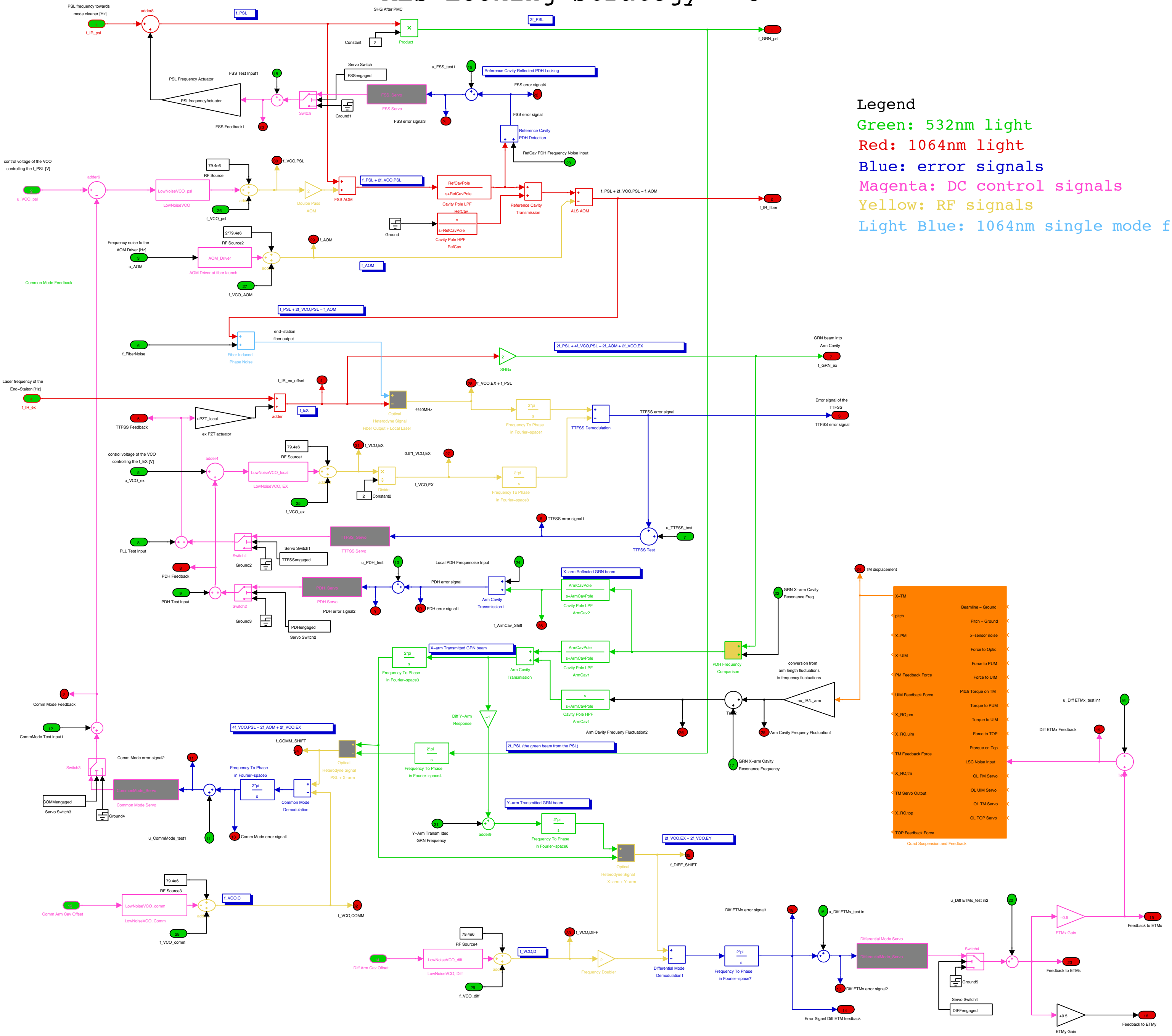
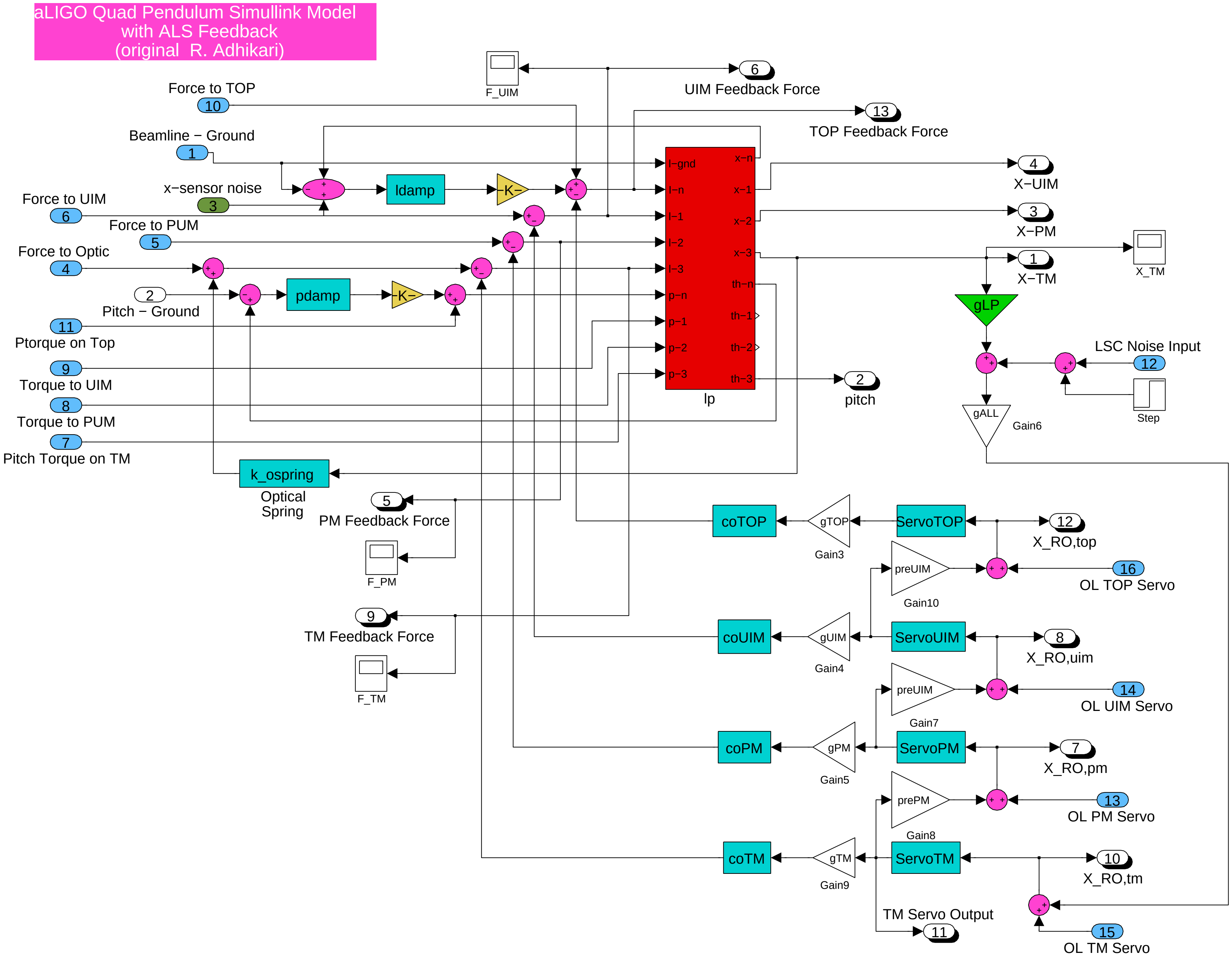
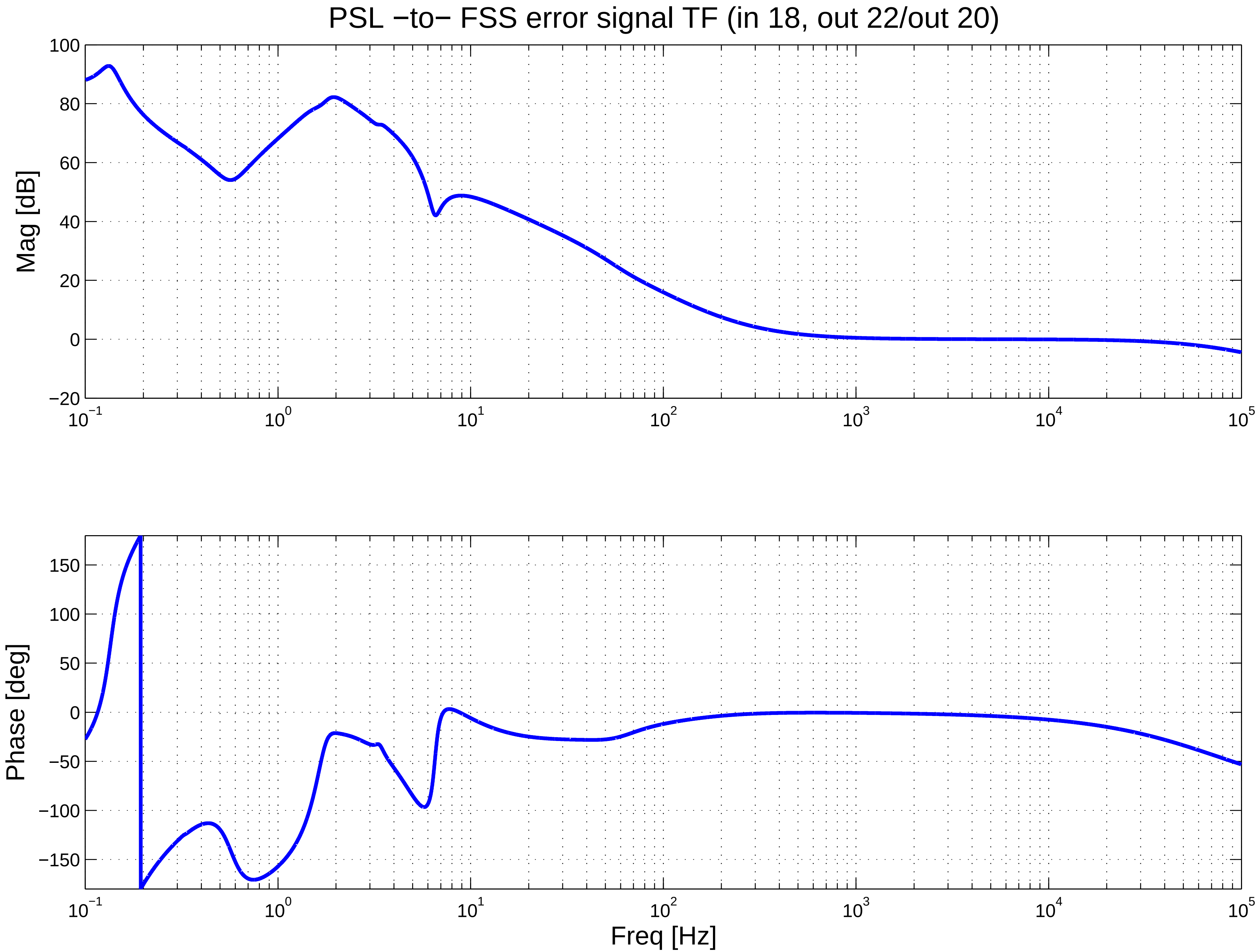
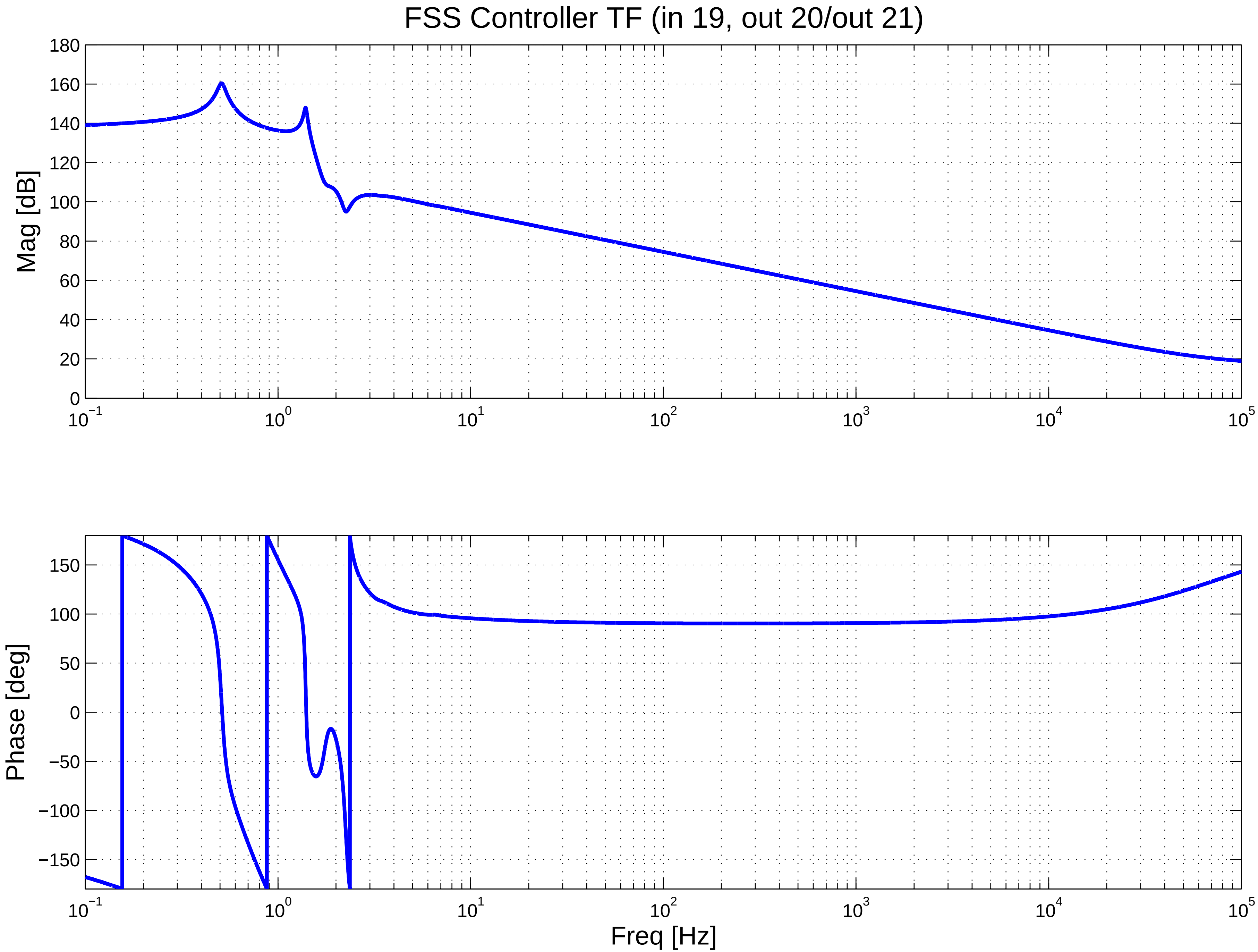
ALS Locking Strategy - 3

Legend
Green: 532nm light
Red: 1064nm light
Blue: error signals
Magenta: DC control signals
Yellow: RF signals
Light Blue: 1064nm single mode fil

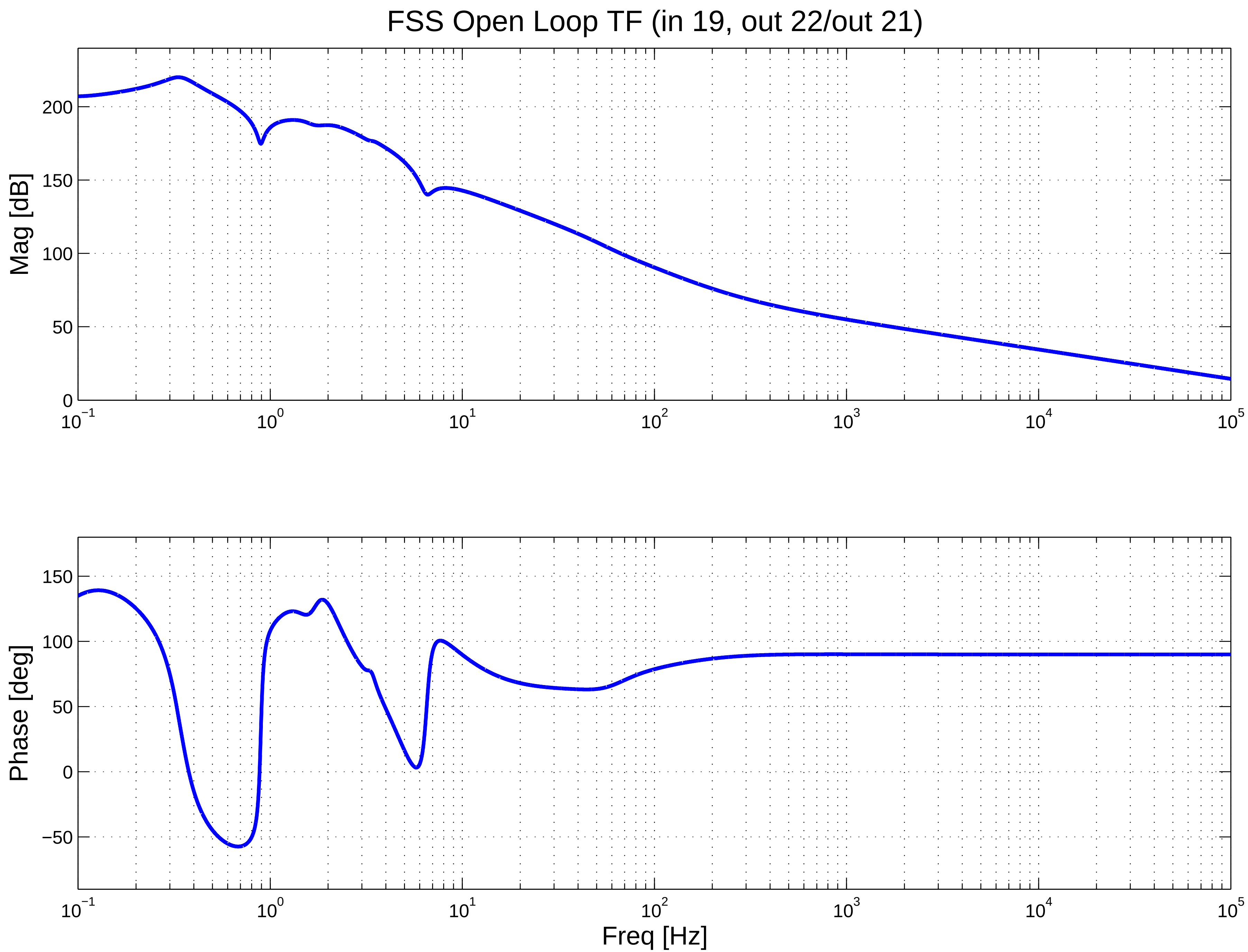


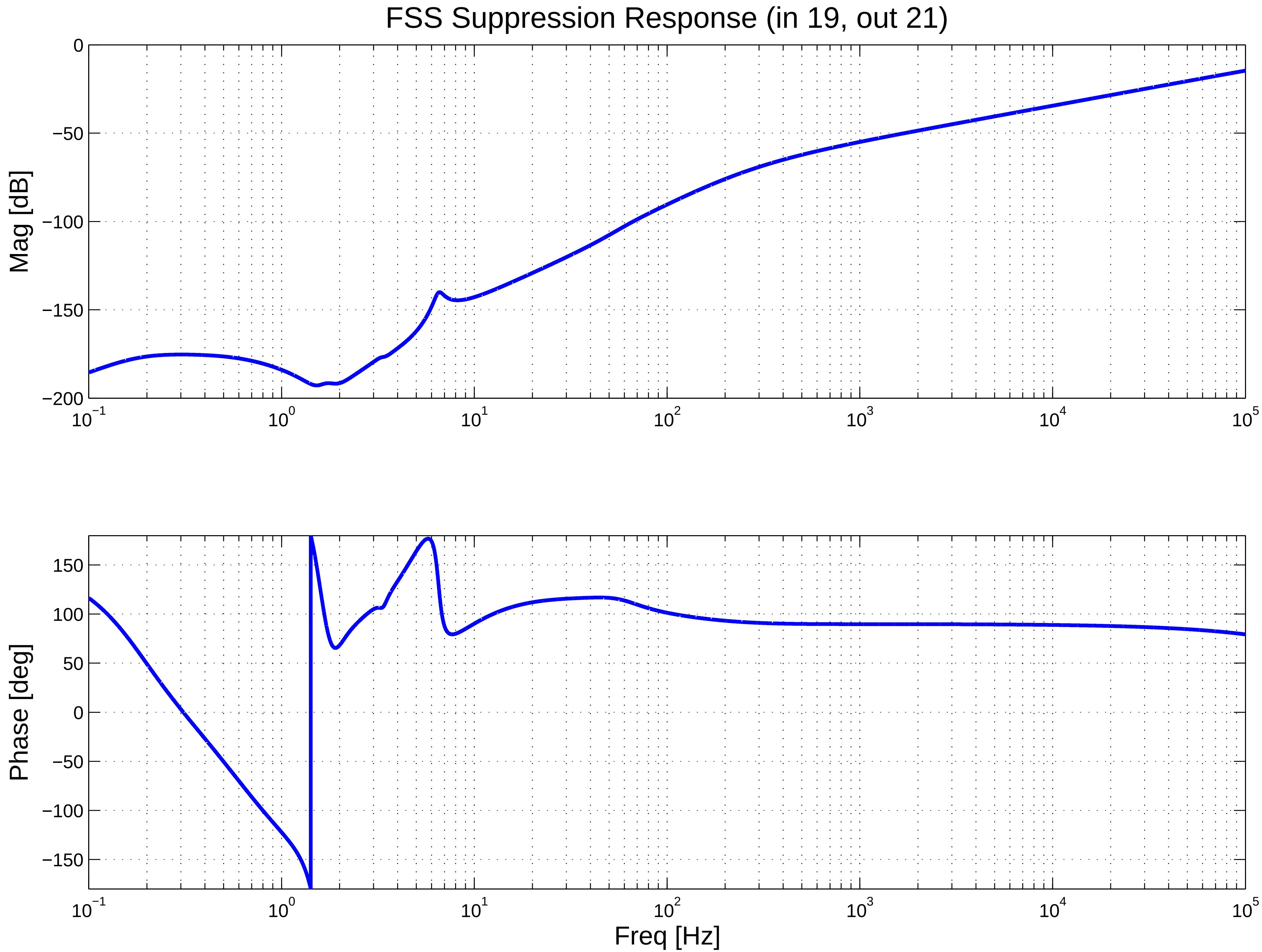

aLIGO Quad Pendulum Simullink Model
with ALS Feedback
(original R. Adhikari)
F_UIM
6
UIM Feedback Force
13
TOP Feedback Force
Force to TOP
10
Beamline − Ground
1
x−sensor noise
3
Force to UIM
6
Force to PUM
5
Force to Optic
4
2
Pitch − Ground
11
Ptorque on Top
9
Torque to UIM
8
Torque to PUM
7
Pitch Torque on TM
ldamp
-K-
pdamp
-K-
l−gnd
l−n
l−1
l−2
l−3
p−n
p−1
p−2
p−3
x−n
x−1
x−2
x−3
th−n
th−1
th−2
th−3
lp
4
X−UIM
3
X−PM
1
X−TM
X_TM
2
pitch
gLP
LSC Noise Input
12
Step
gALL
Gain6
k_ospring
Optical
Spring
5
PM Feedback Force
F_PM
9
TM Feedback Force
F_TM
coTOP
gTOP
Gain3
ServoTOP
12
X_RO,top
preUIM
Gain10
16
OL TOP Servo
coUIM
gUIM
Gain4
ServoUIM
8
X_RO,uim
preUIM
Gain7
14
OL UIM Servo
coPM
gPM
Gain5
ServoPM
7
X_RO,pm
prePM
Gain8
13
OL PM Servo
coTM
gTM
Gain9
ServoTM
10
X_RO,tm
TM Servo Output
11
15
OL TM Servo

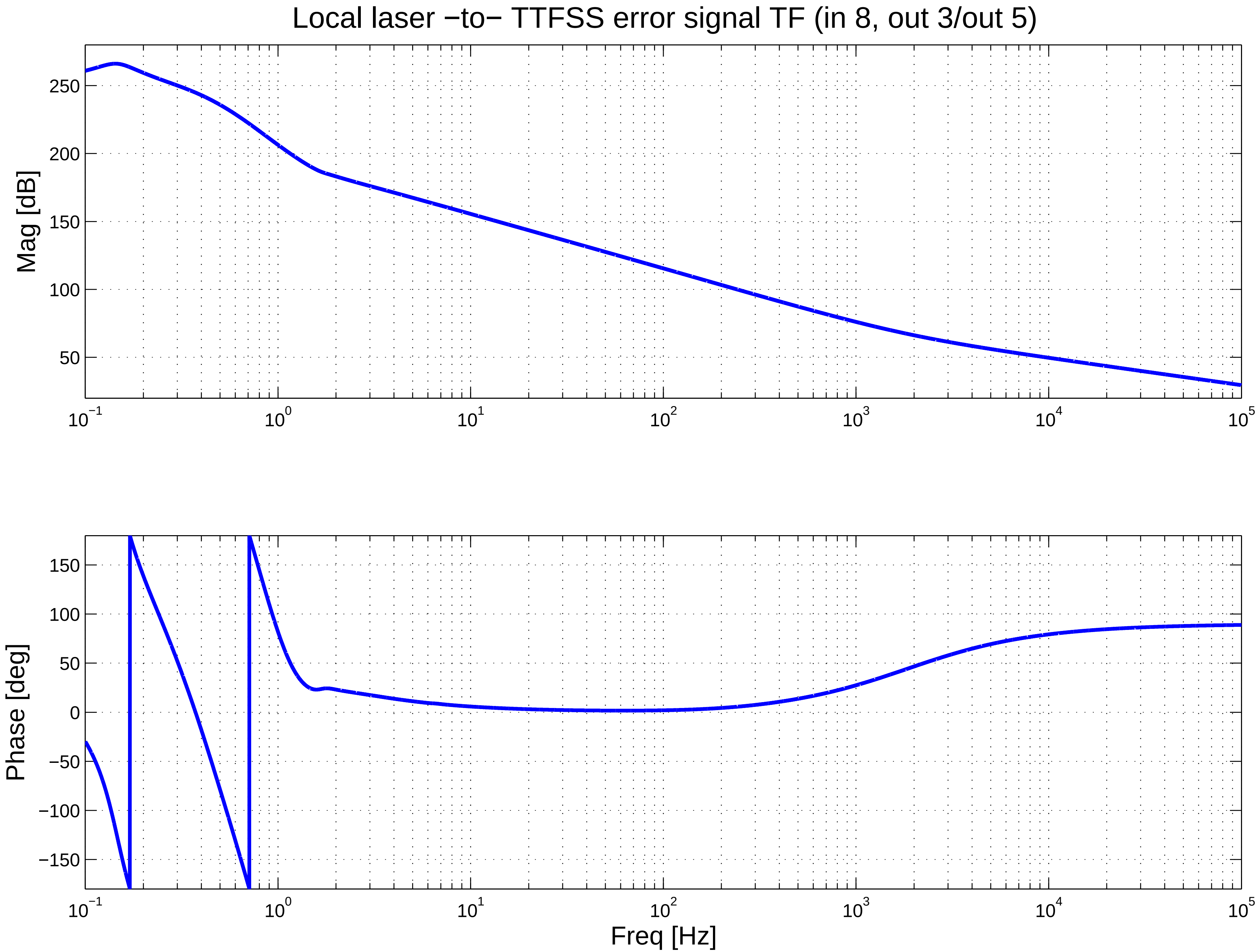

PSL −to− FSS error signal TF (in 18, out 22/out 20)
Mag [dB]
100
80
60
40
20
0
−20
Phase [deg]
150
100
50
0
−50
−100
−150
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Freq [Hz]

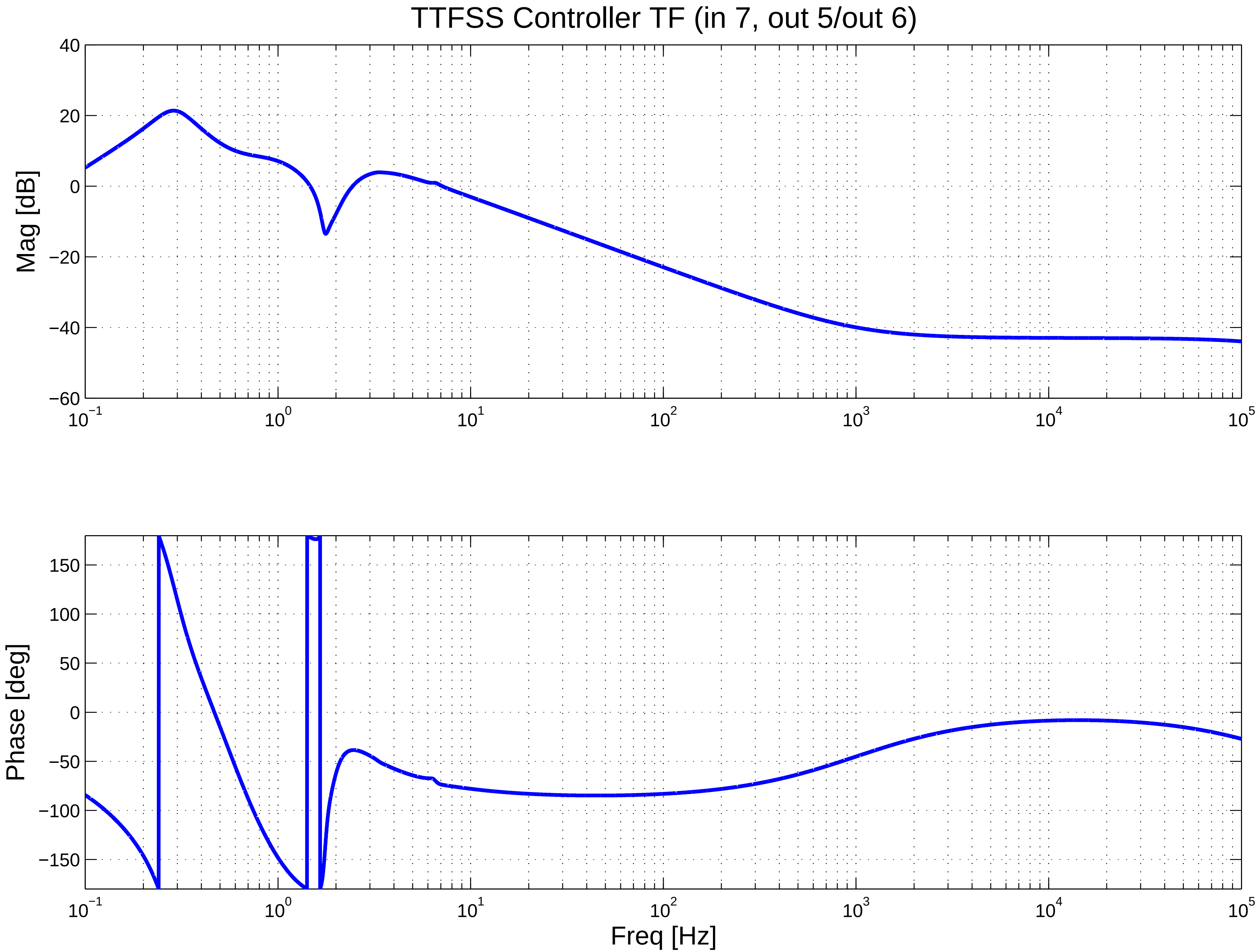


FSS Controller TF (in 19, out 20/out 21)
Mag [dB]
180
160
140
120
100
80
60
40
20
0
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]

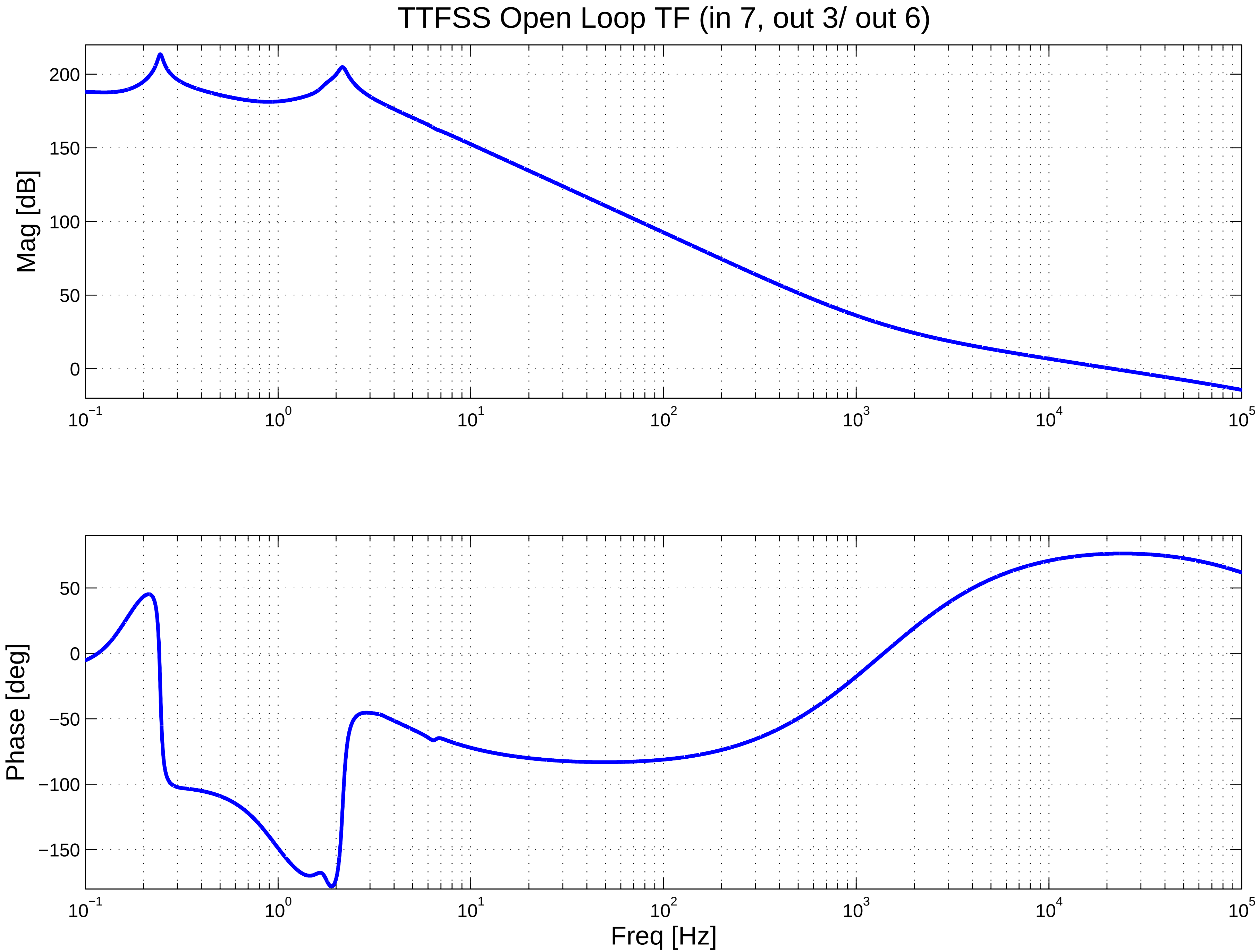

FSS Open Loop TF (in 19, out 22/out 21)
Mag [dB]
0
50
100
150
200
Phase [deg]
−50
0
50
100
150
10^−1
10^0
10^1
10^2
10^3
10^4
10^5
Freq [Hz]


FSS Suppression Response (in 19, out 21)
Mag [dB]
0
−50
−100
−150
−200
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]

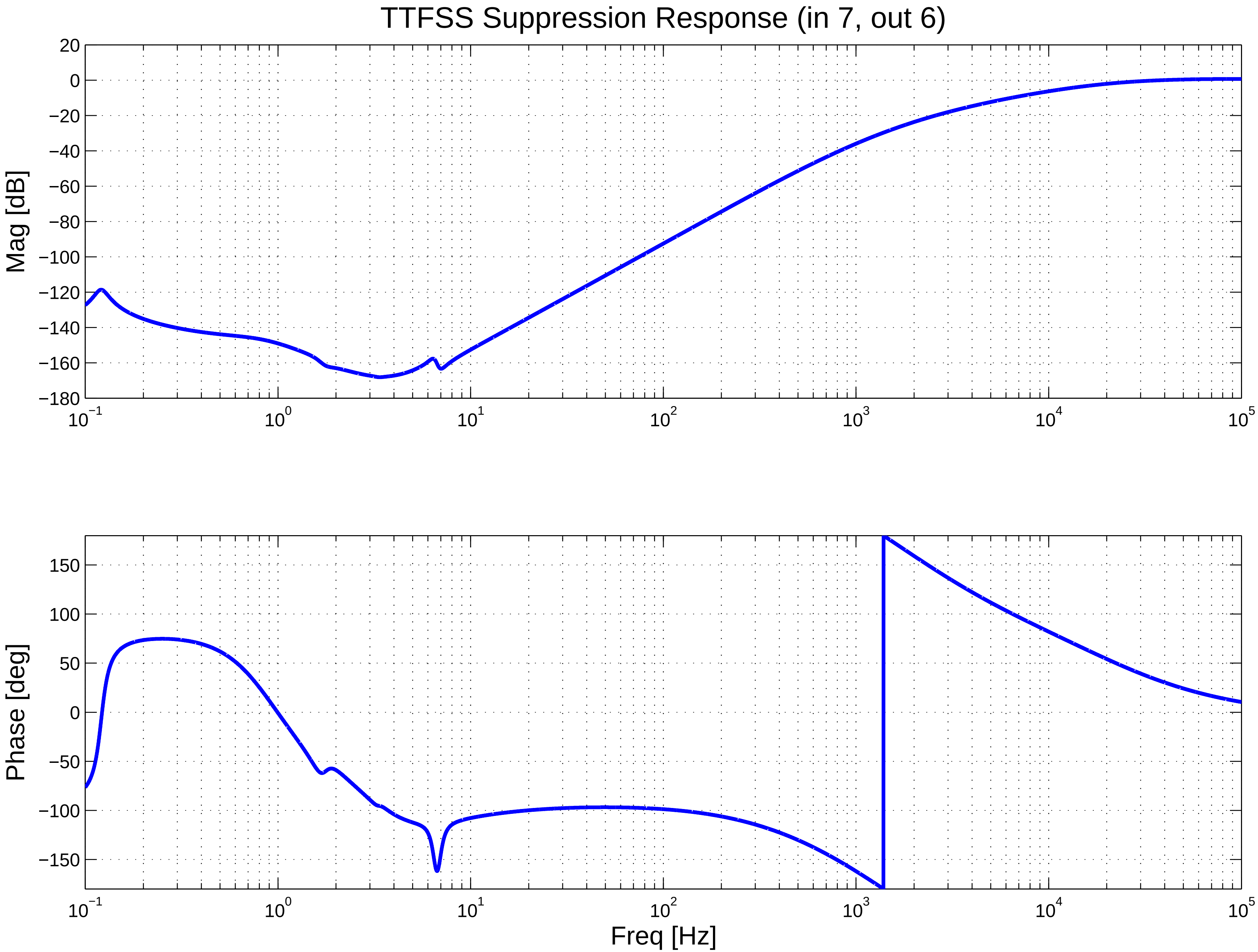

Local laser −to− TTFSS error signal TF (in 8, out 3/out 5)
Mag [dB]
250
200
150
100
50
Phase [deg]
150
100
50
0
−50
−100
−150
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Freq [Hz]



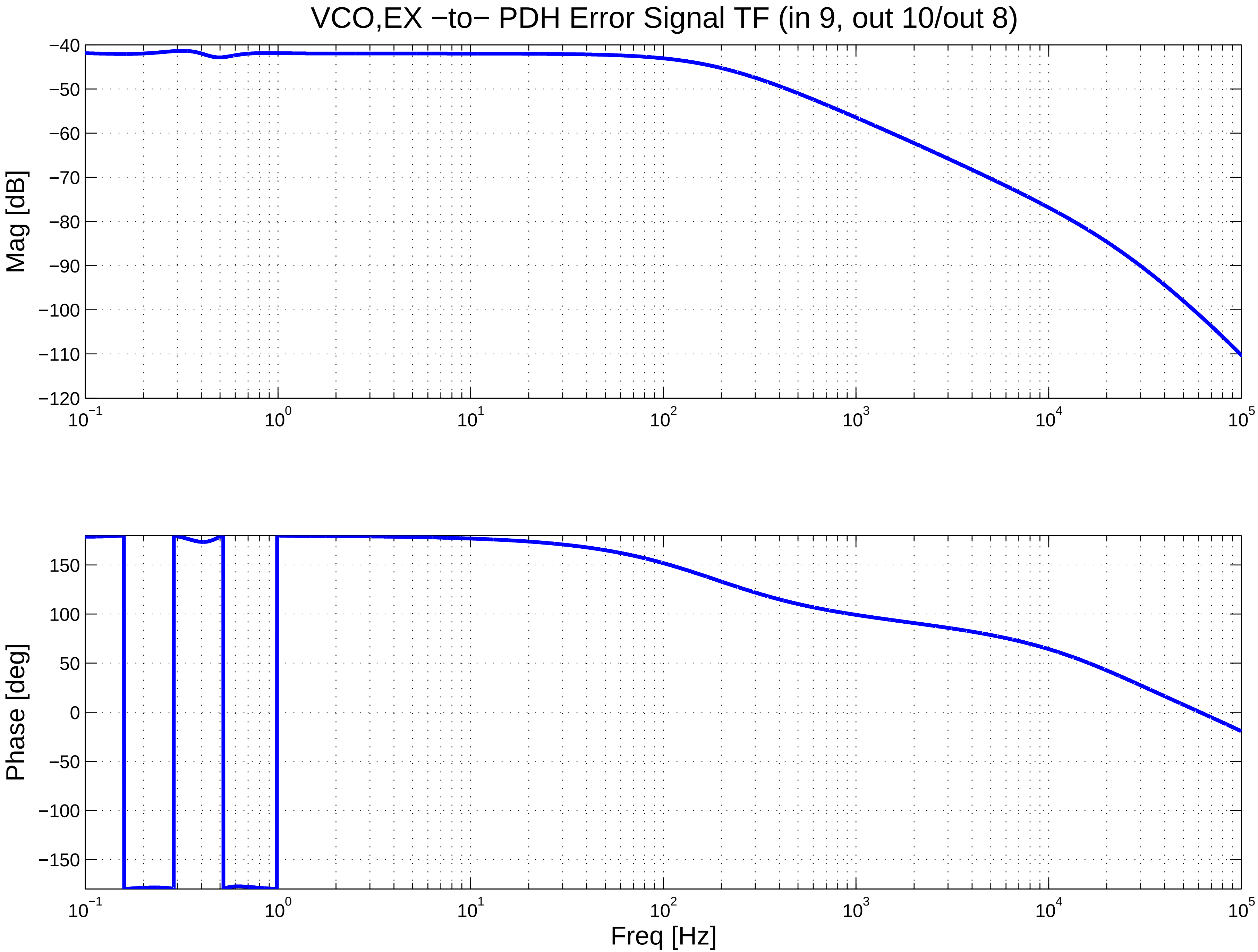
TTFSS Controller TF (in 7, out 5/out 6)
Mag [dB]
40
20
0
−20
−40
−60
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]

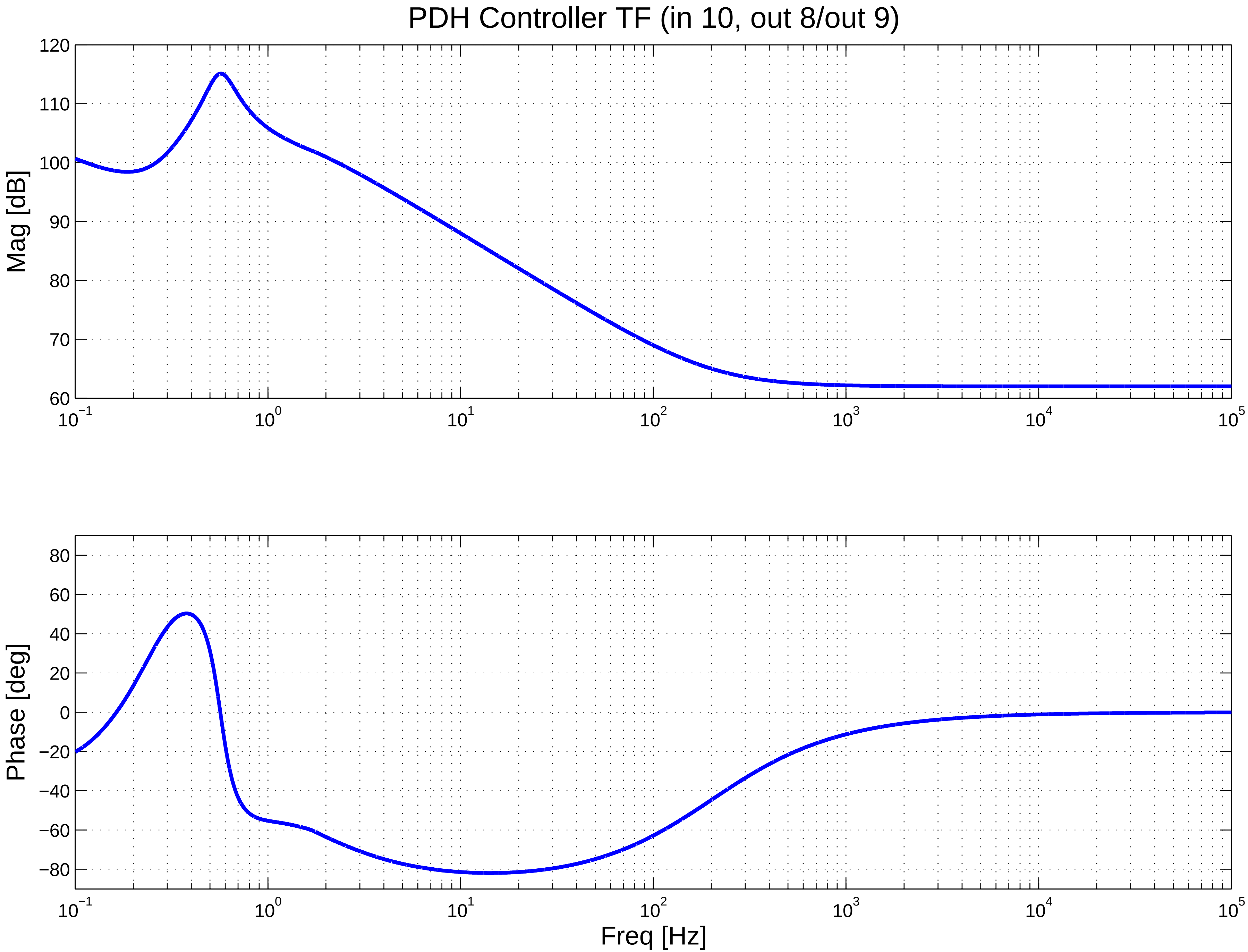


TTFSS Open Loop TF (in 7, out 3/ out 6)
Mag [dB]
200
150
100
50
0
10^−1
10^0
10^1
10^2
10^3
10^4
10^5
Phase [deg]
50
0
−50
−100
−150
Freq [Hz]

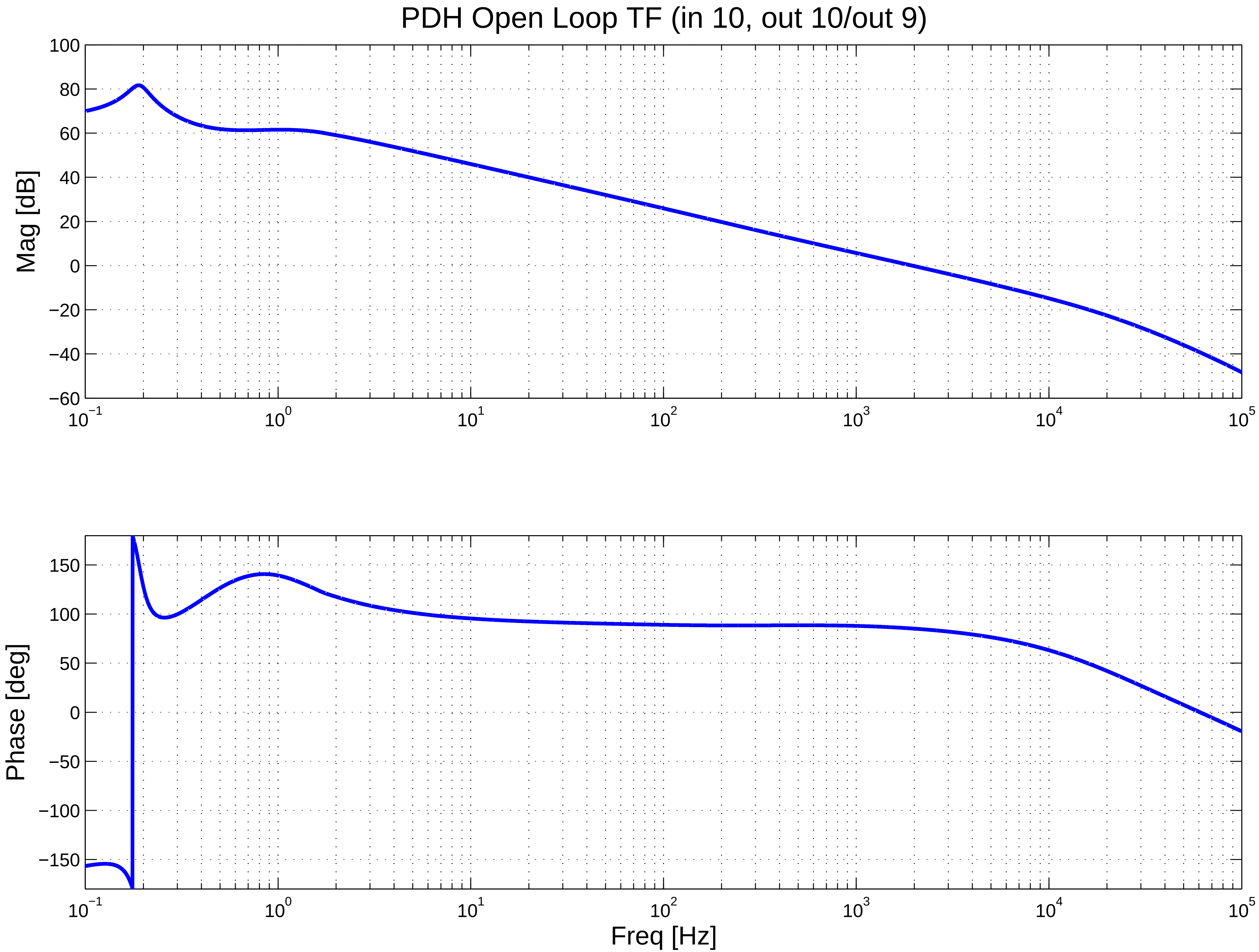


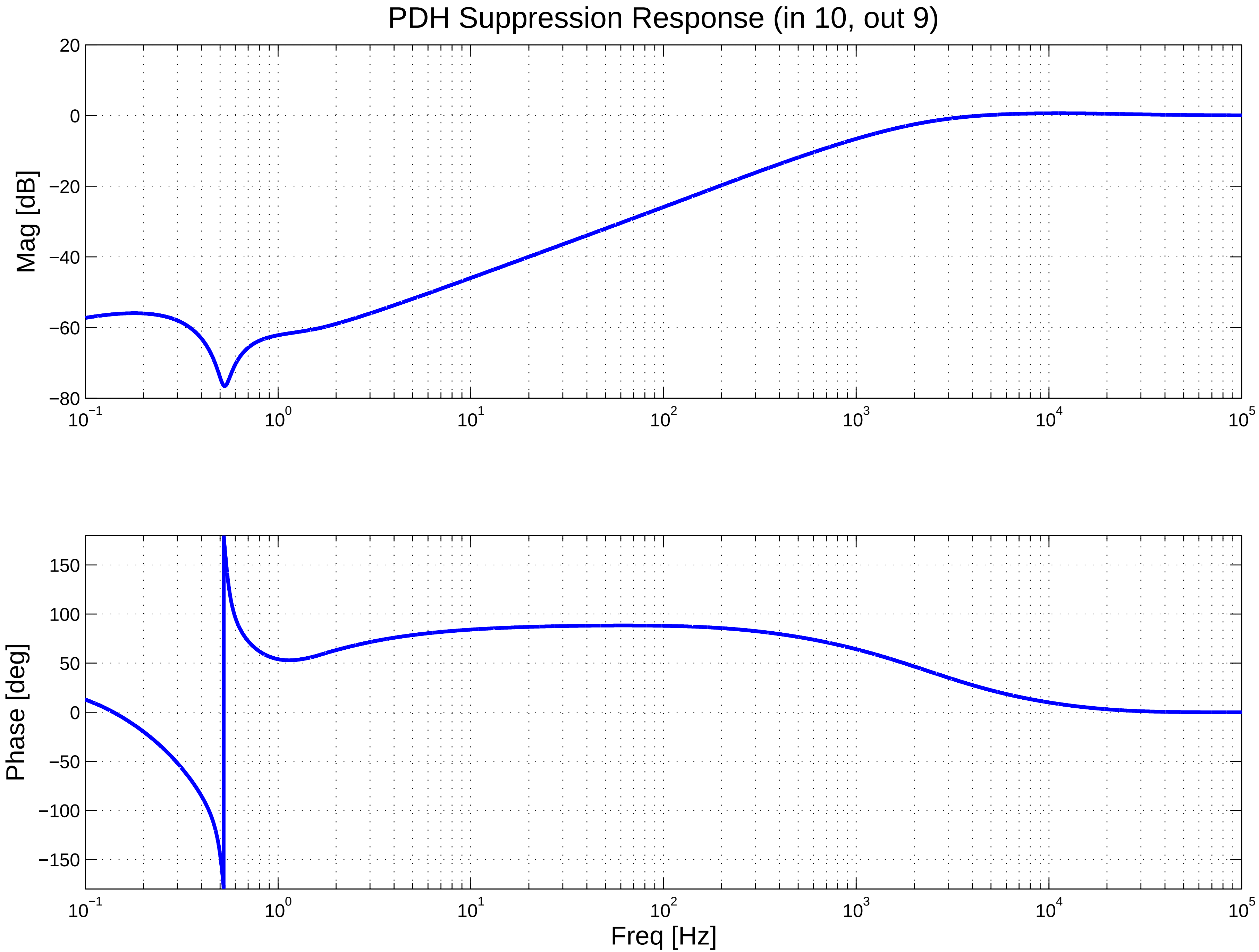
TTFSS Suppression Response (in 7, out 6)
Mag [dB]
20
0
−20
−40
−60
−80
−100
−120
−140
−160
−180
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]

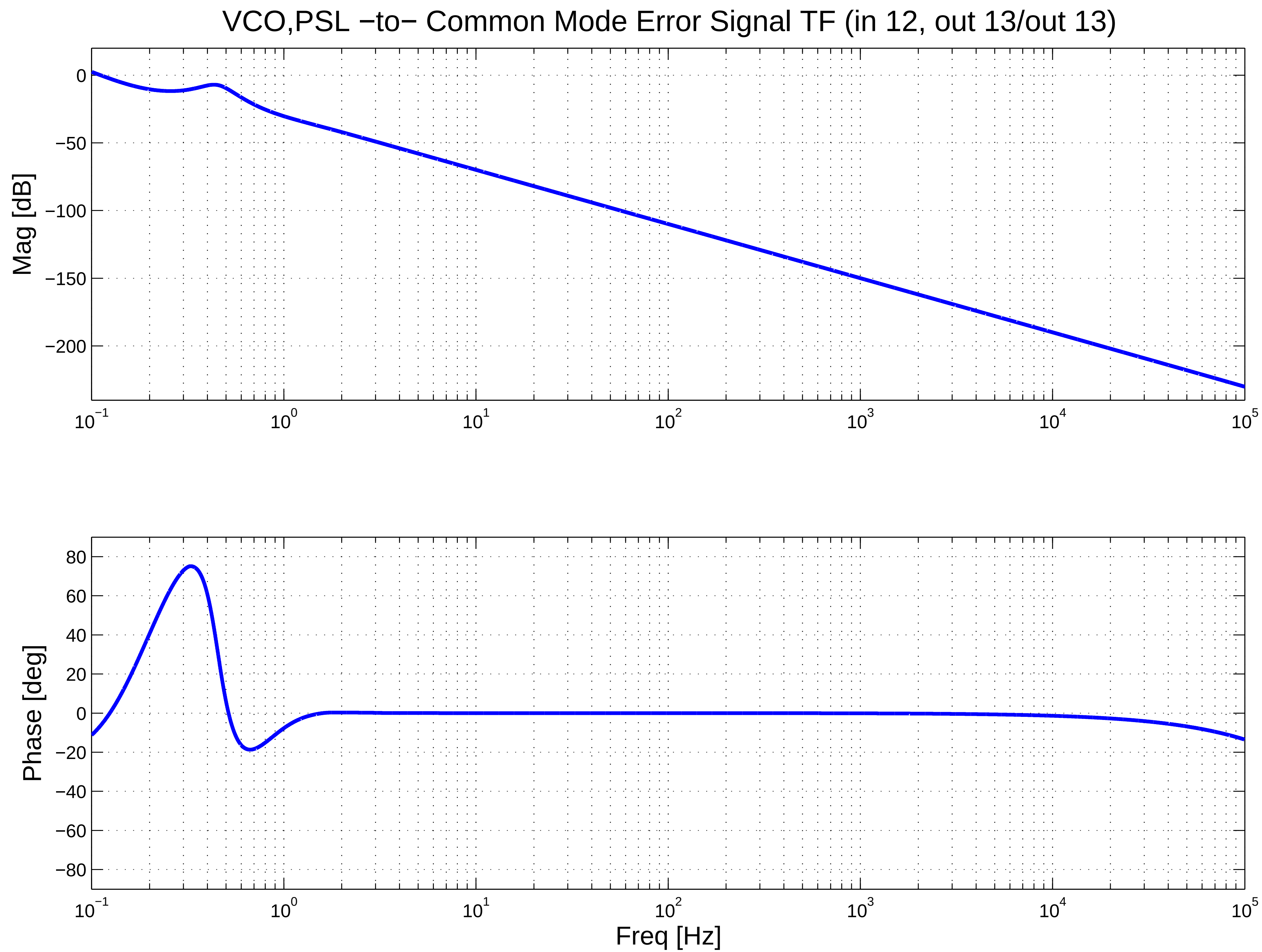

VCO,EX −to− PDH Error Signal TF (in 9, out 10/out 8)
Mag [dB]
−40
−50
−60
−70
−80
−90
−100
−110
−120
10−1
100
101
102
103
104
105
Phase [deg]
150
100
50
0
−50
−100
−150
10−1
100
101
102
103
104
105
Freq [Hz]



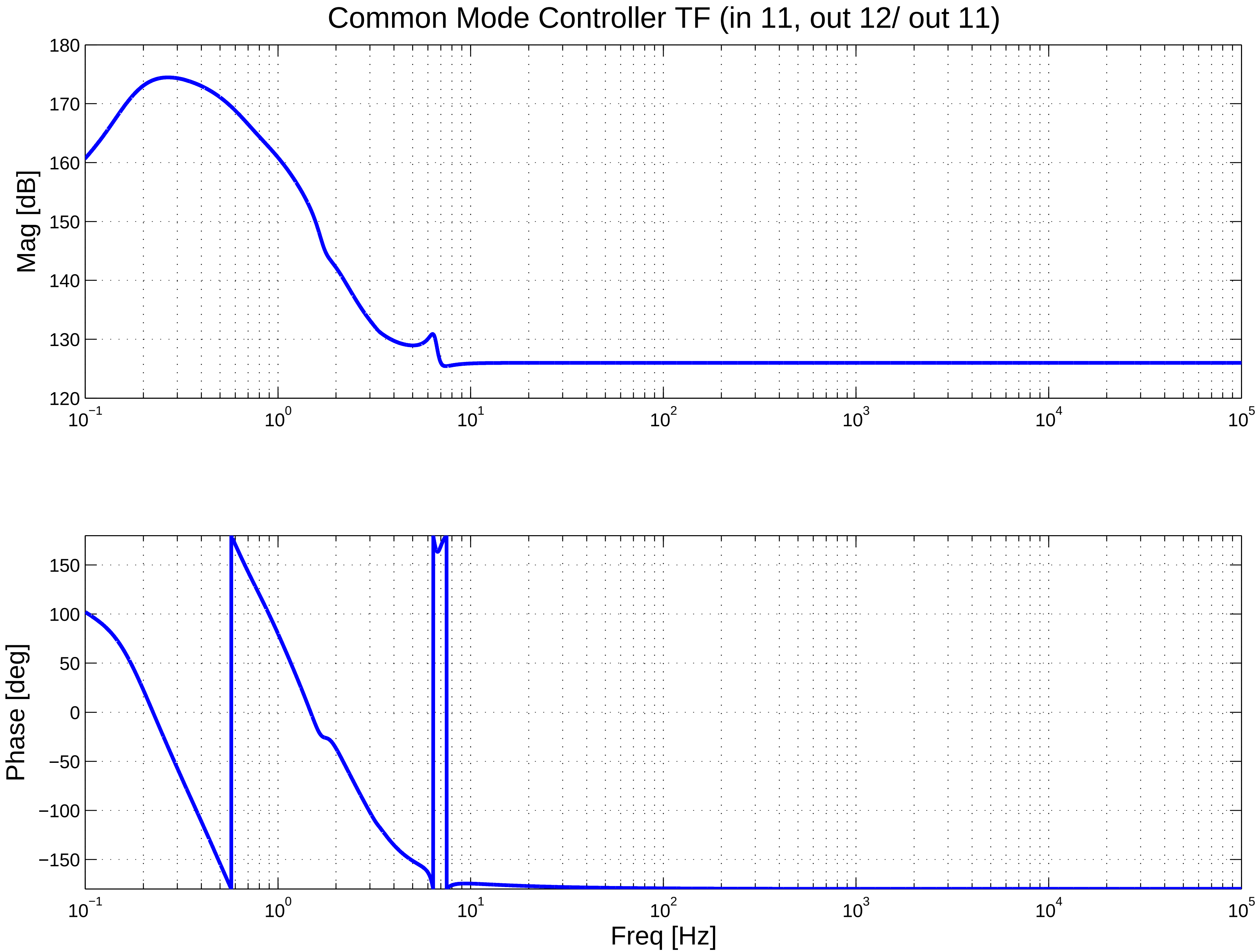
PDH Controller TF (in 10, out 8/out 9)
Mag [dB]
120
110
100
90
80
70
60
Phase [deg]
80
60
40
20
0
−20
−40
−60
−80
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Freq [Hz]


PDH Open Loop TF (in 10, out 10/out 9)
Mag [dB]
100
80
60
40
20
0
−20
−40
−60
Phase [deg]
150
100
50
0
−50
−100
−150
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Freq [Hz]

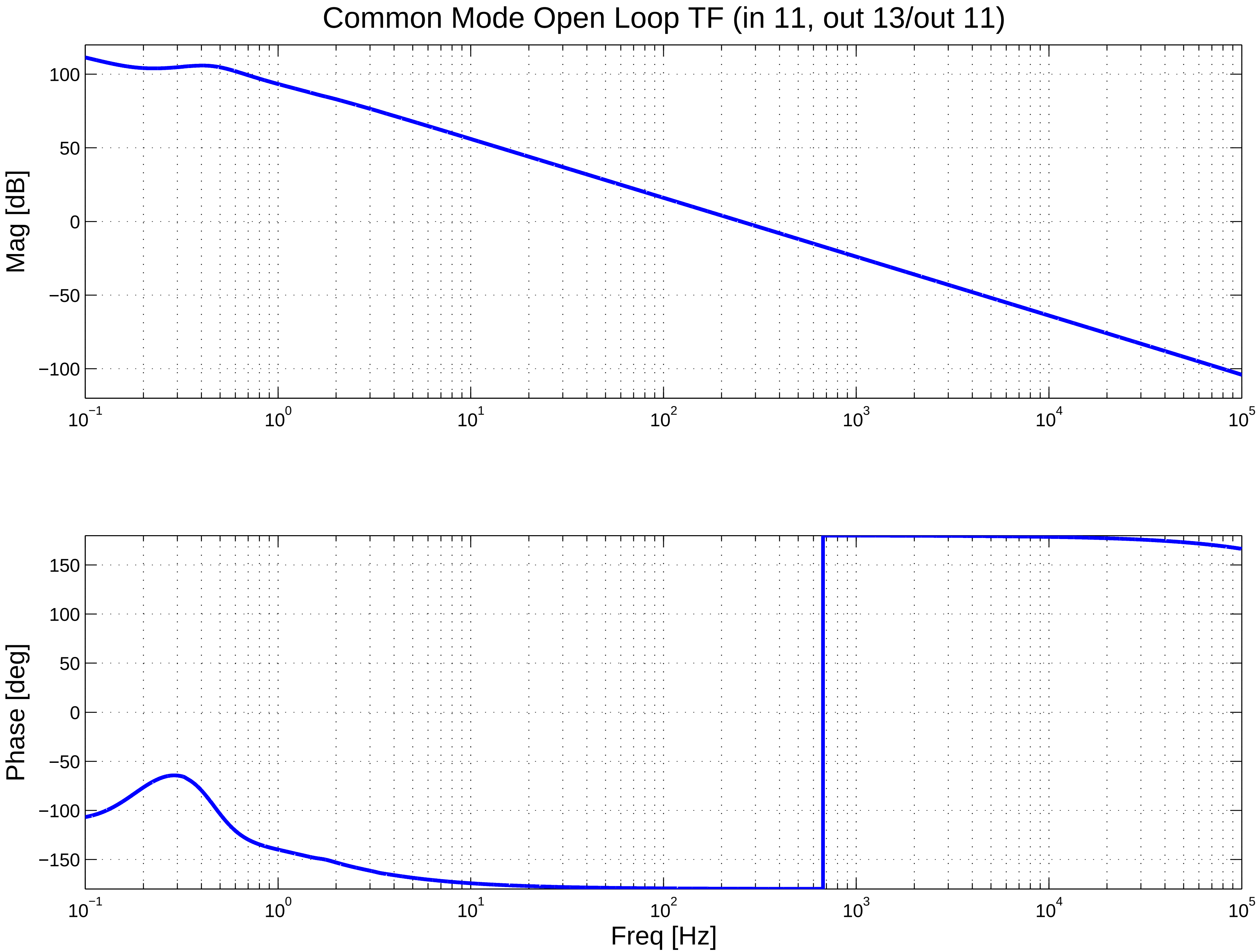

PDH Suppression Response (in 10, out 9)
Mag [dB]
20
0
−20
−40
−60
−80
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]

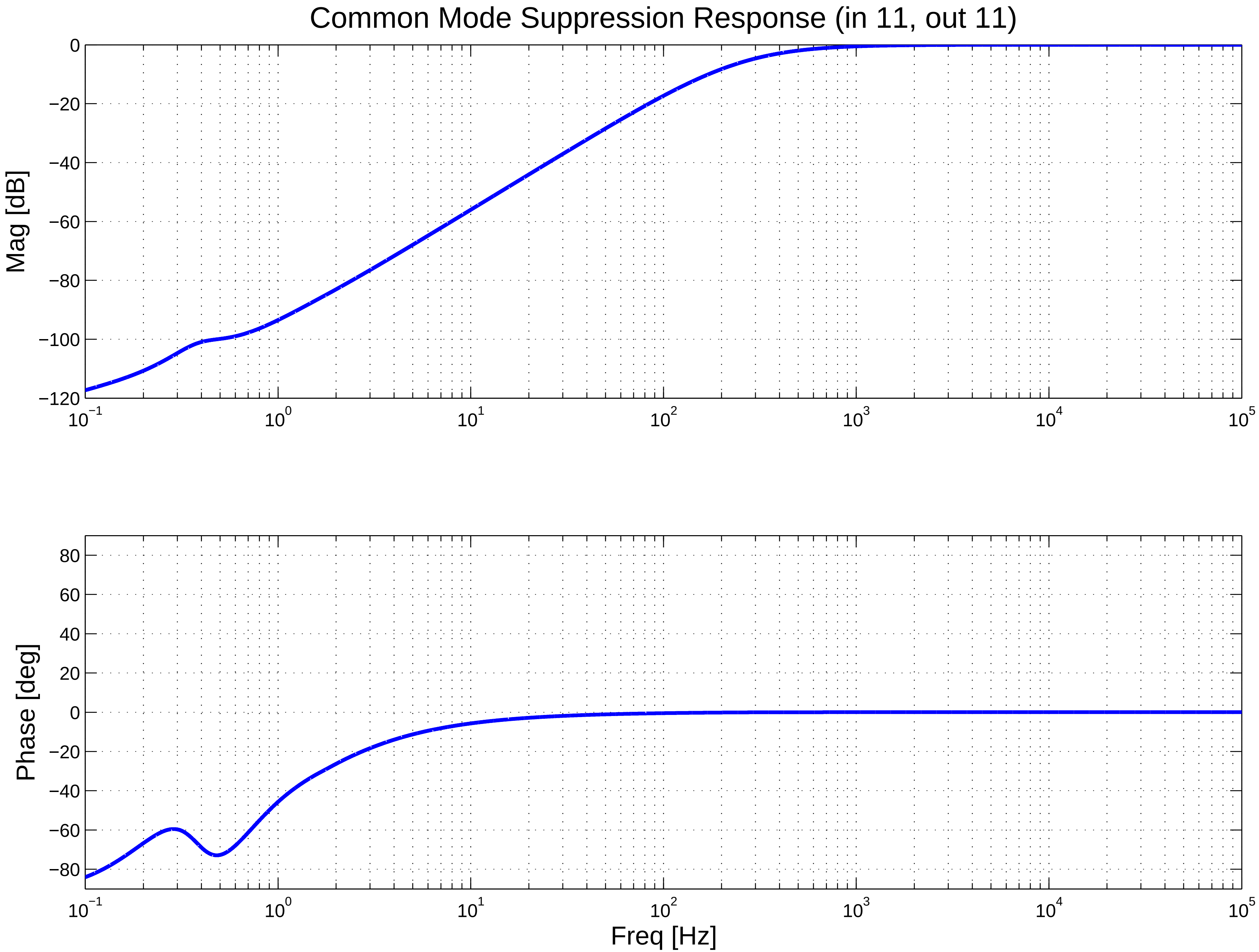


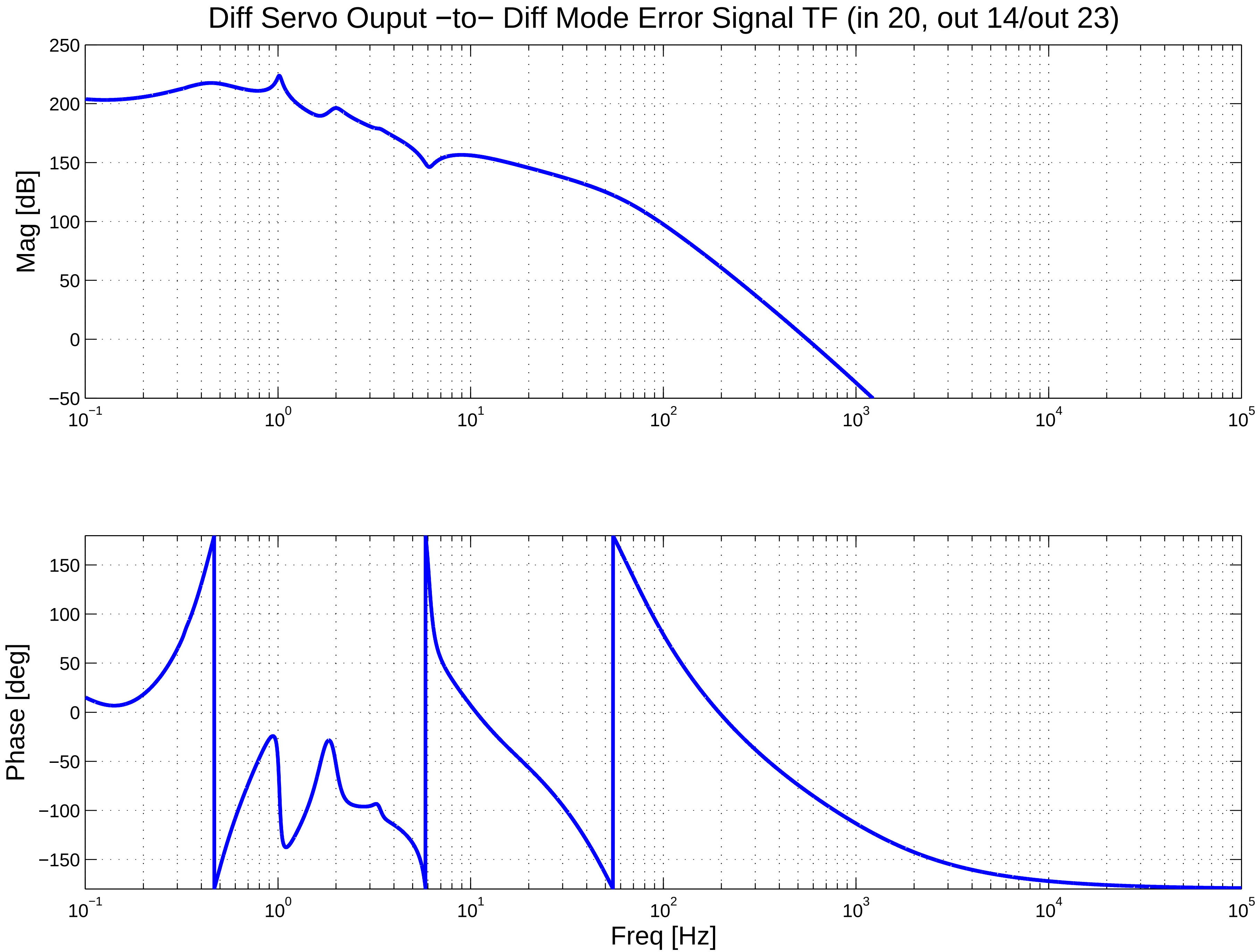
VCO,PSL −to− Common Mode Error Signal TF (in 12, out 13/out 13)
Mag [dB]
0
−50
−100
−150
−200
Phase [deg]
80
60
40
20
0
−20
−40
−60
−80
10^−1
10^0
10^1
10^2
10^3
10^4
10^5
Freq [Hz]

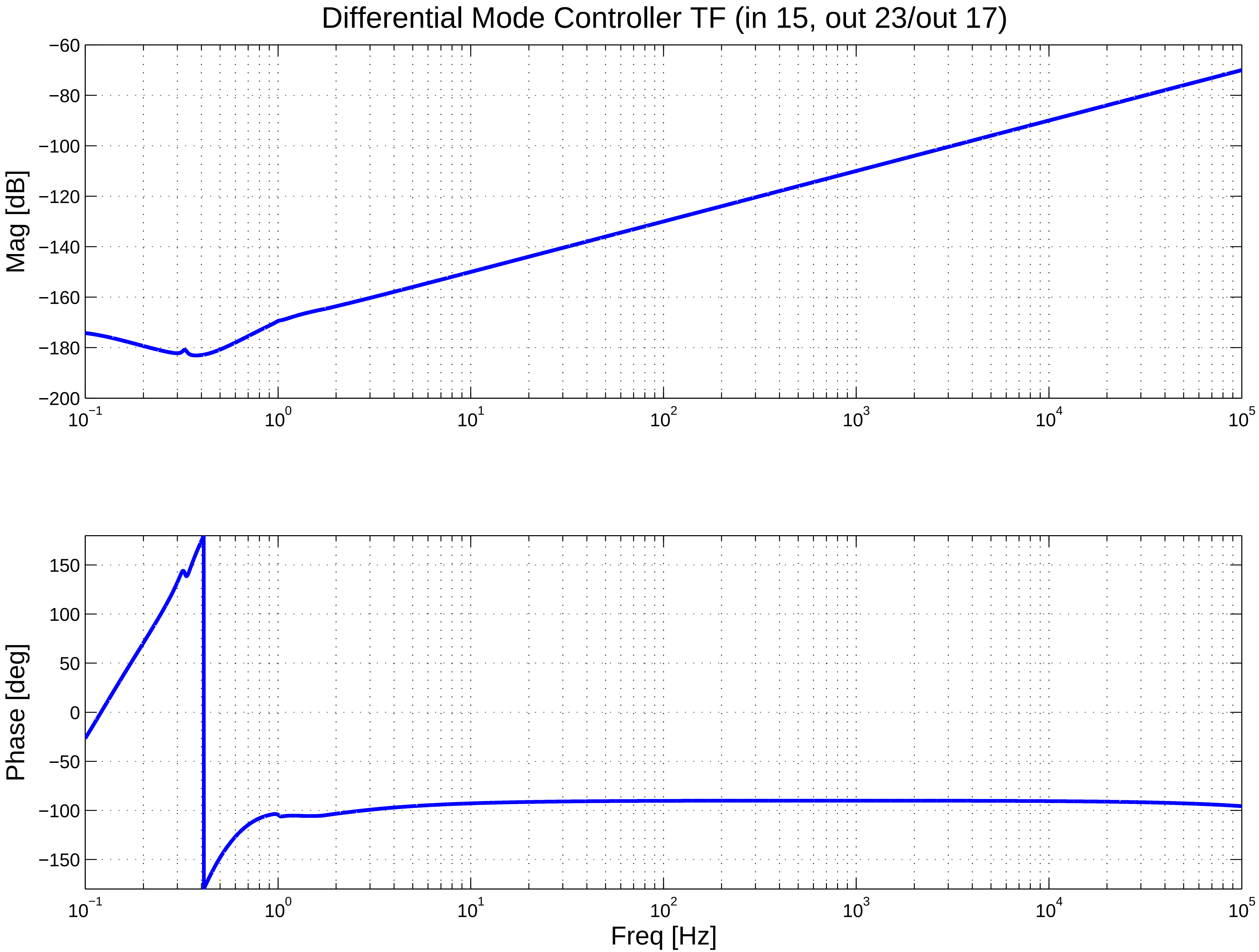


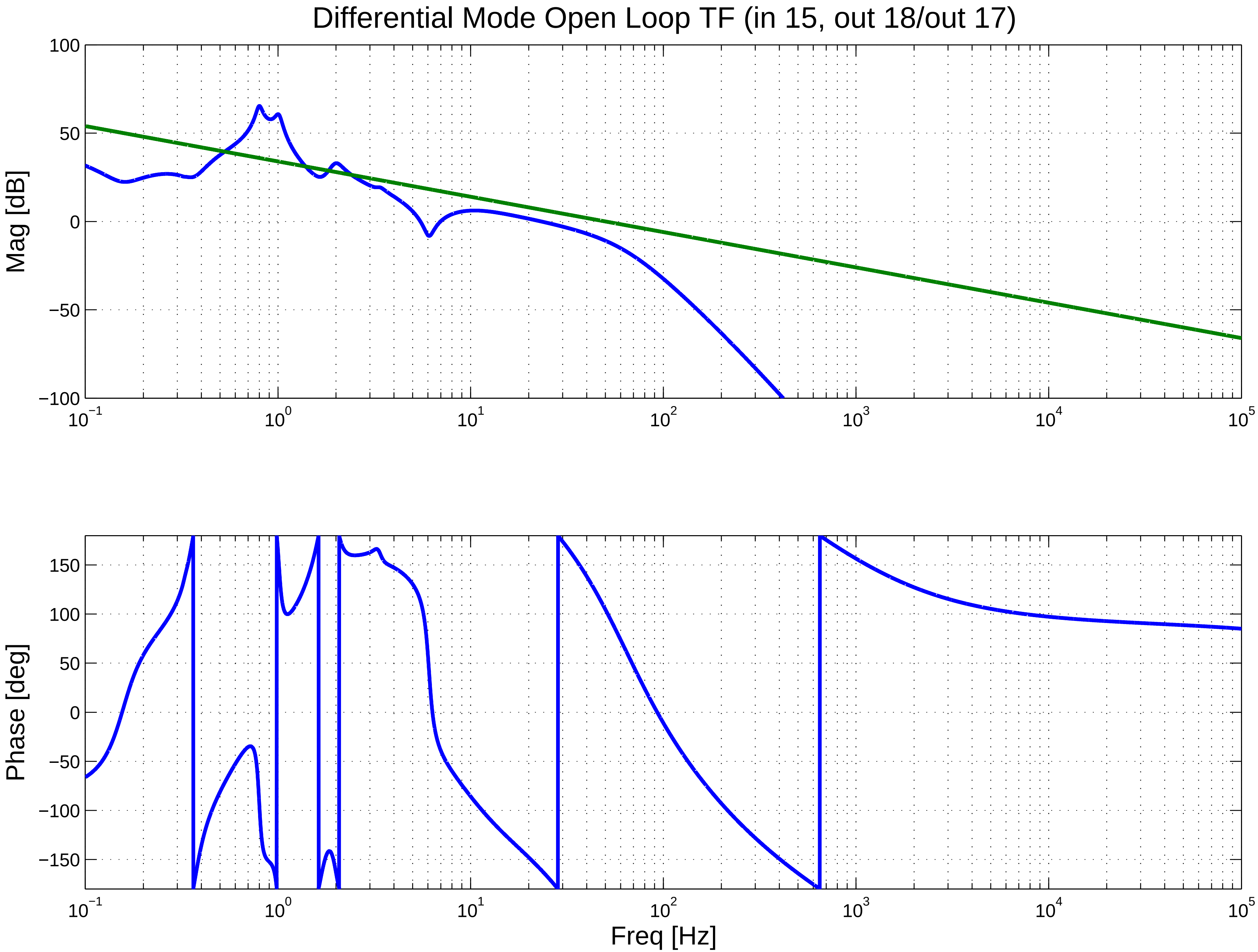
Common Mode Controller TF (in 11, out 12/ out 11)
Mag [dB]
180
170
160
150
140
130
120
Phase [deg]
150
100
50
0
−50
−100
−150
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Freq [Hz]

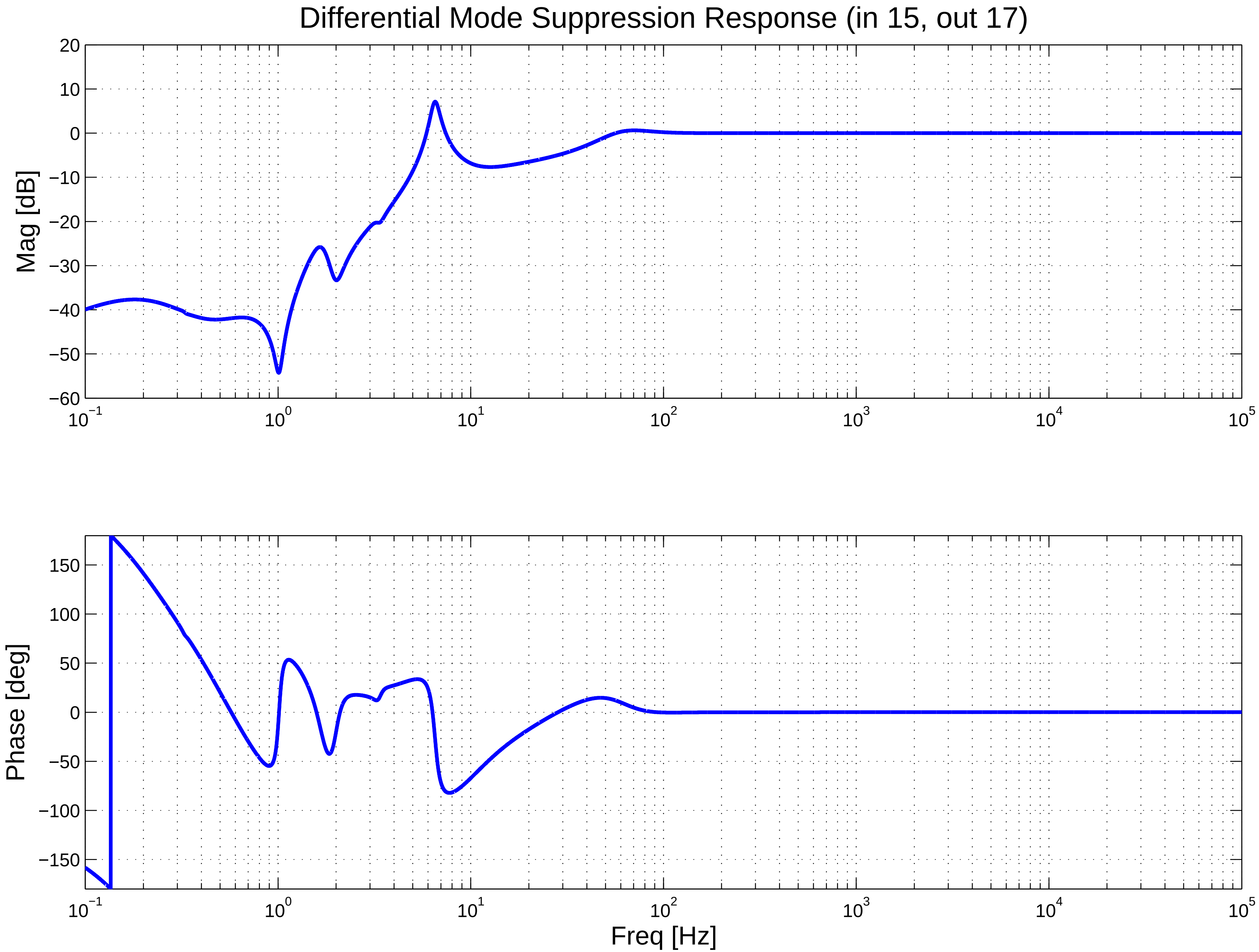

Common Mode Open Loop TF (in 11, out 13/out 11)
Mag [dB]
100
50
0
−50
−100
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]

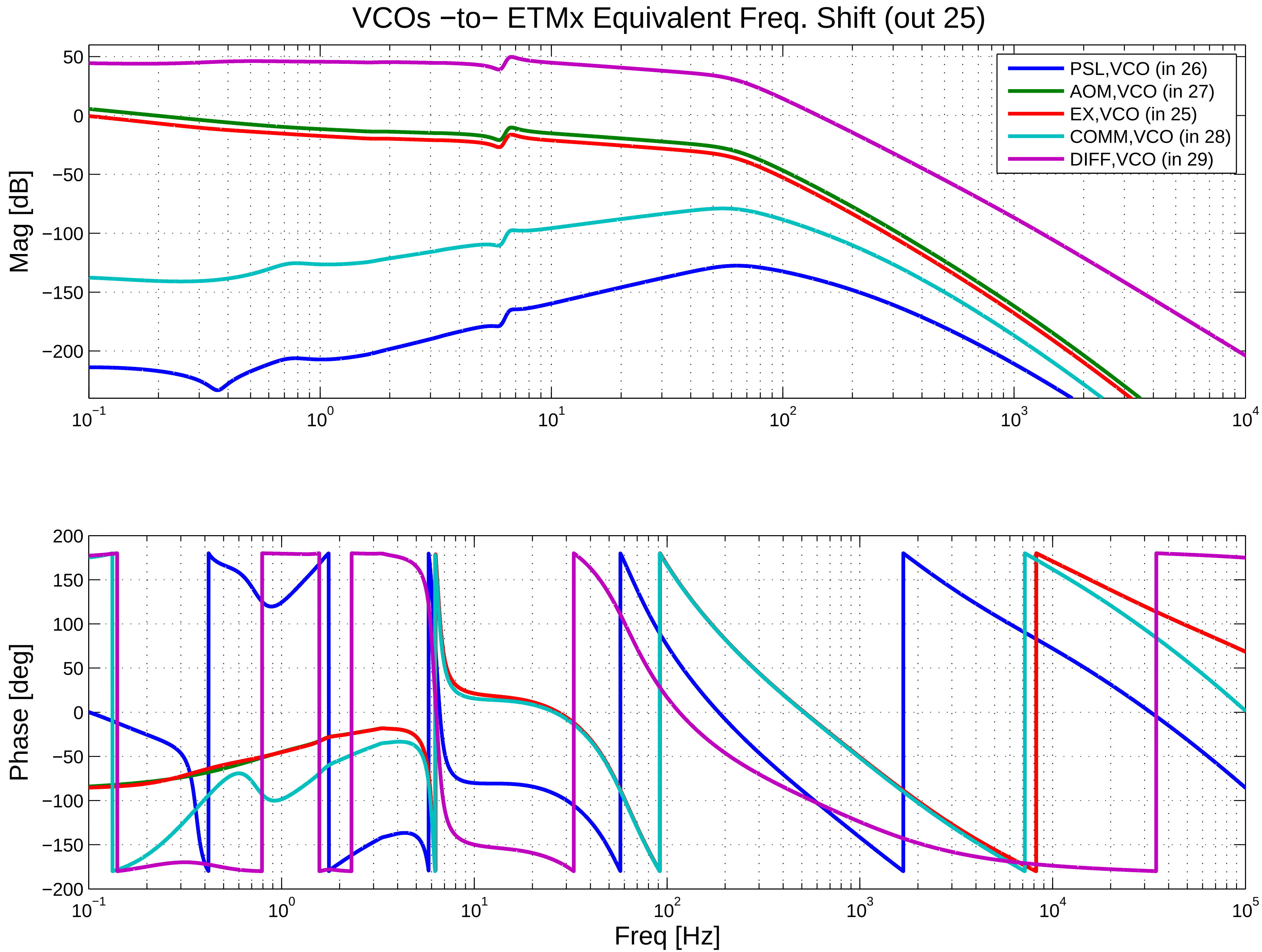

Common Mode Suppression Response (in 11, out 11)
Mag [dB]
0
−20
−40
−60
−80
−100
−120
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
80
60
40
20
0
−20
−40
−60
−80
Freq [Hz]

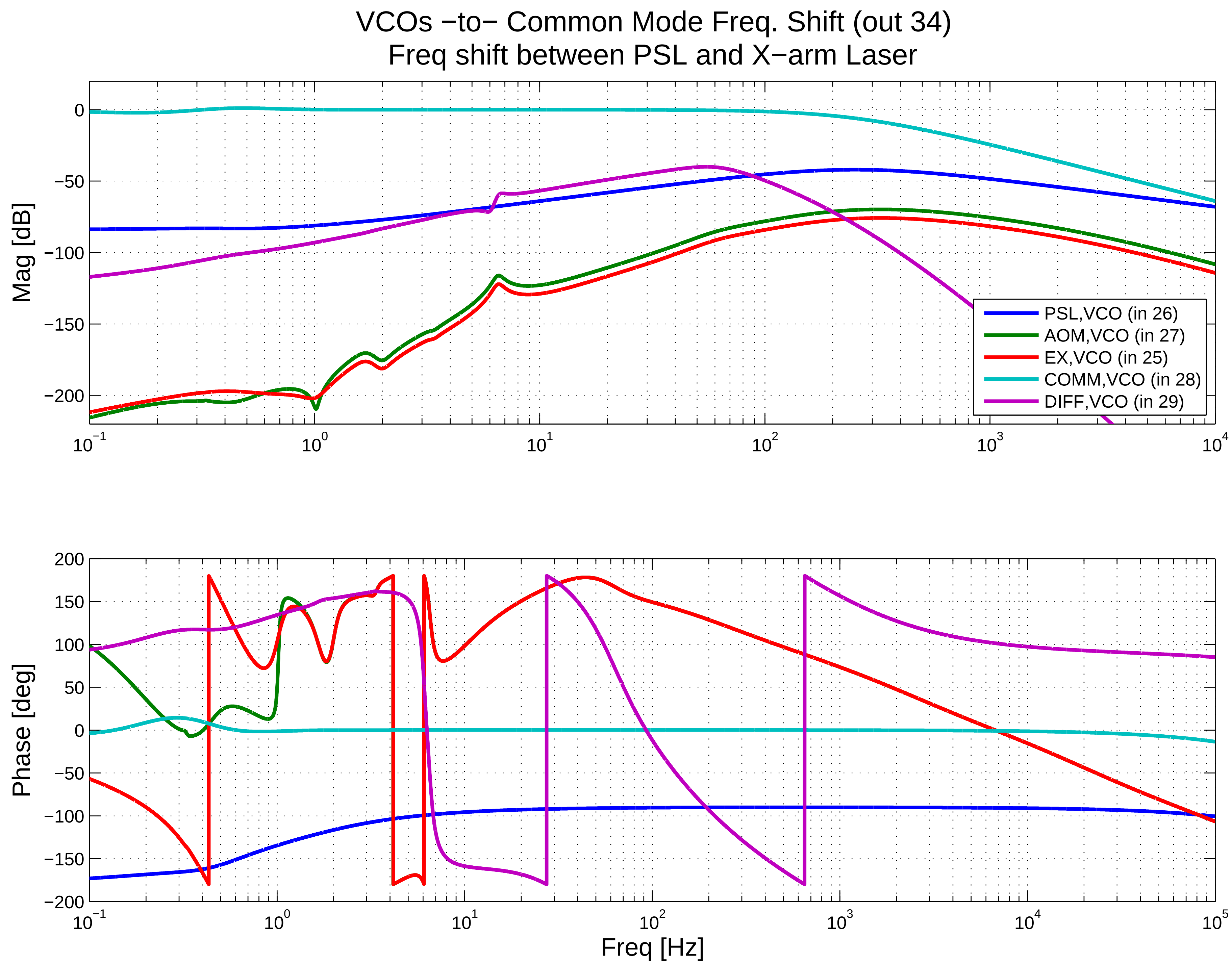

Diff Servo Ouput −to− Diff Mode Error Signal TF (in 20, out 14/out 23)
Mag [dB]
250
200
150
100
50
0
−50
Phase [deg]
150
100
50
0
−50
−100
−150
10^−1
10^0
10^1
10^2
10^3
10^4
10^5
Freq [Hz]

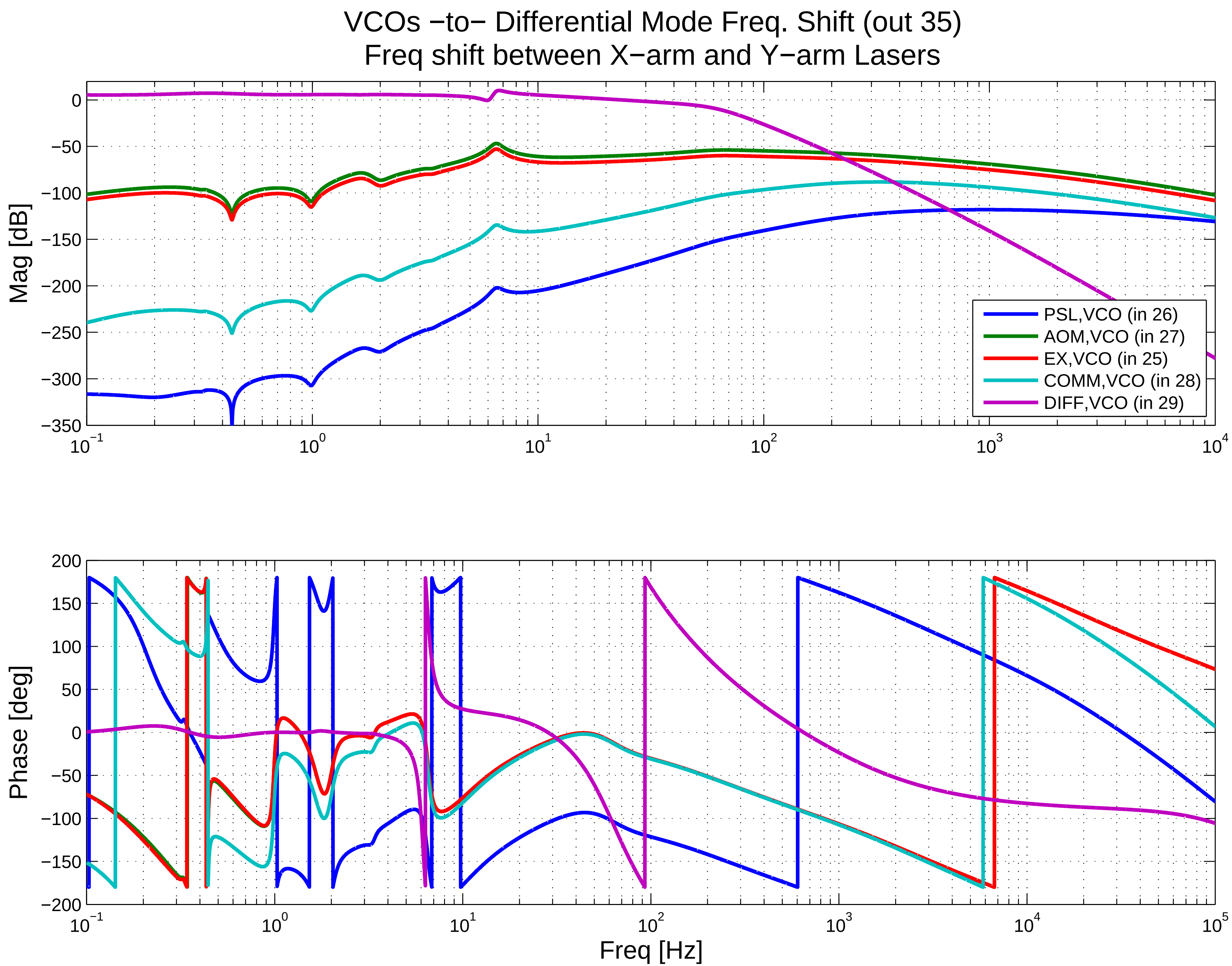

Differential Mode Controller TF (in 15, out 23/out 17)
Mag [dB]
−60
−80
−100
−120
−140
−160
−180
−200
10^−1
10^0
10^1
10^2
10^3
10^4
10^5
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]


Differential Mode Open Loop TF (in 15, out 18/out 17)
Mag [dB]
100
50
0
−50
−100
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
Freq [Hz]

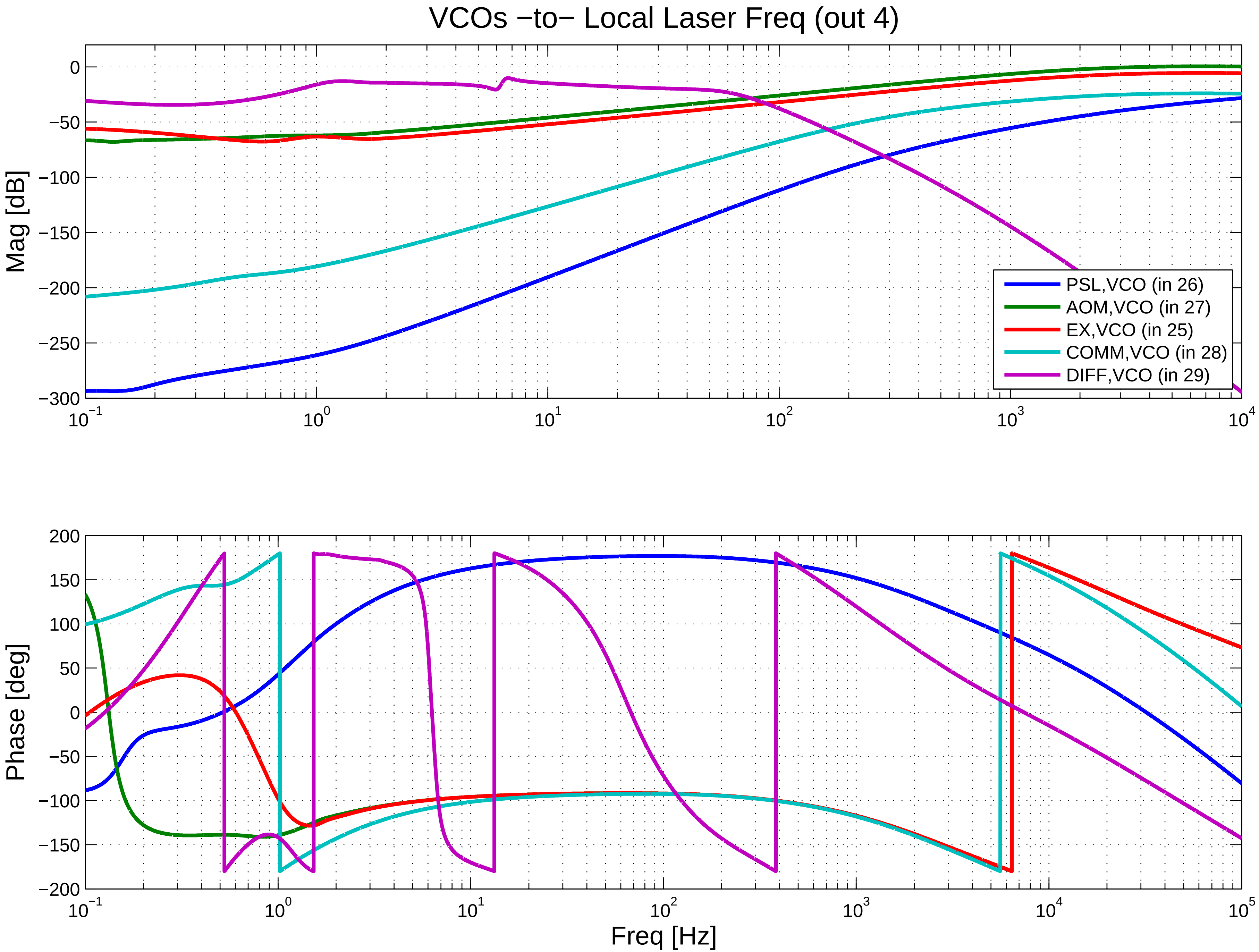

Differential Mode Suppression Response (in 15, out 17)
Mag [dB]
20
10
0
−10
−20
−30
−40
−50
−60
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
150
100
50
0
−50
−100
−150
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Freq [Hz]


VCOs −to− ETMx Equivalent Freq. Shift (out 25)
Mag [dB]
PSL,VCO (in 26)
AOM,VCO (in 27)
EX,VCO (in 25)
COMM,VCO (in 28)
DIFF,VCO (in 29)
Phase [deg]
Freq [Hz]

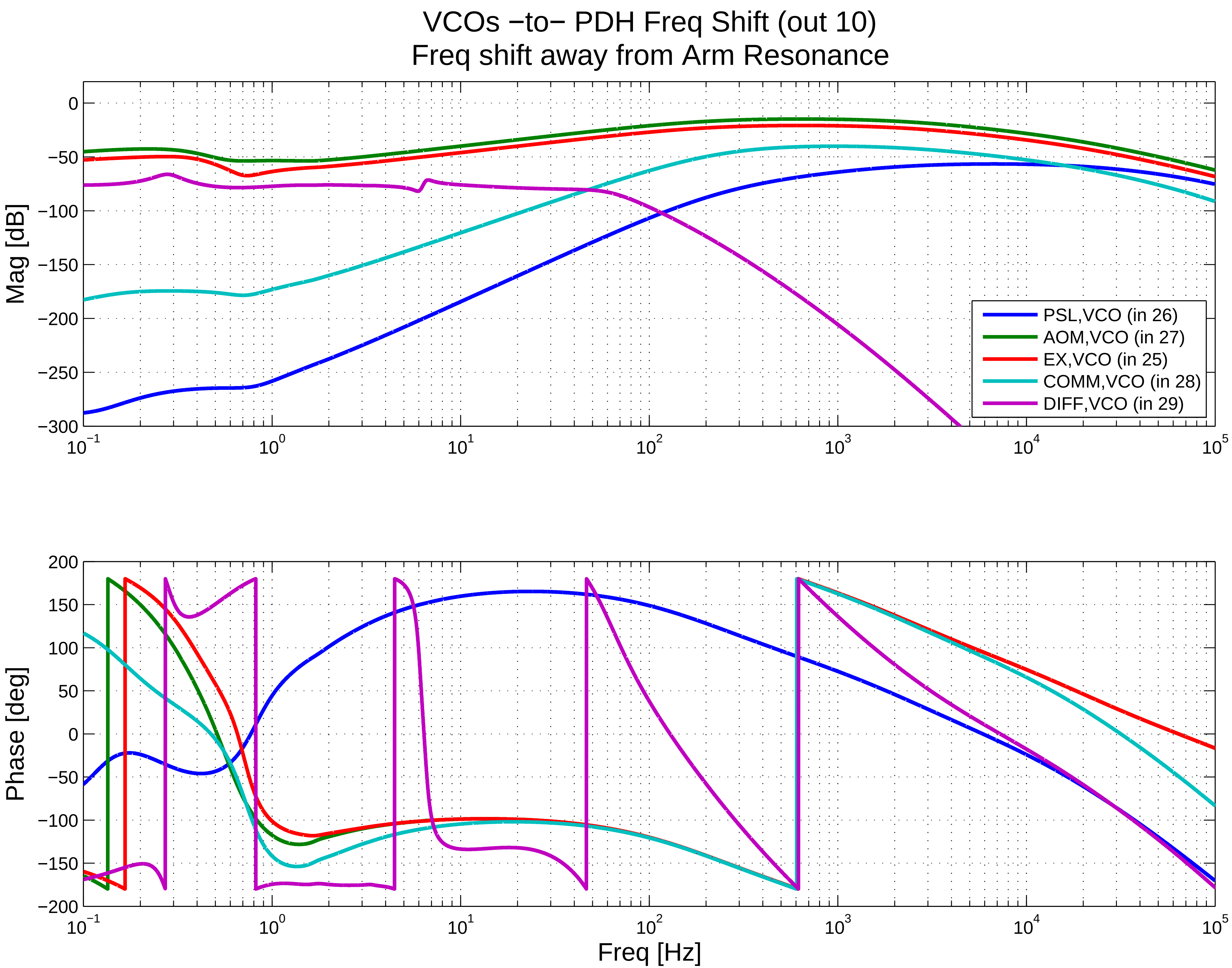

VCOs −to− Common Mode Freq. Shift (out 34)
Freq shift between PSL and X−arm Laser
Mag [dB]
0
−50
−100
−150
−200
10^−1
10^0
10^1
10^2
10^3
10^4
PSL,VCO (in 26)
AOM,VCO (in 27)
EX,VCO (in 25)
COMM,VCO (in 28)
DIFF,VCO (in 29)
Phase [deg]
200
150
100
50
0
−50
−100
−150
−200
10^−1
10^0
10^1
10^2
10^3
10^4
10^5
Freq [Hz]



VCOs −to− Differential Mode Freq. Shift (out 35)
Freq shift between X−arm and Y−arm Lasers
Mag [dB]
0
−50
−100
−150
−200
−250
−300
−350
PSL,VCO (in 26)
AOM,VCO (in 27)
EX,VCO (in 25)
COMM,VCO (in 28)
DIFF,VCO (in 29)
Phase [deg]
200
150
100
50
−50
−150
−200
Freq [Hz]



VCOs −to− Local Laser Freq (out 4)
Mag [dB]
0
−50
−100
−150
−200
−250
−300
PSL,VCO (in 26)
AOM,VCO (in 27)
EX,VCO (in 25)
COMM,VCO (in 28)
DIFF,VCO (in 29)
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
Phase [deg]
200
150
100
50
0
−50
−100
−150
−200
10^{5}
Freq [Hz]



VCOs −to− PDH Freq Shift (out 10)
Freq shift away from Arm Resonance
Mag [dB]
0
−50
−100
−150
−200
−250
−300
PSL,VCO (in 26)
AOM,VCO (in 27)
EX,VCO (in 25)
COMM,VCO (in 28)
DIFF,VCO (in 29)
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
Phase [deg]
200
150
100
50
0
−50
−100
−150
−200
Freq [Hz]

```
% ALS Locking Strategy
%
% Daniel Sigg's idea using 4 VCO's
%
% Needs the ALS_freq3v3.mdl Simulink model
%
% BS - 10 May 2010
%
% 24 June 2010 - modified the titles to reflect the G = b/a, e.g. (in 18, out 22/out 20)
% 13 July 2010 - Added more input/output points to get Hz/Hz transfer
%                functions for VCO noise performance requirements
%

clear all;

% Constants
c = 299792458; % [m/s]

lambda_IR = 1064e-9;
lambda_GRN = 532e-9;

f = logspace(-1, 6, 1e3);

%% Engaging Servo's
FSSengaged = 1;
TTFSSengaged = 1;
PDHengaged = 0;
COMMengaged = 0;
DIFFengaged = 0;

%% Setting up the PSL section
%

% Ref Cav Transmission TF
RefCavFSR = c / (2 * 0.2); % Ref Cav length 20cm?
RefCavFIN = 5000; % Ref Cav Finesse
RefCavPole = 2*pi * RefCavFSR / (2* RefCavFIN); % Ref Cav Pole frequency, 470kHz

% PSL FSS
PSLfrequencyActuator = 1;     % This is just a gain for the FSS feedback to the PSL
(Temp, PZT and Pockell)
zzz = [RefCavPole/2/pi];
ppp = [1];
kkk = 10^( 17 /20);
FSS_Servo = zpk(-2*pi*zzz, -2*pi*ppp, -kkk);

% LowNoiseVCO -> FSS low noise VCO driver TF, [Hz/V]
zzz = [];
ppp = 2e6;                   % Range of the VCO
kkk = ppp / 20;              % VCO Full tuning range (2 MHz) / VCO Input voltage range
(+/-20V)
LowNoiseVCO_psl = zpk(-2*pi*zzz, -2*pi*ppp, kkk);


% AOM Driver at fiber launch
AOM_Driver = 1;

%% Setting up the X-End station
%

FiberPhaseNoise = 100;   % flat fiber induced phase noise, 100 Hz/rtHz
freqnoiseNPRO = abs(1e4 ./ (1 + i.*f/1)); % Freerunning NPRO, 100 Hz/rtHz at 100 Hz

% Arm Cav Transmission TF
L_arm = 3995;
ArmCavFSR = c / (2 * L_arm); % Ref Cav length 20cm?
```

```
ArmCavFIN = 100; % Ref Cav Finesse
ArmCavPole = 2*pi* ArmCavFSR / (2* ArmCavFIN); % Arm Cav Pole frequency, ~1178 Hz

% LowNoiseVCO,EX -> TTFSS low noise VCO driver TF, used to demodulate the
% heterodyne signal in the end-station
zzz = [];
ppp = 2e6;
kkk = ppp / 20;             % VCO Full tuning range (2 MHz) / VCO Input voltage range
(+/-20V)
LowNoiseVCO_local = zpk(-2*pi*zzz, -2*pi*ppp, kkk);

% End-Station laser feedback
uPZT_local = 3e6;   % PZT Volts to IR Frequency conversion, 3 MHz/V

% TTFSS Servo -> TTFSS Locking Servo, lock the laser to the heterodyne beatnote.
zzz = [ 0 1e3];%[0 0 10 RefCavPole/2/pi];
ppp = [1 1 2e5];            % limited by the feedback to the laser PZT?
kkk = 10^(79/20);
TTFSS_Servo = zpk(-2*pi*zzz, -2*pi*ppp, kkk);

% PDH Servo -> PDH locking servo to lock the laser frequency to the arm cavity
% add notch at 1 Hz, Q=10, depth 30 dB (using foton:)

%- When DIFF and COMM are not engaged
% zzz = [1 300 0.003+i*0.999949 0.003-i*0.999949];
% ppp = [1.001+i*0.994872 1.001-i*0.994872 1e5 1e5];
% kkk = 100000000000000;%6000000;

zzz = [200 ];
ppp = [1];
kkk = 10^(62/20);
PDH_Servo = zpk(-2*pi*zzz, -2*pi*ppp, kkk);

%% Setting up the Vertex ALS Demodulation
%

% LowNoiseVCO,Comm -> Vertex ALS Common Mode low noise VCO driver TF
zzz = [];
ppp = 2e6;                  % Frequency range, Hz
kkk = ppp / 20;             % VCO Full tuning range (2 MHz) / VCO Input voltage range
(+/-20V)
LowNoiseVCO_comm = zpk(-2*pi*zzz, -2*pi*ppp, kkk);

% LowNoiseVCO,Diff -> Vertex ALS Differential Mode low noise VCO driver TF
LowNoiseVCO_diff = zpk(-2*pi*zzz, -2*pi*ppp, kkk);

%% Common Mode Servo -> Common Mode locking servo to lock the PSL frequency
%% to the common mode arm cavity length fluctuations
zzz = [0];
ppp = [1];
kkk = 10^( 126 /20);
CommonMode_Servo = zpk(-2*pi*zzz, -2*pi*ppp, -kkk);

%% Differential Signal Feedback to the ETM Quads
nu_IR = c / lambda_IR;

% Differential Mode Servo
zzz = [0.5];
ppp = [1e6];
kkk = 10^( -50 /20);
DifferentialMode_Servo = zpk(-2*pi*zzz, -2*pi*ppp, -kkk);

%% Setting up the Quad Feedback and Control Block
global pend

% Angular radiation pressure torque coefficients
```

```
k_major = 0;
k_minor = 0;
k_ospring = 0;

gLP = 0;
ServoTM = 0;
ServoPM = 0;
ServoUIM = 0;
ServoTOP = 0;

damper = 1; % ECD
% damper = 2; % GEO Damping
% damper = 3; % Damping with fancy LPF
% damper = 4; % no damping

%************************************
ssmake4pv2eMB2; % better blade modeling from MATHEMATICA, Mark Barton
%************************************
localdamp;

if ~exist('k_ospring')
  k_ospring = 0;
else
  % MEVANS
  warning('NOT including optical spring.');
  k_ospring = 0;
end

% Run the Quad Servo Script
PDHservo_All_2010_05_21_11_00_19

% Set the Signal Path Switches
gLP = 0;    % Keep the loop open to make it run within the overall simulation
gTM = 1;    % engage the TM feedback
gPM = 1;    % engage the PM feedback
gUIM = 0;   % engage the UIM feedback
gTOP = 0;   % engage the TOP feedback

%% Implementing the Simulink Model
%
%modelname = 'ALS_freq3v3';
modelname = 'ALS_freq3v4';
%
[AAA,BBB,CCC,DDD] = linmod2(modelname); % linearise the Simulink model
[rw,cl] = find(AAA == Inf); AAA(rw,cl) = 1e20;
[rw,cl] = find(AAA == -Inf); AAA(rw,cl) = -1e20;
[rw,cl] = find(BBB == Inf); BBB(rw,cl) = 1e20;
[rw,cl] = find(BBB == -Inf); BBB(rw,cl) = -1e20;
[rw,cl] = find(CCC == Inf); CCC(rw,cl) = 1e20;
[rw,cl] = find(CCC == -Inf); CCC(rw,cl) = -1e20;
[rw,cl] = find(DDD == Inf); DDD(rw,cl) = 1e20;
[rw,cl] = find(DDD == -Inf); DDD(rw,cl) = -1e20;

SYS = ss(AAA,BBB,CCC,DDD);              % ceates a state-space model of the Simulink
model
                                        % TF = SYS(output, input)

%% Do some test plotting
% a = SYS(27,25);
a = SYS(4,25);
b = SYS(31,4);
figure(99)
% mybodesys(a,f);
%title('VCO\_EX -to- f\_IR\_ex');
```

```
%% Print the Simulink model with all its colours, that works only in
%% Windows!
% set_param(modelname, 'ShowPageBoundaries', 'on');
% print(['-s' modelname], '-dpdf', [modelname '.pdf']); % print the simulink model with
its colors...

%% Obtaining the transfer functions
save_figure = 0;    % controls is the figures are save as .pdf or not

save_figure_all = 0;
save_figure_dir = 'sim/';

if save_figure_all
    FSS=1;
    TTFSS=1;
    PDH=1;
    COMM=1;
    DIFF=1;
    save_figure = 1;
    save_figure_dir = 'sim/';
else
    FSS = 0;    % Plots the FSS loops of th PSL servo
    if FSSengaged
        FSS = 1;
        save_figure_dir = 'ttfss/';
    end
    TTFSS = 0;  % Plots the TTFSS loops in the end-station
    if TTFSSengaged
        TTFSS = 1;
        save_figure_dir = 'ttfss/';
    end
    PDH = 0;    % plots the PDH loops in the end-station
    if PDHengaged
        PDH = 1;
        save_figure_dir = 'pdh/';
    end
    COMM = 0;
    if COMMengaged
        COMM = 1;
        save_figure_dir = 'comm/';
    end
    DIFF = 0;
    if DIFFengaged
        DIFF = 1;
        save_figure_dir = 'diff/';
    end
end         % end save_figure

if FSS
%% FSS Feedback of the laser to the Reference Cavity
    %
%%  % PSL laser -to- FSS error signal
    hdl= figure(101)
    a = SYS(20,18);
    b = SYS(22,18);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('PSL -to- FSS error signal TF (in 18, out 22/out 20)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
```

```
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
        % print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), ' ',get(tt,'string'),
'.pdf']);
    end


%%  % FSS Open Loop response
    hdl= figure(102)
    a = SYS(21,19);
    b = SYS(20,19);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('FSS Controller TF (in 19, out 20/out 21)','FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end


%%  % FSS Close Loop Response Response
    hdl= figure(103)
    a = SYS(21,19);
    b = SYS(22,19);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('FSS Open Loop TF (in 19, out 22/out 21)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

%%  % FSS Supression Response
    hdl= figure(104)
```

```
    G = mybodesys(SYS(21,19),f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('FSS Suppression Response (in 19, out 21)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

end         % end FSS



if TTFSS
%% TTFSS Feedback of the laser to the Heterodyen Signal
    %
    % Local laser to TTFSS error signal
    hdl= figure(1)
    a = SYS(5,8);
    b = SYS(3,8);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('Local laser -to- TTFSS error signal TF (in 8, out 3/out 5)', 'FontSize',
16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end


    % TTFSS Open Loop response
    hdl= figure(2)
    a = SYS(6,7);
    b = SYS(5,7);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('TTFSS Controller TF (in 7, out 5/out 6)','FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
```

```
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end


    % TTFSS Close Loop Response Response
    hdl= figure(3)
    a = SYS(6,7);
    b = SYS(3,7);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('TTFSS Open Loop TF (in 7, out 3/ out 6)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

    % TTFSS Supression Response
    hdl= figure(4)
    G = mybodesys(SYS(6,7),f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('TTFSS Suppression Response (in 7, out 6)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

end         % end TTFSS

if PDH
```

```
%% PDH Feedback of the laser frequency to the arm cavity, via the VCO,EX
    %
    % Local VCO,EX to PDH error signal
    hdl = figure(5)
    a = SYS(8,9);
    b = SYS(10,9);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('VCO,EX -to- PDH Error Signal TF (in 9, out 10/out 8)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%     set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end


    % PDH Controller response
    hdl= figure(6)
    a = SYS(9,10);
    b = SYS(8,10);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('PDH Controller TF (in 10, out 8/out 9)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%     set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end


    % PDH Close Loop Response Response
    hdl = figure(7)
    a = SYS(9,10);
    b = SYS(10,10);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('PDH Open Loop TF (in 10, out 10/out 9)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
```

```
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

    % PDH Supression Response
    hdl = figure(8)
    G = mybodesys(SYS(9,10),f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('PDH Suppression Response (in 10, out 9)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

end         % end PDH

if COMM
%% Common Mode Feedback of the PSL frequency to the common mode arm cavity
%% length fluctuations, via the VCO,C
    %
    % Vertex VCO,PSL to Common Mode error signal
    hdl = figure(9)
    a = SYS(12,12);
    b = SYS(13,12);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('VCO,PSL -to- Common Mode Error Signal TF (in 12, out 13/out 13)',
'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
```

```
    end

    % Common Mode Servo Open Loop response
    hdl= figure(10)
    a = SYS(11,11);
    b = SYS(12,11);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('Common Mode Controller TF (in 11, out 12/ out 11)', 'FontSize',16);
    ylabel('Mag [dB]', 'FontSize', 14)
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end


    % Common Mode Close Loop Response
    hdl = figure(11)
    a = SYS(11,11);
    b = SYS(13,11);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('Common Mode Open Loop TF (in 11, out 13/out 11)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end
%%
    % Common Mode Supression Response
    hdl = figure(12)
    G = mybodesys(SYS(11,11),f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('Common Mode Suppression Response (in 11, out 11)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
```

```
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end
end         % end COMM

if DIFF
%% Differential Mode Feedback to both the ETMs (out of phase). This
%% requires the Quad response and its servo, for now I have a single
%% pendulum replacing the Quad...
    %
    % Diff Mode Servo input to Differential Mode error signal
    hdl = figure(13)
    a = SYS(23,20);
    b = SYS(14,20);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('Diff Servo Ouput -to- Diff Mode Error Signal TF (in 20, out 14/out 23)',
'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
%    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    axis([min(f) max(f) -50 250])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end


    % Diff Mode Controller response
    hdl= figure(14)
    a = SYS(17,15);
    b = SYS(23,15);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('Differential Mode Controller TF (in 15, out 23/out 17)', 'FontSize',16);
    ylabel('Mag [dB]', 'FontSize', 14)
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
```

```
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

    % Differentail Mode Close Loop Response
    hdl = figure(15)
    a = SYS(17,15);
    b = SYS(18,15);
    G = mybodesys(b/a,f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), f, 20*log10(50./f), 'LineWidth', 2)
    tt= title('Differential Mode Open Loop TF (in 15, out 18/out 17)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
%    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    axis([min(f) max(f) -100 100])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

    % Differential Mode Supression Response
    hdl = figure(16)
    G = mybodesys(SYS(17,15),f);
    %
    subplot(211)
    semilogx(f,20*log10(abs(G)), 'LineWidth', 2)
    tt= title('Differential Mode Suppression Response (in 15, out 17)', 'FontSize', 16);
    ylabel('Mag [dB]', 'FontSize', 14);
    axis([min(f) max(f) floor(min(log10(abs(G))))*20 ceil(max(log10(abs(G))))*20])
    grid on
    subplot(212)
    semilogx(f,angle(G)*180/pi, 'LineWidth', 2)
    ylabel('Phase [deg]', 'FontSize', 14);
    xlabel('Freq [Hz]', 'FontSize', 14);
    axis([min(f) max(f) 90*floor(min(angle(G)*180/pi)/90) 90*ceil(max(angle(G)*180/pi)/
90)])
%    set(gca,'YTick',[-1800:90:1800])
    grid on
    if save_figure == 1
        orient(hdl, 'landscape');
        print('-dpdf', [save_figure_dir ,num2str(hdl,'%.3d'), '.pdf']);
    end

end         % end DIFF
```

```
% PDHdervo_All_2010_05_21_11_00_19.m
%
% TM and PM feedback servo settings
%
% J. Miller
%
%flatTOP = [0.4 0.4 1 1 2 2 3.4 3.4];

global w

% if resprun == 0
%     PDHresp;
% end

% clear ServoTM ServoPM ServoUIM ServoTOP S_TM S_PM S_UIM S_TOP;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Feedback Servo Response, Open-loop

% Set overall gain in the closed loop response
if gLP ~= 0
  gLP = 1;
end

gALLdB = 90;
gALL = 10^( gALLdB /20);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% TM Servo
% MEVANS - made zeros complex, added resgain, and increased UGF
fc = 60;

fpTM = 5;

zTM = [0.4 0.4 0.4];%1
pTM = [ 1e-2 5 1e3];
kTM = 5e4 * prod(pTM) / prod(zTM(2:end));

% Set the PM Servo ZPK
zzz = [-2*pi.*zTM];
ppp = [-2*pi.*pTM];
kkk = kTM;

setGainAtF = 1e-2;  % frequency in Hz
gainAtF = 1;     % gain at that frequency

% Set the TM Servo ZPK
zzz = [-2*pi.*zTM];
ppp = [-2*pi.*pTM];

ServoTest = zpk(zzz, ppp, 1);
kkk = gainAtF / abs(evalfr(ServoTest, 2i * pi * setGainAtF));

% Gain stage into the TM actuators
kdB = 0;  %135
TM_gain = 10^(kdB/20);

%preTM = 10^( 0 /20);

% Setting Simulink Blocks
ServoTM = zpk(zzz, ppp, kkk);
gTM = 0;
coTM = cutoff(fc, 4);

% Setting up complex TFs
%S_TM = ss2complex(ServoTM, w);
% cfilter = ss2complex(coTM, w);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PM Servo
% MEVANS - made zeros complex, added resgain, and increased UGF
fc = 10;

fpPM = 0.1;

zPM = [1 1 1];
pPM = [1e-2 3 1e3];%[ fpPM 1000 1000];

setGainAtF = 1e-2;  % frequency in Hz
gainAtF = 1;     % gain at that frequency

% Set the PM Servo ZPK
zzz = [-2*pi.*zPM];
ppp = [-2*pi.*pPM];

ServoTest = zpk(zzz, ppp, 1);
kkk = gainAtF / abs(evalfr(ServoTest, 2i * pi * setGainAtF));

% Gain stage into the PM actuators
kdB = 0;
PM_gain = 10^(kdB/20);

% Setting Simulink Blocks
ServoPM = zpk(zzz, ppp, kkk);
prePM = 5e2;
gPM = 0;
coPM = cutoff(fc, 4);

% Getting up complex TFs
%S_PM = ss2complex(ServoPM, w);
%cfilter = ss2complex(coPM, w);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% UIM Servo
fc = 3;

%gALL = 1000;
preUIM = 3e2; % gain prior the UIM servo

fpUIM = 0.01;

zUIM = [2  2 2];%[fpPM 2 2];
pUIM = [1e-2 1 1e3];%[fpUIM 1000 1000];

setGainAtF = 1e-2;  % frequency in Hz
gainAtF = 1;     % gain at that frequency

% Set the Simulink Blocks
zzz = [-2*pi.*zUIM];
ppp = [-2*pi.*pUIM];

ServoTest = zpk(zzz, ppp, 1);
kkk = gainAtF / abs(evalfr(ServoTest, 2i * pi * setGainAtF));

% Gain stage into the UIM actuators
kdB = 0;
UIM_gain = 10^(kdB/20);

% Setting Simulink Blocks
ServoUIM = zpk(zzz, ppp, kkk);

gUIM = 0;
```

```
coUIM = cutoff(fc, 4);

% Getting up complex TFs
%S_UIM = ss2complex(ServoUIM, w);
%cfilter = ss2complex(coUIM, w);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% TOP Servo
fc = 1;
fpTOP = 0.001;

%zTOP = [0.45*r1];
zTOP = [ 3.4 3.4];%2 2 [fpUIM 3.4 3.4];    % [Hz]
pTOP = [1e-2 1e3];%[fpTOP 2000 2000]; % [Hz]

setGainAtF = 1e-2;  % frequency in Hz
gainAtF = 1;     % gain at that frequency

% Set the Simulink Blocks
zzz = [-2*pi.*zTOP];
ppp = [-2*pi.*pTOP];

ServoTest = zpk(zzz, ppp, 1);
kkk = gainAtF / abs(evalfr(ServoTest, 2i * pi * setGainAtF));

% Gain stage into the TOP actuators
kdB = 0;
TOP_gain = 10^(kdB/20); % becomes gTOP

preTOP = 1e2;

% Setting Simulink Blocks
ServoTOP = zpk(zzz, ppp, kkk);
gTOP = 0;
coTOP = cutoff(fc, 4);

% Getting up complex TFs
%S_TOP = ss2complex(ServoTOP, w);
%cfilter = ss2complex(coTOP, w);
```