# Frequency Response of Discretization Rules

Assignment 6

Due: 2021/03/22

## 1 Introduction

In order to gain insight into the behaviour of different discretization methods, this assignment analyzes the frequency response of a discretized inductor to observe the magnitude and phase effects across a frequency sweep. A frequency-dependent circuit equivalent in terms of $L_e$ and $R_e$ is obtained from the frequency-dependent admittance, which is derived in this assignment. The continuous solution is used as a baseline comparison; a ratio between it and the discretized solution allows us to obtain a measure of how the discretized solution deviates from the true solution.

## 2 Setup

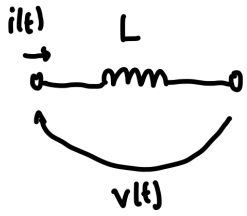


Figure 1: Voltage and current relationship of an inductor

Shown in Figure 1 is the circuit representation of an inductor, a fundamental electrical component which stores its energy in a magnetic field formed by the geometry and material of the conductor that forms it. As a result, it resists changes in current flowing through it, giving it a kind of momentum. Conversely, the magnetic field it forms cannot instantaneously collapse. The relationship of the current flowing through it and the voltage across its terminals is as follows:

$$v_L(t) = L \cdot \frac{di_L}{dt} \tag{1}$$

We see that voltage is a result of a differentiation of the current passing through the inductor multiplied by a constant L, the inductance in units of Henries. If we rearrange the equation so that $v_L(t)$ is the input, we get the following relationship:

$$i_L(t) = \frac{1}{L} \cdot \int v_L(t)dt \tag{2}$$

Now we see that if voltage is the input, the current passing through it is a result of the integration of the voltage multiplied by the reciprocal of the inductance.

These relationships show that the voltage and current of an inductor form a simple first-order system which we can represent with an input, an output, and some kind of operation in between. This is demonstrated in Figure 2.

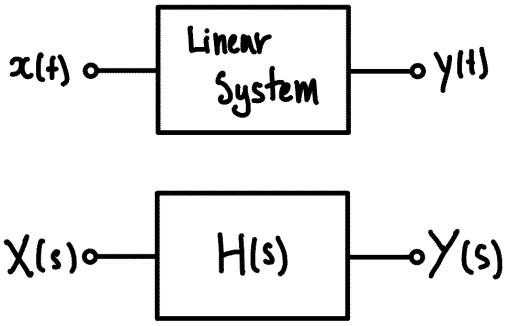


Figure 2: Frequency response of a linear system

In order to get a system's frequency response, we can take the linear system into the frequency domain where the first-order system becomes a transfer function $H(s)$. Since we are in the frequency domain, by assuming a single frequency input into the system, we know that the output will be some transfer function $H(s)$ multiplied by the input. In the case of the inductor where the voltage is the input, the system becomes an integrator as shown in Figure 3.

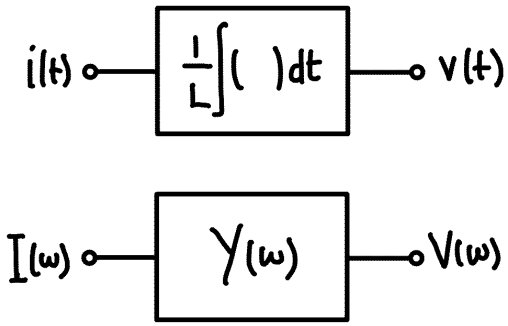


Figure 3: Frequency response of an integrator

The transfer function in the frequency domain becomes the admittance, as admittance times voltage equals current. Similarly, in the frequency domain, the differential relationship of current and voltage across

the inductor (where the current is differentiated with respect to time times the inductance), the transfer function becomes impedance. This relationship is shown in Figure 4.

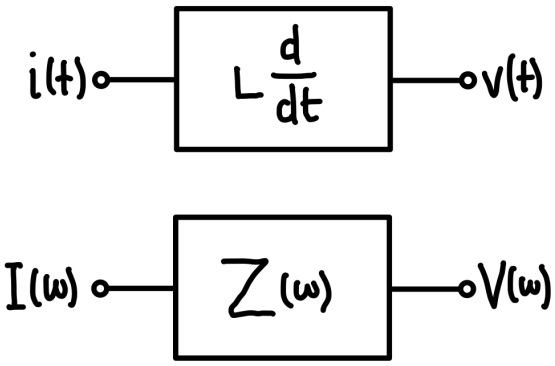


Figure 4: Frequency response of a differentiator

For this assignment, both transfer functions must be considered, as the discretizations used by the approximations use one or the other. Specifically, the assignment compares the following four discretizations of the inductor's current-voltage relationship:

- Trapezoidal (Integration Approximation):
- Backward Euler (Integration Approximation):
- Forward Euler (Integration Approximation):
- Gear's 2$^{\text{nd}}$ Order (Differentiation Approximation):

Trapezoidal, backward Euler, and forward Euler all approximate the integration of voltage to obtain the current. This yields a frequency-dependent admittance. Gear's 2$^{\text{nd}}$ order, on the other hand, approximates the derivative performed on the current to obtain the resulting voltage. This yields a frequency-dependent impedance. For the purpose of comparison when plotting the magnitude and phase distortions, Gear's 2$^{\text{nd}}$ order frequency-dependent impedance is inverted to yield the admittance.

The following sections will go over the frequency-domain equivalent circuits that the frequency-dependent admittance and impedance represent. A plot of the magnitude and phase distortions in comparison to the true answer is also generated. Finally, the results are discussed.

# 3 Results

## 3.1 Trapezoidal Discretization

The frequency response of the trapezoidal discretization is one of two equivalent circuits that are already provided in the class notes. Its frequency-domain equivalent circuit is shown in Figure 5, which consists of a single frequency-dependent inductor $L_e(\omega)$.

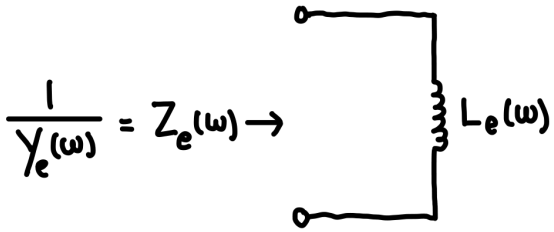


Figure 5: Frequency domain equivalent circuit of trapezoidal discretization

The derivation of the frequency-dependent admittance $Y_e(\omega)$ is shown below:

$$i(t) - i(t-\Delta t) = \frac{\Delta t}{2L}v(t) + \frac{\Delta t}{2L}v(t-\Delta t) \tag{3}$$

$$input:\ v(t) = e^{j\omega t} \tag{4}$$

$$output:\ i(t) = Y_e(\omega)e^{j\omega t} \qquad (I = VY) \tag{5}$$

$$Y_e(\omega)e^{j\omega t} - Y_e(\omega)e^{j\omega(t-\Delta t)} = \frac{\Delta t}{2L}e^{j\omega t} + \frac{\Delta t}{2L}e^{j\omega(t-\Delta t)} \tag{6}$$

$$\boxed{Y_e(\omega) = \frac{\Delta t}{2L}\frac{e^{j\omega t}+1}{e^{j\omega t}-1}} \tag{7}$$

With some algebraic manipulation and an application of Euler's identity, we can get $Y_e(\omega)$ into its impedance $Z_e(\omega)$ and subsequently its circuit equivalent component:

$$Z_e(\omega) = \frac{1}{Y_e(\omega)} \tag{8}$$

$$Z_e(\omega) = \frac{2L}{\Delta t}j\tan\left(\frac{\omega\Delta t}{2}\right) \tag{9}$$

$$\boxed{Z_e(\omega) = j\omega L_e(\omega)} \tag{10}$$

$$\boxed{L_e(\omega) = L\frac{\tan(\frac{\omega\Delta t}{2})}{\frac{\omega\Delta t}{2}}} \tag{11}$$

## 3.2 Backward Euler Discretization

The frequency response of the backward Euler discretization is the second of two equivalent circuits that are already provided in the class notes. Its frequency-domain equivalent circuit is shown in Figure 6, which consists of a constant resistance $R_e$ and a frequency-dependent inductor $L_e(\omega)$ in a parallel configuration. This is due to the transfer function being represented as an admittance, which means the relationship that it forms is the reciprocal addition of $R_e$ and $L_e(\omega)$. The equivalent impedance would therefore be equivalent to the two components in parallel.

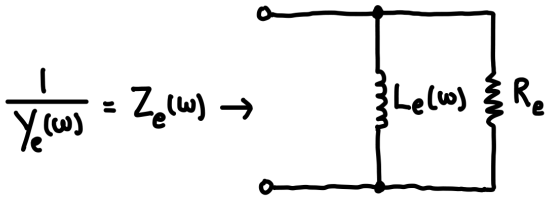


Figure 6: Frequency domain equivalent circuit of backward Euler discretization

The derivation of the frequency-dependent admittance $Y_e(\omega)$ is shown below:

$$i(t) - i(t - \Delta t) = \frac{\Delta t}{L} v(t) \tag{12}$$

$$input: \ v(t) = e^{j\omega t} \tag{13}$$

$$output: \ i(t) = Y_e(\omega) e^{j\omega t} \qquad (I = VY) \tag{14}$$

$$Y_e(\omega) e^{j\omega t} - Y_e(\omega) e^{j\omega(t-\Delta t)} = \frac{\Delta t}{L} e^{j\omega t} \tag{15}$$

$$\boxed{Y_e(\omega) = \frac{\Delta t}{L} \frac{e^{j\omega \Delta t}}{e^{j\omega \Delta t} - 1}} \tag{16}$$

With some algebraic manipulation and an application of Euler's identity, we can get $Y_e(\omega)$ into its impedance $Z_e(\omega)$ and subsequently its circuit equivalent components:

$$Y_e(\omega) = \frac{\Delta t}{2L} + \frac{\Delta t}{2L} \frac{1}{j \tan(\frac{\omega \Delta t}{2})} \tag{17}$$

$$Y_e(\omega) = \frac{1}{R_e} + \frac{1}{j\omega L_e(\omega)} \tag{18}$$

$$Z_e(\omega) = \frac{1}{Y_e(\omega)} = \frac{1}{\frac{1}{R_e} + \frac{1}{j\omega L_e(\omega)}} \tag{19}$$

$$\boxed{Z_e(\omega) = R_e || j\omega L_e(\omega)} \tag{20}$$

$$\boxed{R_e = \frac{2L}{\Delta t}} \tag{21}$$

$$\boxed{L_e(\omega) = L \frac{\tan(\frac{\omega \Delta t}{2})}{\frac{\omega \Delta t}{2}}} \tag{22}$$

### 3.3 Forward Euler Discretization

The frequency response of the forward Euler discretization is the first of two equivalent circuits that must be derived by hand. Its frequency-domain equivalent circuit is shown in Figure 7, which consists of a constant resistance $R_e$ and a frequency-dependent inductor $L_e(\omega)$ in a parallel configuration. Note that this is identical to the frequency-domain circuit representation of the backward Euler discretization. However, as the derivation will show, the value for $R_e$ is now negative.

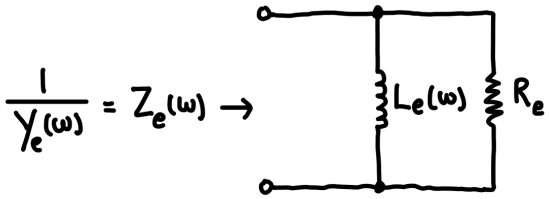


Figure 7: Frequency domain equivalent circuit of forward Euler discretization

The derivation of the frequency-dependent admittance $Y_e(\omega)$ is shown below:

$$\int_{t-\Delta t}^{t} v(t)dt = L\left[i(t) - i(t-\Delta t)\right] \tag{23}$$

$$\int_{t-\Delta t}^{t} v(t)dt \simeq v(t-\Delta t)\Delta t \tag{24}$$

$$v(t-\Delta t)\frac{\Delta t}{L} = i(t) - i(t-\Delta t) \tag{25}$$

$$input:\ v(t) = e^{j\omega t} \tag{26}$$

$$output:\ i(t) = Y_e(\omega)e^{j\omega t} \qquad (I = VY) \tag{27}$$

$$\frac{\Delta t}{L}e^{j\omega(t-\Delta t)} = Y_e(\omega)e^{j\omega t} - Y_e(\omega)e^{j\omega(t-\Delta t)} \tag{28}$$

$$Y_e(\omega) = \frac{\Delta t}{L}\frac{e^{j\omega(t-\Delta t)}}{e^{j\omega t} - e^{j\omega(t-\Delta t)}} \tag{29}$$

$$\boxed{Y_e(\omega) = \frac{\Delta t}{L}\frac{1}{e^{j\omega\Delta t} - 1}} \tag{30}$$

With some algebraic manipulation and an application of Euler's identity, we can get $Y_e(\omega)$ into its impedance $Z_e(\omega)$ and subsequently its circuit equivalent components:

$$Y_e(\omega) = \frac{\Delta t}{L}\frac{1}{e^{j\omega\Delta t} - 1} \tag{31}$$

$$Y_e(\omega) = \frac{\Delta t}{L}\frac{e^{-j\frac{\omega\Delta t}{2}}}{e^{j\frac{\omega\Delta t}{2}} - e^{-j\frac{\omega\Delta t}{2}}} \tag{32}$$

$$Y_e(\omega) = \frac{\Delta t}{L}\frac{\cos(-\frac{\omega\Delta t}{2}) = j\sin(-\frac{\omega\Delta t}{2})}{2j\sin(-\frac{\omega\Delta t}{2})} \tag{33}$$

$$Y_e(\omega) = \frac{\Delta t}{2L}\left[\frac{\cos(\frac{\omega\Delta t}{2})}{j\sin(\frac{\omega\Delta t}{2})} - \frac{j\sin(\frac{\omega\Delta t}{2})}{j\sin(\frac{\omega\Delta t}{2})}\right] \tag{34}$$

$$Y_e(\omega) = \frac{\Delta t}{2L}\left[\frac{1}{j\tan(\frac{\omega\Delta t}{2})} - 1\right] \tag{35}$$

$$Y_e(\omega) = -\frac{\Delta t}{2L} + \frac{\Delta t}{2L}\frac{1}{j\tan(\frac{\omega\Delta t}{2})} \tag{36}$$

$$Y_e(\omega) = \frac{1}{R_e} + \frac{1}{j\omega L_e(\omega)} \tag{37}$$

$$Z_e(\omega) = \frac{1}{Y_e(\omega)} = \frac{1}{\frac{1}{R_e} + \frac{1}{j\omega L_e(\omega)}} \tag{38}$$

$$\boxed{Z_e(\omega) = R_e||j\omega L_e(\omega)} \tag{39}$$

$$\boxed{R_e = -\frac{2L}{\Delta t}} \tag{40}$$

$$\boxed{L_e(\omega) = L\frac{\tan(\frac{\omega\Delta t}{2})}{\frac{\omega\Delta t}{2}}} \tag{41}$$

### 3.4 Gear's Second Order Discretization

Finally, the frequency response of the Gear's second order discretization is the second of two equivalent circuits that must be derived by hand. Its frequency-domain circuit is shown in Figure 8. Instead of being in parallel, the equivalent frequency-domain resistance and inductance are now in series. This is a result of the approximation performing a differentiation instead of an integration, which yields a transfer function representing a frequency-dependent impedance directly instead of a frequency-dependent admittance. As a result, the real and imaginary components of the impedance can be separated and their addition represented as a $R_e(\omega)$ and $L_e(\omega)$ attached in series. The derivation will show that the equivalent resistance now has a frequency dependence. Since it is a real value, however, it is still considered a resistance.

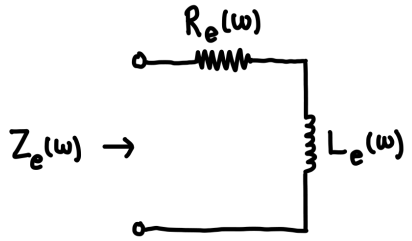


Figure 8: Frequency domain equivalent circuit of Gear's 2nd order discretization

The derivation of the frequency-dependent admittance $Y_e(\omega)$ is shown below:

$$v(t) = L\frac{di(t)}{dt} \tag{42}$$

$$di(t) \simeq \frac{3}{2}\left[i(t) - \frac{4}{3}i(t-\Delta t) + \frac{1}{3}i(t-2\Delta t)\right] \tag{43}$$

$$\frac{\Delta t}{L}v(t) = \frac{3}{2}\left[i(t) - \frac{4}{3}i(t-\Delta t) + \frac{1}{3}i(t-2\Delta t)\right] \tag{44}$$

$$input:\ i(t) = e^{j\omega t} \tag{45}$$

$$output:\ v(t) = Z_e(\omega)e^{j\omega t} \qquad (V = IZ) \tag{46}$$

$$\frac{\Delta t}{L}Z_e(\omega)e^{j\omega t} = \frac{3}{2}\left[e^{j\omega t} - \frac{4}{3}e^{j\omega(t-\Delta t)} + \frac{1}{3}e^{j\omega(t-2\Delta t)}\right] \tag{47}$$

$$Z_e(\omega) = \frac{3L}{2\Delta t}\frac{e^{j\omega t} - \frac{4}{3}e^{j\omega(t-\Delta t)} + \frac{1}{3}e^{j\omega(t-2\Delta t)}}{e^{j\omega t}} \tag{48}$$

$$Z_e(\omega) = \frac{3L}{2\Delta t}\left[1 - \frac{4}{3}e^{-j\omega\Delta t} + \frac{1}{3}e^{-j\omega 2\Delta t}\right] \quad = \frac{L}{2\Delta t}(e^{-j\omega\Delta t} - 1)(e^{-j\omega\Delta t} - 3)\ \texttt{hmmm...} \tag{49}$$

$$Y_e(\omega) = \frac{1}{Z_e(\omega)} \tag{50}$$

$$\boxed{Y_e(\omega) = \frac{2\Delta t}{L}\frac{1}{e^{-j\omega 2\Delta t} - 4e^{-j\omega\Delta t} + 3}} \tag{51}$$

With some algebraic manipulation and an application of Euler's identity, we can get $Y_e(\omega)$ into its impedance $Z_e(\omega)$ and subsequently its circuit equivalent components:

$$Z_e(\omega) = \frac{3L}{2\Delta t}\left[1 - \frac{4}{3}e^{-j\omega\Delta t} + \frac{1}{3}e^{-j\omega 2\Delta t}\right] \tag{52}$$

$$Z_e(\omega) = \frac{3L}{2\Delta t} - \frac{2L}{\Delta t}e^{-j\omega\Delta t} + \frac{L}{2\Delta t}e^{-j\omega 2\Delta t} \tag{53}$$

$$Z_e(\omega) = \frac{3L}{2\Delta t} - \frac{2L}{\Delta t}\left[\cos(-\omega\Delta t) + j\sin(-\omega\Delta t)\right] + \frac{L}{2\Delta t}\left[\cos(-\omega 2\Delta t) + j\sin(-\omega 2\Delta t)\right] \tag{54}$$

$$Z_e(\omega) = \frac{3L}{2\Delta t} - \frac{2L}{\Delta t}\cos(\omega\Delta t) + j\frac{2L}{\Delta t}\sin(\omega\Delta t) + \frac{L}{2\Delta t}\cos(\omega 2\Delta t) - j\frac{L}{2\Delta t}\sin(\omega 2\Delta t) \tag{55}$$

$$Z_e(\omega) = \frac{3L}{2\Delta t} - \frac{2L}{\Delta t}\cos(\omega\Delta t) + \frac{L}{2\Delta t}\cos(\omega 2\Delta t) + j\left[\frac{2L}{\Delta t}\sin(\omega\Delta t) - \frac{L}{2\Delta t}\sin(\omega 2\Delta t)\right] \tag{56}$$

$$\boxed{Z_e(\omega) = R_e(\omega) + j\omega L_e(\omega)} \tag{57}$$

$$\boxed{R_e(\omega) = \frac{L}{\Delta t}\left[\frac{3}{2} - 2\cos(\omega\Delta t) + \frac{1}{2}\cos(\omega 2\Delta t)\right]} \tag{58}$$

$$\boxed{L_e(\omega) = \frac{L}{\omega\Delta t}\left[2\sin(\omega\Delta t) - \frac{1}{2}\sin(\omega 2\Delta t)\right]} \tag{59}$$

Unfortunately, a network where a non-frequency dependent equivelent resistance (where it is constant across all frequencies) was not found despite some trial and error. As shown in Equation 49, we can factor the impedance into what looks like two zeros, but no solution was found further than this. Similarly, the admittance $Y_e(\omega)$ for Gear's second order cannot simplify due to the cosines having different magnitudes.

By making an educated guess, we can assume that two zeros in an impedance transfer function with no poles means that the impedance would become infinite as frequency becomes infinite and have two distinct slopes. This implies that there is an impedance looking into the equivalent circuit that would cause this effect.

## 3.5 Plotting the Magnitude and Phase Distortions

In this section, we take all of the admittances calculated in the previous section and compare them against the continuous solution of an inductance, which is $\frac{1}{j\omega L}$. By plotting the discretized admittance over the true admittance, we can analyze how well the discretization is able to imitate the true solution. This relationship is shown below:

$$\frac{H_e(\omega)}{H(\omega)} = \frac{Y_e(\omega)}{Y(\omega)}\bigg|_{L=1} \tag{60}$$

The frequency is swept from DC to the Nyquist frequency and the scale on the X-axis is converted into per-unit (p.u.). Since the Nyquist frequency is related to $\Delta t$ as $f_{Nyquist} = \frac{1}{2\Delta t}$ then the maximum value on the X-axis becomes 0.5. The resulting complex magnitude and angle are then plotted on two separate plots, yielding curves that represent distortion from the true answer for both quantities.

The true inductance $L$ cancels out as it exists in both numerator and denominator. However, substituting the value with, say, 1 H, is sufficient for this exercise as well. An arbitrary value of 1 $\mu s$ was chosen as $\Delta t$ and the Nyquist frequency was calculated from it.

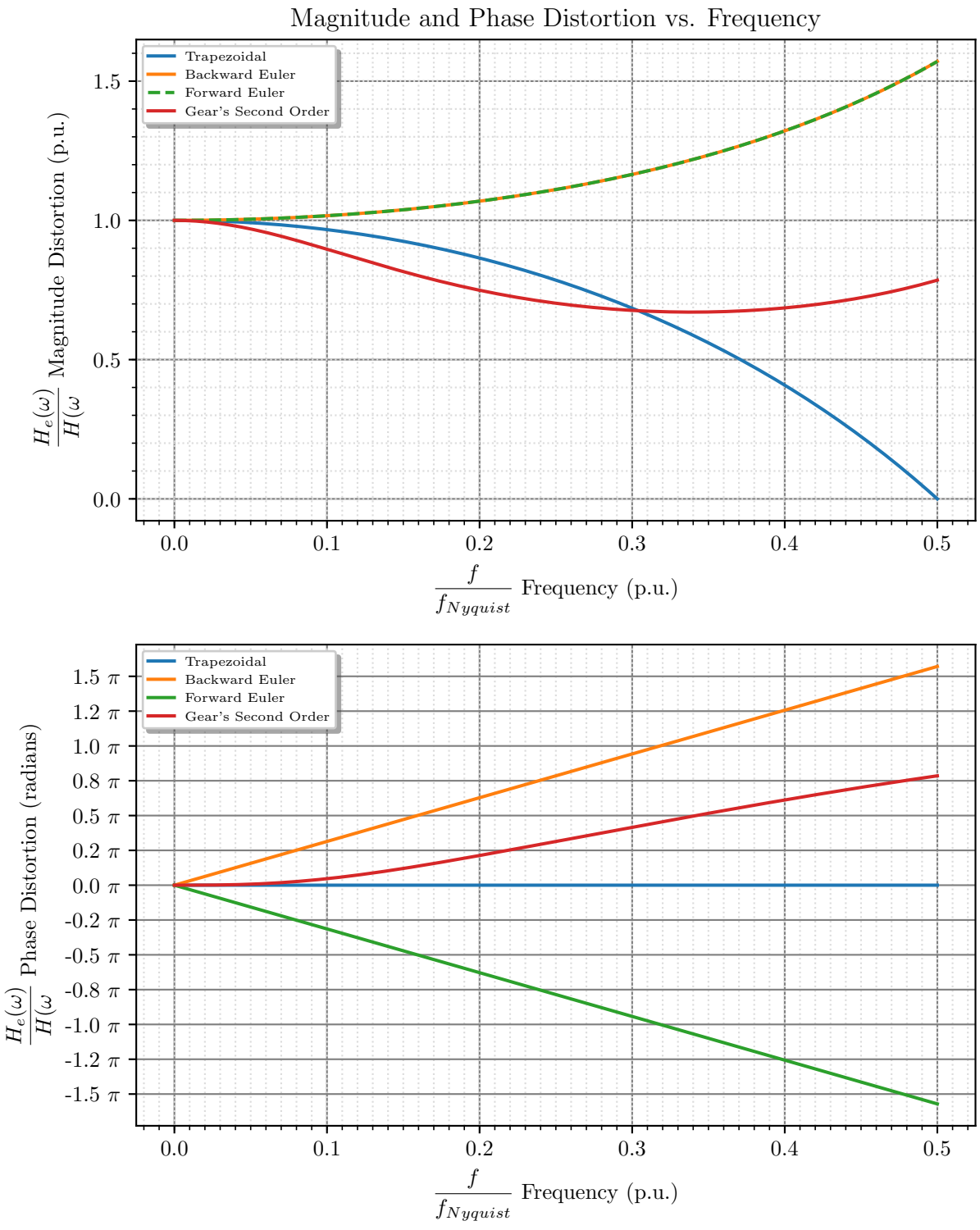


Figure 9: Discretization Magnitude Distortion and Phase Distortion vs. Frequency Plots

# 4 Conclusion

- For trapezoidal discretization, we note that the equivalent frequency-domain inductance value contains a $\tan(\theta)$ function. The plot for this trigonometric function contains asymptotic behaviours at every $\theta = \pi n$. However, we note that $\theta$ remains within 0 and 0.25 if the frequency sweep only goes up to the Nyquist frequency. Within these bounds, the resulting equivalent impedance value is relatively stable.

- For backward and forward Euler discretizations, we note that the equivalent frequency-domain inductance is identical and the equivalent frequency-domain resistance is the same constant, but with a different sign. They both yield a resistance and an inductance in parallel. At low frequencies, the impedance we see is a short and at high frequencies, the impedance we see is equivalent to $R_e$. The inductance $L_e(\omega)$ is identical to that of the trapezoidal discretization, and we've determined that for the frequency ranges we sweep for, its value is stable. For backward Euler, we only add in the denominator of the impedance. However, in forward Euler, it may be possible to obtain a value of 0 since the resistance is negative. This may imply that forward Euler is unstable when the real part of the reactance is approximately equal to the resistance as they will negate out to form a singularity.

- For Gear's second order discretization, we note that the impedance is equivalent to some series combination of $R_e(\omega)$ and $L_e(\omega)$. This alone implies a fairly stable system as there is no asymptotic behaviours or singularities to worry about. Both $R_e(\omega)$ and $L_e(\omega)$ are bounded functions as $\omega$ approaches infinity, which implies decent numerical stability. According to the plots, Gear's second order provides the best performance in terms of both magnitude and phase distortion.

- As noted earlier in the assignment, Gear's second order likely has a better solution than what is proposed in the assignment. Specifically, a value of resistance which does not change with frequency. It would be interesting to apply different frequency domain transformations, such as LaPlace, or different inputs, such as an impulse or a step function, to try to obtain the equivalent frequency domain circuit parameters. The impedance in factored form (Equation 49) seems like a step in the right direction.

- Another exercise that would have yielded interesting results (as well as confirmed the analyses), is generating Bode plots of the frequency-domain impedance we obtain from each of the different discretizations. This might help obtain an equivalent circuit parameter by observing the effects of the slopes in the resulting plot.

# 5 Code Listings and Data

## 5.1 Python Code Listing

The following is the code written in Python to generate the plots used in this report.

```
import argparse
import csv
import matplotlib
import matplotlib.ticker as tck
import matplotlib.pyplot as plt
import numpy as np
import control
import sympy

 # Matplotlib export settings
matplotlib.use('pgf')
import matplotlib.pyplot as plt
matplotlib.rcParams.update({
    'pgf.texsystem': 'pdflatex',
    'font.size': 10,
    'font.family': 'serif',  # use serif/main font for text elements
    'text.usetex': True,     # use inline math for ticks
    'pgf.rcfonts': False     # don't setup fonts from rc parameters
})

# inductance is cancelled out algebraically
# delta can be anything, as long as the nyquist frequency chosen is based on it
delta_t = 1e-6
nyquist_frequency = 1.0/(2.0*delta_t)

# Function that calculates the continuous, frequency domain impedance of an inductor
def z_real(omega):

    return 1j*omega

# Function that calculates the continuous, frequency domain admittance of an inductor
def y_real(omega):

    return 1.0/z_real(omega)

# Function that calculates the trapezoidal discretized, frequency domain admittance of an
    inductor
def y_trapezoidal(omega):

    return (delta_t/2.0)*(np.exp(1j*omega*delta_t) + 1)/(np.exp(1j*omega*delta_t) - 1)

# Function that calculates the backward euler discretized, frequency domain admittance of an
     inductor
def y_backward_euler(omega):

    return (delta_t)*(np.exp(1j*omega*delta_t))/(np.exp(1j*omega*delta_t) - 1)

# Function that calculates the forward euler discretized, frequency domain admittance of an
    inductor
def y_forward_euler(omega):

    return (delta_t)*(1/(np.exp(1j*omega*delta_t) - 1))

# Function that calculates the gear's second order discretized, frequency domain impedance
    of an inductor
def z_gears_second_order(omega):

    return (1/(2*delta_t))*(np.exp(-1j*omega*delta_t) - 1)*(np.exp(-1j*omega*delta_t) - 3)

# Main function
```

```
def main(args):

    # Create a list of frequencies to plot against
    frequencies = np.linspace(0.1, nyquist_frequency, num = 1000)

    # Get magnitudes
    # Create lists
    y_real_vals = []
    z_real_vals = []
    y_trapezoidal_vals = []
    y_backward_euler_vals = []
    y_forward_euler_vals = []
    z_gears_second_order_vals = []
    # Fill lists
    for frequency in frequencies:
        omega = 2*np.pi*frequency
        y_real_vals.append(y_real(omega))
        z_real_vals.append(z_real(omega))
        y_trapezoidal_vals.append(y_trapezoidal(omega))
        y_backward_euler_vals.append(y_backward_euler(omega))
        y_forward_euler_vals.append(y_forward_euler(omega))
        z_gears_second_order_vals.append(z_gears_second_order(omega))
    # Convert lists to numpy arrays
    y_real_vals = np.array(y_real_vals)
    z_real_vals = np.array(z_real_vals)
    y_trapezoidal_vals = np.array(y_trapezoidal_vals)
    y_backward_euler_vals = np.array(y_backward_euler_vals)
    y_forward_euler_vals = np.array(y_forward_euler_vals)
    z_gears_second_order_vals = np.array(z_gears_second_order_vals)

    # Plots for publication
    legend_font_size = 6

    # Plot Z(w) magnitude and phase
    fig, ax = plt.subplots(2)

    ax[0].plot(frequencies*delta_t, np.abs(y_trapezoidal_vals/y_real_vals), label='
        Trapezoidal')
    ax[0].plot(frequencies*delta_t, np.abs(y_backward_euler_vals/y_real_vals), label='
        Backward␣Euler')
    ax[0].plot(frequencies*delta_t, np.abs(y_forward_euler_vals/y_real_vals), linestyle='--'
        , label='Forward␣Euler')
    ax[0].plot(frequencies*delta_t, np.abs((1.0/z_gears_second_order_vals)/(1.0/z_real_vals)
        ), label='Gear\'s␣Second␣Order')
    ax[0].xaxis.set_minor_locator(tck.MultipleLocator(0.01))
    ax[0].yaxis.set_major_locator(tck.MultipleLocator(0.5))
    ax[0].yaxis.set_minor_locator(tck.MultipleLocator(0.1))
    ax[0].set(xlabel='$\\frac{f}{f_{Nyquist}}$␣Frequency␣(p.u.)', ylabel='$\\frac{H_e(\omega
        {})}{H(\omega{}}$␣Magnitude␣Distortion␣(p.u.)', title='Magnitude␣and␣Phase␣
        Distortion␣vs.␣Frequency')
    ax[0].grid(b=True, which='major', color='gray', linestyle='-')
    ax[0].grid(b=True, which='minor', color='gainsboro', linestyle='dotted')
    ax[0].legend(loc='best', prop={'size':legend_font_size}, fancybox=True, shadow=True)

    ax[1].plot(frequencies*delta_t, np.imag(y_trapezoidal_vals/y_real_vals), label='
        Trapezoidal')
    ax[1].plot(frequencies*delta_t, np.imag(y_backward_euler_vals/y_real_vals), label='
        Backward␣Euler')
    ax[1].plot(frequencies*delta_t, np.imag(y_forward_euler_vals/y_real_vals), label='
        Forward␣Euler')
    ax[1].plot(frequencies*delta_t, np.imag((1.0/z_gears_second_order_vals)/(1.0/z_real_vals
        )), label='Gear\'s␣Second␣Order')
    ax[1].xaxis.set_minor_locator(tck.MultipleLocator(0.01))
    ax[1].yaxis.set_major_formatter(tck.FormatStrFormatter('%1.1f␣$\pi$'))
    ax[1].yaxis.set_major_locator(tck.MultipleLocator(base=1/4))
    ax[1].set(xlabel='$\\frac{f}{f_{Nyquist}}$␣Frequency␣(p.u.)', ylabel='$\\frac{H_e(\omega
        {})}{H(\omega{}}$␣Phase␣Distortion␣(radians)')
    ax[1].grid(b=True, which='major', color='gray', linestyle='-')
```

```
    ax[1].grid(b=True, which='minor', color='gainsboro', linestyle='dotted')
    ax[1].legend(loc='best', prop={'size':legend_font_size}, fancybox=True, shadow=True)

    fig.set_size_inches(6.5,8)
    fig.tight_layout()
    fig.savefig('transfer_function_magnitude_and_phase_plot.pgf')
    fig.savefig('transfer_function_magnitude_and_phase_plot.png')

if __name__ == '__main__':
    # the following sets up the argument parser for the program
    parser = argparse.ArgumentParser(description='Assignment␣6␣solution␣generator')

    args = parser.parse_args()

    main(args)
```