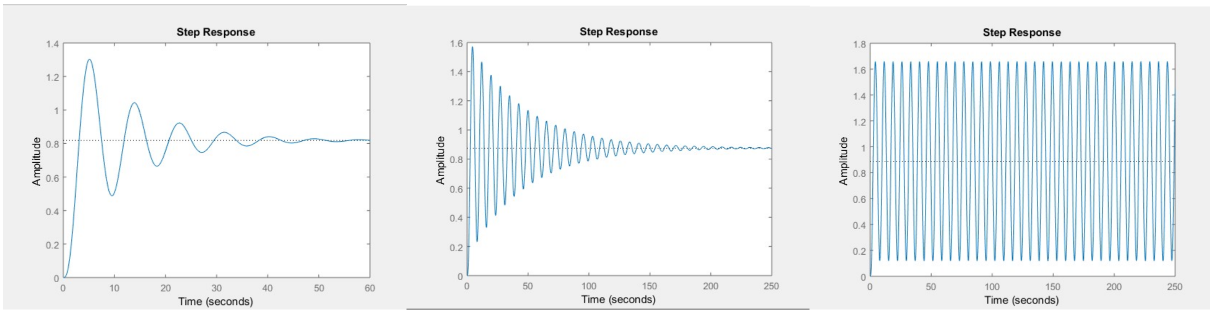
$$H(s) = \frac{1}{8s^3 + 12s^2 + 6s + 1}$$

2.2 Ziegler-Nichols PID System

Χρησιμοποιώντας έναν βρόγχο **for** με βήμα 0.5 από το 1 έως το 20 όπως φαίνεται στην εικόνα 2.1, ορίσαμε έναν controller P και και μία μοναδιαία ανάδραση **feedback(sys*cont,1)** υπολογίσαμε την γραφική παράσταση της μοναδιαίας ανάδρασης με την συνάρτηση **stepinfo()**. Ελέγχοντας σε κάθε loop εάν το **peakTime == Inf** , μπορούμε να βρούμε ακριβώς σε ποιο Kp έχουμε μόνιμες ταλαντώσεις του συστήματος, διακόπτοντας τον βρόγχο με την εντολή break και κρατώντας την τιμή του Kcrit, όπου πειραματικά υπολογίστηκε στην τιμή **8**. Επίσης, μετρώντας την απόσταση μεταξύ δύο κορυφών στην προαναφερθείσα ταλάντωση με το **findpeaks()**και έπειτα χρησιμοποιώντας το **max(diff())**για τον προσδιορισμό μίας μοναδικής τιμής Tcrit, η οποία βγαίνει πειραματικά στην τιμή **7.306870**.

```
for Kp = 1:0.5:20
    C = pid(Kp);  % define a p cnotroller
    T = feedback(C*sys1,1);  % connect it to our plant, sys1
    plantInfo = stepinfo(T);
    figure()
    step(T)
    if plantInfo.PeakTime == Inf;  % we check if T oscillates at the same
        fprintf('The Kp_Crit value is %f\n', Kp);  % amplitude for infinity
        Kp_crit = Kp;
        break  % Kp_critical is found!
    end
end
```

Σχήμα 2.1: Ο βρόγχος for για την εύρεση του Kcrit.



(α′) Απόκριση με μικρό Kp (Kp=4) (β′) Απόκριση με Kp λίγο μικρότερο του Kcrit(Kp=6.5) (γ′) Απόκριση με με μόνιμη ταλάντωση σε Kcrit=8

Σχήμα 2.2

Ελεγκτής	**K**	**T_i**	**T_d**
P	**0.5 $K_{p,crit}$**		
PI	**0.45 $K_{p,crit}$**	**0.85 T_{crit}**	
PID	**0.6 $K_{p,crit}$**	**0.5 T_{crit}**	**0.12 T_{crit}**

Σχήμα 2.3: Ο πίνακας ρύθμισης ελεγκτών Ziegler/Nichols.

Όπως φαίνεται και στο σχέδιο 2.5, οι τρεις τύποι ελεγκτών παρουσιάζουν σημαντικές ιδιομορφίες:

- Αρχικά, ο **ελεγκτής τύπου P** με χρόνο ανύψωσης 1.93 δευτερόλεπτα βλέπουμε ότι δεν κάνει σημαντικό overshoot, και έχει σταθεροποιηθεί αρκετά μέχρι τα 40 δευτερόλεπτα, αλλά παρουσιάζει σφάλμα μόνιμης κατάστασης 0.2 στην περίπτωση μας.
- Έπειτα, ο **ελεγκτής τύπου PI** με χρόνο ανύψωσης 2.2 δευτερόλεπτα, κάνει αισθητό overshoot +0.5 μονάδες, ενώ ταλαντώνεται μέχρι τα 60 δευτερόλεπτα σημαντικά.
- Τέλος, ο **ελεγκτής PID** , κάνει επίσης overshoot κατά 0.4 μονάδες, αλλά σταθεροποιείται σχετικά γρήγορα και έχει αρκετά μικρό χρόνο ανύψωσης 1.99 δευτερόλεπτα.

Για τις μετρήσεις των χρόνων ανύψωσής βλέπουμε το σχήμα σχεδιάστηκαν οι εφαπτόμενες ευθείες στο σημείο καμπής των γραφικών παραστάσεων, με την χρήση των συναρτήσεων gradient(), mean() και plot().

Παρατηρούμε ότι με την μέθοδο Ziegler/Nichols έχουμε ταλαντώσεις για μεγάλο χρονικό διάστημα. Αυτό μπορεί να αποφευχθεί με την χρήση του Control Toolbox της Matlab, καθώς μας δίνει την δυνατότητα να ελέγξουμε το κέρδος των P και ID χειροκίνητα και να πάρουμε τις τιμές που επιθυμούμε ώστε η συμπεριφορά του συστήματος μας να πλησιάζει την ιδανική.

```
Kp = 0.5*Kp_crit;
PID_controller = pidstd(Kp);
p_step = feedback(sys1*PID_controller,1);
```

(α′) Υλοποίηση P.

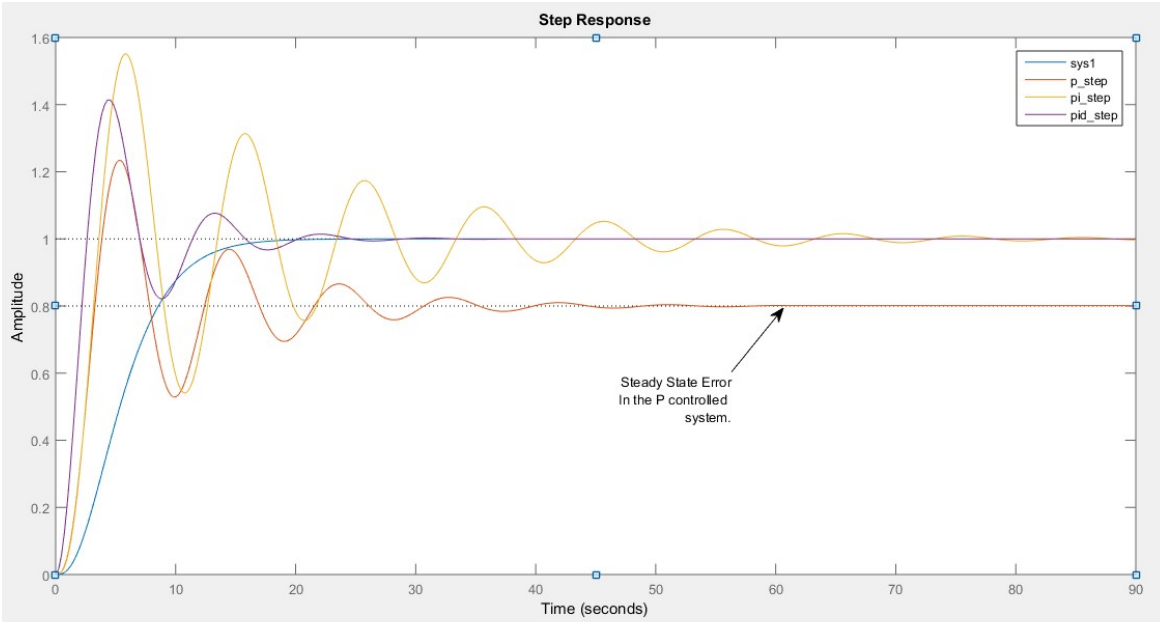
```
Kp = 0.45*Kp_crit;
Ti = 0.85*T_crit;
PID_controller = pidstd(Kp,Ti);
pi_step = feedback(sys1*PID_controller,1);
```

(β′) Υλοποίηση PI.

```
Kp = 0.6*Kp_crit;
Ti = 0.5*T_crit;
Td = 0.12*T_crit;
PID_controller = pidstd(Kp,Ti,Td);
pid_step = feedback(sys1*PID_controller,1);
```

(γ′) Υλοποίηση PID.

Σχήμα 2.4



Σχήμα 2.5: Η τελική απόκριση του των συστημάτων με P, Pi και PID ελεγκτές.

Για να βρούμε την συνάρτηση μεταφοράς του κάθε ελεγκτή, χρησιμοποιήσαμε την μέθοδο **tf()** που μπορεί να χρησιμοποιηθεί για να μετατρέψει ένα δυναμικό σύστημα σε συνάρτηση μεταφοράς, σύμφωνα με το επίσημο documentation της Matlab.

Παρακάτω παρατίθενται οι συναρτήσεις μεταφοράς του κάθε ελεγκτή:

- Το σύστημα P:
$$\text{P=4 (static-gain)}$$
- Το σύστημα PI:
$$\text{PI=}\frac{3.6s + 0.5796}{s} \text{ (continuous time t.f.)}$$
- Το σύστημα PID:
$$\text{PID=}\frac{4.209s^2 + 4.8s + 1.314}{s} \text{ (continuous time t.f.)}$$

Είναι εμφανές ότι το σύστημα P έχει σταθερό κέρδος, ενώ το σύστημα PI περιγράφεται από πρωτοβάθμια, και το PID από δευτεροβάθμια συνάρτηση μεταφοράς.

Μέθοδος CHR

Όμοια με τα προηγούμενα ερωτήματα κατασκευάζουμε την συνάρτηση μεταφοράς σύμφωνα με τα ορίσματα

$$K_s = 1.0, T_1 = 2.0, T_2 = 2.0 \text{ και } T_3 = 2.0.$$

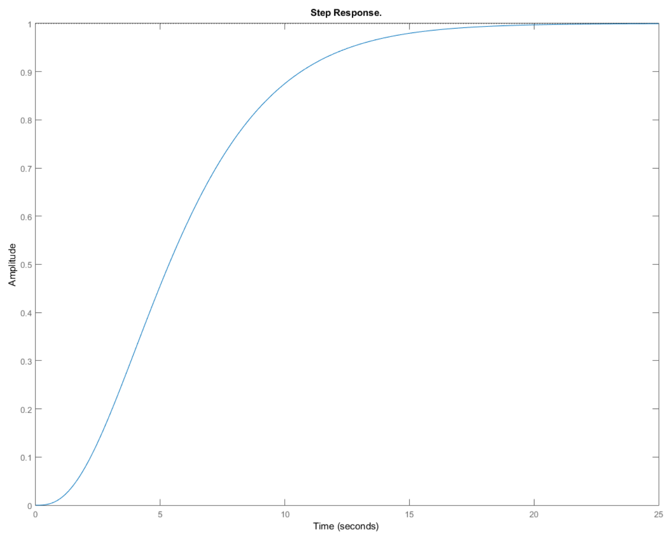
Η συνάρτηση μεταφοράς μας θα έχει όμοια την εξής μορφή:

$$H(s) = K_s \cdot \frac{1}{T_1 s + 1} \cdot \frac{1}{T_2 s + 1} \cdot \frac{1}{T_3 s + 1}$$

Και θα είναι η εξής:

$$H(s) = \frac{1}{8s^3 + 12s^2 + 6s + 1}$$

Για τον υπολογισμό της συνάρτησης αρχικά υπολογίσαμε τις επιμέρους συναρτήσεις μεταφοράς και έπειτα δημιουργήσαμε με την $H(s)$.
Η βηματική απόκριση του συστήματος, χωρίς ανάδραση είναι η εξής:



Σχήμα 1: Βηματική απόκριση συνάρτησης μεταφοράς sys1(s)

Κώδικάς για τον υπολογισμό της:

Για τον υπολογισμό των παραμέτρων K_s, T_u και T_g έγινε η δημιουργία και η χρήση της συνάρτησης paramCalc() η οποία παίρνει σαν όρισμα μια συνάρτηση μεταφοράς και επιστρέφει το πλάτος K_s, το χρόνο αργής μεταβολής T_u και το χρόνο γρήγορης μεταβολής T_g.

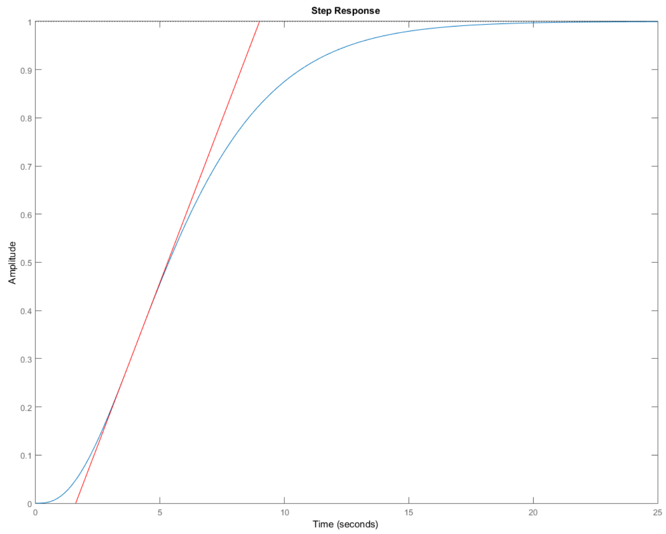
Listing 1: Code of paramCalc

```
function [K, Te, Tb] = paramCalc(SYS)

    [Y,time] = step(SYS);

    %Max amplitude
    K = Y(end);

    %find inflection point using 2nd derivative => '0'
    D = diff(Y)./diff(time);
    inflex = find(diff(D)./diff(time(1:end-1))<0,1);
    A = D(inflex)*time(inflex)-Y(inflex);

    %plots
    tangent = D(inflex)*time - A;
    figure('Name','Transfer function ')
    step(SYS)
    hold on
    plot(time,tangent,'r')

    %find intersection points with y=0 and y=1
    index_0 =  find(tangent>=0, 1);
    index_1 =  find(tangent>=1, 1);
    Te = time(index_0);
    Tb = time(index_1) - Te;
    fprintf('K: %.3f \nTu: %.3f s \nTg: %.3f s\n', K, Te, Tb);
end
```

Ουσιαστικά βρίσκουμε την εφαπτομένη της H(s), υπολογίζοντας την 2^η παράγωγο της συνάρτησης. Έπειτα υπολογίζουμε την μέγιστη και ελάχιστη τιμή της, σύμφωνα με την $H(s)$ και υπολογίζουμε τους χρόνους γρήγορης και αργής μεταβολής.

K_s	0.999
T_u	1.620 s
T_g	7.390 s

Πειραματικά υπολογισμένες σταθερές χρόνου της κρουστικής απόκρισης.



Σχήμα 2: Βηματική απόκριση συνάρτησης μεταφοράς sys1(s) και υπολογισμός χρονικών παραμέτρων

Στην συνέχεια για τις τιμές που δίνονται, $K_s = 1, \quad T_e = 1.7, \quad T_b = 6.7$ και με με βάση τον πίνακα:

Ελεγκτης	K	T_i	T_d
P	$\frac{0.3}{K}\frac{T_b}{T_e}$	·	·
PI	$\frac{0.35}{K}\frac{T_b}{T_e}$	$1.2T_b$	·
PID	$\frac{0.6}{K}\frac{T_b}{T_e}$	T_b	$0.5T_e$

υπολογίζουμε τους συντελεστές των ελεγκτών, για overload 0% για set-point response. Σε κάθε περίπτωση και έχουμε:

0% Overload			
Ελεγκτης	K	T_i	T_d
P	1.18424	·	·
PI	1.3794	8.04	·
PID	2.3647	6.7	0.85

Το υπό έλεγχο σύστημα είναι $3^{\eta s}$ τάξης.

Για την ρύθμιση των Controlers:

Αρχικά δημιουργούμε τους εκάστοτε ελεγκτές P, PI, PID μέσω της συνάρτησης pidstd με όρισμα τις αντίστοιχες υπολογισμένες μεταβλητές $(K_p, K_p - T_i, K_p - T_i - T_d)$ και έχουμε τα εξής συναρτήσεις μεταφοράς των συστημάτων.

- Το σύστημα P:

$$P = 1.182 \quad (static - gain)$$

- Το σύστημα PI:

$$PI = \frac{1.379s + 0.1716}{s} \quad (continuous - time \ \ tf)$$

- Το σύστημα PID:

$$PID = \frac{2.01s^2 + 2.365s + 0.3529}{s} \quad (continuous - time \ \ tf)$$

Για τον υπολογισμό των συνολικών συναρτήσεων μεταφοράς κάναμε χρήση της συνάρτησης feedback με τα αντίστοιχα ορίσματα και έχουμε τις εξής:

- Το σύστημα με P controler:

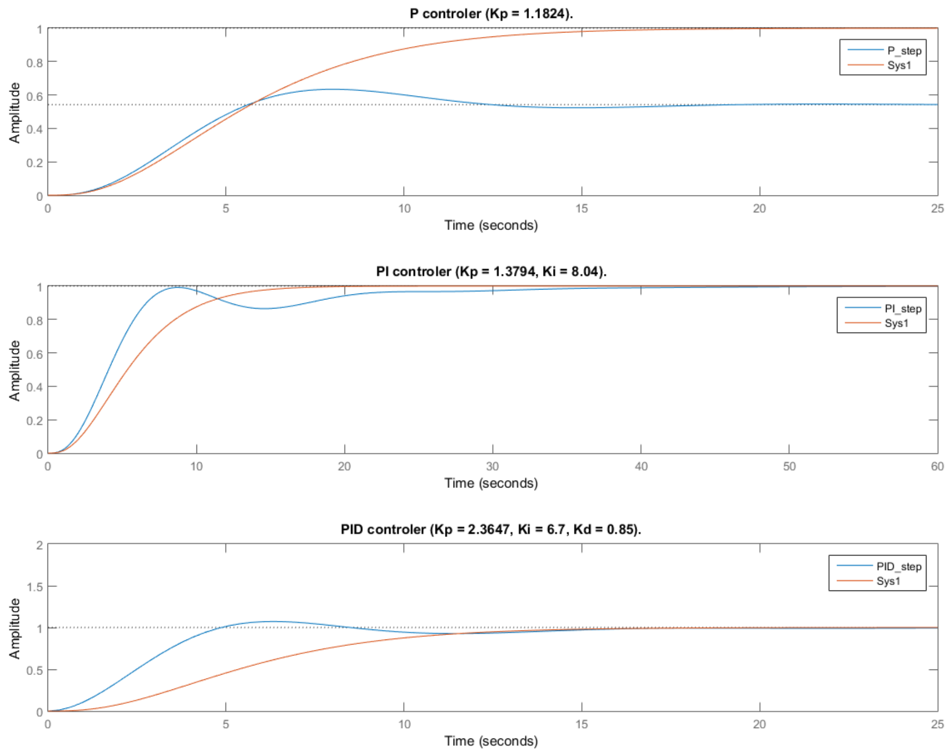
$$Sys_1 = \frac{1.182}{8s^3 + 12s^2 + 6s + 2.182} \quad (continuous - time \ \ tf)$$

- Το σύστημα PI controler:

$$Sys_2 = \frac{1.379s + 0.1716}{8s^4 + 12s^3 + 6s + 2.182s + 0.1716} \quad (continuous - time \ \ tf)$$

- Το σύστημα PID controler:

$$Sys_3 = \frac{2.01s^2 + 2.365s + 0.3529}{8s^4 + 12s^3 + 8.01s^2 + 3.365s + 0.3529} \quad (continuous - time \ \ tf)$$



Σχήμα 3: Βηματικές απόκριση μέ την επίδραση των εκάστοτε controlers.

Για 20%*Overload*: Αρχικά υπολογίζουμε τις μεταβλητές των ελεγκτών για 20%*Overload* και με βάση τον πίνακα:

Ελεγκτης	K	T_i	T_d
P	$\frac{0.7}{K}\frac{T_b}{T_e}$	·	·
PI	$\frac{0.6}{K}\frac{T_b}{T_e}$	T_b	·
PID	$\frac{0.95}{K}\frac{T_b}{T_e}$	$1.4T_b$	$0.47T_e$

υπολογίζουμε τους συντελεστές των ελεγκτών, για overload 0% για set-point response. Σε κάθε περίπτωση και έχουμε:

20% Overload			
Ελεγκτης	K	T_i	T_d
P	2.7588	·	·
PI	2.3647	6.70	·
PID	4.7294	9.38	0.799

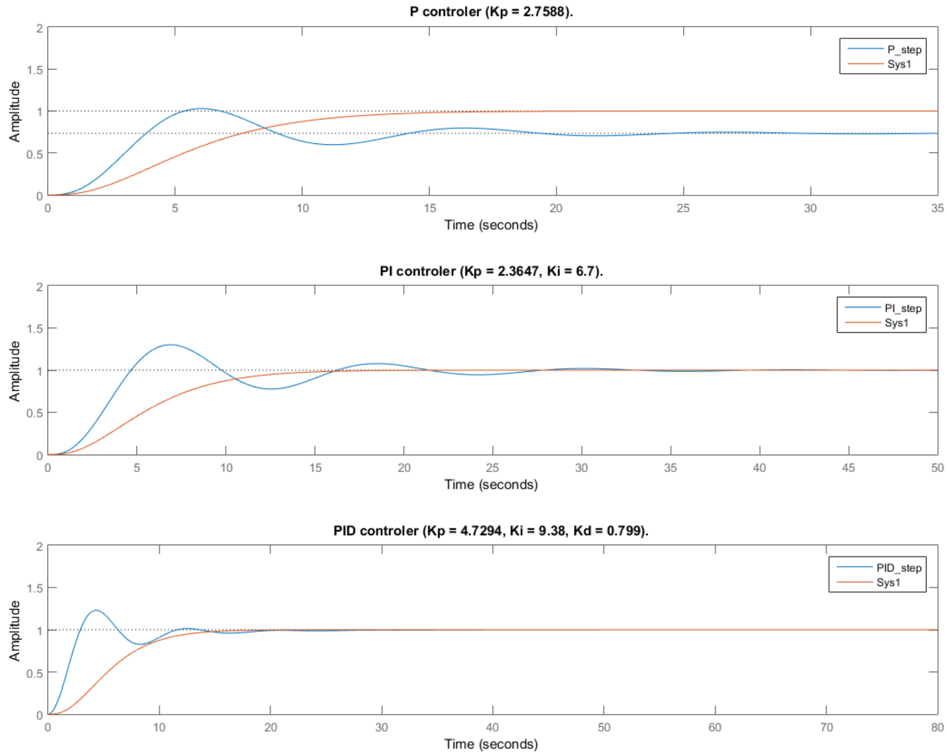
Για την ρύθμιση των Controlers:

Όμοια δημιουργούμε τους εκάστοτε ελεγκτές μέσω της συνάρτησης pidstd και έχουμε τα εξής συναρτήσεις μεταφοράς των συστημάτων.

- Το σύστημα P:
$$P = 2.759 \quad (static - gain)$$
- Το σύστημα PI:
$$PI = \frac{2.365s + 0.3529}{s} \quad (continuous - time \;\; tf)$$
- Το σύστημα PID:
$$PID = \frac{3779s^2 + 4.729s + 0.5042}{s} \quad (continuous - time \;\; tf)$$

Για τον υπολογισμό των συνολικών συναρτήσεων μεταφοράς, με την χρήση της feedback, έχουμε τις εξής:

- Το σύστημα με P controler:
$$Sys_1 = \frac{2.759}{8s^3 + 12s^2 + 6s + 3.759} \quad (continuous - time \;\; tf)$$
- Το σύστημα PI controler:
$$Sys_2 = \frac{2.365s + 0.3529}{8s^4 + 12s^3 + 6s + 3.365s + 0.3529} \quad (continuous - time \;\; tf)$$
- Το σύστημα PID controler:
$$Sys_3 = \frac{3.779s^2 + 4.729s + 0.5042}{8s^4 + 12s^3 + 9.779s^2 + 5.729s + 0.5042} \quad (continuous - time \;\; tf)$$



Σχήμα 4: Βηματικές απόκριση μέ την επίδραση των εκάστοτε controlers με την μέθοδο CHR, set point response και 20% *overload*.