ECE-320: Linear Control Systems
Homework 9

Due: Friday May 14 at the (very) beginning of lab

In this homework we will simulate and study integral control, full order observers coupled with state variable feedback, and then we will combine them. Having an integrator in our system is generally preferable to using a prefilter for achieving a zero steady state error for a step input, since it will help overcome errors in the plant model. Some of the things we will do in this homework are a bit weird, but they are being done for the sake of demonstrating that your programs are working.

For all of these problems, use the state models available on the class website. Both systems are torsional systems.

Be sure you are using a variable step solver, and the maximum time step is something like 1e-5 (look under Simulation-> Configuration Parameters). Also, set the initial conditions on the integrators to just 0 (a scalar).

Some useful Matlab commands are

[n,m] = size(A); zeros(n,m); eye(n)

<u>You need to save plots of every case you are told to run. Save them in a word document with appropriate figure captions and turn that in (or e-mail it). This entire homework is a pre-lab.</u>

1) In this problem we will incorporate integral control for a one degree of freedom system and run some test cases to verify your code is working.

a) Copy your program **Basic_1dof_State_Variable_Model_driver.m** to a new file **sv1_integral_driver.m,** then copy the Simulink file **Basic_1dof_State_Variable_Model.mdl** to **sv1_integral.mdl.** Modify these ***<u>new</u>*** files to implement state variable feedback using integral control. The basic integral control scheme is shown below in Figure 1. Note that the prefilter is equal to 1.

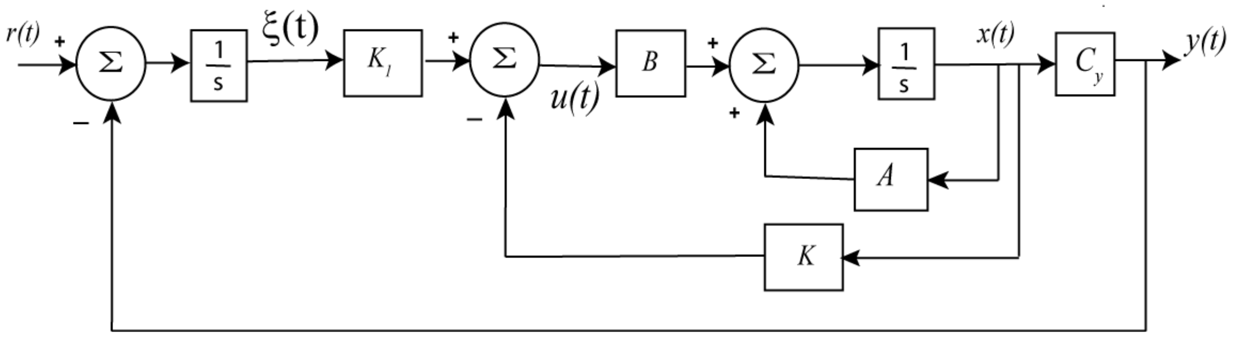

Figure 1. Integral control structure.

b) Modify **sv1_integral_driver.m** to plot the two states and the control effort (the control effort does not get scaled to degrees).

c) Set the step amplitude to 10 degrees (but convert to radians), the final time to 1.5 seconds, place the poles at -5, -10, -15, and run the simulation. You should get results like those shown in Figure 2.

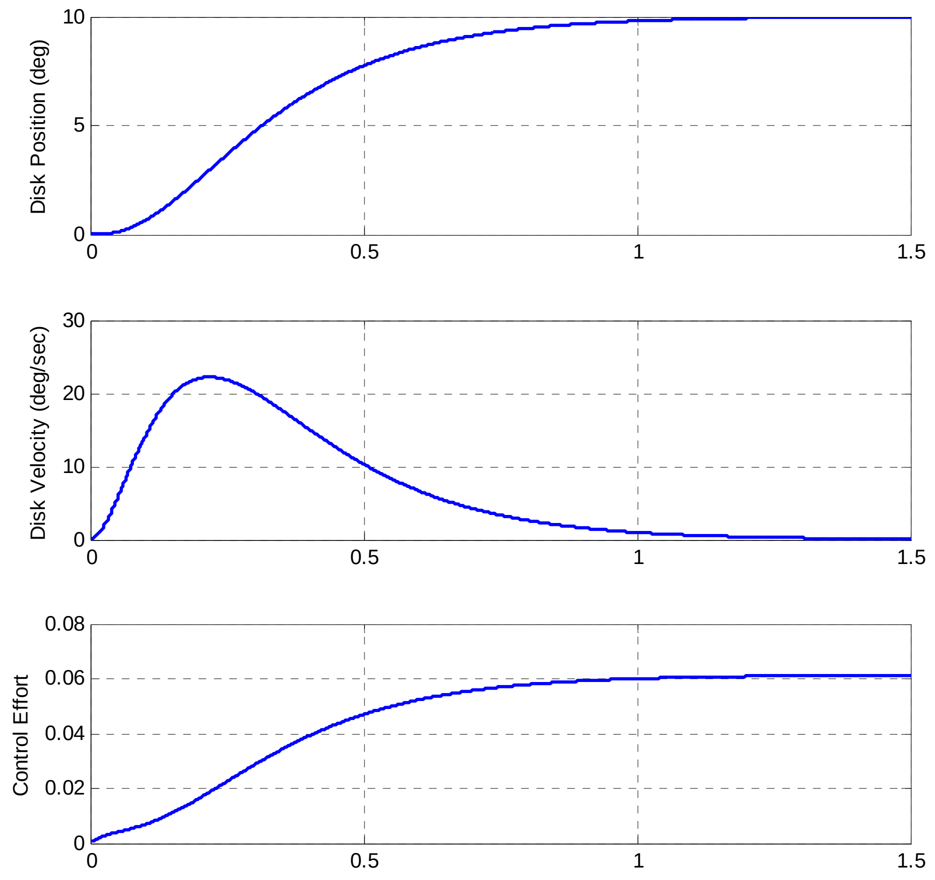

Figure 2. Initial results for the 1 dof system with integral control (problem 1c).

d) At this point there is no real reason for us to use integral control, since we know everything exactly. Let's assume your lab partner is way off on the system model. In your code, just before your run the simulation (just before the **sim** command), type A = A*0.25, which basically says you're a matrix is way off. If you make this change just before the simulation, your gains will be chosen assuming your model is correct, and we will be running the simulation using a different system. Change the final time to 2.5 seconds, and leave everything else the same as before. You should get results like those shown in Figure 3, which look pretty hopeless, but we can improve them.

e) Systems with these integrators have an interesting characteristic. Change the poles to -5, -10, and -1500 and rerun the system. You should get the results shown in Figure 4, which are much better. Note that the control effort did not change much. Before you go on, be sure to remove the line A = A*0.25, or you will have trouble in lab!

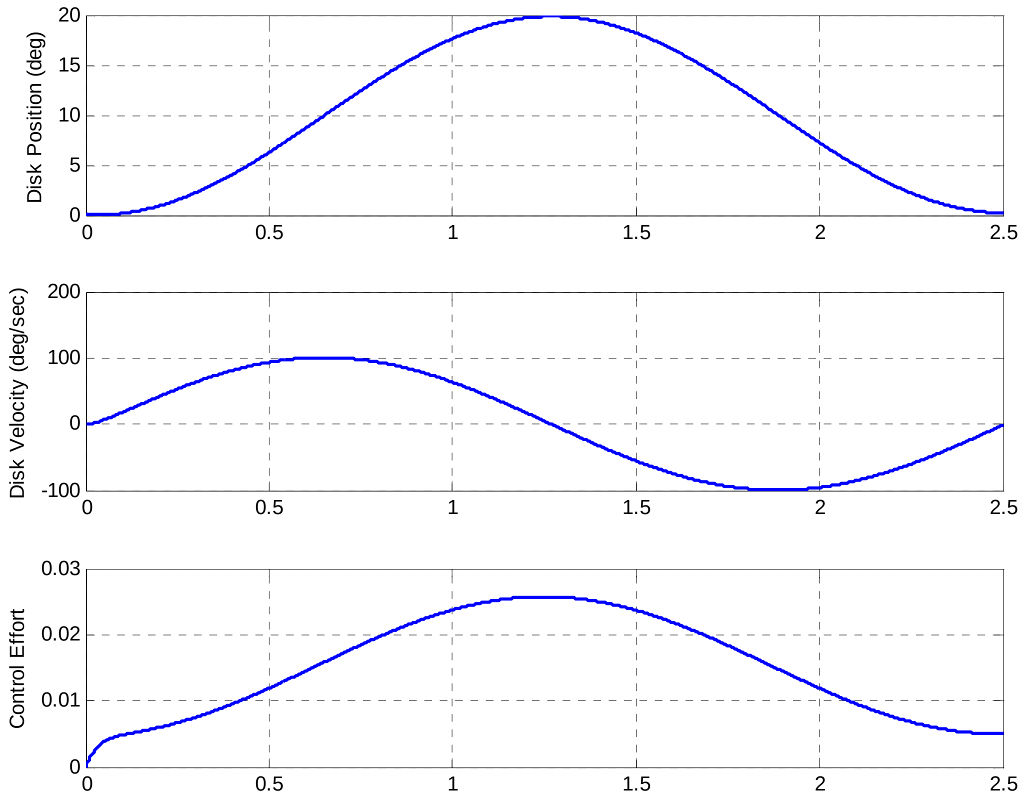

Figure 3. Integral control of a 1 dof system with bad data (problem 1d)

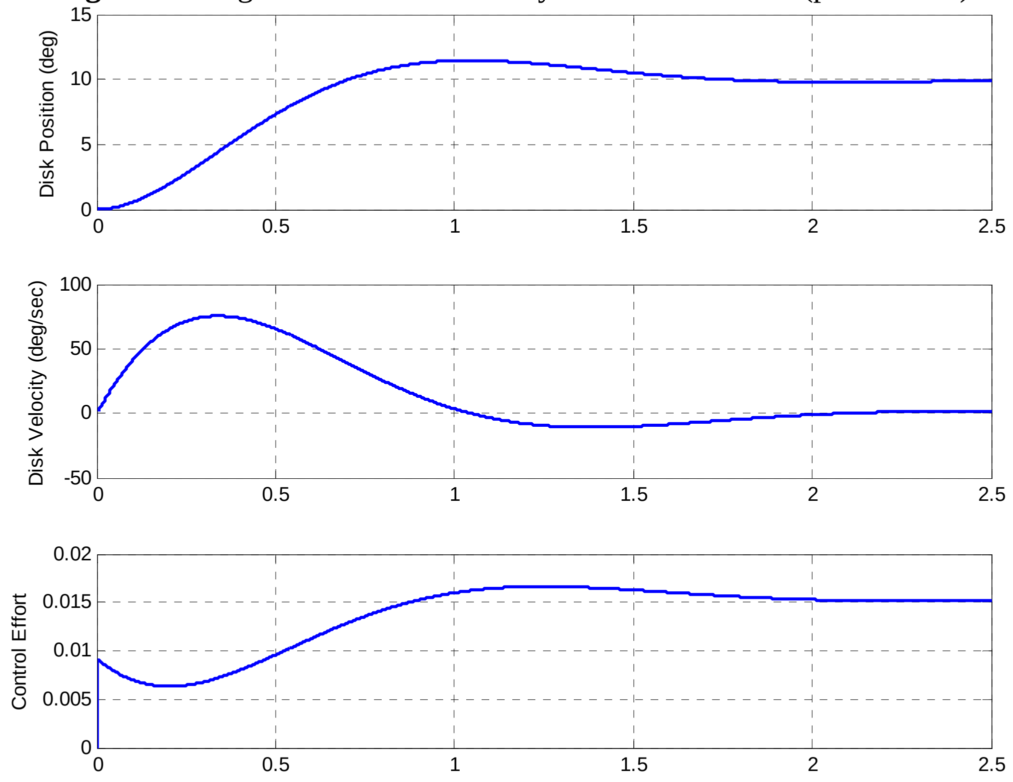

Figure 4. Integral control of a 1 dof system with bad data, but a better result (problem 1e).

2) In this problem we will incorporate integral control for a two degree of freedom system and run some test cases to verify your code is working.

a) Copy your program **Basic_2dof_State_Variable_Model_driver.m** to a new file **sv2_integral_driver.m,** then copy the Simulink file **Basic_2dof_State_Variable_Model.mdl** to **sv2_integral.mdl.** Modify these ***new*** files to implement state variable feedback using integral control.

b) Modify **sv2_integral_driver.m** to plot the four states and the control effort (the control effort does not get scaled to degrees).

c) Set the amplitude to 10 degrees, the final time to 2 seconds, and place the poles at -5, -10, -15, -20, -25. We are trying to control the position of the second disk. Run the simulation and you should get results like those shown in Figure 5.

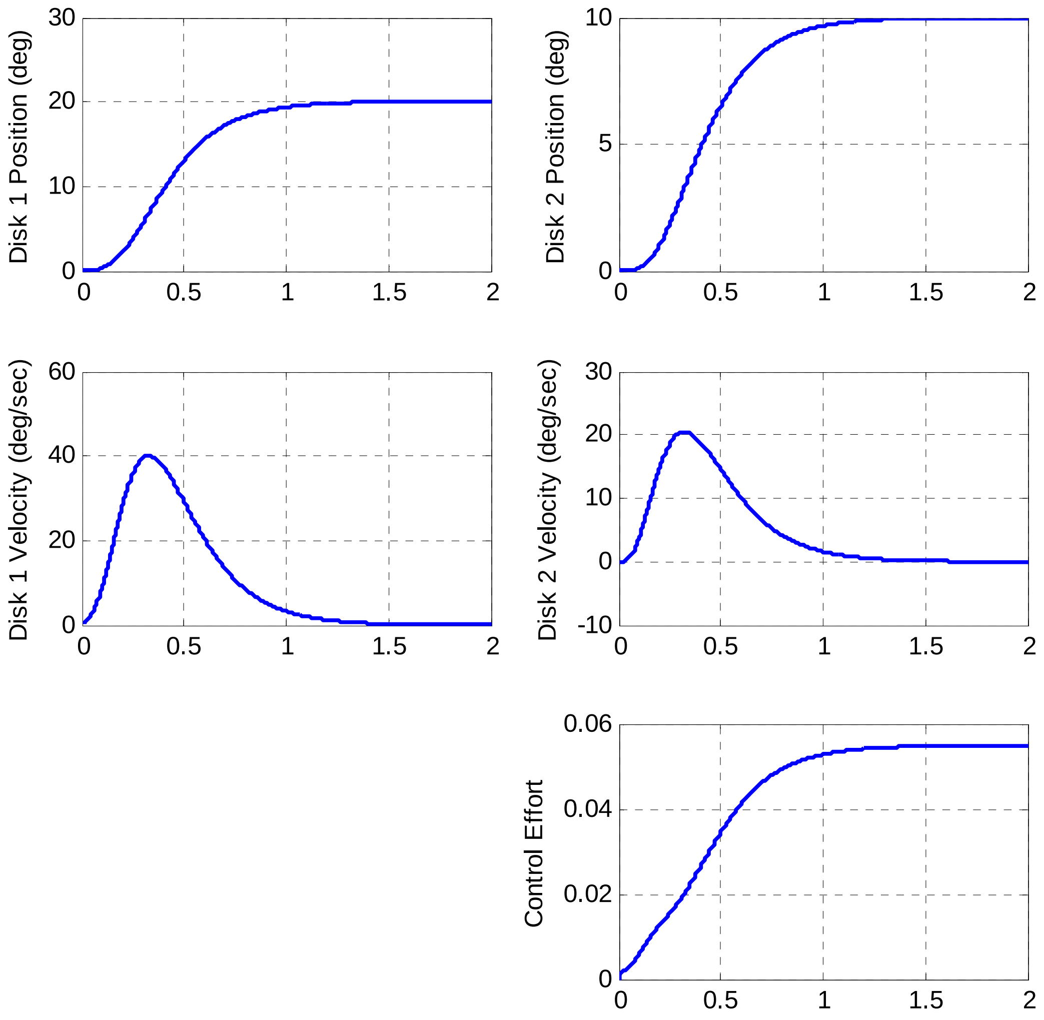

Figure 5. Integral control of a two degree of freedom system. We care controlling the position of the second disk (part 2c).

d) Assume your lab partner screwed up gain, so you model is not accurate. To see what happens when the model does not match the real system, just before the simulation type A = A*0.25, place the poles at -5, -10, -15, -20, -2500, and run the simulation for 10 seconds, you should get a plot like that in Figure 6.

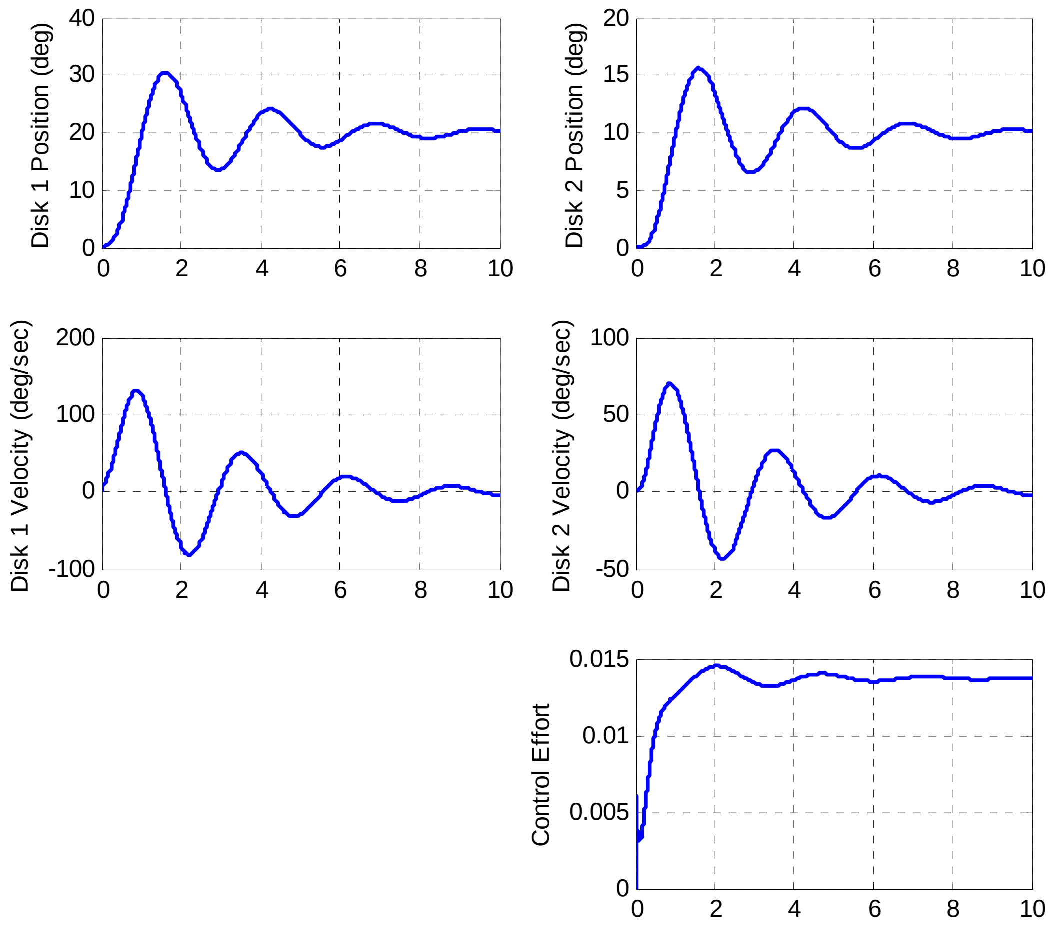

Figure 6. Integral control of a 2 dof system with bad data. We are controlling the position of the second disk (problem 2d).

e) Now assume everything is the same as in part d, but now we want to control the position of the first disk. You should get results like those in Figure 7. If your results agree, remove the line A = A*0.25 from your code.

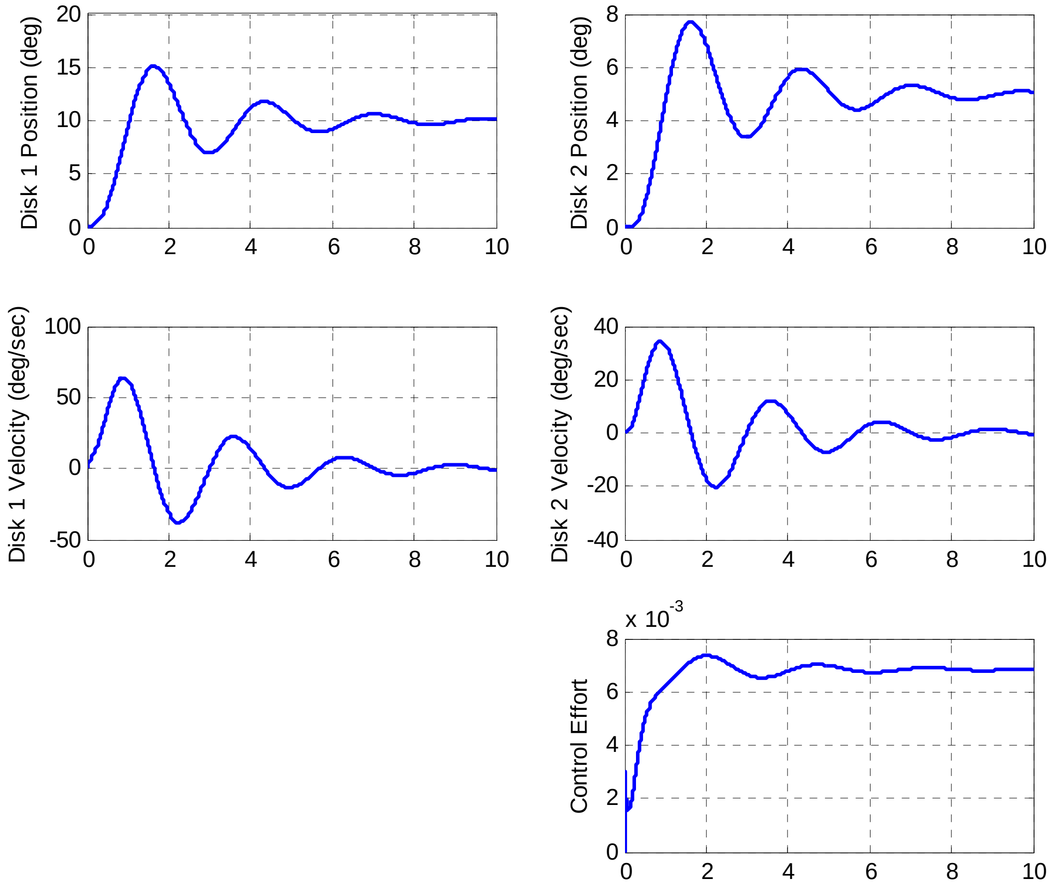

Figure 7. Integral control of a 2 dof system with bad data. We are controlling the position of the first disk (problem 2e).

3) In this problem we will incorporate a full order observer for a one degree of freedom system and run some test cases to verify your code is working.

a) Copy your program **Basic_1dof_State_Variable_Model_driver.m** to a new file **sv1_observer_driver.m,** then copy the Simulink file **Basic_1dof_State_Variable_Model.mdl** to **sv1_observer.mdl.** Modify these ***new*** files to implement state variable feedback using integral control. The basic integral control scheme is shown below in Figure 8.

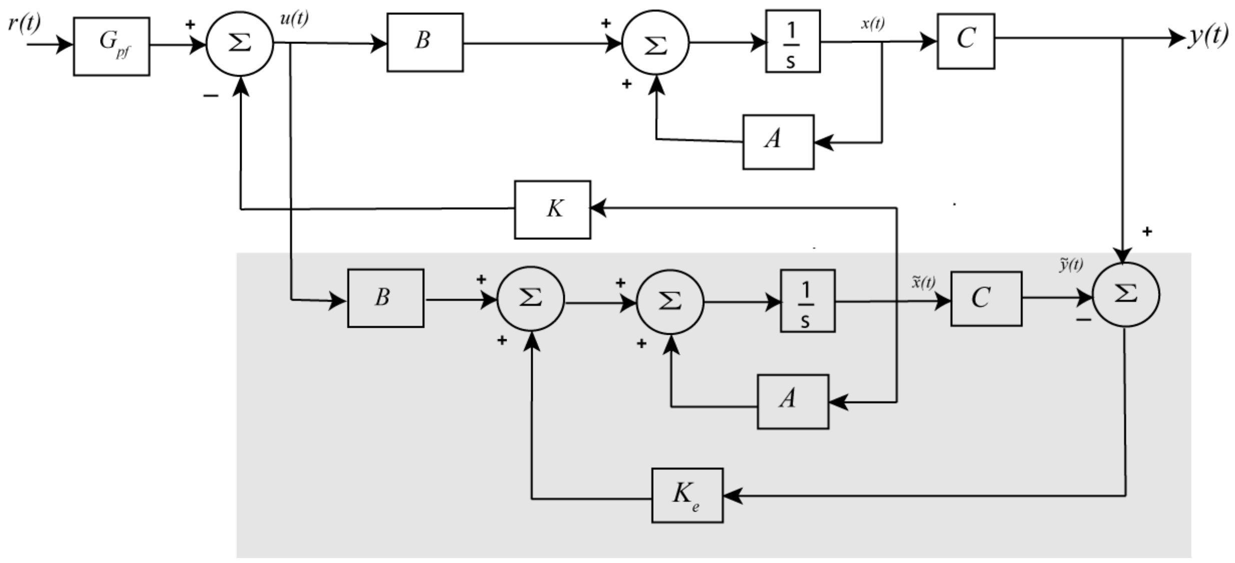

Figure 8. Full order observer state variable feedback structure.

b) Modify **sv1_observer_driver.m** to plot the each of the true and estimated states on the same graph. See Figure 9 (or Figure 10) for an example.

c) For an input of 10 degrees, a final time of 1 second, and **both** the state feedback and observer poles at p = [-10 -12], you should get a graph like that in Figure 9.

d) Now we will try and observe what happens when there is a difference between our model and the real system. Modify the observer subsystem in the model (**.mdl**) file, so the A matrix in the observer is called AA. Just before theyou run the model file, enter the command AA = A*1.5. You should get results like those in Figure 10.

e) Change the observer poles so they are five times further from the imaginary axis than the state feedback poles (observer poles at 5*p). You should get the results shown in Figure 11. If you place the poles 10 times further away you get the results presented in Figure 12. Note that as the observer poles get further away, the estimates get better. Be sure to set AA=A before you go on.

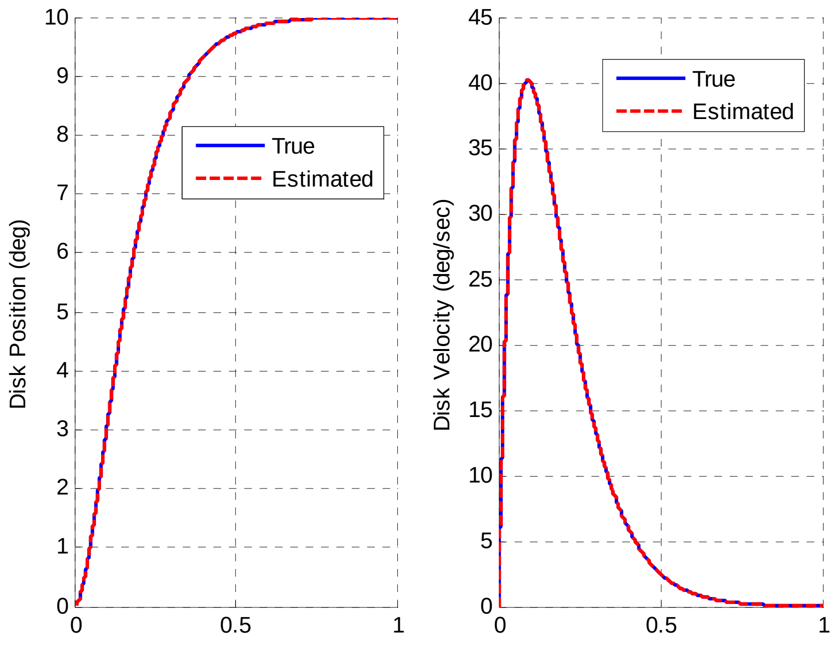

Figure 9. Results for full order observer for 1 dof system (problem 3c).

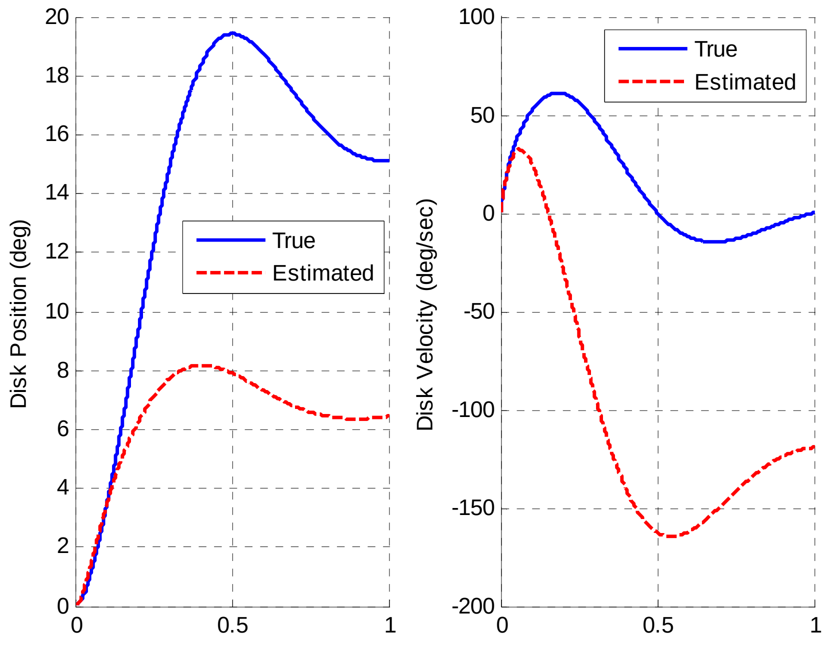

Figure 10. Results for the 1dof full order observer for a significant model error (problem 3d).

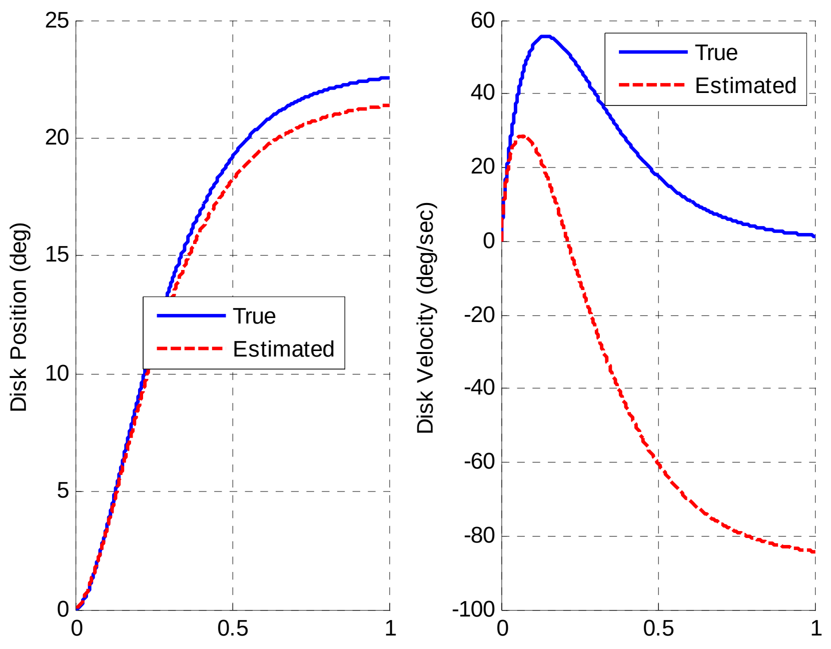

Figure 11. Results for the 1dof full order observer for a significant model error (problem 3e).

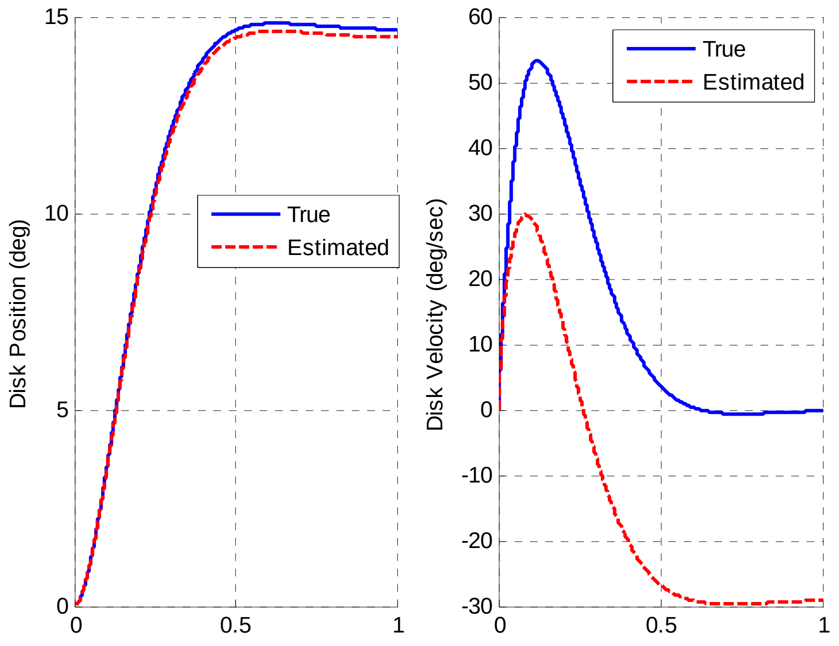

Figure 12. Results for the 1dof full order observer for a significant model error (problem 3e).

4) In this problem we will incorporate a full order observer for a two degree of freedom system and run some test cases to verify your code is working. You may need to include a vector *Cy* to correctly set the prefilter gain for the disk you are trying to control.

a) Copy your program **Basic_2dof_State_Variable_Model_driver.m** to a new file **sv2_observer_driver.m,** then copy the Simulink file **Basic_2dof_State_Variable_Model.mdl** to **sv2_observer.mdl.** Modify these ***new*** files to implement state variable feedback using integral control.

b) Modify **sv2_observer_driver.m** to plot the each of the true and estimated states on the same graph. See Figure 13 for an example.

c) For an input of 10 degrees, a final time of 1.5 second, and both the state feedback and observer poles at p = [-10, -12, -14, -16], controlling the position of the second disk with the input to the observer the position of the second disk, you should get a graph like that in Figure 13.

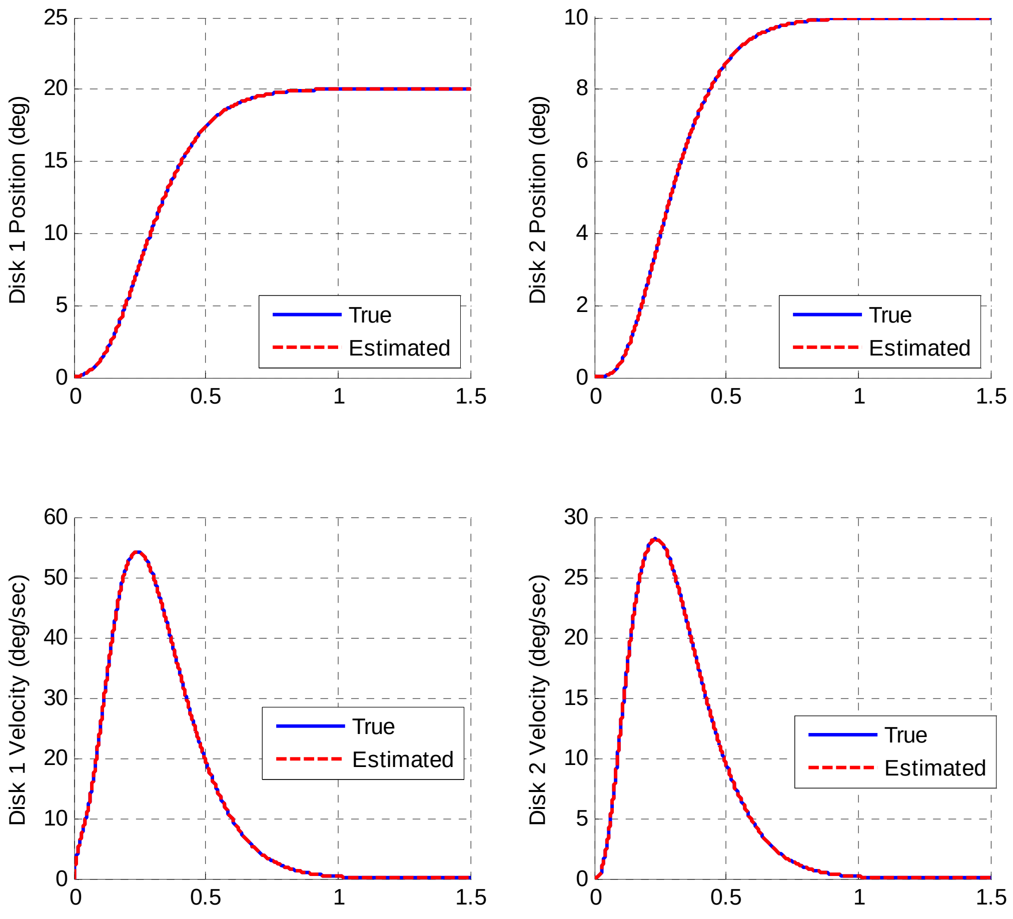

Figure 13. Full order observer for a 2 dof system (problem 4c).

d) Now we will try and observe what happens when there is a difference between our model and the real system. Modify the observer system in the model (**.mdl**) file, so the A matrix is called AA. Just before you run the model file, enter the command AA = A*1.2. Place the state feedback poles at twice their original values (at 2*p), and change the observer poles to be ten times their original values (10*p). You should get results like those in Figure 14. Note that if we do not change the pole locations, we will get an unstable system (just try it!)

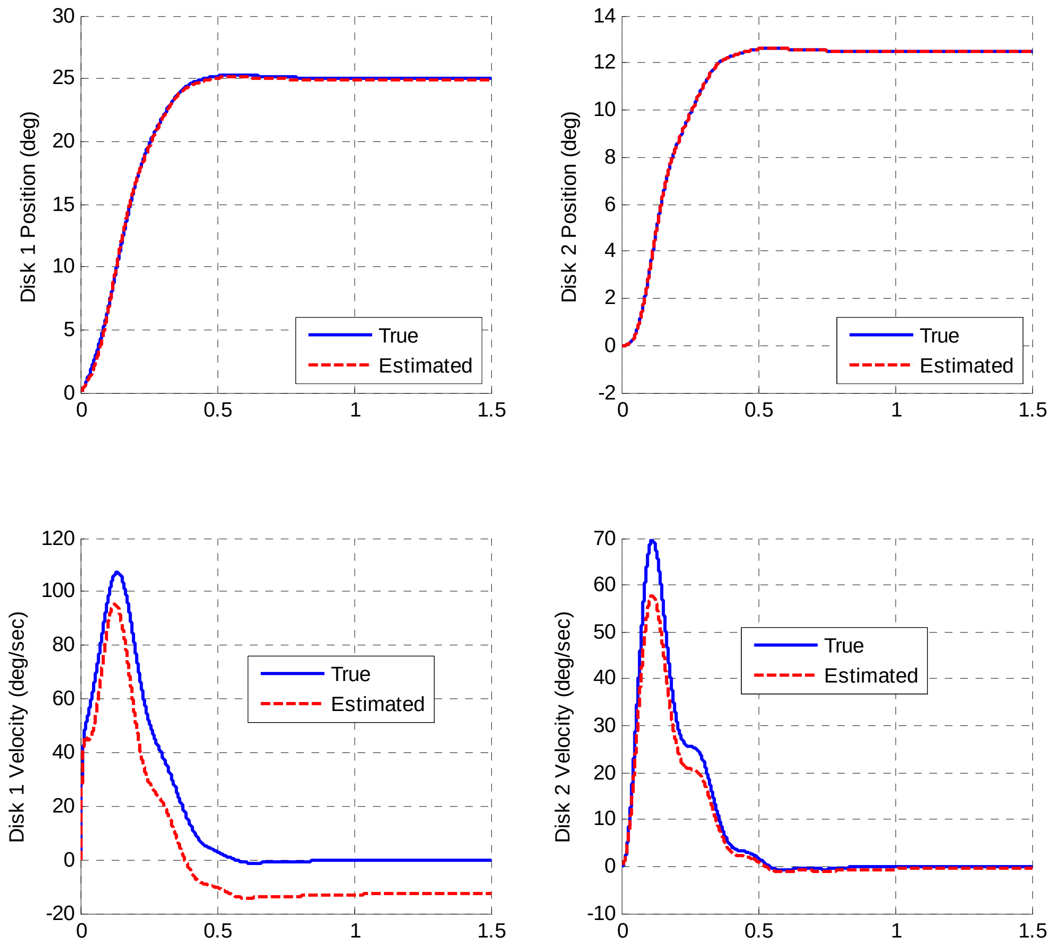

Figure 14. Full order observer for a 2 dof system with a bad model (problem 4d).

e) Now rerun the system assuming the input to the observer is the position of the first and second disk, keeping everything else the same as in part d. You should the results like those in Figure 15.

f) Now rerun the system with the following changes, AA = 0.8*A. The input to the observer is still the position of the first and second disk. You should get results like those in Figure 16.

Before you go on, set AA = A, or change your model back to its original form. Note that with the observers and errors in the system, we do not have zero steady state error.

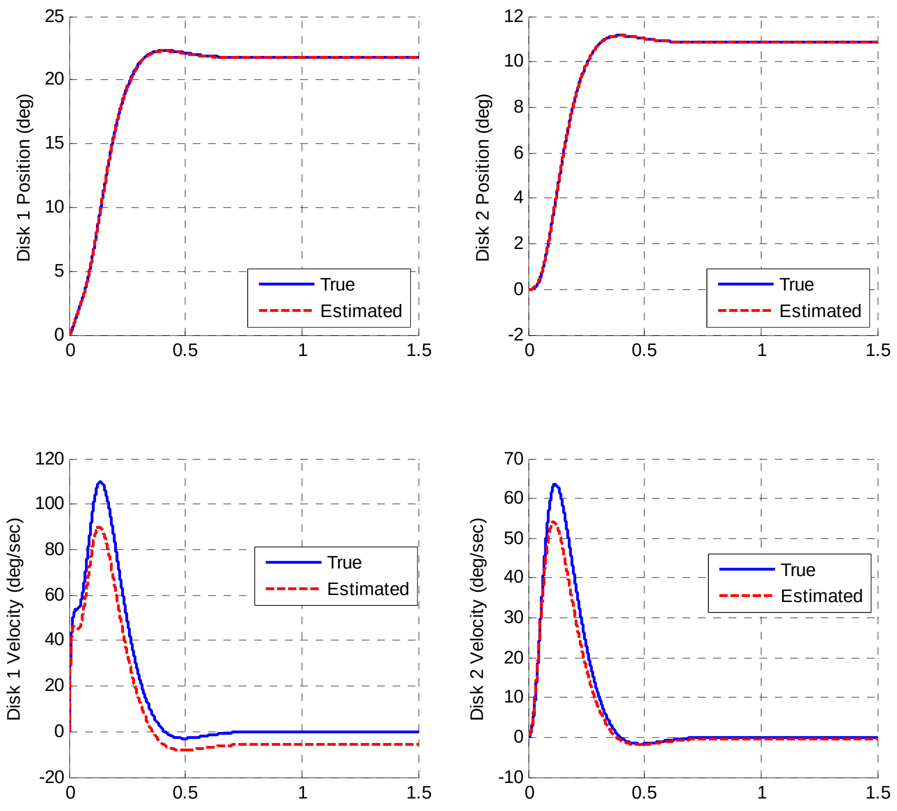

Figure 15. Full order observer for a 2 dof system with a bad model (problem 4e).

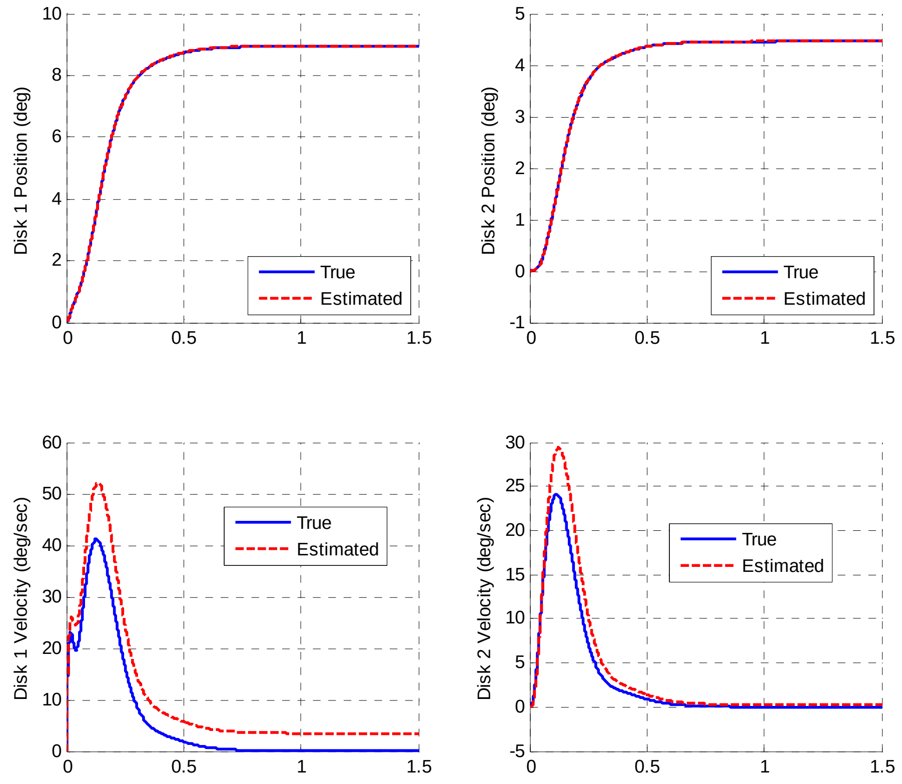

Figure 16. Full order observer for a 2 dof system with a bad model (problem 4f).

5) In this problem we will incorporate a full order observer and an integrator for a one degree of freedom system and run some test cases to verify your code is working.

a) Copy your program **sv1_observer_driver.m** to a new file **sv1_observer_int_driver.m,** then copy the Simulink file **sv1_observer.mdl** to **sv1_observer_int.mdl.** Modify these ***new*** files to implement state variable feedback with an observer using integral control. The basic control scheme is shown below in Figure 17.

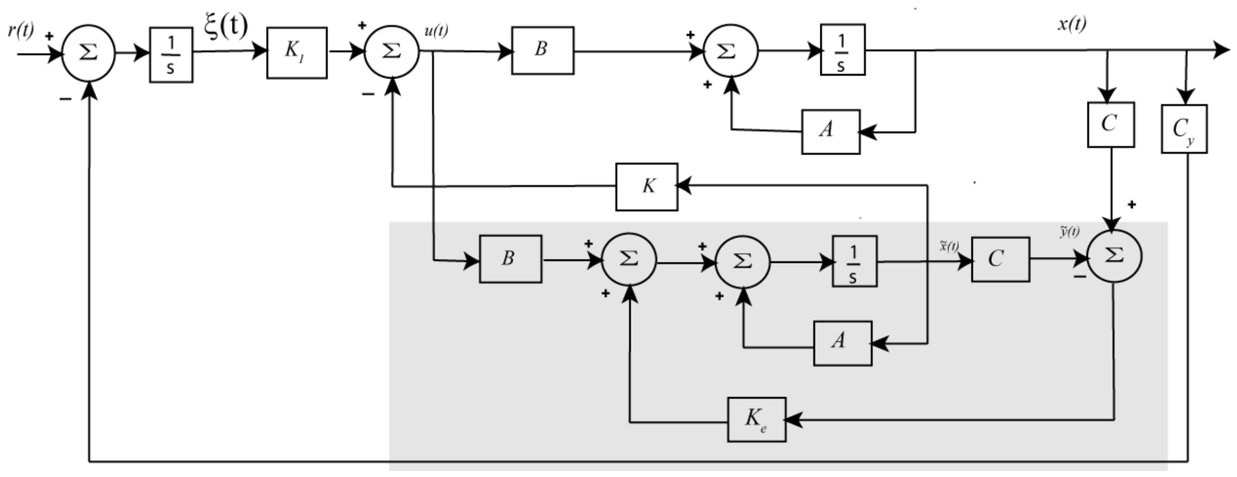

Figure 17. Full order observer state variable feedback structure with integral control.

b) For an input of 10 degrees, a final time of 1 second, place the state feedback poles at ps = [-35, -20+10j, -20-10j] and observer poles at po = [-25 -30], you should get a graph like that in Figure 18.

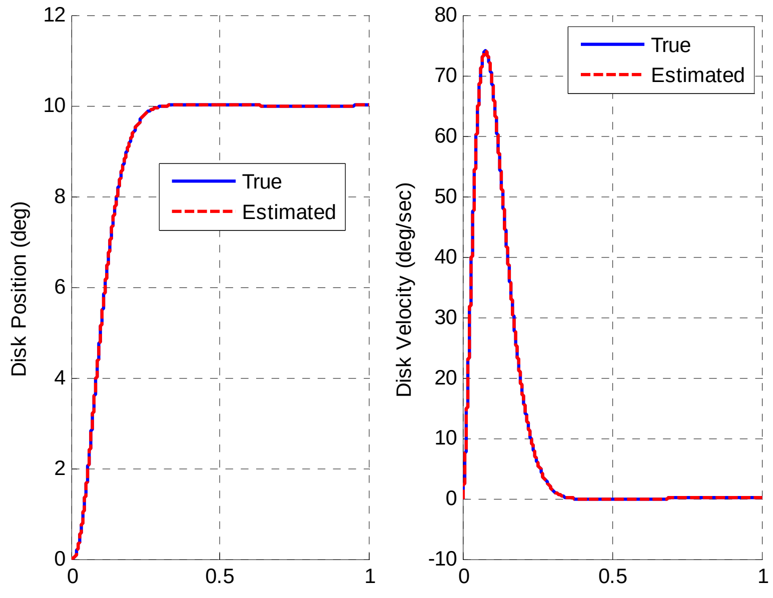

Figure 18. Full order observer state variable feedback with integrator (problem 5b).

c) Now we will try and observe what happens when there is a difference between our model and the real system. Modify the observer block in the model file, so the A matrix is called AA. Just before the you run the model file, enter the command AA = A*1.5. You should get results like those in Figure 19.

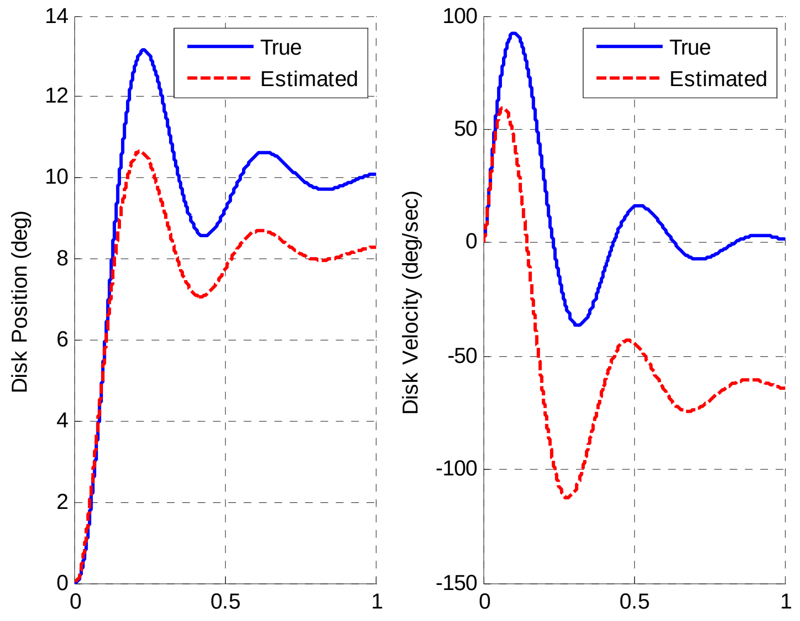

Figure 19. Full order observer with integrator and bad data (problem 5c).

d) Now change the observer poles so they are 10 larger than before (10*po). You should get results like those in Figure 20.

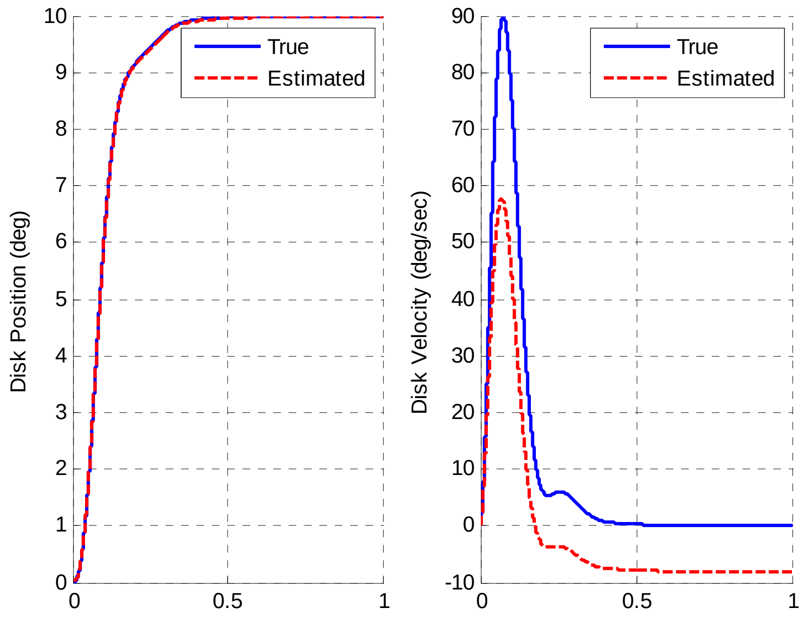

Figure 20. Full order observer with integrator and bad data (problem 5d).

Note that in both Figures 19 and 20, the steady state error is zero, even if the observer is not converged.

6) In this problem we will incorporate a full order observer and an integrator for a two degree of freedom system and run some test cases to verify your code is working. You may need to include a vector Cy to correctly set the prefilter gain for the disk you are trying to control.

a) Copy your program **sv2_observer_driver.m** to a new file **sv2_observer_int_driver.m,** then copy the Simulink file **sv2_observer.mdl** to **sv2_observer_int.mdl.** Modify these ***new*** files to implement state variable feedback with an observer using integral control.

b) For an input of 10 degrees, a final time of 1 second, both the state feedback poles at ps = [-10, -12, -14, -16, -18], the observer poles at po = [-10 -12 -14 -16], the output of the system is the position of the second disk and the input to the observer is the position of both disks, you should get a graph like that in Figure 21.

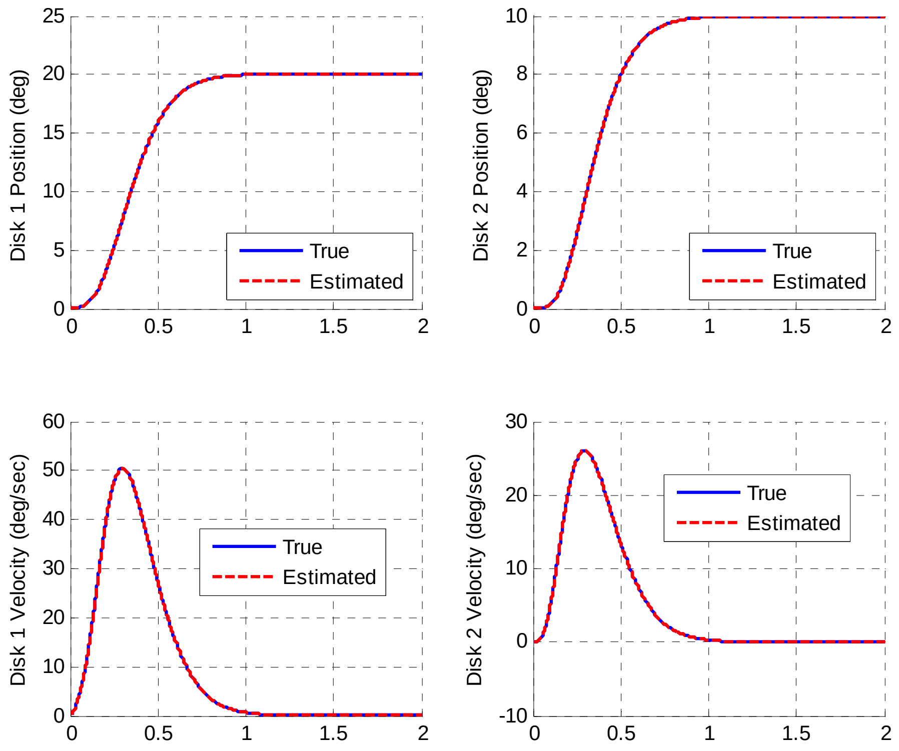

Figure 21. State feedback, full order observer, and integral control for the 2 dof system (problem 6b).

c) Now we will try and observe what happens when there is a difference between our model and the real system. Modify the observer block in the model file, so the A matrix is called AA. Just before you run the model file, enter the command AA = A*1.5. You should get results like those in Figure 22.

d) Now change the observer poles to be 10 times larger than before (10*po). You should get the figure like that in Figure 23.

e) Rerun the same case as in part d, but now use AA = 0.5*A and the output is the position of the first disk. You should get results like those in Figure 24.

f) Rerun the same case as in part e, but now make the state feedback poles twice as large as they originally were (2*ps, leave the observer poles where they are). You should get results like those in Figure 25.

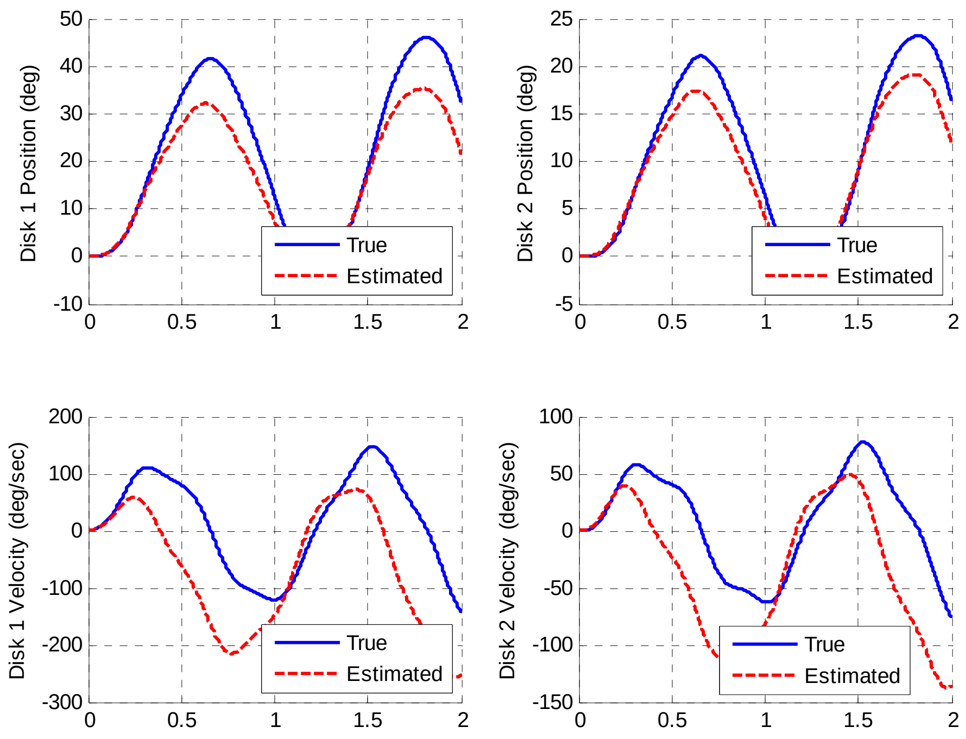

Figure 22. State feedback, full order observer, and integral control for the 2 dof system with a bad model (problem 6c).

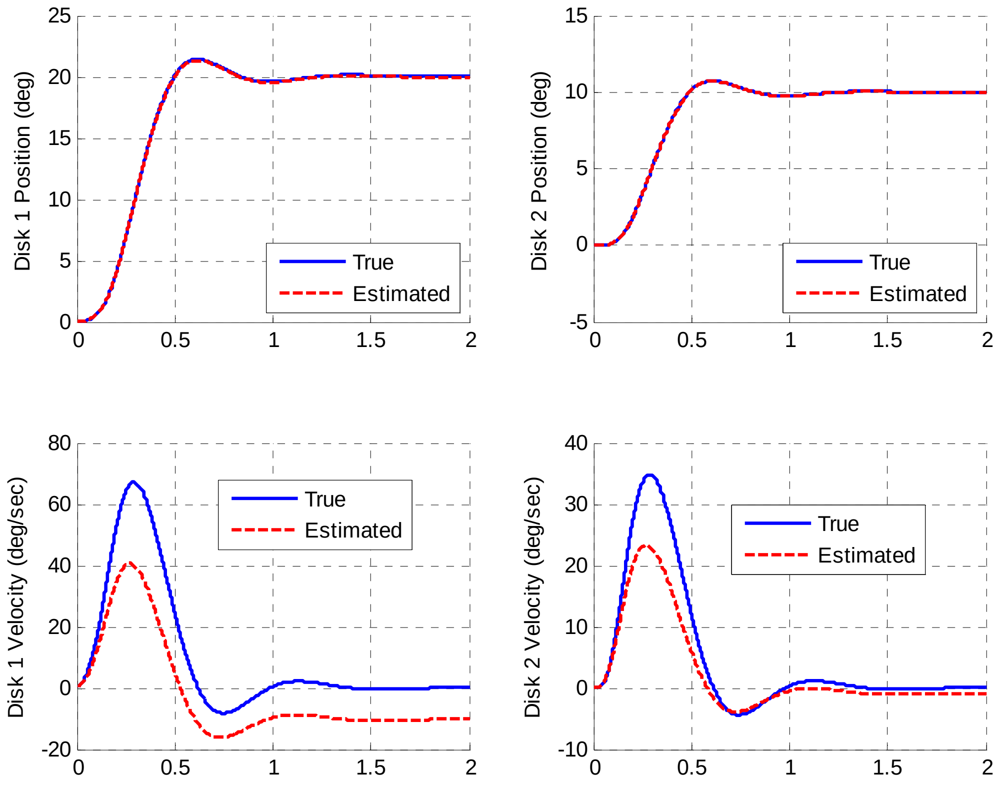

Figure 23. State feedback, full order observer, and integral control for the 2 dof system with a bad model (problem 6d).

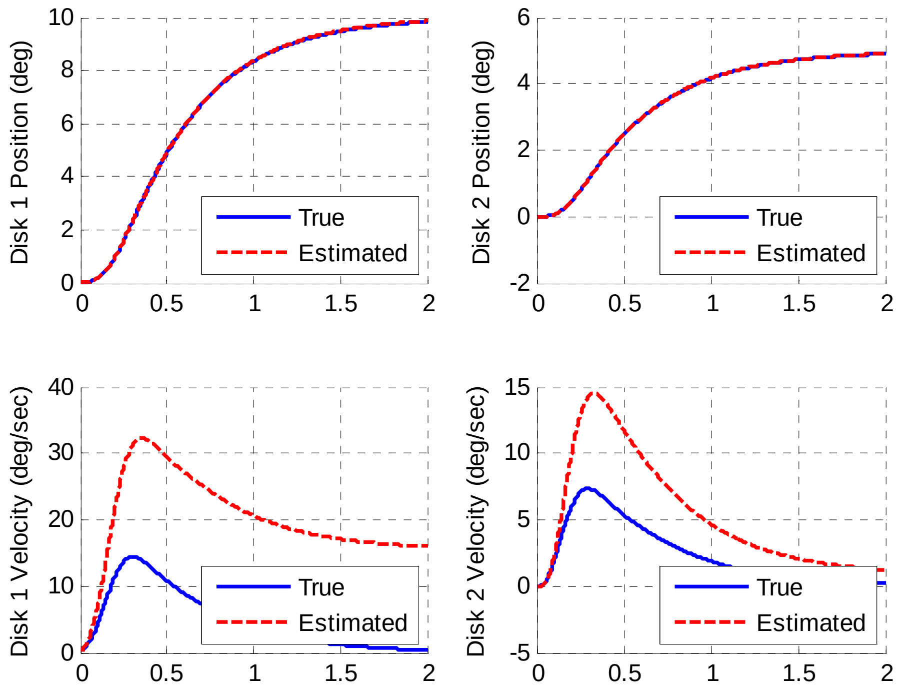

Figure 24. State feedback, full order observer, and integral control for the 2 dof system with a bad model (problem 6e).

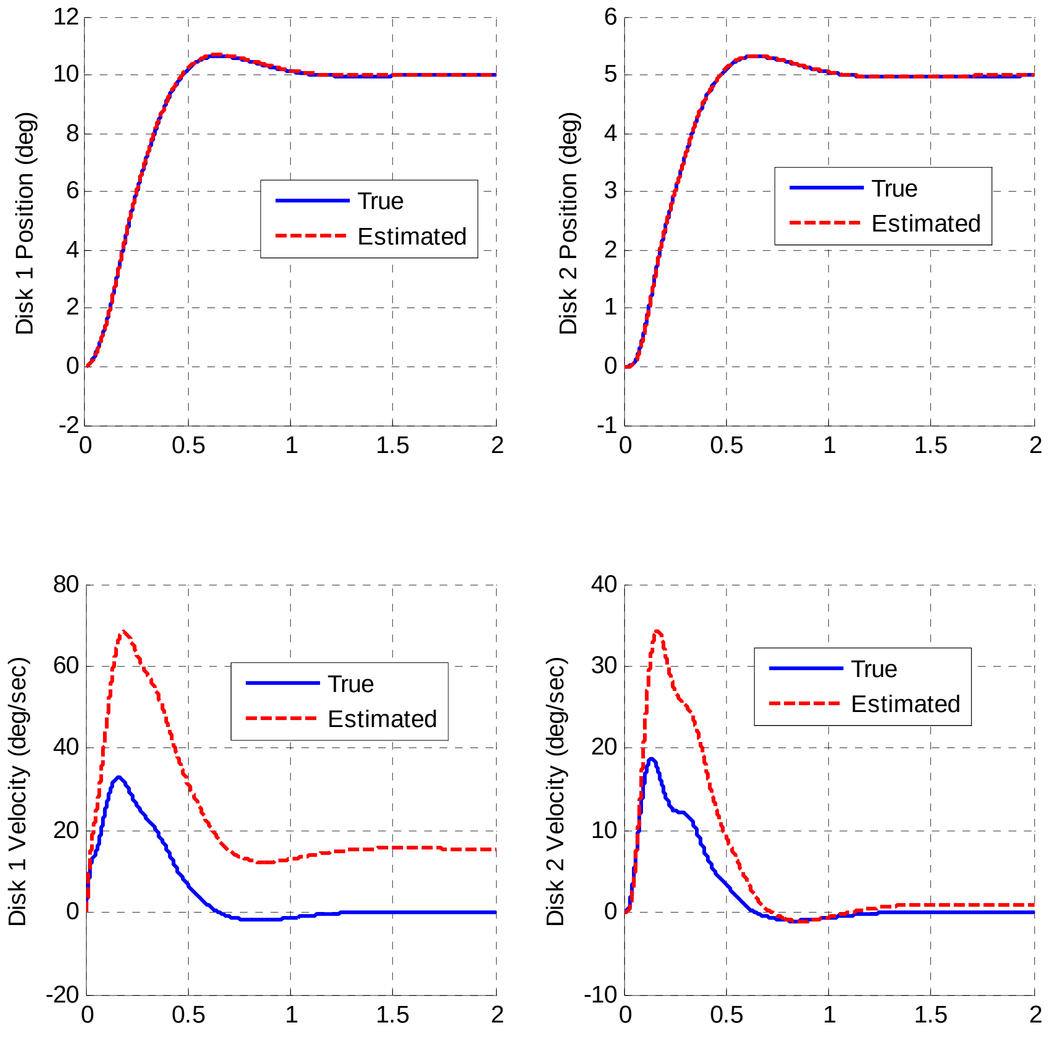

Figure 25. State feedback, full order observer, and integral control for the 2 dof system with a bad model (problem 6f).

You need to turn in all of the graphs you have produced, and also your two degree of freedom simulink models (just copy the models to your word document).