Software Implementation of Digital Filtering via Tustin's Bilinear Transform

Connor W. Herron

January 27, 2025

Abstract

The purpose of this work is to provide some notes on a software implementation for digital filtering via Tustin's Bilinear Transform. The first section discusses how to solve for the input/output coefficients by hand using a generalized approach called Horner's method. The second section presents some results of this generalized digital filtering approach using the IHMC Open Robotics Software (ORS) stack and Simulation Construction Set 2 (SCS2) [2, 3]. This generalized approach can solve for the digital coefficients for any causal transfer function.

1 Introduction

The purpose of this package is to build digital filters which can run in control loops at a consistent known frequency, f_l. We can represent filters in their continuous transfer function form:

$$H(s) = \frac{Y(s)}{X(s)} = \frac{b_0 s^{\mathrm{m}} + b_1 s^{\mathrm{m}-1} + ... + b_{\mathrm{m}-1}s + b_{\mathrm{m}}}{a_0 s^{\mathrm{n}} + a_1 s^{\mathrm{n}-1} + ... + a_{\mathrm{n}-1}s + a_{\mathrm{n}}} \tag{1}$$

where m $\leq$ n for the transfer function to be causal. We are looking to convert the transfer function, $H(s)$, to a digital filter of the following form

$$
\begin{aligned}
y_0 &= \hat{\mathrm{b}}_1 \mathrm{y}_1 + \hat{\mathrm{b}}_2 \mathrm{y}_2 + ... + \hat{\mathrm{b}}_{\mathrm{n}-1}\mathrm{y}_{\mathrm{n}-1} + \hat{\mathrm{a}}_0 \mathrm{x}_0 + \hat{\mathrm{a}}_1 \mathrm{x}_1 + ... + \hat{\mathrm{a}}_{\mathrm{n}-1}\mathrm{x}_{\mathrm{n}-1} \\
&= \hat{\mathbf{b}}^T \mathbf{y} + \hat{\mathbf{a}}^T \mathbf{x}
\end{aligned}
\tag{2}
$$

where $\hat{\mathbf{b}}, \mathbf{y} \in \mathbb{R}^{\mathrm{n}-1 \times 1}$, $\hat{\mathbf{b}} = \mathrm{col}[\hat{\mathrm{b}}_1, \hat{\mathrm{b}}_2, ..., \hat{\mathrm{b}}_{\mathrm{n}-1}]$ and $\hat{\mathbf{a}}, \mathbf{x} \in \mathbb{R}^{\mathrm{n} \times 1}$, $\hat{\mathbf{a}} = \mathrm{col}[\hat{\mathrm{a}}_0, \hat{\mathrm{a}}_1, ..., \hat{\mathrm{a}}_{\mathrm{n}-1}]$ where $\hat{b}_i, \hat{a}_i \in \mathbb{R}$ are scalars with $0 \leq i \leq n$ refers to the output order. The vectors $\mathbf{x} = \mathrm{col}[\mathrm{x}_0, \mathrm{x}_1, ..., \mathrm{x}_{\mathrm{n}-1}]$ and $\mathbf{y} = \mathrm{col}[\mathrm{y}_1, \mathrm{y}_2, ..., \mathrm{y}_{\mathrm{n}-1}]$ represent the input/output "history" where the subscript represents the loop iteration (eg. x_0 is the current input, x_1 is the previous input, x_2 is the previous previous input, etc.). Equation (2) must be solved in a loop running at a consistent frequency, f_l, where the input/output histories must be updated. Failure to achieve a consistent frequency will no longer represent the desired transfer function. The input/output coefficient vectors $\hat{\mathbf{a}}$, $\hat{\mathbf{b}}$ are constant throughout the control sequence and only need to be solved during initialization. **The goal of this work is to describe a general approach which solves for the digital representation of any continuous, causal transfer function, which translates to solving for a set of input/output coefficients, $\hat{\mathbf{a}}$, $\hat{\mathbf{b}}$ to be used in Eq. (2) as part of a control loop.**

2 Examples

The digital transformation used here is Tustin's Bilinear Transform method, which simply represents the continuous frequency variable

$$s = f_l \cdot ln(z) = 2f_l \cdot \left[\frac{z-1}{z+1} + \frac{1}{3} \cdot \left(\frac{z-1}{z+1}\right)^3 + \frac{1}{5} \cdot \left(\frac{z-1}{z+1}\right)^5 + \frac{1}{7} \cdot \left(\frac{z-1}{z+1}\right)^7 + ... \right]$$

$$s = 2f_l \frac{z-1}{z+1} \tag{3}$$

where f_l is the digital loop frequency at which Eq. (2) is solved. After solving the example using the general method, I'll show an alternative approach which is more software friendly.

2.1 General Method

2.1.1 Example 1: General Method

This first example is a simple 1st order, unity-gain, low-pass transfer function of the form

$$H(s) = \frac{1}{\frac{s}{\omega_0} + 1}, \quad \omega_0 = 0.1 \, \text{rad/s}$$

$$H(s) = \frac{1}{10s + 1} \tag{4}$$

where we want to solve for a set of digital input/output coefficient vectors $\hat{\mathbf{a}}, \hat{\mathbf{b}}$ from Eq. (2).

If we have a loop frequency of $f_l = 0.1$ Hz (chosen simply for mathematical convenience, and considerably lower than normal loop speeds), we can substitute Eq. (3) in (4) to get

$$H[z] = \frac{1}{10 \cdot (2 \cdot 0.1\frac{z-1}{z+1}) + 1}$$

$$H[z] = \frac{z+1}{2 \cdot (z-1) + z + 1}$$

$$H[z] = \frac{Y[z]}{X[z]} = \frac{z+1}{3z-1}$$

where we can split the transfer function into $Y[z]$ and $X[z]$ components

$$Y[z](3z - 1) = X[z](z + 1)$$

$$Y[z](1 - \frac{1}{3}z^{-1}) = X[z](\frac{1}{3} + \frac{1}{3}z^{-1})$$

Finally, we can derive our digital filter

$$y_0 = \frac{1}{3}y_1 + \frac{1}{3}x_0 + \frac{1}{3}x_1 \tag{5}$$

where our set of input/output coefficient vectors is

$$\boxed{\hat{\mathbf{a}}^T = [\frac{1}{3}, \frac{1}{3}], \quad \text{and} \quad \hat{\mathbf{b}}^T = [\frac{1}{3}]} \tag{6}$$

While this approach is relatively simple to complete by hand, it's quite difficult to generalize to more difficult examples. In this next example, the desired transfer function is 2nd order.

2.1.2 Example 2: General Method

The 2nd order low-pass, Butterworth is a common filter of the form

$$H(s) = \frac{\omega_c^2}{s^2 + \sqrt{2}\omega_c s + \omega_c^2} \tag{7}$$

For simple numerical values, I'll choose $\omega_c = \sqrt{2}$ rad/s, with $f_l = 1$ Hz. We can start by immediately plugging in Eq. (3) where

$$H(s) = \frac{2}{s^2 + 2s + 2}$$

$$H[z] = \frac{2}{(2f_l\frac{z-1}{z+1})^2 + 2(2f_l\frac{z-1}{z+1}) + 2}$$

$$H[z] = \frac{2(z+1)^2}{(4f_l^2(z-1)^2 + 4f_l(z-1)(z+1)) + 2(z+1)^2}$$

$$H[z] = \frac{2(z^2 + 2z + 1))}{(4(z^2 - 2z + 1) + 4(z^2 - 1) + 2(z^2 + 2z + 1)}$$

$$H[z] = \frac{2z^2 + 4z + 2}{4z^2 - 8z + 4 + 4z^2 - 4 + 2z^2 + 4z + 2}$$

$$H[z] = \frac{Y[z]}{X[z]} = \frac{2z^2 + 4z + 2}{10z^2 - 4z + 2}$$

where we can finally split the transfer function into $Y[z]$ and $X[z]$ components

$$Y[z](10z^2 - 4z + 2) = X[z](2z^2 + 4z + 2)$$

$$Y[z](1 - 0.4z^{-1} + 0.2z^{-2}) = X[z](0.2 + 0.4z^{-1} + 0.2z^{-2})$$

Finally, our digital filter can be derived as

$$y_0 = 0.4y_1 - 0.2y_2 + 0.2x_0 + 0.4x_1 + 0.2x_2 \tag{8}$$

and the set of input/output coefficient vectors is

$$\boxed{\hat{\mathbf{a}} = [0.2,\ 0.4,\ 0.2], \quad \text{and} \quad \hat{\mathbf{b}} = [0.4,\ -0.2]} \tag{9}$$

Using the general method, it can be hard to write an algorithm to find $\hat{\mathbf{a}}$, $\hat{\mathbf{b}}$ for any transfer function. In addition, a generalized method could be computational expensive, which isn't a deal breaker since this task would be completed during initialization prior to loop running, but we may have hundreds of filters and it would be good to consider efficiency.

2.2 Horner's Method

Horner's method is a convienent way of solving the $H(s) \rightarrow H[z]$ conversion in a stepwise manner which is mathematically equivalent and software generalizable to any causal transfer function [1]. Horner's method relies on the fact that we can rewrite Tustin's bilinear transform as

$$\frac{1}{s} = \frac{1}{2f_l} \cdot \frac{z+1}{z-1}$$

$$= \frac{1}{2f_l} \cdot \frac{z+1-1+1}{z-1} \tag{10}$$

$$= \frac{1}{2f_l} \cdot (\frac{2}{z-1} + 1)$$

In this method, instead of simply substituting Eq. (3) into the transfer function, this method retains the polynomial orders of power while separately considering the numerator and denominator. The method will convert

$$F(s) \to F(x) \to F(x+1) \to F(\frac{1}{x}+1) \to F(\frac{2}{x}+1) \to F(\frac{2}{x-1}+1)$$

$$\equiv F\left(\frac{1}{2f_l} \cdot (\frac{2}{z-1}+1)\right) = F[z] \tag{11}$$

Note: The variable x is a general variable to transition between layers of the $F(s) \to F[z]$, where x is not equivalent between layers. In addition, the multiplication of $2f_l$ can take place in an step in transition where I've chosen to the first step where $F(s) \to F(x)$. In addition, there are steps here where the same multiplication will take place across the numerator and denominator for convenience while being mathematically equivalent.

2.2.1   Example 1: Horner's Method

This first example is a simple 1st order, unity-gain, low-pass transfer function of the form

$$H(s) = \frac{1}{\frac{s}{\omega_0}+1}, \quad \omega_0 = 0.1 \, \text{rad/s}$$

$$H(s) = \frac{1}{10s+1} \tag{12}$$

where $f_l = 0.1$ Hz and we want to solve for a set of digital input/output coefficient vectors $\hat{\mathbf{a}}, \hat{\mathbf{b}}$ from Eq. (2).

1) $[\boldsymbol{F(s) \to F(x)}]$ We can start by separating the numerator and denominator, divide the polynomial by s^n, then substitute $s = \frac{2f_l}{x}$. As previously mentioned, you can choose to multiply by $2f_l$ at any step in this transition.

Numerator	**Denominator**
$N(s) = 1$	$D(s) = 10s + 1$
$\frac{N(s)}{s} = \frac{1}{s}$	$\frac{D(s)}{s} = 10 + \frac{1}{s}$
$\frac{N(s)}{s} \to N(x) = \frac{x}{2f_l}$	$\frac{D(s)}{s} \to D(x) = 10 + \frac{x}{2f_l}$
$N(x) = \frac{x}{0.2}$	$D(x) = 10 + \frac{x}{0.2}$
$N(x) = 5x$	$D(x) = 10 + 5x$

2) $\boldsymbol{F(x) \to F(x+1)}]$ Next we can decrease the zeros by 1 using synthetic division where

Numerator	**Denominator**
$N(x) = 5x$	$D(x) = 5x + 10$

$$
\begin{array}{c|cc}
 & 5 & 0 \\
+ & \downarrow & 5 \\
\hline
 & \downarrow & |5 \\
\hline
 & 5 &
\end{array}
\qquad
\begin{array}{c|cc}
 & 5 & 10 \\
+ & \downarrow & 5 \\
\hline
 & \downarrow & |15 \\
\hline
 & 5 &
\end{array}
$$

$$N(x) \to N(x+1) = 5x + 5 \qquad D(x) \to D(x+1) = 5x + 15$$

4

3) $[F(x+1) \rightarrow F(\frac{1}{x}+1)]$ Replace all zeros w/ reciprocals, equivalent to flipping the coefficients.

Numerator	**Denominator**
$N(x+1) = 5x + 5$	$D(x+1) = 5x + 15$
$N(\frac{1}{x}+1) = 5x + 5$	$D(\frac{1}{x}+1) = 15x + 5$

4) $F(\frac{1}{x}+1) \rightarrow F(\frac{2}{x}+1)]$ Scale the polynomial zeroes by 2. Note can you could scale by either $\frac{1}{2}$ or 2 for the orders of power since the polynomials are in both the numerator and denominator.

Numerator

$$N(\frac{1}{x}+1) = 5x + 5$$

$$N(\frac{2}{x}+1) = 5(\frac{x}{2}) + 5$$

$$= 2.5x + 5$$

Denominator

$$D(\frac{1}{x}+1) = 15x + 5$$

$$D(\frac{2}{x}+1) = 15(\frac{x}{2}) + 5$$

$$= 7.5x + 5$$

5) $F(\frac{2}{x}+1) \rightarrow F(\frac{2}{x-1}+1)]$ Increase all polynomial zeros by 1 using synthetic division.

Numerator

$$N(\frac{2}{x}+1) = 2.5x + 5$$

$$
\begin{array}{c}
 \quad 2.5 \quad\ \ 5 \\
-\ \ \underline{\downarrow \quad 2.5} \\
\ \ \ \underline{\downarrow \quad |2.5} \\
\quad 2.5
\end{array}
$$

Denominator

$$D(\frac{2}{x}+1) = 7.5x + 5$$

$$
\begin{array}{c}
 \quad 7.5 \quad\ \ 5 \\
-\ \ \underline{\downarrow \quad 7.5} \\
\ \ \ \underline{\downarrow \quad |-2.5} \\
\quad 7.5
\end{array}
$$

$$N(\frac{2}{x}+1) \rightarrow N(\frac{2}{x-1}+1) = 2.5x + 2.5 \qquad D(\frac{2}{x}+1) \rightarrow D(\frac{2}{x-1}+1) = 7.5x - 2.5$$

6) $F(x) = F[z]]$ Substitute in $x = z$ to equivalently write the transfer function as

$$H[z] = \frac{Y[z]}{X[z]} = \frac{N[z]}{D[z]} = \frac{2.5z + 2.5}{7.5z - 2.5} \tag{13}$$

7) Derive the digital filter where

$$Y[z](7.5z - 2.5) = X[z](2.5z + 2.5)$$

$$Y[z](1 - \frac{1}{3}z^{-1}) = X[z](\frac{1}{3} + \frac{1}{3}z^{-1})$$

where finally we have our digital filter.

$$y_0 = \frac{1}{3}y_1 + \frac{1}{3}x_0 + \frac{1}{3}x_1 \tag{14}$$

with input/output coefficients of

$$\boxed{\hat{\mathbf{a}}^T = [\frac{1}{3}, \frac{1}{3}], \quad \text{and} \quad \hat{\mathbf{b}}^T = [\frac{1}{3}]} \tag{15}$$

Note that this solution matches the solution from the general approach shown in (6).

2.2.2 Example 2: Horner's Method

The 2nd order low-pass, Butterworth is a common filter of the form

$$H(s) = \frac{\omega_c^2}{s^2 + \sqrt{2}\omega_c s + \omega_c^2} \tag{16}$$

For simple numerical values, I'll choose $\omega_c = \sqrt{2}$ rad/s, with $f_l = 1$ Hz. So our transfer function is of the form

$$H(s) = \frac{2}{s^2 + 2s + 2}$$

1) $[\boldsymbol{F(s) \to F(x)}]$ Separate the numerator and denominator, divide each polynomial by s^n, and then substitute $s = \frac{2f_l}{x}$.

Numerator	**Denominator**
$N(s) = 2$	$D(s) = s^2 + 2s + 2$
$\dfrac{N(s)}{s^2} = \dfrac{2}{s^2}$	$\dfrac{D(s)}{s^2} = 1 + \dfrac{2}{s} + \dfrac{2}{s^2}$
$\dfrac{N(s)}{s^2} \to N(x) = \dfrac{x^2}{2}$	$\dfrac{D(s)}{s^2} \to D(x) = 1 + x + \dfrac{x^2}{2}$
$N(s) \to N(x) = 0.5x^2$	$D(s) \to D(x) = 0.5x^2 + x + 1$

2) $[\boldsymbol{F(x) = F(x + 1)}]$ Decrease the zeros by 1 using synthetic division

Numerator	**Denominator**

$$N(x) = 0.5x^2 \qquad\qquad\qquad\qquad D(x) = 0.5x^2 + x + 1$$

```
        0.5   0     0                        0.5   1     1
    +    ↓   0.5   0.5                   +    ↓   0.5   1.5
        0.5   0.5  |0.5                      0.5   1.5  |2.5
    +    ↓   0.5                         +    ↓   0.5
    +   0.5   |1                         +   0.5   |2
        0.5                                  0.5
```

$$N(x + 1) = 0.5x^2 + x + 0.5 \qquad\qquad D(x + 1) = 0.5x^2 + 2x + 2.5$$

3) $[\boldsymbol{F(x + 1) = F(\frac{1}{x} + 1)}]$ Replace all zeros w/ reciprocals, which is equivalent to flipping the coefficients

Numerator	**Denominator**
$N(x + 1) = 0.5x^2 + x + 0.5$	$D(x + 1) = 0.5x^2 + 2x + 2.5$
$N(\dfrac{1}{x} + 1) = 0.5x^2 + x + 0.5$	$D(\dfrac{1}{x} + 1) = 2.5x^2 + 2x + 0.5$

4) $[\boldsymbol{F(\frac{1}{x} + 1) = F(\frac{2}{x} + 1)}]$ Scale the polynomial zeros by 2 (or $\frac{1}{2}$).

Numerator	**Denominator**

$$N(\frac{1}{x}+1) = 0.5x^2 + x + 0.5 \qquad D(\frac{1}{x}+1) = 2.5x^2 + 2x + 0.5$$

$$N(\frac{2}{x}+1) = 0.5(\frac{x}{2})^2 + (\frac{x}{2}) + 0.5 \qquad D(\frac{2}{x}+1) = 2.5(\frac{x}{2})^2 + 2(\frac{x}{2}) + 0.5$$

$$N(\frac{2}{x}+1) = 0.125x^2 + 0.5x + 0.5 \qquad D(\frac{2}{x}+1) = 0.625x^2 + x + 0.5$$

5) $[F(\frac{2}{x}+1) = F(\frac{2}{x-1}+1)]$ Increase all polynomial zeros by 1 using synthetic division.

Numerator	**Denominator**

$$N(x) = 0.125x^2 + 0.5x + 0.5 \qquad D(x) = 0.625x^2 + x + 0.5$$

Numerator synthetic division:

	0.125	0.5	0.5
$-$	$\downarrow$ 0.125	0.375	
	0.125	0.375	$\underline{|0.125}$
$-$	$\downarrow$ 0.125		
$-$	0.125	$\underline{|0.25}$	
	0.125		

Denominator synthetic division:

	0.625	1	0.5
$-$	$\downarrow$ 0.625	0.375	
	0.625	0.375	$\underline{|0.125}$
$-$	$\downarrow$ 0.625		
$-$	0.625	$\underline{|-0.25}$	
	0.625		

$$N(x) = 0.125x^2 + 0.25x + 0.125 \qquad D(x) = 0.625x^2 - 0.25x + 0.125$$

6) $[F(x) = F[z]]$ Substitute in $x = z$ and equivalently write the digital transfer function as

$$H[z] = \frac{Y[z]}{X[z]} = \frac{N[z]}{D[z]} = \frac{0.125z^2 + 0.25z + 0.125}{0.625z^2 - 0.25z + 0.125}$$

7) Derive the digital filter.

$$Y[z](0.625z^2 - 0.25z + 0.125) = X[z](0.125z^2 + 0.25z + 0.125)$$
$$Y[z](1 - 0.4z^{-1} + 0.2z^{-2}) = X[z](0.2 + 0.4z^{-1} + 0.2z^{-2})$$

Finally, our digital filter is

$$y_0 = 0.4y_1 - 0.2y_2 + 0.2x_0 + 0.4x_1 + 0.2x_2 \qquad (17)$$

where the digital input/output coefficients are

$$\boxed{\hat{\mathbf{a}}^T = [0.2,\ 0.4,\ 0.2], \quad \text{and} \quad \hat{\mathbf{b}}^T = [0.4,\ -0.2]} \qquad (18)$$

Notes that this solution is equivalent to that in the general method in (9).

3 Results

3.1 Chirp Signals

I'll show some comparisons of transfer functions using the intended model and results of an input/output response of that model's filter with an exponential chirp input. A chirp signal is a sinusoid whose frequency changes *linearly* or *exponentially* with time depending on the desired waveform. This type of signal can be used to sweep through a frequency spectrum for generating a bode plot of the linear system. Personally, I prefer using exponential chirp signals so that the number of data points spent on each frequency matches the log-scale graph of the bode plot (i.e. we get more data points on lower-frequencies that we usually care more about). Alternatively, one could use a sequence of sinusoids instead of a dynamic sinusoid. That allows you to separate the transient and steady state responses, which is otherwise impossible to separate in a chirp signal response.

Here's the equation of the frequency change, $\omega_c(t)$, with respect to time for linear and exponential chirp signals

$$\omega_c^{\text{lin}}(t) = \left(\frac{\omega_{\max} - \omega_{\min}}{T}\right) \cdot t + \omega_{\min} \tag{19}$$

$$\omega_c^{\text{exp}}(t) = \omega_{\min} \cdot \left(\frac{\omega_{\max}}{\omega_{\min}}\right)^{\frac{t}{T}} \tag{20}$$
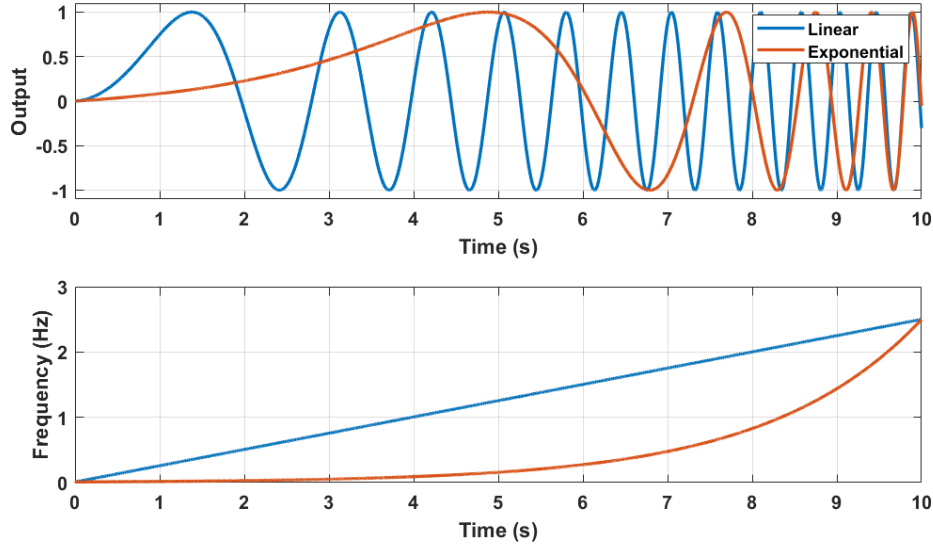

Figure 1: Linear and Exponential Chirp Waveforms from 0.01 to 2.5 Hz with amplitude of 1

Where $\omega_{\min}$ is the minimum frequency in rad/s, $\omega_{\max}$ is the maximum frequency in rad/s, and T is total time in seconds. Using (19) and (20), the chirp signal for a digital system can be

represented as

$$\cos_i = \cos(\omega_c[t] \cdot \Delta t) \cos_{i-1} - \sin(\omega_c[t] \cdot \Delta t) \sin_{i-1} \tag{21}$$

$$\sin_i = \sin(\omega_c[t] \cdot \Delta t) \cos_{i-1} + \cos(\omega_c[t] \cdot \Delta t) \sin_{i-1} \tag{22}$$

Finally, the chirp equation is represented by the following

$$x_i = A \sin_i \tag{23}$$

where A refers to the amplitude. Figure 1 shows a plot of linear and exponential chirp signals where $\omega_{\min} = 2\pi \cdot 0.01$ rad/s, $\omega_{\max} = 2\pi \cdot 2.5$ rad/s, $T = 10$ sec, and $A = 1$.

3.2   Software Implementation of Digital Filtering

The algorithm for generating and using the filter in a control loop can be summarized by the following pseudocode:

Algorithm 1 Using a filter in a control loop.

% Solve for Digital Coefficients, $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$, from Transfer Function, $H_s(s) = \frac{N_s(s)}{D_s(s)}$.
Check $n \geq m$
$N_z = \text{solve}_{\text{Fs} \to \text{Fz}}(N_s)$
$D_z = \text{solve}_{\text{Fs} \to \text{Fz}}(D_s)$
Scale N_z and D_z coefficients by $D_z[0]$.
Multiply D_z coefficients by -1 and solve for the $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$ column vectors.

firstTick = true

% Run Control Loop.
while Control Loop is active **do**
    % Initialize output and input history vectors to first input.
    if firstTick **then**
        $\mathbf{y} = x_0 \cdot \mathbf{col}(1)_{(\text{n}-1)\times 1}$ ▷ Heuristic for avoiding high outputs during initialization.
        $\mathbf{x} = x_0 \cdot \mathbf{col}(1)_{\text{n}\times 1}$
        firstTick = false
    end if

    % Update input vector, where $\mathbf{x} \in \mathbb{R}^{\text{n}\times 1}$.
    $\mathbf{x} = [x_0, x_1, ... x_{\text{n}-1}, x_{\text{n}}]^T$

    % Solve for next filtered output.
    $y_0 = \hat{\mathbf{b}}^T \mathbf{y} + \hat{\mathbf{a}}^T \mathbf{x}$

    % Update output vector, where $\mathbf{y} \in \mathbb{R}^{\text{n}-1\times 1}$.
    $\mathbf{y} = [y_0, y_1, ... y_{\text{n}-2}, y_{\text{n}-1}]^T$
end while

The following pages show some examples of the digital filtering performance compared to an ideal model. This experimentation is completed using IHMC's Open Robotics Software (ORS) [2] and Simulation Construction Set 2 (SCS2) [3].

3.2.1 1st Order Low-Pass Filter:

The following transfer function:

$$H(s) = \frac{1}{\frac{1}{\tau}s + 1}$$

where $\tau = 2\pi \cdot 10$. This transfer function can be digitally represented by the following input/output coefficient vectors with $f_l = 1000$ Hz.

$$\hat{\mathbf{a}} = [3.0459E - 02, 3.0459E - 02]$$

$$\hat{\mathbf{b}} = [9.3908E - 01]$$

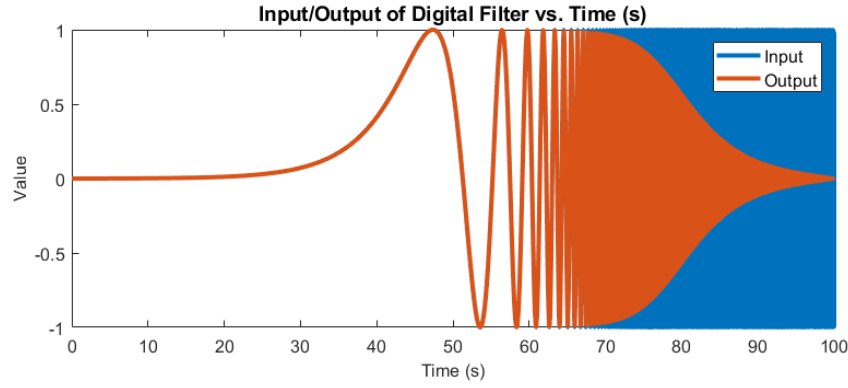The input/output and bode plot from this digital filter are as follows.
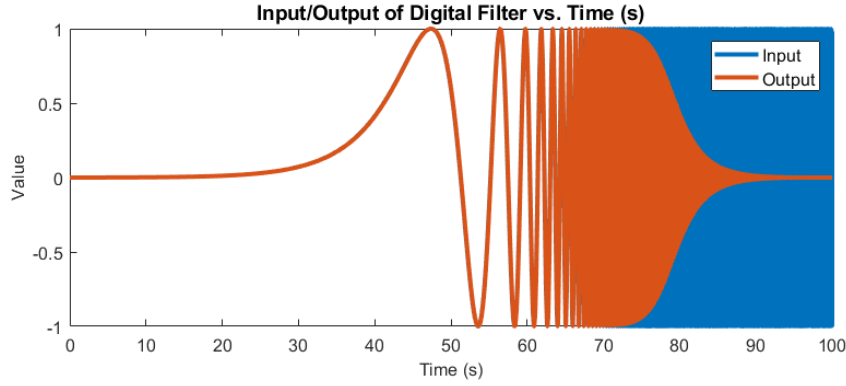

Figure 2: Input-Output plot of the 1st Order LP Filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz
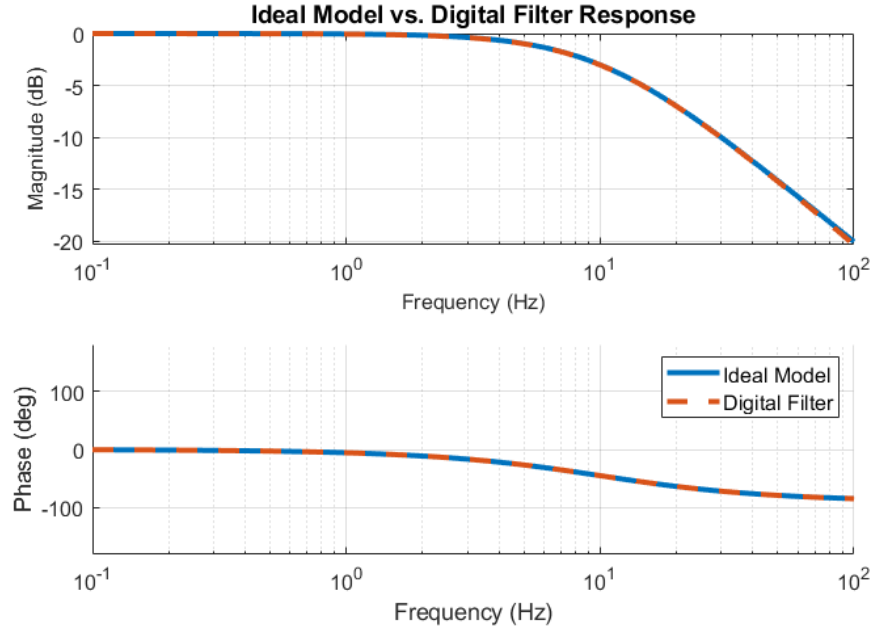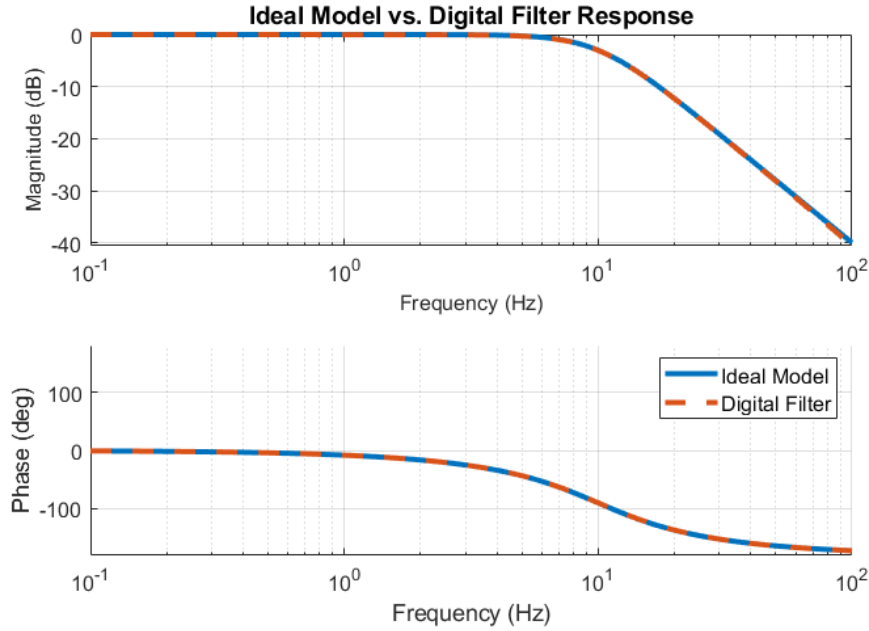

Figure 3: Bode Plot of filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

10

3.2.2 2nd order Low-Pass Butterworth Filter:

The following transfer function:

$$H(s) = \frac{\omega_c^2}{s^2 + \sqrt{2}\omega_c s + \omega_c^2}$$

where $\omega_c = 2\pi \cdot 10$. This transfer function can be digitally represented by the following input/output coefficient vectors with $f_l = 1000$ Hz.

$$\hat{\mathbf{a}} = [9.4408E - 04, 1.8882E - 03, 9.4408E - 04]$$
$$\hat{\mathbf{b}} = [1.9112E + 00, -9.1500E - 01]$$

The input/output and bode plot from this digital filter are as follows.

Figure 4: Input-Output plot of 2nd order LP Butterworth filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

Figure 5: Bode Plot of filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

3.2.3 Notch Filter:

The following transfer function:

$$H(s) = \frac{s^2 + \omega_n^2}{s^2 + \frac{\omega_n}{Q}s + \omega_n^2}$$

where $\omega_n = 2\pi \cdot 60$ and the quality factor, $Q = 5$. This transfer function can be digitally represented by the following input/output coefficient vectors with $f_l = 1000$ Hz.

$$\hat{\mathbf{a}} = [9.6487E - 01, -1.7973E + 00, 9.6487E - 01]$$
$$\hat{\mathbf{b}} = [1.7973E + 00, -9.2975E - 01]$$

The input/output and bode plot from this digital filter are as follows.
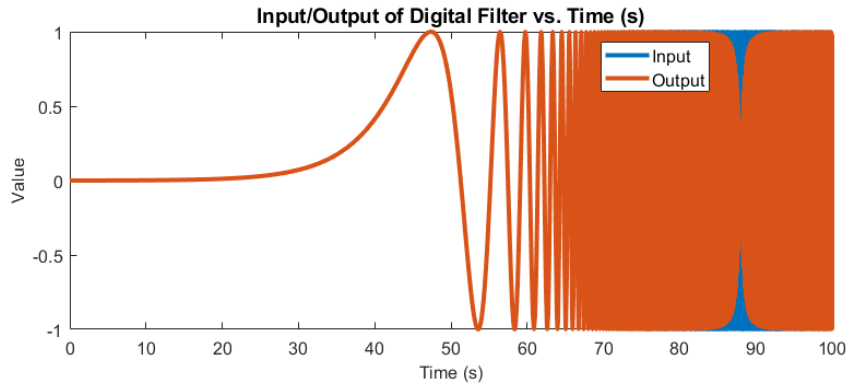
Figure 6: Input-Output plot of Notch Filter set to 60 Hz with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz
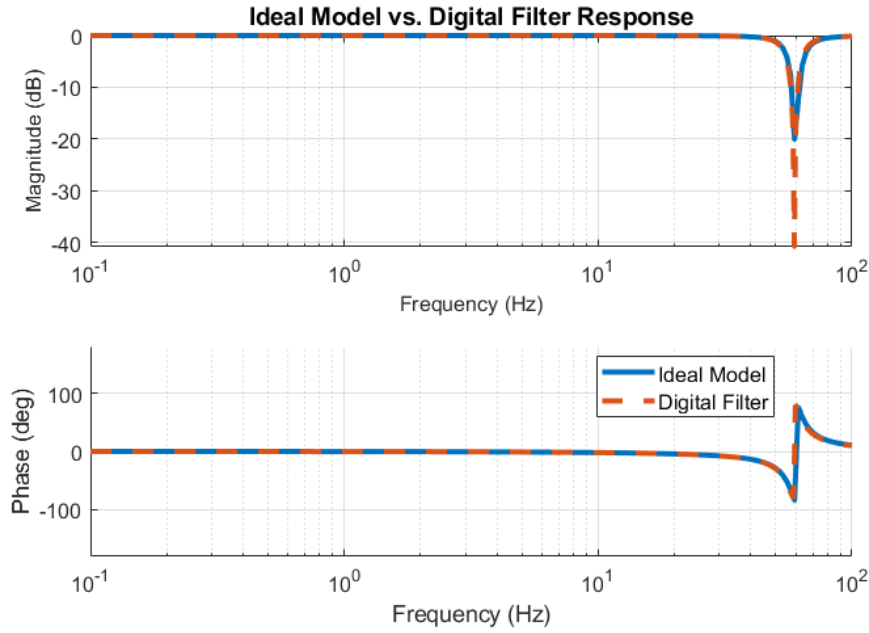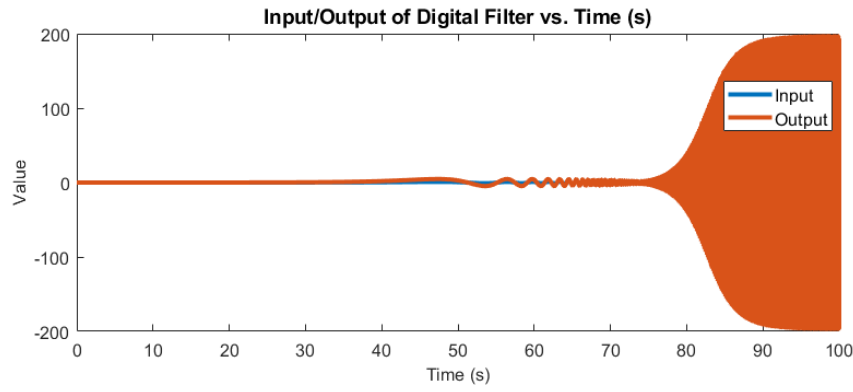

Figure 7: Bode Plot of filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

3.2.4 Complex Multi-order Filter:

The following transfer function:

$$H(s) = \frac{196.92s^3 + 21033.79s^2 + 427573.90s + 18317222.93}{s^3 + 382.16s^2 + 60851.34s + 3875784.59}$$

This transfer function can be digitally represented by the following input/output coefficient vectors with $f_l = 1000$ Hz.

$$\hat{\mathbf{a}} = [1.7198E + 02, -4.9816E + 02, 4.8074E + 02, -1.5455E + 02]$$
$$\hat{\mathbf{b}} = [2.6305E + 00, -2.3162E + 00, 6.8252E - 01]$$

The input/output and bode plot from this digital filter are as follows.

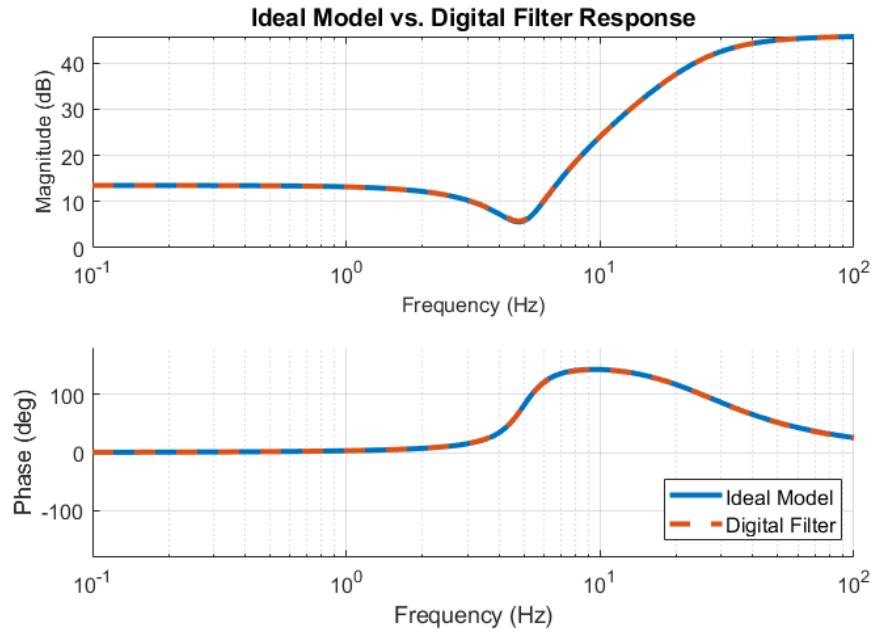Figure 8: Input-Output plot of multi-order filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

Figure 9: Bode Plot of filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

13

3.2.5 PID Controller:

A PID Controller can be represented by the following causal (and unstable) transfer function:

$$H(s) = (K_p + \frac{1}{s}K_i + \frac{\tau s}{s + \tau}K_d) = \frac{(K_p + K_d\tau)s^2 + (K_p\tau + K_i)s + K_i\tau}{s^2 + \tau s}$$

where $K_p = 15.0$, $K_i = 2$, $K_d = 0.25$, and $\tau = 0.0035$ which simply acts as a low-pass filter on the derivative term for cleaner signals and causality. This transfer function can be digitally represented by the following input/output coefficient vectors with $f_l = 1000$ Hz.

$$\hat{\mathbf{a}} = [1.5002E + 01, -3.0002E + 01, 1.5000E + 01]$$
$$\hat{\mathbf{b}} = [2.0000E + 00, -1.0000E + 00]$$

The input/output and bode plot from this digital filter are as follows.
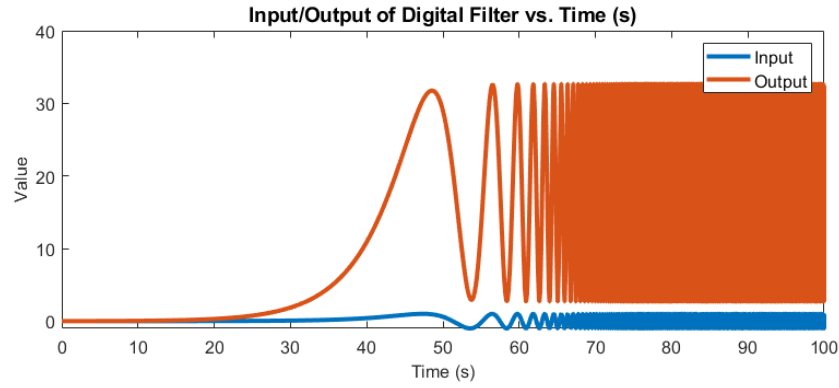

Figure 10: Input-Output plot of PID Controller with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz
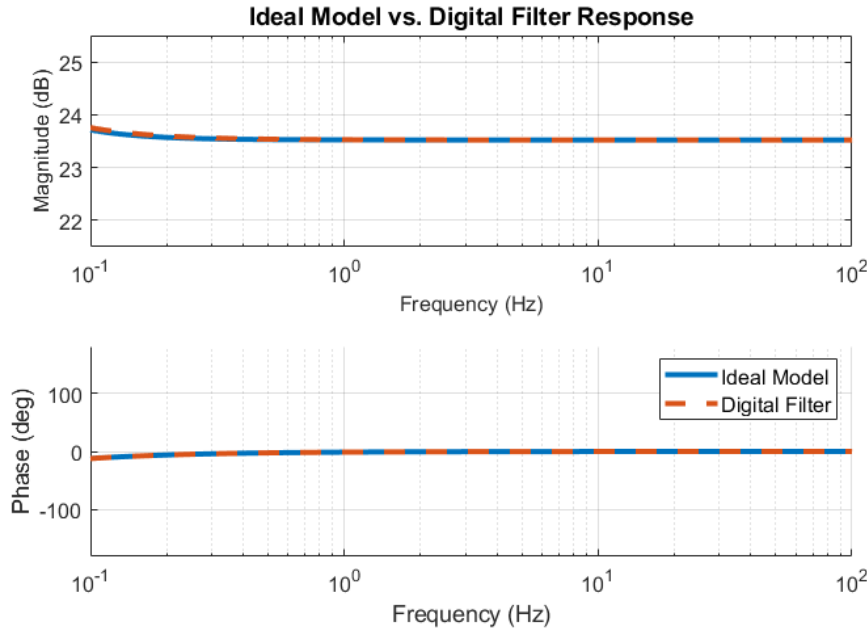

Figure 11: Bode Plot of filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

14

3.2.6 Lead-Lag Compensator:

The following transfer function:

$$H(s) = K\frac{s+z}{s+p}$$

where $K = 10.0$, $z = 2\pi \cdot 1.0$, and $p = 2\pi \cdot 10.0$. This transfer function can be digitally represented by the following input/output coefficient vectors with $f_l = 1000$ Hz.

$$\hat{\mathbf{a}} = [9.7259E + 00, -9.6650E + 00]$$
$$\hat{\mathbf{b}} = [9.3908E - 01]$$

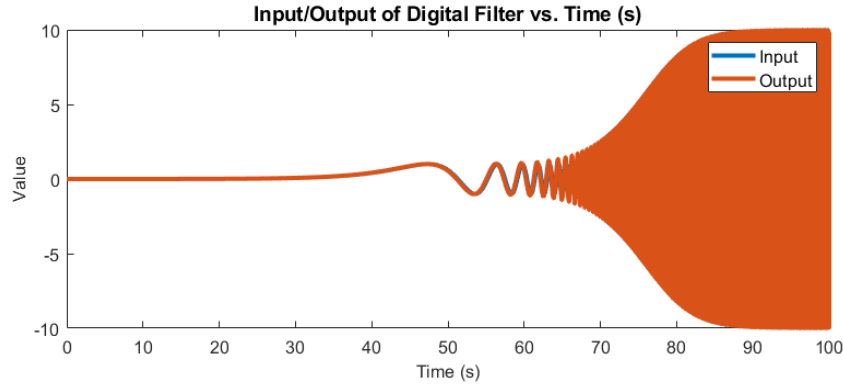The input/output and bode plot from this digital filter are as follows.

Figure 12: Input-Output plot of a Lead-Lag Compensator with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz
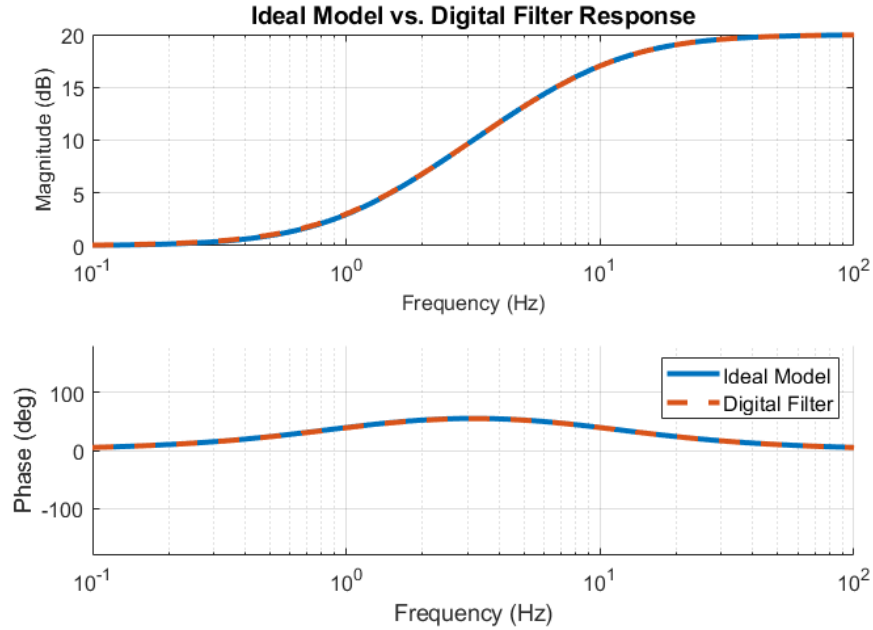

Figure 13: Bode Plot of filter with an exponential chirp input of 1 amplitude from 0.1 to 100 Hz

15

3.2.7 Heuristic for Avoiding High Outputs on Initialization

As mentioned in Algorithm 1, a heuristic is added to set the **x** and **y** history vectors to the initial input. Without this heuristic, the first input and output history vectors can be initialized to zero and tend to have very high solutions at the start. This effect can be shown for the following example with a 2nd order Low-Pass Butterworth and Lead-Lag Controller, with equivalent transfer functions to Sections 3.2.2 and 3.2.6, respectively. The input is a sinusoidal function of the form $x = \sin(2\pi \cdot 100t) + 5$.
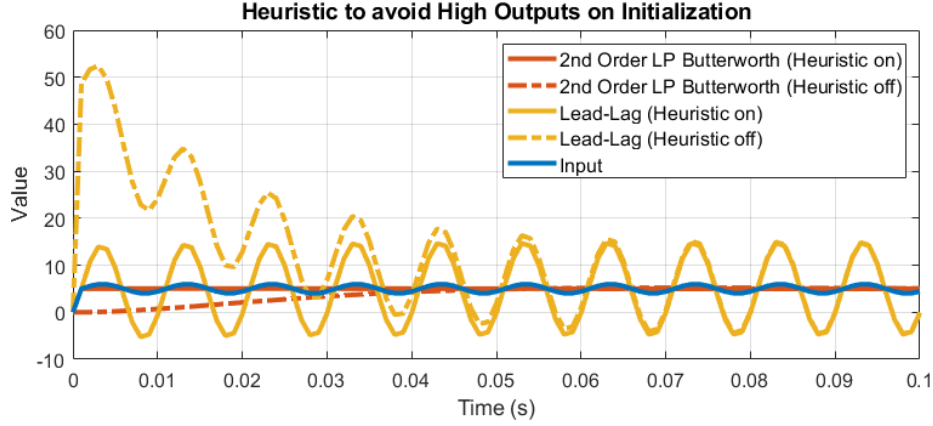

Figure 14: Start up heuristic to avoid large outputs for a 2nd order butterworth and lead-lag controller with an input of $x = \sin(2\pi \cdot 100t) + 5$.

References

[1] A Davies. Bilinear transformation of polynomials. *IEEE Transactions on Circuits and Systems*, 21(6):792–794, 1974.

[2] Institute of Human and Machine Cognition. IHMC Open Robotics Software. *ihmcrobotics.github.io*, January 5, 2024.

[3] Institute of Human and Machine Cognition. IHMC Simulation Construction Set 2. *github.com/ihmcrobotics/simulation-construction-set-2*, January 5, 2024.