Ministry of Scienceand Higher Educationof the Russian Federation Federal State Autonomous Educational Institution of Higher Education
«South Ural State University (national research university)»Electric drive and Mechatronics Department

Chelyabinsk 2023

SOUTH URAL STATE UNIVERSITY (NATIONAL RESEARCH UNIVERSITY)

CONTROLLER DESIGN AND PROCESS SIMULATION USING MATLAB/SIMULINK

COURSEWORK REPORT

on the subject «Automatic control theory»

Coursework supervisor

_______________Aleksandra A. Filimonova

«____»______________2023

Student Hussein Ahmad Hussein

Student group: 354

COURSEWORK TASK

Student: Hussein Ahmad Hussein

Student group P-354

1. **Course unit: Automatic control theory**
2. Topic: Controller design and process simulation using Matlab/Simulink
3. Deadline:
4. List of questions:
   4.1. Tuning of PID Controller using Ziegler-Nichols method.
   4.2. Calculation of the cascade control system of DC motor.
   4.3. Report preparation
5. Source data: PID_Controller_Tuning.pdf,

Calculation_of_the_cascade_control_system_of_DC_motor.pdf

6.Coursework schedule calendar

Name of work item	Completion period	Completion status
Tuning of PID Controller using Ziegler-Nichols method. Modeling the system in the MATLAB software. Studying of the system for stability and quality.		
Calculation of the cascade control system of DC motor Tuning of the internal current loop to technical optimum. Tuning of the external speed loop to technical and symmetrical optimum.		

Modeling the system in the MATLAB/Simulink software. Studying of the system with correction device for stability and quality.		
Report preparation		

Coursework supervisor____________________________Aleksandra A.

ABSTRACT

This coursework describes a methods of controller tuning. The coursework also intends to present an innovative solution for controlling the performance of tuning loops Matlab [Simulink] software .

This coursework has 2 main chapters:

1. Tuning of PID Controller using Ziegler-Nichols method.
2. Calculation of the cascade control system of DC motor.

The advantage of PID controller is its feasibility and easy to be implemented. The PID gains can be designed based upon the system parameters if they can be achieved or estimated precisely .

cascade helps working with nested loop and with which may have disturbance

CONTENT

INTRODUCTION

The aim of the course work is: learn how to use the MATLAB to do the Tuning of PID controller using Ziegler-Nichols Method ,and Calculation of the cascade control system of DC motor

The main objectives of the course work are:

1. Training to do the full coursework in terms of writing ideas and organizing them until you done from it.
2. use the MATLAB with(Simulink) andHow to use the linerzer and show the graph and show its properties .
3. make Tuning of PID controller using Ziegler-Nichols Method.
4. Calculation of the cascade control system of DC motor. Content of the coursework:

1. Tuning of PID controller using Ziegler-Nichols Method

2. Calculation of the cascade control system of DC motor

2.

The Ziegler-Nichols method is a popular method for tuning P, PI, and PID controllers. This method begins by zeroing the integral and differential gains before increasing the proportional gain until the system becomes unstable.

$H_{max} = 0.996$

$H_{ss} = 0.996$

then overshoot $= \frac{0.996-0.996}{0.996} = 0\%$

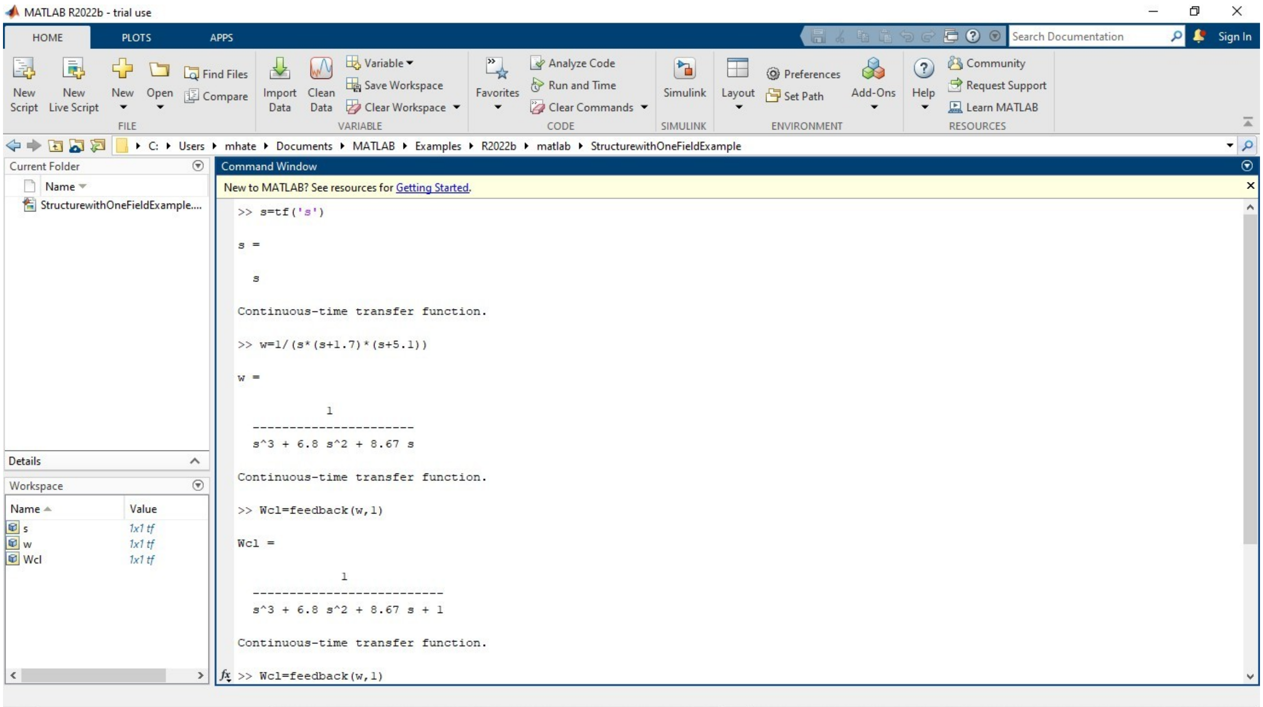
$t_s = -(hss - h0) * 5\% + hss = 95\% h_{ss}$

$t_s = t(0.95 * hss) = 24.3$

$t_r = t_{(0.9)} - t_{(0.1)} = 17.3$

$t_r = 18.9 - 1.61 = 17.3$

With MATLAB:


MATLAB R2022b - trial use
HOME
PLOTS
APPS
Current Folder
StructurewithOneFieldExample....
Workspace
Name
Value
s
1x1 tf
w
1x1 tf
Wcl
1x1 tf
Command Window
New to MATLAB? See resources for Getting Started.
>> s=tf('s')
s =
s
Continuous-time transfer function.
>> w=1/(s*(s+1.7)*(s+5.1))
w =
1
s^3 + 6.8 s^2 + 8.67 s
Continuous-time transfer function.
>> Wcl=feedback(w,1)
Wcl =
1
s^3 + 6.8 s^2 + 8.67 s + 1
Continuous-time transfer function.
>> Wcl=feedback(w,1)

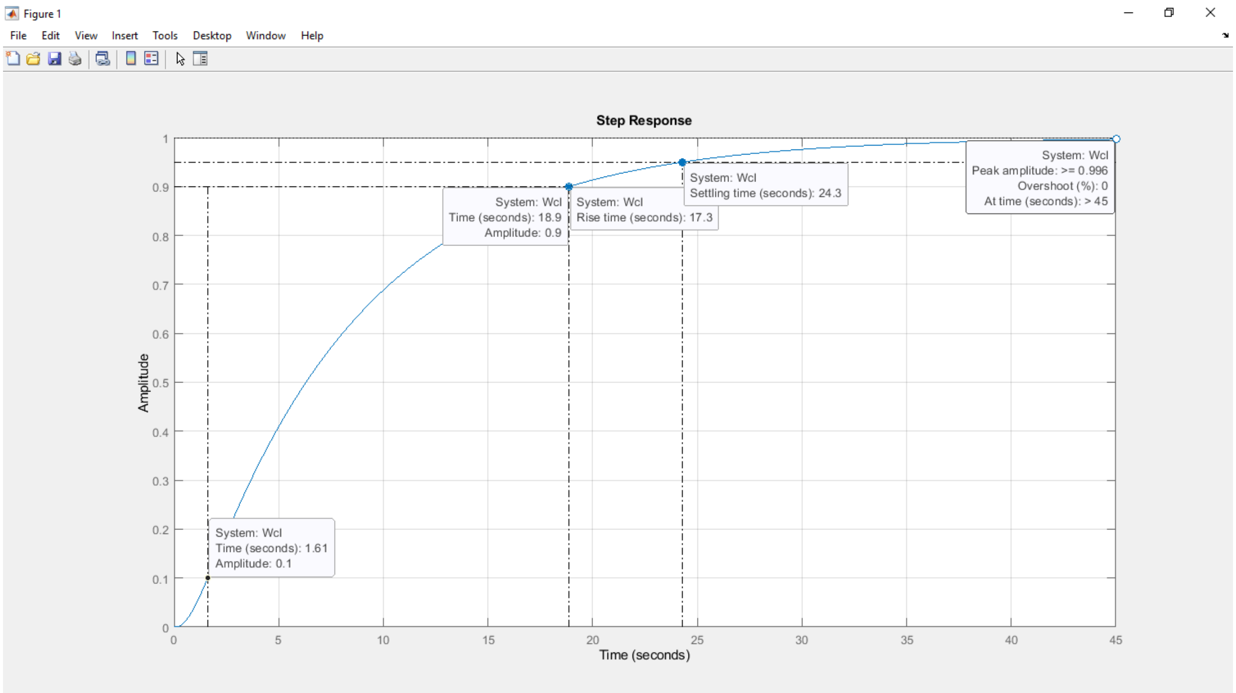
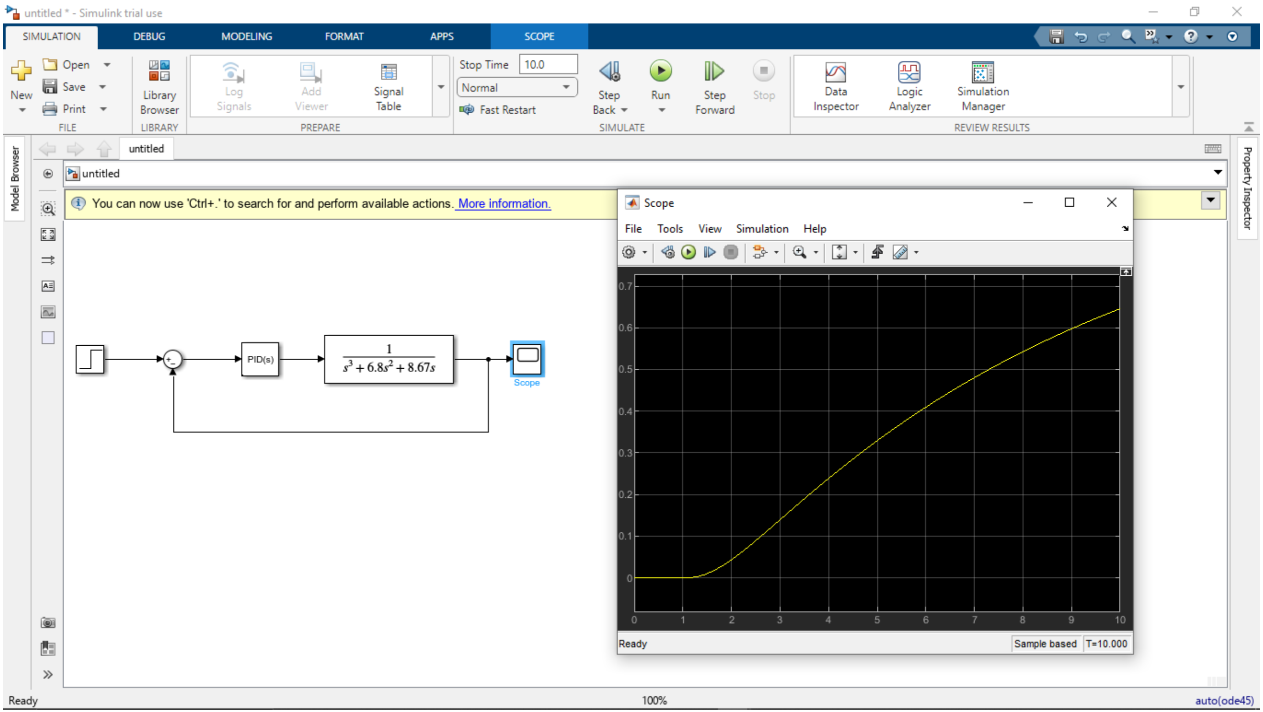


Figure 1
Step Response
System: Wcl
Peak amplitude: >= 0.996
Overshoot (%): 0
At time (seconds): > 45
System: Wcl
Settling time (seconds): 24.3
System: Wcl
Time (seconds): 18.9
Amplitude: 0.9
System: Wcl
Rise time (seconds): 17.3
System: Wcl
Time (seconds): 1.61
Amplitude: 0.1
Amplitude
Time (seconds)

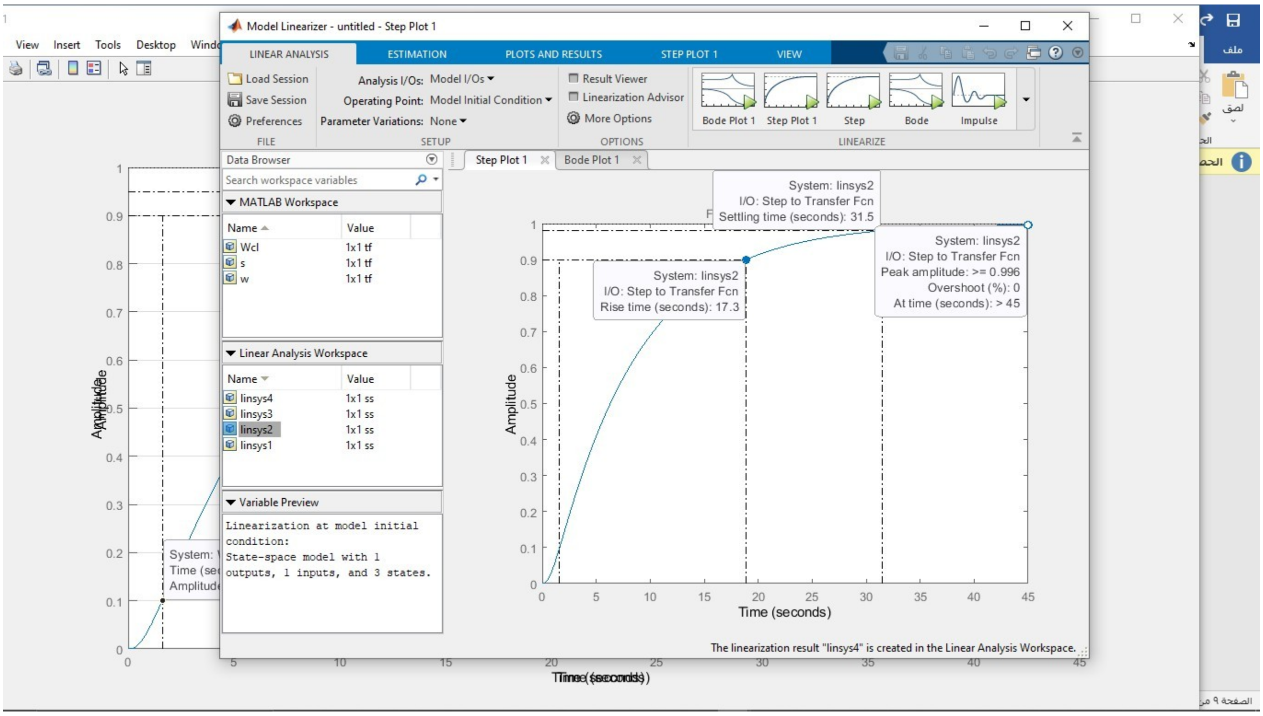
In the second method, we first set Ti = ∞ and Td = 0 for the proportional control action, increase Kp from 0 to a critical value Kcr where the output first exhibits sustained oscillations. Thus, the critical gain Kcr and the corresponding critical period Pcr are experimentally determined as illustrated in Fig. 1.10. According to ZN second method the values of Kp, Ti and Td can be set according to the formula

Type of Controller	K_p	T_i	T_d
P	$0.5K_{cr}$	∞	0
PI	$0.45K_{cr}$	$\frac{1}{1.2}P_{cr}$	0
PID	$0.6K_{cr}$	$0.5P_{cr}$	$0.125P_{cr}$

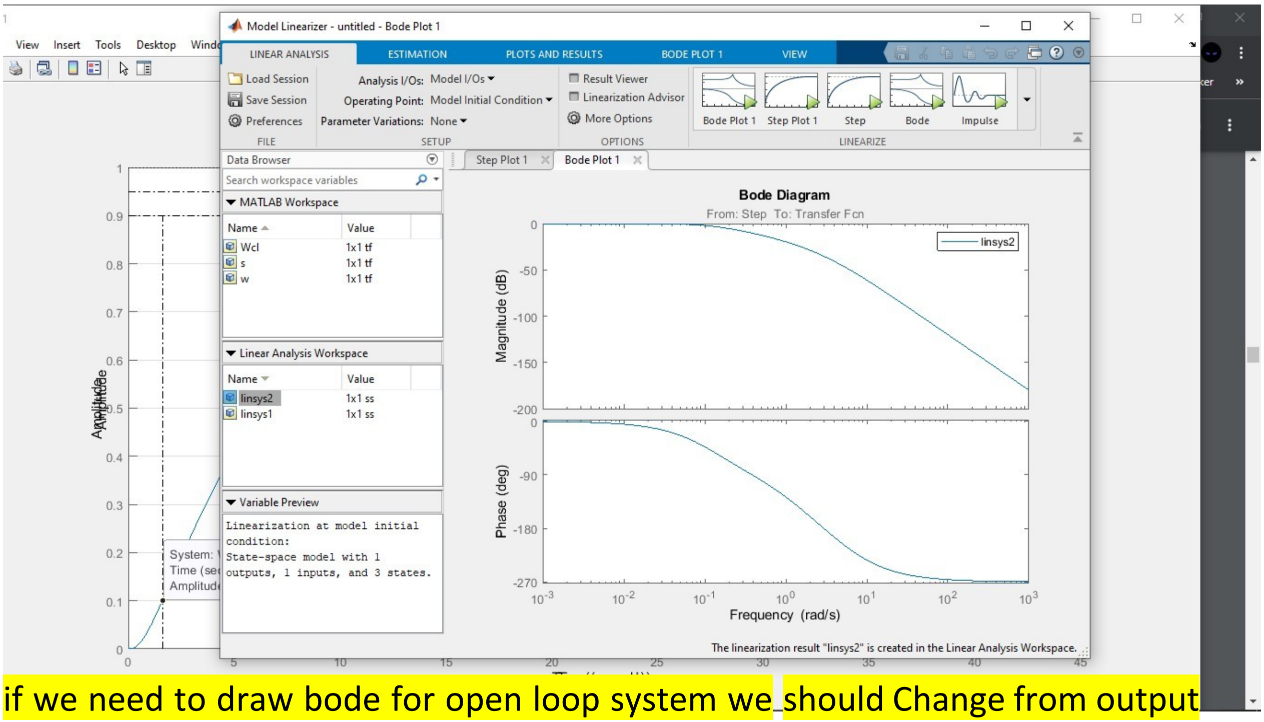
By Simulink:



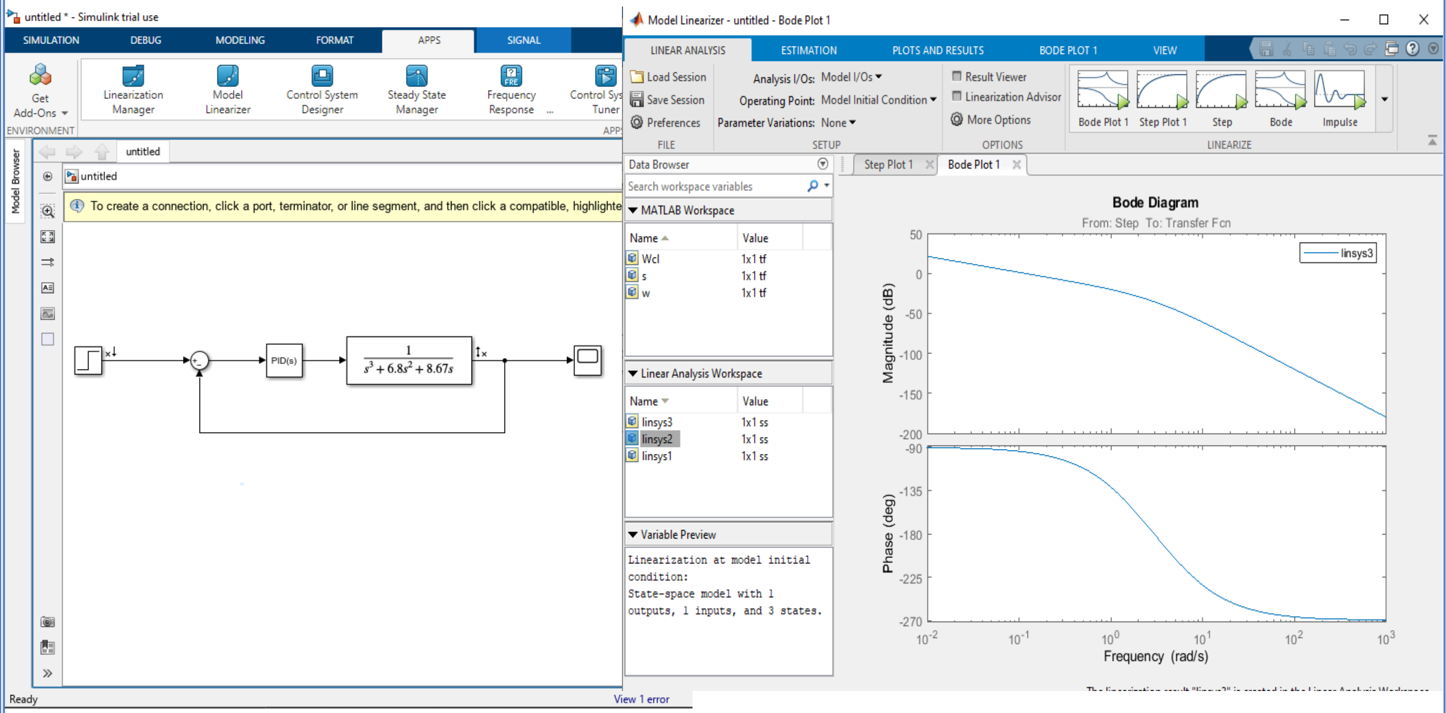
But these graphs are not enough. We show other graphs, we set the measurement of input and output, then we click on model Linerzer , then we get the same graphs and bode.



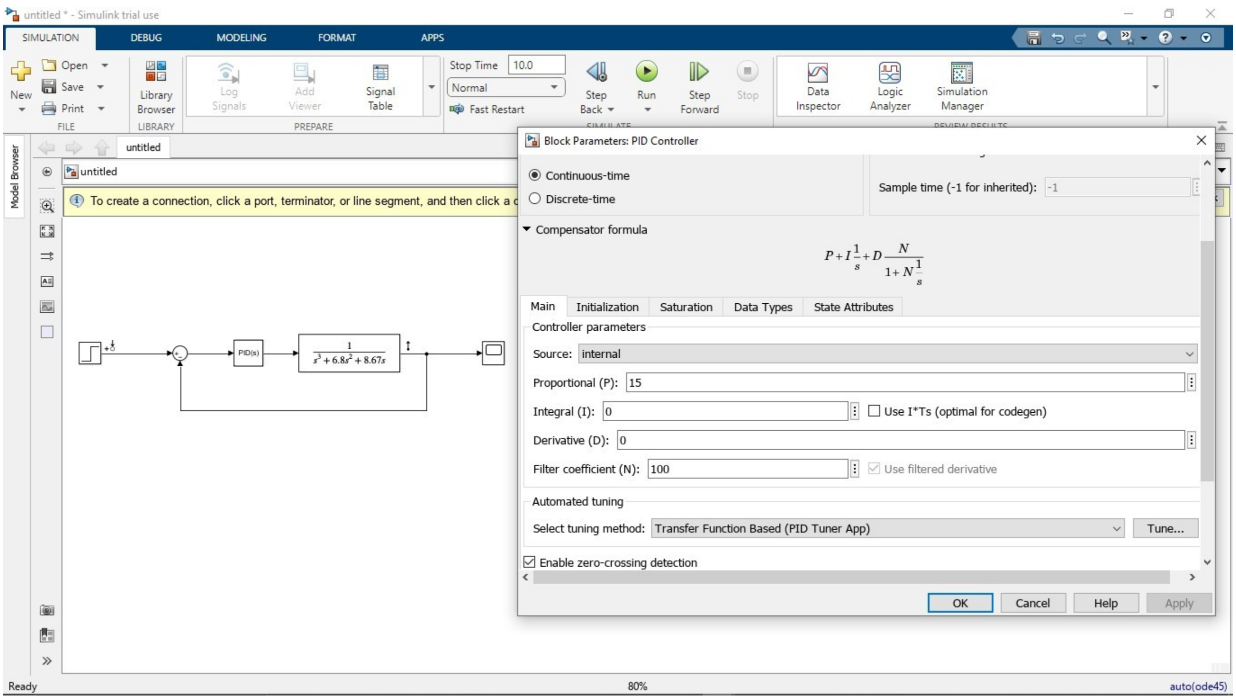
this bode will be wrong because will be should the bode open-loop-system (output and input)



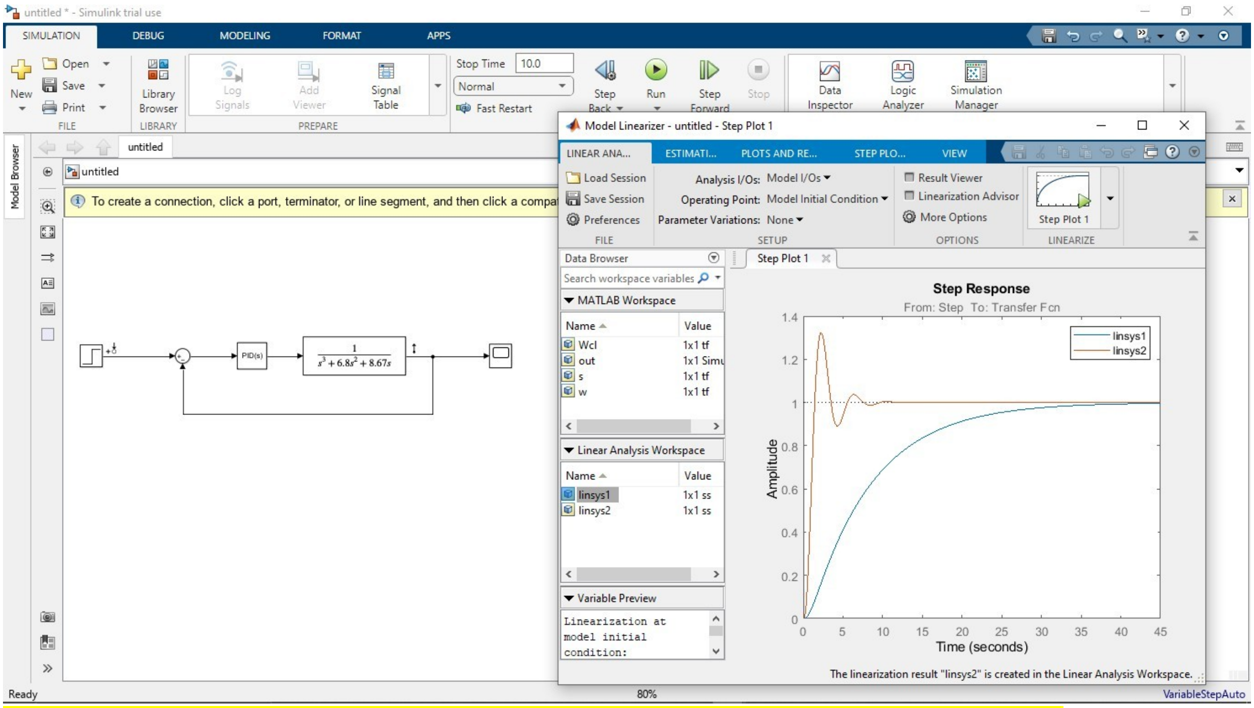
if we need to draw bode for open loop system we should Change from output measurement to open loop system(output) and like same from (input)



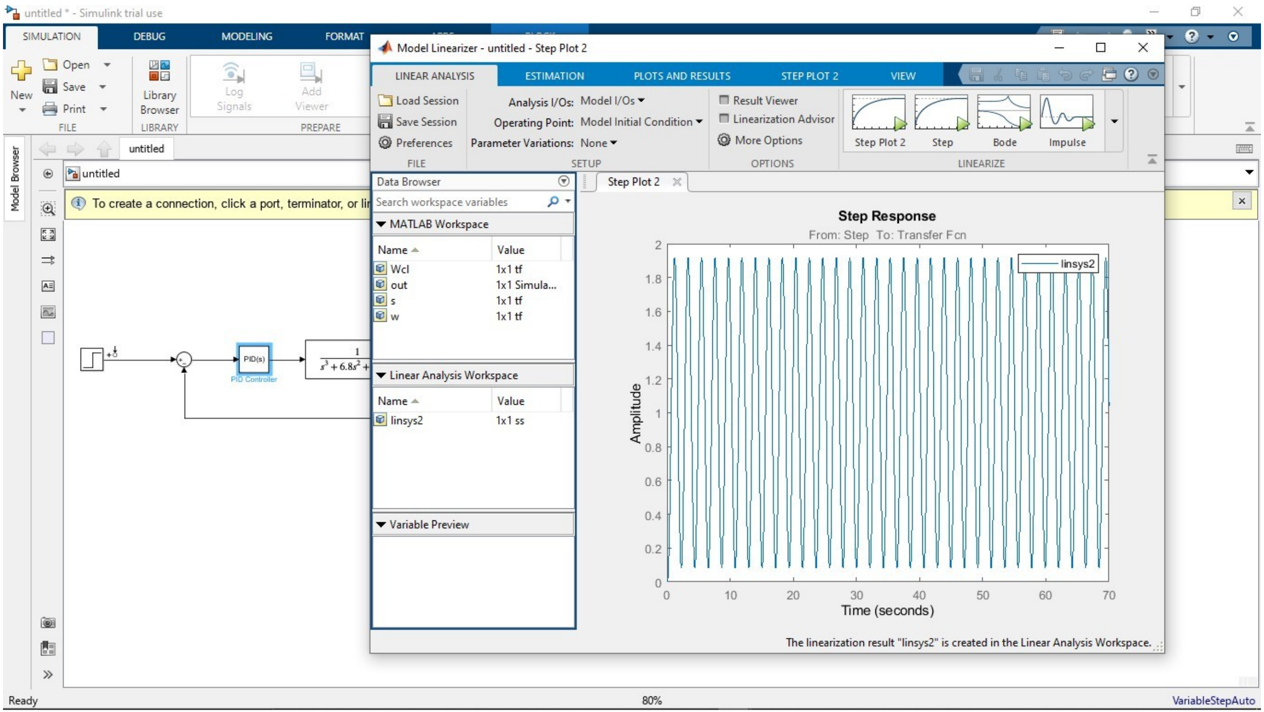
I noticed the line start from zero when he was in measurement and this is wrong.



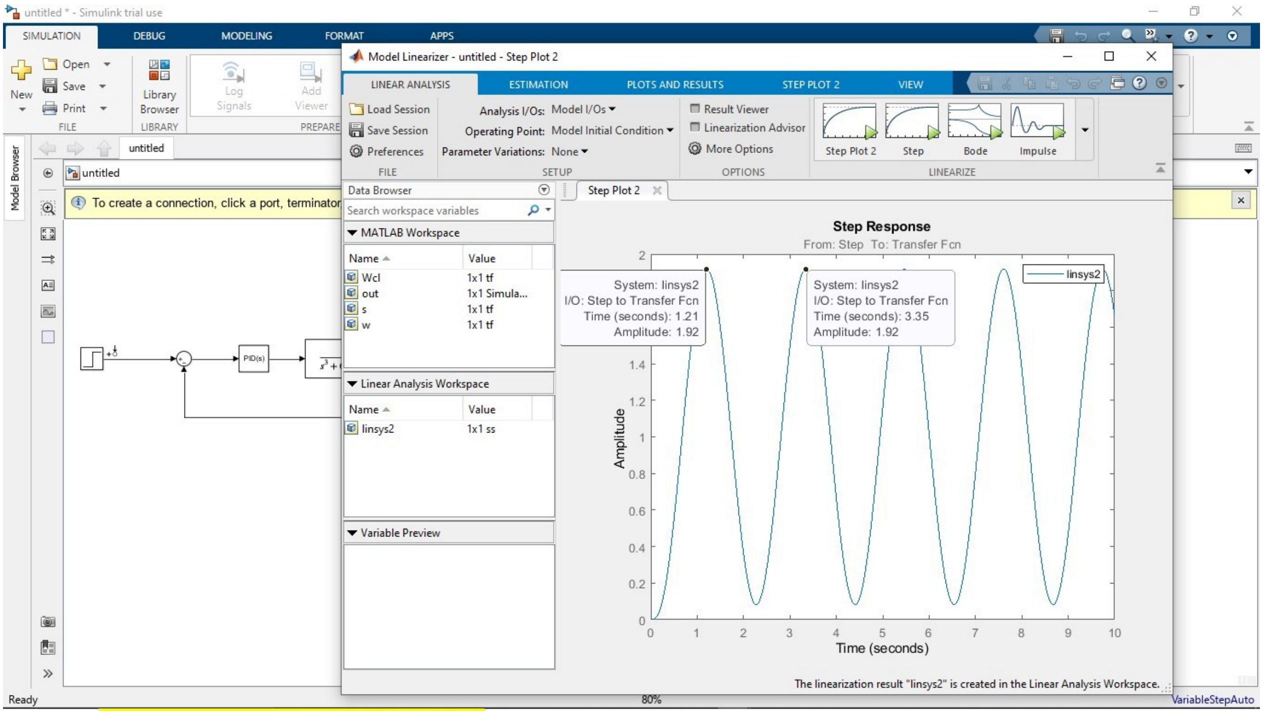
we change proportional (P) from PID Controller to get marginally stable we are put any number we will put 15



I will change the number because it must be stable if I need to get the marginally stable to get tha number proportional (P) : **(6.8*8.67)= 58.956**



And we going to the property > Limits > to change the X Limits to get this graph (make zoon)



1. experimental method:

We change K_p till the system is marginally stable

$K_p = K_{cr}$ = 58.956

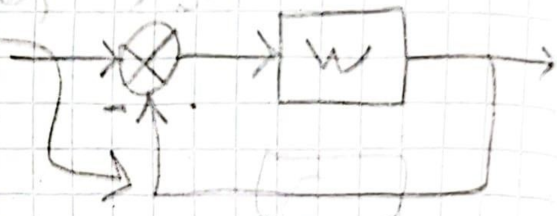
Define pcr = (3.35-1.21)= 2.14

2. Theoretical method:

$$W = \frac{1}{S(S+1,7)(S+5,1)}$$

σ ; $T_s < 5s$; $OV < 20\%$

1 = it's a unity feedback



$h_{max} = 0,996$

$h_{ss} = 0,996$

t_s = setting time

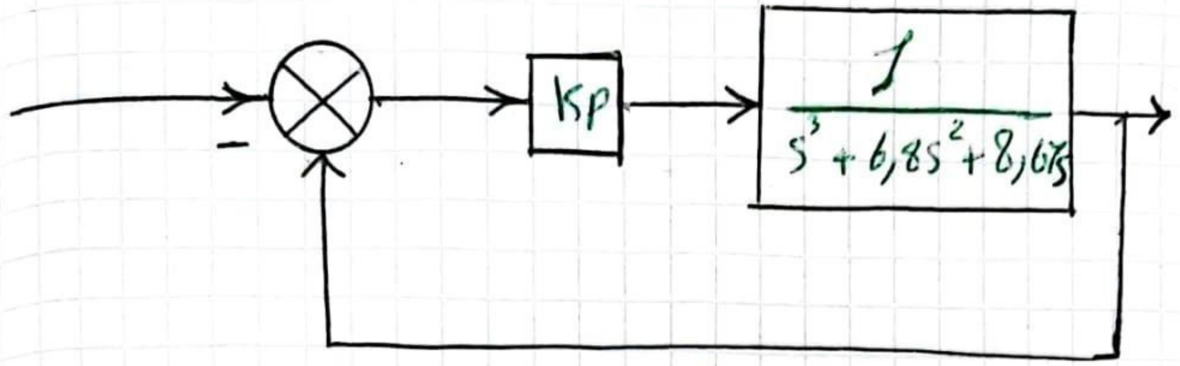
$\sigma = \frac{h_{max} - h_{ss}}{h_{ss}} = \frac{0,996 - 0,996}{0,996} = 0\%$

over shot

$t_s = (h_{ss} - h_0) \times 5\% + h_{ss} = 95\% h_{ss}$

$t_s = t(0,95 \times h_{ss}) = 24,3$

$w = $



Kp = Kcr for marginally stable system -

$$wd = \frac{Kp}{s^3 + 6{,}8s^2 + 8{,}67s + Kp}$$

s^3	1 8,67
s^2	6,8 Kp
s	0
s^0	Kp

$$-\frac{1}{6{,}8}\begin{vmatrix} 1 & 8{,}67 \\ 6{,}8 & Kp \end{vmatrix} = 0$$

$Kp - 58{,}956 = 0$

$Kp = \boxed{58{,}956}$ Kcr

roots

$S = iw$ w = frequency

$(iw)^3 + 6{,}8(iw)^2 + 8{,}67\,iw + 58{,}956 = 0$

$-iw^3 + 6{,}8w^2 + i\,8{,}67\,w + 58{,}956 = 0$

in matlab; $r = [-i \;\; -6{,}8 \;\; 8{,}67i \;\; 58{,}956]$

roots(r) =

$\boxed{w = 2,9445}$

$$P_{cr} = 2 \times \pi/w = \frac{2 \cdot 3,141592659}{2,9445} = \boxed{2,1339}$$

calculate cofficient

① for (P)

$K_{p1} = 0,5 \times K_{cr} = 0,5 \times 58,956 = \boxed{29,478}$

② For (PI):

$$C = K_p\left(1 + \frac{1}{T_i S} + T_d S\right)$$

$k_{p_2} = k_p = 0,4 \times 58,956 =$

$\boxed{k_{p_2} = k_p = 26,5305}$

$\boxed{K_{p_2} = K_p} \Rightarrow k_{I_2} = \frac{K_{p2}}{T_{i2}}$

$T_{I2} = \frac{1}{1,2} P_{cr} = \frac{1}{1,2} \times 2,1339 = \boxed{1,7782}$

$K_{I2} = \frac{26,5305}{1,782} = \boxed{14,9197}$

③ For (PID):

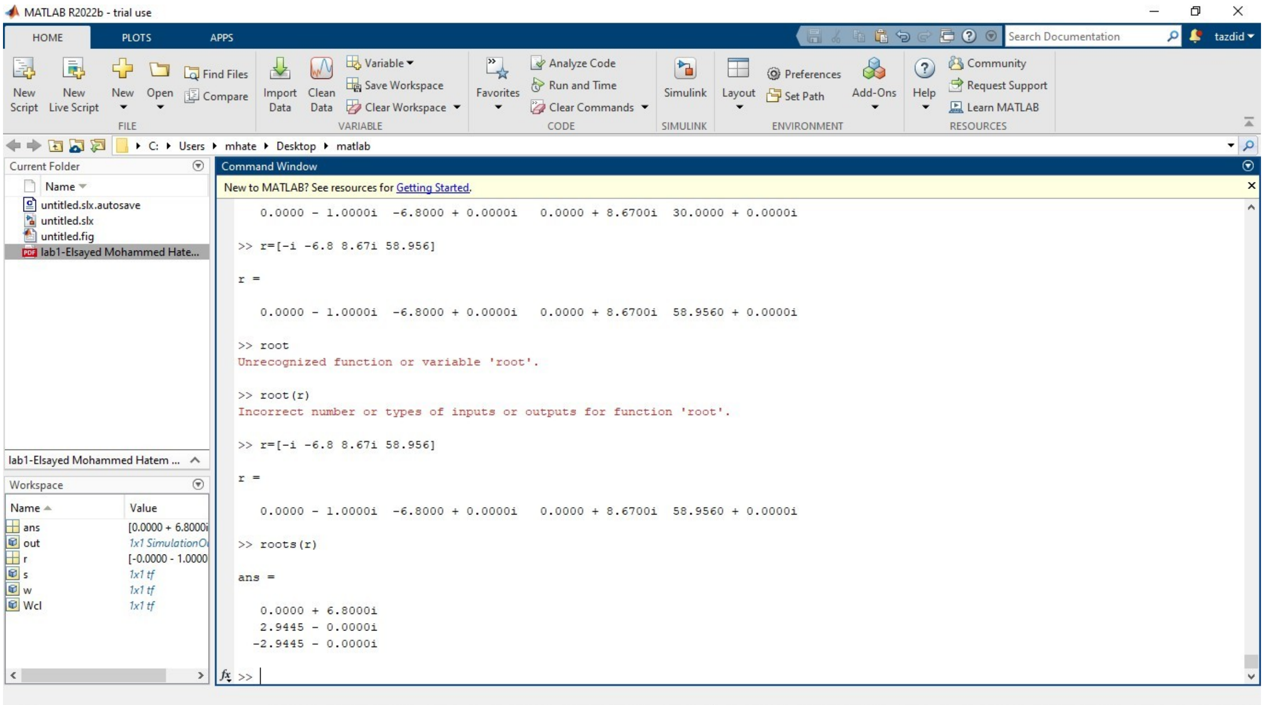
$K_{p_3} = 0,6\ K_{cr} = 0,6 \times 58,956 = \boxed{35,37}$

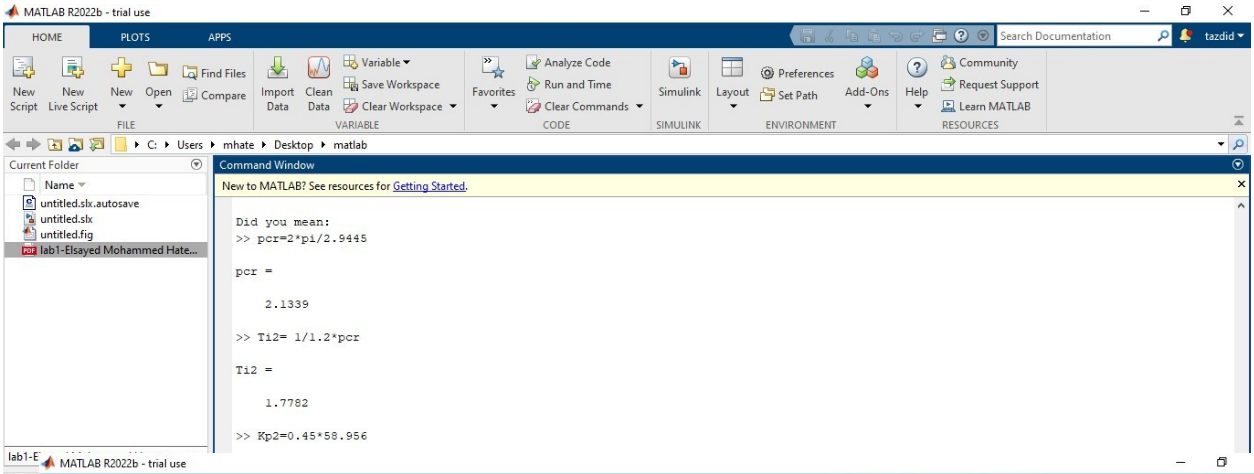
$K_{I_3} = \frac{K_{p3}}{T_{I_3}} = \boxed{33,1544}$

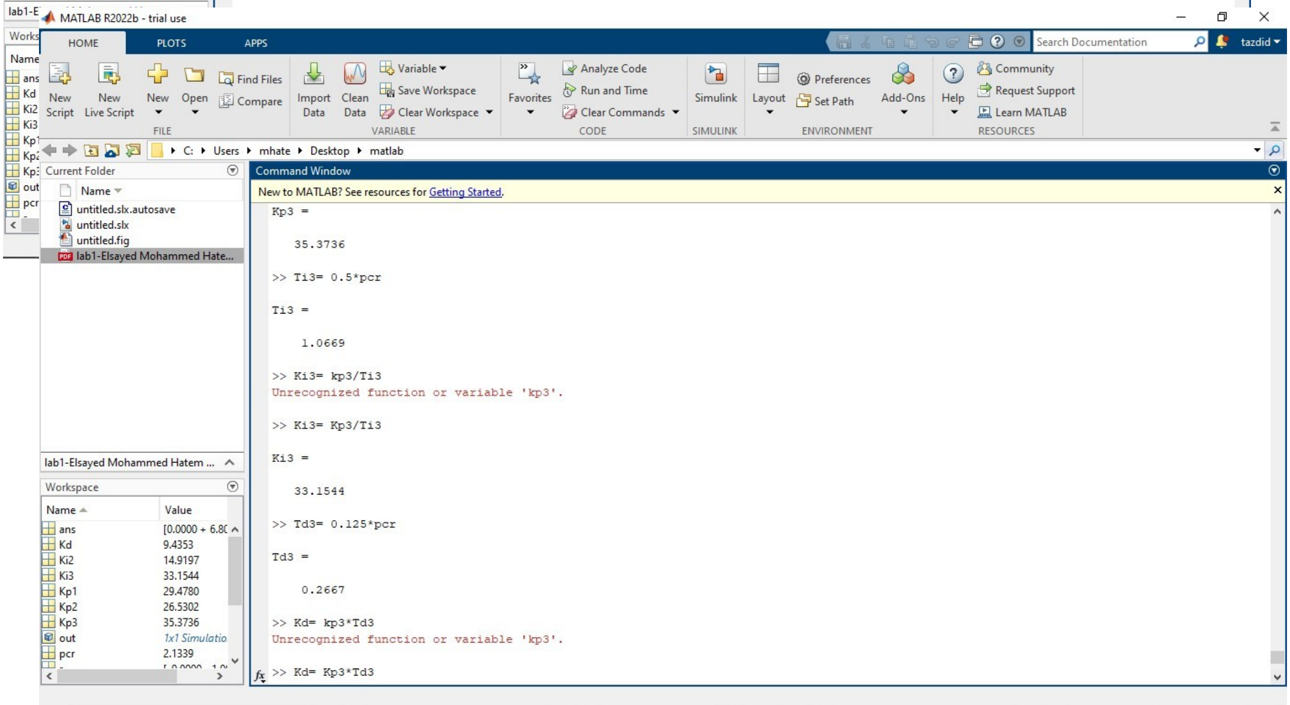
$T_{I_3} = 0,5\ P_{cr} = \boxed{1,0669}$

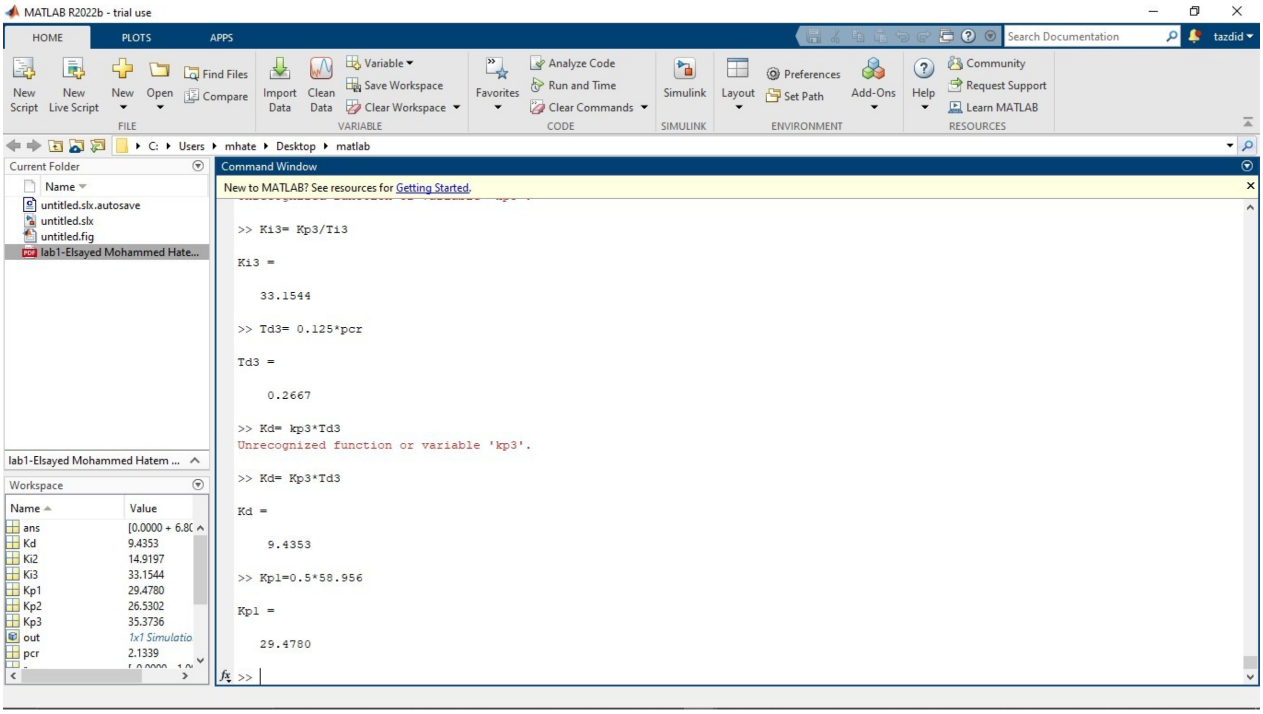
$k_d = k_{p3} \cdot T_{d_3} = \boxed{9,4353}$

$T_{d_3} = 0,125\ P_{cr} = \boxed{0,2667}$


MATLAB R2022b - trial use
Command Window
New to MATLAB? See resources for Getting Started.
0.0000 - 1.0000i -6.8000 + 0.0000i   0.0000 + 8.6700i 30.0000 + 0.0000i
>> r=[-i -6.8 8.67i 58.956]
r =
0.0000 - 1.0000i -6.8000 + 0.0000i   0.0000 + 8.6700i 58.9560 + 0.0000i
>> root
Unrecognized function or variable 'root'.
>> root(r)
Incorrect number or types of inputs or outputs for function 'root'.
>> r=[-i -6.8 8.67i 58.956]
r =
0.0000 - 1.0000i -6.8000 + 0.0000i   0.0000 + 8.6700i 58.9560 + 0.0000i
>> roots(r)
ans =
0.0000 + 6.8000i
2.9445 - 0.0000i
-2.9445 - 0.0000i
>>

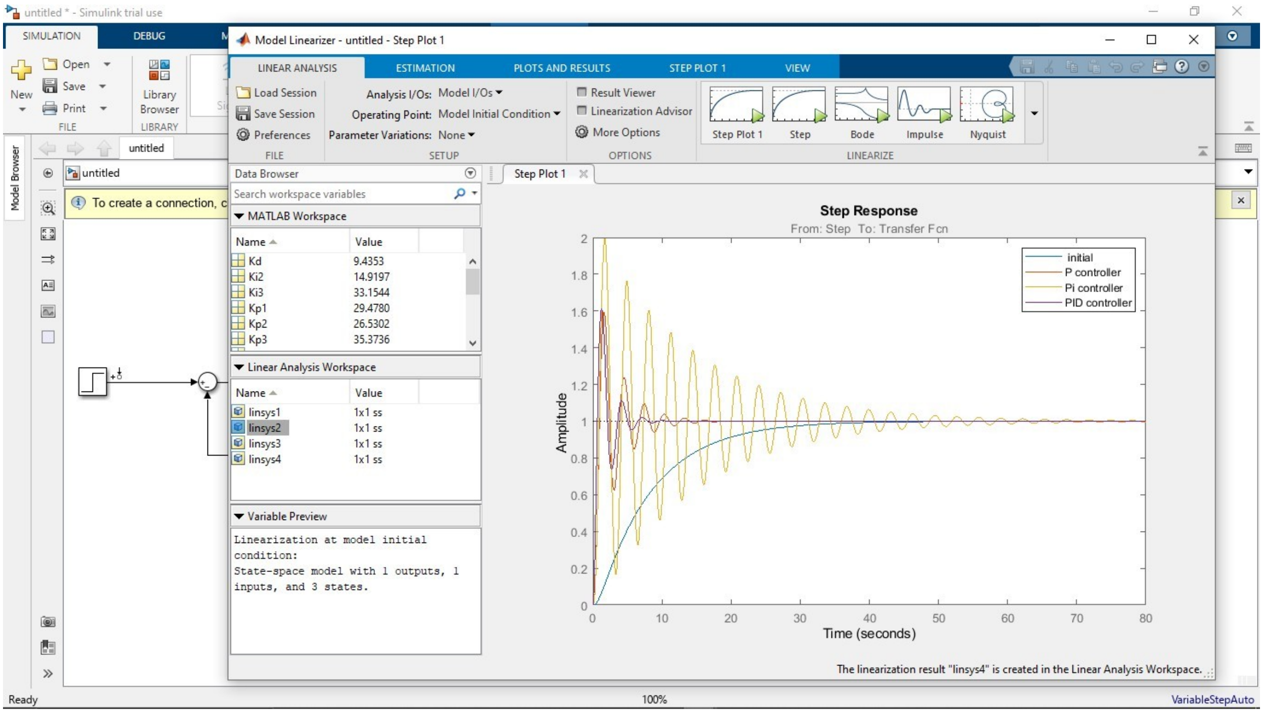

MATLAB R2022b - trial use
Command Window
New to MATLAB? See resources for Getting Started.
Did you mean:
>> pcr=2*pi/2.9445
pcr =
2.1339
>> Ti2= 1/1.2*pcr
Ti2 =
1.7782
>> Kp2=0.45*58.956


MATLAB R2022b - trial use
Command Window
New to MATLAB? See resources for Getting Started.
Kp3 =
35.3736
>> Ti3= 0.5*pcr
Ti3 =
1.0669
>> Ki3= kp3/Ti3
Unrecognized function or variable 'kp3'.
>> Ki3= Kp3/Ti3
Ki3 =
33.1544
>> Td3= 0.125*pcr
Td3 =
0.2667
>> Kd= kp3*Td3
Unrecognized function or variable 'kp3'.
>> Kd= Kp3*Td3
Workspace
ans [0.0000 + 6.8
Kd 9.4353
Ki2 14.9197
Ki3 33.1544
Kp1 29.4780
Kp2 26.5302
Kp3 35.3736
out 1x1 Simulatio
pcr 2.1339



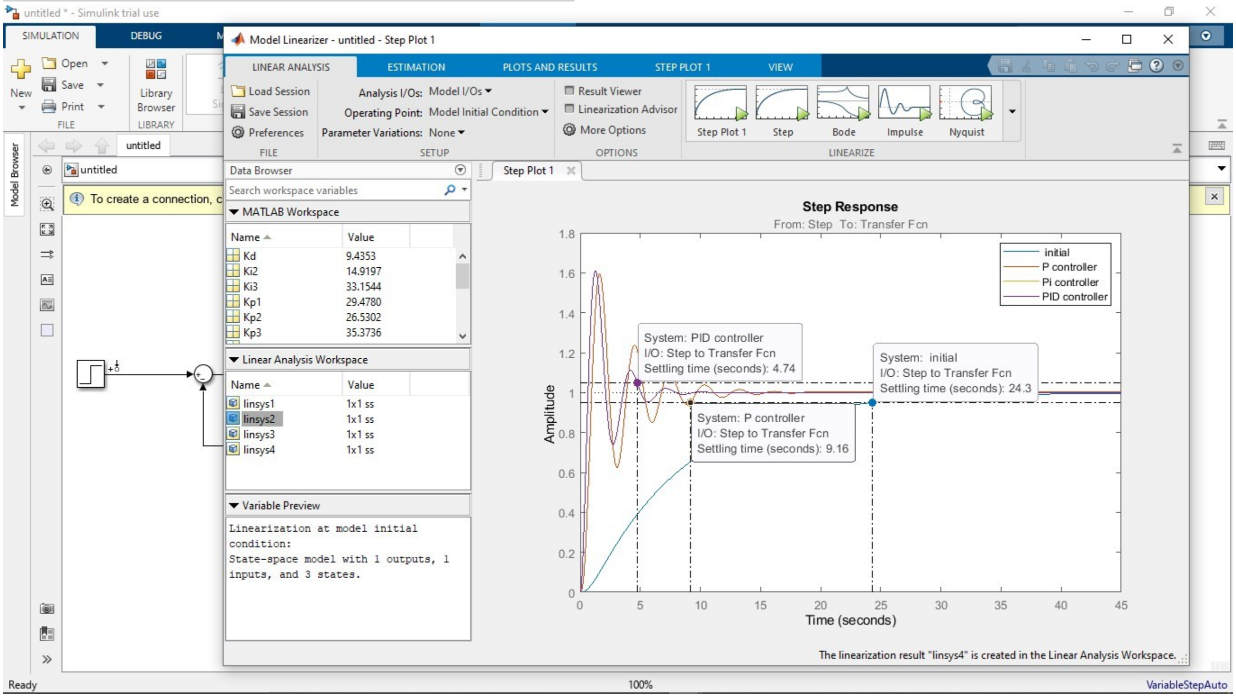
Type of Controller	K_p	T_i	T_d
P	$0.5K_{cr}$ =29.478	∞	0
PI	$0.45K_{cr}$ =26.5302	$\frac{1}{1.2}P_{cr}$ =1.7782	0
PID	$0.6K_{cr}$ =35.3736	$0.5P_{cr}$ =1.0669	$0.125P_{cr}$ =0.2667

WE NEED TO GET GRAPHS (WAVE) FOR ALL PID CONTROLLER BUT WE HAVE TO CHANGE ALWAYS (PID CONTROLLER) FOR PROPORTIONAL (P) TO KP1 THE FIRST, AND CHANGE THE (KP2 PROPORTIONAL, KI2 INTEGRAL) PID IS THE SECOND (PROPORTIONAL KP2, INTEGRAL KI3, DERIVATIVE KD)THE THIRD AND WE NEED TO CHANGE NAME LINSYS

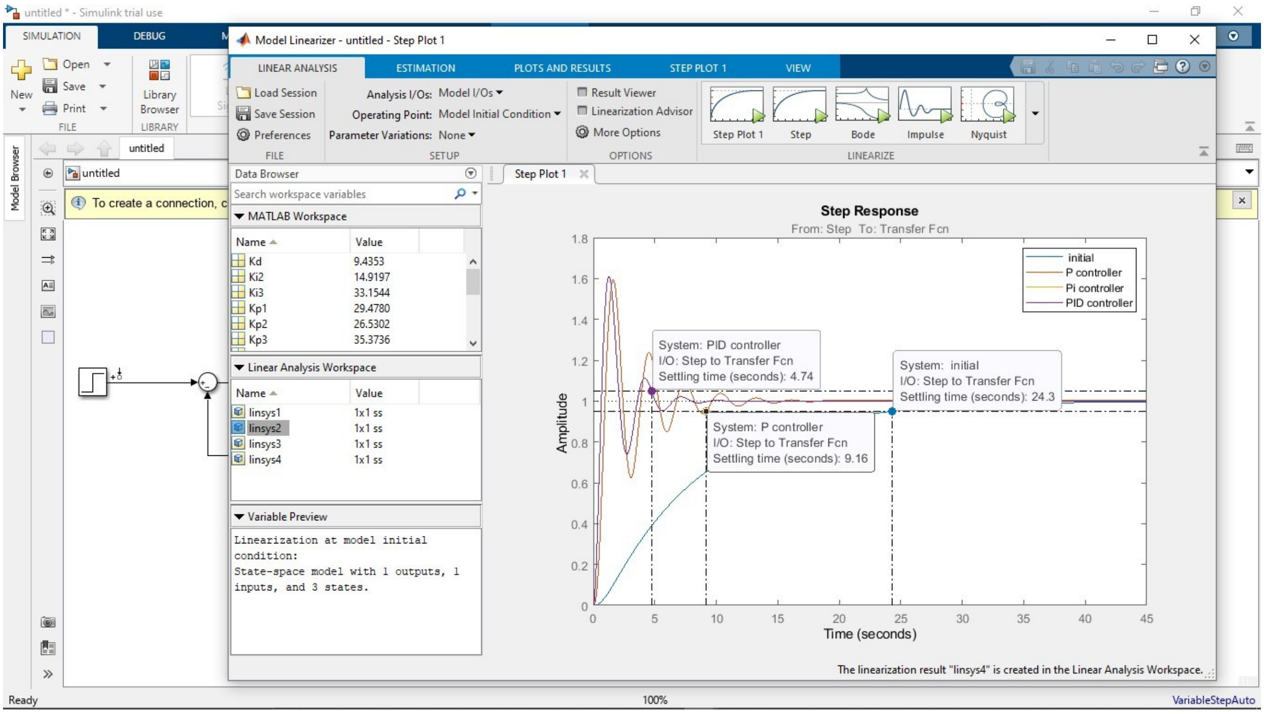


AND WE ADJUST THE SETTLING TIME WITHIN 5% LIKE FIRST GRAPH

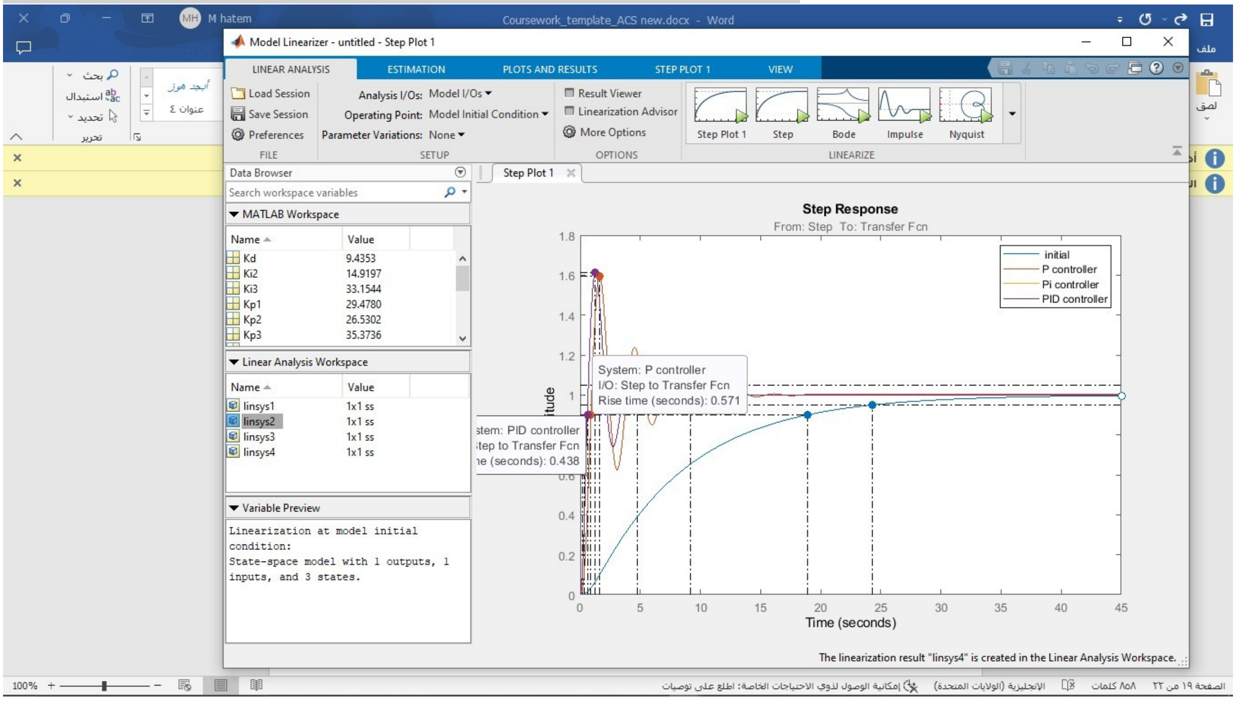
FIRST WE SHOW SETTLING TIME



THE SECOND WE WILL SHOW THE PEAK RESPONSE (OVER SHOT)



THE SECOND WE WILL SHOW THE RISE TIME:



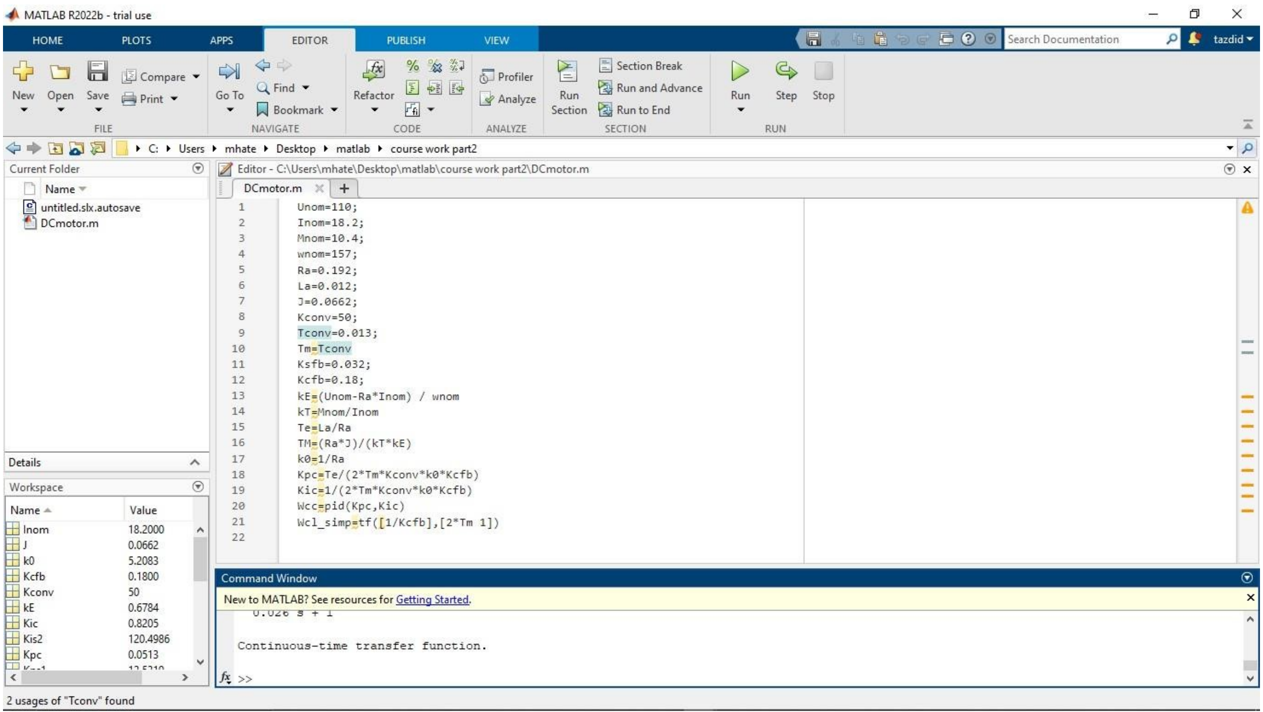
2. CALCULATION OF THE CASCADE CONTROL SYSTEM OF DC MOTOR

In the cascade control system, the internal current loop compensates the electromagnetic time constant TE, and the external one compensates for the electromechanical TM time constant. As the uncompensated (small) time constant Tμ, the time constant of the thyristor converter Tconv = Tμ is taken.

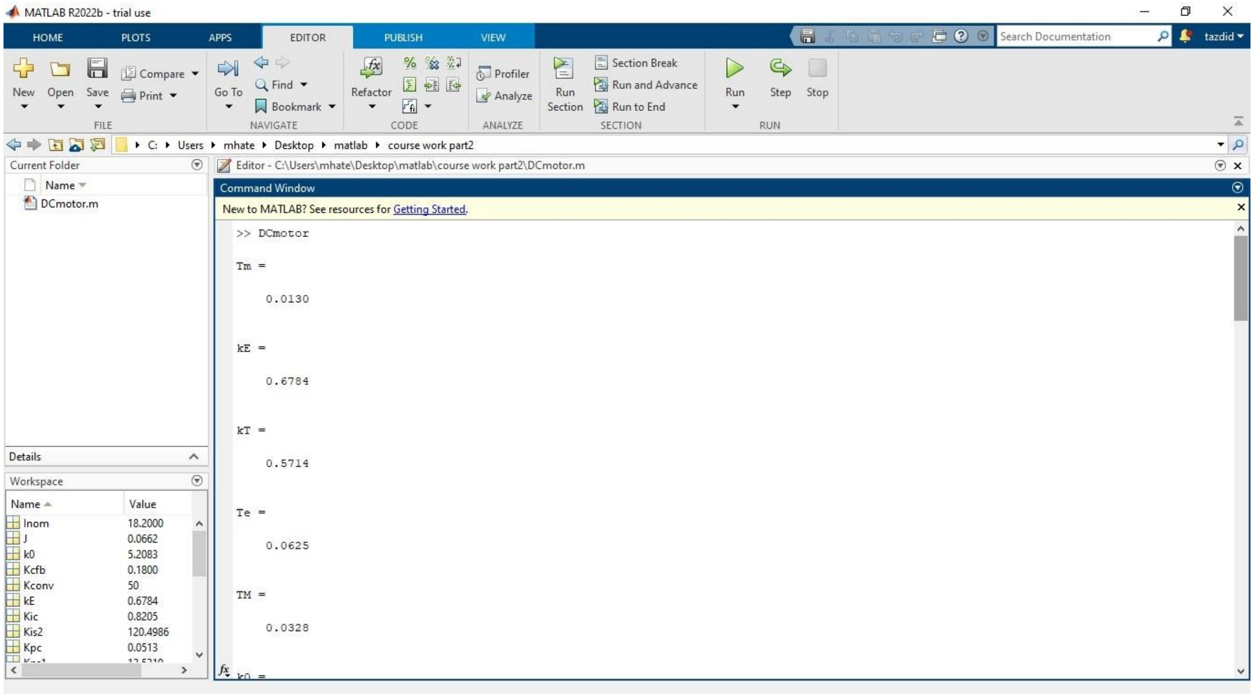
Set the internal current loop to a modular (technical) optimum (MO)

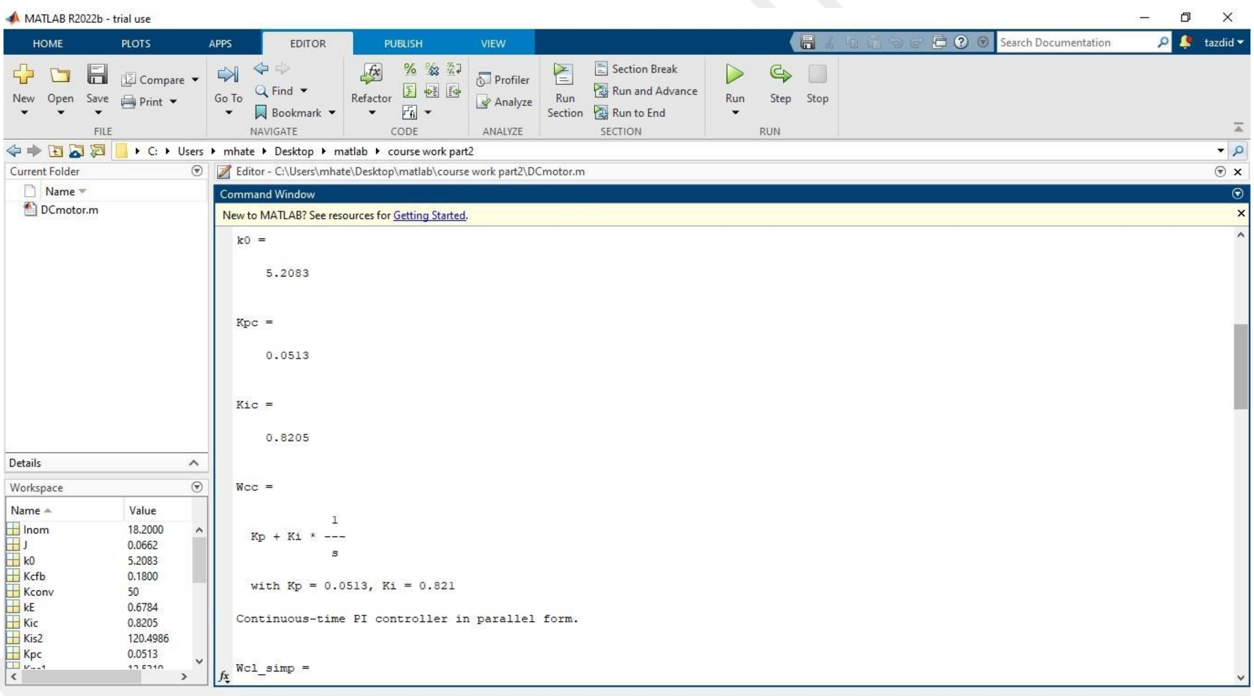
№	U_{nom}	I_{nom}	M_{nom}	ω_{nom}	R_a	L_a	J	K_{conv}	T_{conv}	K_{Sfb}	K_{Cfb}
28	110	18,2	10,4	157	0,192	0,012	$662 \cdot 10^{-4}$	50	0,013	0,032	0,18

(1) First we open new script:



(2) Sconde we save this file and change name to (DCmotor) take carful must be without spece and press run:


MATLAB R2022b - trial use
Editor - C:\Users\mhate\Desktop\matlab\course work part2\DCmotor.m
Command Window
>> DCmotor
Tm =
0.0130
kE =
0.6784
kT =
0.5714
Te =
0.0625
TM =
0.0328
Workspace
Inom 18.2000
J 0.0662
k0 5.2083
Kcfb 0.1800
Kconv 50
kE 0.6784
Kic 0.8205
Kis2 120.4986
Kpc 0.0513

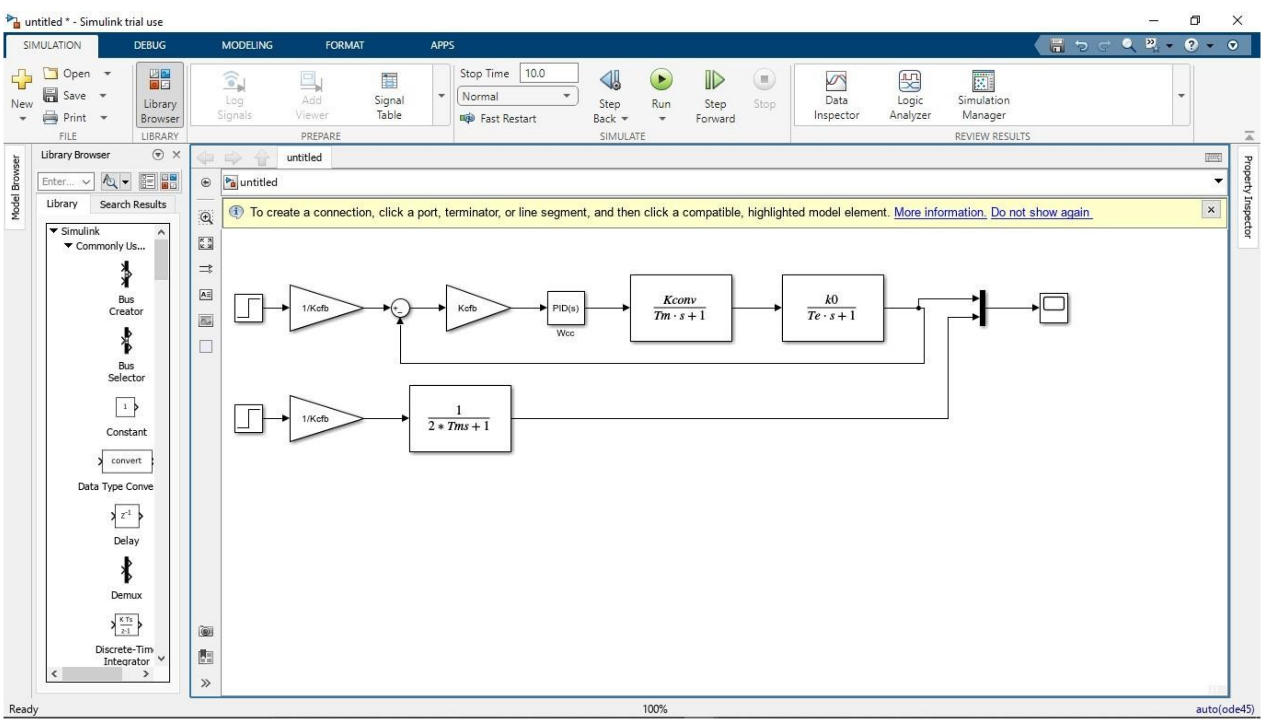

MATLAB R2022b - trial use
Command Window
k0 =
5.2083
Kpc =
0.0513
Kic =
0.8205
Wcc =
Kp + Ki * 1/s
with Kp = 0.0513, Ki = 0.821
Continuous-time PI controller in parallel form.
Wcl_simp =
Workspace
Inom 18.2000
J 0.0662
k0 5.2083
Kcfb 0.1800
Kconv 50
kE 0.6784
Kic 0.8205
Kis2 120.4986
Kpc 0.0513


Wcl_simp =
5.556

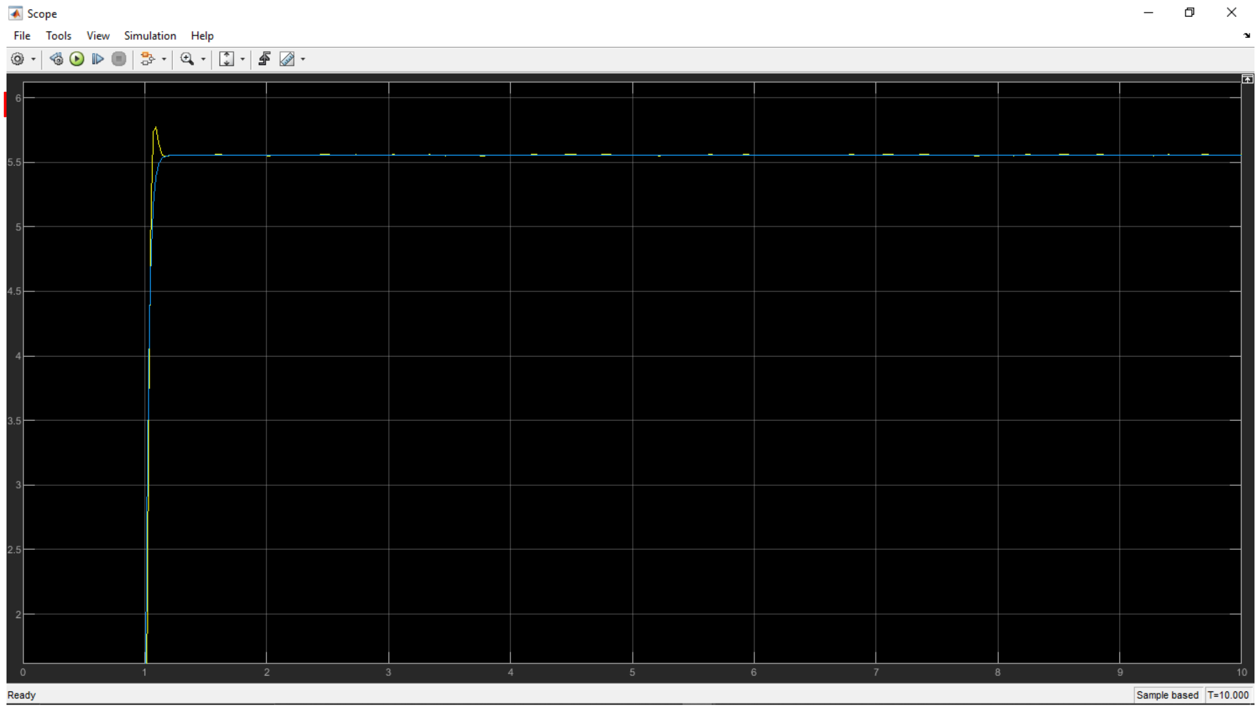
0.026 s + 1
Continuous-time transfer function.
Inom 18.2000
J 0.0662
k0 5.2083
Kcfb 0.1800
Kconv 50
kE 0.6784
Kic 0.8205
Kis2 120.4986
Kpc 0.0513

And we are open the (Simulink) to get the graph and first we need to put the :1-step ,2-sum,3-PID controller,4-transfer Fcn,5-Gain,6-scope. And change the (sum)>>lsit of sign(+-),in the PID Controller change(propritonal:Kpc),(integral:Kic), In the transfer Fcn change (numerator:Kconv),(Donominator:[Tm 1]), add Gain 0 and cheang the volume to:1/Kcfb,add new one and change the volume to :Kcfb and change name PID Controller to (Wcc),add new one transferFcn(numerator:[K0]),(Donominator:[Te 1]) and we need to add the (Mux) external loopto connecting

transferFcn(numerator:[1]),(Donominator:[2*Tm 1]) even got the Technical optimum or maximum production level (TOL) (Technical optimum) and full function



And press the run >> open the scope to see the difference and the difference



The first function is inner Loop & Second function is External Loop we have 2

coefficient $\frac{1/K_{cfb}}{2T_m(T_m s+1)+1} =$ for first function inner loop, and we have

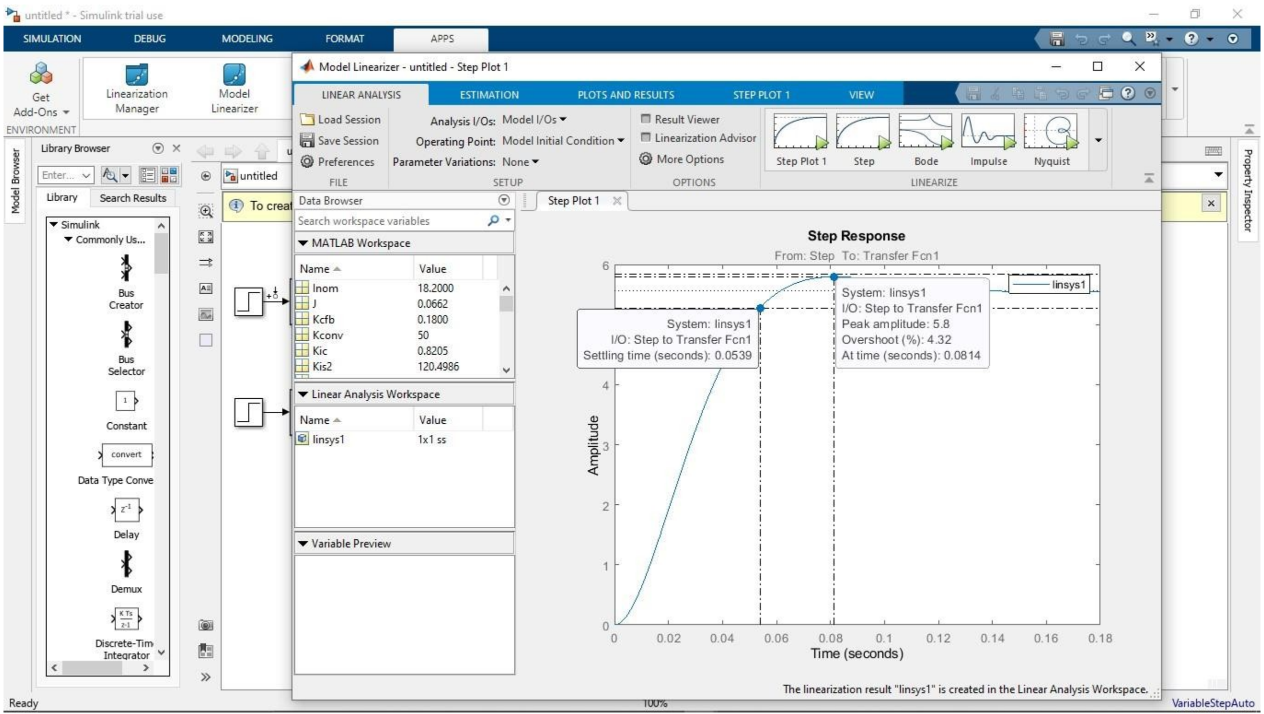
othe coefficient $2T_m^2 s^2$ this is vary small Because $T_m << 1$ so the m .

The first function is inner Loop & Second function is External Loop we have 2

And we need to add (output measurement),(input perturbation) and open

model Linerzer to see the Graph but first going to Property to ADJUST THE

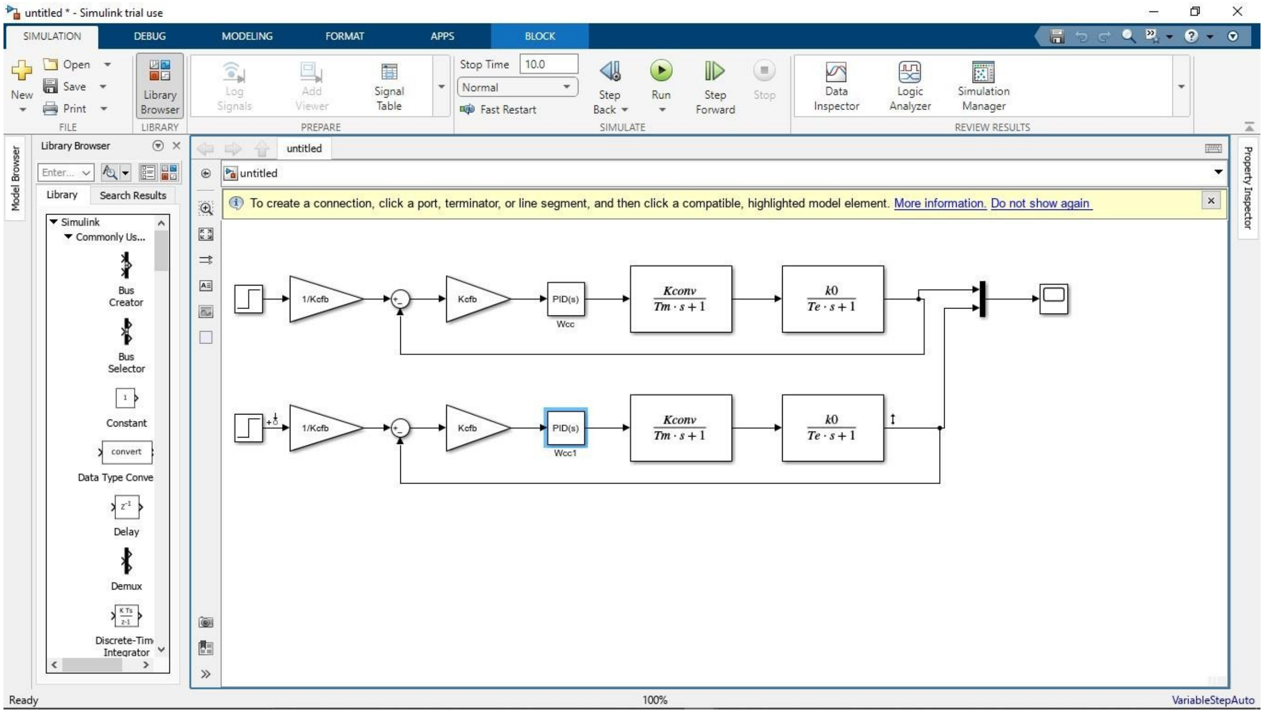
SETTLING TIME WITHIN 5%:

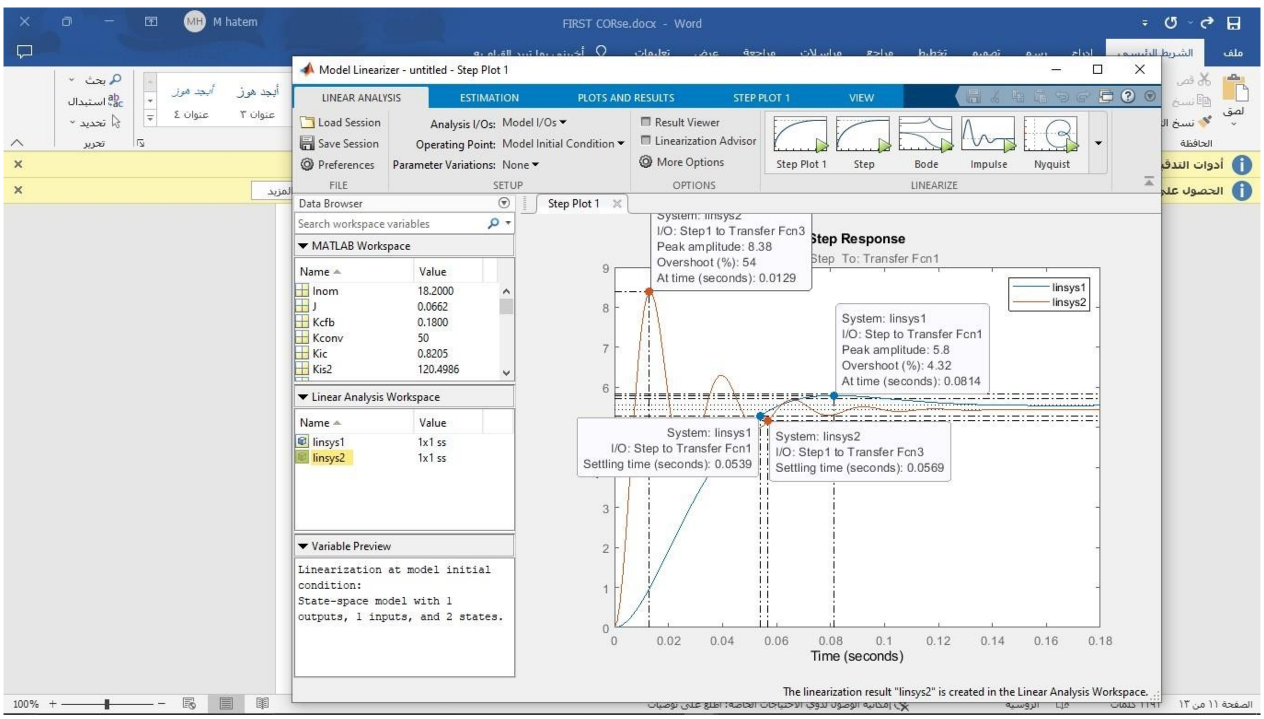


The overshoot in this case is σ = 4.3% in all system to modular optimum have 4.3% overshoot

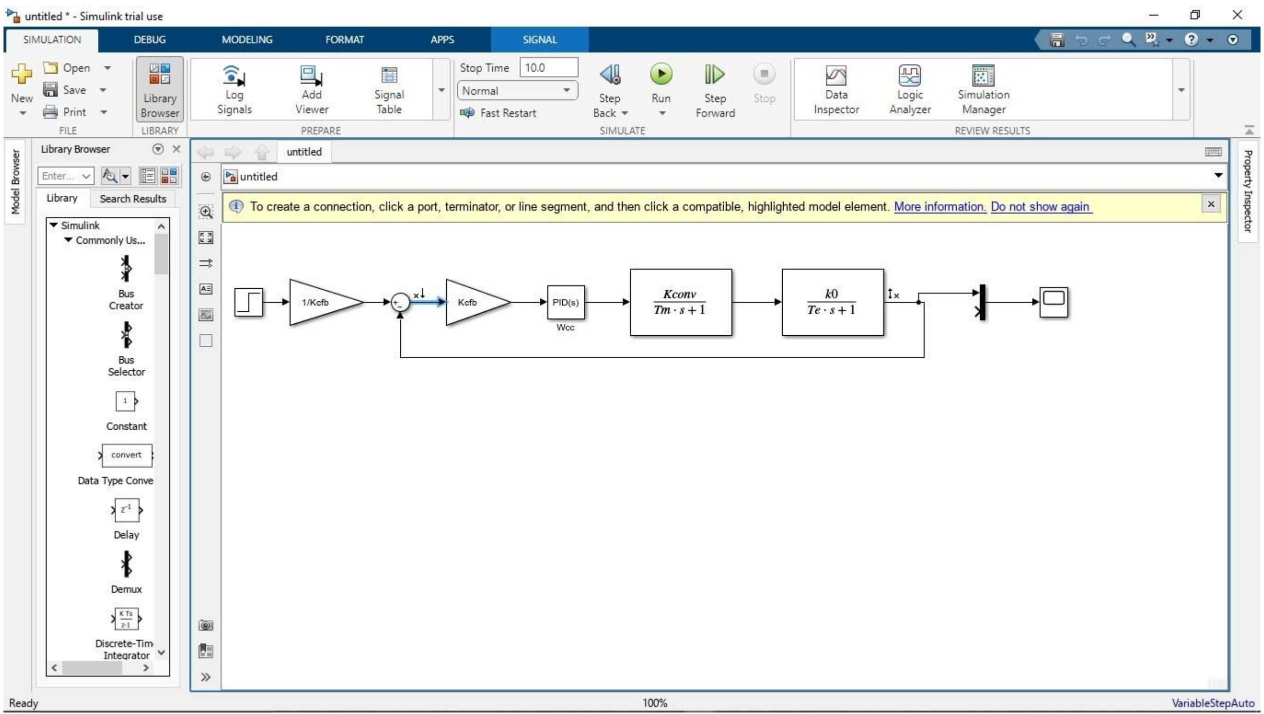
---

The initial data: The power part of the electric motor is a thyristor converter system – a DC motor with independent excitation We need compared with initial system will be the system without PID controller and we change the (proportion(p):1),(integral:0)and remove ones from (input ,output) because input and output must be one:

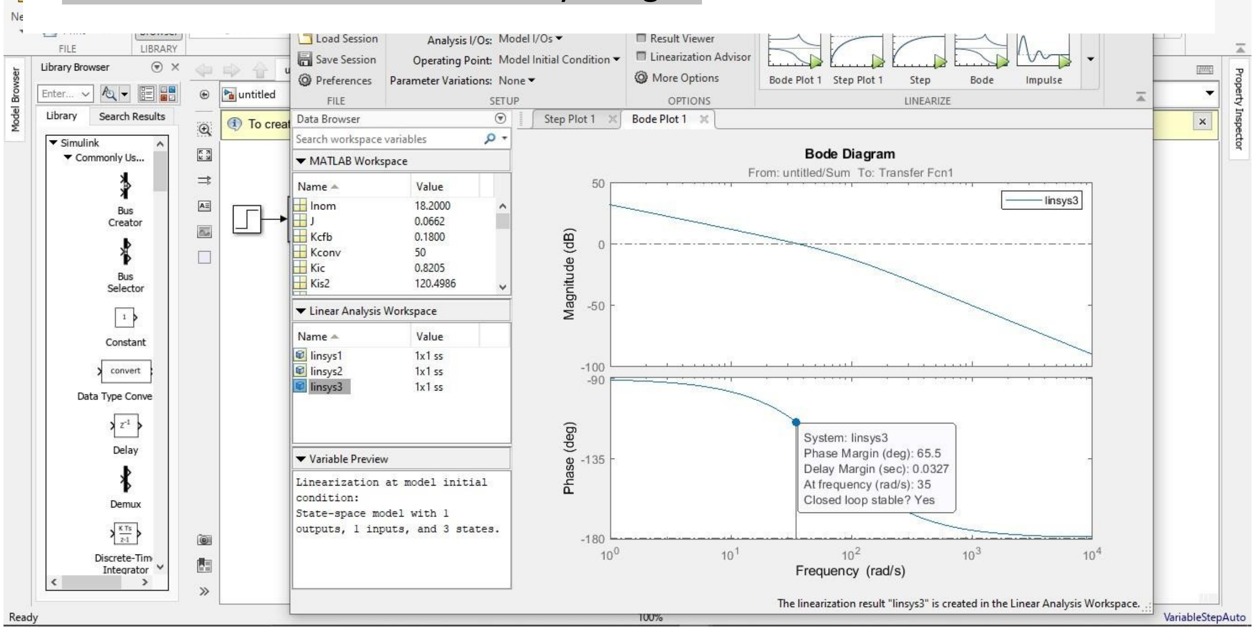




We need bode function the bode function be open-loop system and put open-loop system outbut and open-loop system input

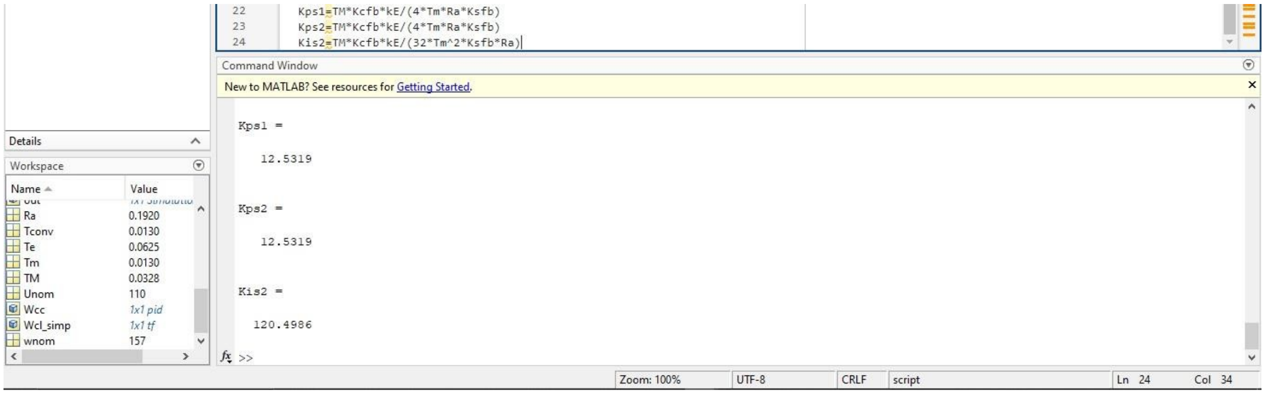


We need to showe the all stability margins

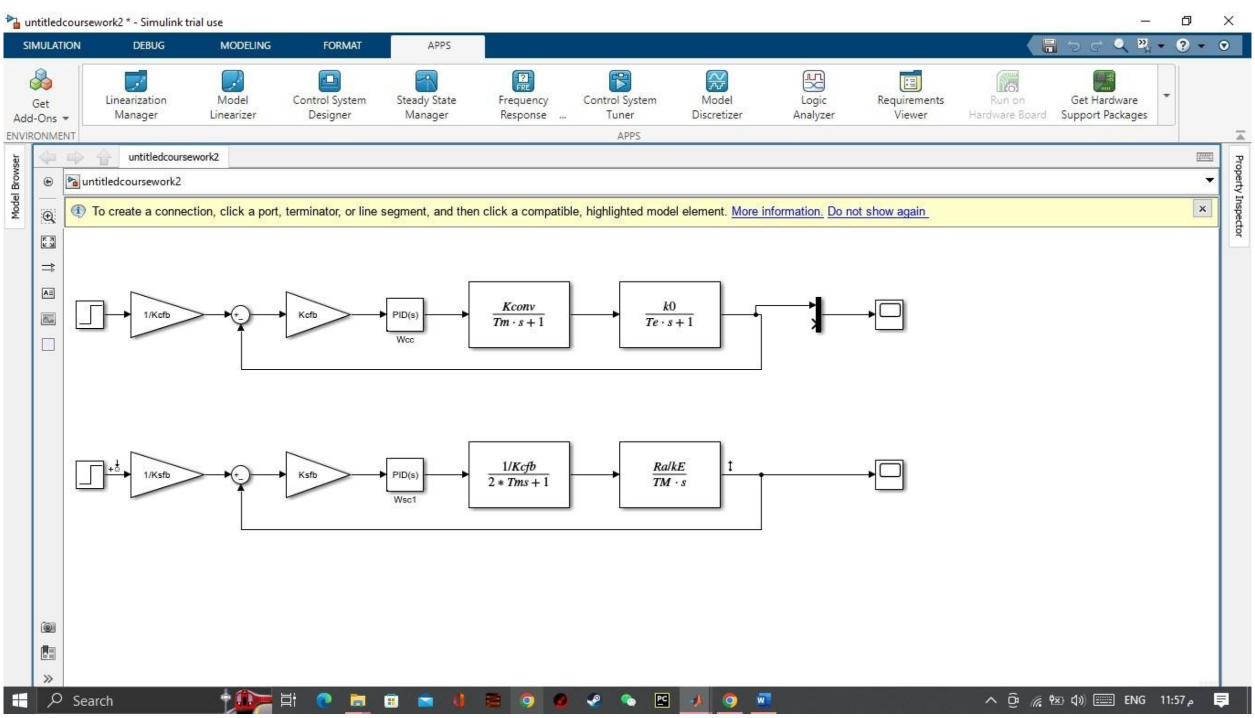


he system is stable

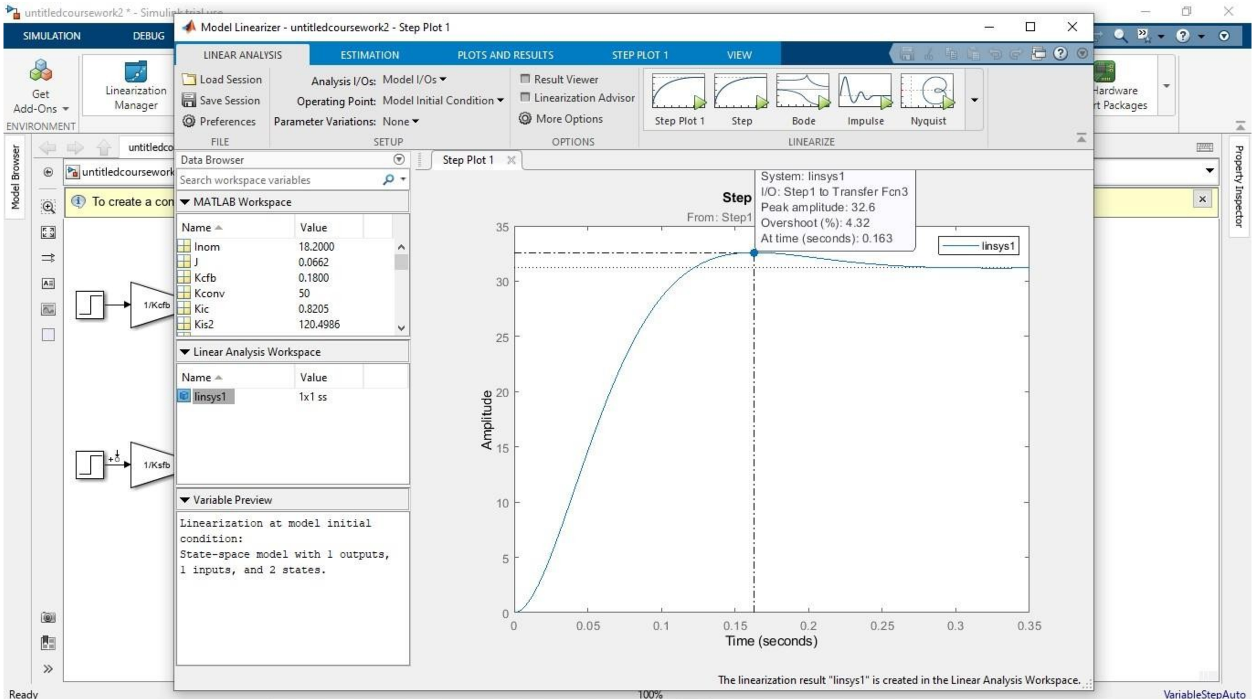
And we continue the speed controller tuned technical for the symmetrical optimum: we are are add some coefficient in the in matlab script:



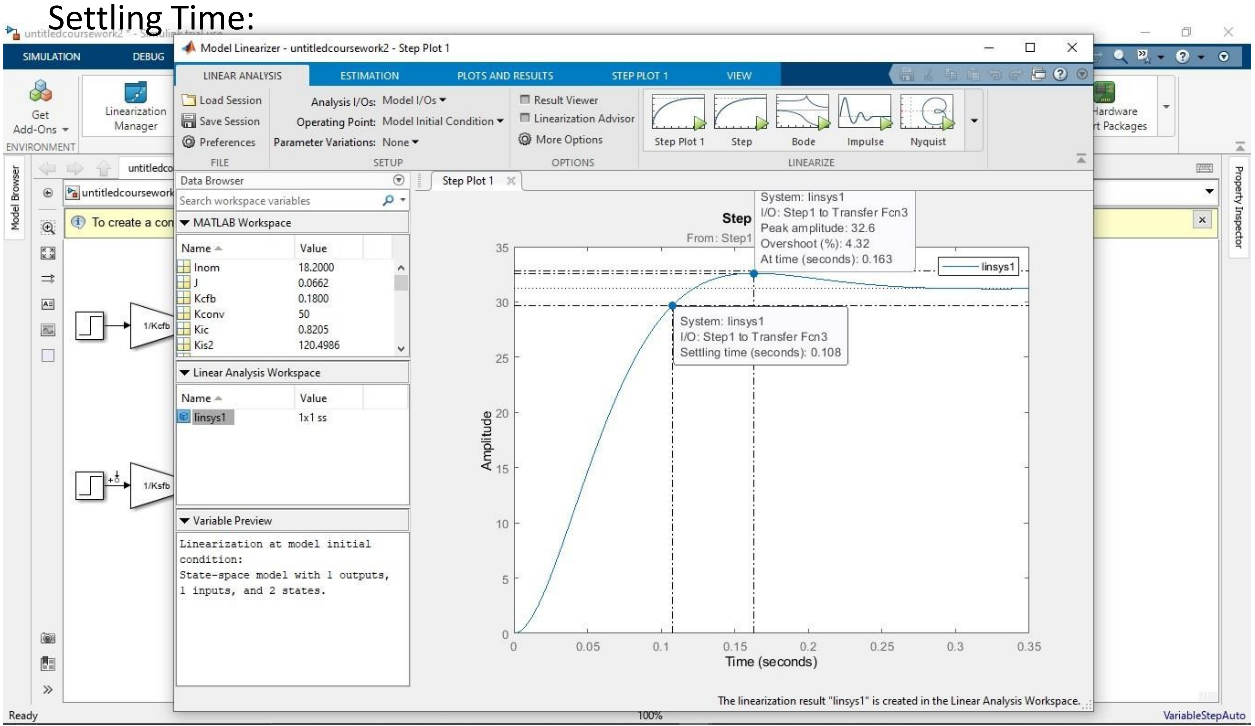
Before start we must delete the open-loop system(input,output) make new system and change the Value of (Gain2 = 1/Ksb),)Gan3=Ksfb),in the PID controller change the (propration=Kps),(integral=0) change name to (Wsc),in TF_2(Numerator=[R_a/kE]),(Donominator=[T_M 0]) should be (Input Preturbation ,Output Measurment):



Going to Analysis open new function (step) and we need to see the peak respone

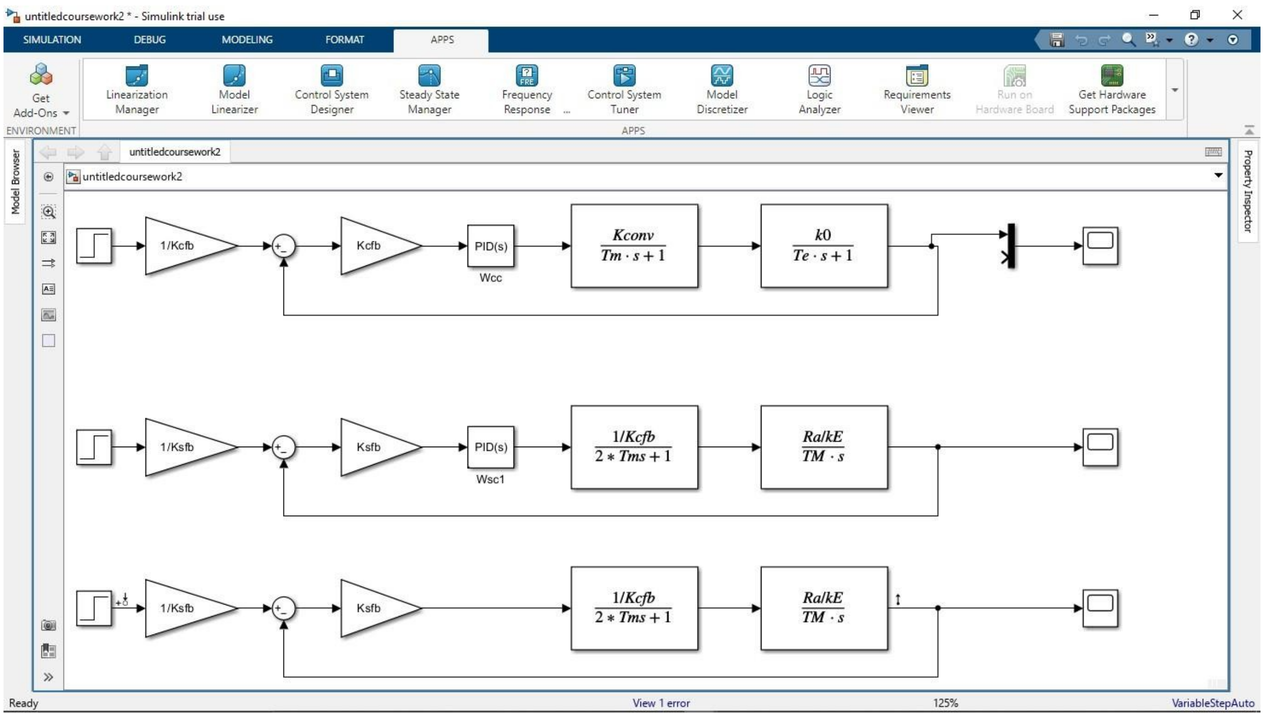


the overshoot always 4.32% because this is technical optimum and going to Property to ADJUST THE SETTLING TIME WITHIN 5% to show the Settling Time:

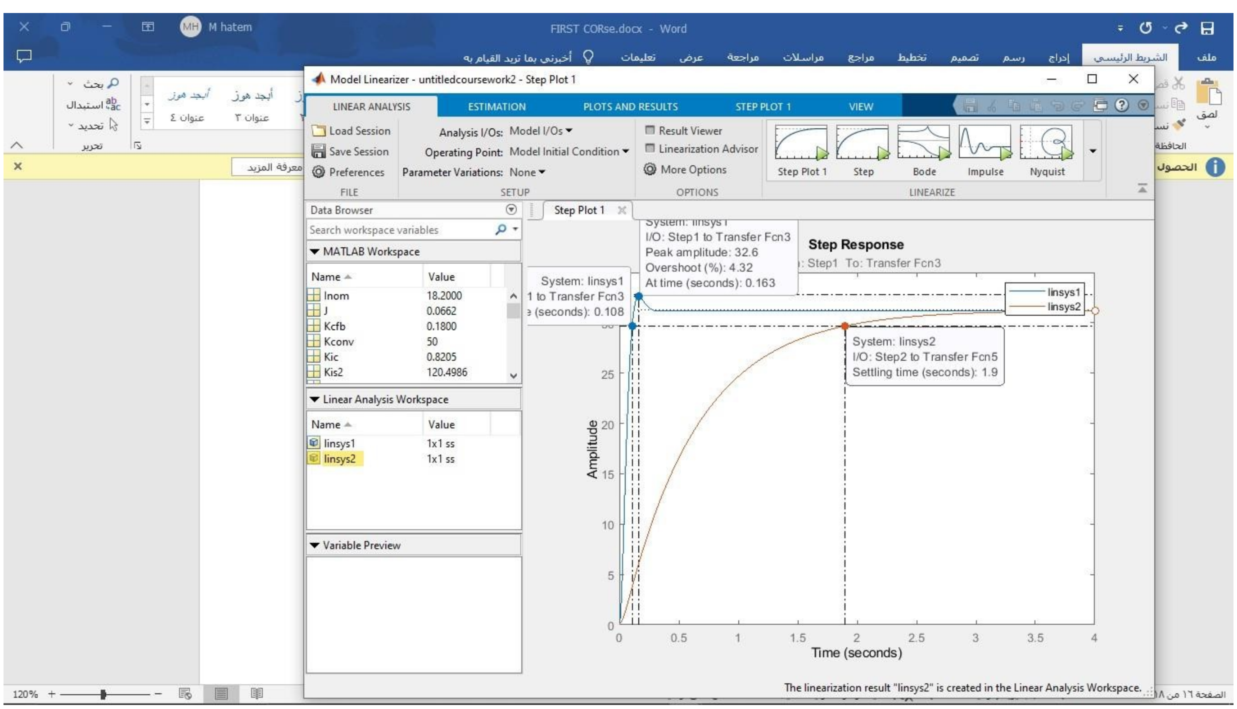


The settling time be 2 times bigger than currentloop because time of the firststeady-state value achievement=4.7 T_m and here theres coefficient= 2*T_m

we can compare system without controller (PID):



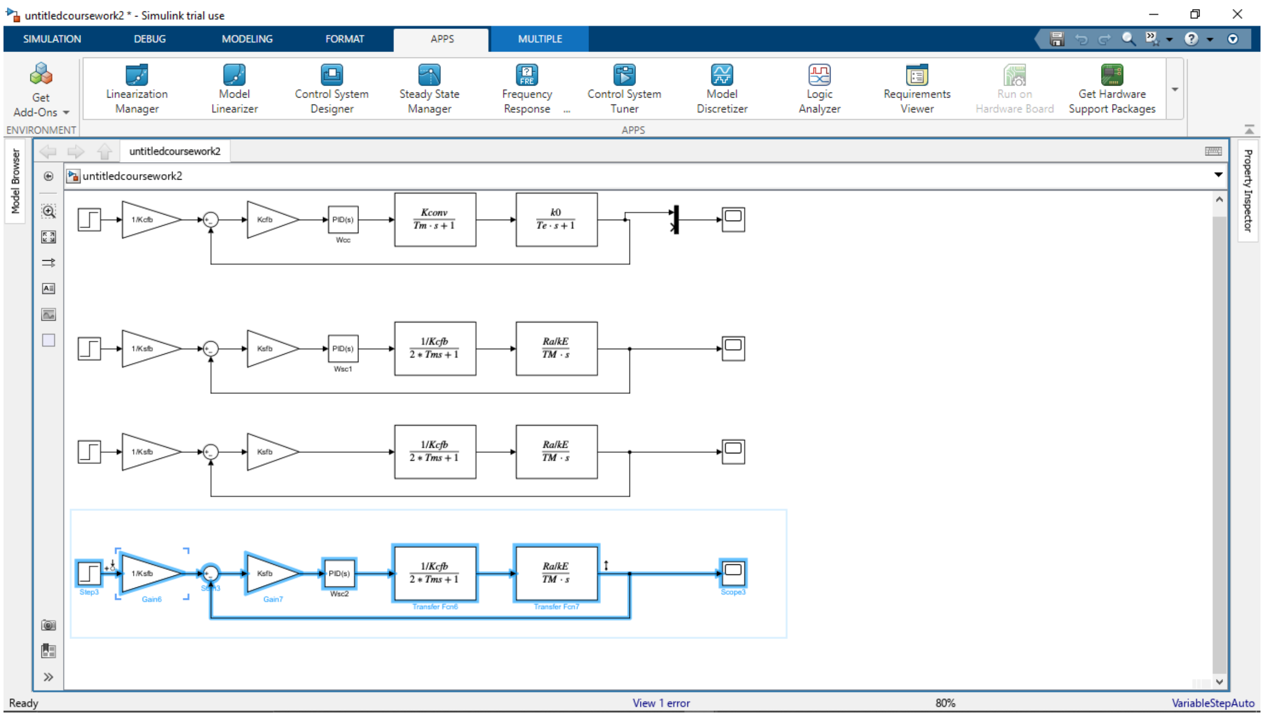
This is initial system (linsys1=Technical optimum) without Controller(PID) forSpeed

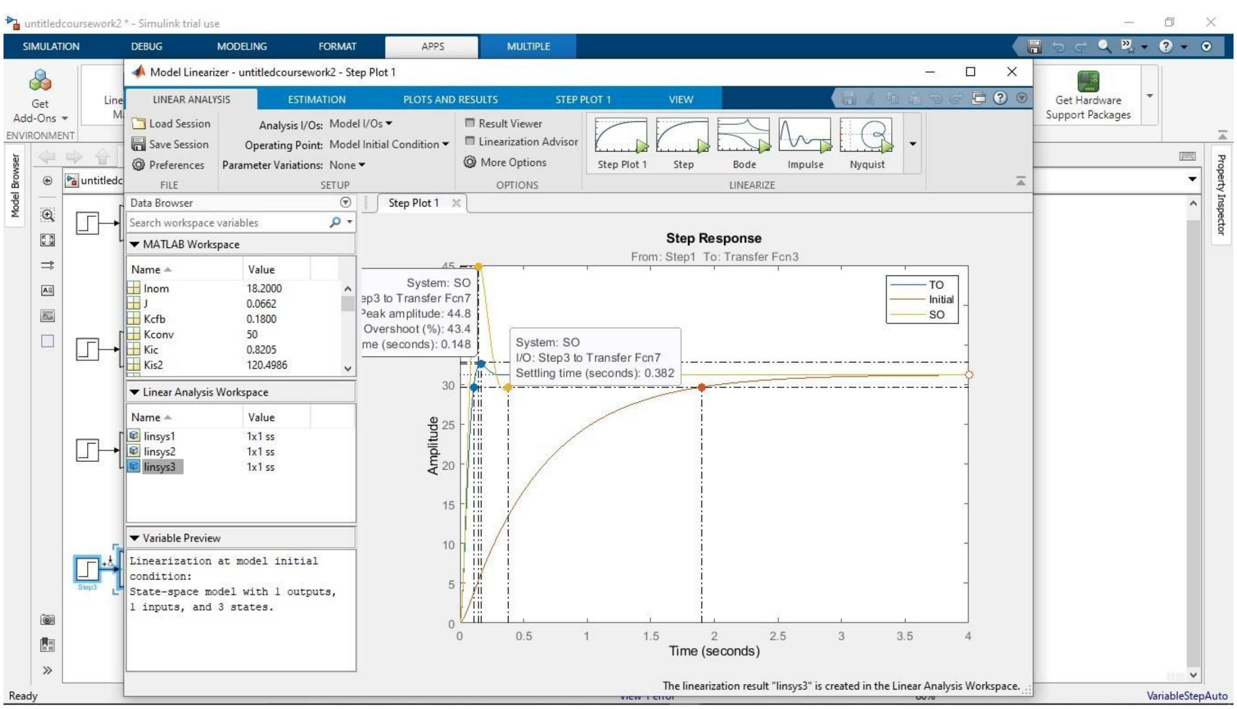


The (linsys1= Technical Optimum) is tuned controller to (technical optimum) not the (symmetric optimum)

---

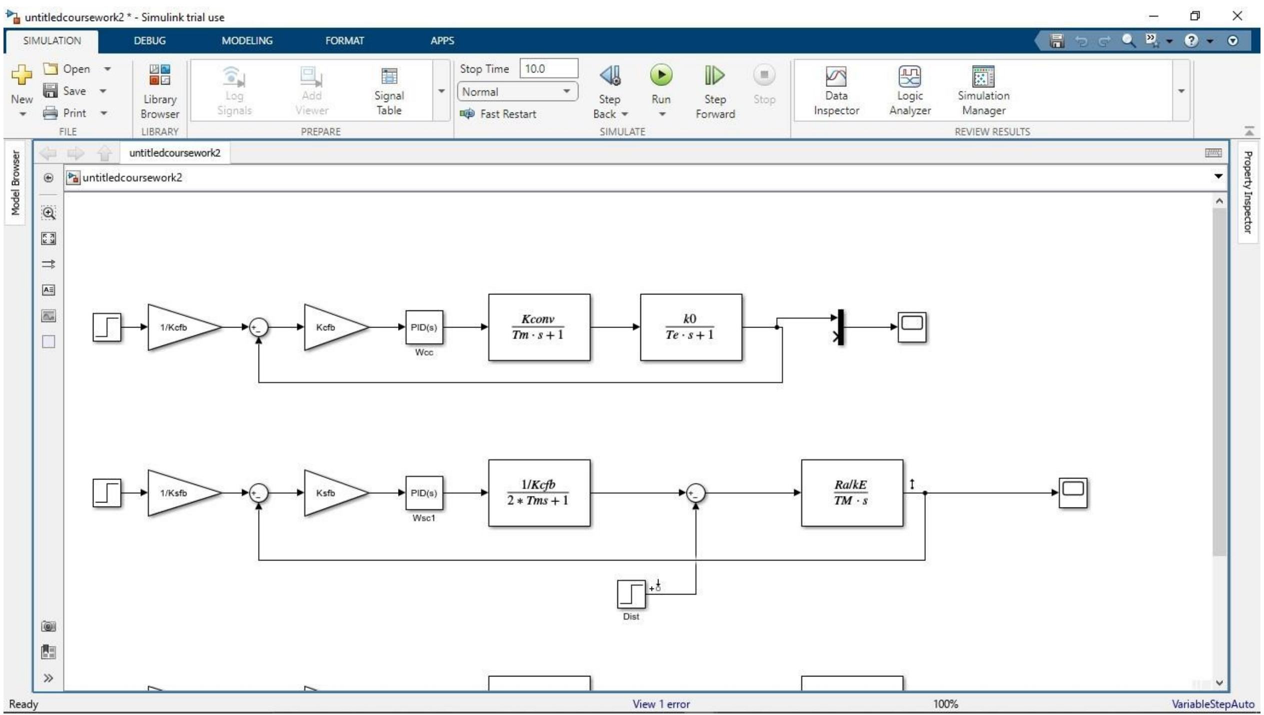
and we add new one loop for (symmetric optimum) in the symmetric optimum we add (PID)controller ((P)=Kps2),((I)=Kis2):



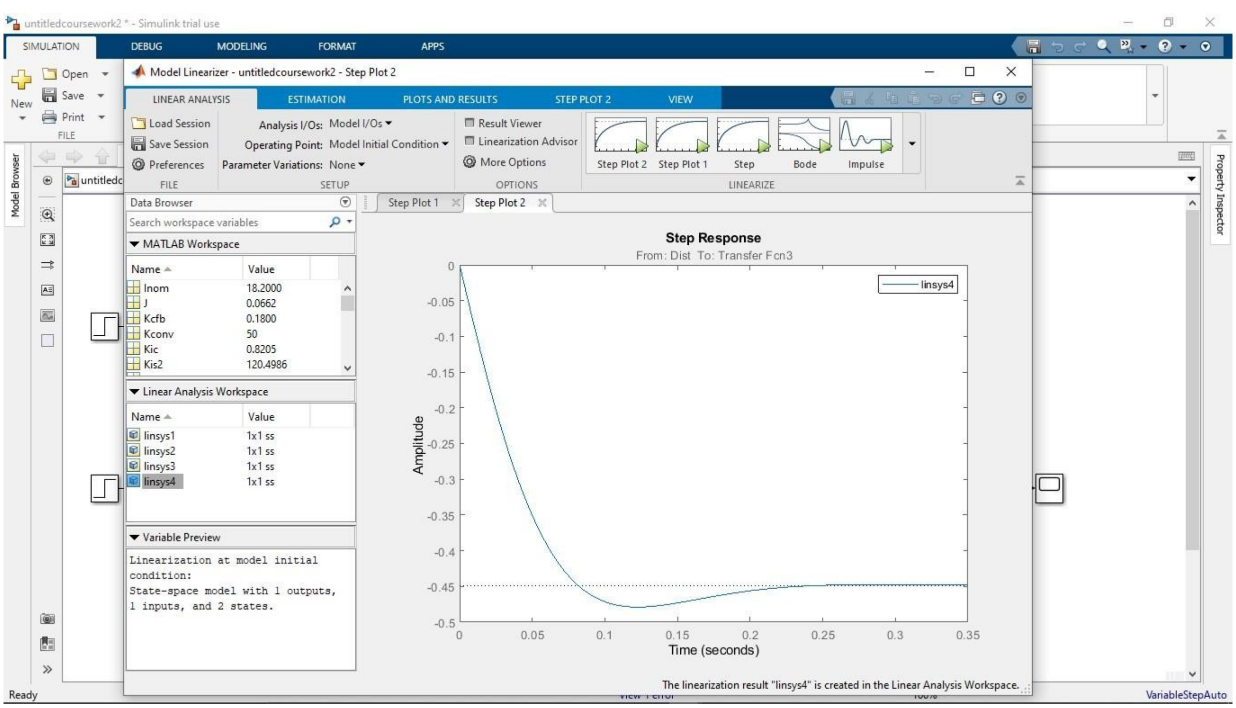


This is (SO= Symmetric Optimum)

We do new one for Disturbance(Dist) we will add the othe one from (sum) and other one form (step) but we change it to (Dist) bit. by disturbance be closed loop. But we will do somthing different in (input) we put it in the (Dist).:

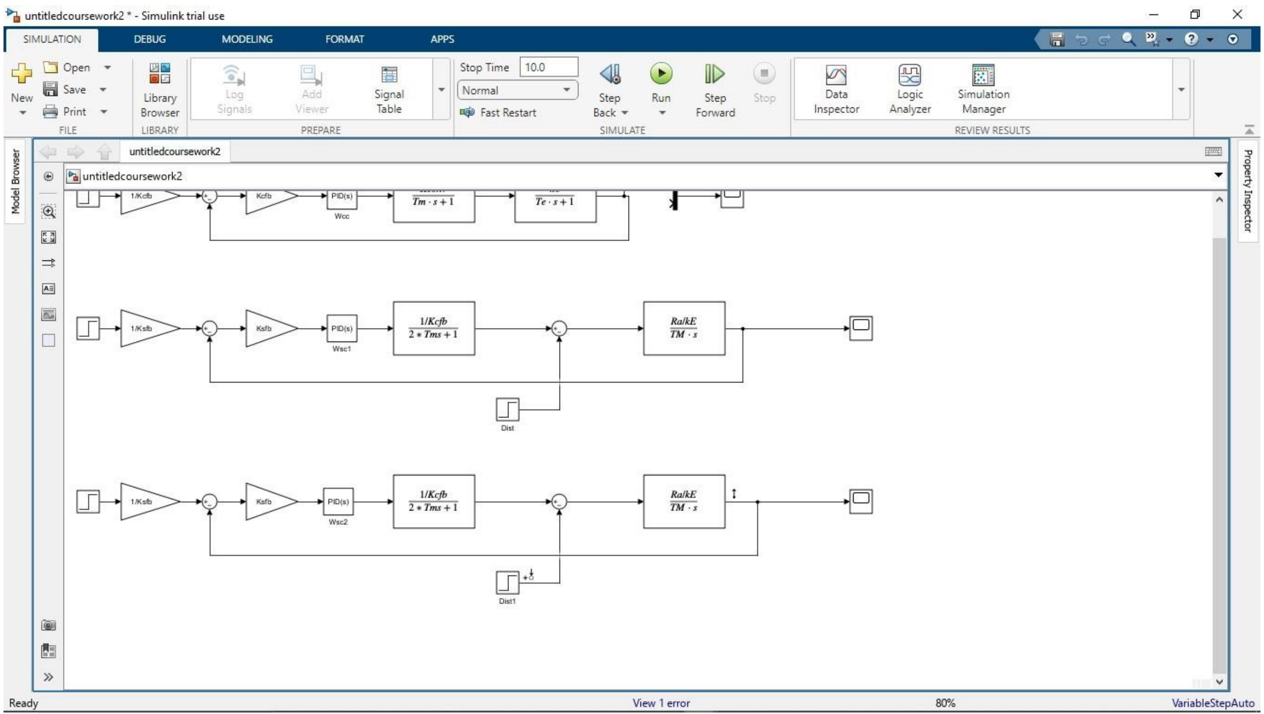


We can se that's >>

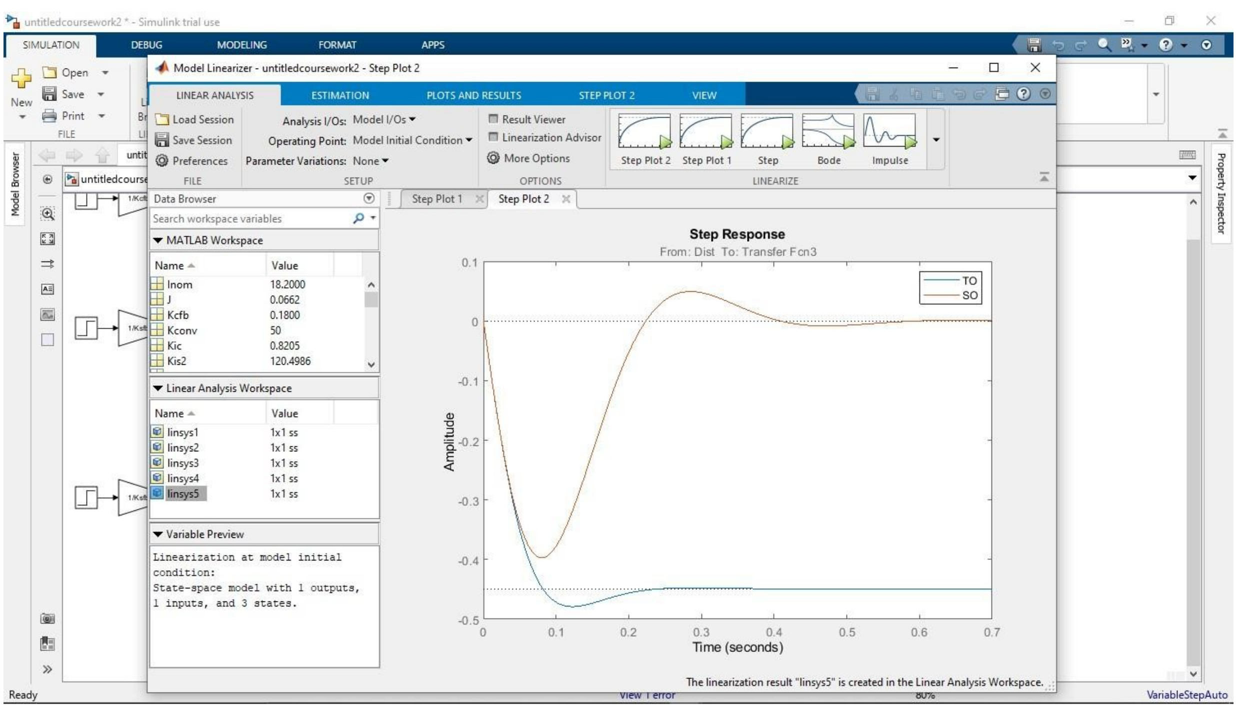


It’s a Technical Optimum(TO) is start from (0) after that we will get (ERROR) we have non (0) as a result

We need to put the Symmtric Optimum(SO):

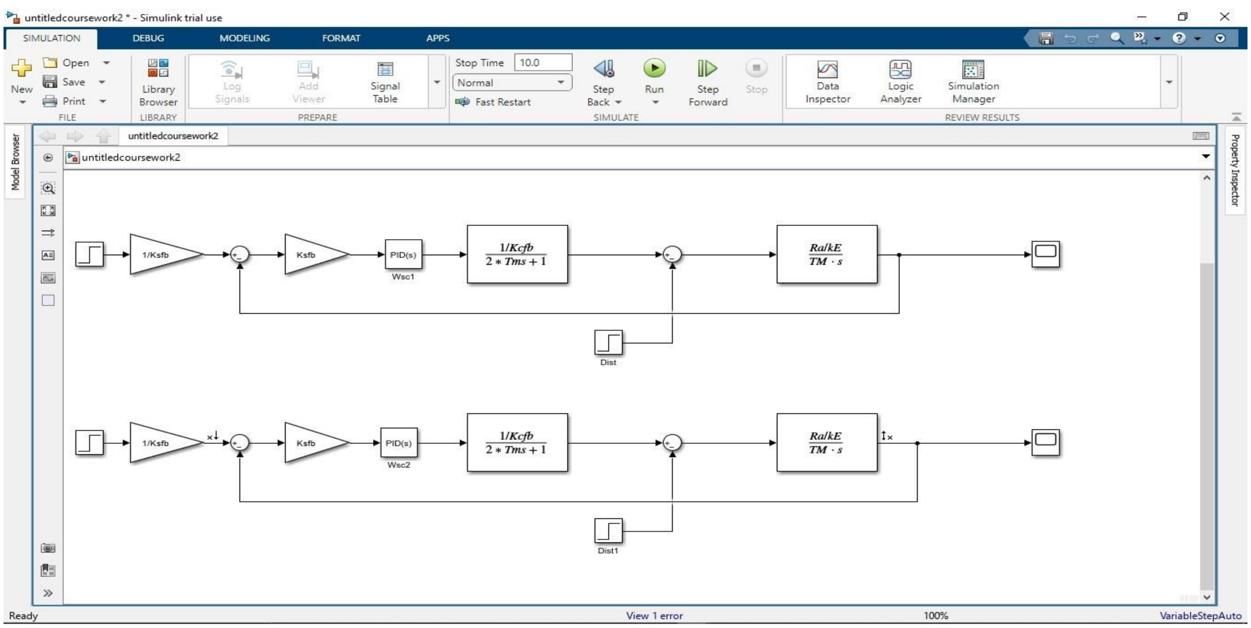


In the Symmtric Optimum( ) his a Disturbance comes to back to 0

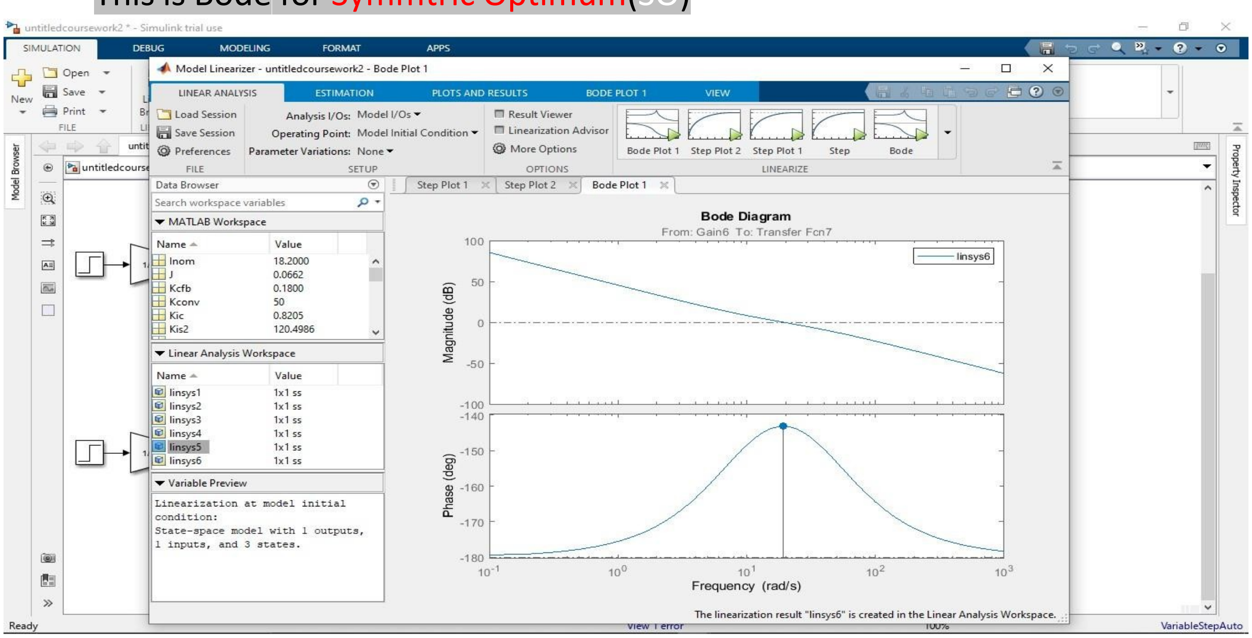


We will see difference between Technical Optimum(TO) & Symmtric Optimum(SO), Because It's a Technical Optimum(TO) is start from (0) after that we will get (ERROR) we have non (0) as a result, and the when Symmtric Optimum(SO)get the statics error comes to 0. The characteristics is better for Technical Optimum(TO) but id speech about the disturbance signals we should Symmtric Optimum(SO).

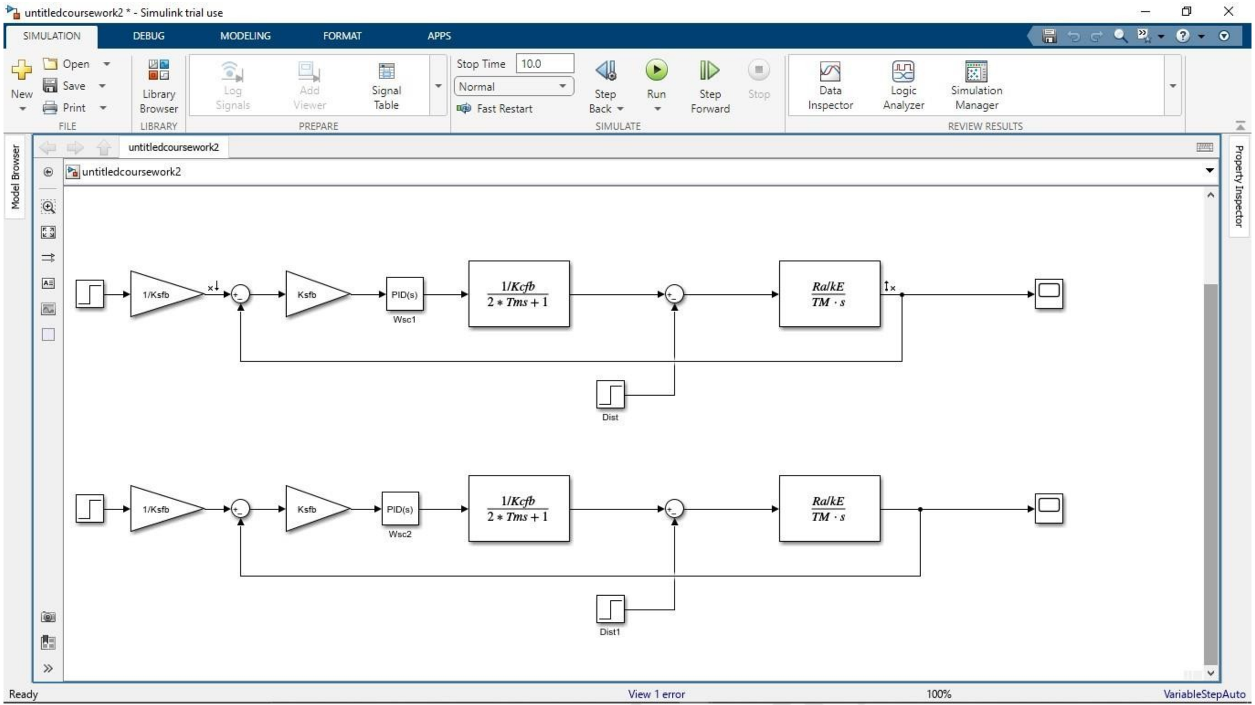
And the last one we need the (Bode) but we need to change the (Output &Input) to (Open-Loop Outpu&Input):

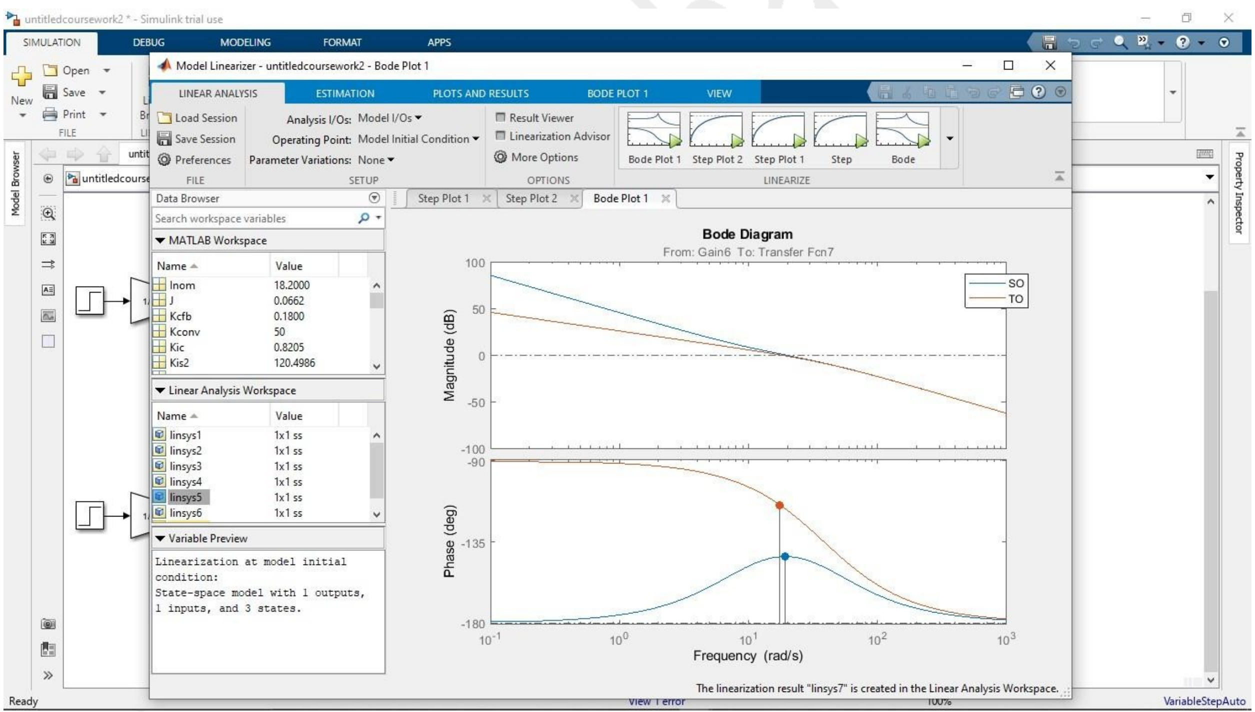


This is Bode for Symmtric Optimum(SO)

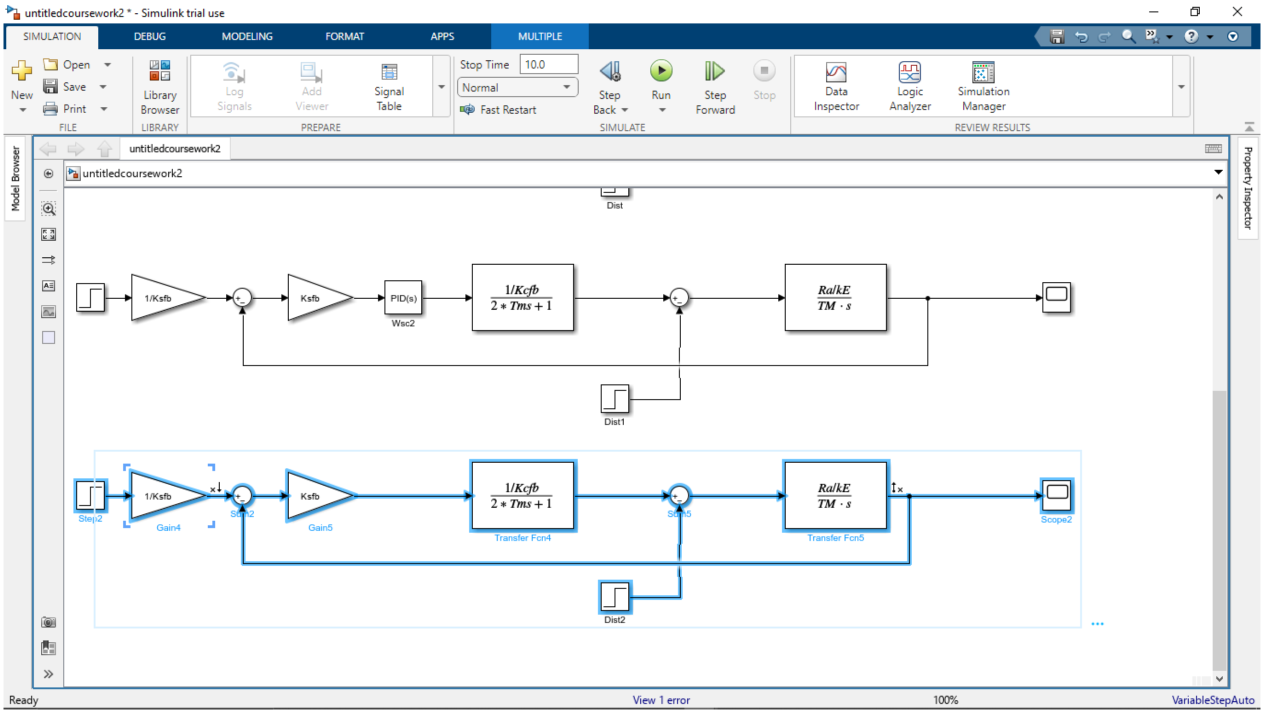


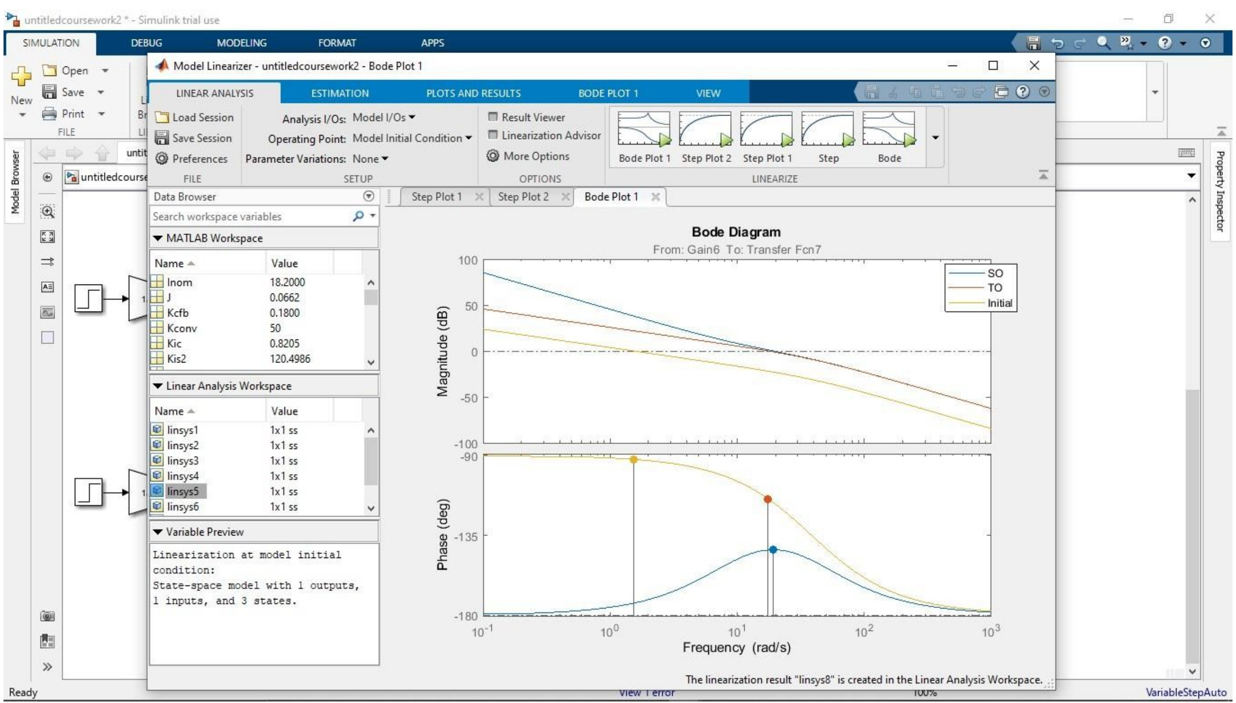
For the Technical Optimum(TO):





For The Initial System:





CONCLUSION

- The advantage of the Ziegler-Nichols method is **that the tuning rules are very simple to use. Disadvantages are: Further fine tuning is needed. Controller settings are aggressive, resulting inlarge overshoot and oscillatory responses.**
- Advantages of cascade control **A faster inner loop can respond more**

quickly to disturbances than the outer loop. Therefore, it reduces the severity of disturbances and limits variability that would affect the heating process The proposed cascade control system is **designed and implemented using a SIMULINK to track the motor speed and determine the armature current under theno load case**.

REFERENCES

- Fortmann, Thomas E. An Introduction to Linear Control Systems (Control and System Theory) / Fortmann, Thomas E., Hitz, Konrad L. Publisher: CRC Press,1977.
- Distefano, J. Schaum's Outline of Feedback and Control Systems, 3rd Edition (Schaum's Outlines) / Joseph Distefano III; Allen R. Stubberud; Ivan J. Williams. Publisher: McGraw-Hill Education, 2013.
- Williams II, Robert L. Linear State-Space Control Systems / Williams II, Robert L.; Lawrence, Douglas A. Publisher: John Wiley & Sons, 2007.
- Norman S. Nise. Control Systems Engineering / Norman S. Nise. Publisher:Wiley, 2015.