Feedback Simulation

Nidhi Sinha

December 2020

Abstract

Feedback systems have applications in everything from physics to economics. In this project, I simulate a general feedback system to explore how the stability of open gain affects the system's outputs. I also create two separate models to experiment with how the delay filter affects the output signal.

1 Introduction

Feedback systems are created when the output of said system is entered again as an input, forming a loop. This has many real-world applications, such as in biology and climate sciences. For example, drought is a positive feedback loop, meaning it only intensifies over time; as the soil dries up, less water vapor is released into the atmosphere, causing less clouds to form, causing less rain to fall upon the soil.

While there are many types of feedback systems, in this project, I am simulating a general electrical signal feedback system in Matlab. I experiment with varying stabilities of open gain to explore its effects on the output of the system. I am using Professor Charles Peskin's notes from his class on Mathematical Modeling and Simulation at New York University as a reference.

2 Governing Equations

2.1 Gain, Delay, Low-Pass Filters

Feedback systems, at the bare minimum, require an input signal, a gain, and an output signal. Gain is the factor by which the input signal is multiplied to get the output signal. Gain changes the feedback signal to be of a different frequency. The formula for a system that only has gain is

$$r = G(s - r) \tag{1}$$

which can be simplified to

$$r = \frac{G}{1+G}s \tag{2}$$

where s and r are the input and output signals respectively, and G is the gain. One may note that when the gain is very large, the output signal is almost exactly equal to the input.

In real-life feedback systems, delays are inevitable. Delay filters cause these delays in the system by filtering the signal before feeding it back. To account for a delay, you must modify the initial formula (1) to be

$$r(t) = G(s(t) - r(t - \tau_D)) \tag{3}$$

where t is the time and τ_D is the delay filter.

Another possible filter in a feedback system is the low-pass filter, which lowers the frequency of the signal if it is too high before feeding it back into the system. Since the feedback signal is now different from the original input signal, I designate q as the feedback signal and τ_F as the low-pass filter.

$$\tau_F \frac{dr}{dt} + r(t) = G(s(t) - q(t)) \tag{4}$$

However, we have not yet defined what $q(t)$ would be in this equation. Therefore we must do a series of calculations accounting for the low-pass filter. We also introduce a variable called τ_F to account for the delay introduced by the low-pass filter and also take the rate of change of q over time as $\frac{dq}{dt}$. Through a series of calculations on (4) as shown by (5) and (6) below we are able to derive (7) as follows.

$$q(t) + \tau_F \frac{dq}{dt} = r(t) = G(s(t) - q(t)) \tag{5}$$

$$(1 + G)q(t) + \tau_F \frac{dq}{dt} = Gs(t) \tag{6}$$

$$q(t) = (\frac{\tau_F}{1+G})\frac{dq}{dt} = \frac{G}{1+G}s(t) \tag{7}$$

Finally if we assume that $s(t) = 1$ for any time $t > 0$ and that for any time $t < 0$, $s(t), q(t), r(t) = 0$, then we get the final equation for any $t > 0$ as

$$q(t) = \frac{G}{1+G}(1 - e^{-(1+G)(\frac{t}{\tau_F})}) \tag{8}$$

2.2 Real-World Feedback Systems

Feedback systems in the real world typically have both a low-pass and delay filter. So how would we model that? We explore two options.

If we have both the low-pass and delay filter on the feedback path, we need to account for the changes to the feedback signal, q, over time. We get the equation

$$q(t) + \tau_F \frac{dq}{dt}(t) = r(t - \tau_D) = G(s(t - \tau_D) - q(t - \tau_D)) \tag{9}$$

However, if we place the low-pass filter on the feed-forward path, we get the equation

$$\tau_F \frac{dr}{dt} + r(t) = G(s(t) - r(t - \tau_D)) \tag{10}$$

In either model, we need to calculate the G value. Since this paper explores the stability of G, I calculate the critical value of $G*$. The equation calculating the critical value of G, notated as $G*$, can be derived from (9) to yield the following

$$G* = \sqrt{(1 + (\Theta \frac{\tau_F}{\tau_D})^2)} \tag{11}$$

where Θ must fulfill the equation

$$\Theta + arctan(\Theta \frac{\tau_F}{\tau_D}^2) = \pi \tag{12}$$

One may note that when $\frac{\tau_F}{\tau_D}$ is significantly smaller than 1, $G* = 1$. However, when $\frac{\tau_F}{\tau_D}$ is significantly larger than 1, $G*$ approaches $\frac{\pi}{2}$.

3 Numerical Scheme

For the numerical scheme, we need to first calculate the value of $G*$ which is dependent on Θ. In order to find the critical points of G, equations (11) and (12) are needed. First, $\Theta \frac{\tau_F}{\tau_D}$ must be calculated, which can be derived by (12) as follows.

$$arctan(\Theta \frac{\tau_F}{\tau_D}) = \pi - \Theta \tag{13}$$

$$\Theta \frac{\tau_F}{\tau_D} = tan(\pi - \Theta) \tag{14}$$

Using (14), we can now calculate $G*$ by substituting $\Theta \tau_F / \tau_D$ into (11).

$$G* = \sqrt{1 + (tan(\pi - \Theta))^2} \tag{15}$$

$$G* = \frac{1}{cos(\pi - \Theta)} \tag{16}$$

Now using (16) we can vary the G value of our equation as needed, by either setting it less than, equal to, or greater than $G*$.

Next, we need to use Modified Forward Euler's Method on both models. For our first model, which has the low-pass and delay filters on the feedback path, we need to calculate the feedback signal, since the output signal here is dependent on it.

$$q(t + \Delta t) = (1 - \frac{\Delta t}{\tau_F})q(t) + \frac{\Delta t}{\tau_F} G(s(t - \tau_D) - q(t - \tau_D)) \tag{17}$$

$$r(t) = G(s(t) - q(t)) \tag{18}$$

For our second model, which has the low-pass filter on the feed-forward path, we instead model the equation

$$r(t+\Delta t) = \Delta t \frac{G(s(t) - r(t-\tau_D)) - r(t)}{\tau_F} + r(t) \tag{19}$$

4 Code

The code for this simulation was relatively straightforward. First, I created a matrix of zeroes to store all of the q values over time and a matrix of zeroes to store all of the r values over time. I made both of these slightly larger than the size of the time step. Then using (15), I calculated the critical point of G and saved that in a variable. I set a timer initialized to 0, which was then increased in increments of the set time step, as the simulation ran. Finally, I created a for loop iterating through the q matrix to determine each value of q and r as t, the time, increased. With the critical value stored as a variable, I could now change the G value each time I ran the simulation to find the optimal stability.

5 Results

5.1 Determining G*

In order to determine the critical value of G, I created a graph of all possible $G*$ values with respect to $\Theta T_F/T_D$. Since I had chosen for T_F/T_D to equal 10, I simply referred to the graph and estimated what the critical value at that point would be. As according to the figure below, I approximated the critical value to be 16.5 when $\tau_F/\tau_D = 10$.

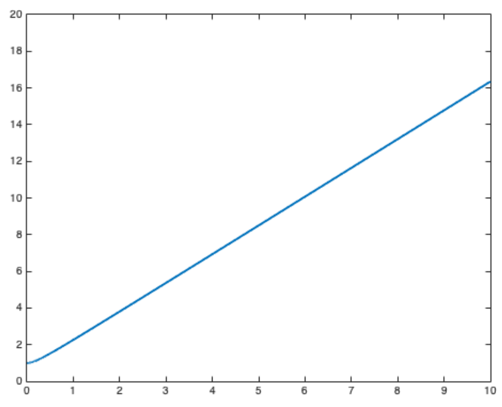


Figure 1: $G*$ as a function of τ_F/τ_D

5.2 Model 1

For this model, I placed the low-pass filter on the feed-forward path, and the delay filter on the feedback path. So, I use (17) to calculate the feedback signal and (19) to calculate the output signal given the feedback signal.

I experimented with varying G three times: $G = 1/2G*$, $G \approx G*$, and $G = 2G*$. I found that the optimal stability for the graphs was when $G = 1/2G*$. As you can see in the following figures, Figure 1 shows a stable graph that quickly stabilizes. Figure 2 shows an unstable graph whose output signal oscillates without actually stabilizing. Finally, Figure 3 shows an extremely unstable graph that oscillates at an increasing rate, rather than stabilizing, as the G value is far too big.

Note that the feedback delay causes the output signal to stop until the feedback signal is finished being processed, before resuming output. The blue line denotes the output signal while the red line denotes the feedback signal.

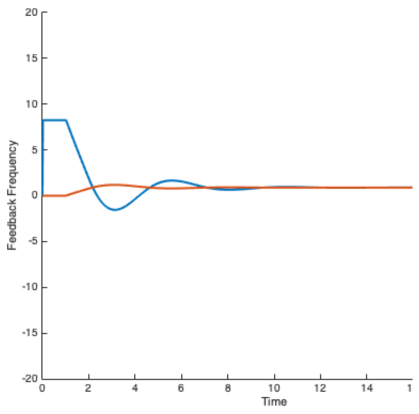


Figure 2: $G = \frac{1}{2}G*$

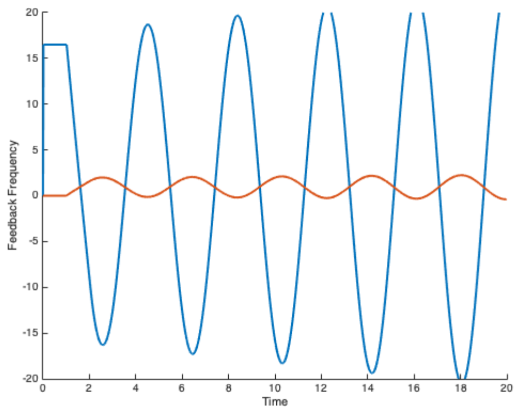


Figure 3: $G \approx G*$

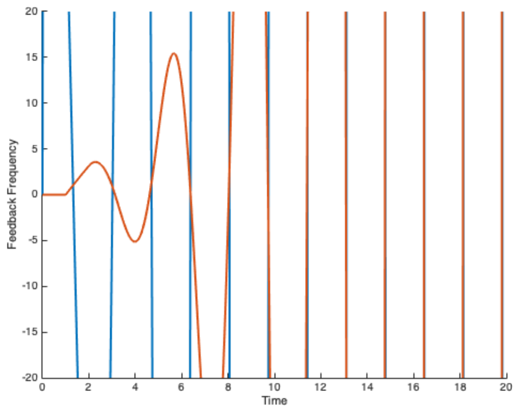


Figure 4: $G = 2G*$

5.3 Model 2

For this model, I placed the low-pass and delay filter on the feedback path and then calculated the output signal. I used (18) to model the effects of varying G values on the output signal.

The results are similar to Model 1's except the delay is not as noticeable here. However, the system clearly is at its most stable when G is significantly smaller than its critical value, and at its least stable when G is significantly larger than its critical value.

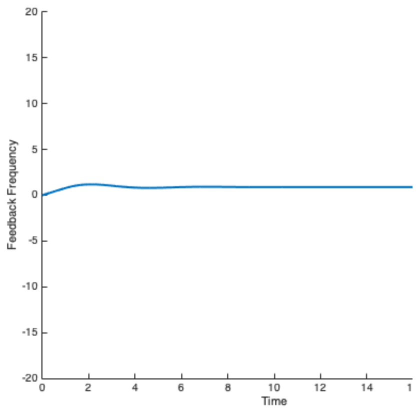


Figure 5: $G = \frac{1}{2}G*$

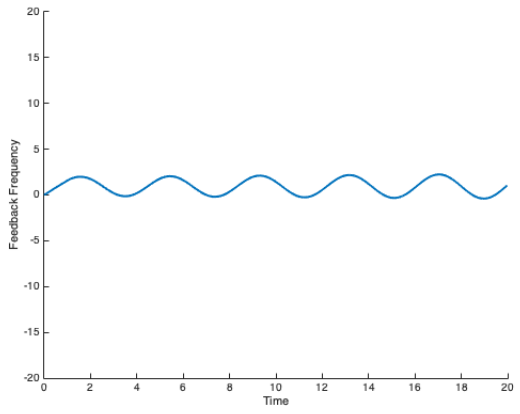


Figure 6: $G \approx G*$

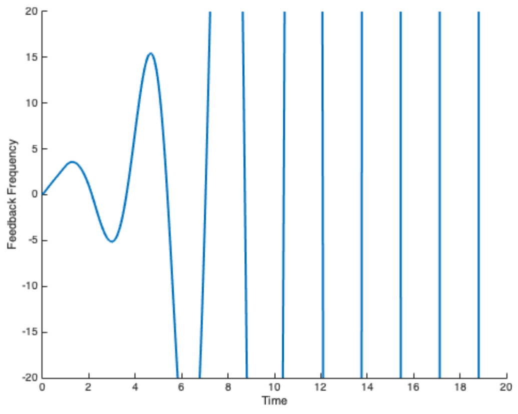


Figure 7: $G = 2G*$

6 Ideas for Further Study

For further study, one could experiment with the effect of noise on the system in conjunction with the delay and low-pass filters.

7 Conclusion

In conclusion, we can use our governing equations to create a realistic model of a signal feedback system to run experiments on. In both models, the output signal was at its most stable state when the gain was less than its critical point. Therefore, it is crucial to make sure the gain is significantly lower than the critical point to ensure that the system stabilizes over time. If the gain is significantly larger than the critical point, the system becomes extremely unstable.

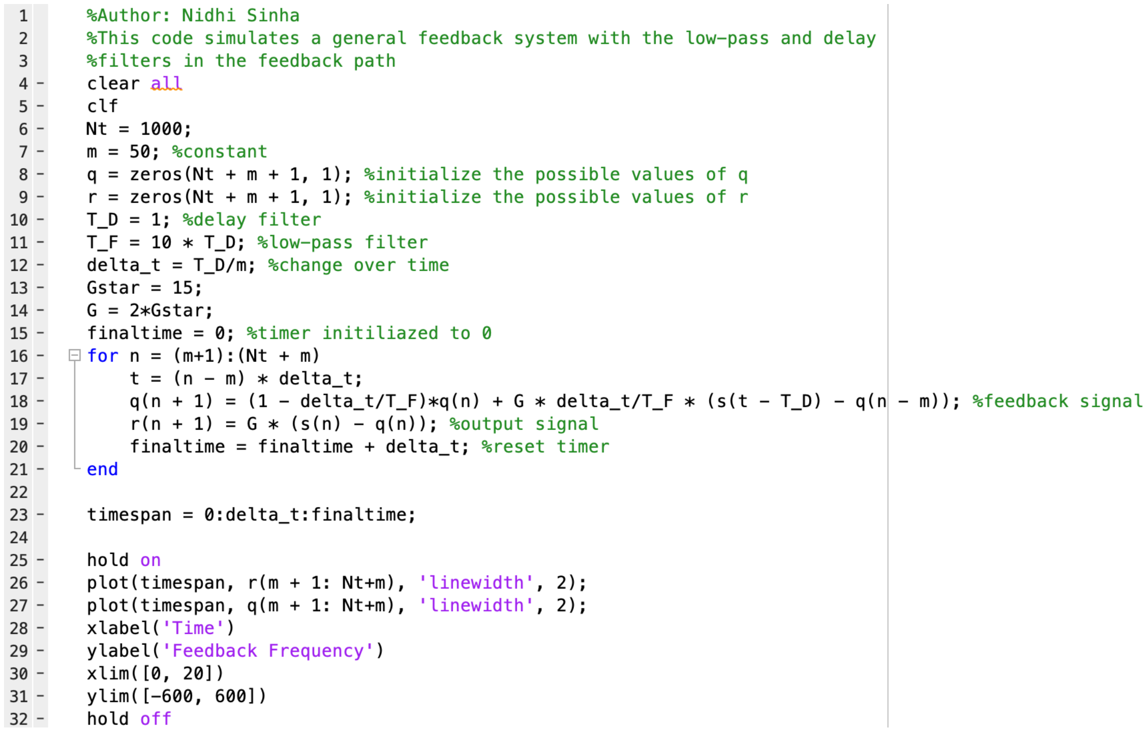
8 Acknowledgements

I'd like to thank Professor Charles Peskin and Scott Weady from the Courant Institute of Mathematical Sciences at New York University for their assistance with this project.

9 References

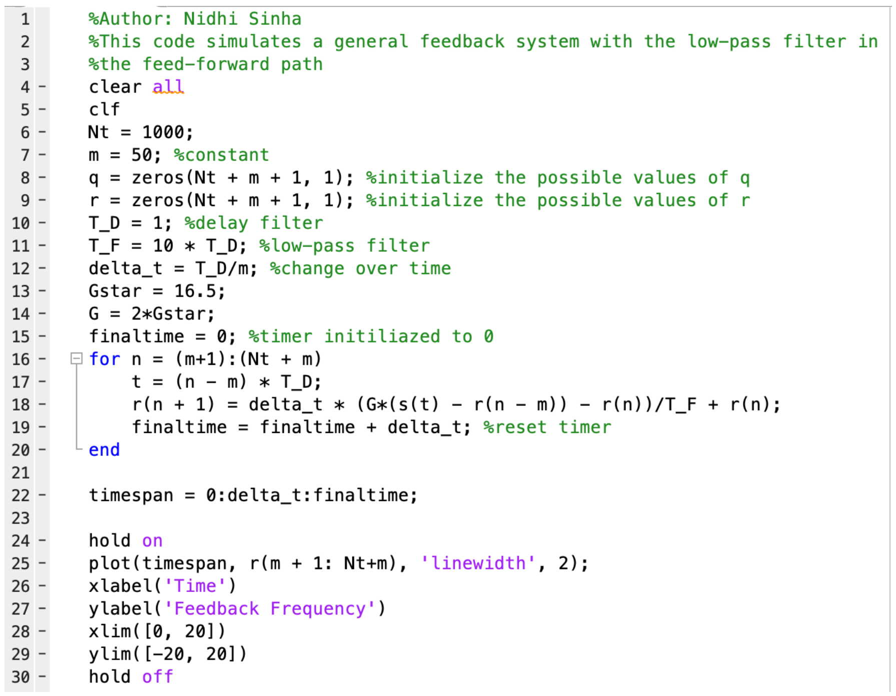
1. Charles Peskin's lecture notes on Feedback Systems (Fall 2020).
2. Charles Peskin's code (Fall 2020).

10 Appendix



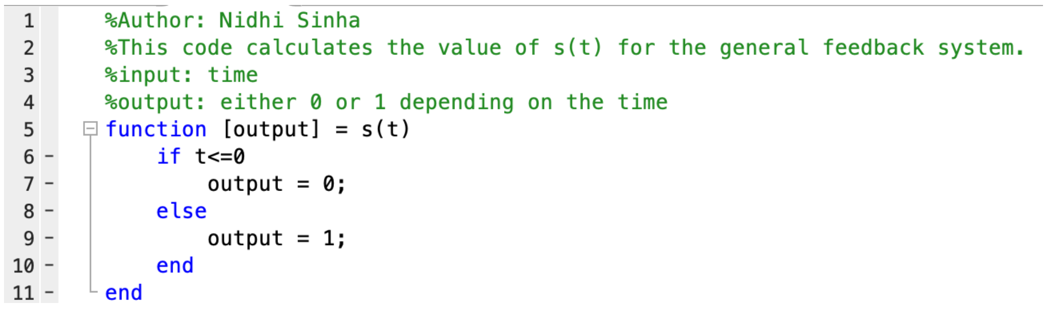
```matlab
%Author: Nidhi Sinha
%This code simulates a general feedback system with the low-pass and delay
%filters in the feedback path
clear all
clf
Nt = 1000;
m = 50; %constant
q = zeros(Nt + m + 1, 1); %initialize the possible values of q
r = zeros(Nt + m + 1, 1); %initialize the possible values of r
T_D = 1; %delay filter
T_F = 10 * T_D; %low-pass filter
delta_t = T_D/m; %change over time
Gstar = 15;
G = 2*Gstar;
finaltime = 0; %timer initiliazed to 0
for n = (m+1):(Nt + m)
    t = (n - m) * delta_t;
    q(n + 1) = (1 - delta_t/T_F)*q(n) + G * delta_t/T_F * (s(t - T_D) - q(n - m)); %feedback signal
    r(n + 1) = G * (s(n) - q(n)); %output signal
    finaltime = finaltime + delta_t; %reset timer
end

timespan = 0:delta_t:finaltime;

hold on
plot(timespan, r(m + 1: Nt+m), 'linewidth', 2);
plot(timespan, q(m + 1: Nt+m), 'linewidth', 2);
xlabel('Time')
ylabel('Feedback Frequency')
xlim([0, 20])
ylim([-600, 600])
hold off
```

Figure 8: Model 1



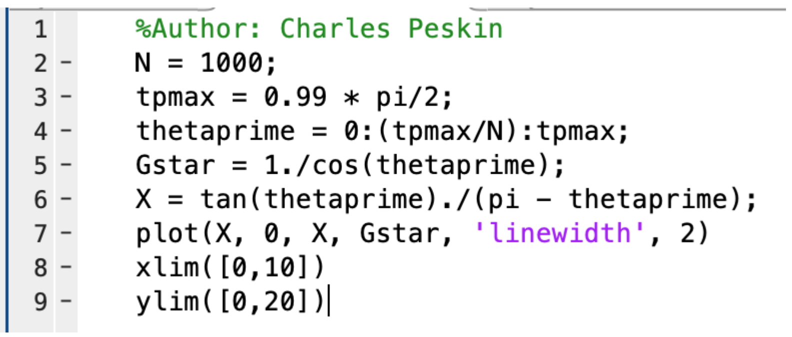
```
%Author: Nidhi Sinha
%This code simulates a general feedback system with the low-pass filter in
%the feed-forward path
clear all
clf
Nt = 1000;
m = 50; %constant
q = zeros(Nt + m + 1, 1); %initialize the possible values of q
r = zeros(Nt + m + 1, 1); %initialize the possible values of r
T_D = 1; %delay filter
T_F = 10 * T_D; %low-pass filter
delta_t = T_D/m; %change over time
Gstar = 16.5;
G = 2*Gstar;
finaltime = 0; %timer initiliazed to 0
for n = (m+1):(Nt + m)
    t = (n - m) * T_D;
    r(n + 1) = delta_t * (G*(s(t) - r(n - m)) - r(n))/T_F + r(n);
    finaltime = finaltime + delta_t; %reset timer
end

timespan = 0:delta_t:finaltime;

hold on
plot(timespan, r(m + 1: Nt+m), 'linewidth', 2);
xlabel('Time')
ylabel('Feedback Frequency')
xlim([0, 20])
ylim([-20, 20])
hold off
```

Figure 9: Model 2



```
%Author: Nidhi Sinha
%This code calculates the value of s(t) for the general feedback system.
%input: time
%output: either 0 or 1 depending on the time
function [output] = s(t)
    if t<=0
        output = 0;
    else
        output = 1;
    end
end
```

Figure 10: S(t) function



```
%Author: Charles Peskin
N = 1000;
tpmax = 0.99 * pi/2;
thetaprime = 0:(tpmax/N):tpmax;
Gstar = 1./cos(thetaprime);
X = tan(thetaprime)./(pi - thetaprime);
plot(X, 0, X, Gstar, 'linewidth', 2)
xlim([0,10])
ylim([0,20])
```

Figure 11: G* function