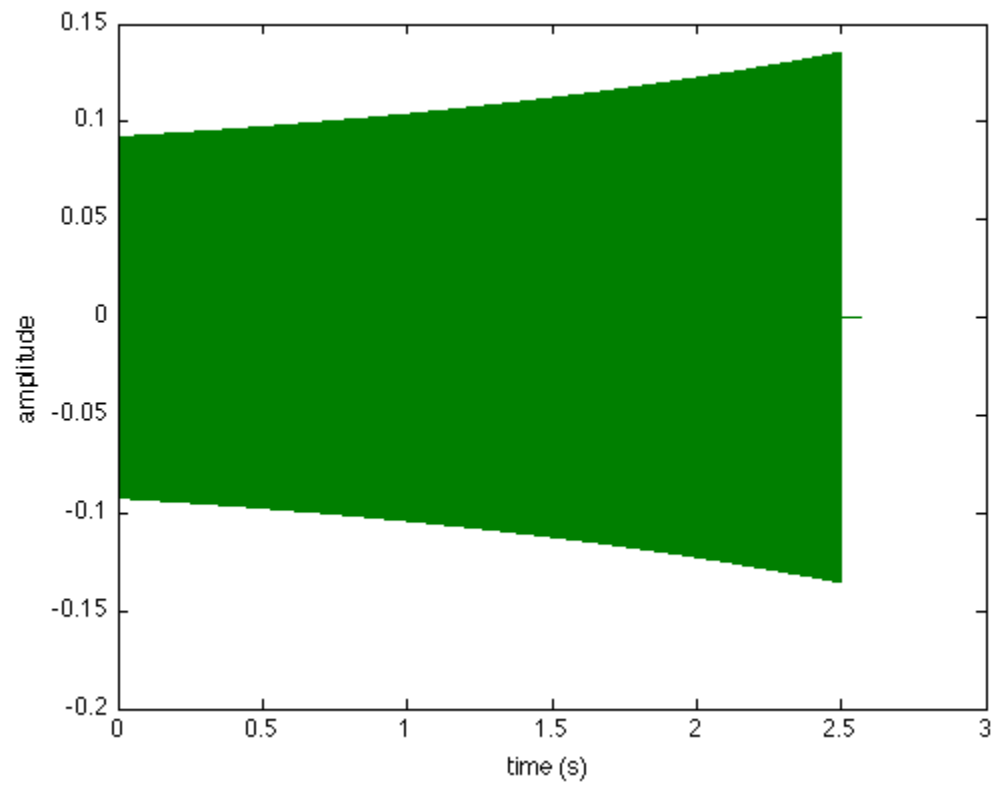```
% Problem 1
% Output plotted with compression ratio of 5

wavePlot('p1ratio5.wav');
```



*Published with MATLAB® 7.10*

```
% problem 2

r0 = 100000;
lambda_t=1;

% feedforward
input_level = (1:0.01:100);
rp = r0 .* (input_level/lambda_t).^(-0.75);
gain = 2*rp./(r0+rp);
ff_output = input_level.*gain;
plot(input_level,ff_output);
xlabel('\lambda (multipe of \lambda_T)');
ylabel('\lambda_\infty (multipe of \lambda_T)');
title('Static compression of feedforward architecture');
% semilogx(20*log(lambda)/log(10),ff_output);

%feedback
hold on;
out = (1: 0.01 : 100);
input_estimate = ((out + lambda_t^(-0.75)*out.^(1.75))./2);
plot(input_estimate, out, 'r');
axis([1 100 0 20]);
```
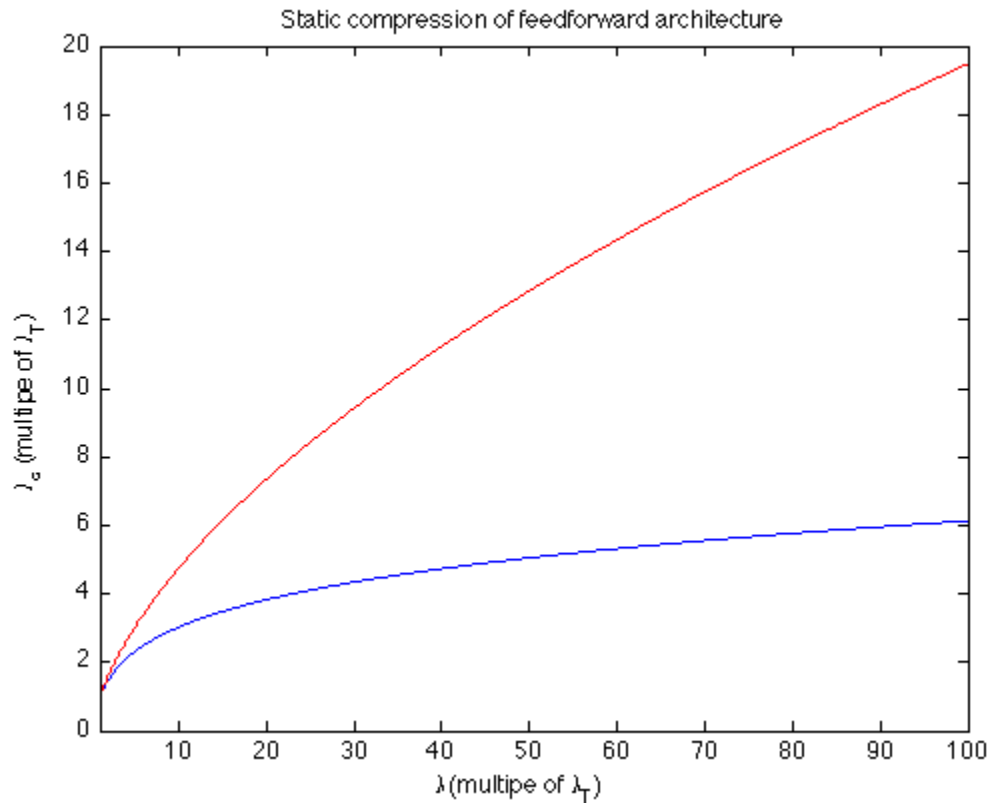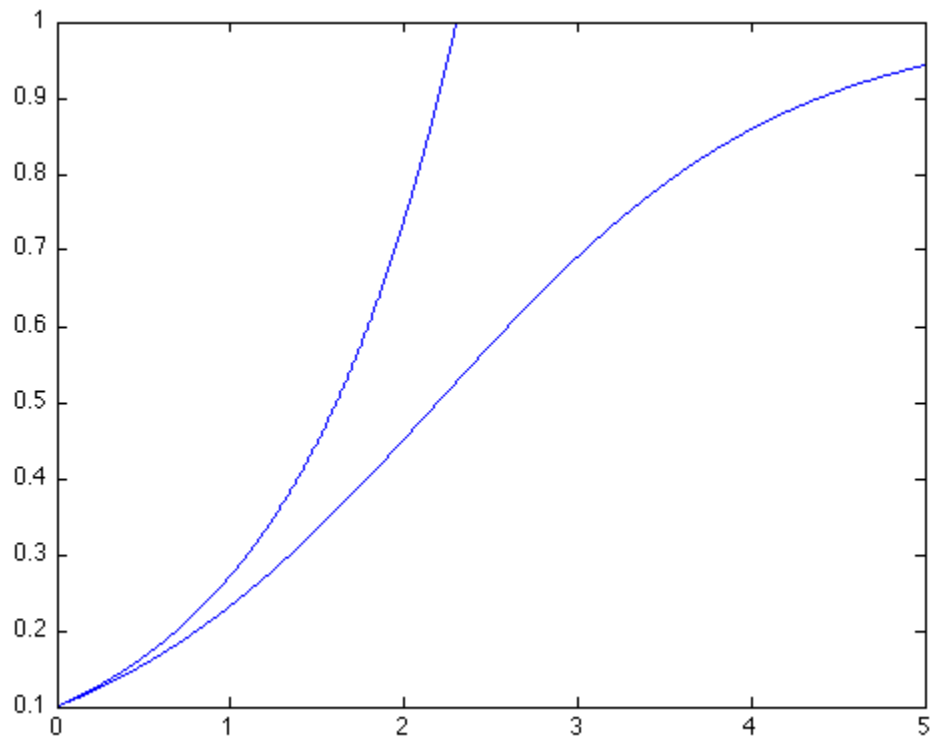


*Published with MATLAB® 7.10*

```matlab
% problem 3b

fs=44100;
t=0:1/fs:5;
threshold = 0.1;

% release-to-zero
level_estimate = 1*exp(-t);
gain = min(1,0.1./level_estimate);
plot(t,gain);

% release-to-threshold
hold on;
level_estimate = 0.9*exp(-t)+0.1;
gain = min(1,0.1./level_estimate);
plot(t,gain);

% release-to-threshold is more smooth since it never will have a
% discontinuity in the derivative of the gain
```
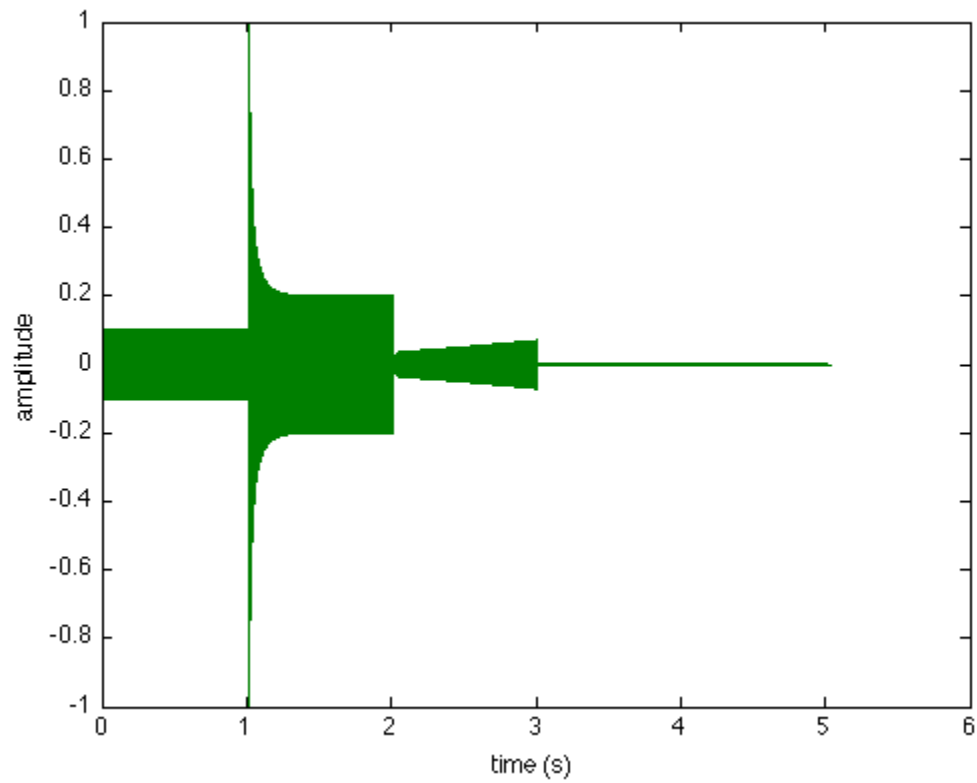

Published with MATLAB® 7.10

```
% problem 3c
% the compressor was modified and the output taken on the given specs.
% Since there were a lot more specs to decide, I went with release time of
% 50ms to see the effect of program dependence maximally
```

```
wavePlot('output.wav')
```


Published with MATLAB® 7.10

```matlab
% 4a) Threshold of compression = 0.12 (-42 dB)
%       Compression Ratio = 1.65


% 4b) pseudo code:

% feedforward
% compRatio = 1.65
% threshold = 0.12
% if (input > threshold)
% dBpsi_f = dB(input/threshold)/(compRatio)*(1-compRatio)
% psi_f = db2lin(dBpsi_f)

% feedback
% compRatio = 1.65
% threshold = 0.12
% if (output > threshold)
% dBpsi_b = dB(output/threshold)*(1-compRatio)
% psi_b = db2lin(dBpsi_b)

% 4c) Peak detection - compressor does not change behavior when dealing
% with low-frequency sine wave of amplitude 1 OR DC at amplitude 1

% 4d) Attack time = 17 ms
%       Release time = 400 ms

% 4e) pseudo code


% b0_r = 1 - exp( -1.0 / ( 5 * fs ) )
% b0_a = 1 - exp( -1.0 / ( 0.1 * fs ) )

% feedforward
% if ( input > levelEstimate )
%     levelEstimate += b0_a * ( input - levelEstimate );
% else
%     levelEstimate += b0_r * ( input - levelEstimate);

% feedback
% if ( output > levelEstimate )
%     levelEstimate += b0_a * ( output - levelEstimate );
% else
%     levelEstimate += b0_r * ( output - levelEstimate);
```

Published with MATLAB® 7.10