

**دانشکده مهندسی برق و الکترونیک**

**گزارش کار پروژه درس کنترل صنعتی**(PID)

**دانشجو:**

نیما جهان بازفرد (400113020)

**استاد درس:**

جناب آقای دکتر مختار شاصادقی

**آذر 1403**



**فهرست مطالب**

**عنوان** **صفحه**

بخش اول(طراحی PID به صورت حلقه باز)

[1-1به روش ZN 2](#_Toc182860324)

[2-1به روش CC 10](#_Toc182860325)

[3-1به روش CHR-SERVO 12](#_Toc182860326)

[4-1به روشREGULATORY CHR- 12](#_Toc182860326)

[بخش دوم(طراحی PID به صورت حلقه بسته)](#_Toc182860333)

[1-2 به روش ZN 22](#_Toc182860334)

[2-2به روش ZN همراه با ورودی مرجع وزن دار 22](#_Toc182860334)

[3-2 به روش ZN نوسان میرا 22](#_Toc182860334)

[بخش سوم(طراحی PID بر اساس حد بهره و حد فاز-ZN تعمیم یافته)](#_Toc182860346)

[1-3دستیابی به حد بهره6dB 31](#_Toc182860347)

[2-3دستیابی به حد فاز45 deg 31](#_Toc182860347)

[3-3بهینه سازی برای دست یابی به حد فاز 45 deg و حد بهره 6 dB 31](#_Toc182860347)

بخش چهارم(بررسی و مقایسه و انجام عمل integral anti windup برای بخش 3-3 و مقایسه و بررسی آن)

[1-4بررسی کلی تاثیرات عوامل خارجی بر پاسخ 31](#_Toc182860347)

[2-4تشکیل جدول مقایسه و بررسی تاثیر حد فاز و حد بهره(پاسخ پله،اغتشاش پله) 31](#_Toc182860347)

**مقدمه**

قبل از شروع گزارشکار این پروژه باید چند نکته ذکر شود. اولین مورد در بخش آخر پروژه قبلی که مدل سازی سیستم های با رفتار انگرالی بود در صورتی که در بخش سیمولینک توابع تبدیل مدل شده در ضرب شده ولی به طور واضح این مورد در گزارشکار ذکر نشده است ، پس این مورد باید ذکر می شد که توابع مدل شده بر اساس ورودی ضربه باید در انتها در  ضرب می شدند. دومین مورد این است که بجای سیستم  که رفتار مناسبی نداشت از سیستم  استفاده شده است. مورد بعدی در مورد بخش بندی این گزارشکار این گونه می باشد که در ابتدای امر کد های مربوطه به هر بخش ابتدای کار قرار داده میشود که هر بخش کد با کمک کامنت های قرار داده شده به طور واضح مشخص است که چکاری انجام می دهد . در ادامه کد ها توضیحات مربوط به خروجی کد های زده شده و همچنین خروجی سیستم در شرایط مختلف (ورودی پله،اغتشاش،نوبز،تغییر پارامتر و اشباع محرک) نشان داده شده و توضیحات مربوط به آن داده میشود. قابل ذکر است که مورد تغییر پارامتر در همه ی طراحی ها به صورت  در نظر گرفته شده و همچنین برای بررسی تاثیر نویز بر سیستم هم برای همه ی طرای ها نویز سینوسی با فرکانس 60 هرتز و با دامنه ی 0.1 در نظر گرفته شده است. و مورد بعدی این می باشد که حد فاز و حد بهره ی پلنت مورد بررسی به ترتیب برابر با 2.04 deg و 0.6 dB می باشد.

# بخش اول طراحی PID به صورت حلقه باز

## به روش ZN

بعد از تغییر سیستم و مدلسازی بر اساس G5 که بهترین عملکرد را بین مدل ها داشت این گونه ادامه می دهیم .

%ZN OPEN LOOP

k=15.7452;

T=2.37;

T\_d=0.5788;

alpha=T\_d/T;

%PID CONTROLER PARAMERTERS

kp=1.2/(k\*alpha);

Ti=2\*T\_d;

Td=Ti/4;

%KP=0.3121

%Ti=1.1576

%Td=0.2894

s=tf('s');

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

g=63/((s+0.5)\*(s+2)\*(s+4));

hold on

nyquist(g)

nyquist(c\*g)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'r')

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.04 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=42.05 deg

%GMcg=25.8672 dB

%calculating of charactristics of system response to pulse input

t = out.zop(:, 1);

y = out.zop(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.62

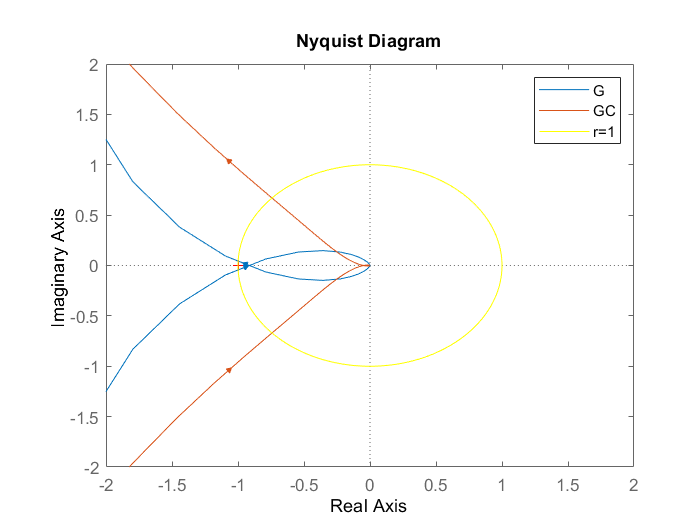
%Ts%2=6.1

%overshoot%=43.7%

%d=0.08

%IAE=1.2990

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده بیشتر شده و به 45 درجه نزدیک شده است(42) و همچنین بهره سیستم جبران شده برحسب dB خیلی بیشتر شده است(25.87).

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.62

Ts%2=6.1

overshoot%=43.7%

d=0.08

IAE=1.2990



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 10 اعمال شده است):

Ts(5%)=7.45

Ts(10%)=6.03

overshoot%=228%

d=0.08

IAE=4.4827

t = out.zop1(:, 1);

y = out.zop1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-1.2990;

%Ts5%=7.45

%Ts10%=6.03

%overshoot%=228%

%d=0.08

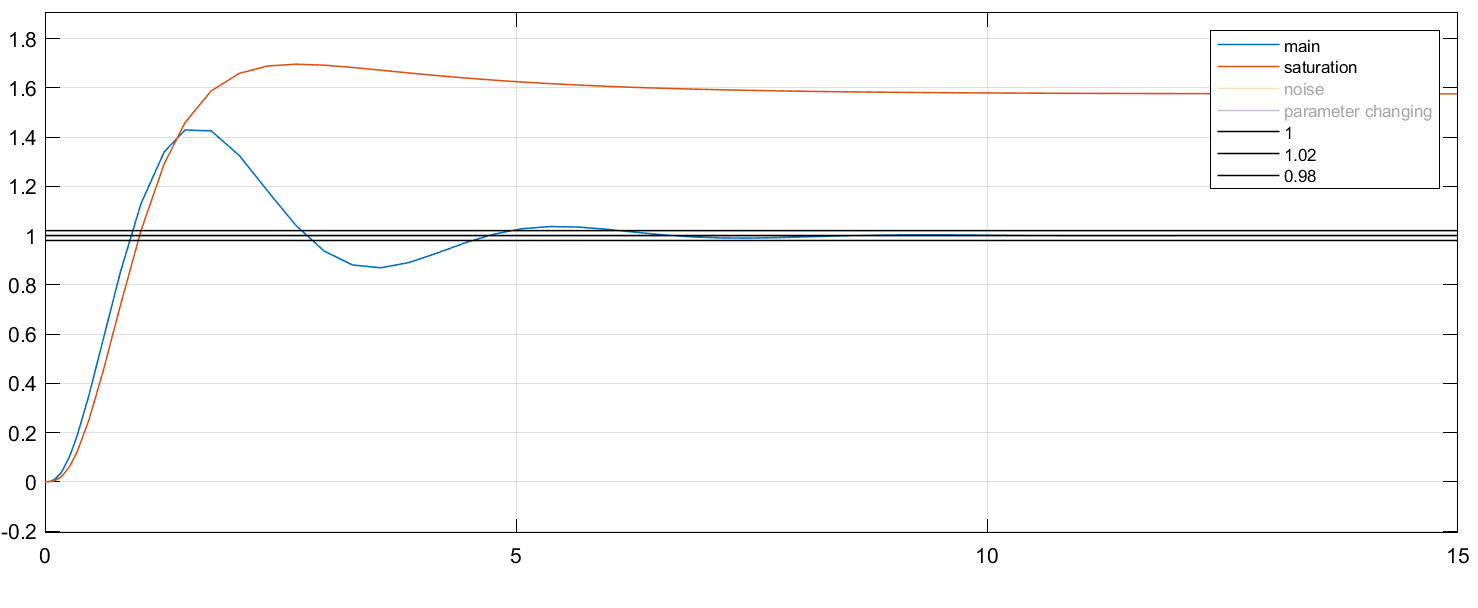
%IAE=4.4827



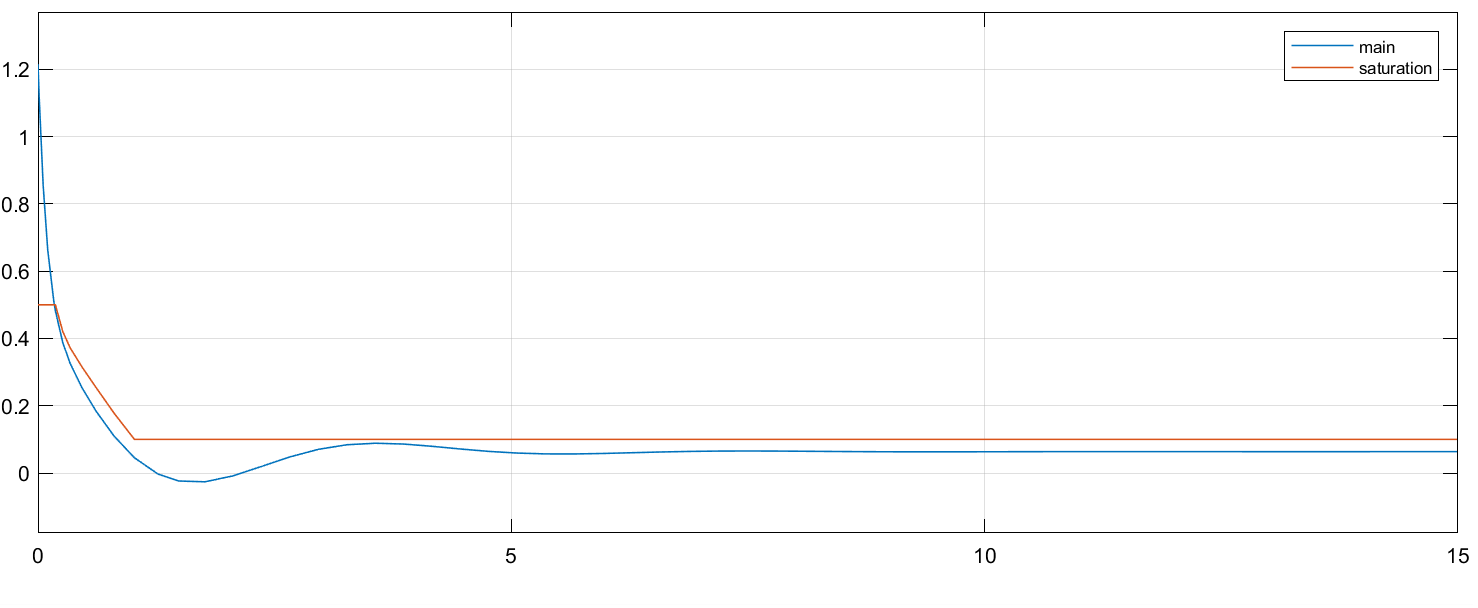
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

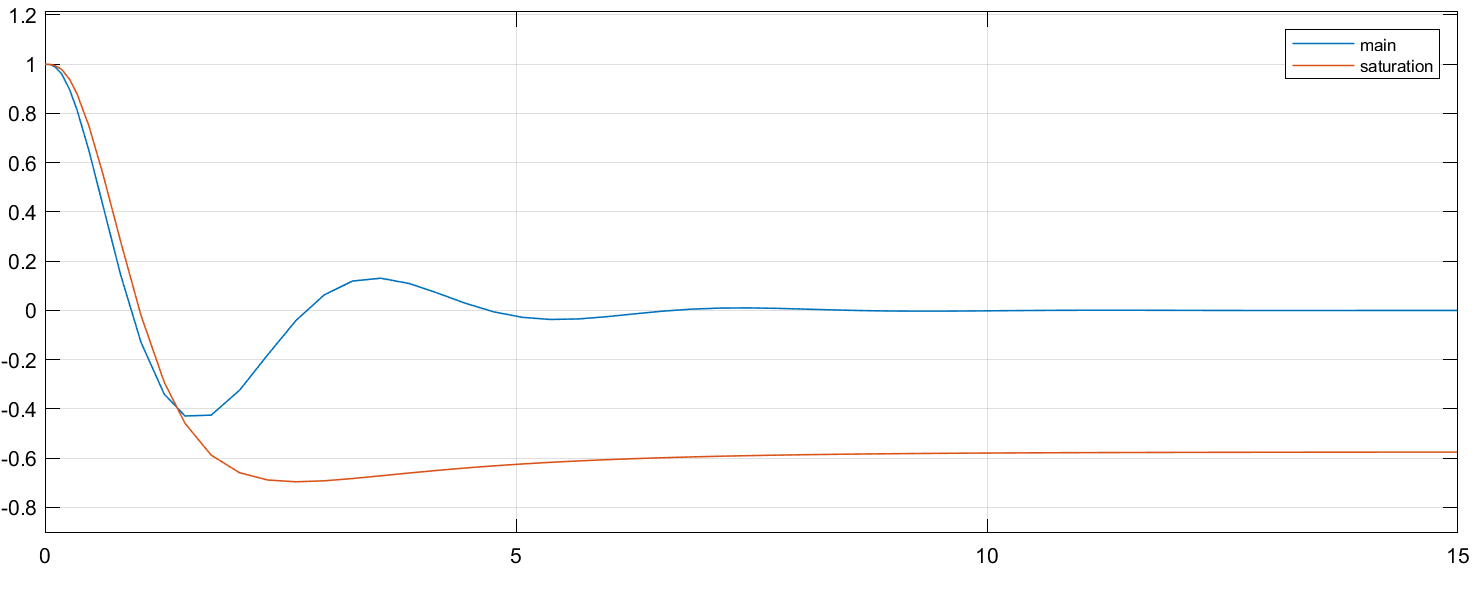
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.

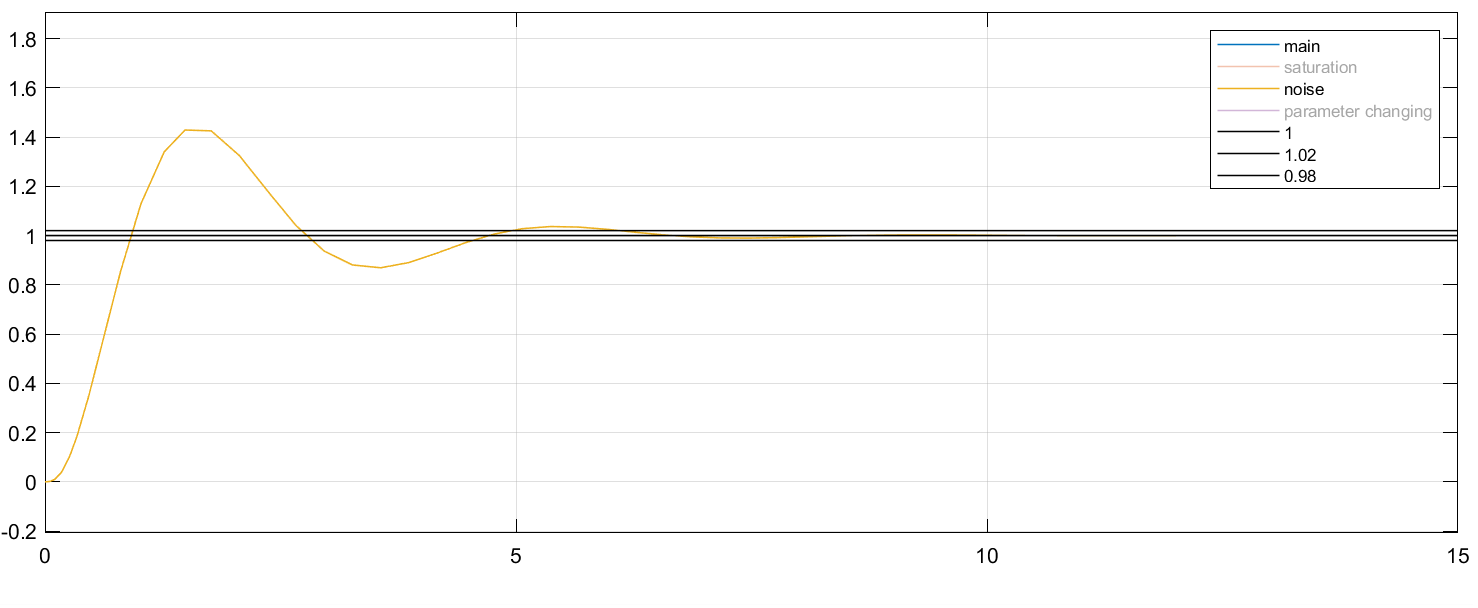


همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:

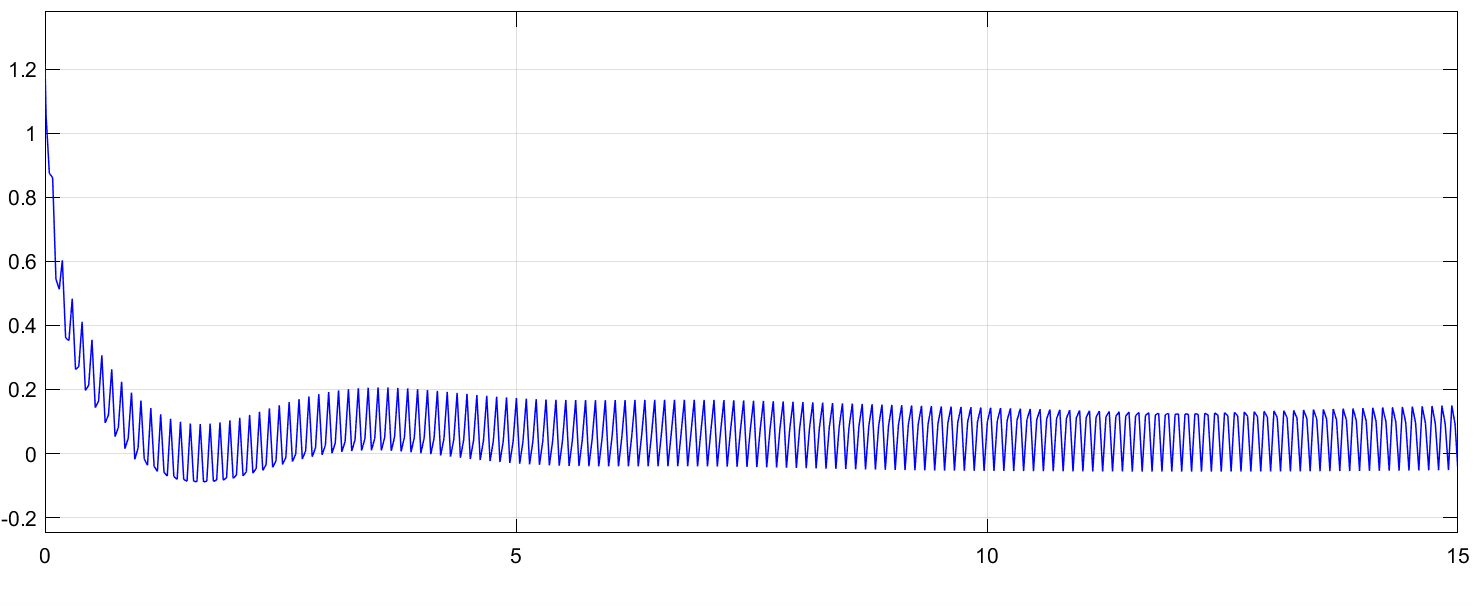
  
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

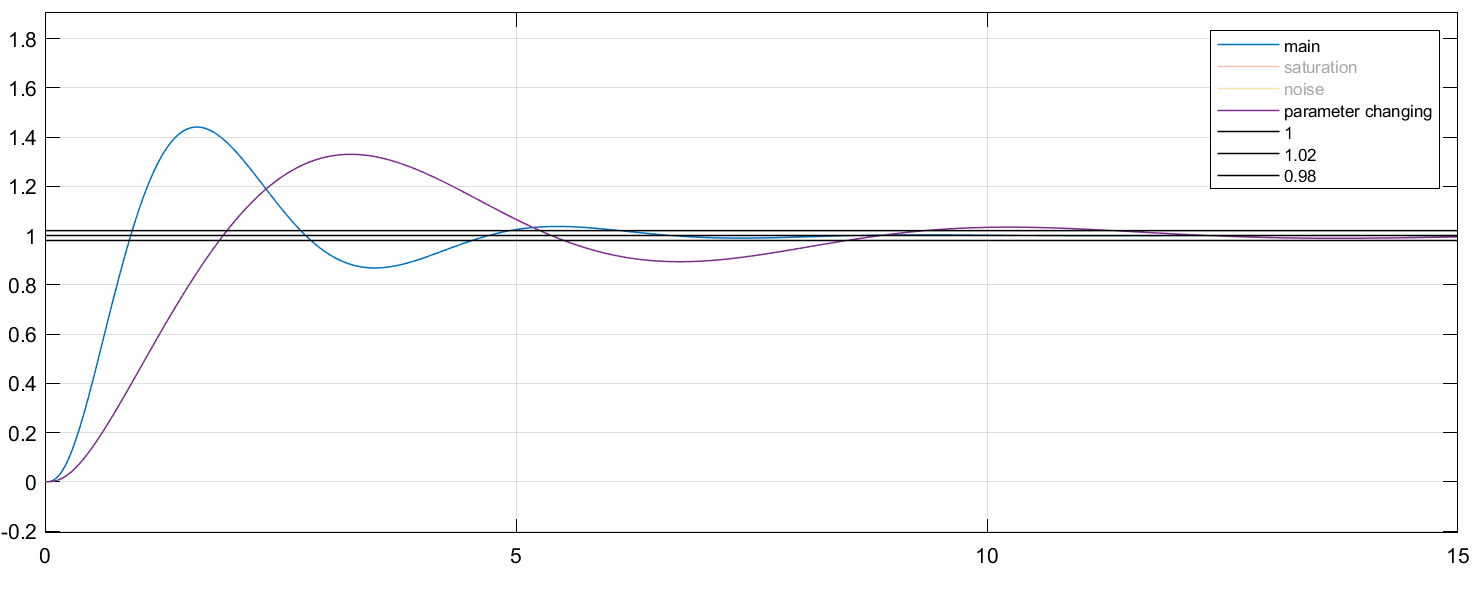


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم که:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و کاهش ماکزیمم فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

## به روش CC

%CC OPEN LOOP

k=15.7452;

T=2.37;

T\_d=0.5788;

alpha=T\_d/T;

%PID CONTROLER PARAMERTERS

kp=(1/k)\*((1.35/alpha)+0.27);

Ti=T\_d\*((2.5+(0.5\*alpha))/(1+(0.6\*alpha)));

Td=T\_d\*(0.37/(1+0.2\*alpha));

%KP=0.3682

%Ti=1.3237

%Td=0.2042

s=tf('s');

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

g=63/((s+0.5)\*(s+2)\*(s+4));

hold on

nyquist(g)

nyquist(c\*g)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'r')

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.4 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=35.47 deg

%GMcg=25.59 dB

t = out.cop(:, 1);

y = out.cop(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.56

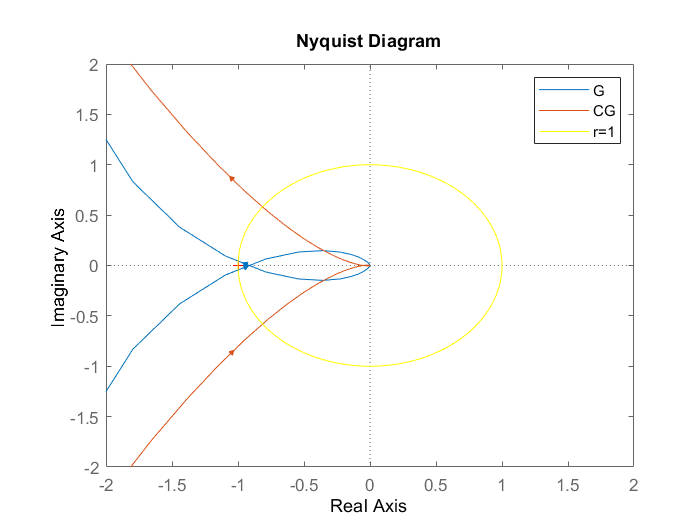
%Ts%2=9.85

%overshoot%=57.38%

%d=0.34

%IAE=1.9798

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده بیشتر شده و به 45 درجه نزدیک شده است(35.5) و همچنین بهره سیستم جبران شده برحسب dB خیلی بیشتر شده است(25.6).

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.56

Ts%2=9.85

overshoot%=57.38%

d=0.34

IAE=1.9798

  
در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 15 اعمال شده است):

Ts5%=10.89

Ts10%=9.4

overshoot%=227%

d=0.27

IAE=5.34

t = out.cop1(:, 1);

y = out.cop1(:, 2);

plot(t, y '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-1.9798;

%Ts5%=10.89

%Ts10%=9.4

%overshoot%=227%

%d=0.27

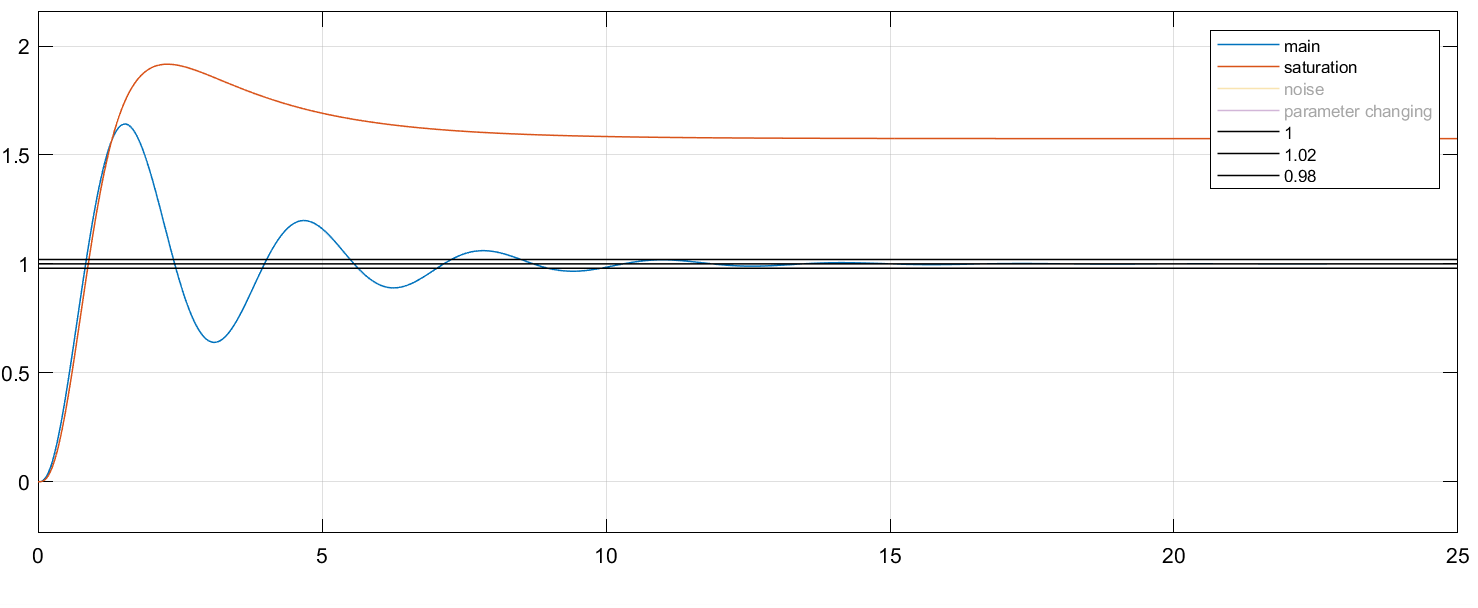
%IAE=5.34



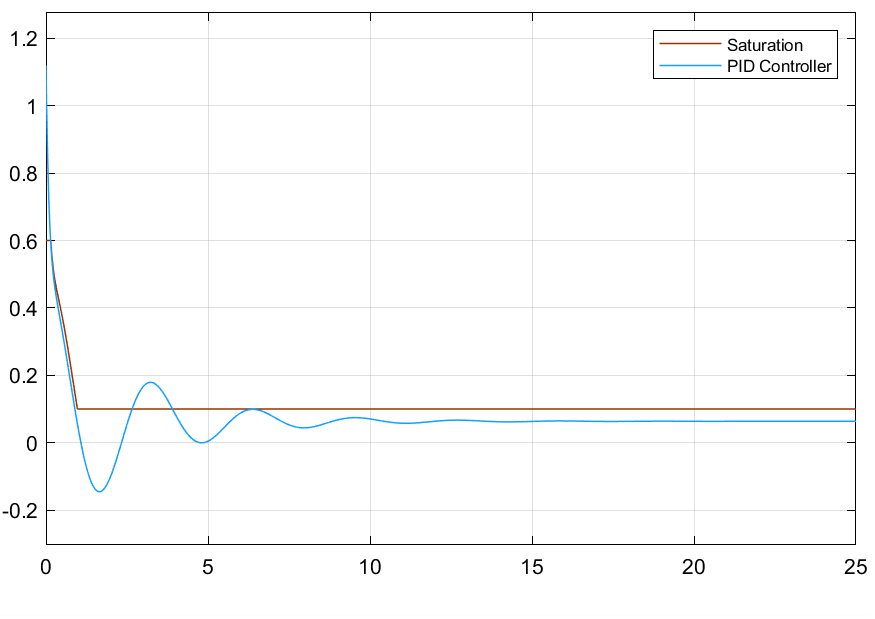
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

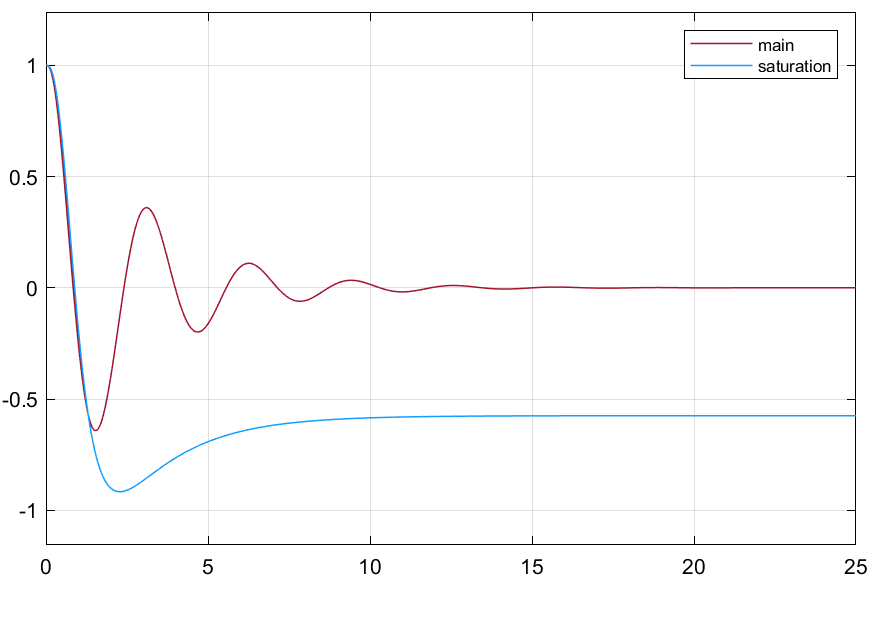
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



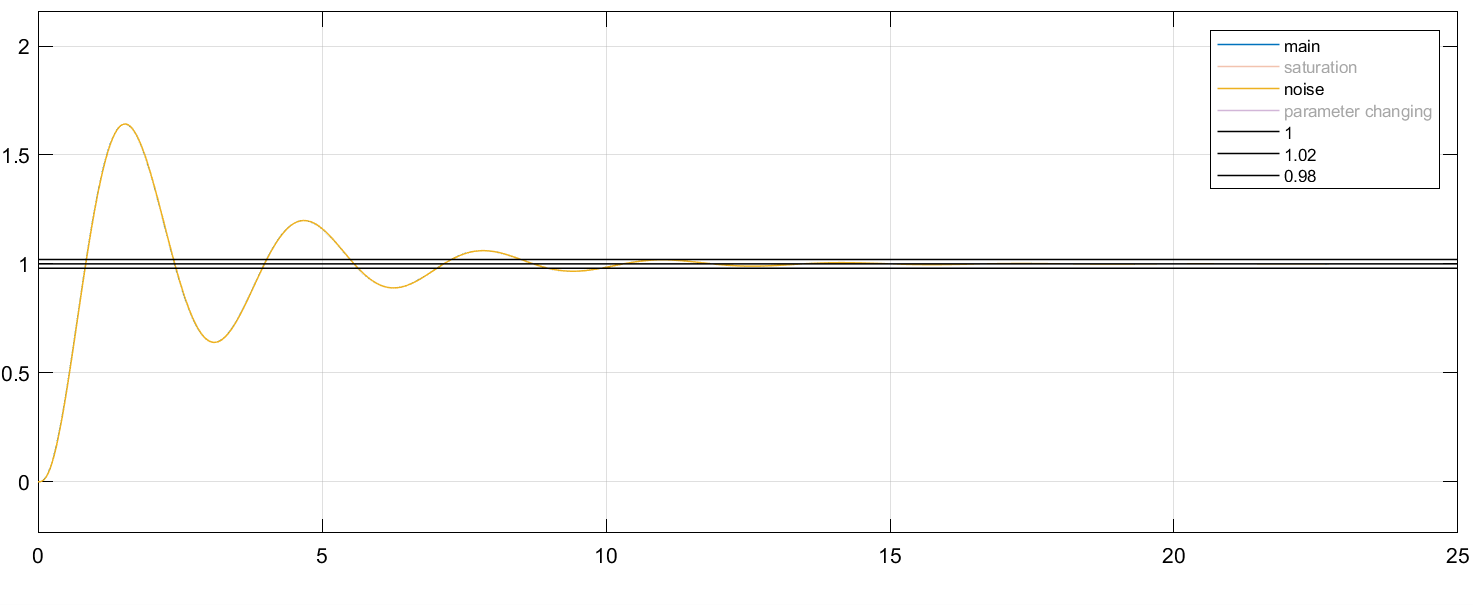
همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



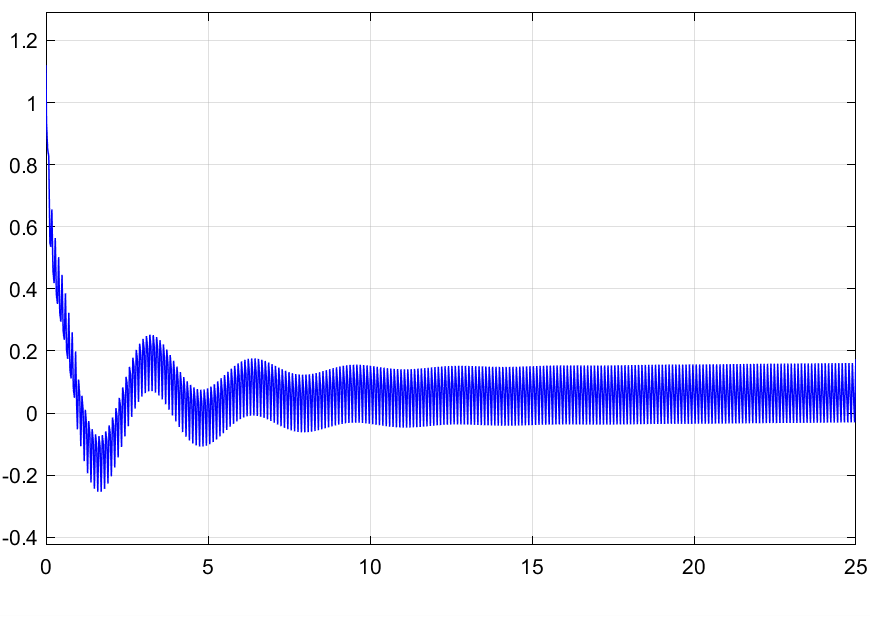
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

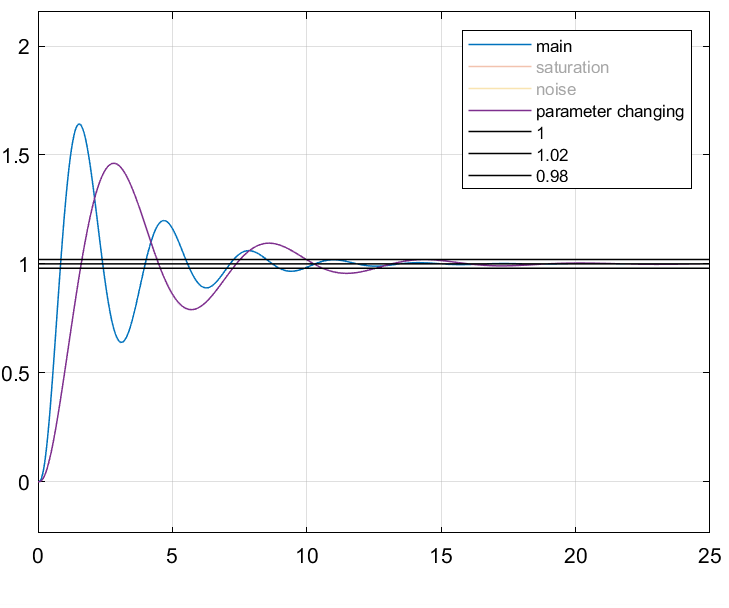


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و کاهش ماکزیمم فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

## به روش CHR-SERVO

%CHR\_SERV OPEN LOOP

k=15.7452;

T=2.37;

T\_d=0.5788;

alpha=T\_d/T;

%PID CONTROLER PARAMERTERS

kp=0.6/(k\*alpha);

Ti=T\_d;

Td=T\_d\*0.5;

%KP=0.1560

%Ti=0.5788

%Td=0.2894

s=tf('s');

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

g=63/((s+0.5)\*(s+2)\*(s+4));

hold on

nyquist(g)

nyquist(c\*g)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'r');

legend('G','GC','r=1');

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.4 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=16.8863 deg

%GMcg=31.5971 dB

t = out.chos(:, 1);

y = out.chos(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

error=1-y;

absIntegralError = trapz(t, abs(error));

%parameters for step

%Tr=1.24

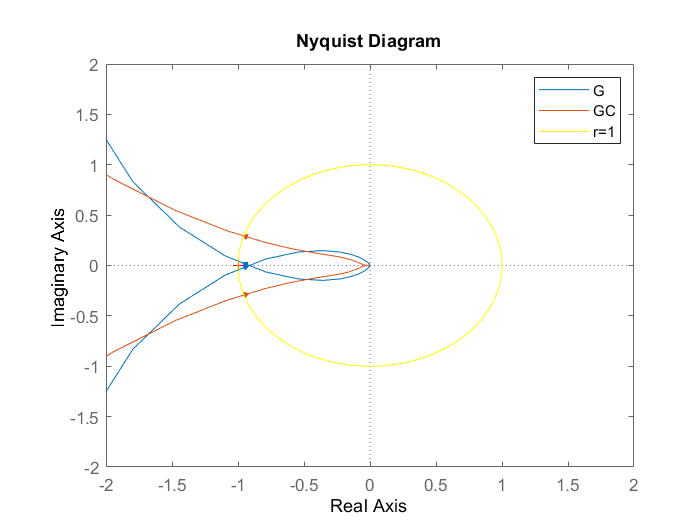
%Ts%2=5.16

%overshoot%=9.8%

%d=-

%IAE=1.1537

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده بیشتر شده و به 45 درجه نزدیک شده است(16.9) و همچنین بهره سیستم جبران شده برحسب dB خیلی بیشتر شده است(31.6).

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=1.24

Ts%2=5.16

overshoot%=9.8%

d=-

IAE=1.1537



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 8 اعمال شده است):

Ts5%=6.16

Ts10%=5.94

overshoot%=382%

d=-

IAE=11.08

t = out.chos1(:, 1);

y = out.chos1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-1.1537;

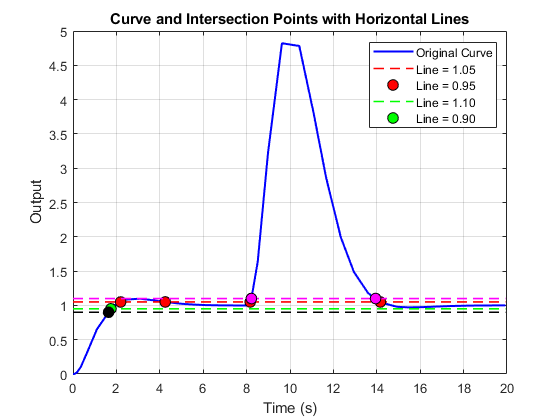
%Ts5%=6.16

%Ts10%=5.94

%overshoot%=382%

%d=-

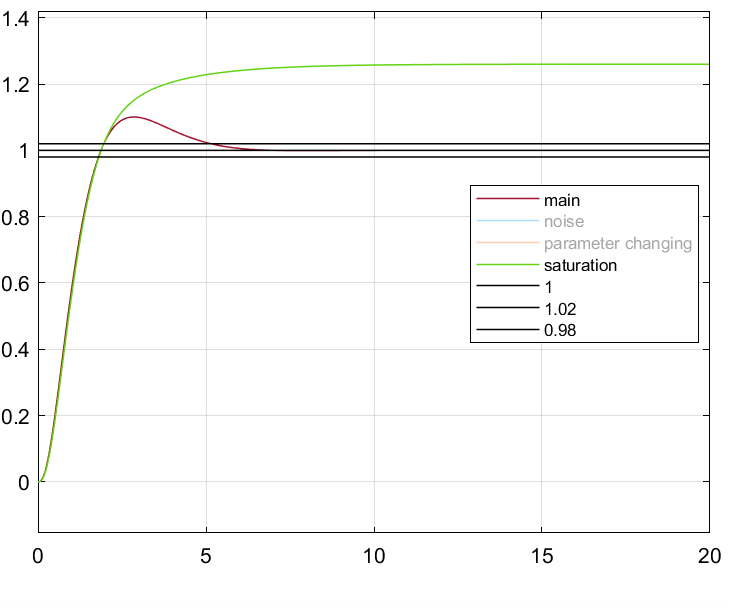
%IAE=11.08



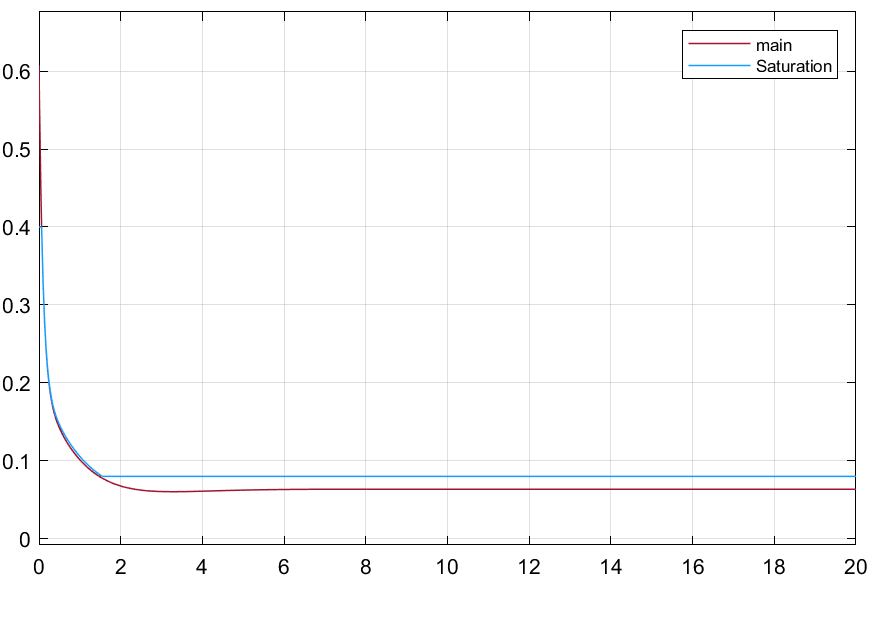
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

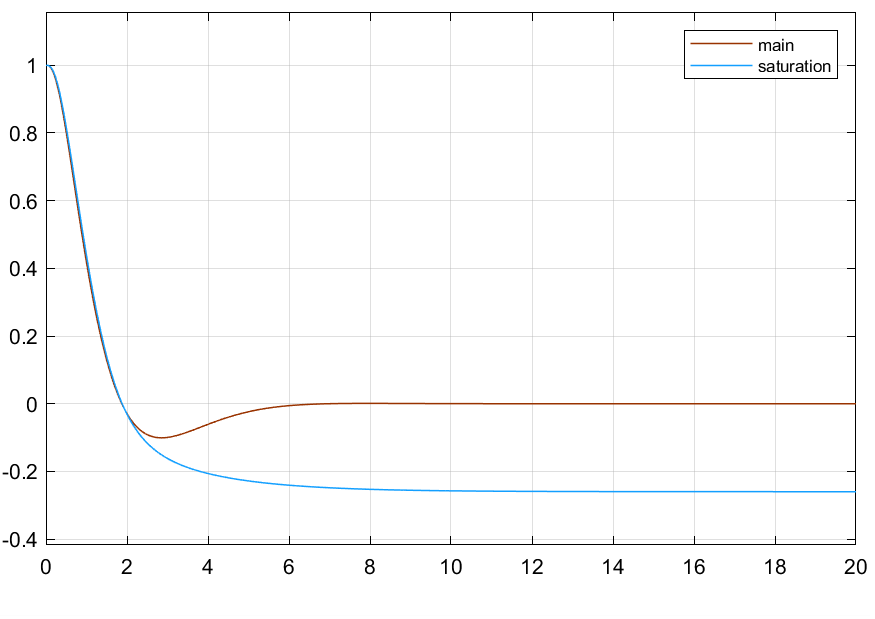
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



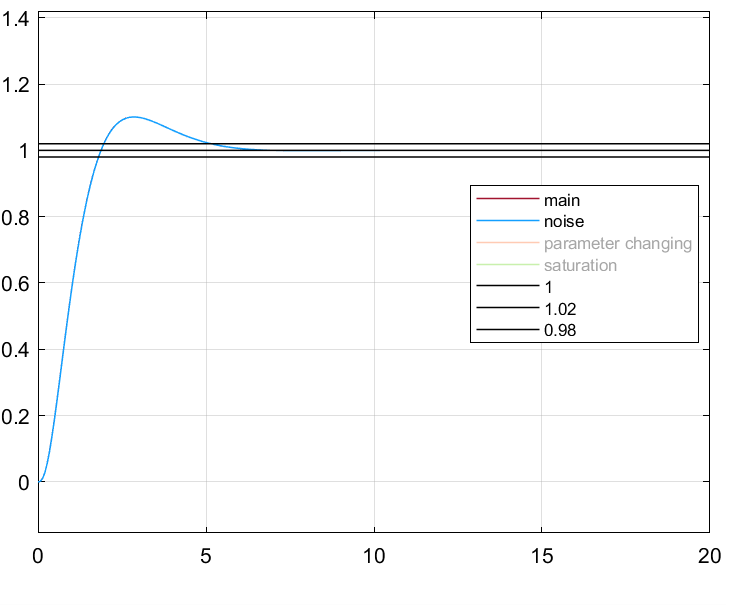
همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



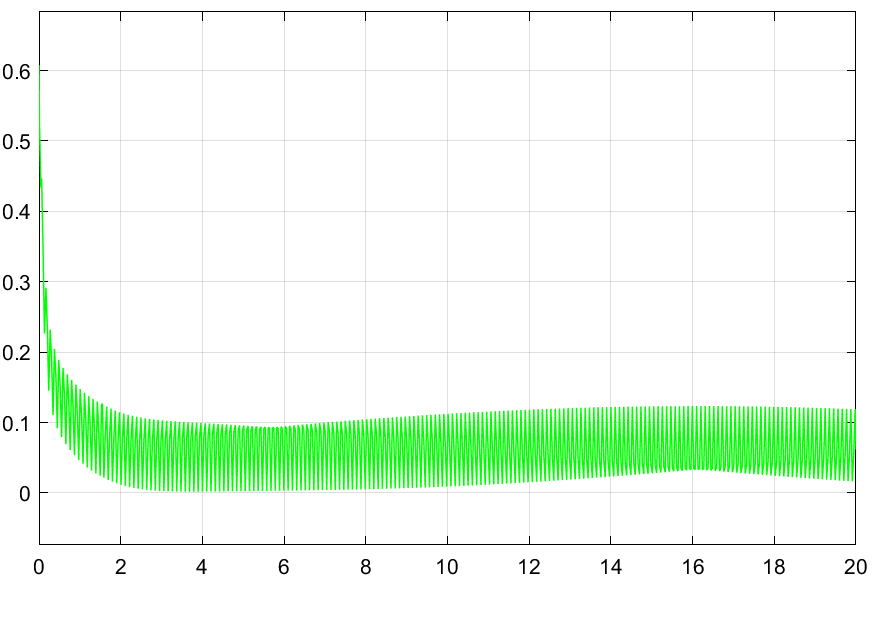
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

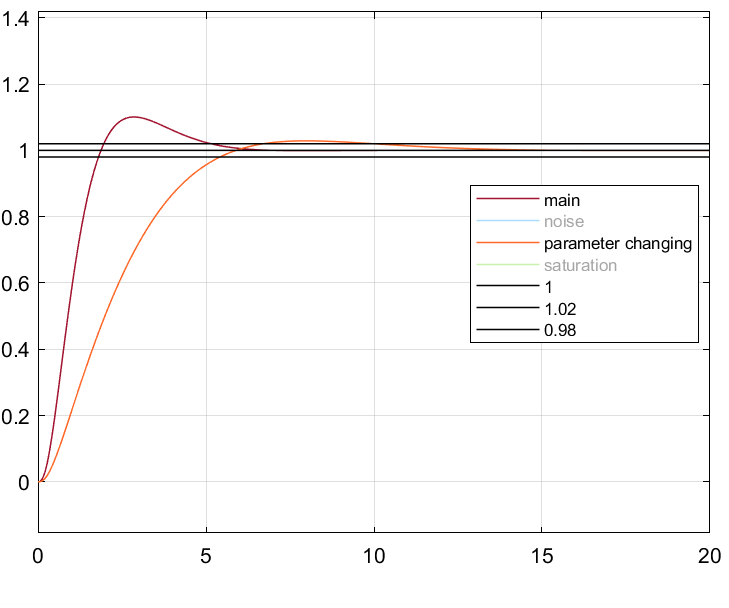


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و کاهش ماکزیمم فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

## به روش CHR-REGULATORY

%CHR\_REG OPEN LOOP

k=15.7452;

T=2.37;

T\_d=0.5788;

alpha=T\_d/T;

%PID CONTROLER PARAMERTERS

kp=0.95/(k\*alpha);

Ti=T\_d\*2.38;

Td=T\_d\*0.42;

%KP=0.2471

%Ti=1.3775

%Td=0.2431

s=tf('s');

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

g=63/((s+0.5)\*(s+2)\*(s+4));

hold on

nyquist(g)

nyquist(c\*g)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'y');

legend('G','GC','r=1');

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.4 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=44.94 deg

%GMcg=28.92 dB

t = out.chor(:, 1);

y = out.chor(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.69

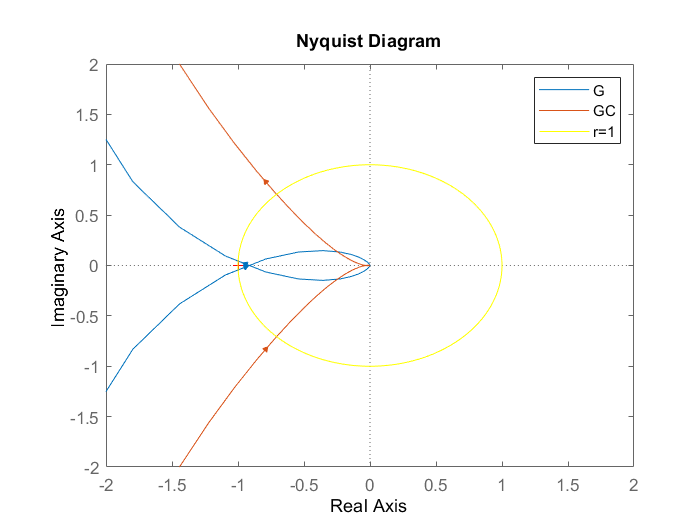
%Ts%2=10.58

%overshoot%=55%

%d=0.22

%IAE=2.0213

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده بیشتر شده و به 45 درجه نزدیک شده است(44.94) و همچنین بهره سیستم جبران شده برحسب dB خیلی بیشتر شده است(28.92).

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.69

Ts%2=10.58

overshoot%=55%

d=0.22

IAE=2.0213



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 20 اعمال شده است):

Ts5%=12.25

Ts10%=10.25

overshoot%=272%

d=0.234

IAE=7.44

t = out.chor1(:, 1);

y = out.chor1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-2.0213;

%Ts5%=12.25

%Ts10%=10.25

%overshoot%=272%

%d=0.234

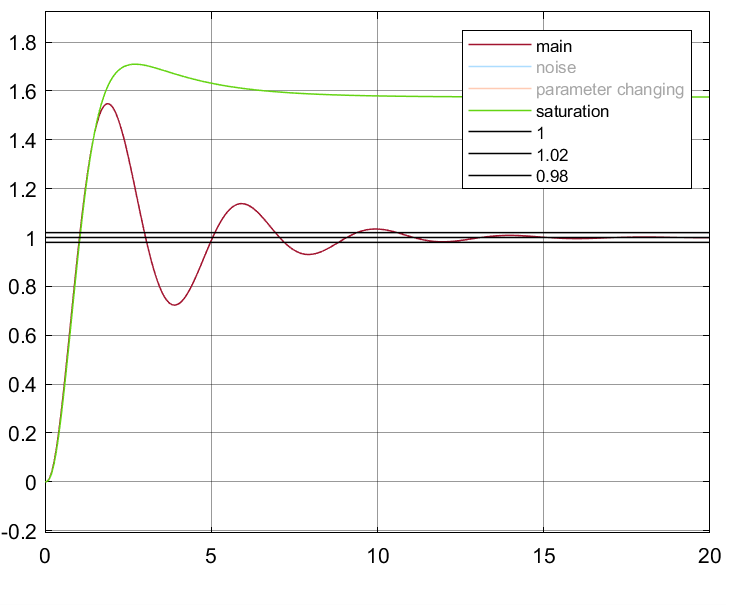
%IAE=7.44



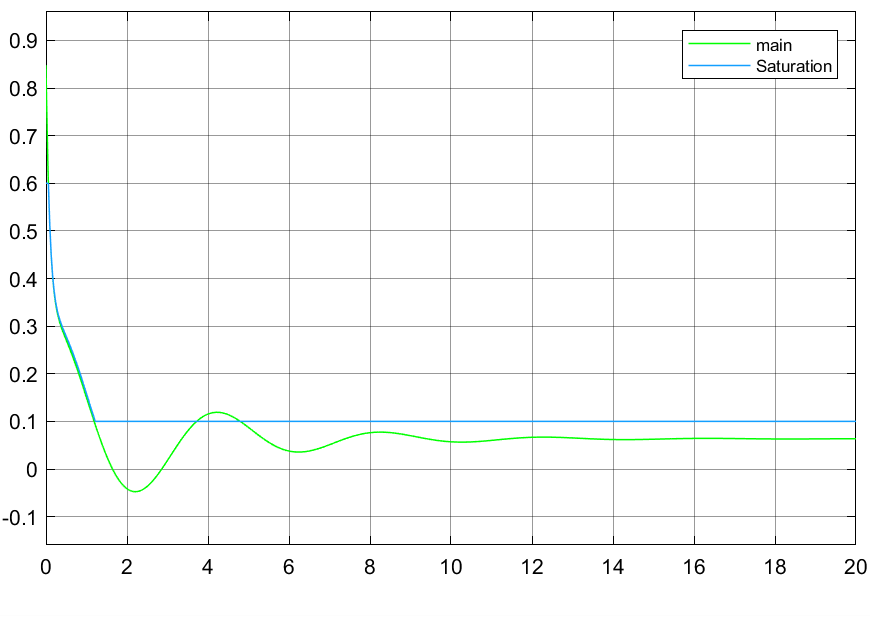
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

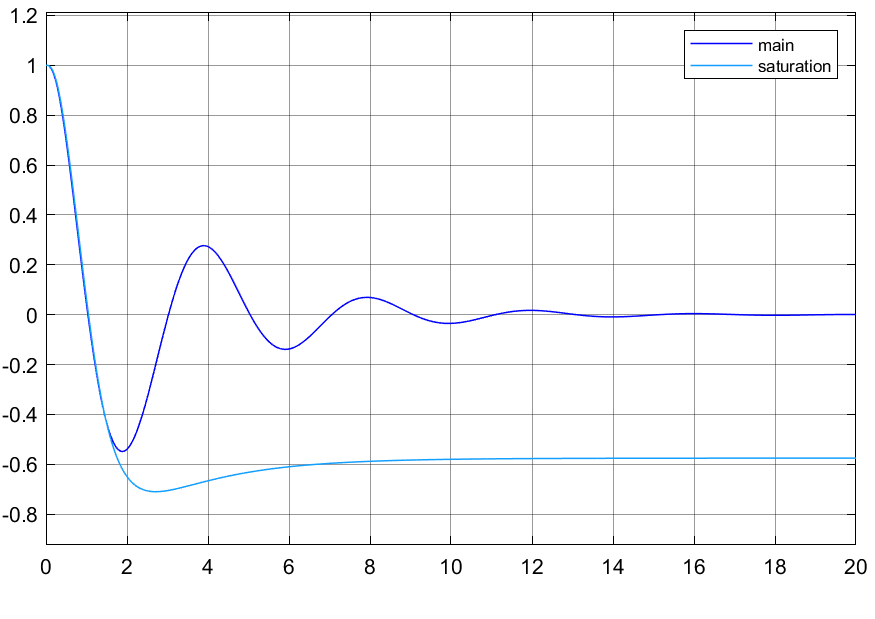
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



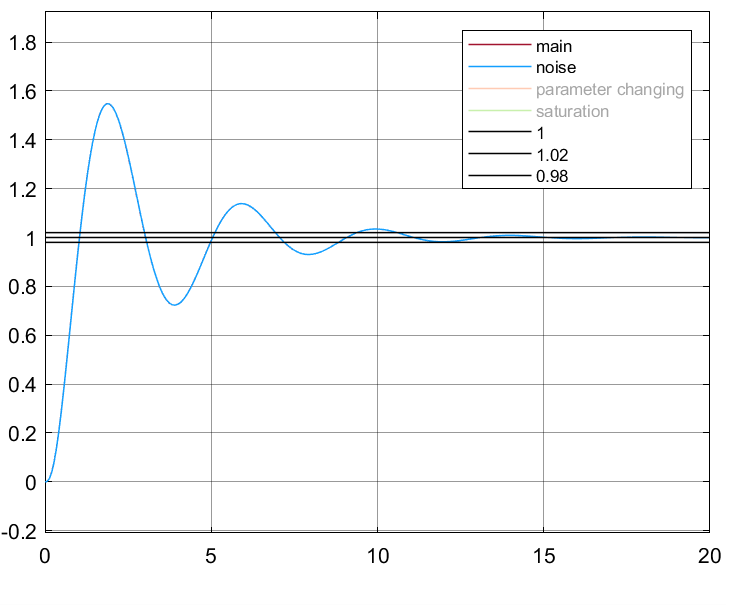
همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



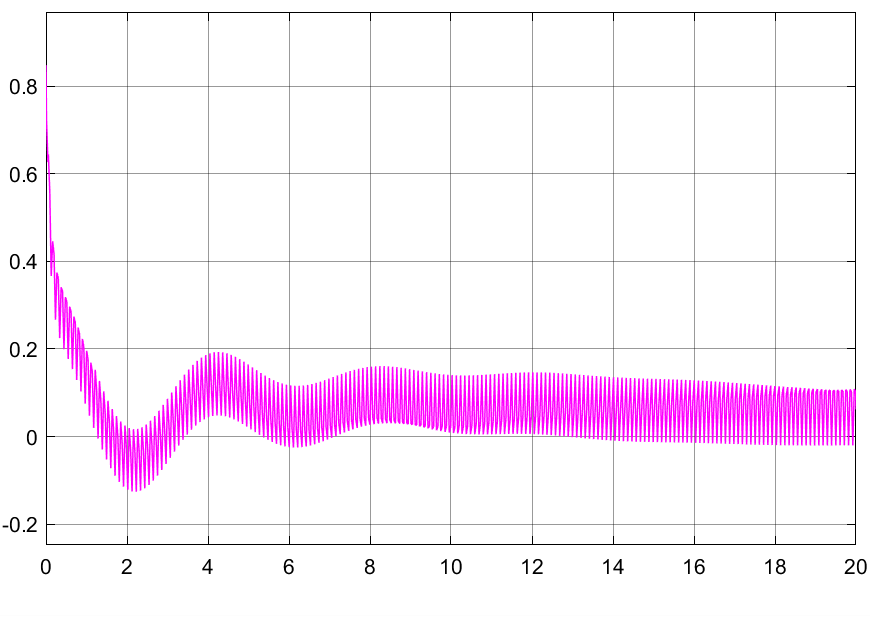
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

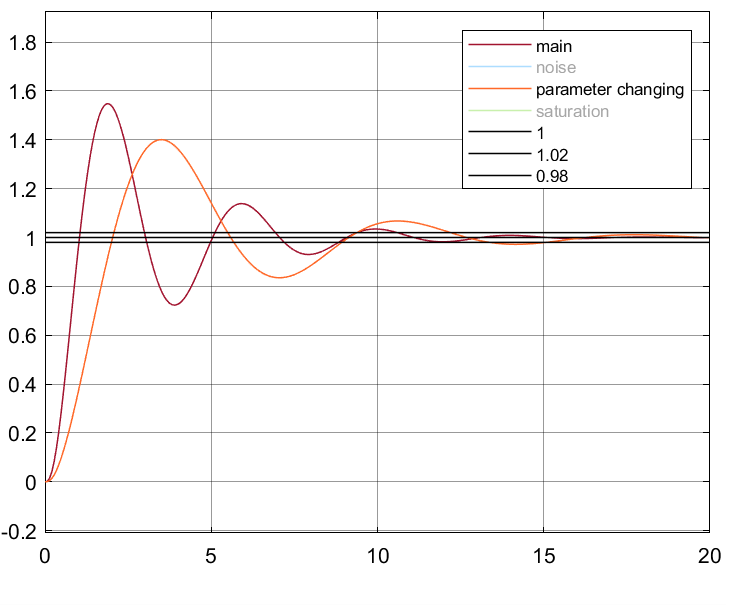


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و کاهش ماکزیمم فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

# بخش دوم

# طراحی PID به صورت حلقه بسته

## 1-2 به روش ZN

s=tf('s');

g=63/((s+0.5)\*(s+2)\*(s+4));

%sisotool(g)

ku=1.085;

Tu=2.29;

%PID controler

kp=0.6\*ku;

Ti=Tu/2;

Td=Tu/8;

%KP=0.6510

%Ti=1.1450

%Td=0.2863

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

hold on

nyquist(g)

nyquist(c\*g)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'y');

legend('G','GC','r=1');

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.04 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=35.1161 deg

%GMcg=19.5548 dB

t = out.zpc(:, 1);

y = out.zpc(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.40

%Ts%2=4

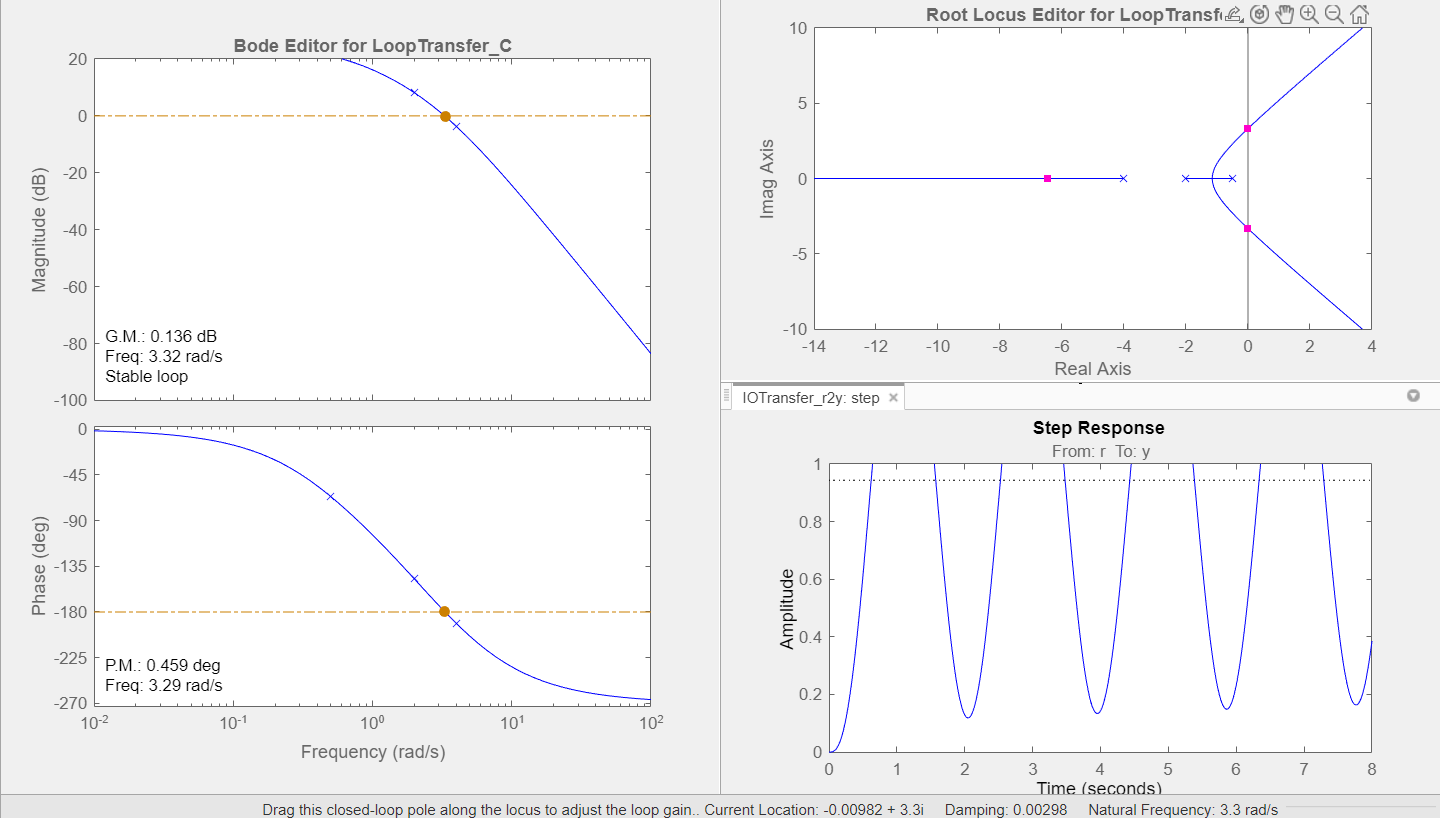
%overshoot%=45.9%

%d=0.076

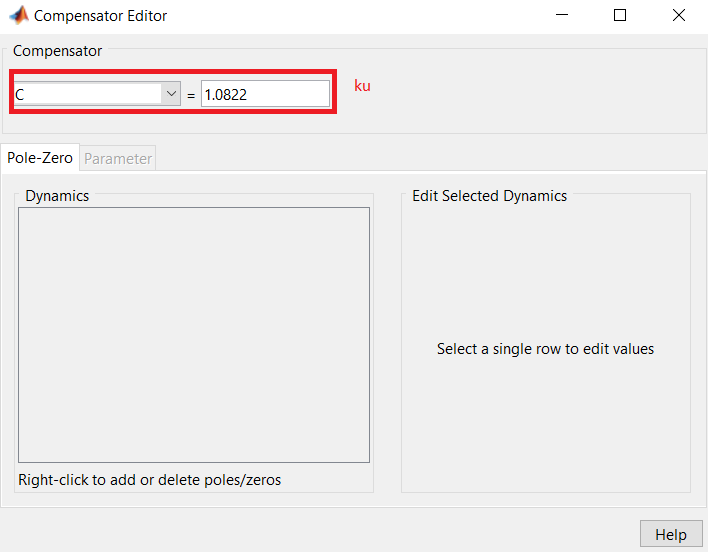
%IAE=0.8461

در این روش برای بدست آوردن مقادیر ضرایب PID نیازی به پارامتر های مدل نیست در ابتدای کار با استفاده از یک کنترل کننده KP بهره را آننچنان تغییر میدیم که که پاسخ خروجی نوسان پایدار شود که با ضریب بهره ای به دست آمده که آن را ku قرار می دهیم و دوره تناوب نوسان پایدار را Tu می نامیم و در ادامه ضرایب را از روی این مقادیر بدست می آوریم.(استفاده کردن از (sisotool

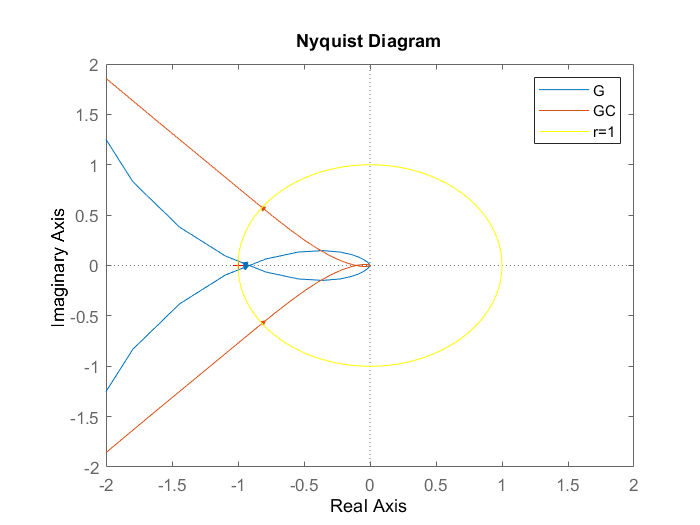
قرار دادن قطب ها روی محور موهومی برای دست یابی به نوسان پایدار:



بدست آوردن Tu اروی پاسخ پله و بدست آوردن ضریب ku.



در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده بیشتر شده و به 45 درجه نزدیک شده است(35.1) و همچنین بهره سیستم جبران شده برحسب dB خیلی بیشتر شده است(19.6).

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.40

Ts%2=4

overshoot%=45.9%

d=0.076

IAE=0.8461



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 7 اعمال شده است):

Ts5%=4.1

Ts10%=3.09

overshoot%=139%

d=0.046

IAE=1.7800

t = out.zpc1(:, 1);

y = out.zpc1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-0.8461;

%Ts5%=4.1

%Ts10%=3.09

%overshoot%=139%

%d=0.046

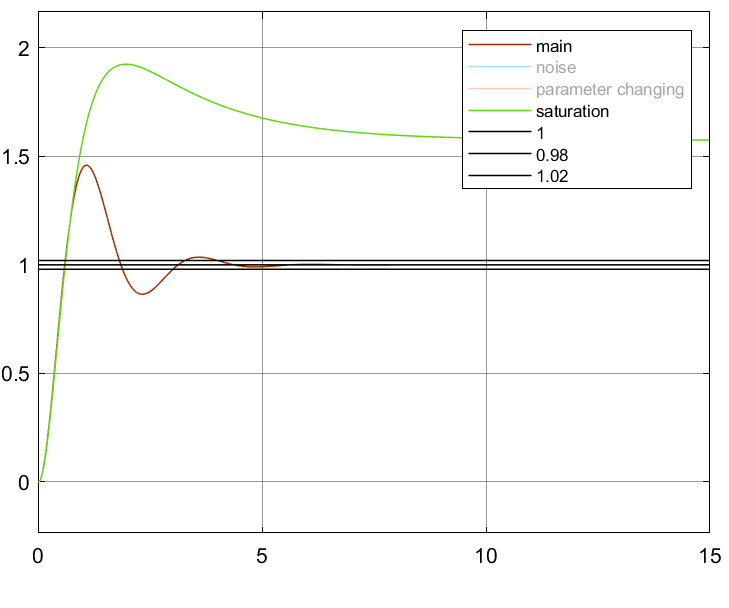
%IAE=1.7800



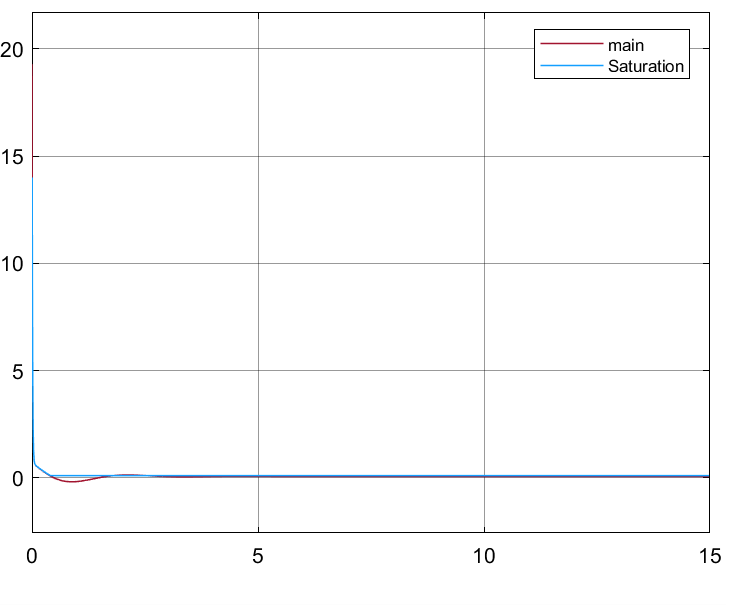
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

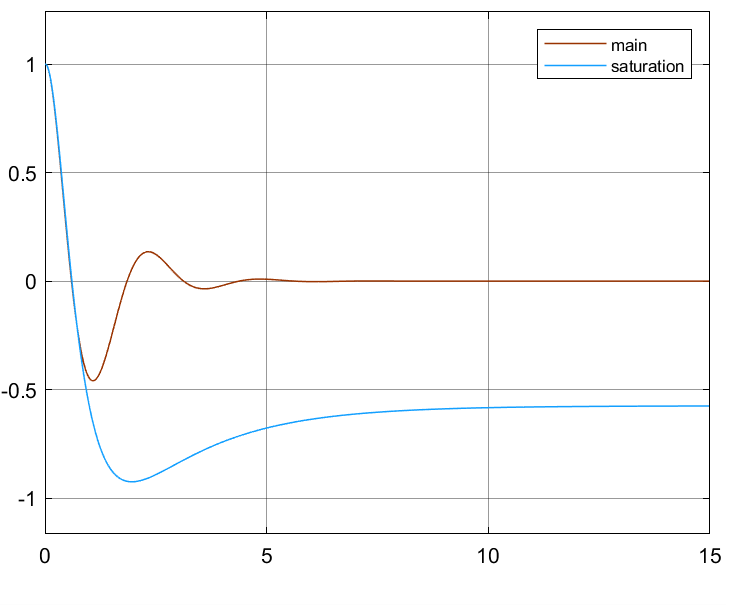
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



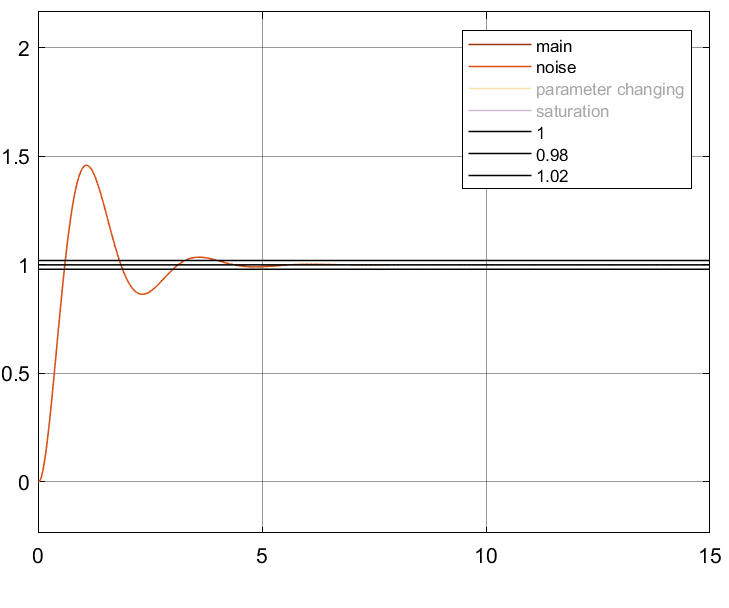
همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



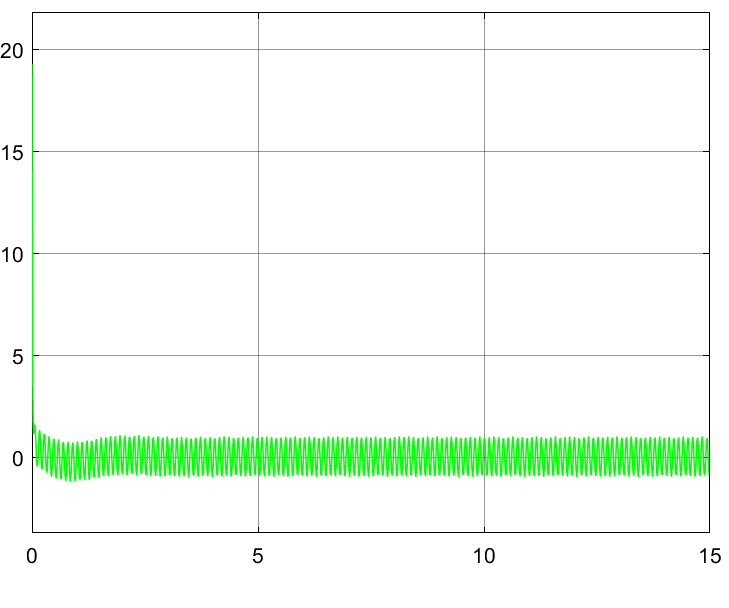
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

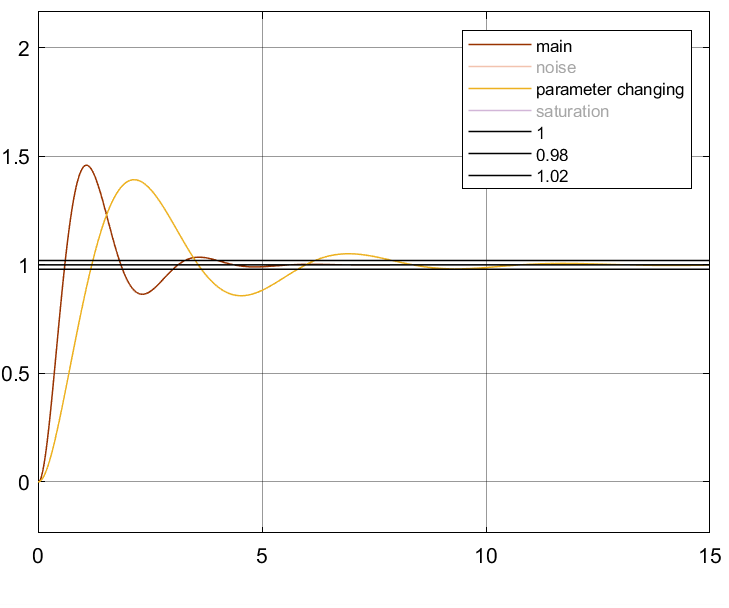


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثرعدم قطعیت:**

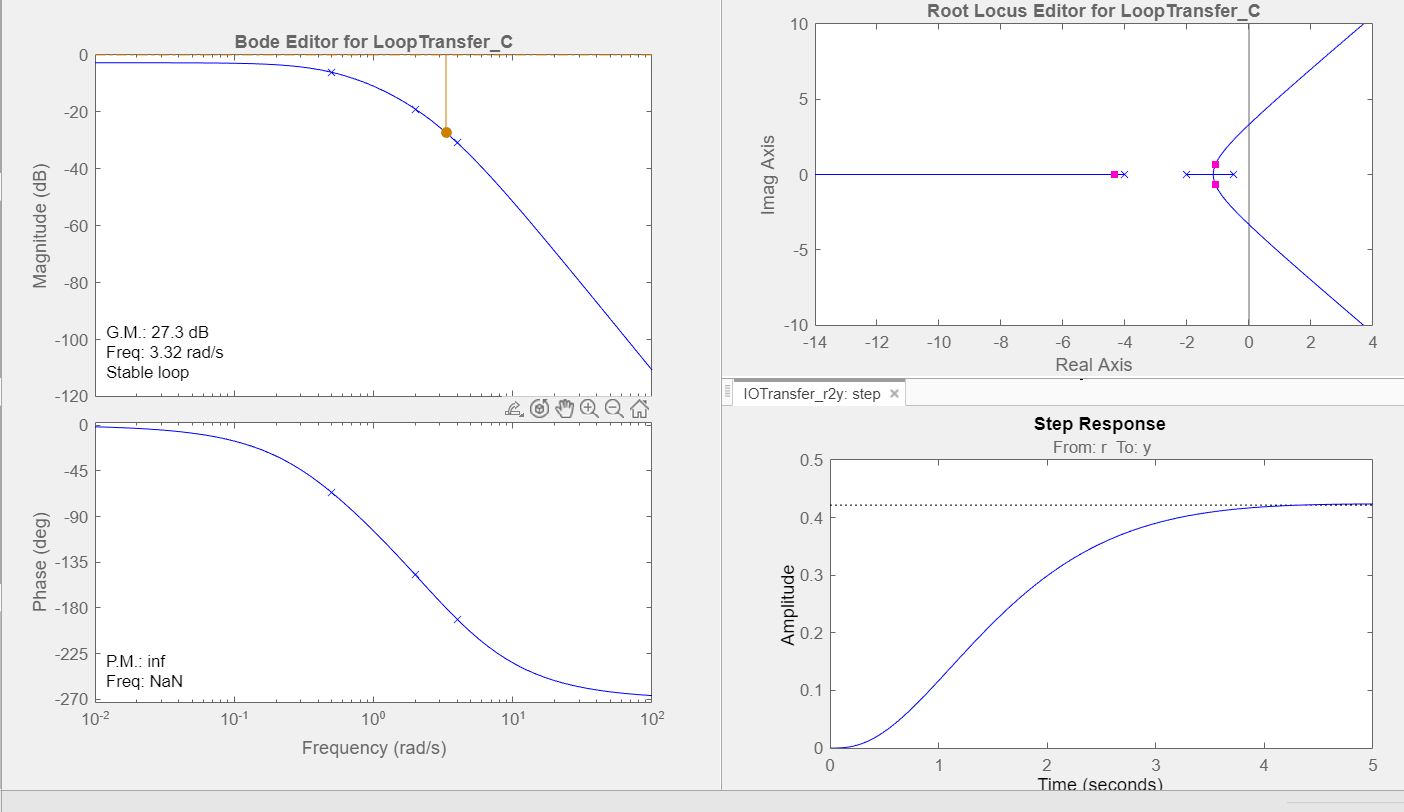
در ادامه با مقایسه خروجی ها داریم:



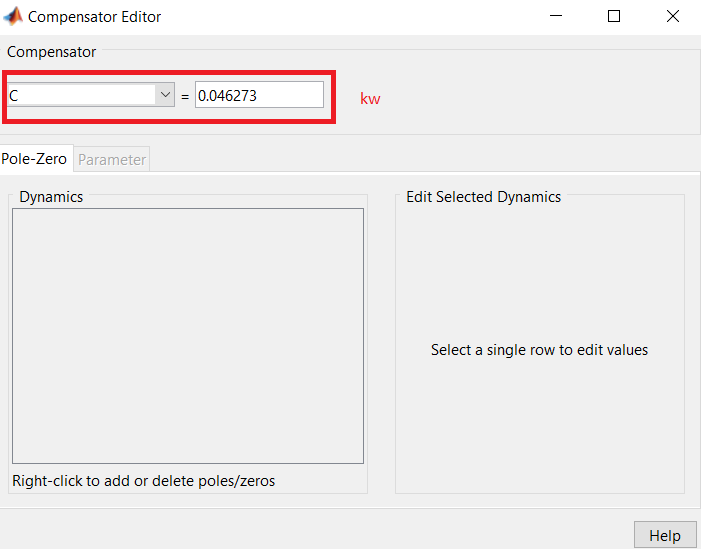
تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و کاهش ماکزیمم فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

## 2-2 به روش ZN همراه با ورودی مرجع وزن دار

در این بخش علاوه بر مراحل طی شده در بخش قبل برای بدست آوردن مقادیر ku و Tu برای بدست آوردن وزن جهت ورودی مرجع وزن دار باید kw را این گونه بدست آوریم که ضریب بهره ای که باعث می شود پاسخ پله میرایی بحرانی شود و داریم :



و برای ضریب kw داریم:



در ادامه روند را مانند روش های قبل طی می کنیم.

s=tf('s');

g=63/((s+0.5)\*(s+2)\*(s+4));

sisotool(g)

ku=1.085;

Tu=2.29;

%PID controler

kp=0.6\*ku;

Ti=Tu/2;

Td=Tu/8;

%KP=0.6510

%Ti=1.1450

%Td=0.2863

kw=0.045;

a=kw/kp;

%a=0.0691

Gc=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

Gff=kp\*a+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

G=(Gff\*Gp)/(1+(Gp\*Gc)-(Gff\*Gp));

hold on

nyquist(Gp)

nyquist(G)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'y');

legend('Gp','Geq','r=1');

[GMg\_1,PMg,~,~]=margin(Gp);

GMg=20\*log10(GMg\_1);

%PMg=2.04 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(G);

GMcg=20\*log10(GMcg\_1);

%PMcg=78 deg

%GMcg=26.51 dB

t = out.zpc(:, 1);

y = out.zpc(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=2.225

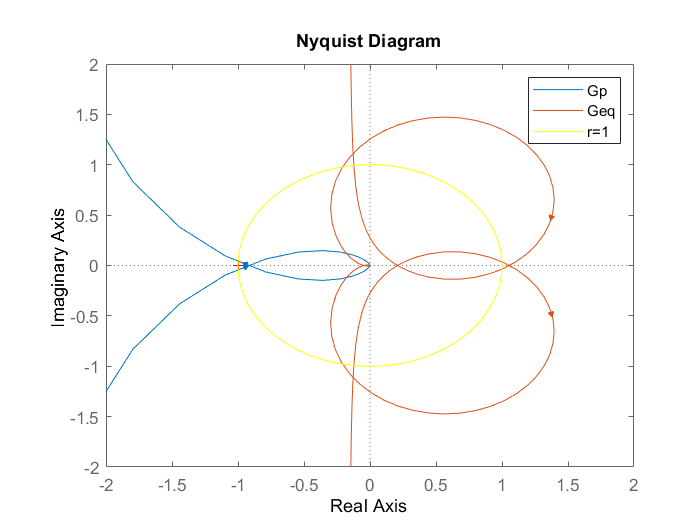
%Ts%2=2.78

%overshoot%=-

%d=-

%IAE=1.1838

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده بیشتر شده و به90 درجه نزدیک شده است(78) و همچنین بهره سیستم جبران شده برحسب dB خیلی بیشتر شده است(26.5).

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=2.225

Ts%2=2.78

overshoot%=-

d=-

IAE=1.1838



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 5 اعمال شده است):

Ts5%=2.43

Ts10%=2.32

overshoot%=143.5%

d=-

IAE=1.7669

t = out.zcw1(:, 1);

y = out.zcw1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-1.1838;

%Ts5%=2.43

%Ts10%=2.32

%overshoot%=143.5%

%d=-

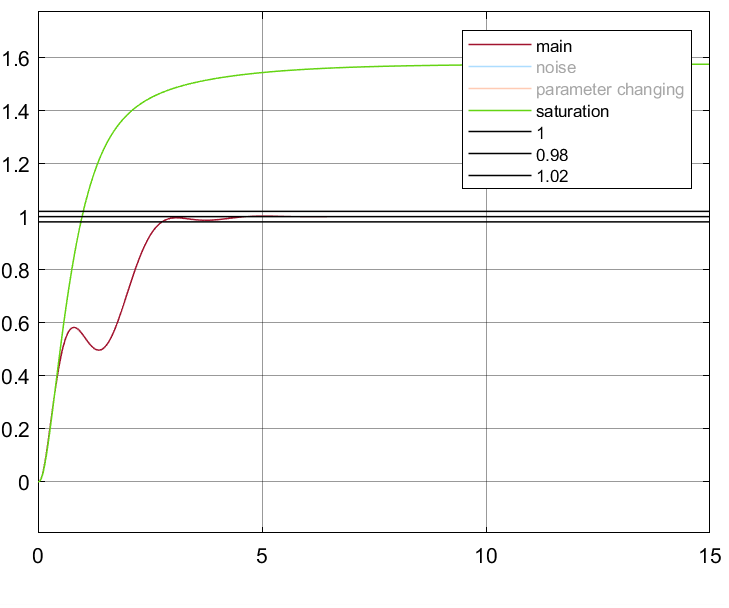
%IAE=1.7669



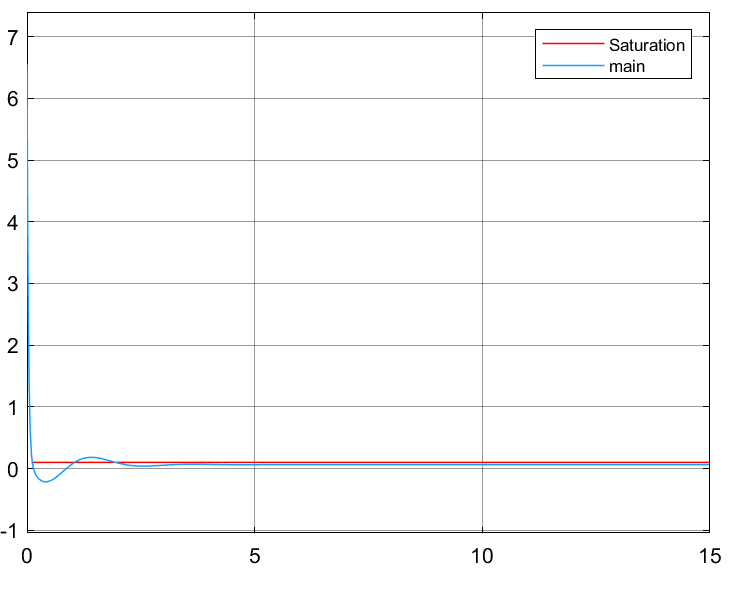
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:

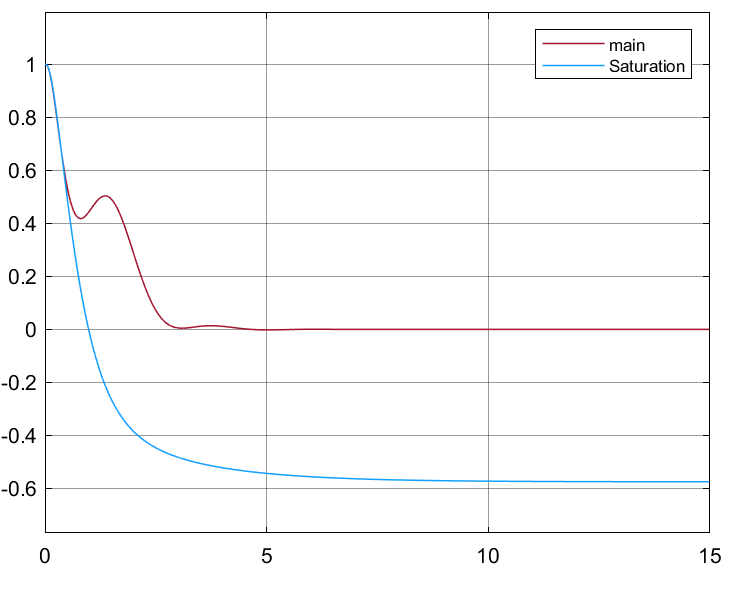


همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.

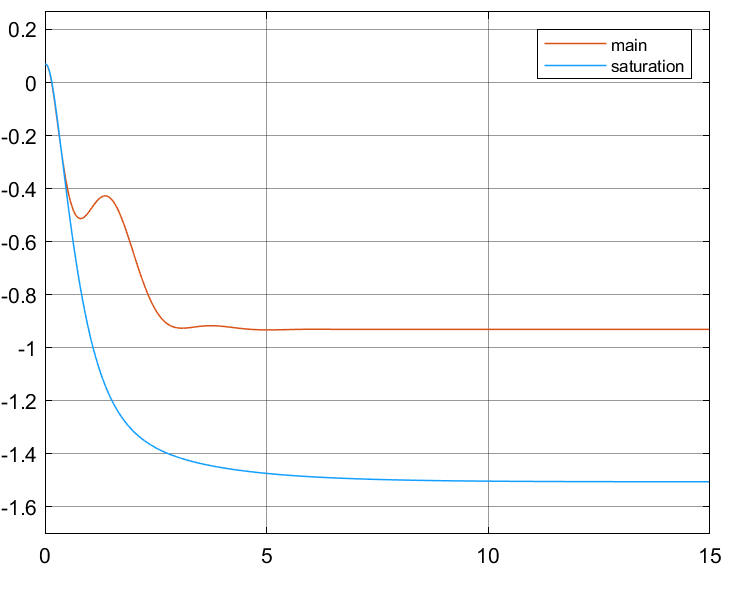


همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:

سیگنال خطای اعمالی به DI :



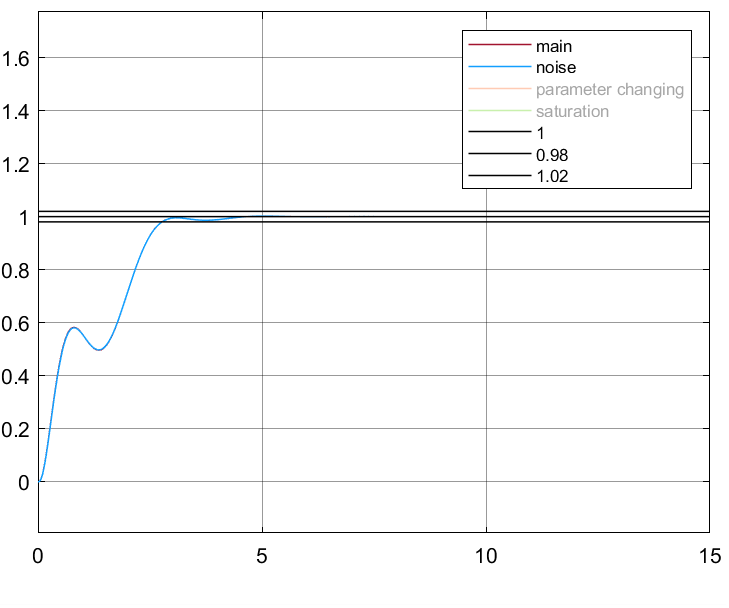
سیگنال خطای اعمالی به P:



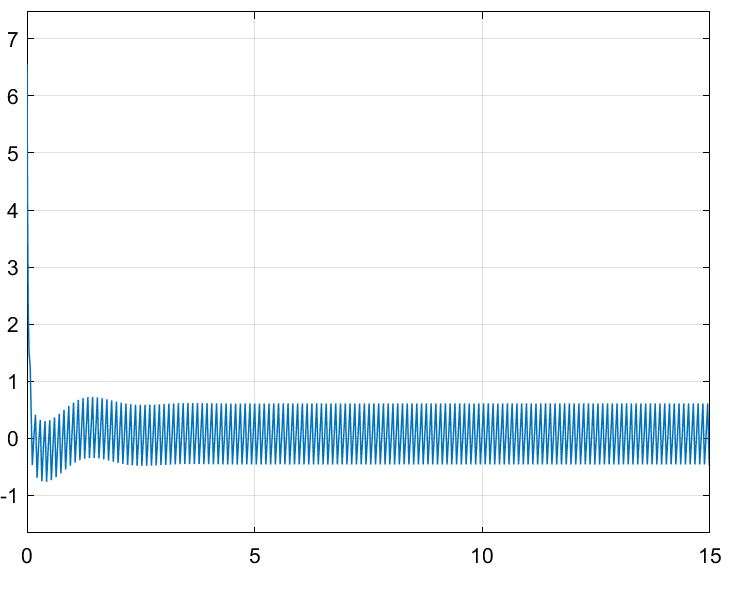
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.(البته خطای اعمالی به ترم P نیز به دلیل وجود ترم وزن دار نیز صفر نمی شود.)

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

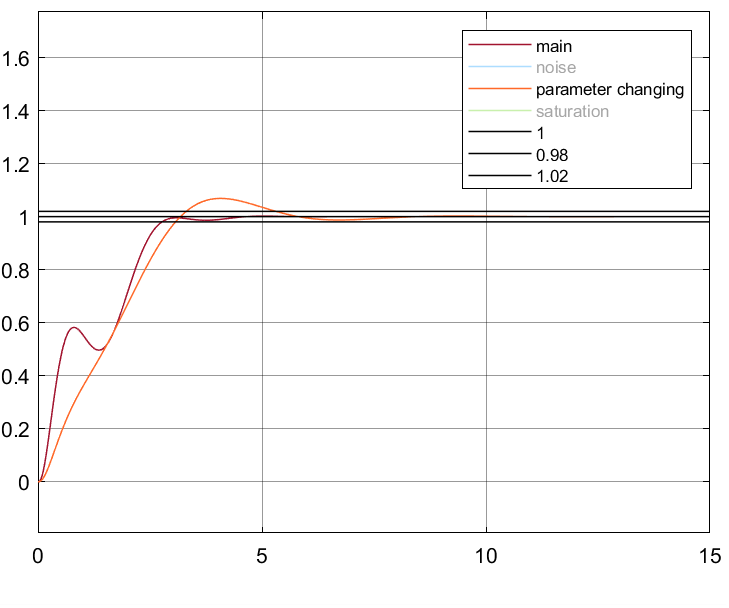


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثرعدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و کاهش ماکزیمم فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

## 3-2 به روش ZN نوسان میرا

در این روش ضریب kp را به گونه ای تغییر می دهیم که پاسخ پله با نسبت افت 0.25 به دست آید. با بدست آوردن این ضریب که نام آن را Kdmp و همچنین دوره تناوب این پاسخ را Tdmp گذاشته ایم.

که با Kdmp=0.545 داریم:



در ادامه روند رو مثل قبل طی می کنیم:

کد مربوطه:

%for finding kdmp(d=0.25) and Tdmp

n=1;

t = out.znn(:, 1);

y = out.znn(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

k=mean(y(end-n+1:end));

horizontal\_lines = [k];

colors = ['r'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

kdmp=0.545;

Tdmp=2.44;

%PID controler parameters

kp=1.1\*kdmp;

Ti=Tdmp/3.6;

Td=Tdmp/9;

%kp=0.5995

%Ti=0.6778

%Td=0.2711

s=tf('s');

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

g=63/((s+0.5)\*(s+2)\*(s+4));

hold on

nyquist(g)

nyquist(c\*g)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'r')

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.04 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=22.64 deg

%GMcg=20.39 dB

t = out.znn(:, 1);

y = out.znn(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.4

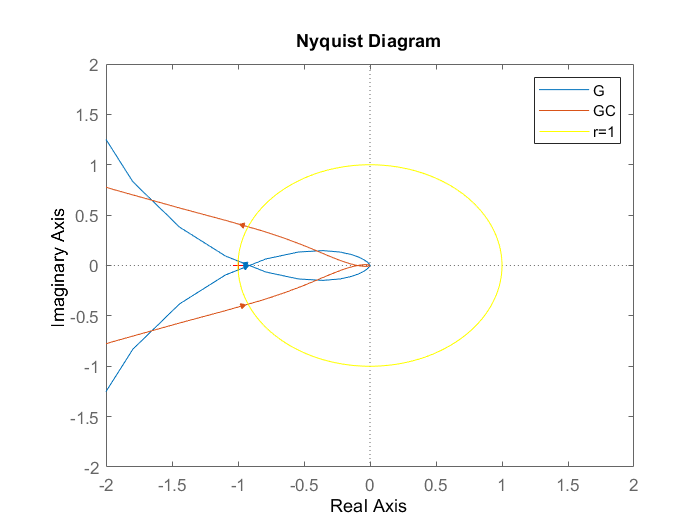
%Ts%2=3.53

%overshoot%=43.18%

%d=0.14

%IAE=0.8309

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده بیشتر شده و به حدود22.64 درجه نزدیک شده استو همچنین بهره سیستم جبران شده برحسب dB خیلی بیشتر شده است(20.39).

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.4

Ts%2=3.53

overshoot%=43.18%

d=0.14

IAE=0.8309



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 7 اعمال شده است):

Ts5%=4.22

Ts10%=3.95

overshoot%=157%

d=-

IAE=2.4369

t = out.znn1(:, 1);

y = out.znn1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-0.8309;

%Ts5%=4.22

%Ts10%=3.95

%overshoot%=157%

%d=-

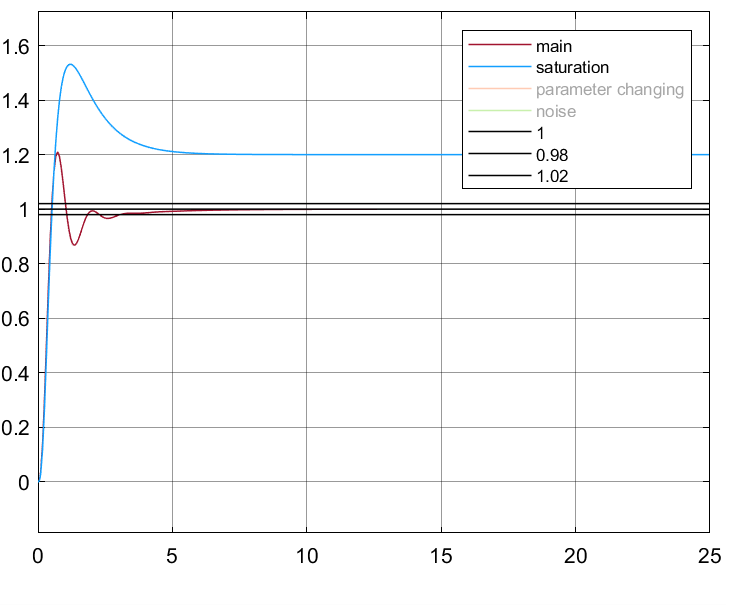
%IAE=2.4369



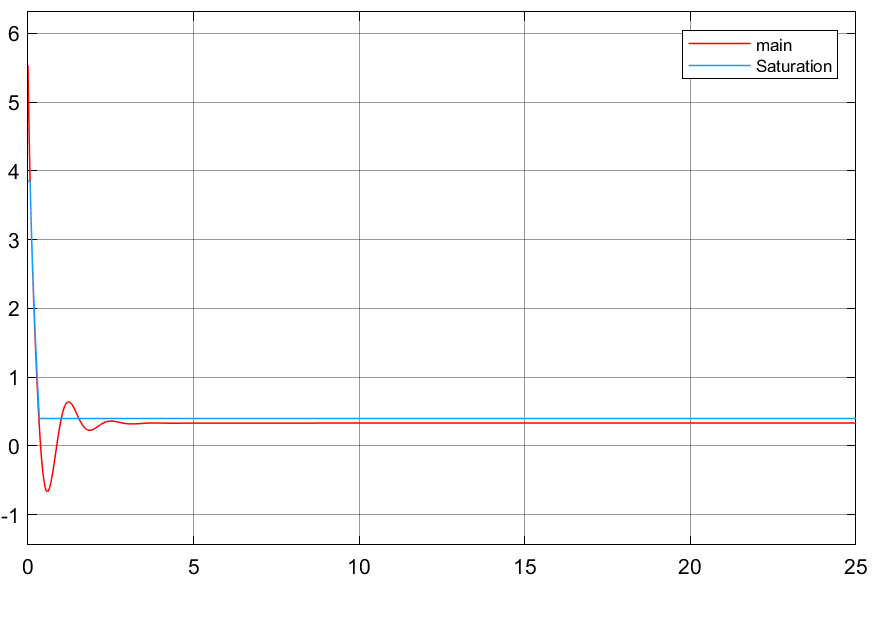
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

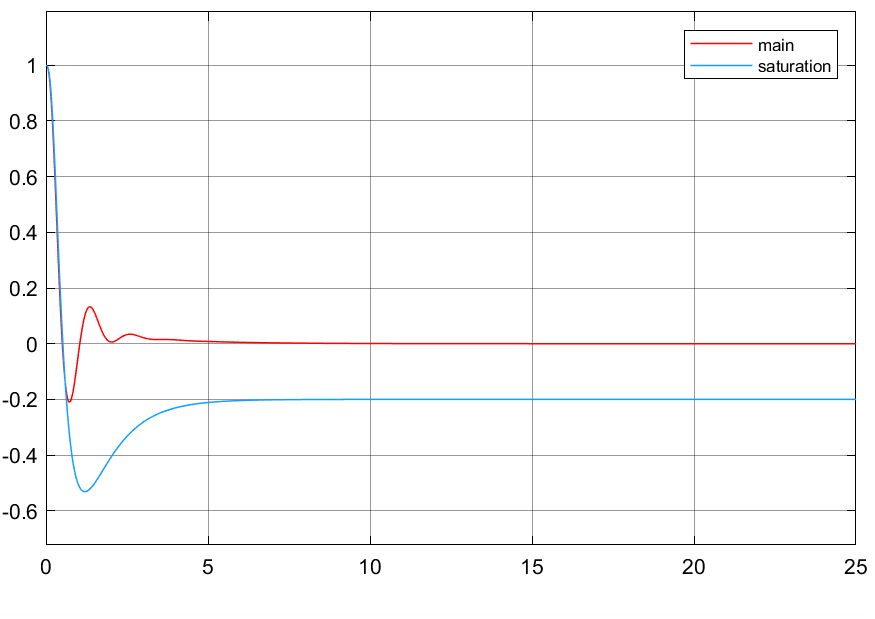
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



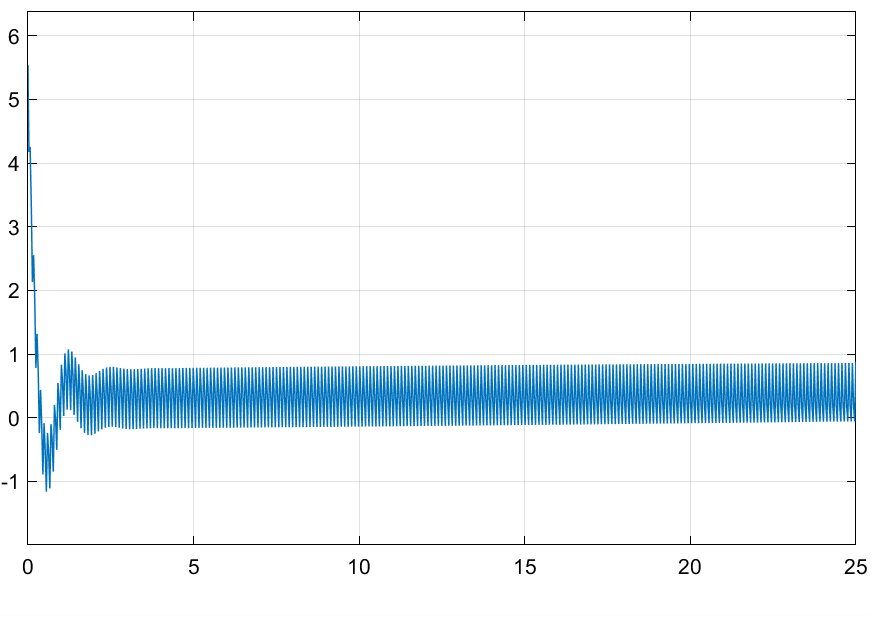
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

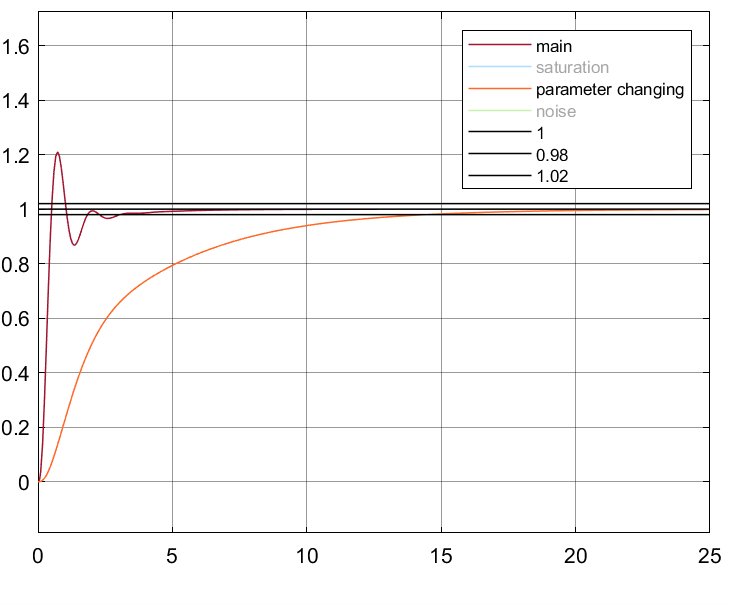


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و با از بین رفتن فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

# بخش سوم

# طراحی PID بر اساس حد بهره و حد فاز-ZN تعمیم یافته

## 1-3دستیابی به حد بهره 6 dB

با استخراج معادلات لازم بای طراحی PID بر اساس دست یابی به حد بهره 6 dB ، معادلات را اینگونه داریم که:

نکته: با توجه به اینکه کنترل کننده PID که به صورت  می باشد اما معادلات از روی فرم ایده آل به دست آمده است ()اما تقریبا می توان گفت مقادیر بدست آمده برای PID از روی هردوی معادلات هیچ تفاوتی وجود ندارد.

معدلات به دست آمده از روی :







معدلات به دست آمده از روی:







که در آن w فرکانس برخورد دیاگرام نایکوییست سیستم جبران نشده با محور حقیقی و rA محل برخورد دیاگرام نایکوییست سیستم جبران نشده با محور حقیقی و rB محل برخورد دیگرام نایکوییست سیستم جبران شده با محور حقیقی می باشد.

همانطور که در کد هم یک نکته به صورت کامنت آمده است که با kp بدست آمده از این روابط به حد بهره مورد نظر نمیرسیم و با سعی و خطا و تغییر آن به این امر دست می یابیم.

داریم:

rA=0.914;

rB=0.5012; %6dB

w=3.37;

%finding kp & Ti & Tp for PID controler

kp=rB/rA;

Td=1/(2\*(w));

Ti=2/w;

%kp=0.5484

%Td=0.1484

%Ti=0.5935

%with kp=5484 we can achieve gain margin=6 dB so by chinging kp=0.39 we can

kp=0.39;

s=tf('s');

g=63/((s+0.5)\*(s+2)\*(s+4));

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

cg=c\*g;

hold on

nyquist(g)

nyquist(cg)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'r')

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.04 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=3.52 deg

%GMcg=6.05 dB

t = out.g(:, 1);

y = out.g(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.61

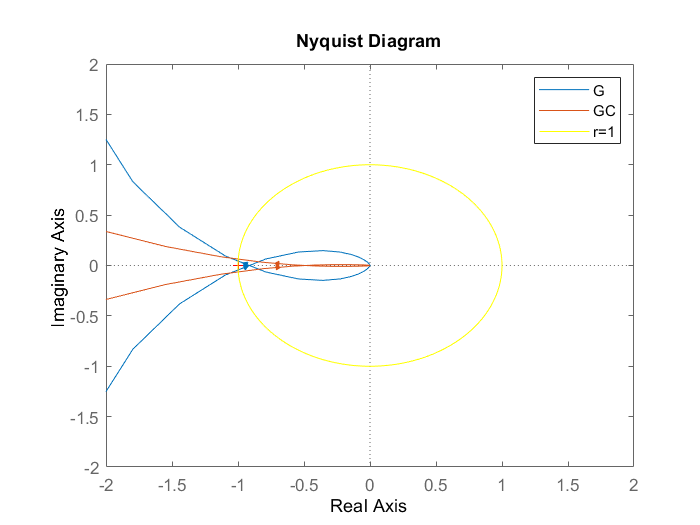
%Ts%2=5.91

%overshoot%=41.25%

%d=0.14

%IAE=1.1894

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده تقریبا ثابت مانده است و برابر3.52درجه استو همچنین بهره سیستم جبران شده مورد نظر که برای دست یابی به آن تلاش شده است برابر با6 dB است که تقریبا برابر با 6 dB مورد نظر است.

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.61

Ts%2=5.91

overshoot%=41.25%

d=0.14

IAE=1.1894



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 7 اعمال شده است):

Ts5%=5.63

Ts10%=5.34

overshoot%=247%

d=0.13

IAE=4.26

t = out.g1(:, 1);

y = out.g1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-1.1894;

%Ts5%=5.63

%Ts10%=5.34

%overshoot%=247%

%d=0.13

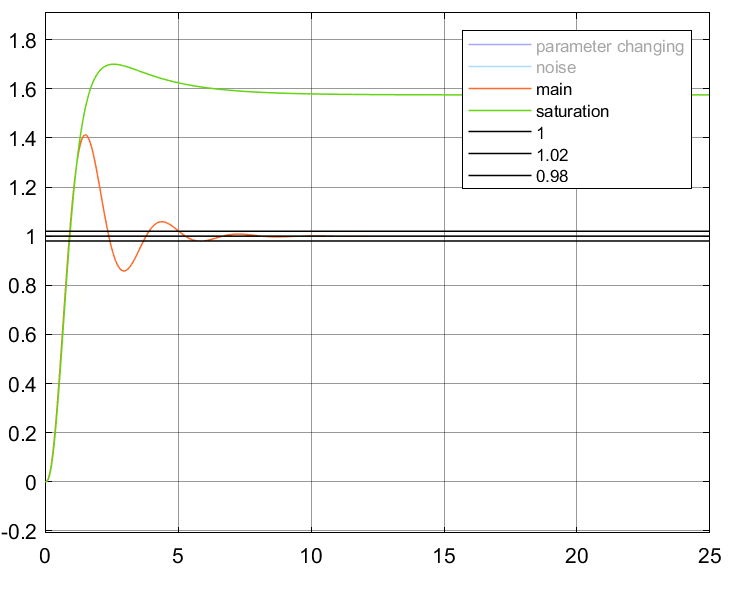
%IAE=4.26



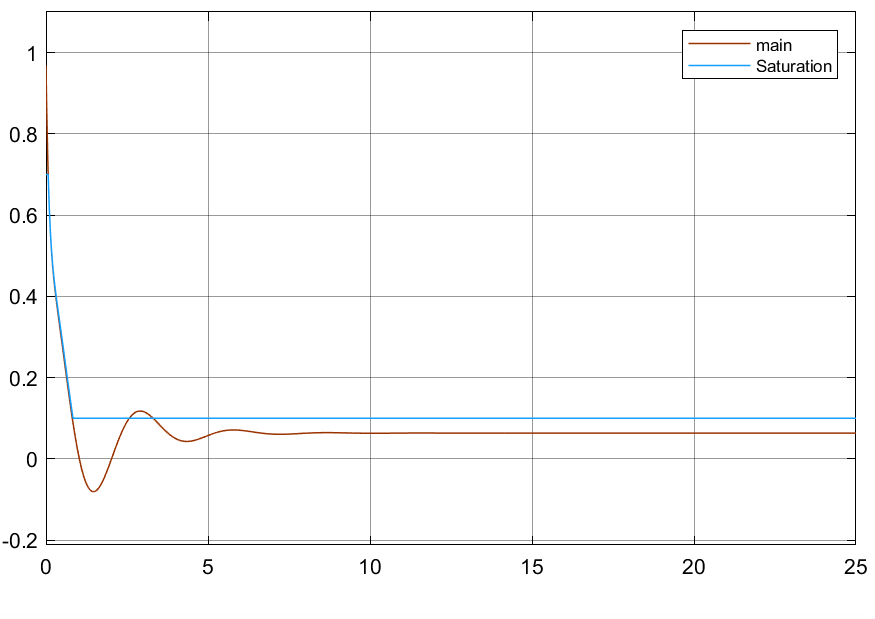
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

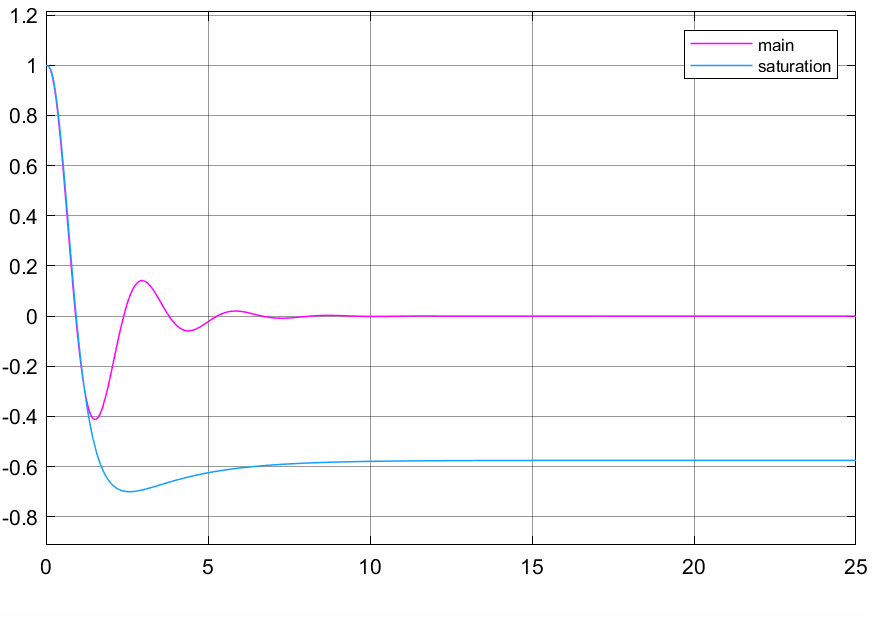
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



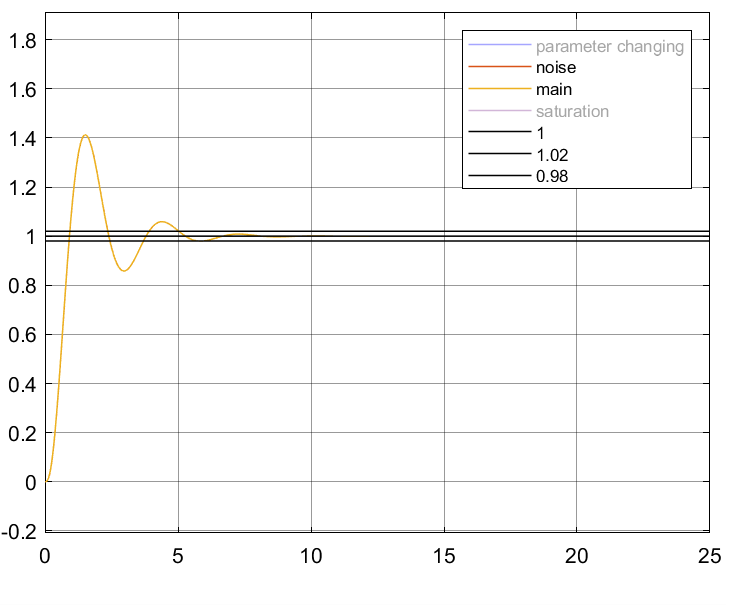
همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



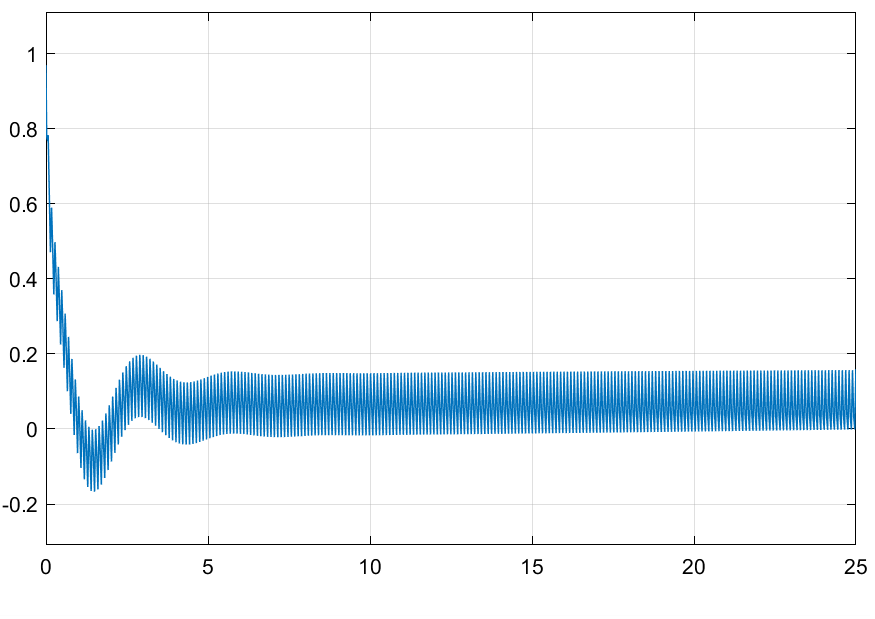
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

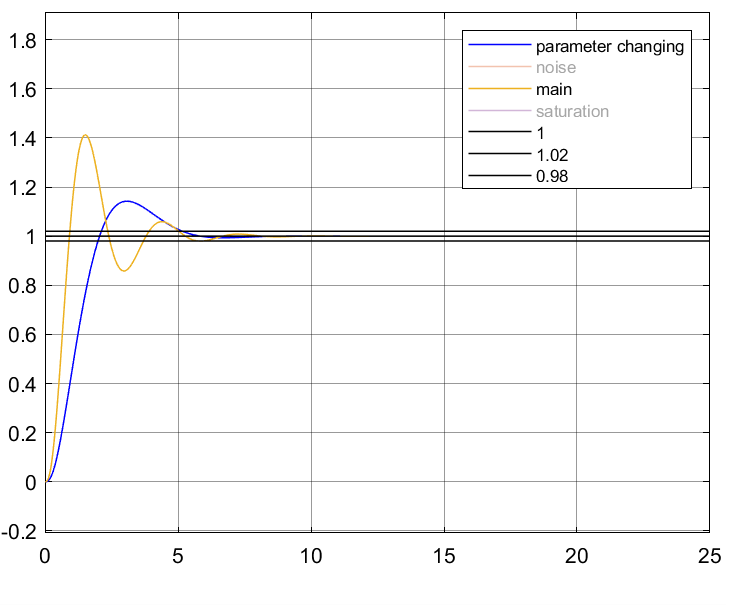


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و با از بین رفتن فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

## 2-3 دستيابي به حد فازdeg 45

با استخراج معادلات لازم بای طراحی PID بر اساس دست یابی به حد فاز 45 deg ، معادلات را اینگونه داریم که:

نکته: با توجه به اینکه کنترل کننده PID که به صورت  می باشد اما معادلات از روی فرم ایده آل به دست آمده است ()اما تقریبا می توان گفت مقادیر بدست آمده برای PID از روی هردوی معادلات هیچ تفاوتی وجود ندارد.

معدلات به دست آمده از روی :







معدلات به دست آمده از روی:







که در آن w فرکانس برخورد دیاگرام نایکوییست سیستم جبران نشده با دايره به شعاع 1و زاويه بردار محل برخورد دیاگرام نایکوییست سیستم جبران نشده با دايره به شعاع 1 و زاويه بردار محل برخورد دیاگرام نایکوییست سیستم جبران شده با دايره به شعاع 1مي باشد.

همانطور که در کد هم یک نکته به صورت کامنت آمده است که با kp بدست آمده از این روابط به حد فاز مورد نظر نمیرسیم و با سعی و خطا و تغییر آن به این امر دست می یابیم.

در ادامه داريم:

%point of touch Re=-0.45 & Im=-0.893

QA=2.04;

QB=45;

w=3.23;

%finding kp & Ti & Tp for PID controller

kp=cosd(QB-QA);

Td=(tand(QB-QA)/(2\*w))+((sqrt(((tand(QB-QA)/w)^2)+(1/(w^2))))/2);

Ti=4\*Td;

%kp=0.7318

%Td=0.3515

%Ti=1.4061

%with this kp we cant achieve phase margin=45 deg so by kp=0.54704 we can

kp=0.54704;

s=tf('s');

g=63/((s+0.5)\*(s+2)\*(s+4));

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

cg=c\*g;

hold on

nyquist(g)

nyquist(cg)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'r')

plot(x,y,'y');

legend('G','GC','r=1')

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.04 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=45 deg

%GMcg=19.39 dB

t = out.p(:, 1);

y = out.p(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth',1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.4

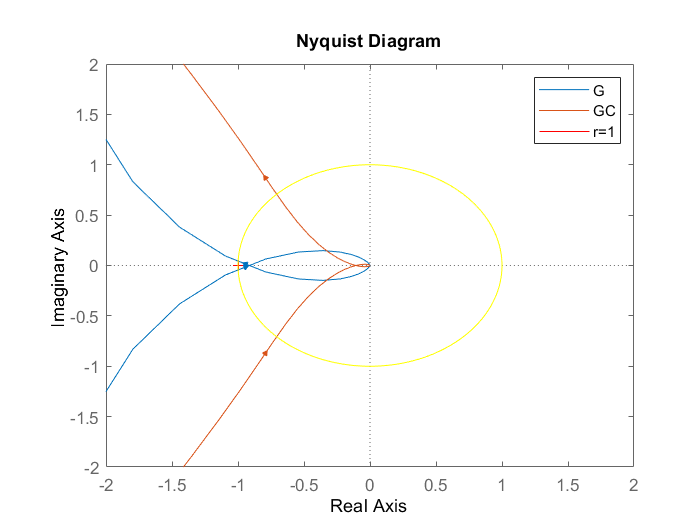
%Ts%2=2.99

%overshoot%=54%

%d=0.04

%IAE=0.9259

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور هم که در کد نشان داده شده فاز سیستم جبران شده که براي آن تلاش شده است برابر با 45 درجه مي باشد و همچنین بهره سیستم جبران شده برابر با 19.38 dB است که نسب به سيستم جبران نشده خيلي بيشتر شده است.

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.4

Ts%2=2.99

overshoot%=54%

d=0.04

IAE=0.9259



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 7 اعمال شده است):

Ts5%=3.9

Ts10%=3.29

overshoot%=151%

d=0.013

IAE=1.9568

t = out.p1(:, 1);

y = out.p1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-0.9259;

%Ts5%=3.9

%Ts10%=3.29

%overshoot%=151%

%d=0.013

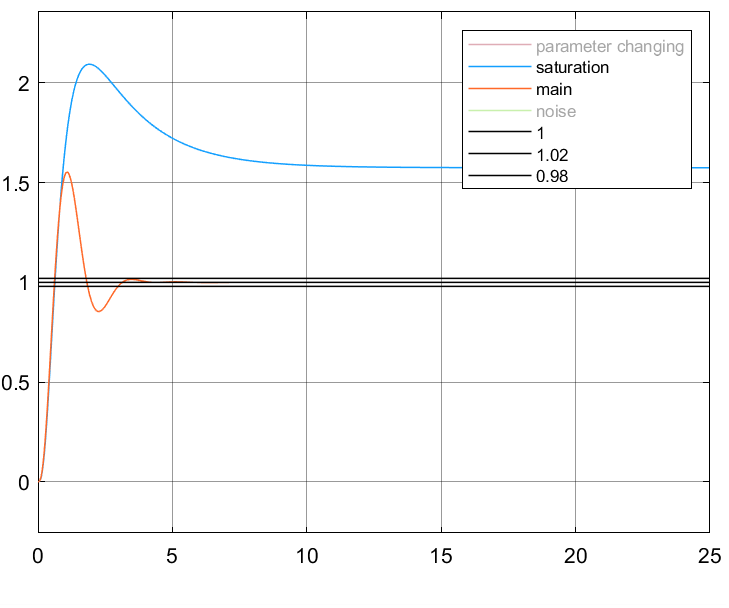
%IAE=1.9568



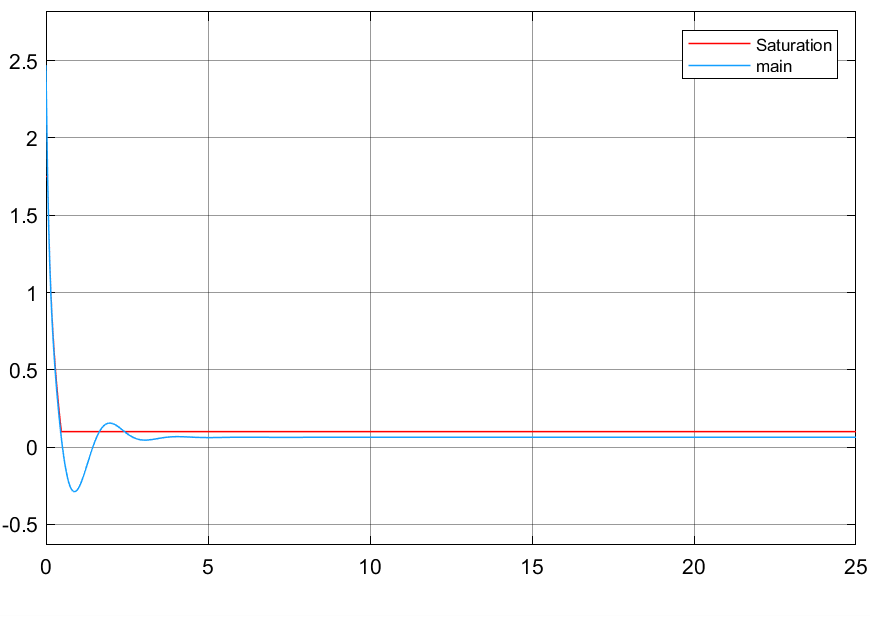
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

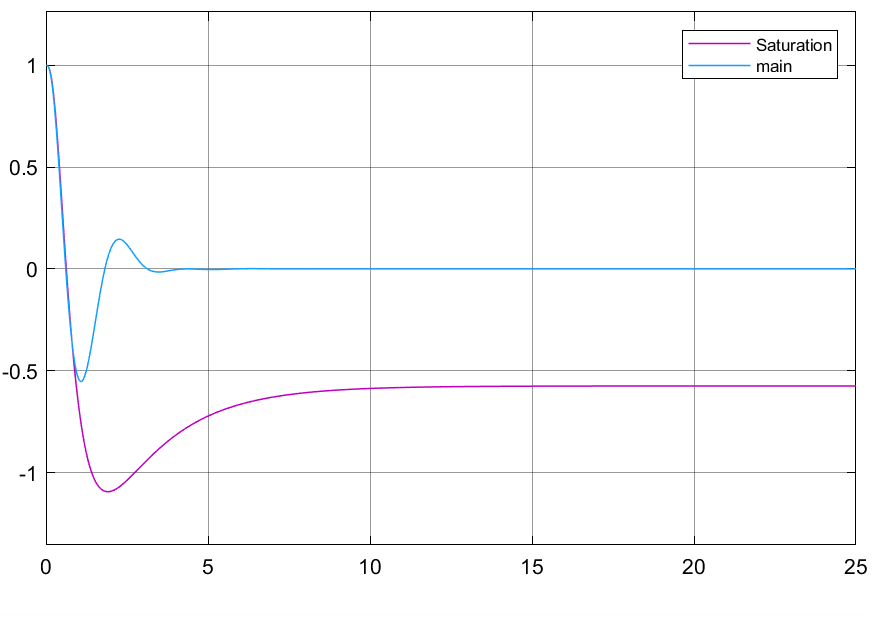
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



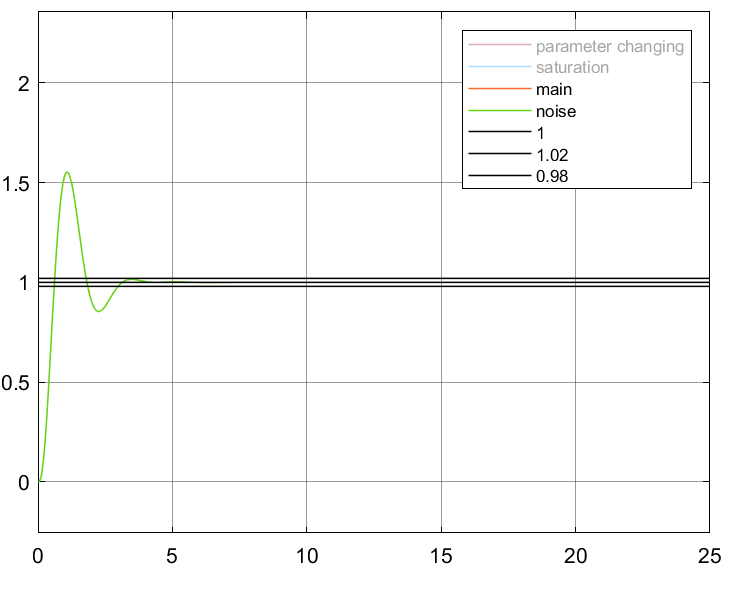
همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



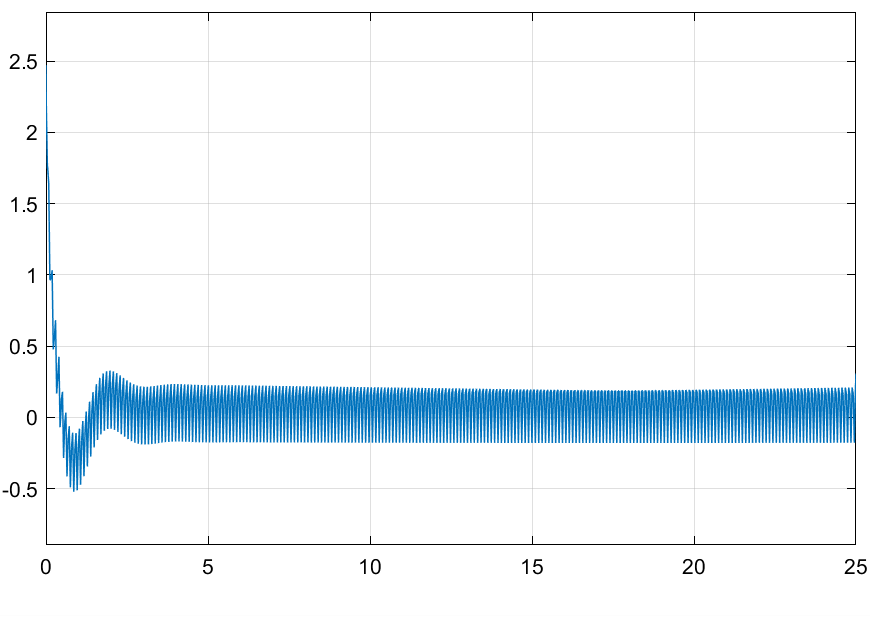
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

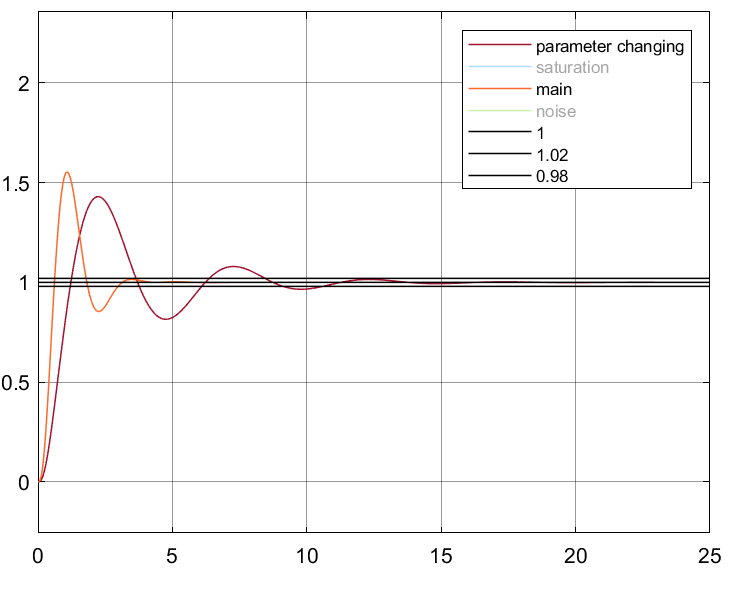


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و با از بین رفتن فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

## 3-3 بهینه سازی برای دست یابی به حد فاز 45 deg و حد بهره 6 dB

در این بخش تلاش این بوده است که با بهینه سازی هرودی فاز 45 deg و حد بهره ی 6 dB را به طور همزمان داشته باشیم. به دلیل اینکه این دو مورد در فرکانس های متفاوت صورت می گیرند رسیدن دقیق به این دو مقدار ممکن نمی باشد و تلاش این است با توابع خطا و توابع بهینه سازی به این دو مقدار نزدیک شویم(به دلیل ضعیف بودن cpu سیستم توابعی همچون ژنتیک را نتوانستم run بگیرم). در این بخش با کد اعمالی و با سعی و خطا و انتخاب بازه های منااسب برای رسیدن به این دو هدف ثمره ی حد فاز 37.83 deg و حد بهره 7.4541 dB صورت پذیرفته است. در ادامه کد بهینه سازی را داریم:

% Desired Gain Margin and Phase Margin

GM\_desired = 2; % Desired Gain Margin in 1/|gc(jw)|

PM\_desired = 45; % Desired Phase Margin in degrees

% Optimization bounds

lb = [0.001, 0.001, 0.001]; % Lower bounds for Kp, Ti, Td

ub = [10, 10, 10]; % Upper bounds for Kp, Ti, Td

% Optimization options using Particle Swarm Optimization (PSO)

options = optimoptions('particleswarm', 'SwarmSize', 50, 'MaxIter', 100, 'UseParallel', true);

% Optimization function

opt\_func = @(x) cost\_function(x(1), x(2), x(3), GM\_desired, PM\_desired);

% PSO optimization

[x\_opt, fval\_opt] = particleswarm(opt\_func, 3, lb, ub, options);

% Display optimized parameters and results

disp(['Optimized Kp: ', num2str(x\_opt(1))]);

disp(['Optimized Ti: ', num2str(x\_opt(2))]);

disp(['Optimized Td: ', num2str(x\_opt(3))]);

% Compute Gain Margin and Phase Margin with optimized parameters

[GM\_opt, PM\_opt] = compute\_margins(x\_opt(1), x\_opt(2), x\_opt(3));

disp(['Optimized Gain Margin (1/|gc(jw)|): ', num2str(GM\_opt)]);

disp(['Optimized Phase Margin (degrees): ', num2str(PM\_opt)]);

disp(['GM Error: ', num2str(abs(GM\_opt - GM\_desired))]);

disp(['PM Error: ', num2str(abs(PM\_opt - PM\_desired))]);

% Define the optimized PID controller

s = tf('s');

PID\_opt = x\_opt(1) + (x\_opt(1) / (x\_opt(2) \* s)) + ((x\_opt(1) \* x\_opt(3) \* s) / (((x\_opt(3) / 10) \* s) + 1));

% Closed-loop transfer function

G = 63 / ((s + 0.5) \* (s + 2) \* (s + 4)); % Example plant

L\_opt = PID\_opt \* G;

% Local Functions

function cost = cost\_function(Kp,Ti,Td, GM\_desired, PM\_desired)

% Compute Gain Margin and Phase Margin

[GM, PM] = compute\_margins(Kp,Ti,Td);

% Compute normalized errors

GM\_error = abs((GM - GM\_desired) / GM\_desired);

PM\_error = abs((PM - PM\_desired) / PM\_desired);

% Total cost

cost = GM\_error^2 + PM\_error^2;

end

function [GM, PM] = compute\_margins(Kp,Ti,Td)

% Define the plant (example system)

s = tf('s');

G = 63 / ((s + 0.5) \* (s + 2) \* (s + 4)); % Example plant

% Define PID controller

PID = Kp + (Kp / (Ti \* s)) + ((Kp \* Td \* s) / (((Td / 10) \* s) + 1));

% Open-loop transfer function

L = PID \* G;

% Compute Gain Margin and Phase Margin

[GM, PM,~,~] = margin(L);

end

در ادامه روند را مثل قبل ادامه می دهیم:

kp= 0.14267;

Ti= 0.14637;

Td=2.7491 ;

s=tf('s');

g=63/((s+0.5)\*(s+2)\*(s+4));

c=kp+(kp/(Ti\*s))+((kp\*Td\*s)/(((Td/10)\*s)+1));

cg=c\*g;

hold on

nyquist(g)

nyquist(cg)

r=1;

for j=0:360

x(j+1)=r\*cos((pi/180)\*j);

y(j+1)=r\*sin((pi/180)\*j);

end

hold on

axis([-2 2 -2 2])

plot(x,y,'r')

plot(x,y,'y');

legend('G','GC','r=1')

[GMg\_1,PMg,~,~]=margin(g);

GMg=20\*log10(GMg\_1);

%PMg=2.04 deg

%GMg=0.6 dB

[GMcg\_1,PMcg,~,~]=margin(c\*g);

GMcg=20\*log10(GMcg\_1);

%PMcg=37.83 deg

%GMcg=7.4541 dB

t = out.gp(:, 1);

y = out.gp(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.02, 0.98, 0.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

%Tr=0.83

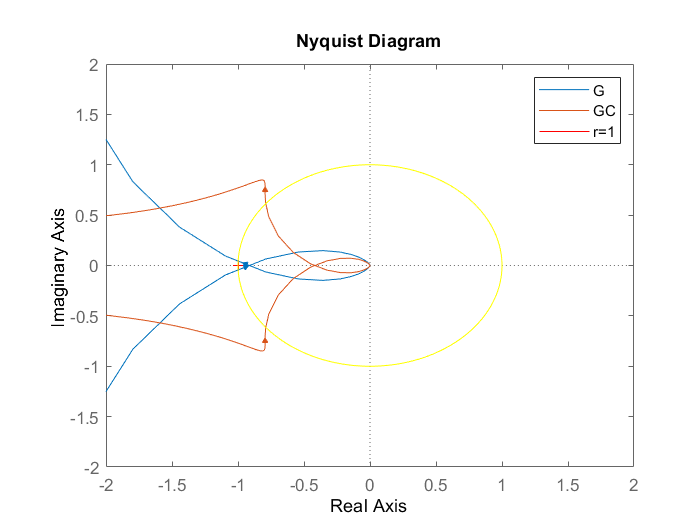
%Ts%2=4.32

%overshoot%=-

%d=-

%IAE=3.05

در ابتدای کد با داشتن مقادیر سیستم مدل شده شروع به محاسبه ی ضرایب PID می کنیم که با مقایسه ی دایاگرام های نایکوِییست داریم که:



همانطور که مشخص است فاز سیستم جبران شده به 45 درجه نزدیک شده و حد بهره ی سیستم جبران شده هم نیز به به بهره ی 6 دسی بل بسیار نزدیک شده که این نشان دهنده ی عملکرد بسیار خوبه بهینه سازی و ضرایب تعیین شده برای PID می باشد.

در ادامه مشخصات پاسخ پله سیستم حلقه بسته به دست آمده است که همه ی موارد به صورت کامنت در کد آمده است که در پایان گزارش کار برای همه ی طراحی ها در یک جدول این موارد آورده می شود. مشخصات گفته شده از روی این نمودار بدست آمده است:

Tr=0.83

Ts%2=4.32

overshoot%=-

d=-

IAE=3.05



در ادامه کد مربوط به استخراج مشخصات پاسخ سیستم به ورودی اغتشاش آورده می شود که مشخصات استخراج شده به صورت کامنت آورده شده و برای همه ی طراحی ها این مشخصات در جدول انتهای گزارشکار آورده می شود. نمودار مربوط به مشخصات استخراج شده برای ورودی اغتشاش(اغتشاش در ثانیه ی 7 اعمال شده است):

Ts5%=34.22

Ts10%=30.82

overshoot%=355%

d=-

IAE=47.81

t = out.pg1(:, 1);

y = out.pg1(:, 2);

plot(t, y, '-b', 'LineWidth', 1.5);

hold on;

horizontal\_lines = [1.05, 0.95, 1.1, 0.9];

colors = ['r', 'g', 'm', 'k'];

% finding and showing points

for i = 1:length(horizontal\_lines)

line\_value = horizontal\_lines(i);

%finding points

idx = find((y(1:end-1) - line\_value) .\* (y(2:end) - line\_value) <= 0);

% calculating touching points

t\_intersect = t(idx) + (t(idx+1) - t(idx)) .\* ...

(line\_value - y(idx)) ./ (y(idx+1) - y(idx));

y\_intersect = line\_value \* ones(size(t\_intersect)); % value of output in touching points

%drawing lines

plot(t, line\_value \* ones(size(t)), '--', 'Color', colors(i), 'LineWidth', 1.2);

% showing touching points

plot(t\_intersect, y\_intersect, 'o', 'MarkerSize', 8, ...

'MarkerFaceColor', colors(i), 'MarkerEdgeColor', 'k');

end

xlabel('Time (s)');

ylabel('Output');

title('Curve and Intersection Points with Horizontal Lines');

legend\_entries = [{'Original Curve'}, ...

arrayfun(@(x) sprintf('Line = %.2f', x), horizontal\_lines, 'UniformOutput', false)];

legend(legend\_entries{:});

grid on;

hold off;

%calculating IAE

error=1-y;

absIntegralError = trapz(t, abs(error));

IAE=absIntegralError-3.05;

%Ts5%=34.22

%Ts10%=30.82

%overshoot%=355%

%d=-

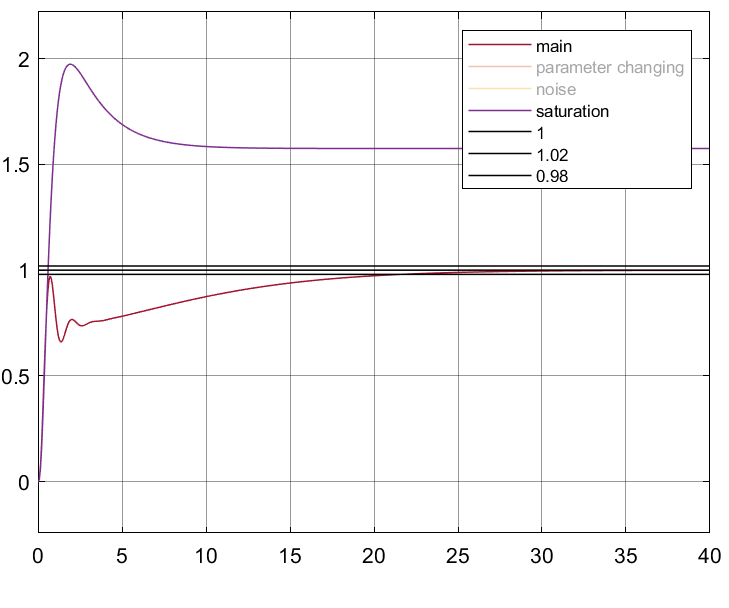
%IAE=47.81



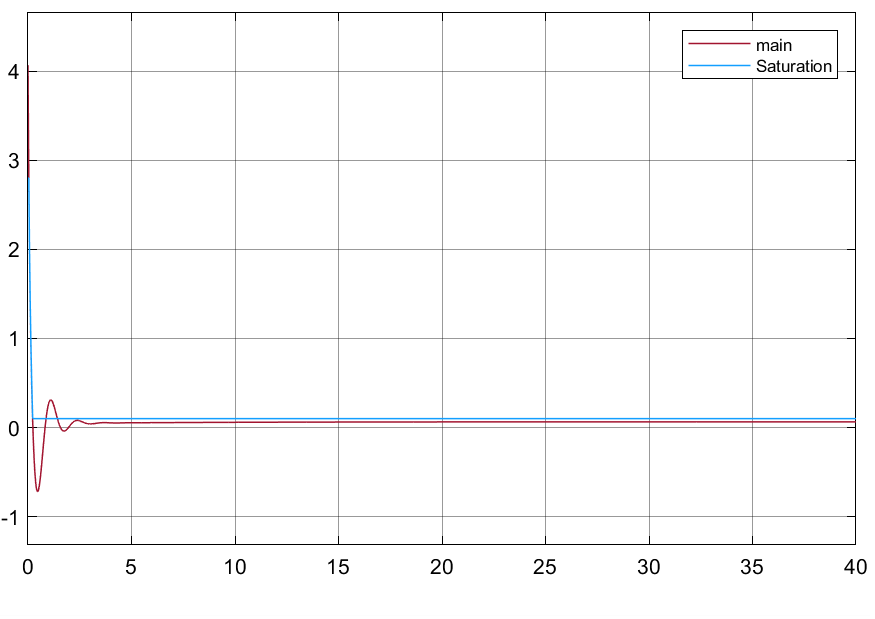
در ادامه ی این بخش خرجی سیستم حلقه بسته در شرایط مختلف نشان داده می شود(نویز ، تغییر پارامترو...):

**اشباع محرک**:

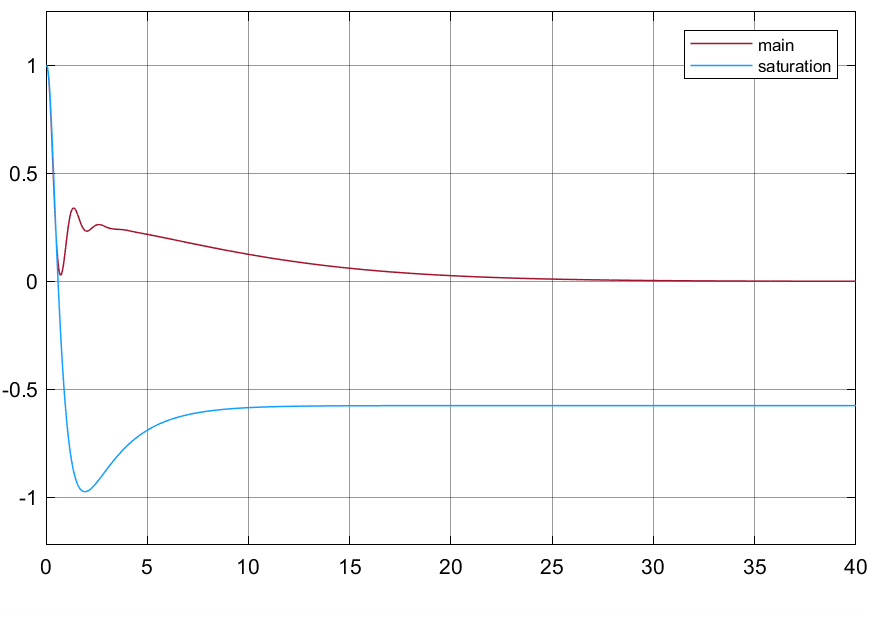
در این بخش به مقایسه ی خروجی ها بر اثر اشباع و خروجی اصلی می پردازیم:



همانطور که مشخص مي باشد افزودن بلوک اشباع جوري که سيگنال فرمان به اشباع رود باعث مي شود که سيستم از سيگنال فرمان ، فرمان نپذيرد و باعث انباشته شدن خطا شده است(خطا صفر نمي شود). در ادامه به مقايسه ي سيگنال هاي فرمان ميپردازيم که نشان دهنده ي نوع عملکرد بلوک اشباع مي باشد.



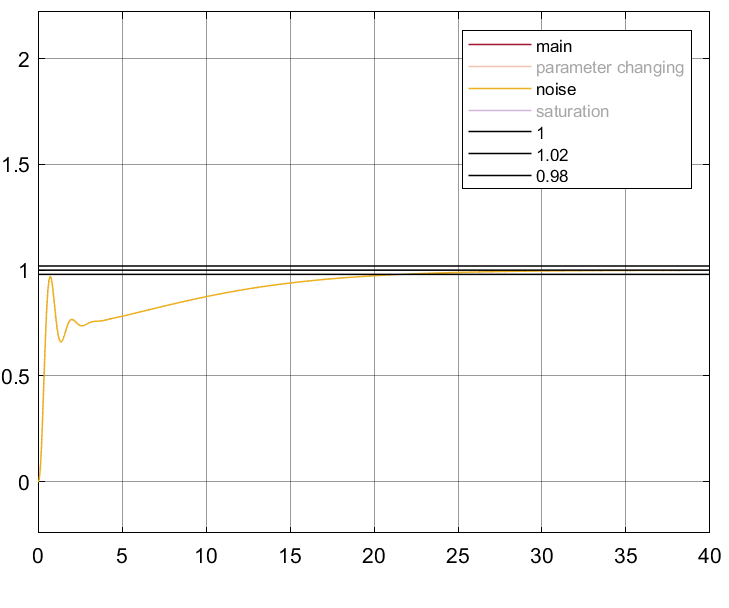
همانطور که قابل ملاحظه می باشد با توجه به بلوک اشباع سیگنال فرمان به اشباع رفته باعث انباشته شدن خطا می شود ادامه با مقایسه ی سیگنال های خطا داریم:



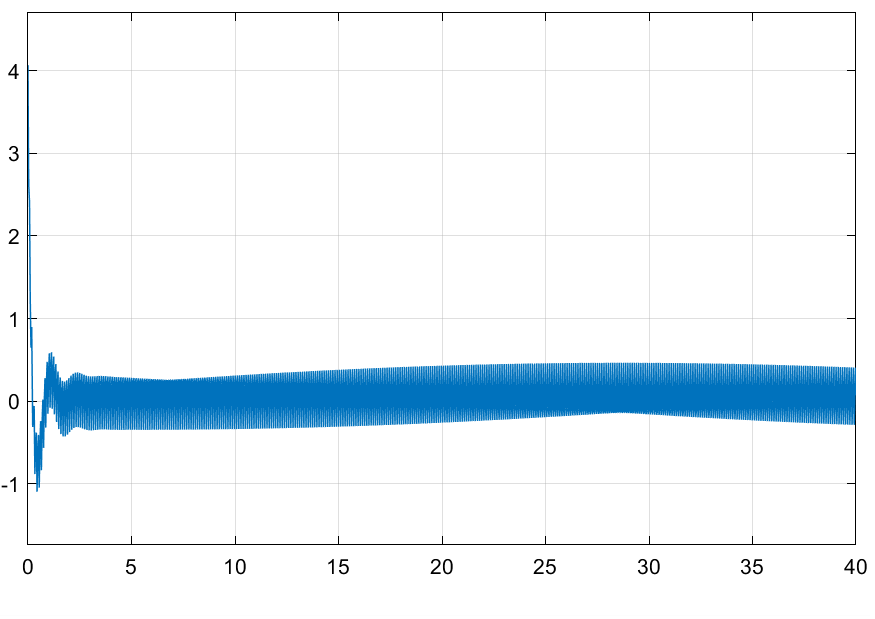
همانطور که قابل مشاهده است با مقایسه ی سیگنال های خطا میتوان متوجه شد که به دلیل وجود بلوک اشباع خطا انباشته شده صفر نمی شود.

**اثر نویز:**

با مقایسه ی خروجی سیستم با اثر نویز و بدون اثر نویز داریم:

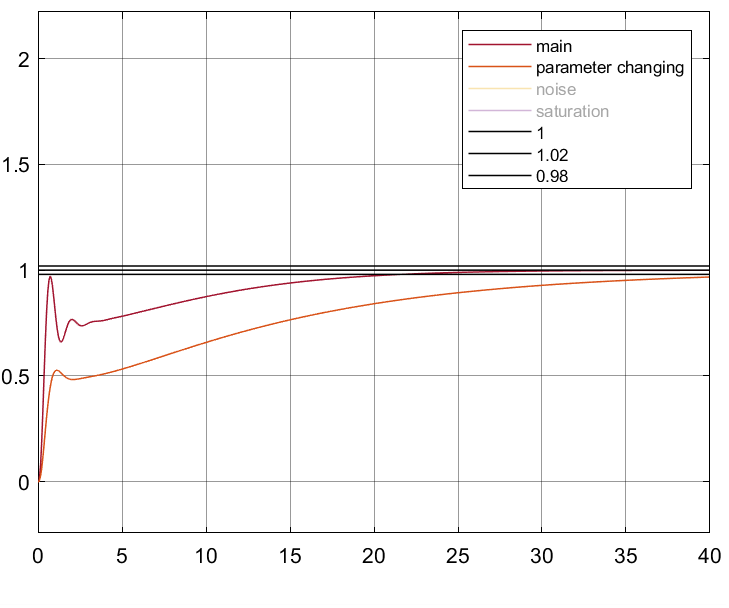


همانطور که مشخص است هر دو روی یکدیگر افتاده اند و این مورد به دلیل این می باشد که نویز فرکانس بالا می باشد و سیستم مورد نظر فرکانس پایین و نوییز تاثیری ندارد روی سیستم و میتوان گفت که تاثیر بسیار کمی دارد. می دانیم که اثر نویز با بهره ی کنترل کننده رابطه مستقیم دارد و این رابطه مستقیم تاثیر زیادی روی سیگنال فرمان می گذرد(به خصوص وجود ترم مشتق گیر و استفاده فیلتر پایین گذر در آن). این مورد را می توان از روی سیگنال فرمان که در ادامه نشان داده میشود متوجه شد:



**اثر عدم قطعیت:**

در ادامه با مقایسه خروجی ها داریم:



تغییر پارامتر مشخصا باعث ایجاد تغییر در پاسخ گذرا که افزایش زمان خیزش و با از بین رفتن فراجهش و همچنین افزایش زمان نشست می شود ولی با وجود فیدبک و کنترل کننده به حالت ماندگار می رسد.

# بخش چهارم

# بررسی و مقایسه

## 1-4 بررسی کلی تاثیرات عوامل خارجی بر پاسخ

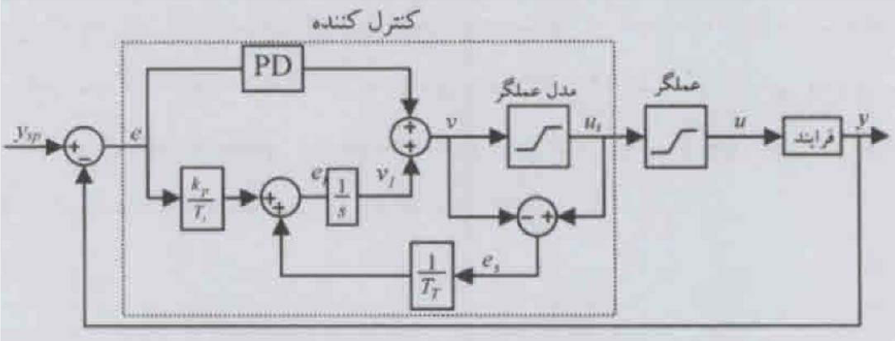
**نویز**:

با توجه به خروجی های سیستم که در بخش های قبل نشان داده شد کاملا واضح بود به دلیل اینکه نویز فرکانس بالا می باشد و سیستم خود فرکانس پایین می باشد ، نویز تاثیر چندانی برروی پاسخ خروجی ندارد مگر اینکه توسط کنترل کننده آنچنان تقویت شود که برروی خروجی تاثیر بگذارد که تقویت فرکانس رابطه ی مستقیم با بهره کنترل کننده دارد که برای PID بهره ی کنترل کننده در فرکانس بی نهایت برابر با Kp(1+N) می باشد که از خروجی سیگنال ها فرمان همگی بخش ها کاملا مشخص است که هرکدام ضریب KP بزرگ تری دارند ،؟ نویز اثر بیشتری بر سیگنال فرمان می گذارد.

**اشباع سیگنال تحریک:**

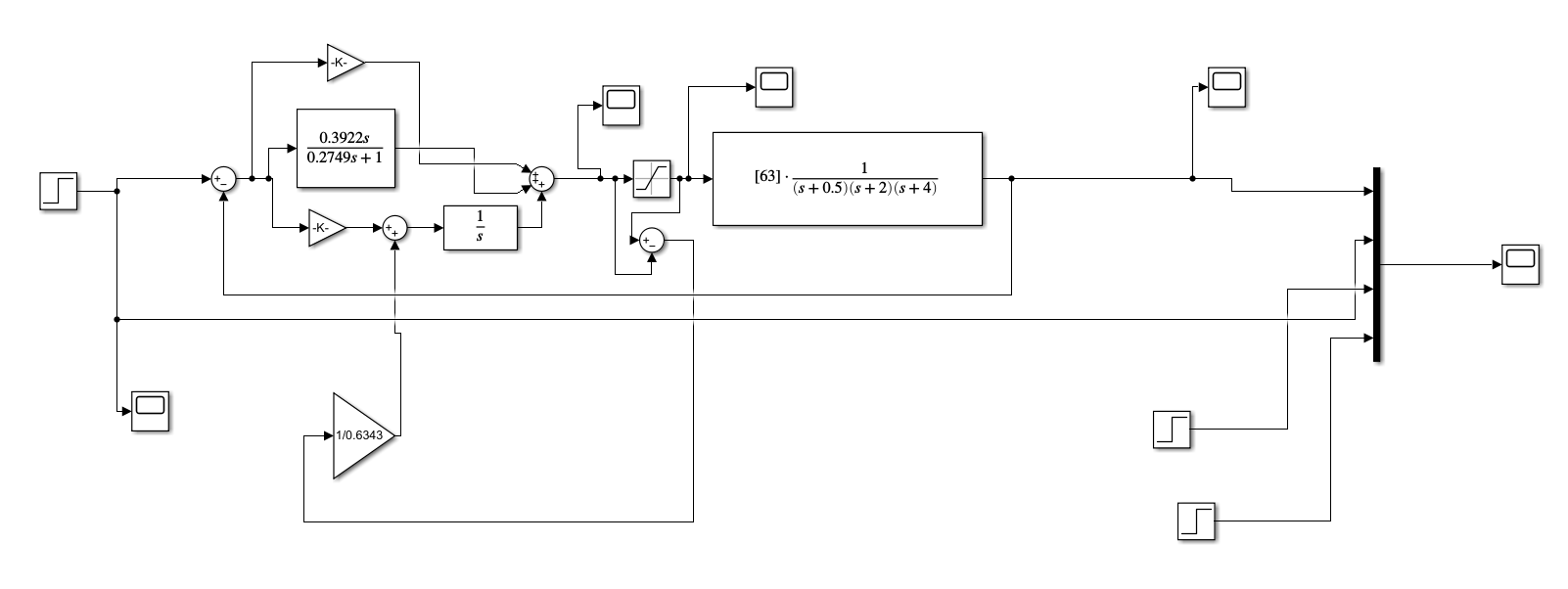
بدلیل استفاده کنترل کننده PID و وجود ترم انتگرال گیر به نوبه ی خود کار را بسیار خراب میکند زمانی که سیگنال فرمان به اشباع می رود. اینگونه که باعث جمع خطاها شده و باعث می شود که سیگنال فرمان از اشباع خارج نشود و باعث می شود سیستم مورد کنترل دیگر از کنترل کننده فرمان نپذیرد که این نشان دهنده ی باز شدن حلقه ی فیدبک می باشد.برای رفع این مشکل از عمل integral anti windup باید استفاده شود. برای زیر بخش آخر بخش سوم اینکار انجام شده و نتیجه به اینگونه شده است. با مقایسه با استفاده نکردن از این عمل:

با پیاده سازی چنین ساختاری برای عمل integral anti windup:

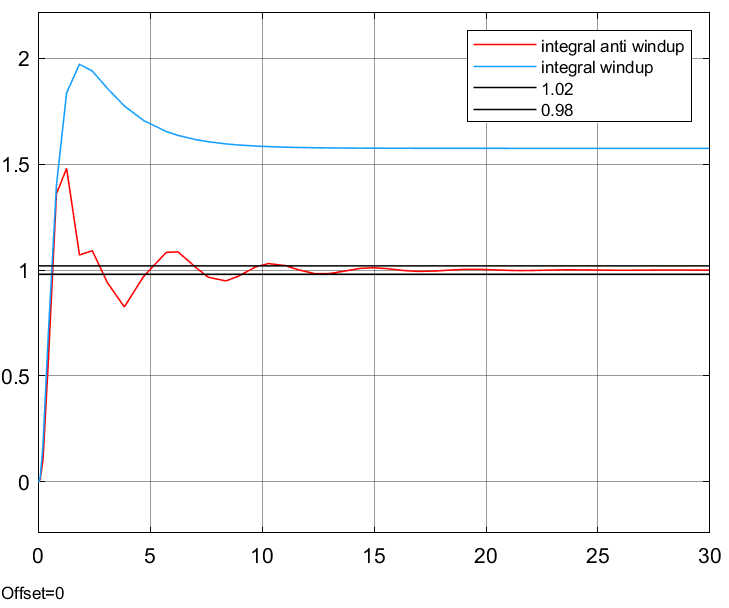


که مقدار Tt از رابطه ی  داریم: Tt=0.6343 همانطور که در ابتدا گفته شد این عمل برای بخش 3-3 در حال انجام است(طراحی کنترل کننده با دست یابی به حد بهره 6 دسی بل و حد فاز 45 درجه)

که برای ساختار کلی داریم:

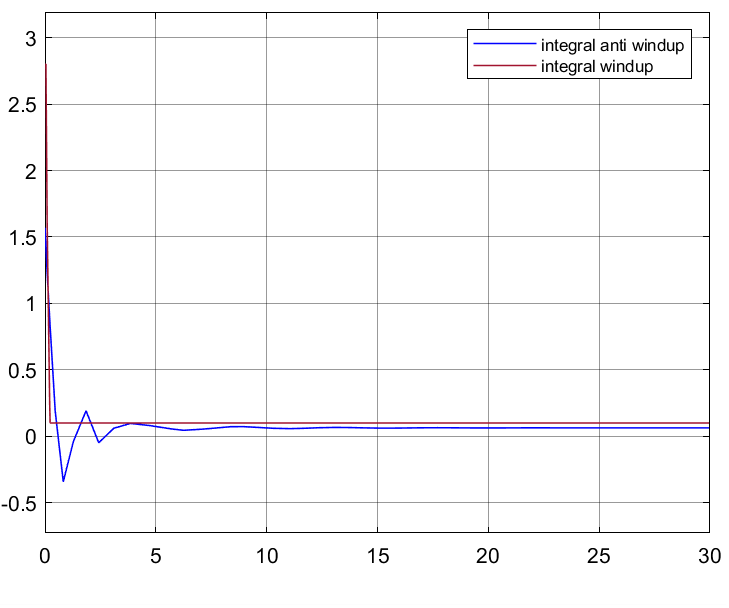


مقایسه ی خروجی با استفاده از عمل integral anti windup و بدون استفاده از این عمل:



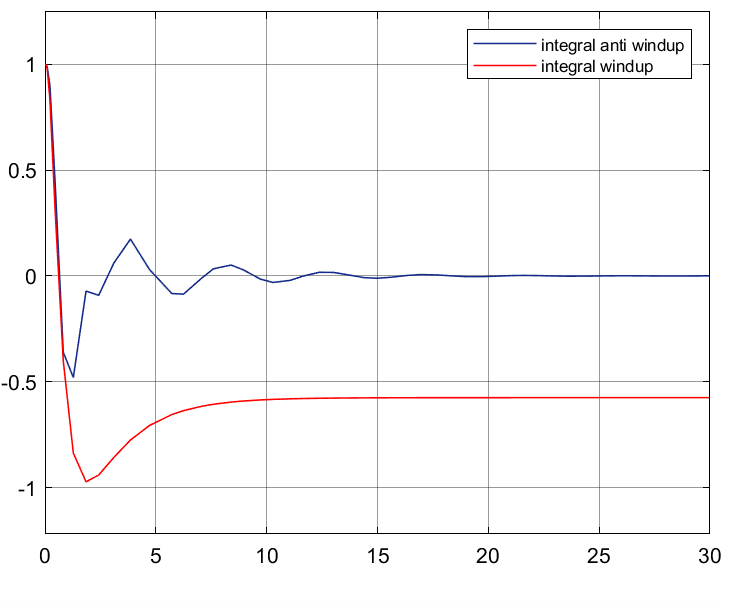
کاملا قابل مشاهده است که عمل integral anti windup باعث به اشباع نرفتن سیگنال کنترل شده و خروجی به حالت ماندگار خود می رسد.

مقایسه ی سیگنال های فرمان:



با مقایسه ی سیگنال های فرمان هم می توان به تلاش عمل integral anti windup برای به اشباع نرفتن سیگنال کنترل را نیز متوجه شد.

سیگنال های خطا:



**عدم قطعیت:**

با تئجه به خروجي هاي بخش هاي قبل کاملا قابل مشاهده بود که عدم قطعيت پاسخ گذرا را متفاوت مي کرد ولي به حالت داِم مي رسيد(به دليل حلقه بسته بودن و استفاده از کنترل کننده) اين متفاوت بودن پاسخ گذرا با نحوه ي تعيين مقادير کنترل کننده (حد فاز و حد بهره) کاملا رابطه دارد که در قسمت بعد به آن پرداخته مي شود.

**اغتشاش:**

حذف اغتشاش مهم ترين مورد در سيستم هاي کنترلي است که حذف سريع تر آن و کم کردن اثر آن جز اهداف کنترلي مي باشد. در بخش هاي قبل با توجه به پاسخ هاي مشاهده شده در اثر ورودي اغتشاش هر کدام سرعت متفاوتي براي حف اغتشاش داشتند، همگي به دليل وجود انتگرال گير اغتشاش را حذف مي کردند ولي مهم آن است که کدام فراجهش کم تر در اثر اعمال اغتشاش و سرعت بيشتر براي رسيدن به حالت ماندگار داشته است.که در قسمت بعد به آن پرداخته مي شود.

## 2-4 تشکیل جدول مقایسه و بررسی تاثیر حد فاز و حد بهره(پاسخ پله،اغتشاش پله)

پاسخ پله:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Phase margin(deg | Gain margin(dB) | IAE | d | Overshoot(%) | Ts(%2) | Tr |  |
| 42.05 | 25.87 | 1.2990 | 0.08 | 43.7 | 6.1 | 0.6 | ZN-ol |
| 35.47 | 25.59 | 1.9798 | 0.34 | 57.38 | 9.85 | 0.6 | CC |
| 16.89 | 31.6 | 1.1537 | - | 9.8 | 5.16 | 1.2 | CHR-ser |
| 44.94 | 28.92 | 2.0213 | 0.22 | 55 | 10.58 | 0.7 | CHR-reg |
| 35.1161 | 19.5548 | 0.8461 | 0.08 | 45.9 | 4 | 0.4 | ZN-cl |
|  |  | 1.1838 | - | - | 2.78 | 2.2 | ZN-cl-weighted |
| 22.64 | 20.39 | 0.8309 | 0.14 | 43.18 | 3.53 | 0.4 | ZN-dmp |
| 3.52 | 6 | 1.1894 | 0.14 | 41.25 | 5.91 | 0.6 | Gain margin(6dB) |
| 45 | 19.39 | 0.9259 | 0.04 | 54 | 2.99 | 0.4 | Phase margin(45 deg) |
| 37.83 | 7.4541 | 3.05 | - | - | 4.32 | 0.8 | Gain(6dB) & phase(45deg) |

ورودی اغتشاش:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Phase margin(deg | Gain margin(dB) | IAE | d | Overshoot% | Ts10% | Ts5% |  |
| 42.05 | 25.87 | 4.4827 | 0.08 | 228 | 6.03 | 7.5 | ZN-ol |
| 35.47 | 25.59 | 5.34 | 0.27 | 227 | 9.4 | 10.89 | CC |
| 16.89 | 31.6 | 11.08 | - | 382 | 5.94 | 6.16 | CHR-ser |
| 44.94 | 28.92 | 7.44 | 0.23 | 272 | 10.25 | 12.25 | CHR-reg |
| 35.1161 | 19.5548 | 1.78 | 0.05 | 139 | 3.09 | 4.1 | ZN-cl |
|  |  | 1.7669 | - | 143.5 | 2.3 | 2.4 | ZN-cl-weighted |
| 22.64 | 20.39 | 2.43 | - | 157 | 3.95 | 4.2 | ZN-dmp |
| 3.52 | 6 | 4.26 | 0.13 | 247 | 5.34 | 5.63 | Gain margin(6dB) |
| 45 | 19.39 | 1.96 | 0.01 | 151 | 3.29 | 3.9 | Phase margin(45 deg) |
| 37.83 | 7.4541 | 47.81 | - | 355 | 30.82 | 34.22 | Gain(6dB) & phase(45deg) |

با مقایسه ی کلی که از مشخصات سیستم (هم پاسخ پله و هم اغتشاش پله) با کم بودن زمان خیزش و زمان نشست مقدار IAE کاهش پیدا می کند. حد فاز و حد بهره ارتباط مستقیمی با این مشخصات دارند. به گونه ای که داریم:

**حد فاز:**

1. هرچه حد فاز زیاد باشد ، پایداری سیستم بیشتر می باشد یعنی نوسانات هم کمتر می باشد و بالعکس.
2. هر چه حد فاز زیاد باشد ، زمان نشست کم می باشد و بالعکس.
3. هرچه حد فاز زیاد باشد زمان خیزش کاهش می یابد و بالعکس.
4. هرچه حد فاز زیاد باشد، فراجهش کاهش می یابد و بالعکس.
5. هرچه حد فاز بیشتر شود، نسبت افت افزایش می یابد و بالعکس.

**حد بهره:**

1. حد بهره ی بالا باعث افزایش پایداری می باشد و بالعکس.
2. حد بهره ی بالا باعث کاهش زمان خیزش می شود و بالعکس.
3. افزایش حد بهره باعث کاهش زمان نشست می شود و بالعکس.
4. با افزایش حد بهره مقدار فراجهش افزایش می یابد و بالعکس.
5. با افزایش حد بهره نسبت افت کاهش می یابد و بالعکس.

همانطور که از جداول مشخص می باشد کاملا از موارد بالا تبعیت می کنند.

\*از جداول مقایسه می توان به این مورد پی برد که روش ZN حلقه بسته (چه نوع اول و چه برای بهبود Tracking ) عملکرد بهتر برای کنترل ارایه کرده اند.