MAE 3134: Homework 7

Due date: TBD, 0935

Problem 1 For each of the systems identified below, copnmute the magnitude and angle of the transfer function when evaluated at the specified points in the imaginary plane (s-plane). You should use algebra to transform each function into a general complex number, then evaluate at each desired point. You must show your work for full credit. The magnitude should be reported in decibels (dB) and the angle in degrees.

1. Accelerometer model:

$$G(s) = \frac{X(s)}{F(s)} = \frac{0.5}{s^2 + 2s + 10}$$

   evaluated at the following points:

   (a) $s = j2$
   (b) $s = j3.1623$
   (c) $s = j2.8284$

2. Low-pass filter:

$$G(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{5}{s + 6}$$

   evaluated at the following points:

   (a) $s = j0.6$
   (b) $s = j6$
   (c) $s = j60$

3. High-pass filter:

$$G(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{s}{s + 35}$$

   evaluated at the following points:

   (a) $s = j2$
   (b) $s = j35$
   (c) $s = j500$

4. Lead filter:

$$G(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{0.21(s + 2)}{s + 3.05}$$

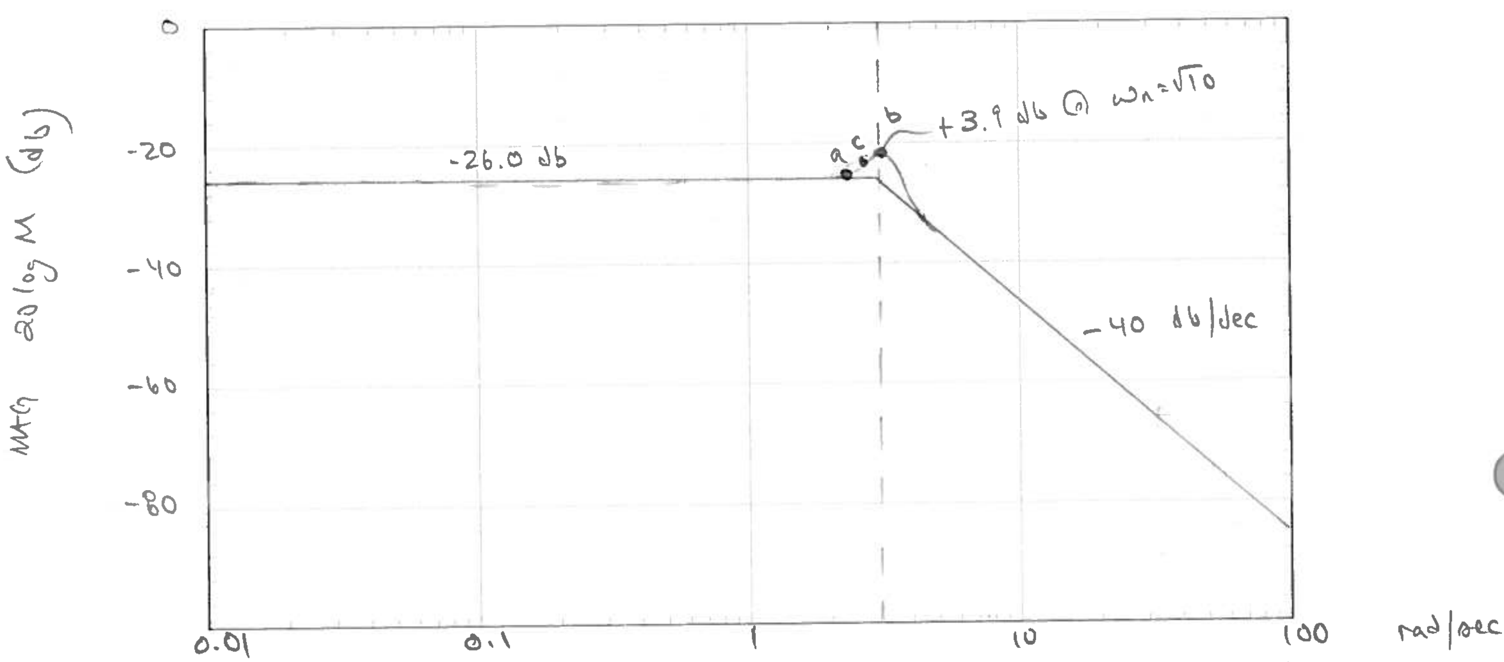
   evaluated at the following points:

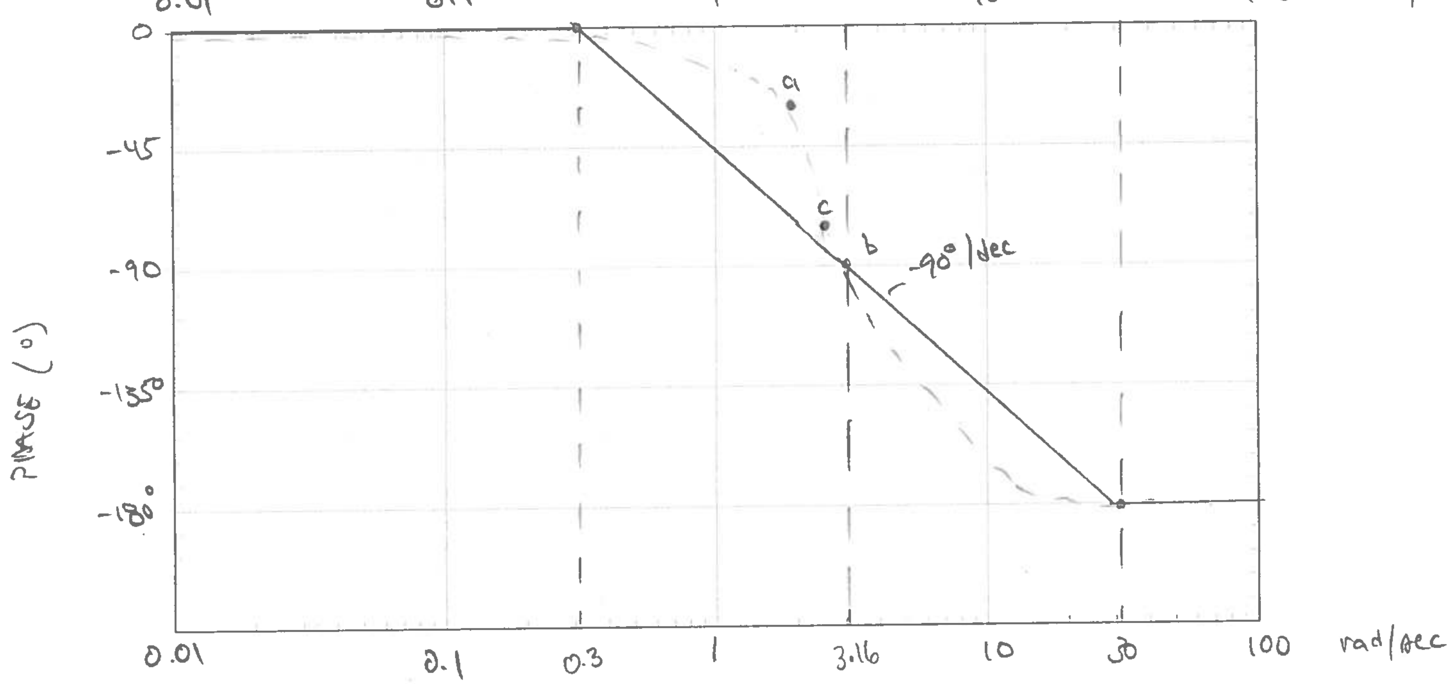
   (a) $s = j0.247$
   (b) $s = j2.47$
   (c) $s = j24.7$

You should produce HIGH quality Bode plots using the approximation tools learned in class. This means that you should use a ruler and write neatly and clearly.

Problem 2 Using the tranfer function of the **accelerometer model** given in Problem 1:

1. Draw an asymptotic Bode plot on the graphs provided.
2. Plot the true magnitude and phase of the specific points given previously.
3. Sketch your estimate of the actual Bode plot.

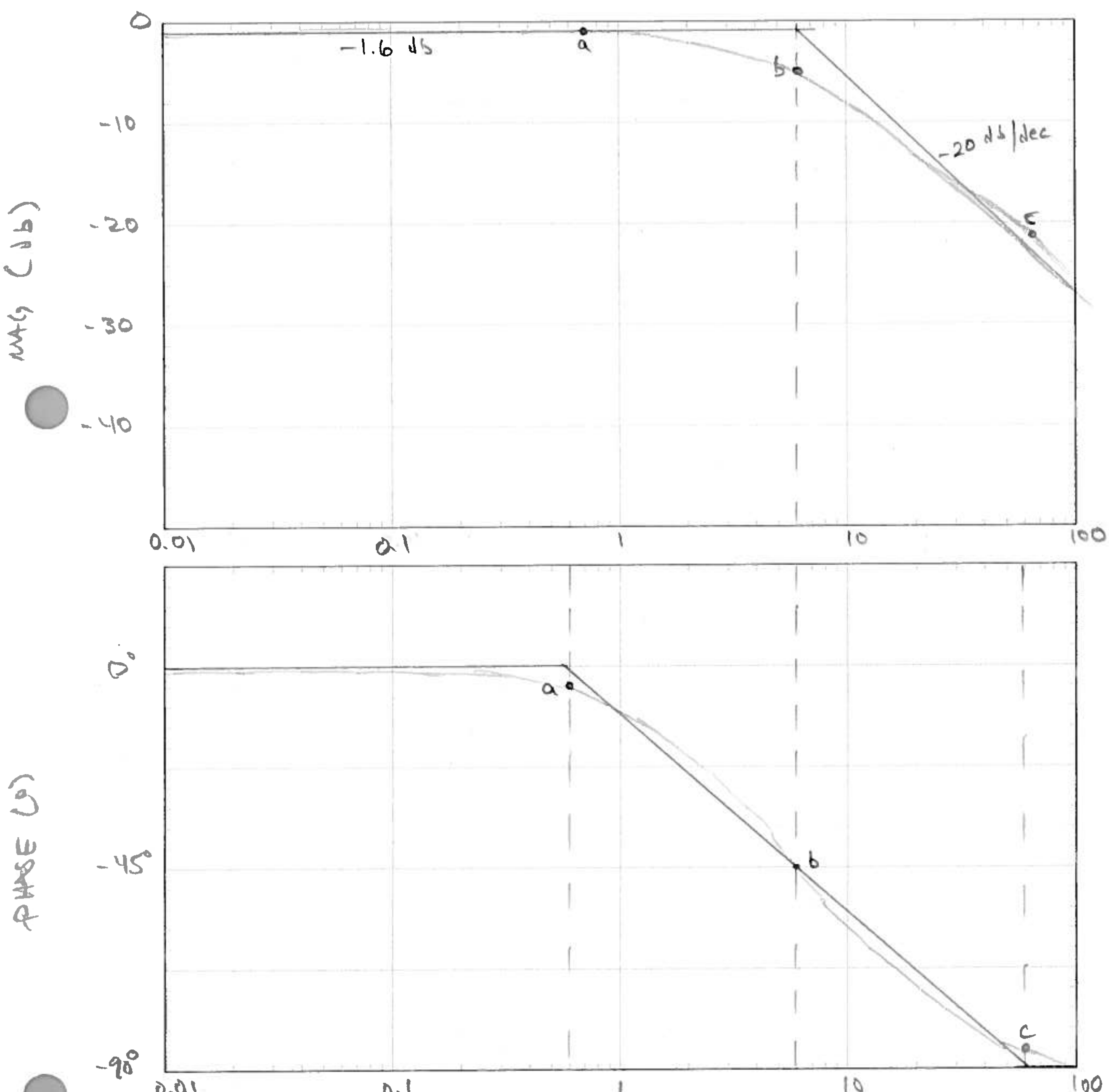




You should produce HIGH quality Bode plots using the approximation tools learned in class. This means that you should use a ruler and write neatly and clearly.

Problem 3 Using the tranfer function of the **low-pass filter** given in Problem 1:

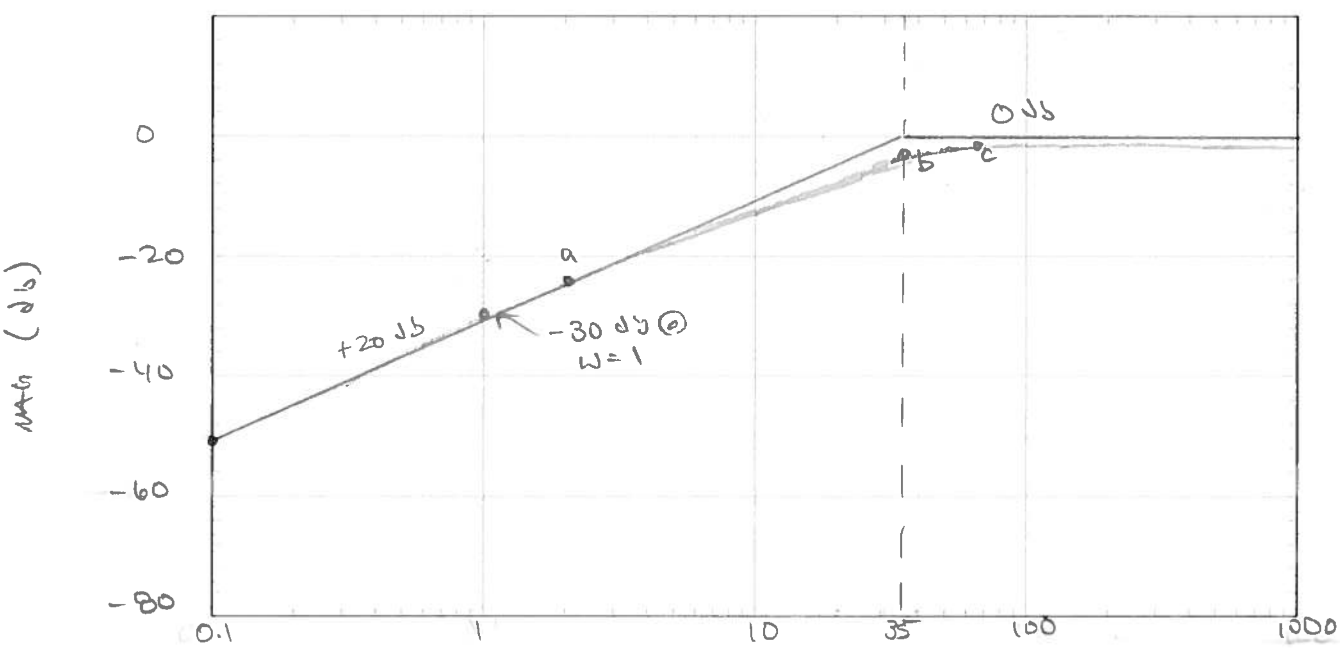
1. Draw an asymptotic Bode plot on the graphs provided.
2. Plot the true magnitude and phase of the specific points given previously.
3. Sketch your estimate of the actual Bode plot.

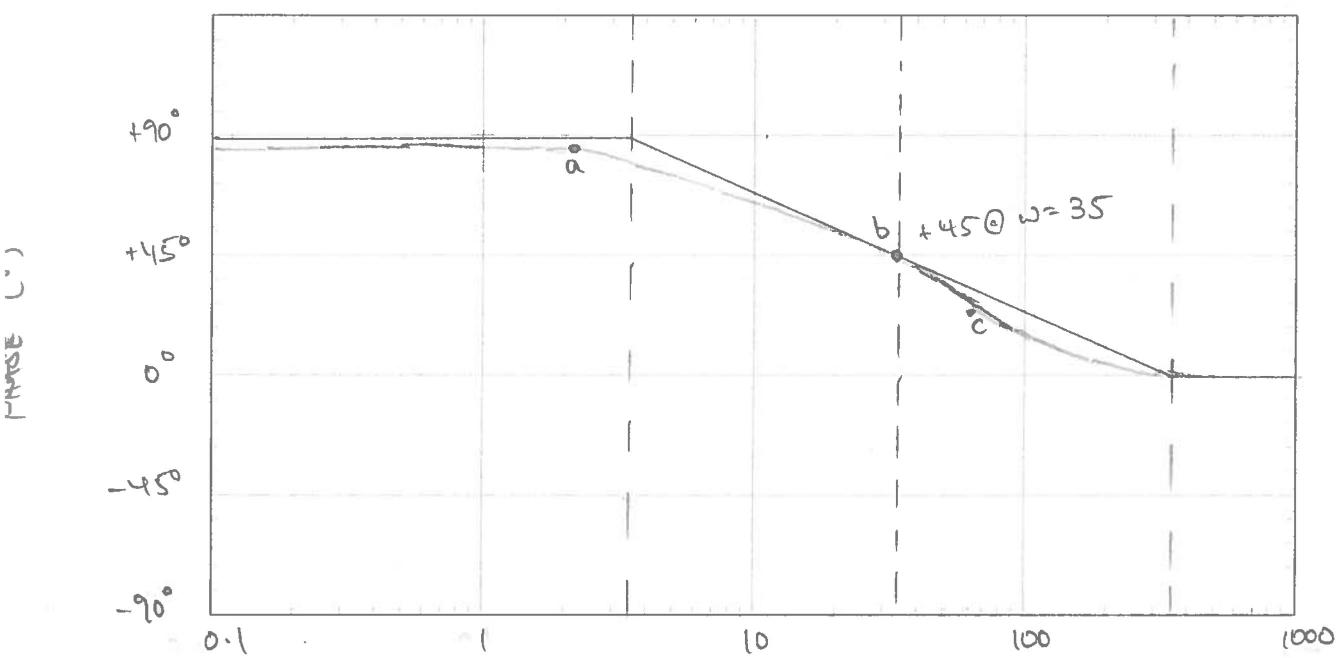


You should produce HIGH quality Bode plots using the approximation tools learned in class. This means that you should use a ruler and write neatly and clearly.

Problem 4 Using the tranfer function of the **high-pass filter** given in Problem 1:

1. Draw an asymptotic Bode plot on the graphs provided.
2. Plot the true magnitude and phase of the specific points given previously.
3. Sketch your estimate of the actual Bode plot.

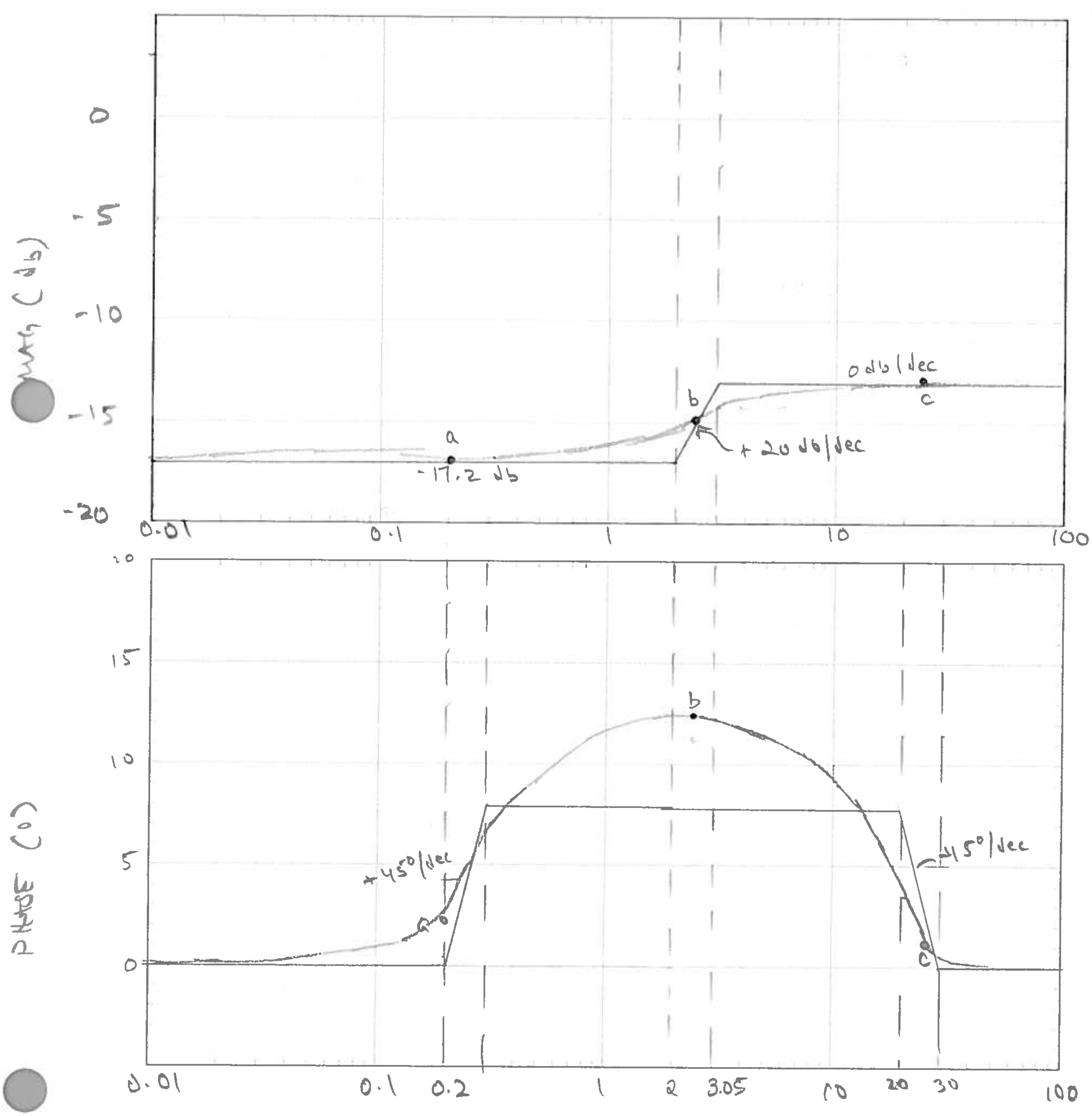




You should produce HIGH quality Bode plots using the approximation tools learned in class. This means that you should use a ruler and write neatly and clearly.

Problem 5 Using the tranfer function of the **lead ~~pass~~ filter** given in Problem 1:

1. Draw an asymptotic Bode plot on the graphs provided.
2. Plot the true magnitude and phase of the specific points given previously.
3. Sketch your estimate of the actual Bode plot.



Problem 6 Answer the following questions:

1. The model for a typical spring-mass-damper accelerometer is:

$$G(s) = \frac{X(s)}{F(s)} = \frac{0.5}{s^2 + 2s + 10}.$$

If the input to this system is

$$f(t) = 17.333 \sin 2t,$$

what is the steady-state output of the system?

2. We can model a low-pass filter as

$$G(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{5}{s + 6}.$$

For the following inputs, determine the steady-state output of the system:

(a) $v(t) = 10 \sin 0.6t$

(b) $v(t) = 10 \sin 60t$

WHY IS THIS A LOW PASS FILTER?

3. Now let's look at the high-pass filter modeled as

$$G(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{s}{s + 35}.$$

Determine the steady-state output corresponding to these inputs:

(a) $v(t) = 10 \sin 2t$

(b) $v(t) = 10 \sin 500t$

WHY IS THIS A HIGH PASS FILTER

HOMEWORK 7

PROBLEM 1

1. $G(s) = \dfrac{0.5}{s^2+2s+10}$ $G(j\omega) = \dfrac{0.5}{(j\omega)^2+2j\omega+10} = \dfrac{0.5}{-\omega^2+2j\omega+10}$

$$G(j\omega) = \frac{0.5}{(10-\omega^2)+2\omega j} \cdot \frac{(10-\omega^2)-2\omega j}{(10-\omega^2)-2\omega j}$$

$$= \frac{0.5(10-\omega^2-2\omega j)}{(10-\omega^2)^2 - \cancel{2\omega j(10-\omega^2)} + \cancel{2\omega j(10-\omega^2)} - 2^2\omega^2 j^2}$$

$$= \frac{0.5(10-\omega^2-2\omega j)}{(10-\omega^2)+4\omega^2} = \frac{1/2(10-\omega^2)-\omega j}{(10-\omega^2)^2+4\omega^2}$$ $(10-\omega^2) = 100-20\omega^2+\omega^4$

$$G(j\omega) = \frac{1/2(10-\omega^2)}{\omega^4-16\omega^2+100} - \frac{\omega j}{\omega^4-16\omega^2+100}$$

@ $s=2j$ $G(j\omega) = 0.057692 - 0.038462j$

$= 0.069338 \angle -33.6905^\circ$

$= -23.1806\ db \angle -33.6905^\circ$

@ $s = 3.1623j$ $G(j\omega) = -0.000002 - 0.079056j$

$= 0.079056 \angle -90.00\ deg$

$= -22.04\ db \angle -90^\circ$

@ $s = 2.8284j$ $G(j\omega) = 0.02778 - 0.078567j$

$= 0.0833 \angle -70.527^\circ$

$= -21.5\ db \angle -70.52^\circ$

PROBLEM 1

2. $G(s) = \frac{5}{s+6}$ $G(j\omega) = \frac{5}{j\omega+6} \cdot \frac{6-j\omega}{6-j\omega)}$

$$G(j\omega) = \frac{30 - 5\omega j}{36 + 6\omega j - 6\omega j - j^2\omega^2} = \frac{30 - 5\omega j}{36 + \omega^2}$$

$$G(j\omega) = \frac{30}{36+\omega^2} - \frac{5\omega}{36+\omega^2} j$$

(a) $s = 0.6j$ $G(j\omega) = 0.825083 - 0.082508j$
$= 0.829178 \angle -5.710°$
$= -1.62$ db $\angle -5.7°$

(c) $s = 6j$ $G(j\omega) = 0.41 - 0.41j$
$= 0.58 \angle -45°$
$= -4.5$ db $\angle -45°$

(e) $s = 60j$ $G(j\omega) = 0.0082 - 0.0825j$
$= 0.082 \angle -84.3°$
$= -21.6$ db $\angle -84.3°$

3. $G(s) = \frac{s}{s+35}$ $G(j\omega) = \frac{j\omega}{j\omega+35} \cdot \frac{35-j\omega}{35-j\omega}$

$$G(j\omega) = \frac{35j\omega - j^2\omega^2}{35^2 - j^2\omega^2} = \frac{\omega^2 + 35\omega j}{\omega^2 + 35^2} = \frac{\omega^2}{\omega^2+35^2} + \frac{35\omega}{\omega^2+35^2} j$$

(a) $s = 2j$ $G(j\omega) = 0.00325 + 0.057j$
$= 0.057 \angle 86.73°$
$= -24.8$ db $\angle 86.73°$

(b) $s = 35j$ $G(j\omega) = 0.5 + 0.5j$
$= 0.707 \angle 45°$
$= -3$ db $\angle 45°$

(c) $s = 500j$ $G(j\omega) = 0.995124 + 0.0696j$
$= 0.99 \angle 4.00°$
$= -0.021$ db $\angle 4.0°$

PROBLEM 1

4. $G(s) = \dfrac{0.21(s+2)}{s+3.05}$ $G(j\omega) = \dfrac{0.21(j\omega+2)}{j\omega+3.05}$

$$G(j\omega) = \frac{0.21(2+j\omega)}{3.05+j\omega} \cdot \left(\frac{3.05-j\omega}{3.05+j\omega}\right) = \frac{0.21(2+j\omega)(3.05-j\omega)}{(3.05+j\omega)(3.05-j\omega)}$$

$$= \frac{0.21(6.10-2j\omega+3.05j\omega-j^2\omega^2)}{3.05^2-j^2\omega^2} = \frac{0.21(6.1+\omega^2+1.05j\omega)}{3.05^2+\omega^2}$$

$$G(j\omega) = \frac{0.21(6.1+\omega^2)}{3.05^2+\omega^2} + \frac{0.21(1.05\,\omega)}{3.05^2+\omega^2}\,j$$

(a) $s = 0.247j$ $G(j\omega) = 0.13 + 0.0058j$
$= 0.13 \angle 2.41^\circ$
$= -17 \text{ db} \angle 2.41^\circ$

(b) $s = 2.47j$ $G(j\omega) = 0.166 + 0.035j$
$= 0.17 \angle 12.0^\circ$
$= -15.3 \text{ db} \angle 12.0^\circ$

(c) $s = 24.7j$ $G(j\omega) = 0.21 + 0.008793j$
$= 0.209 \angle 2.41^\circ$
$= -13.6 \text{ db} \angle 12.41^\circ$

PROBLEM 2

$$G(s) = \frac{0.5}{s^2 + 2s + 10} \qquad G(j\omega) = \frac{0.5}{10}\,\frac{1}{\left(\frac{s^2}{10} + \frac{2}{10}s + 1\right)}$$

$\omega_n = \sqrt{10} = 3.16$ rad/sec

CONSTANT $\frac{0.5}{10} \rightarrow M = -26.02$ db 0° PHASE

MAG – 0 db UNTIL ω_n THEN -40 db/dec.

$M_{\omega_n} = -20 \log 2\zeta$ @ ω_n

PHASE – $0^\circ \rightarrow -180^\circ$ WITH -90° @ ω_n

$2\zeta\omega_n = 2 \qquad \omega_n = \sqrt{10} \Longrightarrow \zeta = \frac{2}{2\sqrt{10}} = 0.316$

$M_{\omega_n} = +3.97$ db @ $\omega_n = 3.16$

PROBLEM 3

$$G(s) = \frac{5}{s+6} \qquad G(j\omega) = \frac{5}{6}\,\frac{1}{s/6 + 1}$$

$\omega_n = 6$ rad/sec

CONSTANT $M = +20 \log \frac{5}{6} = -1.58$ db WITH 0° PHASE

POLES $s/6+1$ – 0 db UNTIL $\omega = 6$ THEN -20 db/dec.

$0^\circ \rightarrow -90^\circ$ WITH -45° @ $\omega = 6$

PROBLEM 4

$G(s) = \frac{s}{s+35}$ $\qquad G(j\omega) = \frac{1}{35} \frac{s}{\frac{s}{35}+1}$

$\omega_n = 35$ rad/sec

CONSTANT - $+20\log 1/35 = -30.0$ db

POLES 0 db UNTIL $\omega = 35$ THEN -20 db/dec

PHASE $0^\circ \rightarrow -90^\circ$ WITH -45° @ $\omega = 35$

FREE "s" $+20$ db/dec WITH 0 db @ $\omega = 1$

$+90^\circ$ PHASE SHIFT

PROBLEM 5

$G(s) = \frac{0.21(s+2)}{s+3.05}$ $\qquad G(s) = \frac{0.21(2)}{3.05}\left(\frac{s/2+1}{s/3.05+1}\right)$

MAGNITUDE $+20\log \frac{0.21(2)}{3.05} = -17.22$ db

ZERO 0 db UNTIL $\omega = 2$ THEN $+20$ db/dec

$0 \rightarrow +90^\circ$ WITH $+45^\circ$ @ $\omega = 2$ $\quad 0.2 \rightarrow 20$ rad/sec

POLES 0 db UNTIL $\omega = 3.05$ THEN -20 db/dec

$0 \rightarrow -90^\circ$ WITH -45° @ $\omega = 3.05$ $\quad 0.305 \rightarrow 30.5$ RAD/sec

PROBLEM 6

1. $G(s) = \dfrac{0.5}{s^2+2s+10}$ $\qquad G(2j) = 0.69 \angle -33^\circ$

$\cdot\ x(t) = 0.69 \cdot 17.333 \sin(2t - 33^\circ)$ FROM $f(t) = 17.333 \sin 2t$

$= 1.2 \sin(2t - 33^\circ)$

2. $G(s) = \dfrac{5}{s+6}$

$v(t) = 10 \sin 0.6t \rightarrow x(t) = 8.291 \sin(0.6t - 5.7^\circ)$

$v(t) = 10 \sin 60t \rightarrow x(t) = 0.829 \sin(60t - 84^\circ)$

3. $G(s) = \dfrac{s}{s+35}$

$v(t) = 10 \sin 2t \longrightarrow x(t) = 0.57 \sin(2t + 86.7^\circ)$

$v(t) = 10 \sin 500t \longrightarrow x(t) = 8.6 \sin(500t + 30^\circ)$

$G(s) = \dfrac{5}{s+6} \longrightarrow$ ATTUNATES HIGH FREQ.

$|G(j\omega)| \rightarrow -\infty$ AS $\omega \rightarrow \infty$

ALLOWS LOW FREQ.

$G(s) = \dfrac{s}{s+35} \longrightarrow$ ATTENUATES LOW FREQ

$|G(j\omega)| \rightarrow -\infty$ AS $\omega \rightarrow 0$

ALLOWS HIGH FREQ.

In [7]:
```
%matplotlib notebook
import numpy as np
import sympy
from scipy import signal
import matplotlib as mpl
import matplotlib.pyplot as plt

figsize = (8,4)

def magdb(mag):
    return 20*np.log10(mag)

def phasedeg(phase):
    return phase*180/np.pi
```

Problem 1

We're looking for students to show the algebra required to transform the transfer function into the equivalent frequency response function. Then they should do the algebra to transform it into a more easily accessible form, ie. a complex number with real and imaginary parts.

Part 1

$$G(s) = \frac{X(s)}{F(s)} = \frac{0.5}{s^2 + 2s + 10}$$

The equivalent frequency response function is

$$G(j\omega) = \frac{\frac{1}{2}(10 - \omega^2)}{\omega^4 - 16\omega^2 + 100} - j\frac{\omega}{\omega^4 - 16\omega^2 + 100}$$

We can now evaluate these at the desired points

In [8]:
```
s, w = sympy.symbols('s w')
G1s = 0.5/(s**2+2*s+10)
G1jw = 0.5*(10-w**2)/(w**4 - 16*w**2 + 100) - (w)/(w**4 - 16*w**2 +10
0)*sympy.I

# make sure my derivation is correct
np.testing.assert_equal(G1s.subs([(s,2*sympy.I)]).evalf(),
G1jw.subs([(w, 2)]).evalf())
np.testing.assert_almost_equal(G1s.subs([(s,3.1623*sympy.I)]).evalf(),
 G1jw.subs([(w, 3.1623)]).evalf())
np.testing.assert_almost_equal(G1s.subs([(s,2.8284*sympy.I)]).evalf(),
 G1jw.subs([(w, 2.8284)]).evalf())

# output correct answers in multiple forms
G1a = complex(G1jw.subs([(w, 2)]).evalf())
print("Part 1a:    G(2j) = %05f %+05fj" % (np.real(G1a),
np.imag(G1a)))
print("                  = %05f < %05f deg" % (np.linalg.norm(G1a),ph
asedeg(np.angle(G1a))))
print("                  = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G1a)),phasedeg(np.angle(G1a))))

G1b = complex(G1jw.subs([(w, 3.1623)]).evalf())
print("Part 1b:    G(3.1623j) = %05f %+05fj" % (np.real(G1b),
np.imag(G1b)))
print("                  = %05f < %05f deg" % (np.linalg.norm(G1b),ph
asedeg(np.angle(G1b))))
print("                  = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G1b)),phasedeg(np.angle(G1b))))

G1c = complex(G1jw.subs([(w, 2.8284)]).evalf())
print("Part 1c:    G(2.8284j) = %05f %+05fj" % (np.real(G1c),
np.imag(G1c)))
print("                  = %05f < %05f deg" % (np.linalg.norm(G1c),ph
asedeg(np.angle(G1c))))
print("                  = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G1c)),phasedeg(np.angle(G1c))))
```

```
Part 1a:    G(2j) = 0.057692 -0.038462j
                  = 0.069338 < -33.690068 deg
                  = -23.180633 db < -33.690068 deg
Part 1b:    G(3.1623j) = -0.000002 -0.079056j
                  = 0.079056 < -90.001280 deg
                  = -22.041261 db < -90.001280 deg
Part 1c:    G(2.8284j) = 0.027780 -0.078567j
                  = 0.083333 < -70.527225 deg
                  = -21.583625 db < -70.527225 deg
```

Part 2

$$G(s) = \frac{X(s)}{F(s)} = \frac{5}{s+6}$$

The equivalent frequency response function is

$$G(j\omega) = \frac{30}{36+\omega^2} - j\frac{5\omega}{36+\omega^2}$$

We can now evaluate these at the desired points

In [9]:
```
G2s = 5/(s+ 6)
G2jw = 30/(36 + w**2) - (5*w)/(36 + w**2)*sympy.I

# make sure my derivation is correct
np.testing.assert_equal(G2s.subs([(s,0.6*sympy.I)]).evalf(), G2jw.sub
s([(w, 0.6)]).evalf())
np.testing.assert_almost_equal(G2s.subs([(s,6*sympy.I)]).evalf(), G2j
w.subs([(w, 6)]).evalf())
np.testing.assert_almost_equal(G2s.subs([(s,60*sympy.I)]).evalf(), G2
jw.subs([(w, 60)]).evalf())

# output correct answers in multiple forms
G2a = complex(G2jw.subs([(w, 0.6)]).evalf())
print("Part 2a:    G(0.6j) = %05f %+05fj" % (np.real(G2a), np.imag(G2
a)))
print("                    = %05f < %05f deg" % (np.linalg.norm(G2a),ph
asedeg(np.angle(G2a))))
print("                    = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G2a)),phasedeg(np.angle(G2a))))

G2b = complex(G2jw.subs([(w, 6)]).evalf())
print("Part 2b:    G(6j) = %05f %+05fj" % (np.real(G2b),
np.imag(G2b)))
print("                    = %05f < %05f deg" % (np.linalg.norm(G2b),ph
asedeg(np.angle(G2b))))
print("                    = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G2b)),phasedeg(np.angle(G2b))))

G2c = complex(G2jw.subs([(w, 60)]).evalf())
print("Part 2c:    G(60j) = %05f %+05fj" % (np.real(G2c),
np.imag(G2c)))
print("                    = %05f < %05f deg" % (np.linalg.norm(G2c),ph
asedeg(np.angle(G2c))))
print("                    = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G2c)),phasedeg(np.angle(G2c))))
```

```
Part 2a:    G(0.6j) = 0.825083 -0.082508j
                  = 0.829198 < -5.710593 deg
                  = -1.626839 db < -5.710593 deg
Part 2b:    G(6j) = 0.416667 -0.416667j
                  = 0.589256 < -45.000000 deg
                  = -4.593925 db < -45.000000 deg
Part 2c:    G(60j) = 0.008251 -0.082508j
                  = 0.082920 < -84.289407 deg
                  = -21.626839 db < -84.289407 deg
```

Part 3

$$G(s) = \frac{X(s)}{F(s)} = \frac{s}{s+35}$$

The equivalent frequency response function is

$$G(j\omega) = \frac{\omega^2}{35^2 + \omega^2} + j\frac{35\omega}{35 + \omega^2}$$

We can now evaluate these at the desired points

In [10]:
```
G3s = s/(s + 35)
G3jw = w**2/(35**2 + w**2) + (35*w)/(35**2 + w**2)*sympy.I

# make sure my derivation is correct
np.testing.assert_equal(G3s.subs([(s,2*sympy.I)]).evalf(),
G3jw.subs([(w, 2)]).evalf())
np.testing.assert_almost_equal(G3s.subs([(s,35*sympy.I)]).evalf(), G3
jw.subs([(w, 35)]).evalf())
np.testing.assert_almost_equal(G3s.subs([(s,60*sympy.I)]).evalf(), G3
jw.subs([(w, 60)]).evalf())

# output correct answers in multiple forms
G3a = complex(G3jw.subs([(w, 2)]).evalf())
print("Part 3a:    G(2j) = %05f %+05fj" % (np.real(G3a),
np.imag(G3a)))
print("                  = %05f < %05f deg" % (np.linalg.norm(G3a),ph
asedeg(np.angle(G3a))))
print("                  = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G3a)),phasedeg(np.angle(G3a))))

G3b = complex(G3jw.subs([(w, 35)]).evalf())
print("Part 2b:    G(35j) = %05f %+05fj" % (np.real(G3b),
np.imag(G3b)))
print("                  = %05f < %05f deg" % (np.linalg.norm(G3b),ph
asedeg(np.angle(G3b))))
print("                  = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G3b)),phasedeg(np.angle(G3b))))

G3c = complex(G3jw.subs([(w, 500)]).evalf())
print("Part 2c:    G(500j) = %05f %+05fj" % (np.real(G3c), np.imag(G3
c)))
print("                  = %05f < %05f deg" % (np.linalg.norm(G3c),ph
asedeg(np.angle(G3c))))
print("                  = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G3c)),phasedeg(np.angle(G3c))))
```

```
Part 3a:    G(2j) = 0.003255 +0.056957j
                  = 0.057050 < 86.729512 deg
                  = -24.874919 db < 86.729512 deg
Part 2b:    G(35j) = 0.500000 +0.500000j
                  = 0.707107 < 45.000000 deg
                  = -3.010300 db < 45.000000 deg
Part 2c:    G(500j) = 0.995124 +0.069659j
                  = 0.997559 < 4.004173 deg
                  = -0.021228 db < 4.004173 deg
```

Part 4

$$G(s) = \frac{X(s)}{F(s)} = \frac{0.21(s+2)}{(s+3.05)}$$

The equivalent frequency response function is

$$G(j\omega) = \frac{0.21(6.1+\omega^2)}{3.05^2+\omega^2} + j\frac{0.21(1.05\omega)}{3.05^2+\omega^2}$$

We can now evaluate these at the desired points

In [11]:
```
G4s = 0.21*(s+2)/(s + 3.05)
G4jw = 0.21*(6.1+w**2)/(3.05**2 + w**2) + (0.21*1.05*w)/(3.05**2 + w*
*2)*sympy.I

# make sure my derivation is correct
np.testing.assert_almost_equal(G4s.subs([(s,.247*sympy.I)]).evalf(),
G4jw.subs([(w, 0.247)]).evalf())
np.testing.assert_almost_equal(G4s.subs([(s,2.47*sympy.I)]).evalf(),
G4jw.subs([(w, 2.47)]).evalf())
np.testing.assert_almost_equal(G4s.subs([(s,24.7*sympy.I)]).evalf(),
G4jw.subs([(w, 24.7)]).evalf())

# output correct answers in multiple forms
G4a = complex(G4jw.subs([(w, 0.247)]).evalf())
print("Part 4a:    G(0.247j) = %05f %+05fj" % (np.real(G4a),
np.imag(G4a)))
print("                      = %05f < %05f deg" % (np.linalg.norm(G4a),ph
asedeg(np.angle(G4a))))
print("                      = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G4a)),phasedeg(np.angle(G4a))))

G4b = complex(G4jw.subs([(w, 2.47)]).evalf())
print("Part 4b:    G(2.47j) = %05f %+05fj" % (np.real(G4b), np.imag(G
4b)))
print("                      = %05f < %05f deg" % (np.linalg.norm(G4b),ph
asedeg(np.angle(G4b))))
print("                      = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G4b)),phasedeg(np.angle(G4b))))

G4c = complex(G4jw.subs([(w, 24.7)]).evalf())
print("Part 4c:    G(24.7j) = %05f %+05fj" % (np.real(G4c), np.imag(G
4c)))
print("                      = %05f < %05f deg" % (np.linalg.norm(G4c),ph
asedeg(np.angle(G4c))))
print("                      = %05f db < %05f deg" % (magdb(np.linalg.nor
m(G4c)),phasedeg(np.angle(G4c))))
```

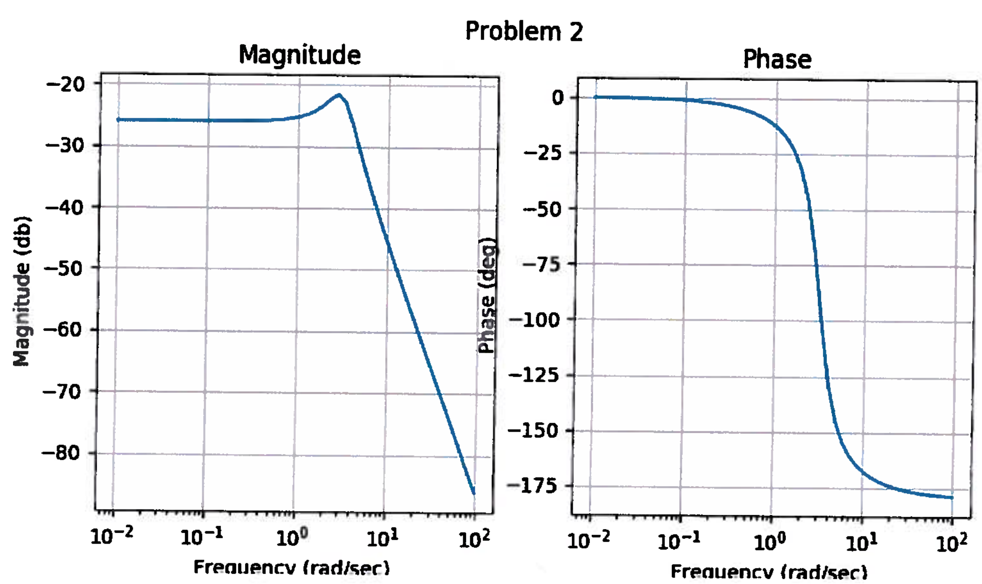
```
Part 4a:    G(0.247j) = 0.138176 +0.005817j
                  = 0.138298 < 2.410464 deg
                  = -17.183661 db < 2.410464 deg
Part 4b:    G(2.47j) = 0.166339 +0.035358j
                  = 0.170056 < 12.000533 deg
                  = -15.388179 db < 12.000533 deg
Part 4c:    G(24.7j) = 0.208914 +0.008793j
                  = 0.209099 < 2.410116 deg
                  = -13.592953 db < 2.410116 deg
```

Problem 2

We now need to draw the Bode approximations for each of the four systems. The students should provide clear and professional looking plots. In addition, the approximations should be accurate and the points computed above should be identified on the plots.

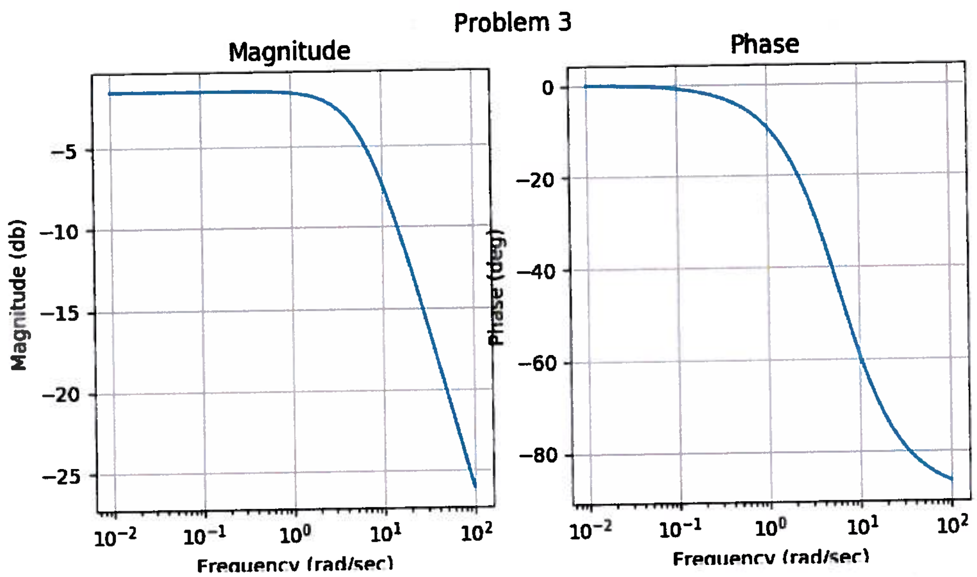
In [12]:
```
# Problem 2-5
# generate all the bode plots and then plot them all
sys1 = signal.TransferFunction([0.5],[1, 2, 10])
sys2 = signal.TransferFunction([5], [1, 6])
sys3 = signal.TransferFunction([1, 0], [1, 35])
sys4 = signal.TransferFunction([0.21, 0.42], [1, 3.05])

w = np.logspace(-2, 2)

w1, mag1, phase1 = signal.bode(sys1, w)
w2, mag2, phase2 = signal.bode(sys2, w)
w3, mag3, phase3 = signal.bode(sys3, np.logspace(-1,3))
w4, mag4, phase4 = signal.bode(sys4,w )
```

In [13]:
```
# Problem 2
fig, axarr=plt.subplots(1,2, figsize=figsize)
axarr[0].semilogx(w1,mag1)
axarr[0].set_title('Magnitude')
axarr[0].set_xlabel('Frequency (rad/sec)')
axarr[0].set_ylabel('Magnitude (db)')
axarr[0].grid(True)

axarr[1].semilogx(w1, phase1)
axarr[1].set_title('Phase')
axarr[1].set_xlabel('Frequency (rad/sec)')
axarr[1].set_ylabel('Phase (deg)')
axarr[1].grid(True)

fig.suptitle('Problem 2')
plt.show()
```

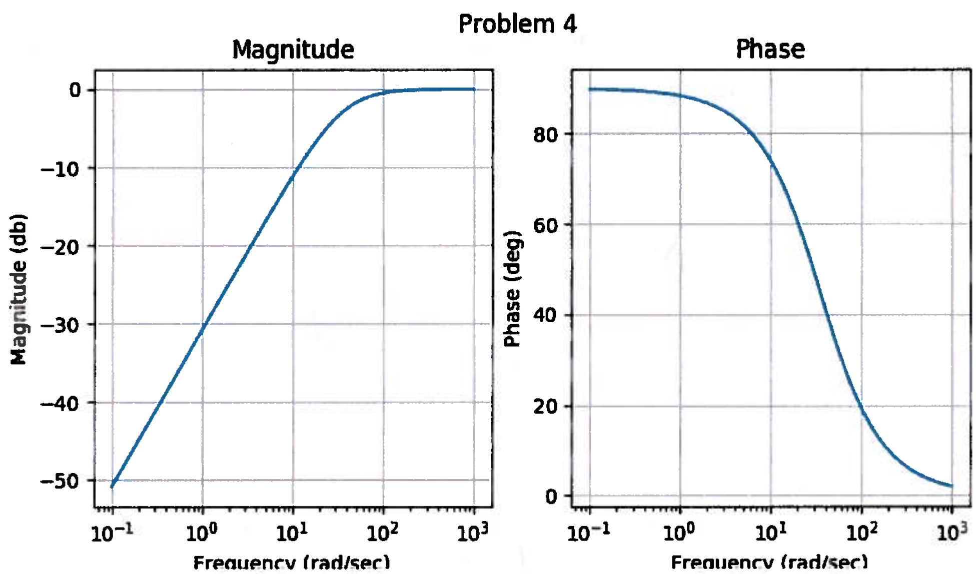


In [14]:

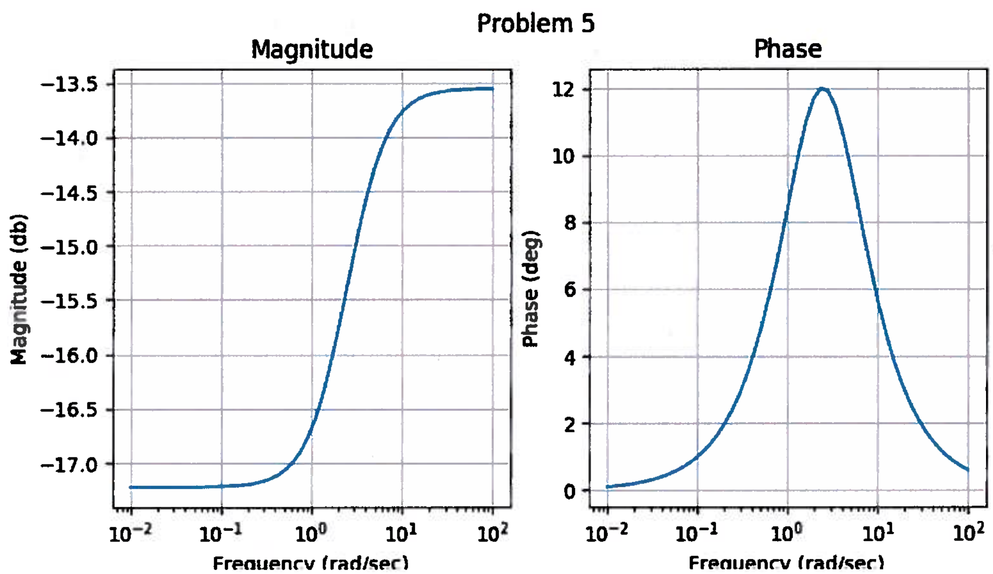
```
# Problem 3
fig, axarr=plt.subplots(1,2, figsize=figsize)
axarr[0].semilogx(w2,mag2)
axarr[0].set_title('Magnitude')
axarr[0].set_xlabel('Frequency (rad/sec)')
axarr[0].set_ylabel('Magnitude (db)')
axarr[0].grid(True)

axarr[1].semilogx(w2, phase2)
axarr[1].set_title('Phase')
axarr[1].set_xlabel('Frequency (rad/sec)')
axarr[1].set_ylabel('Phase (deg)')
axarr[1].grid(True)

fig.suptitle('Problem 3')
plt.show()
```



In [15]:
```
# Problem 4
fig, axarr=plt.subplots(1,2, figsize=figsize)
axarr[0].semilogx(w3,mag3)
axarr[0].set_title('Magnitude')
axarr[0].set_xlabel('Frequency (rad/sec)')
axarr[0].set_ylabel('Magnitude (db)')
axarr[0].grid(True)

axarr[1].semilogx(w3, phase3)
axarr[1].set_title('Phase')
axarr[1].set_xlabel('Frequency (rad/sec)')
axarr[1].set_ylabel('Phase (deg)')
axarr[1].grid(True)

fig.suptitle('Problem 4')
plt.show()
```



In [16]:
```
# Problem 5
fig, axarr=plt.subplots(1,2, figsize=figsize)
axarr[0].semilogx(w4,mag4)
axarr[0].set_title('Magnitude')
axarr[0].set_xlabel('Frequency (rad/sec)')
axarr[0].set_ylabel('Magnitude (db)')
axarr[0].grid(True)

axarr[1].semilogx(w4, phase4)
axarr[1].set_title('Phase')
axarr[1].set_xlabel('Frequency (rad/sec)')
axarr[1].set_ylabel('Phase (deg)')
axarr[1].grid(True)

fig.suptitle('Problem 5')
plt.show()
```



Problem 6

For this question, the students should use the computations from Problem 1 and list what the steady state output equations are. Most will probably forget that the magnitude is a ratio between output/input magnitudes.

In [17]:

```
# part 1
print("61a. SS output: x(t) = %5f sin(2t + %5f)" % (np.linalg.norm(G1
a)*17.333, phasedeg(np.angle(G1a))) )
print("")
print("61a. SS output: x(t) = %5f sin(0.6t + %5f)" %
(np.linalg.norm(G2a)*10, phasedeg(np.angle(G2a))))
print("61b. SS output: x(t) = %5f sin(60t + %5f)" % (np.linalg.norm(G
2c)*10, phasedeg(np.angle(G2c))))
print("")
print("62a. SS output: x(t) = %5f sin(2t + %5f)" % (np.linalg.norm(G3
a)*10, phasedeg(np.angle(G3a))))
print("62b. SS output: x(t) = %5f sin(500t + %5f)" %
(np.linalg.norm(G3c)*10, phasedeg(np.angle(G3c))))
```

```
61a. SS output: x(t) = 1.201827 sin(2t + -33.690068)

61a. SS output: x(t) = 8.291977 sin(0.6t + -5.710593)
61b. SS output: x(t) = 0.829198 sin(60t + -84.289407)

62a. SS output: x(t) = 0.570498 sin(2t + 86.729512)
62b. SS output: x(t) = 9.975590 sin(500t + 4.004173)
```