

Math 471 Project 4 signal processing

Erik Erhardt

December 5, 2005

Figure 1 on page 2 gives the signals. Observe the windowed signal goes to 0 at the extremes. Also, the filtered and quantized signal is slightly different than the original signal, particularly a little rougher in the right tail. Figure 2 on page 2 shows the power spectrum before and after the lowpass filter. Differences are hard to see from the quantization, but they are there.

The lowpass filter above the 270th coefficient, corresponding to 11628 Hz, was chosen before muting of the sharp clacks of the castanets was experienced. The quantization used 4 significant bits before a wavering echo appeared from 3 and lower. Thus, the effective compression achieved was

$$\frac{11628}{22050} \frac{4}{16} = 0.1316$$

which is a little more than 10%, roughly the savings of an mp3 encoding.

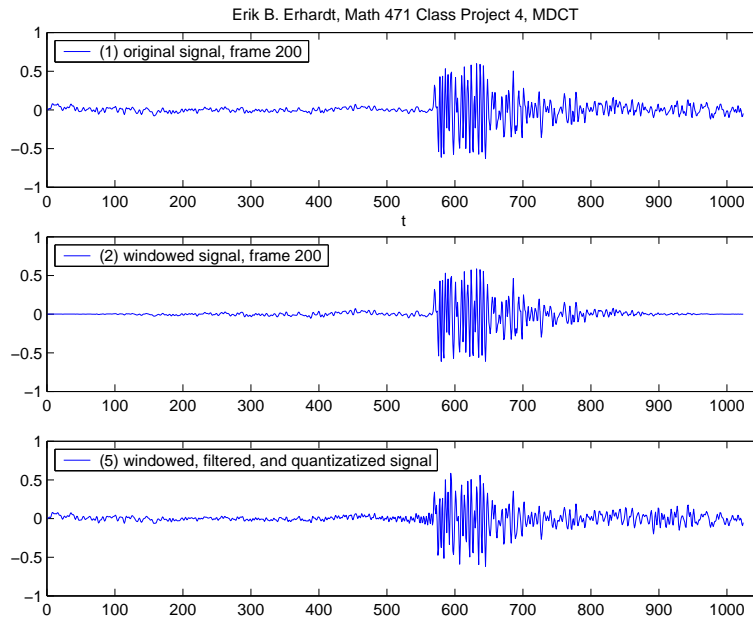


Figure 1: Signals

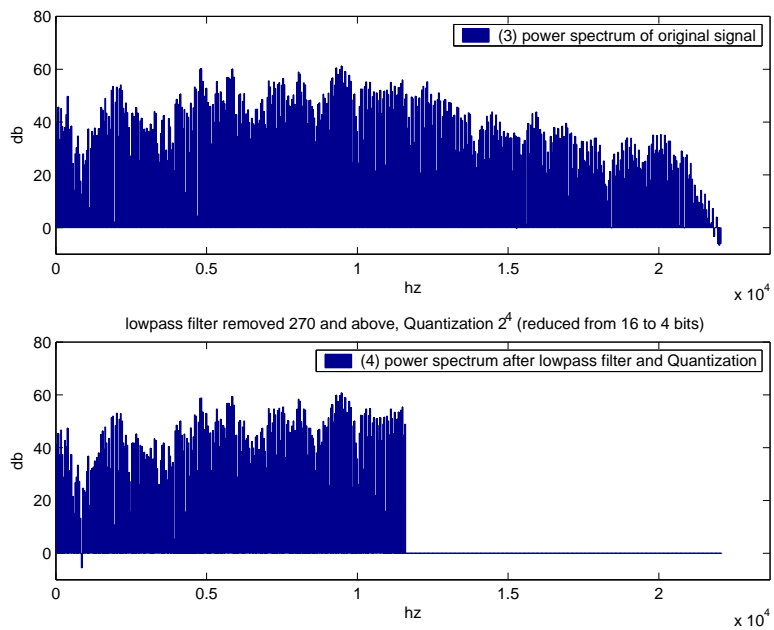


Figure 2: Power spectrums

Appendix

Matlab code

```

% Erik B. Erhardt
% Sci Comp Project 4
%
% test routine for the MDCT code from marios athineos
% http://www.ee.columbia.edu/~marios
%
% NOTES:
% x is original data, y is transformed/inversed data
clear;

hop=512; % frame size (not hz)
[x,rate,nb]=wavread('./castanets.wav');
x=.5*(x(:,1)+x(:,2));
%sound(x,rate,nb);

[fx,fpad] = linframe(x,hop,2*hop,'sym'); % break the input into overlapping frames
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% transform/inverse and check error
fx = winit(fx,'oggwin'); % kbd win is TDAC % window
FX = mdct4(fx); % transform
fy = imdct4(FX); % inverse transform
fy = winit(fy,'oggwin'); % rewind
y = linunframe(fy,hop,fpad); % OLA
e = mean((x-y).^2); % so our error for mdct4
%should have small error
disp(sprintf('Error in MDCT forward,backward transformation: %e ',e));

fx1 = winit(fx,'oggwin'); % kbd win is TDAC % window
FX1 = mdct4(fx1); % transform
% part 1, Lowpass filter
highcut=270; FX1(highcut:hop,:)=0; %%% zero high frequencies (250 loses just a little, 300 clear)
% part 2, Quantization
s = 4; nz=find(FX1>0); % nz are nonzero indicies %FX1(find(FX1<=0))=2^(-60);
a = floor(s - log2(abs(FX1(nz))) + .5);
FX1(nz) = 2.^(-a).*floor(2.^a .* FX1(nz));
%% pow=2^10; FX1=floor(FX1*pow)/pow; %%% noise starts at 2^9 with absolute quant
fy1 = imdct4(FX1); % inverse transform
fy1 = winit(fy1,'oggwin'); % rewind
y1 = linunframe(fy1,hop,fpad); % OLA

%wavwrite(y1,rate,nb,'./castanets2.wav'); % write file to see the size. (is 1/2 since mono regardless of compression)
%sound(y1,rate,nb);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%200th window is the frame
frame=200;
figure(1);clf;

% 1. original data
subplot(3,1,1)
offset=frame*hop - 1170;
plot(x(offset:offset+2*hop-1),'b')
title('Erik B. Erhardt, Math 471 Class Project 4, MDCT')
legend(' (1) original signal, frame 200',2)
xlabel('t'); ylabel('')
axis([0 1050 -1 1])

% 2. windowed signal
subplot(3,1,2)
plot(fx(:,frame))
legend(' (2) windowed signal, frame 200',2)
axis([0 1050 -1 1])

figure(2);clf;

% 3. spectrum
%% GET X-AXIS IN HZ
subplot(2,1,1)
db=FX(:,frame).*FX(:,frame);
db=10*log10(db) + 70; % good normalization for this MDCT and 16bit data
hophz=[1:hop]*44100/1024;
bar(hophz,db)
legend(' (3) power spectrum of original signal',1)
ylabel('db');xlabel('hz')
axis([0 44100/1024*550 -10 80])

% 4. filtered spectrum
%% GET X-AXIS IN HZ

```

```
subplot(2,1,2)
db=FX1(:,frame).*FX1(:,frame);
nz=find(db>0);
db(nz)=10*log10(db(nz)) + 70; % good normalization for this MDCT and 16bit data
hophz=[1:hop]*44100/1024;
bar(hophz,db)
legend('(4) power spectrum after lowpass filter and Quantization',1)
title('lowpass filter removed 270 and above, Quantization 24 (reduced from 16 to 4 bits)')
ylabel('db');xlabel('hz')
axis([0 44100/1024*550 -10 80])

figure(1)
% 5. windowed filtered signal
subplot(3,1,3)
offset=frame*hop - 1170;
plot(y1(offset:offset+2*hop-1),'b')
legend('(5) windowed, filtered, and quantized signal',2)
axis([0 1050 -1 1])

%%%%%%%%%%%% LABEL HZ plots 3 and 4
% hop=a_p+1 - a+p = 12,
% support of gp=1024
% for each 1024 sample window overlapping it will give you 512 MDCT coef.
% k'th coef has k oscillations in 1024 samples
% and there are 44100 samples per seconds
% so the k'th coef is k*44100/1024 hz
```