Lab 6: Impulse Responses

Ron Guglielmone

INTRODUCTION

Impulse response measurements were made in memorial church at Stanford University using AKG-414 large diaphragm condenser microphones, speakers of an unknown make, and a Zoom F8 field-recording unit. Balloon pops, sine sweeps, and golay codes were each used separately.

PROBLEM 1: PART A

Estimate the T60 decay time at 250 Hz, 1 kHz, and 4 kHz. Approximate the church as a box 25m x 40m x 10m. Use plywood for the ceiling and half the floor, glass for a tenth of the walls, and plywood for another tenth, and marble for everything else.

Absorption Coefficients, S						
material	*frequency*					
	125	250	500	1000	2000	4000
marble	0.01	0.01	0.01	0.01	0.02	0.02
brick	0.03	0.03	0.03	0.04	0.05	0.07
concrete block	0.36	0.44	0.31	0.24	0.39	0.25
plywood	0.28	0.22	0.17	0.22	0.10	0.11
cork	0.14	0.25	0.40	0.25	0.34	0.21
glass window	0.35	0.25	0.18	0.12	0.07	0.04
drapery	0.10	0.25	0.46	0.60	0.56	0.52
carpet	0.02	0.06	0.14	0.37	0.66	0.65
hardwood	0.15	0.11	0.10	0.07	0.06	0.07
grass	0.11	0.26	0.60	0.69	0.92	0.94

Figure 1, Sabine coefficients for various materials.

T60 for air (from page 144 in course reader): [250 Hz 1 kHz 4 kHz; 130 30 6.8].

For our solid surfaces, we use equation 1 to calculate the first T_{60} value.

$$T_{60}(\omega) \;=\; -2\ln(0.001)\frac{1}{gc\sum_i A_i/V + \alpha(\omega)} \quad \textit{(eqn 1)}$$

Then we use equation 2 to calculate the T_{60} values for air.

$$T_{60}(\omega) = \left[\frac{1}{T_{60\mathsf{surfaces}}(\omega)} + \frac{1}{T_{60\mathsf{air}}(\omega)}\right]^{-1} \quad \textit{(eqn 2)}$$

The following MATLAB script was written to solve for the T_{60} value (Figure 2).

```
% Ron Guglielmone
% MUSIC 424, CCRMA, Stanford University
% May 23, 2017
%
% Lab 6 - Problem 1 - Part A

% Constants:
c = 340.27; % speed of sound m/s
g = 1/4; % geometric constant
k = -2*log(0.001)*(1/(g*c));

% Sabine coefficients:
freqs = [250 1000 4000];
Swood = [0.22 0.22 0.11];
Sglass = [0.25 0.12 0.04];
Smarb = [0.01 0.01 0.02];

% Room dimensions:
length = 25; % meters
width = 40; % meters
height = 10; % meters
volume = length*width*height;

% Calculate surface areas:
ceilArea = length*width;
floorArea = ceilArea;
wallArea = length*height*2 + ...
            width*height*2;

% Calculate material areas:
woodArea = ceilArea + ...
           0.5*floorArea + ...
           0.1*wallArea;
glassArea = 0.1*wallArea;
marbleArea = 0.5*floorArea + ...
             0.8*wallArea;

% Pre-allocate:
t60Solids = [];

% Calculate t60 for surfaces:
for i = 1 : 3

    t60Solids(i) = k / ...
                  ((woodArea*Swood(i) +...
                  glassArea*Sglass(i) + ...
                  marbleArea*Smarb(i)) / ...
                  volume);
end

% Print output:
% freqs
% t60Solids

% Calculate total t60:
t60Air = [130 30 6.8];
totalT60 = ( (1./t60Solids) + (1./t60Air) ) .^-1
```

Figure 2, MATLAB code for problem 1, part A.

The estimated values were as follows in Figure 3.

250 Hz	1 kHz	4 kHz
3.8761 seconds	3.6600 seconds	3.5762 seconds

Figure 3, estimated T_{60} values for Memorial Church.

PROBLEM 1: PART B

If the church were half the size, we see the values change as in Figure 4.

250 Hz	1 kHz	4 kHz
1.9674 seconds	1.9489 seconds	2.4260 seconds

Figure 4, estimated T_{60} values for (1/2) sized Memorial Church.

PROBLEM 1: PART C

If the church floor were instead carpet, we see the values change as in Figure 5.

250 Hz	1 kHz	4 kHz
4.4618 seconds	2.3243 seconds	1.5629 seconds

Figure 5, estimated T_{60} values for carpeted Memorial Church.

PROBLEM 2: PART A

The following script implements a linear chirp from 0 to fs/2 (Figure 6).

```
function [ sweep ] = chrpLin( duration, fs )

% Ramps from 0 to fs/2 with constant frequency
% trajectory (linear).  Returns the signal.

L = ceil(fs*duration)+1;
n = 0:L-1;
t = n/fs;
sweep = chirp(t,0,duration,fs/2);

end
```

Figure 6, MATLAB code for linear chirp.

PROBLEM 2: PART B

The following script implements a logarithmic chirp from f_1 to f_2 (Figure 7).

```
function [ sweep ] = chrpLog( f0, f1, duration )

% Ramps from 0 to fs/2 with logarithmic frequency
% trajectory (equal time per octave).

L = ceil(fs*duration)+1;
n = 0:L-1;
t = n/fs;
sweep = chirp(t,f0,duration,f1, logarithmic);

end
```

Figure 7, MATLAB code for logarithmic chirp.

PROBLEM 2: PART C

The following script retrieves an impulse response by deconvolving a test signal with the output of an LTI system to that same signal (Figure 8).

```
function [ ir ] = chirp2ir( ss, rs )

% Takes a sine sweep and the response
% to the sweet and deconvolves them
% to retrieve the impulse response.

ir = ifft(fft(rs)./fft(ss));

end
```

Figure 8, deconvolution script.

PROBLEM 2: PART D

The following script extracts an RMS envelope from a signal (Figure 9).

```
function [ env ] = energyEnvelope( sig, fs, eta )

% Takes an input signal, sample rate, and smoothing
% time, and returns a running RMS smoothed
amplitude
% envelope of the signal.  Eta is in milliseconds.

[yupper,~] = envelope(sig,eta,'rms');

env = yupper;

end
```

Figure 9, RMS extraction script.

PROBLEM 3: PART A

The following script was used with the helper functions from Problem 2 to characterize the impulse response of the mystery system (Figure 10).

```
duration = 1/32;
fs = 44100;
sweep = chrpLin(duration,fs);
response = hmeasure(sweep)';
ir = chirp2ir(sweep,response);
subplot(3,1,1);
plot(ir);
hold on;
```

Figure 10, MATLAB script.

Several different impulse response measurements are plotted in Figure 11.

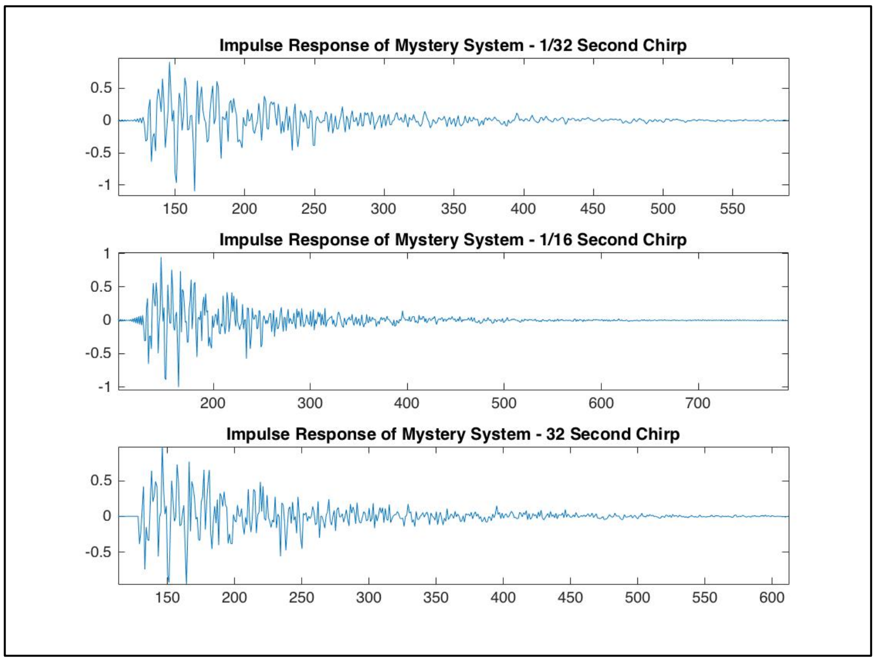

Figure 11, IR measurements from different length chirps.

PROBLEM 3: PART B

SNR estimate for 1/32 second chirp: 48.52 dB
SNR estimate for 1/16 second chirp: 53.37 dB
SNR estimate for 32 second chirp: 86 dB

The relationship seems to be exponential.

PROBLEM 4: PART A

The file 'ssxr_48_04000.wav' was chosen, and the left channel was selected to transform into an impulse response. The log-spectrogram of this IR is shown in Figure 12.

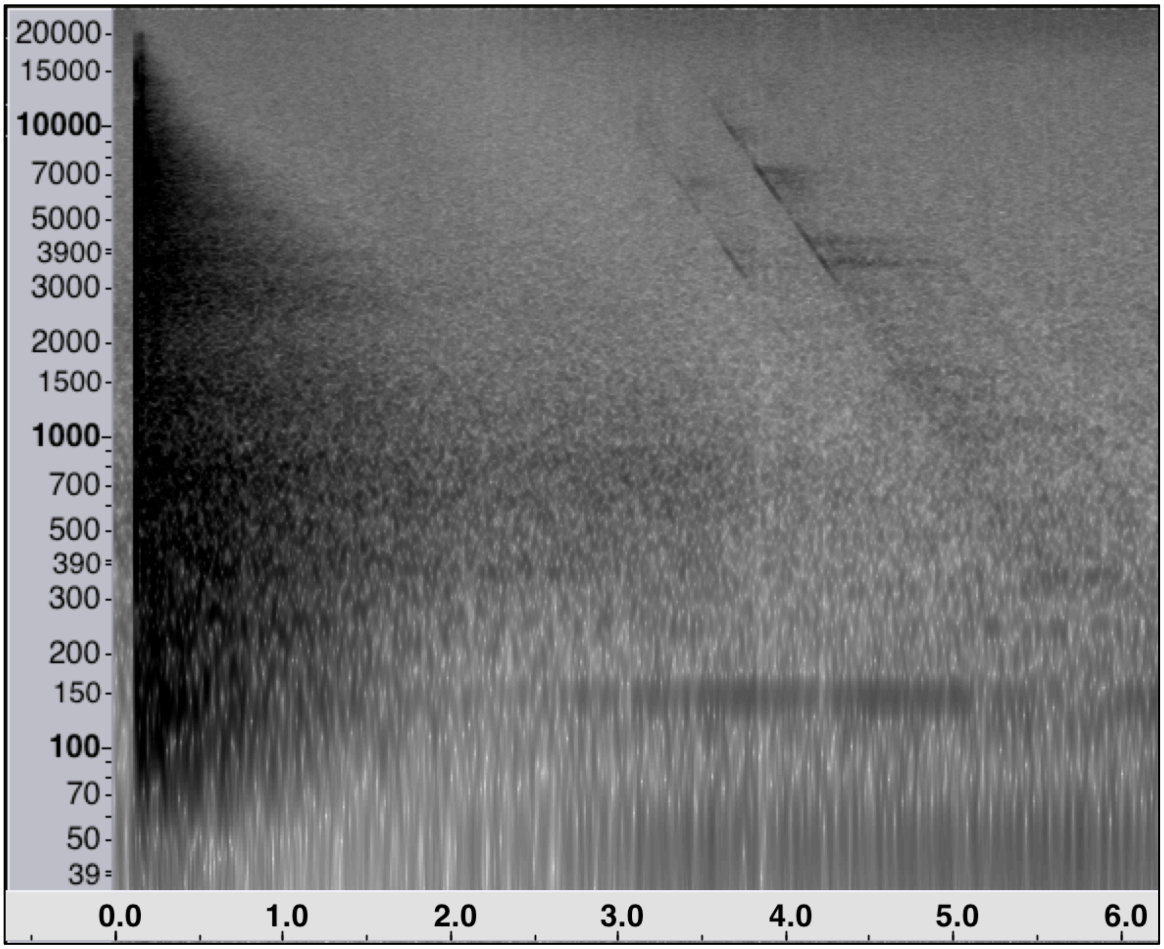

Figure 12, log spectrogram of church impulse response.

PROBLEM 4: PART B

For the selected signal, 'ssxr_40_04000.wav', the SNR seems to be about 57 dB.

PROBLEM 4: PART C

// TODO

PROBLEM 4: PART D

// TODO

PROBLEM 5

It definitely doesn't sound like the church, but it is certainly a reverb effect-- very cool! It sounds more metallic than the church, and there is some distortion because my impulse response is normalized to 0dB. Still, overall, I think the result is very exciting for a first experiment. I used the following MATLAB command to apply the filter:

```
Output = ifft(fft(sig).*fft(xs));
sound(Output,fs)
```

PROBLEM 6: PART A

Considering the following all-pass filter cascade:

$$G_n(z) = \left(\frac{\rho + z^{-1}}{1 + \rho z^{-1}}\right)^n$$

The following MATLAB script generates the impulse response $h_g(t)$.

```
% Given:
rho = 0.5;
n = 64;

% Filter coeff:
B = [rho, 1];
A = [1, rho];

% Build IR:
[h,~] = impz(B,A);

% Cascade:
h_final = 1;
for i = 1 : n
  h_final  = conv(h_final,h);
end

% Normalize:
h_final =
h_final./max(abs(h_final));

% Probe:
response =
hmeasure(h_final);

% Plot:
plot(response)
```

Figure 13, MATLAB.

The impulse response for the cascade of 64 G(z) filters is plotted in Figure 14.

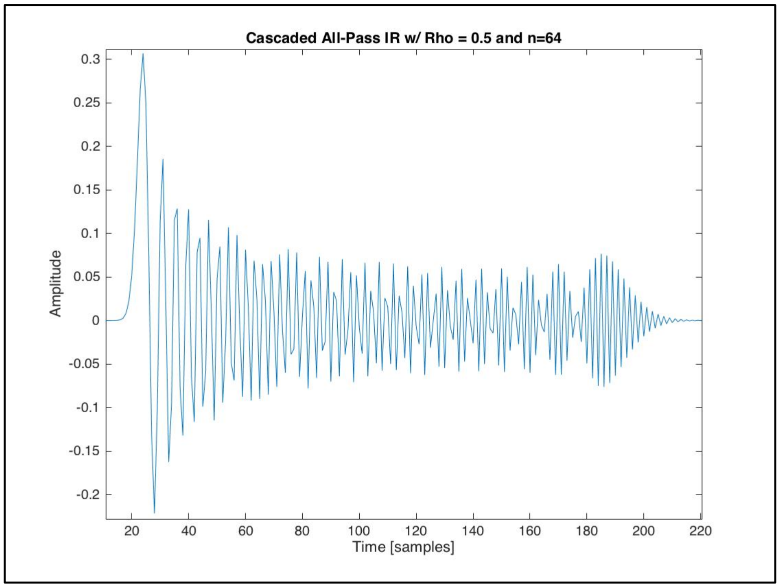

Figure 14, IR of cascaded G(z) filters.

PROBLEM 6: PART B

I'm not sure what this question is asking.

PROBLEM 6: PART C

The response from hmeasure.p is shown in Figure 15 below.

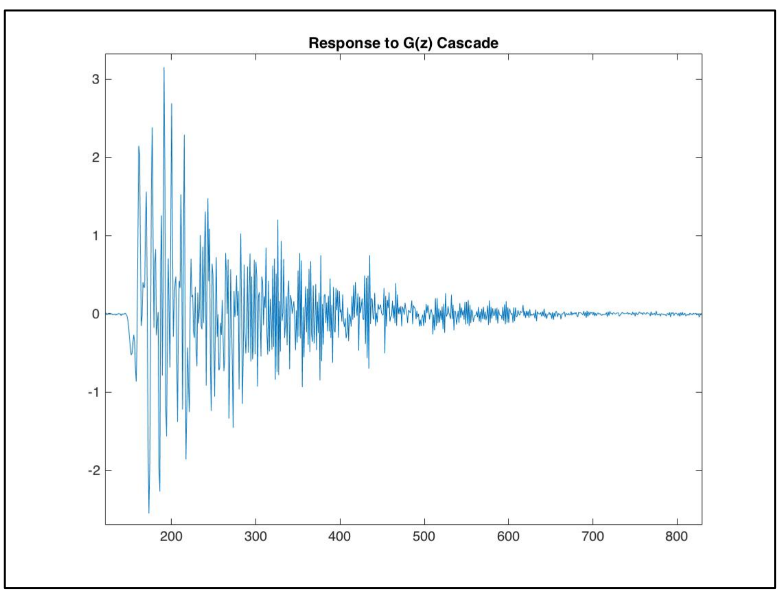

Figure 15, response from hmeasure.p