1. Question

The hooke jeeves algoritm that we wrote;

```
   syms x1 x2 f(x1,x2)
   f(x1,x2)=3*x1^2 - 2*x1*x2 + 4*x1 + x2^2 + 3*x2;
% x1 and x2 are references
x1=0;
x2=0;
firstf=f(x1,x2);
fnew=firstf;


iteration = 4;
h= [1 0.5 0.25 0.125]; % changes of step magnitude

for m= 1: iteration
for i=1:10  %  We try for 10 times for the same step magnitude
step=h(m);
if f(x1+step,x2)<f(x1,x2);
            fnew=f(x1+step,x2);
            x1=x1+step;
        elseif f(x1-step,x2)<f(x1,x2);
            fnew=f(x1-step,x2);
            x1=x1-step;
        end
        if f(x1,x2+step)<f(x1,x2);
            fnew=f(x1,x2+step);
            x2=x2+step;
        elseif f(x1,x2-step)<f(x1,x2);
            fnew=f(x1,x2-step);
            x2=x2-step;
        end

    end

end

x1
x2
result = fnew
```

The result is as below;

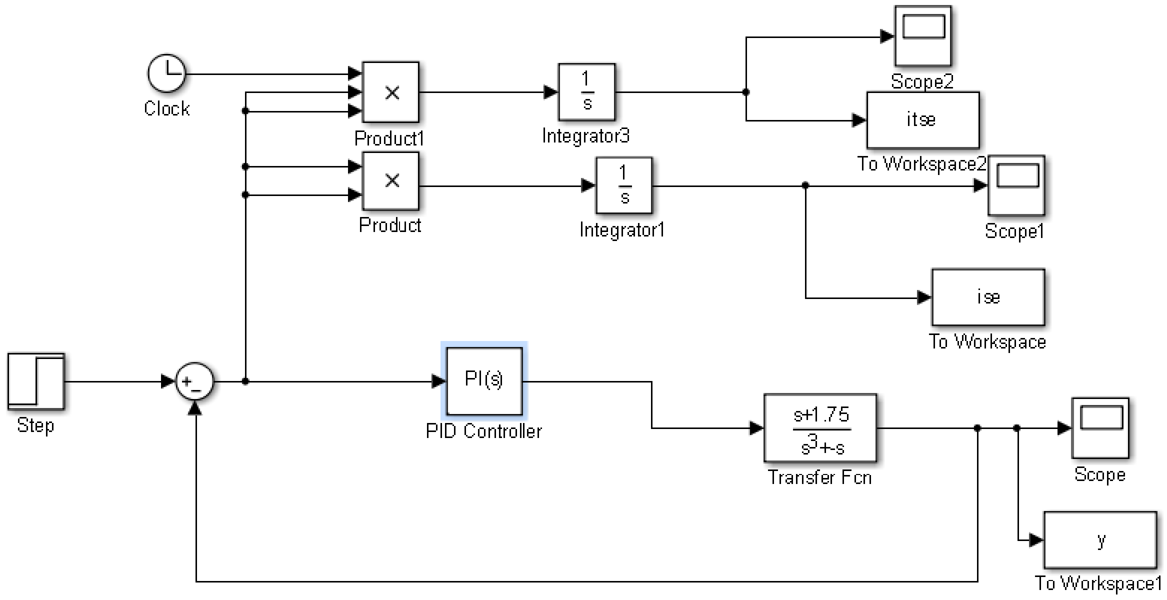
```
>> q1_a

x1 = -1.7500
x2 = -3.2500
```

```
result = -67/8
```

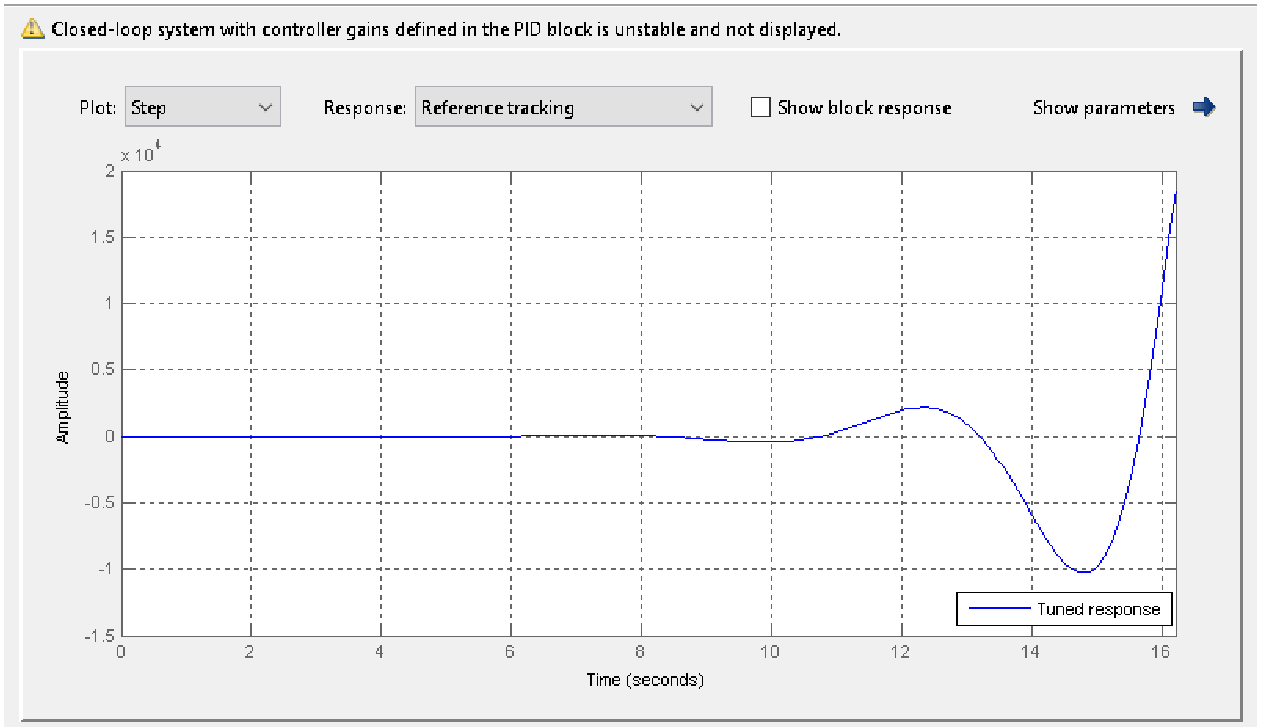
2. Question

First, in order to choose the controller if PI or PD is the best, we have used Simulink and tuning feature.

PI Controller;



When we used PI controller and used the tuning feature, it has been seen that system can not be stable.



⚠ Closed-loop system with controller gains defined in the PID block is unstable and not displayed.

Plot: Step

Response: Reference tracking

□ Show block response

Show parameters

We have seen that also from Mathematica;

```
Fs □Kp □ Ki/s
```

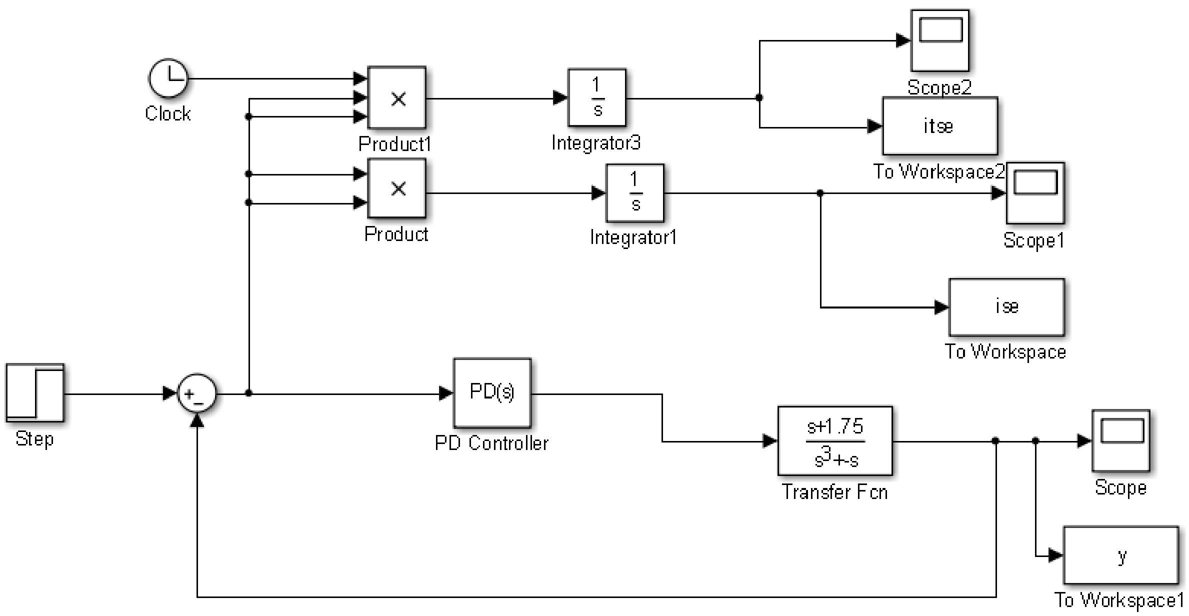
```
Ovrsht □0.01;
ts □2;
□desired □ □Log Ovrsht / □^2 □Log Ovrsht ^2
wdesired □ 4 / ts □ □ □ □ □desired
pds □s^2 □2 □w_n □s □w_n^2 □ □ □desired , w_n □ wdesired
pes □ s^4 □a s^3 □b s^2 □c
```

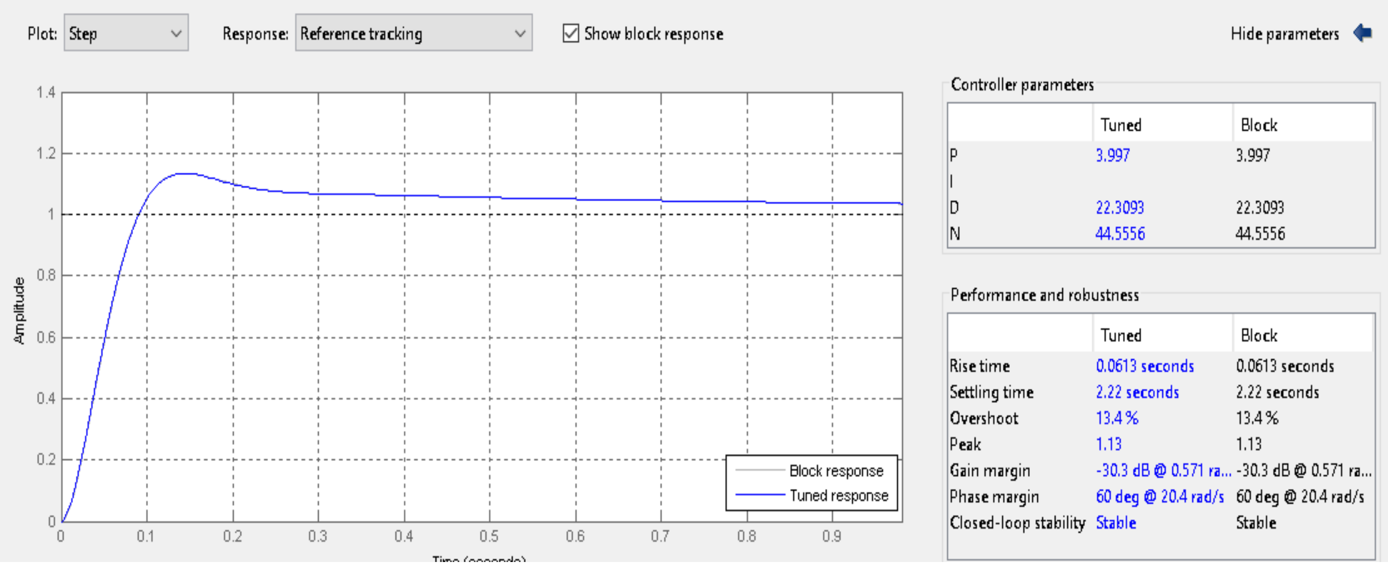
```
RouthTabulation[pcs]
```

```
Power::infy □: □"Infinite expression 1/0 encountered . \□\ \□ButtonBox More… \
    ButtonData :□\"Power::infy\", ButtonStyle □□\"RefGuideLinkText \", ButtonFrame □□\"None\"
```

```
L, 0, ComplexInfinity , ComplexInfinity , 3, 1 L
  Det    ComplexInfinity  ,  3, 1  ComplexInfinity , ComplexInfinity , 0
     ComplexInfinity ,  3, 1  ComplexInfinity , ComplexInfinity , 0
  Det   3, 1  ComplexInfinity , ComplexInfinity , 0  0
 ─────────────────────────────────────────────────────────
                          0
```

PD Controller;





Controller parameters

	Tuned	Block
P	3.997	3.997
I		
D	22.3093	22.3093
N	44.5556	44.5556

Performance and robustness

	Tuned	Block
Rise time	0.0613 seconds	0.0613 seconds
Settling time	2.22 seconds	2.22 seconds
Overshoot	13.4 %	13.4 %
Peak	1.13	1.13
Gain margin	-30.3 dB @ 0.571 ra...	-30.3 dB @ 0.571 ra...
Phase margin	60 deg @ 20.4 rad/s	60 deg @ 20.4 rad/s
Closed-loop stability	Stable	Stable

When we used this tuned controller, we looked at the max value of ise which is the error.

```
>> max(ise)
```

```
ans =

   0.0381
```

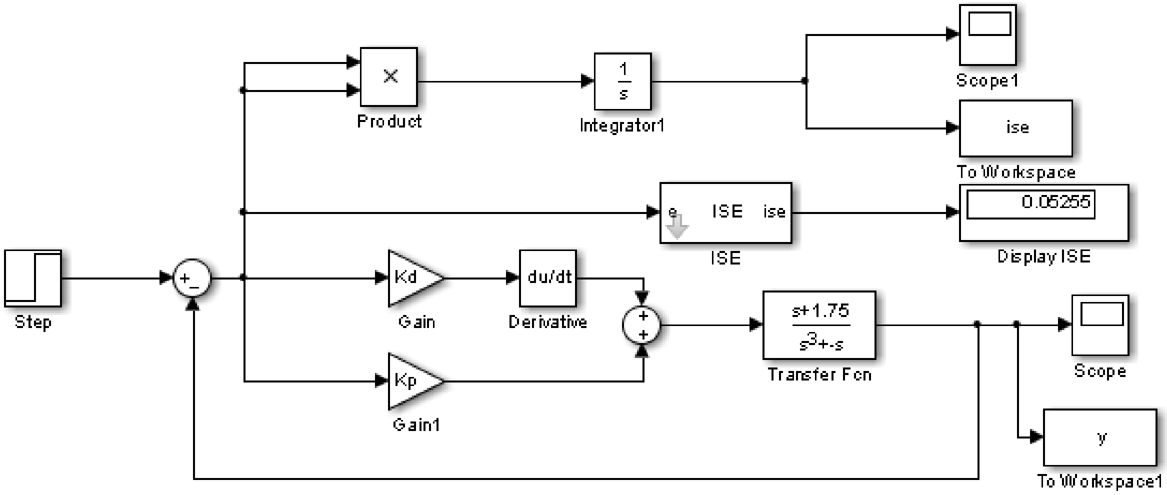
So, we choose PD controller;

For J1;

We have written a function f to take the ise value from Simulink;

```
function sonuc = f(Kd_temp, Kp_temp) % This function is used to
find the ISE value
    global Kp;
    global Kd;
    Kp = Kp_temp;
    Kd = Kd_temp;
    sim('q2_a_simulink_PD');
    sonuc = ise(end);
end
```

Below we can see the Simulink block diagrams;



And now it is time to use our hooke jeeves algorithm;

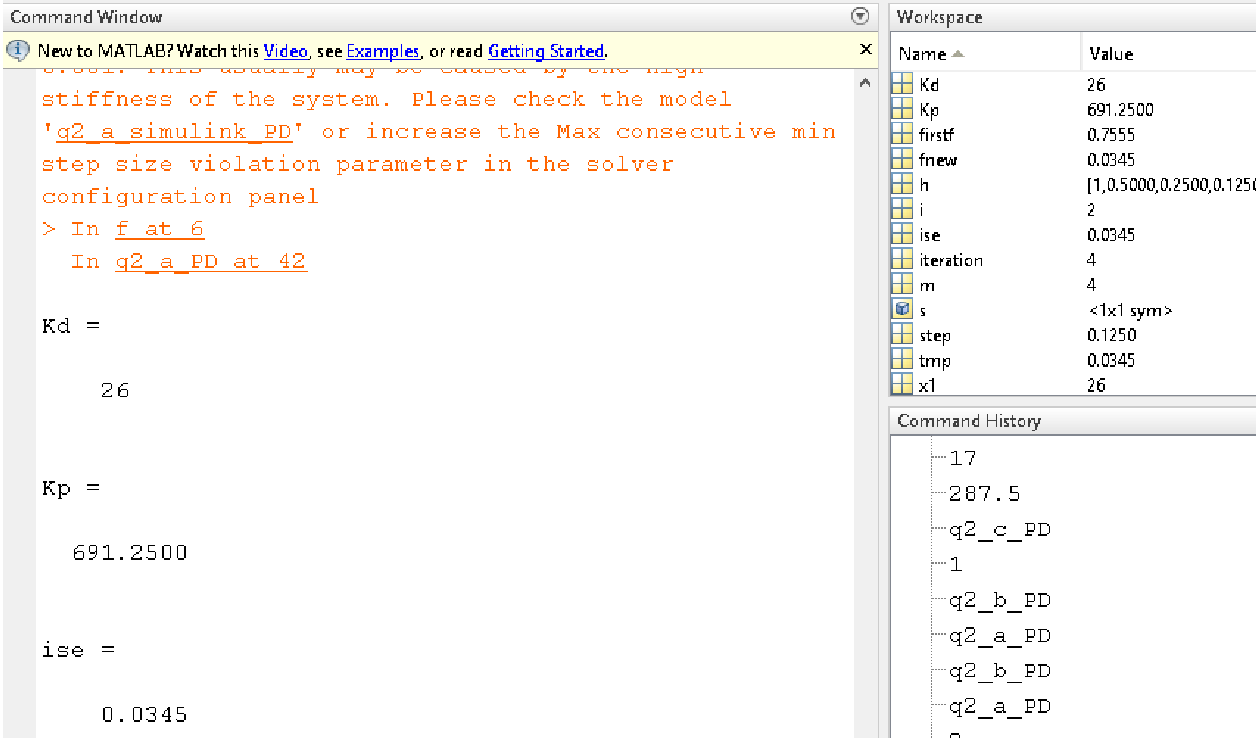
```
global Kp Kd;
global x1 x2;
syms s;
Kd = input('Kd: ');
Kp = input('Kp: ');
```

```
% Hooke Jeeves
% We did not want to change all x1 and x2 from the code
x1=Kd;
x2=Kp;
firstf=f(Kd,Kp);
fnew=firstf;

iteration = 4; % number of step magnitudes
% We declare the step magnitudes as below;
h= [1 0.5 0.25 0.125]; % changes of step magnitude

for m= 1: iteration
    for i= 1:1000
        step=h(m);
        tmp=fnew; % the value before the check
        if f(x1+step,x2)<f(x1,x2);
            fnew=f(x1+step,x2)
            x1=x1+step;
        elseif f(x1-step,x2)<f(x1,x2);
            fnew=f(x1-step,x2)
            x1=x1-step;
        end
        if f(x1,x2+step)<f(x1,x2);
            fnew=f(x1,x2+step)
            x2=x2+step;
        elseif f(x1,x2-step)<f(x1,x2);
            fnew=f(x1,x2-step)
            x2=x2-step;
        end
        if fnew==tmp
            break;
        end
    end

end

Kd=x1
Kp=x2
ise = fnew
```

The starting point is given as [Kd=2,Kp=1.125]

```
Kd = 26

Kp = 691.2500

ise = 0.0345
```

We can see the ISE as 0.0345 from above by hooke jeeves algorithm.



New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
0.001. This usually may be caused by the high
stiffness of the system. Please check the model
'q2_a_simulink_PD' or increase the Max consecutive min
step size violation parameter in the solver
configuration panel
> In f at 6
  In q2_a_PD at 42

Kd =

    26


Kp =

  691.2500


ise =

    0.0345
```

Name ▲	Value
Kd	26
Kp	691.2500
firstf	0.7555
fnew	0.0345
h	[1,0.5000,0.2500,0.125
i	2
ise	0.0345
iteration	4
m	4
s	<1x1 sym>
step	0.1250
tmp	0.0345
x1	26

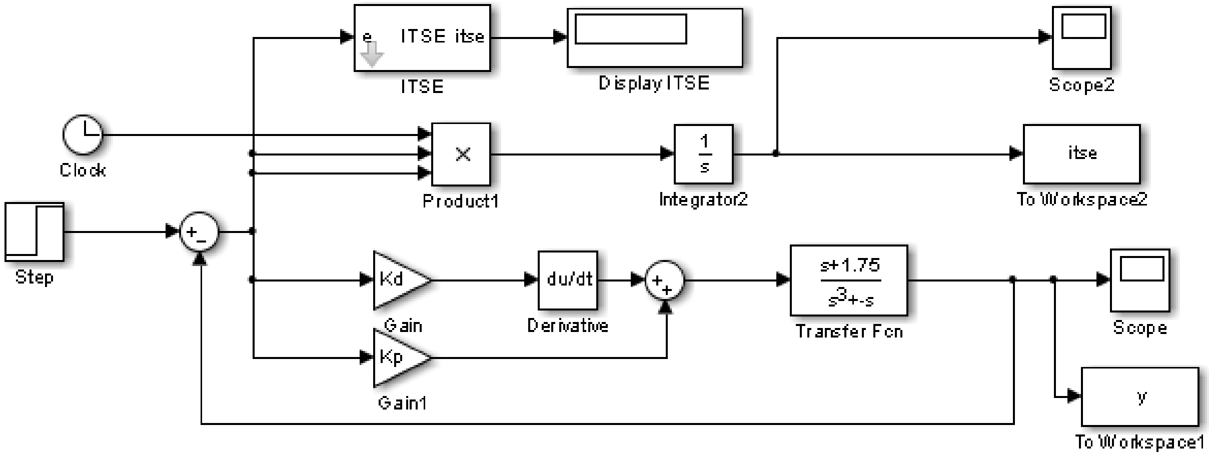
```
17
287.5
q2_c_PD
1
q2_b_PD
q2_a_PD
q2_b_PD
q2_a_PD
```

For J2;

We have written a function f to take the itse value from Simulink;

```
function sonuc = f(Kd_temp, Kp_temp) % This function is used to
find the ISE value
    global Kp;
    global Kd;
    Kp = Kp_temp;
    Kd = Kd_temp;
    sim('q2_b_simulink_PD');
    sonuc = itse(end);
end
```

Below we can see the Simulink block diagrams;



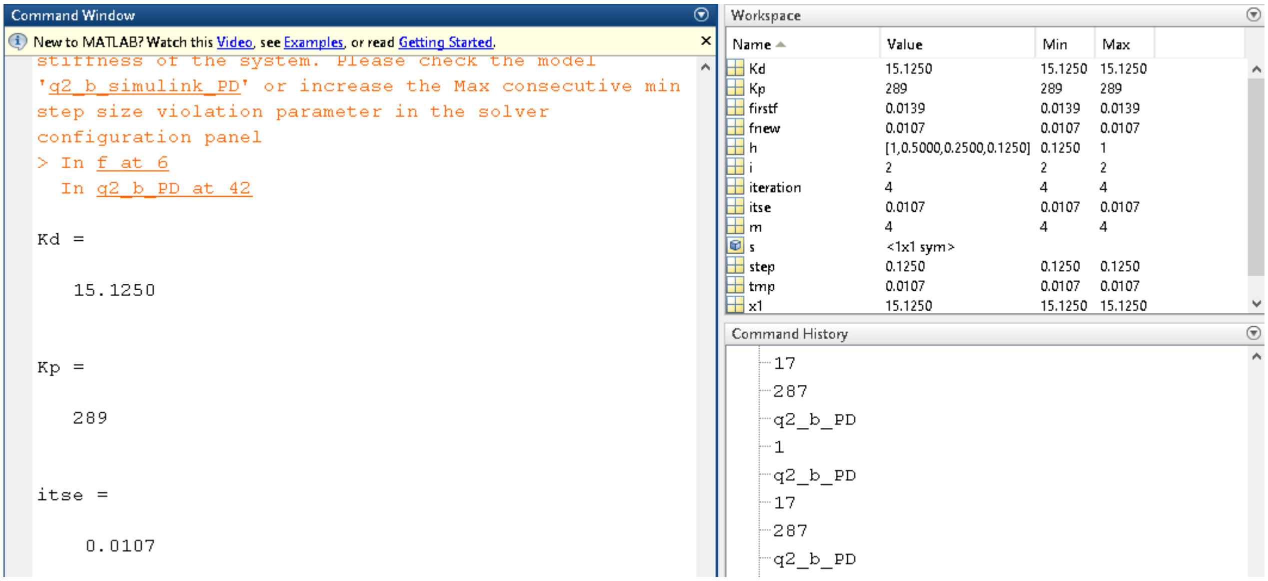
From previous hooke jeeves algorithm, we only written the result as itse.

The starting point is given as [Kd=17,Kp=287.5]

Kd = 15.1250

Kp = 289

itse = 0.0107


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.
Workspace
Command History

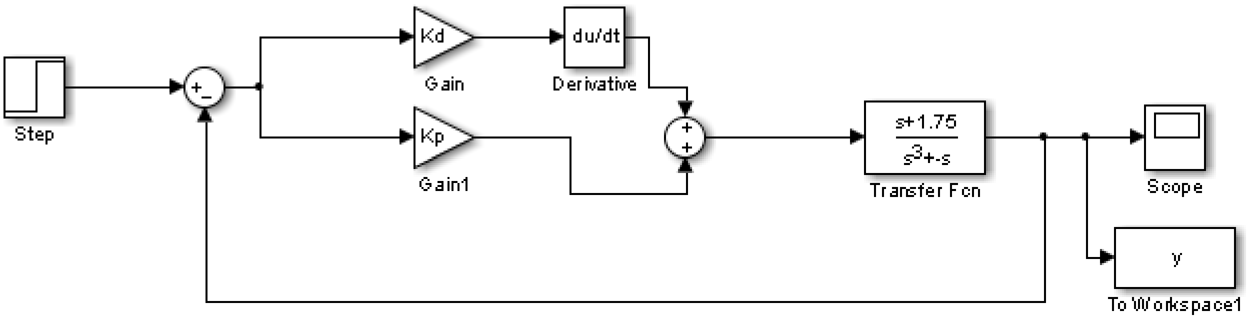
```
stiffness of the system. Please check the model
'q2_b_simulink_PD' or increase the Max consecutive min
step size violation parameter in the solver
configuration panel
> In f at 6
  In q2_b_PD at 42

Kd =

   15.1250


Kp =

   289


itse =

    0.0107
```

Name	Value	Min	Max
Kd	15.1250	15.1250	15.1250
Kp	289	289	289
firstf	0.0139	0.0139	0.0139
fnew	0.0107	0.0107	0.0107
h	[1,0.5000,0.2500,0.1250]	0.1250	1
i	2	2	2
iteration	4	4	4
itse	0.0107	0.0107	0.0107
m	4	4	4
s	<1x1 sym>		
step	0.1250	0.1250	0.1250
tmp	0.0107	0.0107	0.0107
x1	15.1250	15.1250	15.1250

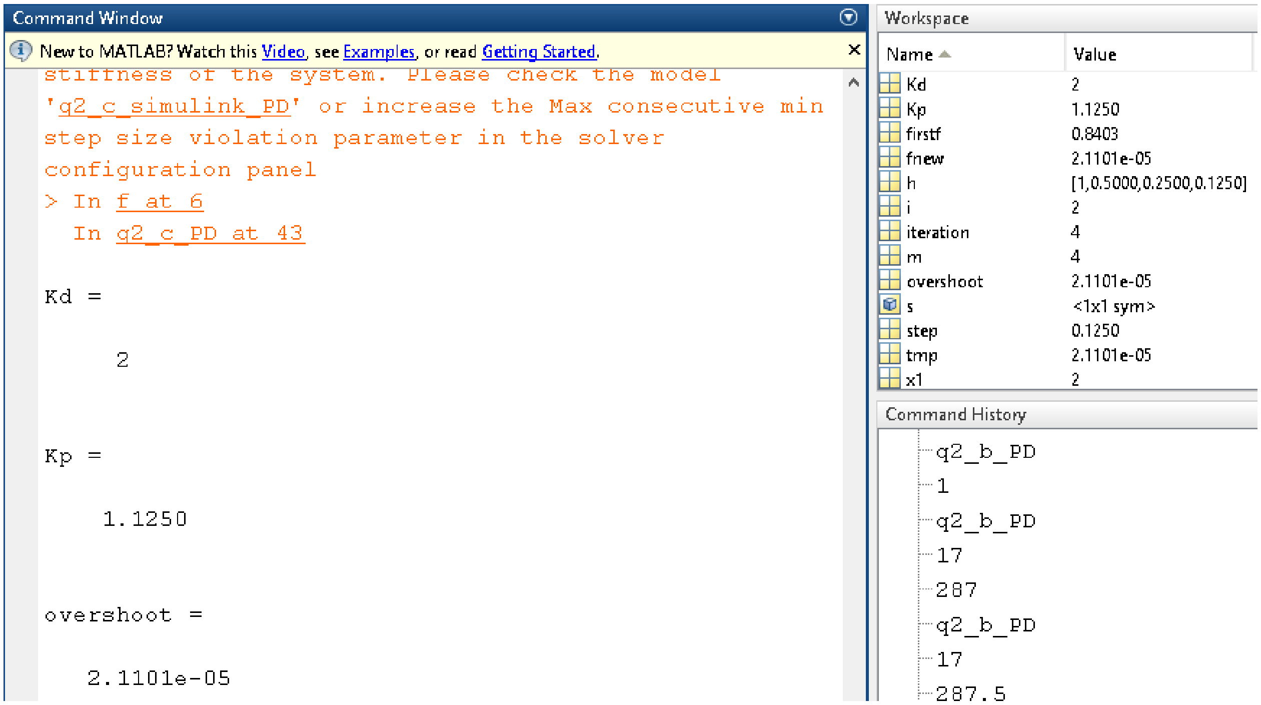
```
17
287
q2_b_PD
1
q2_b_PD
17
287
q2_b_PD
```

For J3;

We have set the overshoot as the objective function.



We have set the starting point as Kd=1; Kp=1;



New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
stiffness of the system. Please check the model
'q2_c_simulink_PD' or increase the Max consecutive min
step size violation parameter in the solver
configuration panel
> In f at 6
  In q2_c_PD at 43

Kd =

     2


Kp =

    1.1250


overshoot =

   2.1101e-05
```

Name	Value
Kd	2
Kp	1.1250
firstf	0.8403
fnew	2.1101e-05
h	[1,0.5000,0.2500,0.1250]
i	2
iteration	4
m	4
overshoot	2.1101e-05
s	<1x1 sym>
step	0.1250
tmp	2.1101e-05
x1	2

```
q2_b_PD
1
q2_b_PD
17
287
q2_b_PD
17
287.5
```

Overshoot is 0.0000211.

Genetic Algorithm;

3. Question

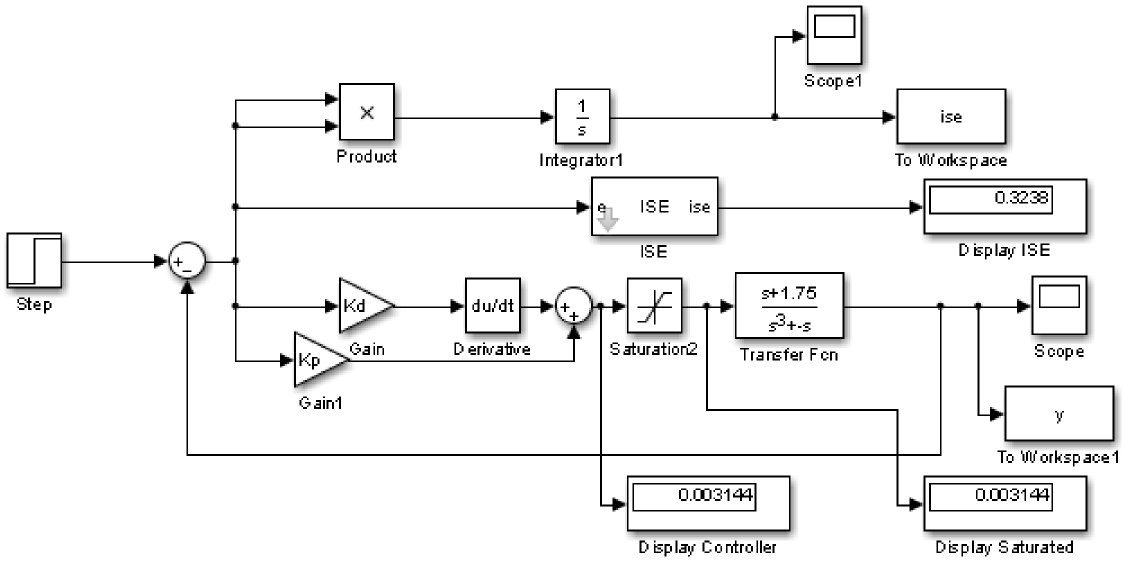
   Comment

   As you can see that the controllers we have found in hooke jeeves algorithm are better controllers.

4. Question

For J1;

We have used the same script and same function as question-2, we just added the saturation block in Simulink as below;



And we determined saturation intervals as 5,-5.

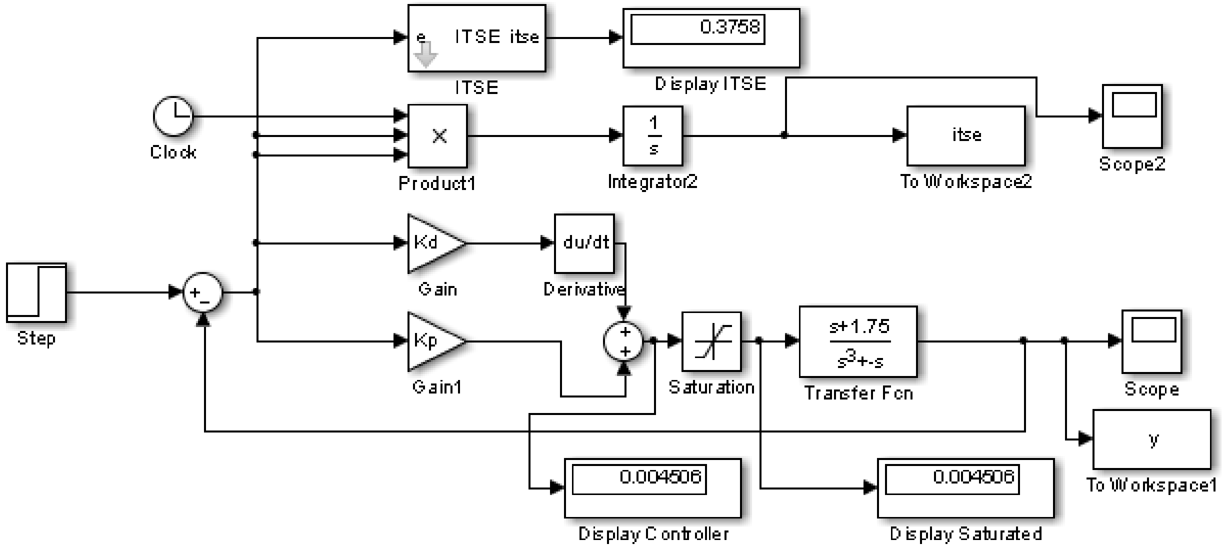
The results were as below;

Kd = 5;

Kp= 12.5;

ise = 0.3238

For J2;



And we determined saturation intervals as 5,-5.

The results were as below;

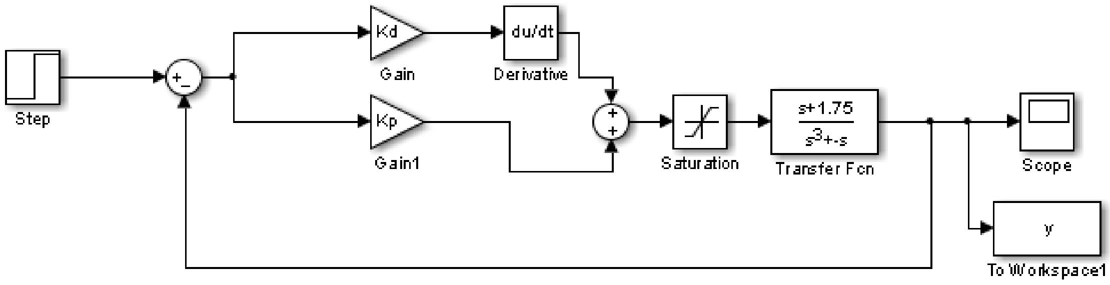
Kd = 4;

Kp= 9;

ise = 0.3758

<u>For J3;</u>

As we have determined overshoot as objective function, in Simulink, we just need to add the saturation block.



The result;

Kd = 2

Kp = 1.1250

overshoot = 1.1001e-06

5. Question

The Controllers we have found in Question-4 are

For J1;

Kd = 5;

Kp = 12.5;

For J2;

Kd=4;

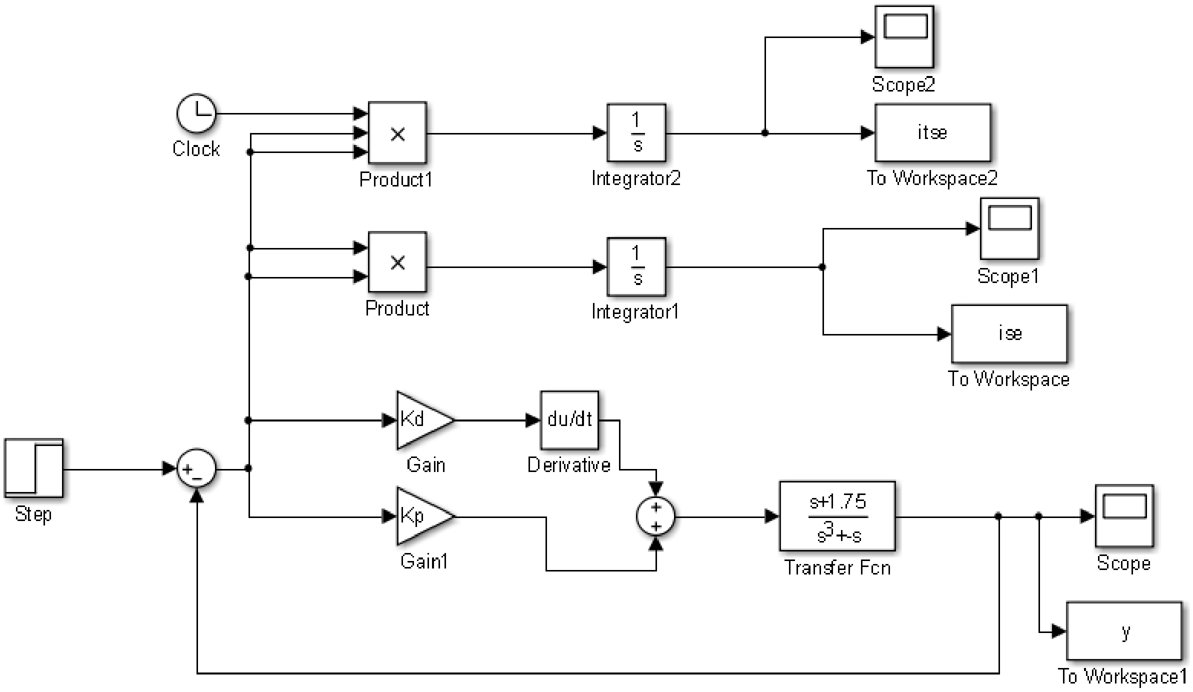
Kp=9;

For J3;

Kd = 2

Kp = 1.1250

Since our system has an integrator(type-1), dead time leads system to infinity.


Scope2
Clock
×
Product1
1
s
Integrator2
itse
To Workspace2
Scope1
×
Product
1
s
Integrator1
ise
To Workspace
Kd
Gain
du/dt
Derivative
Step
+
_
Kp
Gain1
+
+
s+1.75
s3+-s
Transfer Fcn
Scope
y
To Workspace1