ME 581 HW4

In [ ]:
```
import numpy as np # numerical library
import matplotlib.pyplot as plt # plotting library
%config InlineBackend.figure_format='retina' # high-res plots
import control.matlab as ctm # matlab layer for control systems library
import control as ct # use regular control library for a few things
ct.set_defaults('statesp', latex_repr_type='separate')
```

1a.

$$\begin{aligned} C_{PI} &= C_P + C_I \\ &= K_P + \frac{K_I T z}{z-1} \\ &= \frac{(K_P + K_I T)z - K_P}{z-1} \end{aligned}$$

$$C_D = \frac{K_D}{T}(1 - z^{-1}) = \frac{K_D z - K_D}{Tz}$$

1b.

In [ ]:
```
v0 = 25
Ka = 1599
tau_a = 0.5
M = 1670
B0 = 27.8
g = 9.806
KP = 0.6
KI = 0.01
KD = 0.08

T = 2
```

In [ ]:
```
CKa = ctm.tf2ss(Ka, [tau_a, 1], inputs = 'ubar', outputs = 'ft')
CM = ctm.tf2ss(1, [M, 0], inputs = 'f', outputs = 'vbar')
CB = ctm.tf2ss(B0, 1, inputs = 'vbar', outputs = 'b')
sum = ct.summing_junction(['ft', 'fd', '-b'], 'f')
plantcont = ct.interconnect([CKa, CM, CB, sum], inputs = ['ubar', 'fd'], outputs = ['vbar'])
plant_simulator = ctm.c2d(plantcont, T, 'zoh')

CPI = ctm.tf2ss([KP + KI*T, -KP], [1, -1], T, inputs = 'e', outputs = 'u')
CD = ctm.tf2ss([KD, -KD], [T, 0], T, inputs = 'vbar', outputs = 'd')
sum1 = ct.summing_junction(['vref', '-vbar'], 'e')
sum2 = ct.summing_junction(['u', '-d'], 'ubar')
sys = ct.interconnect([CPI, CD, sum1, sum2, plant_simulator], inputs = ['vref', 'fd'], outputs = [
display(sys)
```

$$A = \begin{pmatrix} 1 & 0 & 0 & -0.000599 \\ 0 & 0 & 0 & 0.000599 \\ 0.00982 & 0.0196 & 0.0183 & -0.000194 \\ 47.6 & 95.2 & 1.53 \cdot 10^{3} & 0.0265 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0.304 & 1.32 \cdot 10^{-18} \\ 1.48 \cdot 10^{3} & 1.97 \end{pmatrix}, \; dt$$

$$C = \begin{pmatrix} 0 & 0 & 0 & 0.000599 \\ 0.02 & 0.04 & 0 & -0.000395 \end{pmatrix} \quad D = \begin{pmatrix} 0 & 0 \\ 0.62 & 0 \end{pmatrix}$$

$$= 2$$

1c.

In [ ]:

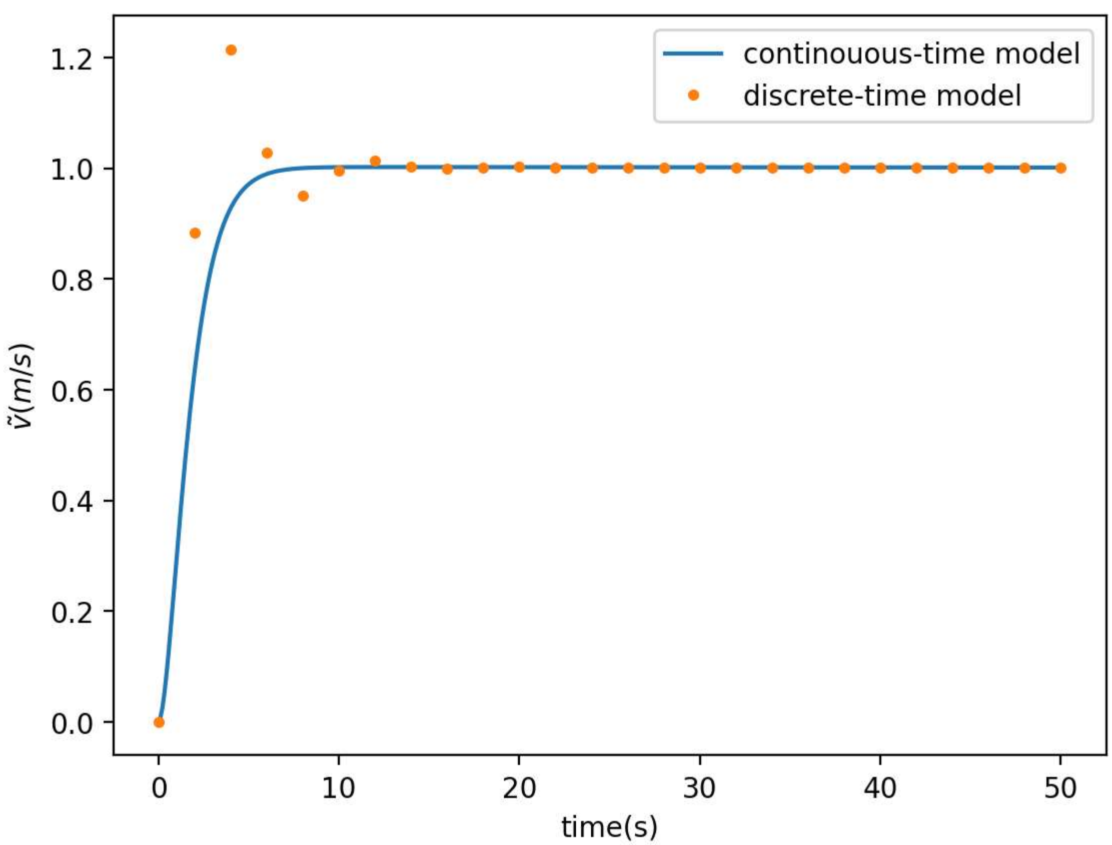
```
CPI = ctm.tf2ss([KP, KI], [1, 0], inputs = 'e', outputs = 'u')
CKa = ctm.tf2ss(Ka, [tau_a, 1], inputs = 'ubar', outputs = 'ft')
CM = ctm.tf2ss(1, M, inputs = 'f', outputs = 'a')
Cs = ctm.tf2ss(1, [1, 0], inputs = 'a', outputs = 'v')
CB = ctm.tf2ss(B0, 1, inputs = 'v', outputs = 'b')
CKd = ctm.tf2ss(KD, 1, inputs = 'a', outputs = 'd')
sum1 = ct.summing_junction(['vref', '-v'], 'e')
sum2 = ct.summing_junction(['u', '-d'], 'ubar')
sum3 = ct.summing_junction(['ft', 'fd', '-b'], 'f')
sysc = ct.interconnect([CPI, CKa, CM, Cs, CB, CKd, sum1, sum2, sum3], inplist = ['vref', 'fd'], ou
```

In [ ]:

```
y, t = ctm.step(sysc[0, 0], 50)
plt.plot(t, y, label='continouous-time model')
y, t = ctm.step(sys[0, 0], 50)
plt.plot(t, y, '.', label='discrete-time model')
plt.legend()
plt.ylabel(r'$\tilde{v} (m/s)$')
plt.xlabel('time(s)')
```

Out[ ]:

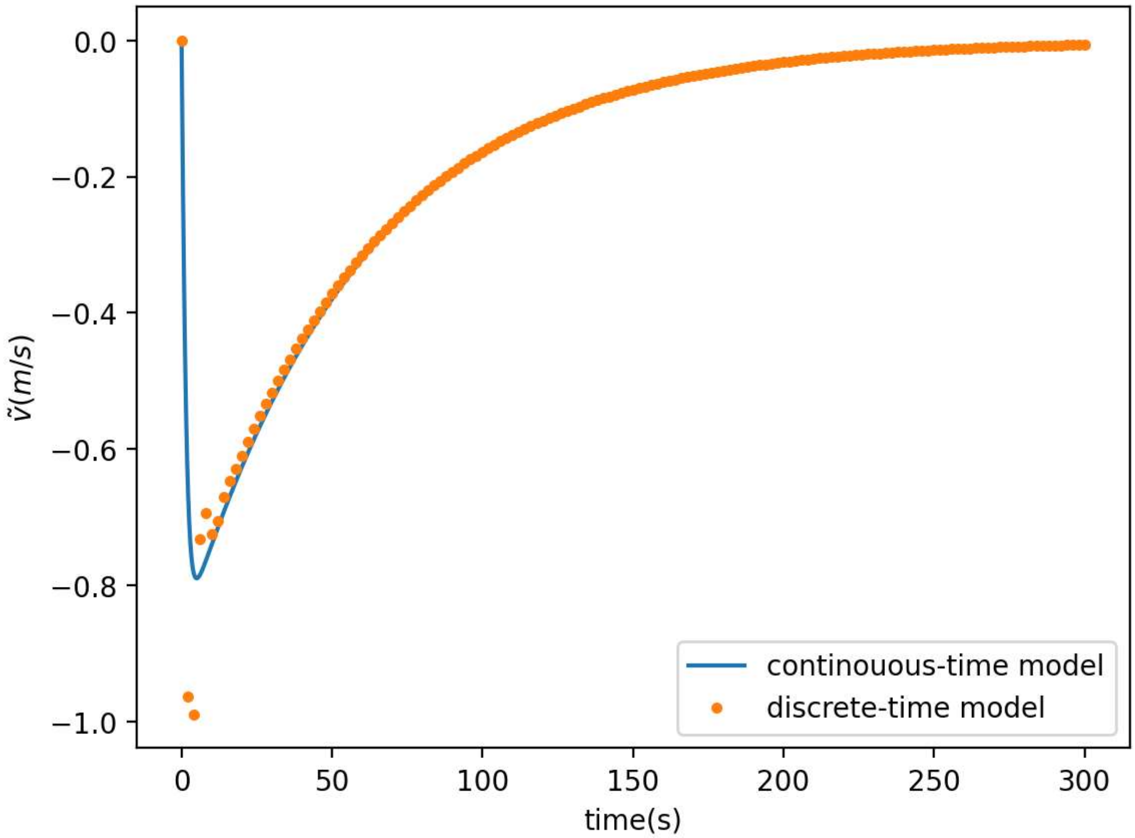
```
Text(0.5, 0, 'time(s)')
```



1d.

In [ ]:
```
scale = - M * g * np.sin(0.05)
y, t = ctm.step(sysc[0, 1], 300)
plt.plot(t, y*scale, label='continouous-time model')
y, t = ctm.step(sys[0, 1], 300)
plt.plot(t, y*scale, '.', label='discrete-time model')
plt.legend()
plt.ylabel(r'$\tilde{v} (m/s)$')
plt.xlabel('time(s)')
```

Out[ ]:
```
Text(0.5, 0, 'time(s)')
```



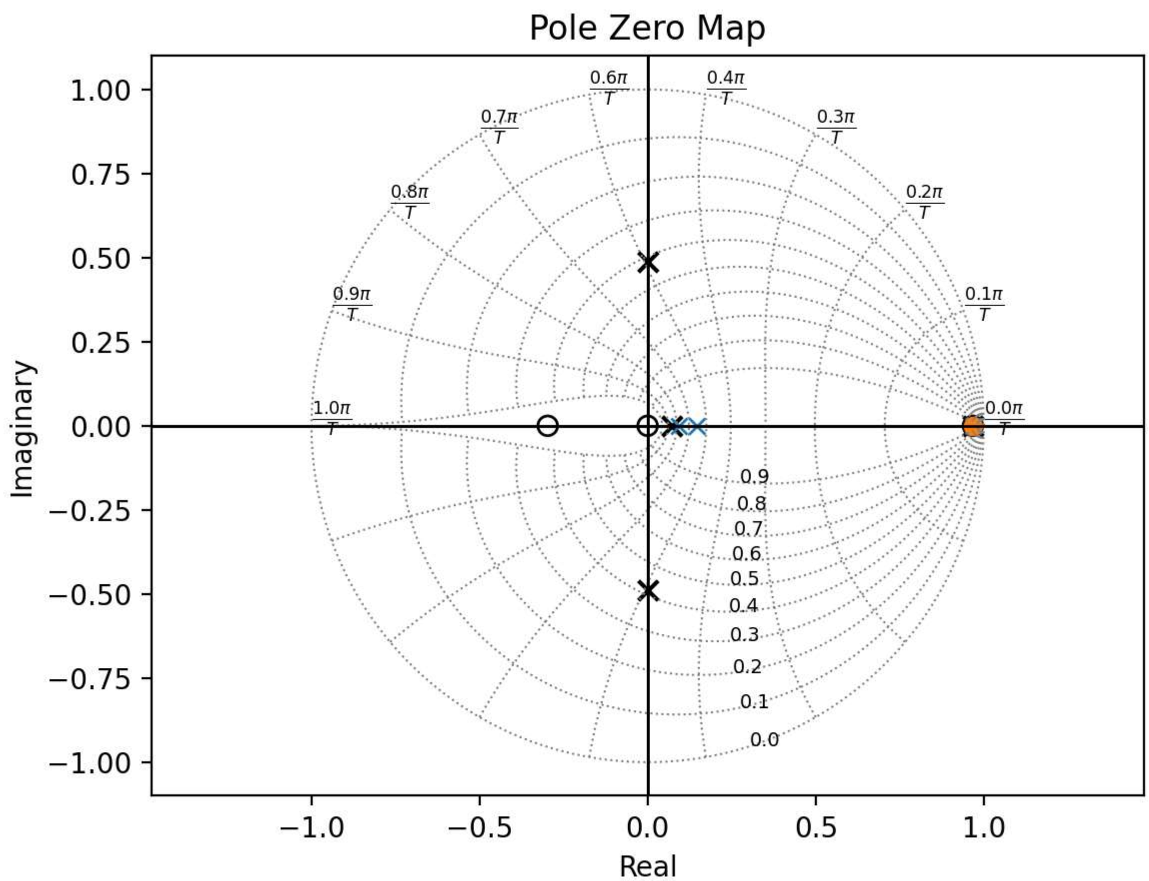
1e.

In [ ]:
```
[omegan,zeta,poles] = ct.damp(sys[0, 0])
pole = ct.pole(sysc[0, 0])
zero = ct.zero(sysc[0, 0])

p_e = np.exp(pole*T)
z_e = np.exp(zero*T)

OS = np.exp(-zeta*np.pi/np.sqrt(1-zeta**2)) * 100
print(zeta, OS)

ct.pzmap(sys[0, 0])
plt.plot(np.real(p_e), np.imag(p_e), 'x')
plt.plot(np.real(z_e), np.imag(z_e), 'o')
ct.grid.zgrid()
```

```
_____Eigenvalue______ Damping___ Frequency_
     0.9677                    1   -0.9677
   0.001904  +0.4886j    0.4158     0.8614
   0.001904  -0.4886j    0.4158     0.8614
    0.07336                    1  -0.07336
[1.         0.41575562 0.41575562 1.        ] [ 0.         23.78442227 23.78442227  0.        ]
```

```
C:\Users\YENPANG_HUANG\AppData\Local\Temp\ipykernel_19372\3149570422.py:8: RuntimeWarning: divide by zero encountered in divide
  OS = np.exp(-zeta*np.pi/np.sqrt(1-zeta**2)) * 100
```

Out[ ]:
```
(<Axes: title={'center': 'Pole Zero Map'}, xlabel='Real', ylabel='Imaginary'>,
 <Figure size 640x480 with 1 Axes>)
```



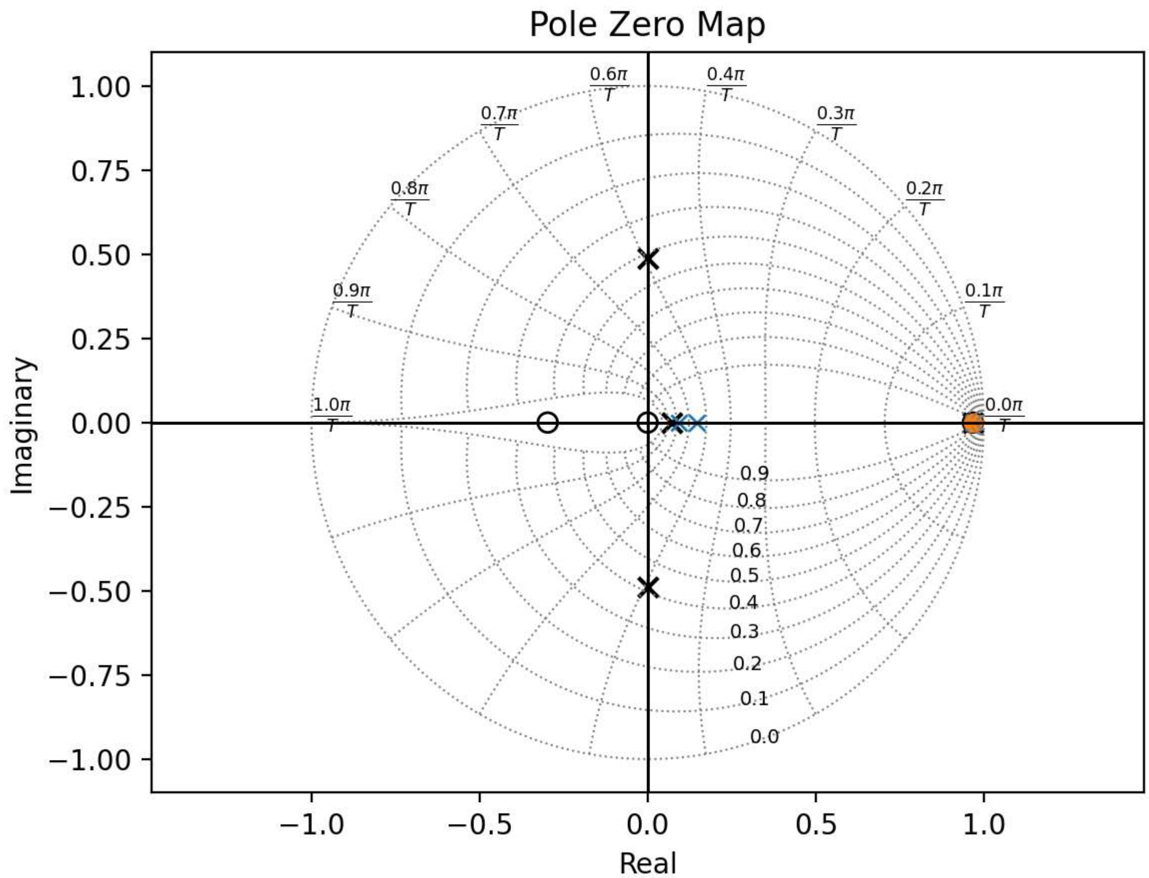
Based on the poles location on the plot, damping ratio (zeta) is approximately 0.42 and the percent overshoot is approximately 24%.

1f.

In [ ]:
```
tau = 1/(zeta*omegan)

print(tau)

ct.pzmap(sys[0, 0])
plt.plot(np.real(p_e), np.imag(p_e), 'x')
plt.plot(np.real(z_e), np.imag(z_e), 'o')
ct.grid.zgrid()
```

```
[60.88355533  2.79217102  2.79217102  0.76558489]
```

Out[ ]:
```
(<Axes: title={'center': 'Pole Zero Map'}, xlabel='Real', ylabel='Imaginary'>,
 <Figure size 640x480 with 1 Axes>)
```



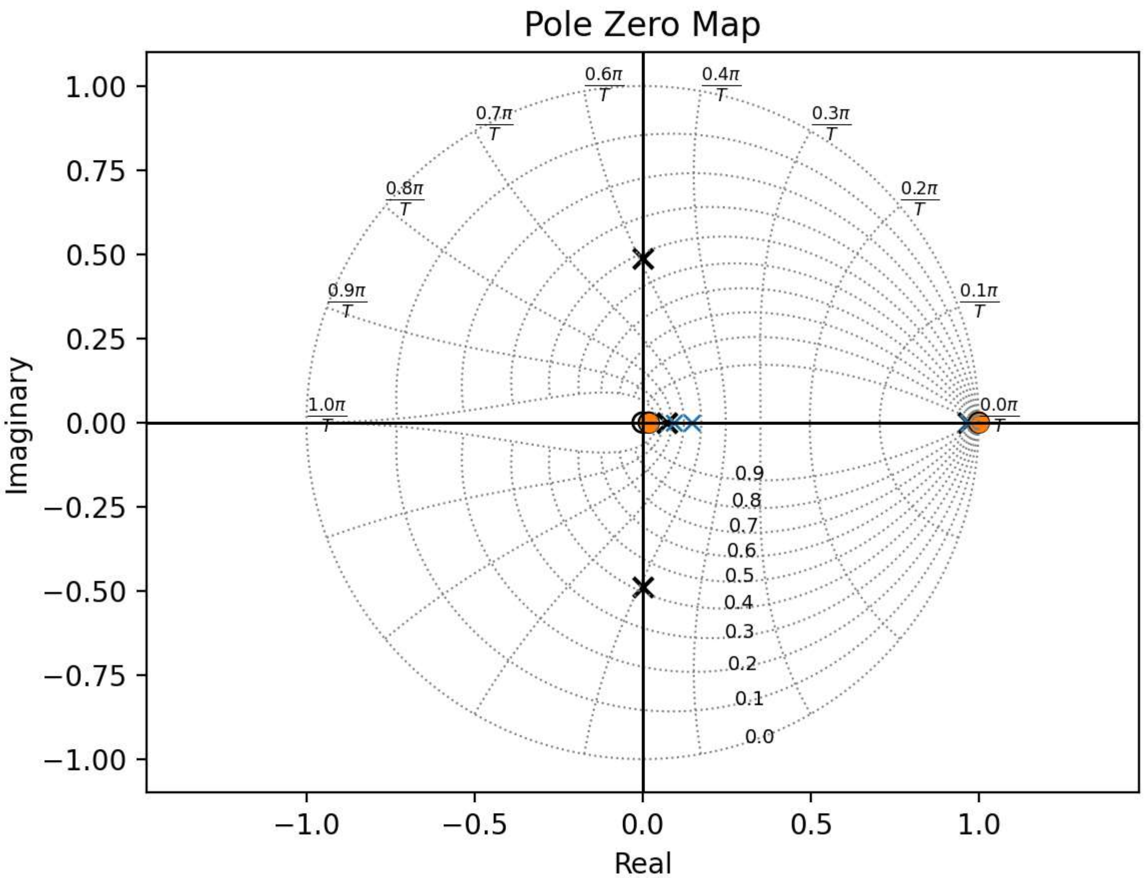
The time constant decays exponentially due to multiple poles. The time constant values showing above is also decaying.

1g.

In [ ]:
```
[omegan,zeta,poles] = ct.damp(sys[0, 1])

pole = ct.pole(sysc[0, 1])
zero = ct.zero(sysc[0, 1])

p_e = np.exp(pole*T)
z_e = np.exp(zero*T)

tau = 1/(zeta*omegan)

print(tau)

ct.pzmap(sys[0, 1])
plt.plot(np.real(p_e), np.imag(p_e), 'x')
plt.plot(np.real(z_e), np.imag(z_e), 'o')
ct.grid.zgrid()
```

```
_____Eigenvalue______ Damping___ Frequency_
    0.9677                    1    -0.9677
  0.001904    +0.4886j    0.4158     0.8614
  0.001904    -0.4886j    0.4158     0.8614
   0.07336                    1   -0.07336
[60.88355533  2.79217102  2.79217102  0.76558489]
```

Out[ ]:
```
(<Axes: title={'center': 'Pole Zero Map'}, xlabel='Real', ylabel='Imaginary'>,
 <Figure size 640x480 with 1 Axes>)
```



2a.

```
In [ ]: T = np.pi
        s = -0.3 + 1.2j
        z = np.exp(s*T)
        z
```

```
Out[ ]: (-0.3152424821841266-0.22903706994074025j)
```

For

$$s = \frac{ln(z)}{T} = -0.3 \pm 1.2i$$

$$z = e^{-0.3\pi \pm 1.2\pi i} = -0.3152 \pm 0.2290i$$

2b.

```
In [ ]: omegan = np.abs(np.log(z)/T)
        omegan
```

```
Out[ ]: 0.8544003745317532
```

$$\omega_n = \left|\frac{ln(z)}{\pi}\right| \approx 0.8544$$

2c.

```
In [ ]: zeta = -np.cos(np.angle(np.log(z)))
```

```
zeta
```

Out[ ]: 0.3511234415883916

$$\zeta = -cos[\measuredangle(ln\ z)] \approx 0.3512$$

2d.

In [ ]:
```
tau = 1/(zeta * omegan)
tau
```

Out[ ]: 3.333333333333334

$$\tau = \frac{1}{\zeta\omega_n} = \frac{1}{0.8544 * 1.2369} \approx 3.3333$$

2e.

In [ ]:
```
PO = np.exp(-np.pi*zeta/np.sqrt(1-zeta**2))*100
PO
```

Out[ ]: 30.786397132849913

$$PO = 100e^{-\pi\zeta/\sqrt{1-\zeta^2}} \approx 30.78$$

3a.

$$G(z) = \frac{1}{z+0.5}\ , \quad dt = 0.1$$

$$Pole : -0.5$$

Hence, the system is stable since the pole is within the unit cycle.

3b.

In [ ]:
```
w = 20
dt = 0.1
z = np.exp(1j*w*dt)
def G(z):
    return 1 / (z + 0.5)

abs(G(z))
```

Out[ ]: 1.095103607556263

In [ ]: