PES UNIVERSITY

(Established under Karanataka Act No. 16 of 2013)
100 Feet Ring Road, Bengaluru, Karnataka, India — 560 085

Report on

PID tuning using extremum seeking: online, model-free performance optimization,

By Nick J. Killingsworth and Miroslav Krstic´

Submitted by

Vittal Srinivasan (PES1201700310)

Jan — May 2020

STUDENT OF ENGINEERING
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
PROGRAM B.TECH.

ABSTRACT

This paper presents a way to optimize step response of closed loop systems with PID controllers.The parameter tuning is done using discrete version of Extremum Seeking algorithm. An adaptive tuning is performed where the parameter values and output are changed with respect to the previous output and PID values.The paper focuses on four different systems and compares the extremum seeking output with that of Zeigler-Nichols, Iterative feedback tuning and Internal Model Control methods.Discussions on cost function comparisons, control saturation and techniques to select parameters for tuning ES are discussed. Simulations for all cases are given.

Contents

1 Introduction

PID controllers are used nowadays extensively in the process industry but their effectiveness is limited due to poor tuning techniques.Methods of manual tuning are that followed are very time consuming. Other systematic methods rely on the plant, knowledge of the model or require specific experiments to determine the plant model. Many a times, the model is unknown and the open loop process is not desirable. Hence we tend to look at the closed loop system and methods for tuning the PID parameters within a closed loop are more advantageous.

Methods such as Relay feedback tuning replace the feedback controller with relays. This makes systems oscillate. On determining one point on the Nyquist plot which is stable,the PID parameters are chosen.

Other methods such as Unfalsified Control requires a finite set of candidate PID controllers that must be specified.This method uses input-output parameters to choose the model.

The Iterative Feedback tuning method optimizes the controllers with respect a cost function derived from the output signal of the closed loop system. It does not require opening the loop.

This paper presents a method to optimize step response of closed loop systems with PID controllers using the discrete version of the Extremum Seeking Algorithm.The ES algorithm, a nonmodel based method, minimises a cost function and iteratively modifies the argument of the cost function to reach a local minima.

2 Motivation

2.1 Manual Calculation

The earlier techniques such as Zeigler Nichols tuning or Internal Model Control technique require manual calculation of the PID parameters from the Nyquist minimum gain and frequency as in the case of ZN tuning or the transfer function of the open loop system in the case of IMC technique. The ES method is automated and does not require manual calculation.

2.2 Prior Knowledge of the plant

Traditional techniques require prior knowledge of the model to compute the PID parameters. There are cases when the plant model is unknown or requires experiments to conclude the plant model. The ES algorithm is better to use as it is a nonmodel- based model and can be utilized to find parameter values for unknown systems with ease.

2.3 Ease in usage

The ES algorithm can be used many a times when the open loop system is not desirable.It can be used for systems of any order and does not require manual calculation. It can be implemented with an cost function.

3 Literature Review

[1] is the main reference for this report. The figures of block diagram and simulations are taken from the paper. The approach of ES is implemented from this paper.
The IMC algorithm is also implemented in this paper.
[2] is referred in the derivation in section 4.4.
[3] is referred in making the Zeigler Nichols simulation.

4 System Models

4.1 Cost Function and PID controllers

ES is used to tune PID parameters by minimeing the cost function. The cost function that is used is an Integrated square error (ISE) cost function,

$$J(\theta) = \frac{1}{T - t_0} \int_{t_0}^{T} e^2(t, \theta) dt \tag{1}$$

This is concluded from comparing it with Integrated absolute error (IAE), Integrated time dependent weighting(ITSE) and Integrated time absolute error (ITAE).It was seen that the ISE cost function has the best response and decrease the time for output to initially reach the setpoint. The cost function can be changed with respect to the problem but the stability of the ES must be ensured.

The error $e(t,\theta) = r(t) - y(t,\theta)$ is the difference between the reference (taken as a step input) and the output of the closed loop system.Also,

$$\theta = [K, T_i, T_d]^T \tag{2}$$

It contains the PID parameters.The block diagram of the closed loop system is shown in Figure 1, G is the unknown plant, the controller parameters are defined as

$$C_r(s) = K(1 + \frac{1}{T_i s}) \tag{3}$$

$$C_y(s) = K(1 + \frac{1}{T_i s} + T_d s) \tag{4}$$

r,u and y are the reference, control and output signals respectively.

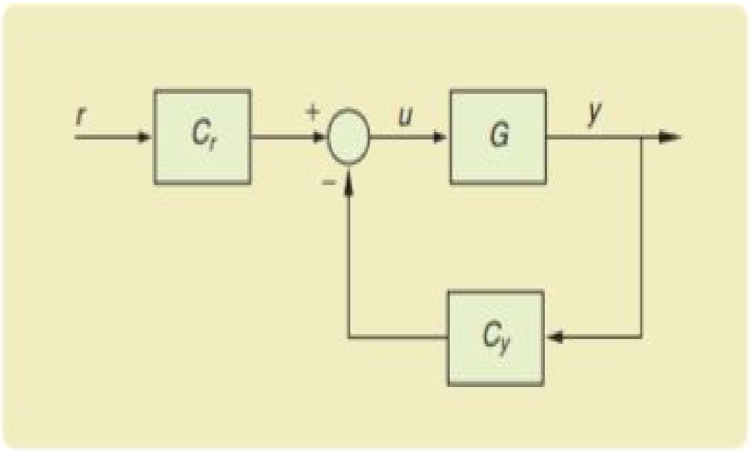


Figure 1

4.2 Working

The cost function $J(\theta)$ maps the PID parameters $[K, T_i, T_d]$ to track the performance.Its is a gradient method and finds the minimiser of $J(\theta)$.The cost function is calculated at the end of the step response experiment. In every iteration, the cost function is calculated and is used to estimate the new values of theta.This is achieved by perturbing the input parameter $\theta(k)$ of the system. A corresponding perturbation is seen in $J(\theta)$. The gradient is determined by passing cost function through a high pass filter ,removing the DC components and demodulating it by multiplying it with discrete time signal of same frequency. The overall ES PID tuning scheme is shown in Figure 2

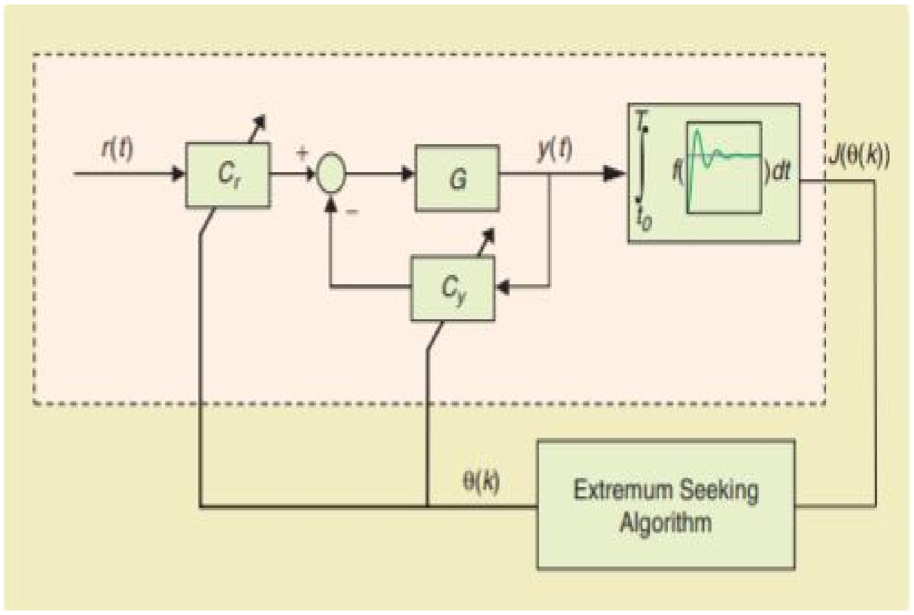


Figure 2

4.3 Algorithm

The time-domain implementation of discrete time ES algorithm is given by,

$$\zeta(k) = -h\zeta(k-1) + J(\theta(k-1)) \tag{5}$$

$$\hat{\theta}_i(k+1) = \hat{\theta}_i(k) - \gamma_i\alpha_i cos(\omega_i k)[J(\theta(k)) - (1+h)\zeta(k)] \tag{6}$$

$$\theta_i(k+1) = \hat{\theta}_i(k+1) + \alpha_i cos(\omega_i(k+1)) \tag{7}$$

where,$\zeta(k)$ is a scalar, γ_i is the adaptive gain and α_i is the perturbation amplitude.The modulation frequency ω_i is chosen such that $\omega_i = a_i\pi$ where $0 < a < 1$ and $0 < h < 1$.

Figure 3 shows the time domain implementation of the discrete ES algorithm.

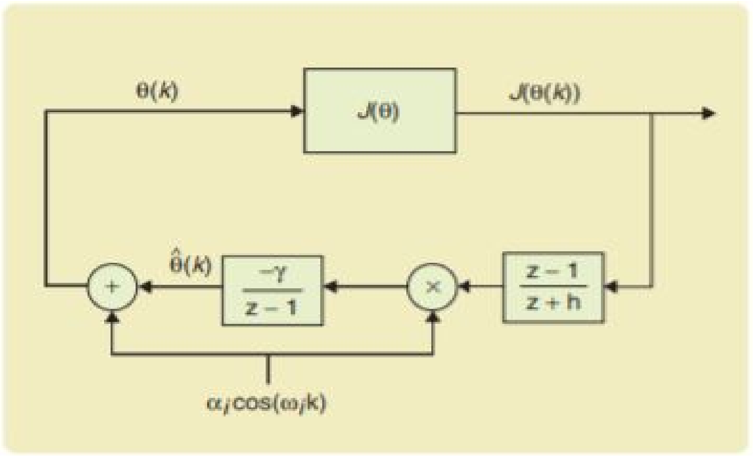


Figure 3

4.4 Derivation- How does ES work?

Take x be a system input and perturb it,

$$x(t) = x(0) + cos(\omega t)$$

This leads to the output function,

$$F(x(t)) = f(x(0)) + cos(\omega t)$$

depending on where the extremum point is w.r.t this point, the controller will change parameter to approach extremum point.

Assume a quadratic cost function,

$$J(\theta) = f^* + \frac{f''}{2}(\theta - \theta^*)^2 \tag{8}$$

where $f'' > 0$
w.k.t,any C^2 function can be expressed as (8) with it's taylor series expansion. Also, gradient of $J(\theta) = 0$ at extremum point.

We also have error,

$$\bar{\theta_1} = \theta_1^* - \hat{\theta_1} \tag{9}$$

therefore,

$$\theta_1 - \theta_1^* = asin(\omega t) - \bar{\theta_1} \tag{10}$$

substitute (10) in (8) and expand

$$J(\theta) = f^* + \frac{f^{''}}{2}(asin(\omega t) - \bar{\theta_1})^2$$

$$= f^* + \frac{f^{''}}{2}(a^2 sin^2(\omega t) + \bar{\theta_1}^2 - 2asin(\omega t)\bar{\theta_1})$$

w.k.t, $2sin^2(\omega t) = 1 - cos(2\omega t)$

$$= f^* + \frac{f^{''}}{2}(\frac{a^2}{2}(1 - cos(2\omega t)) + \bar{\theta_1}^2 - 2asin(\omega t)\bar{\theta_1})$$

$$= f^* + \frac{f^{''}}{4}a^2 + \frac{f^{''}}{2}\bar{\theta_1}^2 - af^{''}sin(\omega t)\bar{\theta_1} + \frac{f^{''}a^2}{4}cos(2\omega t)$$

Pass the function through a high pass washout filter. This removes the DC component. Multiple with $sin(\omega t)$

$$\psi = \frac{z-1}{z+h}[J(\theta)] = \frac{f^{''}}{2}\bar{\theta_1}^2 sin(\omega t) - af^{''}\bar{\theta_1}sin^2(\omega t) + \frac{f^{''}a^2}{4}sin(\omega t)cos(2\omega t)$$

w.k.t,

$$sin^2(\omega t) = \frac{1 - cos(2\omega t)}{2} and cos(2\omega t)sin(\omega t) = \frac{sin(3\omega t) - sin(\omega t)}{2}$$

$$\psi = f^{''}\bar{\theta_1}^2 sin(\omega t) - \frac{af^{''}}{2}\bar{\theta_1} + \frac{a^2 f^{''}}{2}\bar{\theta_1}cos(2\omega t) + \frac{a^2 f^{''}}{8}(sin(3\omega t) - sin(\omega t))$$

Pass ψ through low pass filter, all the frequency terms will disappear,

from (9)

$$\bar{\theta_1} = \theta_1^* - \hat{\theta_1}$$

thus,

$$\bar{\theta_1}^{\cdot} = -\hat{\theta_1}^{\cdot}$$

$$\bar{\theta_1} = \frac{-\gamma}{z-1}[\frac{-af^{''}}{2}\bar{\theta_1}]$$

Therefore,

$$\hat{\theta}(k+1) = (1 - \frac{\gamma a f^{''}}{2})\hat{\theta}(k)$$

If $\frac{\gamma a f^{''}}{2}$ is sufficiently small and positive, then $\hat{\theta}(k)$ will die down.

We can also note that, $f^{''}\hat{\theta}(k)$ is the gradient derivative of

$$J(\theta) = f^* + \frac{f^{''}}{2}(asin(\omega t) - \bar{\theta_1})^2$$

if $a = 0$

$$J^{\cdot}(\theta) = \frac{f^{''}}{2} 2\bar{\theta} = f^{''}\bar{\theta}$$

Thus, adding the perturbations and integrals is to estimate the gradient of $J(\theta)$ Hence, we can conclude that we use a gradient based optimization.

5 Results and Discussions

The ES algorithm, ZN tuning method, IMC tuning method and IFT tuning method has been implemented on four open loop systems:

$$G_1 = \frac{1}{1+20s} exp^{-5s} \tag{11}$$

$$G_2 = \frac{1}{1+20s} exp^{-20s} \tag{12}$$

$$G_3 = \frac{1}{(1+10s)^8} \tag{13}$$

$$G_4 = \frac{1}{(1+10s)(1+20s)} \tag{14}$$

G_1 has a time delay of 5 seconds,G_2 has a time delay of 20 seconds,G_3 has eight repeated roots,G_4 has non minimal phase.

Parameters and Assumptions:

- The close loop system is simulated with time step of 0.01s.
- The time delay is of the third order Pade approximation.
- PID controller values found using ZN method is starting point of ES.
- For all simulations a=0.8 and h=0.5.

5.1 Simulations

5.1.1 Tuning for G_1

Parameters set:

Time delay= 5 seconds

$\alpha = [0.1, 1, 0.1]_T$

$\gamma = [200, 200, 200]_T$

$\omega_i = a^i \pi$

Observations:

The ES algorithm estimates the cost function to produce the local minima.

ES increases the T_i three times to that of ZN approximation, thus reducing the intergral portion of the controller.

Performance of ES is better than that of ZN and IMC.

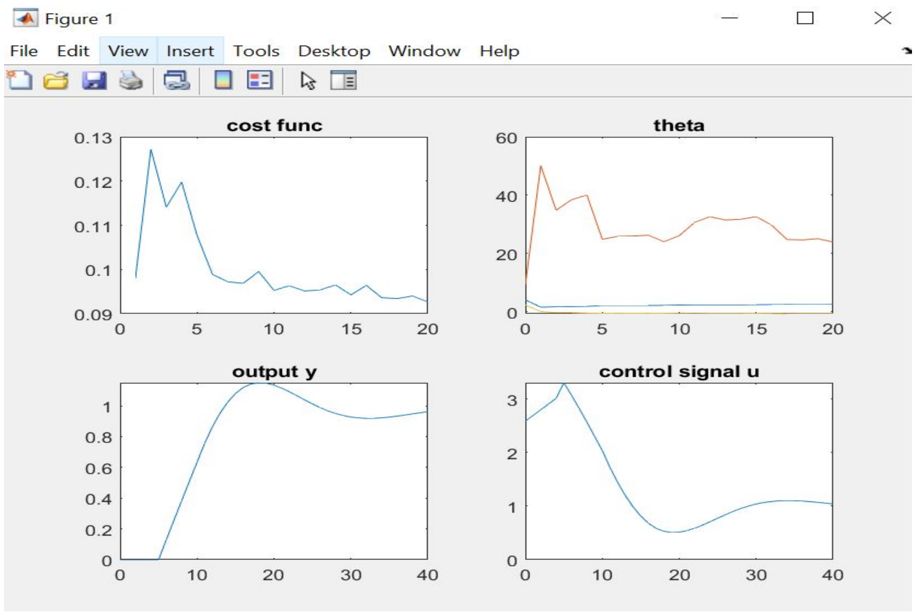


Figure 4: Simulation of ES for G1

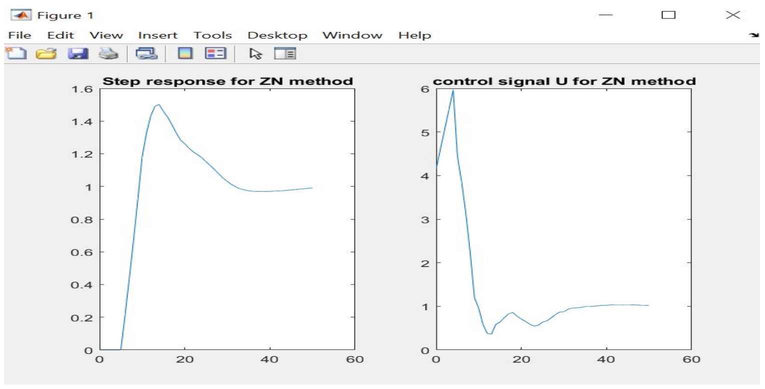


Figure 5: Simulation of ZN for G1

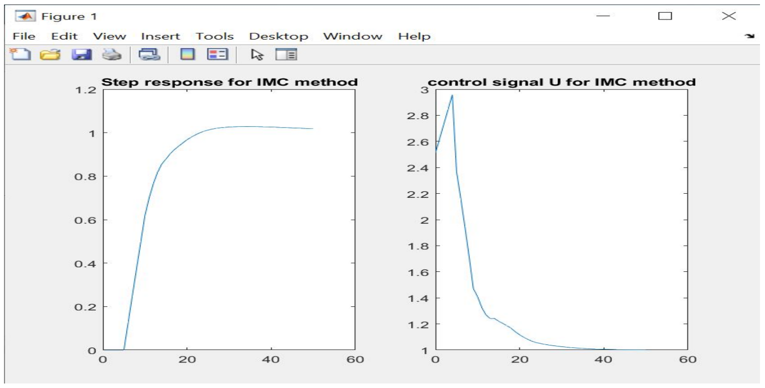


Figure 6: Simulation of IMC for G1

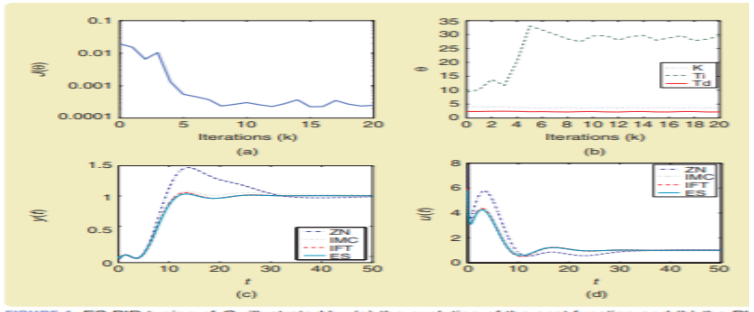


FIGURE 4 ES PID tuning of G_1 illustrated by (a) the evolution of the cost function and (b) the PI
parameters during ES tuning of the closed-loop system with $G_1(s)$. The lower plots present (
the output signal and (d) the control signal during step-response experiments of the closed-loo
systems with $G_1(s)$ and the PID controllers obtained from the four methods. ES reduces the co
function in (a) by increasing the integral time in (b), which produces a more favorable ste
response similar to that given by IFT in (c).

Figure 7: Simulation done in the paper

5.1.2 Tuning for G_2

Parameters set:

Time delay= 20 seconds

$\alpha = [0.06, 0.3, 0.2]_T$

$\gamma = [2500, 2500, 2500]_T$

$\omega_i = a^i\pi$

Observations:

The ES algorithm reduces the cost function in less than ten iterations.

ES has a closed loop system close to IMC and IFT with improved overshoot and settling time.

5.1.3 Tuning for G_3

Parameters set:

System has a repeating pole at -0.1 (repeated eight times)

$\alpha = [0.06, 1.1, 0.5]_T$

$\gamma = [800, 3500, 300]_T$

$\omega_1 = \omega_2\pi$

$\omega_3 = a^3\pi$

Observations:

The ES algorithm improves the step response behaviour set by ZN.

ES reduces the T_i and K values to reduce the value of the cost function.

ES response similar to IFT, yet with smaller settling time than IMC.

Requires 30 iterations for parameter convergence.

5.1.4 Tuning for G_4

Parameters set:

System has nonminimal phase

$\alpha = [0.05, 0.6, 0.2]_T$

$\gamma = [2000, 10000, 2000]_T$

$\omega_1 = \omega_2\pi$

$\omega_3 = a^3\pi$

Observations:

The ES algorithm produces step response similar to IFT.Both ES and IFT have no overshoot and smaller settling time as compared to Zn and IMC

ES produces larger initial control signal

ES has increased T_i for better system performance.

5.2 Control saturation

There is a possibility of actuator saturation that leads to integrator saturation, where the feedback becomes temporarily disconnected as controller output is no longer attached to the feedback.During this time T_i keeps growing regardless of error, that can be difficult to unwind.

To examine ES in the presence of saturation we execute it with and without the tracking antiwinding scheme.

Observations:
On comparing ES and IMC , we see that in IMC there is an overshoot problem.
In ES the integral time T_i increases to increase performance.
ES finds controller parameters that work almost as fine as with windup. But ES with large T_i shows poor performance.
Therefore, it is always preferable to use the antiwindup rather than without.

5.3 Selecting parameters for ES tuning

The main parameters are:
α_i (perturbation amplitude)
γ_i (adaptation gains)
ω_i (pertubation frequency)
h (high pass filter)
Minimiser found is fairly insensitive to ES parameters. Test is done using G_2: varying α and γ
Observations:
ES yields the same PID parameters even if α reduces by 50 percent and γ is similar to G_1 but convergence is slower as the amplitude and adaptation gains are reduced.
A tradeoff is made between speed of convergence and domain of initial conditions that yield the minimizer θ^*.

5.4 Comparison of tuning parameters

ES and IFT use the same cost function and thus yeild similar outputs. Both methods are non-model based and estimate the gradient of cost function w.r.t controller parameters.Estimated gradient is then used to find local minimiser.

The difference is in how the two methods find the gradient. IFT uses the signal information from three experiments including specific feedback experiment and assumes system to be Linear time Invarient.
ES uses simple filters along with modulation of sin function to estimate gradient but, ES requires several design parameters while IFT requires only step size.

Even though ES and IFT are harder to implement , they show better results than ZN and IMC.
The benefit is also seen in nonminimal phase.
ES algorithm is greater than IMC in control saturation.

6 Conclusion

Extremum seeking can be concluded to be better than the three other methods.It has advantages over model based in actuator saturation, but it requires initial values. The cost function can be chosen to reflect the desired performance attributes.

6.1 Future scope

The Extremum seeking approach is a local optimizer, he it only finds the local minima of the cost function. It can be extended to find the global minima of the cost function. Currently the parameters for the ES algorithm mainly -α, γ ,a and h have to be selected along with the initial values of θ. This can be automated to improve the method.

7 References

References

[1] N. J. Killingsworth and M. Krstic, "PID tuning using extremum seeking: online, model-free performance optimization," in IEEE Control Systems Magazine, vol. 26, no. 1, pp. 70-79, Feb. 2006.

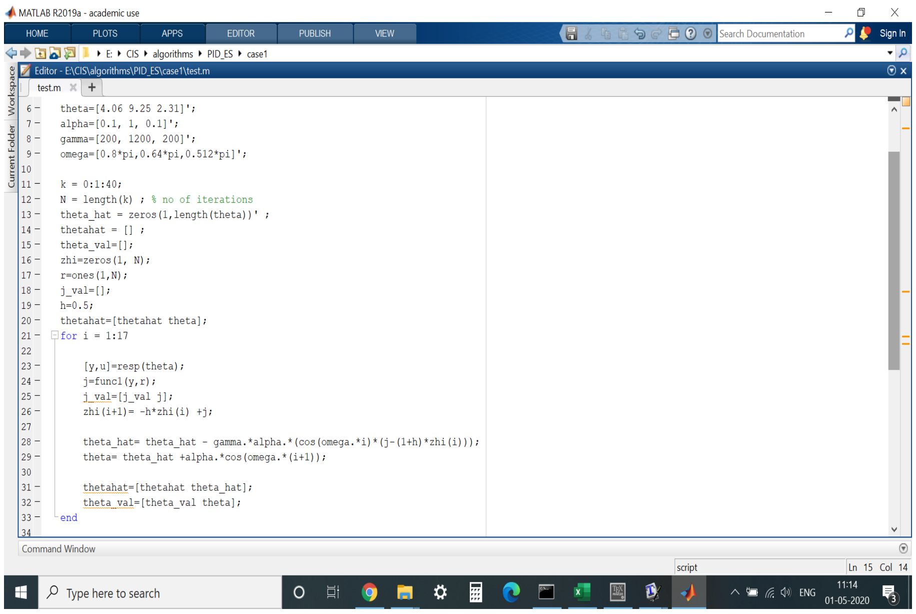
[2] Rahnama, Arash and Xia, Meng and Wang, Shige and Antsaklis, Panos. (2016). An Extremum-Seeking Co-Simulation Based Framework for Passivation Theory and its Application in Adaptive Cruise Control Systems.

[3] Waleed El-Badry,(2014) PID Tuning using Ziegler Nicholas - MATLAB Approach. https://www.slideshare.net/wbadry/pid-tuning-42460795

8 Appendix

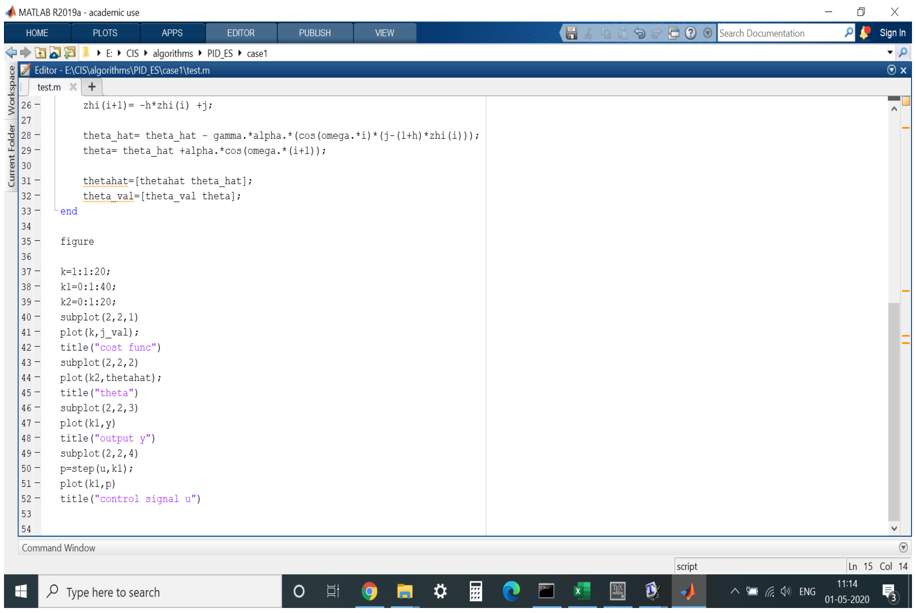
ES Method

Screenshots for the code of ES is given below. Figure 8 and 9 shows codes for the main ES algorithm.
Figure 10 shows function for getting the step response y of the closed loop output and control signal u.
Figure 11 shows function to generate the cost function $J(\theta)$



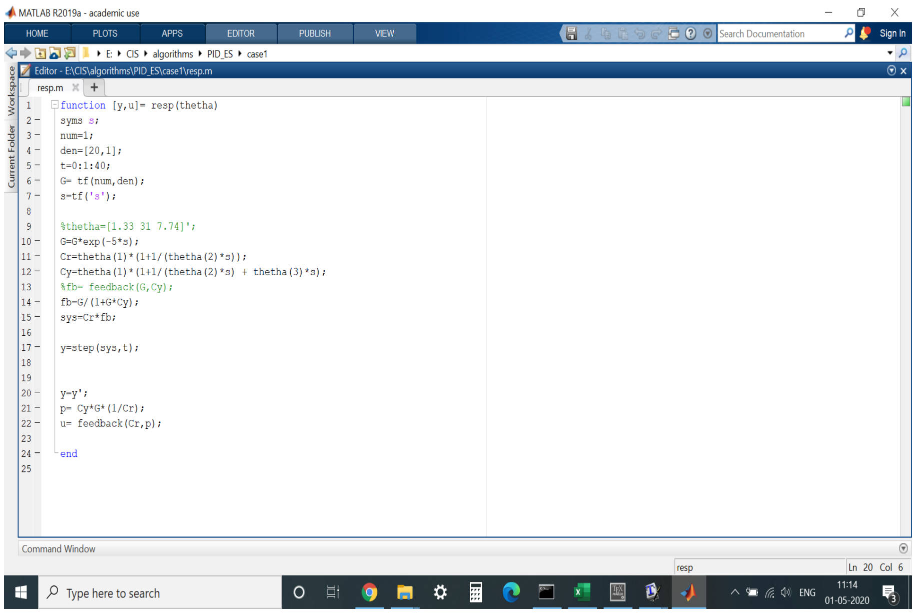
```
theta=[4.06 9.25 2.31]';
alpha=[0.1, 1, 0.1]';
gamma=[200, 1200, 200]';
omega=[0.8*pi,0.64*pi,0.512*pi]';

k = 0:1:40;
N = length(k) ; % no of iterations
theta_hat = zeros(1,length(theta))' ;
thetahat = [] ;
theta_val=[];
zhi=zeros(1, N);
r=ones(1,N);
j_val=[];
h=0.5;
thetahat=[thetahat theta];
for i = 1:17

    [y,u]=resp(theta);
    j=func1(y,r);
    j_val=[j_val j];
    zhi(i+1)= -h*zhi(i) +j;

    theta_hat= theta_hat - gamma.*alpha.*(cos(omega.*i)*(j-(1+h)*zhi(i)));
    theta= theta_hat +alpha.*cos(omega.*(i+1));

    thetahat=[thetahat theta_hat];
    theta_val=[theta_val theta];
end

```

Figure 8



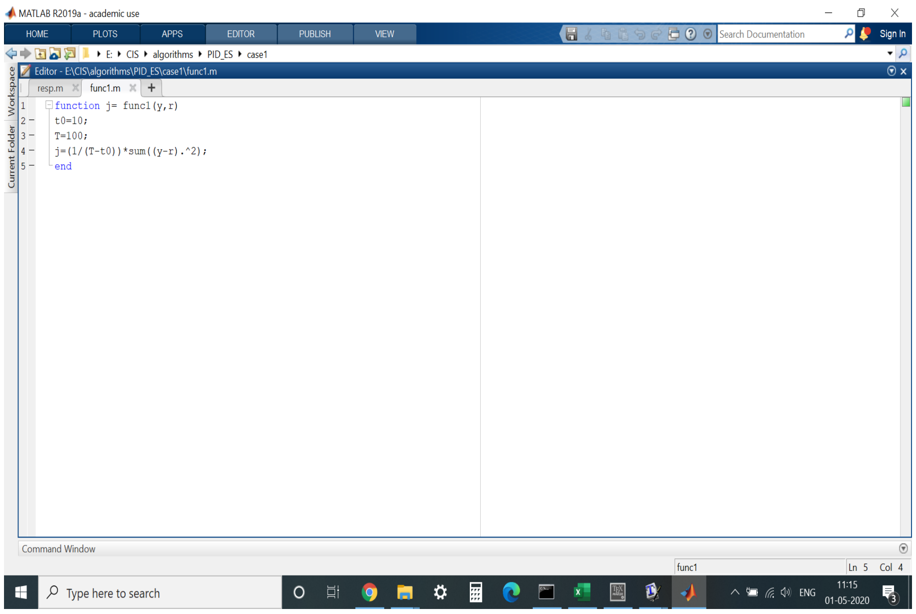
```
    zhi(i+1)= -h*zhi(i) +j;

    theta_hat= theta_hat - gamma.*alpha.*(cos(omega.*i)*(j-(1+h)*zhi(i)));
    theta= theta_hat +alpha.*cos(omega.*(i+1));

    thetahat=[thetahat theta_hat];
    theta_val=[theta_val theta];
end

figure

k=1:1:20;
k1=0:1:40;
k2=0:1:20;
subplot(2,2,1)
plot(k,j_val);
title("cost func")
subplot(2,2,2)
plot(k2,thetahat);
title("theta")
subplot(2,2,3)
plot(k1,y)
title("output y")
subplot(2,2,4)
p=step(u,k1);
plot(k1,p)
title("control signal u")

```

Figure 9



```
function [y,u]= resp(thetha)
syms s;
num=1;
den=[20,1];
t=0:1:40;
G= tf(num,den);
s=tf('s');

%thetha=[1.33 31 7.74]';
G=G*exp(-5*s);
Cr=thetha(1)*(1+1/(thetha(2)*s));
Cy=thetha(1)*(1+1/(thetha(2)*s) + thetha(3)*s);
%fb= feedback(G,Cy);
fb=G/(1+G*Cy);
sys=Cr*fb;

y=step(sys,t);


y=y';
p= Cy*G*(1/Cr);
u= feedback(Cr,p);

end
```

Figure 10

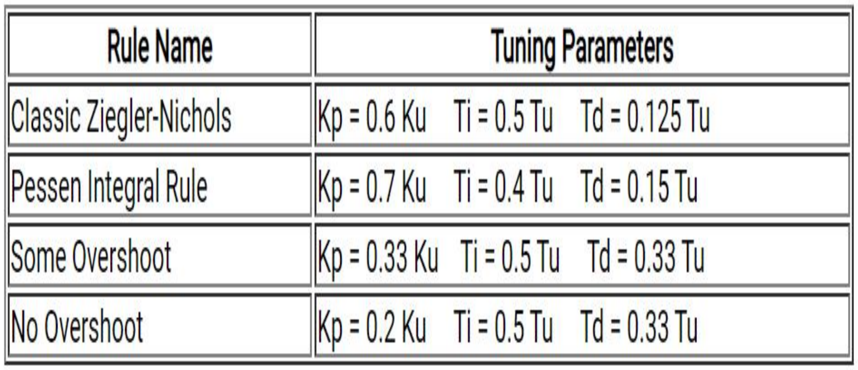


```
function j= func1(y,r)
t0=10;
T=100;
j=(1/(T-t0))*sum((y-r).^2);
end
```

Figure 11

Zeigler Nichols Tuning Method

The following picture is taken from http://www.mstarlabs.com/control/znrule.html Figure 12 shows the ZN tuning parameters

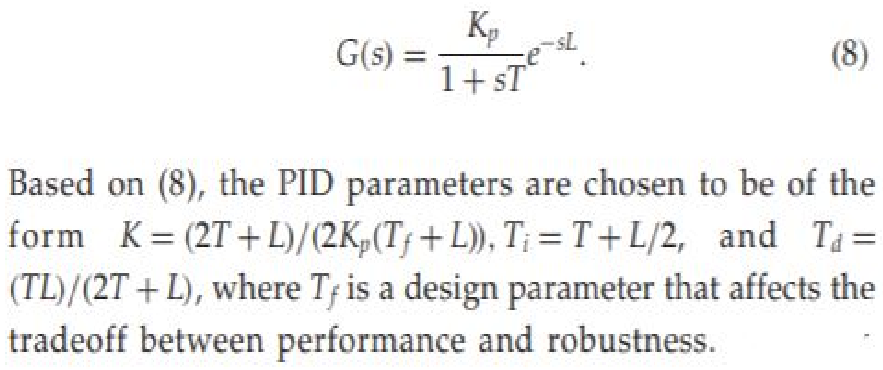


Rule Name	Tuning Parameters
Classic Ziegler-Nichols	Kp = 0.6 Ku Ti = 0.5 Tu Td = 0.125 Tu
Pessen Integral Rule	Kp = 0.7 Ku Ti = 0.4 Tu Td = 0.15 Tu
Some Overshoot	Kp = 0.33 Ku Ti = 0.5 Tu Td = 0.33 Tu
No Overshoot	Kp = 0.2 Ku Ti = 0.5 Tu Td = 0.33 Tu

Figure 12

Internal Model Control Tuning Method

The following picture shows the implementation of IMC method.



$$G(s) = \frac{K_p}{1+sT}e^{-sL}. \tag{8}$$

Based on (8), the PID parameters are chosen to be of the form $K = (2T+L)/(2K_p(T_f+L))$, $T_i = T + L/2$, and $T_d = (TL)/(2T+L)$, where T_f is a design parameter that affects the tradeoff between performance and robustness.

Figure 12 @miscredmon2015look, title=You Only Look Once: Unified, Real-Time Object Detection, author=Joseph Redmon and Santosh Divvala and Ross Girshick and Ali Farhadi, year=2015, eprint=1506.02640, archivePrefix=arXiv, primaryClass=cs.CV