4th Year Computer Engineering

2020

CSE471 (UG2003) - Computer Controlled Systems

Assignment [3] Delivery

Group [24]

Presented to : Dr. Mostafa Gomaa

Presented by:

عمرو عصام الدين صلاح فوده
هشام سمير عبد العزيز حجاج السيد
هيثم مصطفى محمود عبد الحليم الرفاعى
يمني مجدي سليمان عيد
يوسف محمد ابو المعاطى العزازى
يوسف محمود زكى عبدالمنعم الارناؤوطى
محمد شريف صبرى رجب
محمود مجدي محمد علي الصباغ
مصطفى محمد عبد المجيد ابراهيم طوبار
معتصم عادل عبد التواب عبد الخالق

Description

This Project aims to find optimal values of Kp and Ki of Pi controller That minimizes integrated squared error.

The code starts with defining the transfer function of our system then takes from the user the required range of Kp and ki.

The next step is to loop over all combinations of Kp and Ki and apply them to our system in a closed loop control system, Then we enter a unit step input to the system. then we use trapezoidal method to calculate integrated squared error of each pair of Ki and Kp, we used trapz function in this step as follows:

```
% Error calculation function

function E = Error(Y,ts)
      E = ts * trapz((Y-1).^2);
End
```

This function Takes all points of the output of the system in time domain and sampling time and then return integrated squared error.

We use our defined Error function to calculate integrated squared error for all combinations of current, next and previous values of Kp and Ki. Then we add all points of Kp and Ki that belong to our boundary to errors_array. Then we get the min of this array. If this min is the error of the current Ki and kp then current Ki and Kp are local minimum point.

After founding This local minimum point we print the result then break from this loop.

We used a Matlab script to implement our idea. The code is available in the next section of this document or at this link:

https://github.com/hesham-samir/Optimizing_Pi_variables

Code

```
% define transfer functions
s = tf('s');
C =@(Kp,Ki) Kp + Ki/s;
G = tf([1 3],[1 0.6 1.05]);

% Take input
KpMin=input('Write Minimum value of Kp = ');
KpMax=input('Write Maximum value of Kp = ');

KiMin=input('Write Minimum value of Ki = ');
KiMax=input('Write Maximum value of Ki = ');
k_step = 0.01;

% loop through given combinations of Kp and Ki
for Kp = KpMin:k_step:KpMax
    for Ki = KiMin:k_step:KiMax
    % Create Closed Loop TF
    H0 = feedback(C(Kp,Ki)*G,1);
    H1 = feedback(C(Kp,Ki+k_step)*G,1);
    H2 = feedback(C(Kp,Ki-k_step)*G,1);
    H3 = feedback(C(Kp+k_step,Ki)*G,1);
    H4 = feedback(C(Kp+k_step,Ki+k_step)*G,1);
    H5 = feedback(C(Kp+k_step,Ki-k_step)*G,1);
    H6 = feedback(C(Kp-k_step,Ki)*G,1);
    H7 = feedback(C(Kp-k_step,Ki+k_step)*G,1);
    H8 = feedback(C(Kp-k_step,Ki-k_step)*G,1);

    % Create a unit Step input
    [Y0,T0] = step(H0,50);
    [Y1,T1] = step(H1,50);
    [Y2,T2] = step(H2,50);
    [Y3,T3] = step(H3,50);
    [Y4,T4] = step(H4,50);
    [Y5,T5] = step(H5,50);
    [Y6,T6] = step(H6,50);
    [Y7,T7] = step(H7,50);
    [Y8,T8] = step(H8,50);

    % Calculate square integral error
    e0 = Error(Y0,T0(2) - T0(1));
    e1 = Error(Y1,T1(2) - T1(1));
    e2 = Error(Y2,T2(2) - T2(1));
    e3 = Error(Y3,T3(2) - T3(1));
    e4 = Error(Y4,T4(2) - T4(1));
    e5 = Error(Y5,T5(2) - T5(1));
    e6 = Error(Y6,T6(2) - T6(1));
    e7 = Error(Y7,T7(2) - T7(1));
    e8 = Error(Y8,T8(2) - T8(1));
```

```matlab
    % Add point inside our boundary to errors array, ignore other points

    errors_array = e0;
    if Ki + k_step <= KiMax
        errors_array = [errors_array,e1];
    end

    if Ki - k_step >= KiMin
        errors_array = [errors_array,e2];
    end

    if Kp + k_step <= KpMax
        errors_array = [errors_array,e3];
    end

    if Kp + k_step <= KpMax && Ki + k_step <= KiMax
        errors_array = [errors_array,e4];
    end

    if Kp + k_step <= KpMax && Ki - k_step >= KiMin
        errors_array = [errors_array,e5];
    end

    if Kp - k_step >= KpMin
        errors_array = [errors_array,e6];
    end

    if Kp - k_step >= KpMin && Ki + k_step <= KiMax
        errors_array = [errors_array,e7];
    end

    if Kp - k_step >= KpMin && Ki - k_step >= KiMin
        errors_array = [errors_array,e8];
    end
    % if current Kp and Ki is min value then break

    min_error = min(errors_array);
    if e0 == min_error
       fprintf("Optimal Solution IS:");
       Ki
       Kp
       min_error

       break
    end
    end
end
% Error calculation function
function E = Error(Y,ts)
    E = ts * trapz((Y-1).^2);
end
```

Test Cases

Input 1:

Using step size = 0.01.

```
Command Window
Write Minimum value of Kp = 2
Write Maximum value of Kp = 2.2
Write Minimum value of Ki = 1
Write Maximum value of Ki = 1.2
Found :
Ki =

     1


Kp =

    2.2000


min_error =

    0.1892

>>
```

Input 2:

Using step size = 0.01.

```
Command Window
    0.2050

>> Assignment3
Write Minimum value of Kp = 5.5
Write Maximum value of Kp = 5.7
Write Minimum value of Ki = 1.6
Write Maximum value of Ki = 1.8
Found :
Ki =

    1.8000


Kp =

    5.7000


min_error =

    0.0819

>>
```

Input 3:

Using step size = 0.1.

```
Command Window
>> Assignment3
Write Minimum value of Kp = 2
Write Maximum value of Kp = 4
Write Minimum value of Ki = 2
Write Maximum value of Ki = 4

Found :
Ki =

     2


Kp =

     4


min_error =

    0.1133

>>
```

Input 4:

Using step size = 0.1.

```
Command Window
>> Assignment3
Write Minimum value of Kp = 7
Write Maximum value of Kp = 9
Write Minimum value of Ki = 6
Write Maximum value of Ki = 7
Optimal Solution IS :
Ki =

     6


Kp =

     9


min_error =

    0.0539

>>
```