1 Solving FB control design via pole placement

1. Build your own software for solving feedback control design via pole placement for both the control and observer coupled with input scaling.
2. Apply it to the in class DC motor example and check if you can get better gain and phase margin than the in class example.Why and why not?

1.1 Software Description

1.1.1 Function Call

The Program consists of a set of 2 files first is "FBcontrol" and the second is "DTsystem", the user is only supposed to call FBcontrol which primarily parse the input to see if the inputs are all fine, and feed in the default parameter values if they are not fed by the user. DTsystem file is the main code which include the primary computations, direct access to this is possible but not recommended for easier debugging of your inputs, in case an error is flagged.
All the outputs that are generated are for Discrete Systems, even if you enter a CT system or CT poles, These system and poles will be modified internally to corresponding DT system and poles.
An example function call looks like :

```
FBcontrol(A,B,C,D,'FeedbackType','OutputFB',...
    'ObserverPoles',ObserverPoles,...
    'SystemType','Continuous',...
    'FinalPoles',FinalPoles,...
    'Frequency',100);
```

If a system is completely Discrete in nature then Frequency need not to be Specified, this new function call will look like

```
FBcontrol(A,B,C,D,'FeedbackType','OutputFB',...
    'ObserverPoles',ObserverPoles,...
    'SystemType','Discrete',...
    'FinalPoles',FinalPoles);
```

We may have a Continuous system and Discrete Pole input, such a function call will me made as following

```
FBcontrol(A,B,C,D,'FeedbackType','OutputFB',...
    'ObserverPoles',ObserverPoles,...
    'SystemType','Continuous',...
    'PolesType','Discrete',...
    'FinalPoles',FinalPoles,...
    'Frequency',100);
```

If "PolesType" is not specified then PolesType will set equal to "SystemType" by default, if none of them are specified then both of them will be set to "Discrete"

So following three function calls are same

First Style

```
FBcontrol(A,B,C,D,'FeedbackType','OutputFB',...
        'ObserverPoles',ObserverPoles,...
        'SystemType','Discrete',...
        'PolesType','Discrete',...
        'FinalPoles',FinalPoles);
```

Second Style

```
FBcontrol(A,B,C,D,'FeedbackType','OutputFB',...
        'ObserverPoles',ObserverPoles,...
        'SystemType','Discrete',...
        'FinalPoles',FinalPoles);
```

Third Style

```
FBcontrol(A,B,C,D,'FeedbackType','OutputFB',...
        'ObserverPoles',ObserverPoles,...
        'FinalPoles',FinalPoles);
```

I will be implementing Kalman Filter as an Observer to this code in further revisions. Currently code is working fine with SISO systems. It will not be hard to extend it to MIMO systems.

1.1.2 What this function return

This function return following parameters in sequence

$$A, B, C, D, K, L, CLSystem, Gm, Pm, StepResponse$$

A: Discrete time equivalent of input A
B: Discrete time equivalent of input B
C: Discrete time equivalent of input C
D: Discrete time equivalent of input D
K: Feedback Gain Matrix
L: Observer Gain Matrix
CLSystem : Final closed loop designed system
Gm : Gain Margin of closed loop system
Pm : Phase Margin of closed loop system
StepResponse : This object contains transient response characteristics

1.2 Performance of class DC motor example

Plots correspond to class DC motor example are plotted and a pole placement choice which will give better gain and phase margin is discussed in the "main.m" which is the code calling the above described function "FBcontrol"

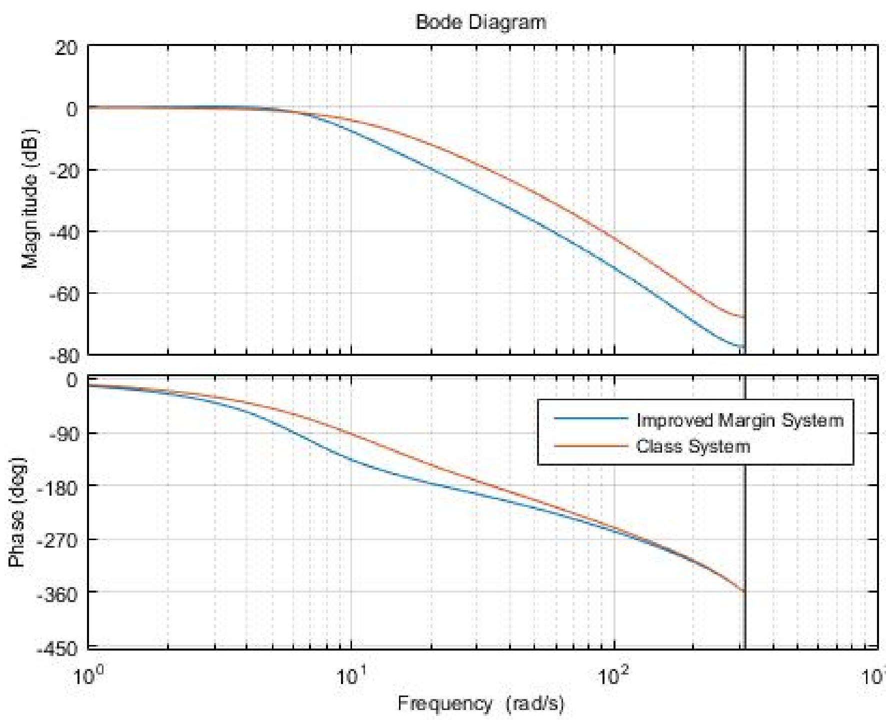
New poles are

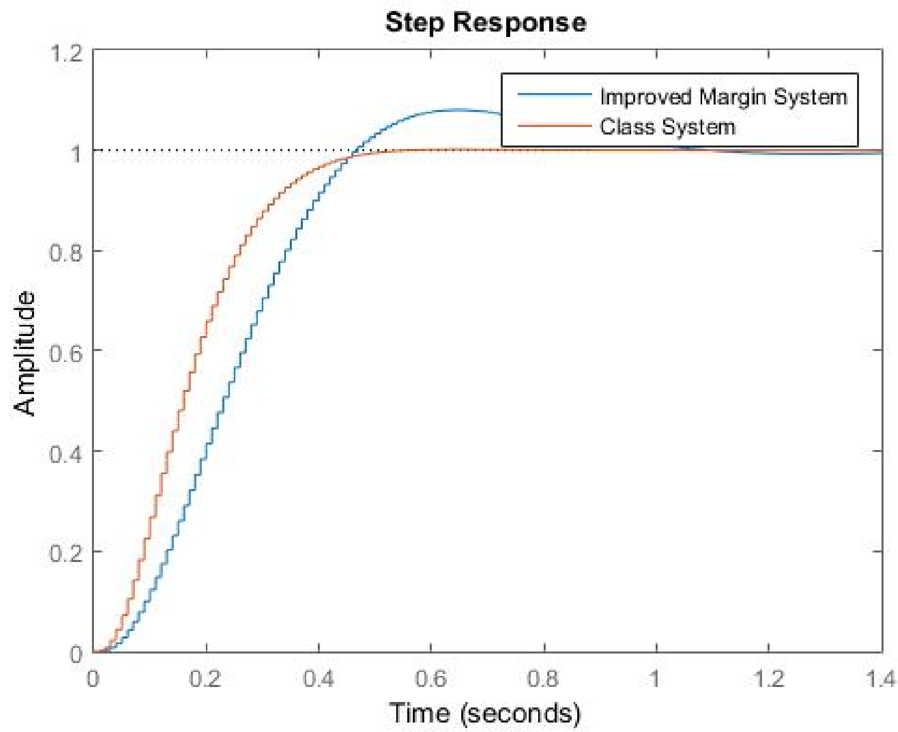
$$Better_Poles = [-3 - 5i, -3 + 5i, 80]; \tag{1}$$

Betterpoles are better in the sense that they give improved GM and PM, but they definitely lead to worse settling time or System BW.

1.2.1 Performance Improvement Analysis

Though it is possible to have a better gain and phase margin but this is happening at the cost of system bandwidth or rise time.




Step Response
Improved Margin System
Class System
Amplitude
Time (seconds)
0
0.2
0.4
0.6
0.8
1
1.2
1.4