MUS421/EE367B Applications Lecture 9B: Cross Synthesis
Using Cepstral Smoothing or Linear Prediction for Spectral Envelopes

Julius O. Smith III (jos@ccrma.stanford.edu)
Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University
Stanford, California 94305

March 24, 2014

Outline

- Cross Synthesis
- Spectral Envelope Extraction
  - Cepstral smoothing
  - Linear Prediction
- Example: Speech vowel "ah" [a]
- Matlab code

Application Example: Cross-Synthesis

Cross-synthesis is generally concerned with impressing the spectral envelope of one sound on the flattened spectrum of another.

Let's call the first signal the "modulating" signal, and the other the "carrier" signal.

A classic example is for the modulator to be voice and the carrier to be a spectrally rich sound such as wind, rain, creaking noises, or musical instrument sound.

Example: A "talking organ"

- "Carrier"
- "Modulator"
- Modulated Carrier

Commercial "vocoders" used as musical instruments consist of a keyboard synthesizer (the carrier sounds) with a microphone for picking up the voice of the performer (to extract the modulation envelope).

More Examples:

- Voice "modulator"

- Cow "carrier"
- Voice-modulated cow

- Gong "carrier"
- Voice-modulated gong

- Airplane "carrier"
- Voice-modulated plane

- Creaking ship's mast "carrier"
- Voice-modulated creaking mast
- Same with modified spectral envelopes
- Previous example with modified spectral envelopes 2

Cross-Synthesis Procedure

Cross-synthesis may be summarized as consisting of the following steps:

1. Perform a Short-Time Fourier Transform (STFT) of both the modulator and carrier signals
2. Compute the spectral envelope of each time-frame
3. Divide the spectrum of each carrier frame by its own envelope, thereby flattening it
4. Multiply the flattened spectral frame by the envelope of the corresponding modulator frame, thereby replacing the carrier's envelope by the modulator's envelope.

Spectral Envelope Extraction

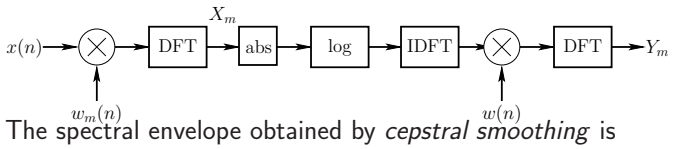
Let X_m denote the spectrum of the mth frame of the modulating signal $x(n)$.

We desire $Y_m = \textsc{Envelope}(X_m)$ to be the *upper spectral envelope* of X_m.

There are several definitions of spectral envelope:

- Cepstral smoothing
- Linear Prediction
- Piecewise linear peak connection or splines

Cepstral Smoothing



The spectral envelope obtained by *cepstral smoothing* is defined as

$$Y_m = \mathrm{DFT}[w \cdot \underbrace{\mathrm{DFT}^{-1}\log(|X_m|)}_{\text{real cepstrum}}]$$

where w is a lowpass window in the cepstral domain, e.g.,

$$w(n) = \begin{cases} 1, & |n| < n_c \\ 0.5, & |n| = n_c \\ 0, & |n| > n_c \end{cases}$$

- The log-magnitude-spectrum of X_m is thus *lowpass filtered* (y_m is "liftered") to obtain a smooth spectral envelope
- Set n_c below the period in the periodic case
- Cepstral coefficients are typically used in *speech recognition* (with *frequency warping* according to the Mel frequency scale — "MFCC")

Linear Prediction Spectral Envelope

Linear prediction itself:

$$y(n) = -a_1 y(n-1) - a_2 y(n-2) - \cdots - a_M y(n-M) + e(n)$$

- Signal $y(n)$ is predicted from its M past samples
- $e(n)$ is the *prediction error* or *innovations sequence*
- Spectral Model:

$$\begin{aligned} Y(z) &= \frac{E(z)}{A(z)} \\ &\approx \hat{Y}(z) \stackrel{\Delta}{=} \frac{\hat{\sigma}_e}{\hat{A}(z)} \end{aligned}$$

- Prediction error $E(z)$ is *spectrally flat* ($e(n)$ approximates white noise or an impulse).

Linear Prediction is Peak Sensitive

By Rayleigh's energy theorem (Parseval's theorem):

$$\begin{aligned} \sum_{n=-\infty}^{\infty} \hat{e}^2(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\hat{E}\left(e^{j\omega}\right)\right|^2 d\omega \\ &\stackrel{\Delta}{=} \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\hat{A}\left(e^{j\omega}\right) Y\left(e^{j\omega}\right)\right|^2 d\omega \\ &= \frac{\hat{\sigma}_e^2}{2\pi}\int_{-\pi}^{\pi} \left|\frac{Y\left(e^{j\omega}\right)}{\hat{Y}\left(e^{j\omega}\right)}\right|^2 d\omega \end{aligned}$$

From this "ratio error" expression in the frequency domain, we can see the following:

- Contributions to the error are smallest when $|\hat{Y}(e^{j\omega})| > |Y(e^{j\omega})|$.
- Therefore, LP tends to *overestimate peaks*.
- LP cannot set $\hat{Y} = \infty$ because the log of the amplitude response of every minimum-phase monic polynomial $A(z)$ is *zero-mean*.

Computation of Linear Prediction Coefficients

The prediction coefficients $\{a_i\}_{i=1}^M$ are easily computable from the *autocorrelation function*:

$$r_{x_m}(l) \stackrel{\Delta}{=} \sum_{n=-\infty}^{\infty} x_m(n)x_m(n+l) = \mathrm{DFT}^{-1}\left|X_m\right|^2$$

To obtain the Mth-order linear predictor coefficents $\{a_1, \ldots, a_M\}$, solve the $M \times M$ system of linear equations:

$$\sum_{i=1}^{M} a_i r_{x_m}(|i-j|) = -r_{x_m}(j), \qquad j = 1, 2, \ldots, M$$

In Matlab, "a=R\p", where `p(j)` $= r_{x_m}(j)$, and `R(i,j)` $= r_{x_m}(|i-j|)$.

- Solution always exists
- If rank is M, solution is unique
- Unique solution is always *stable* (roots of $A(z)$ are inside the unit circle in the z plane)
- Since R is *Toeplitz*, an $O(M^2)$ solution exists

LPC Spectral Envelope

$$Y_m(\omega_k) = \frac{g}{A\left(e^{j\omega_k}\right)} \quad \text{or} \quad \frac{g}{\left|A\left(e^{j\omega_k}\right)\right|}$$

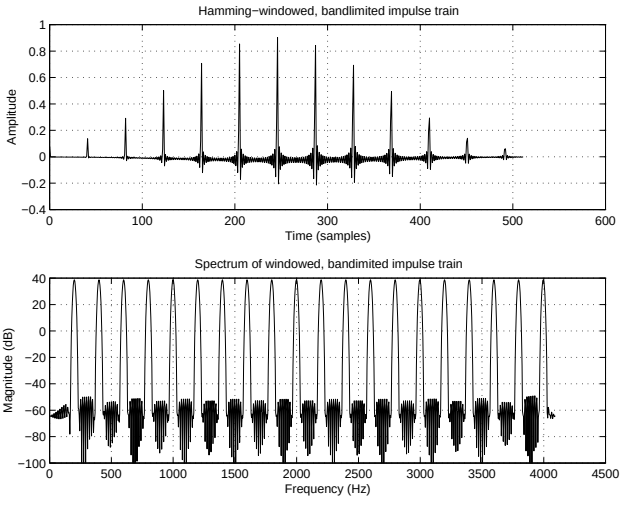
- Typically, $g = \| E \|_2$
- Note that $\log\left[A\left(e^{j\omega_k}\right)\right]$ is *zero mean*
- For voice, M should be at least twice the number of spectral *formants*.
- For best results, use the *Bark bilinear transform*[1] to warp the spectral axis.

[1] http://ccrma.stanford.edu/~jos/bbt/

LPC Envelope Example: Speech vowel "ah"

```
% Let's make an "ah" [a] vowel:
% Ref: Dennis H. Klatt, "Software for a
% cascade/parallel formant synthesizer,"
%      JASA, vol. 67, pp. 13-33, 1980.
F = [700, 1220, 2600]; % Formant frequencies in Hz
B = [130,   70,  160]; % Formant bandwidths in Hz

fs = 8192;  % Sampling rate in Hz
    % ("telephone quality" for speed)
R = exp(-pi*B/fs);     % Pole radii
theta = 2*pi*F/fs;     % Pole angles
poles = R .* exp(j*theta) % Poles
[B,A] = zp2tf(0,[poles,conj(poles)],1); % control/

f0 = 200; % Pitch in Hz
w0T = 2*pi*f0/fs;

nharm = floor((fs/2)/f0); % number of harmonics
sig = zeros(1,nsamps);
n = 0:(nsamps-1);
% Synthesize bandlimited impulse train
for i=1:nharm,
    sig = sig + cos(i*w0T*n);
end;
sig = sig/max(sig);
soundsc(sig,fs); % Let's hear it
```

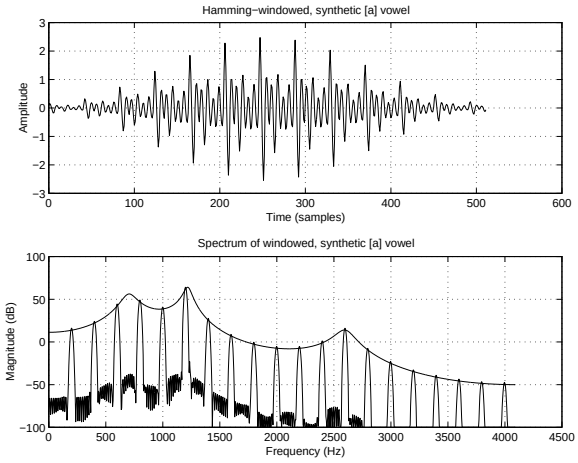

Impulse Train



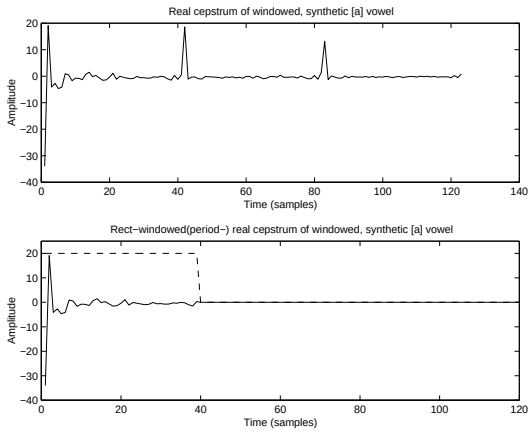
Impulse train sound example

Speech Vowel and its Spectrum

```
speech = filter(1,A,sig); % impulse-train -> 'Ah' filter
soundsc([sig,speech],fs);
winspeech = w .* speech(1:length(w));

sspec = fft([winspeech,zeros(1,3*nplot)]); % interpolated spectrum
dbsspecfull = 20*log(abs(sspec));
dbsspec = dbsspecfull(1:nspec);
dbenv = 20*log(abs(freqz(1,A,nspec)'));
dbsspecn = dbsspec + ones(1,nspec)*(max(dbenv) ...
                   - max(dbsspec)); % normalize
plot(f,[max(dbsspecn,-100);dbenv]); grid;
```

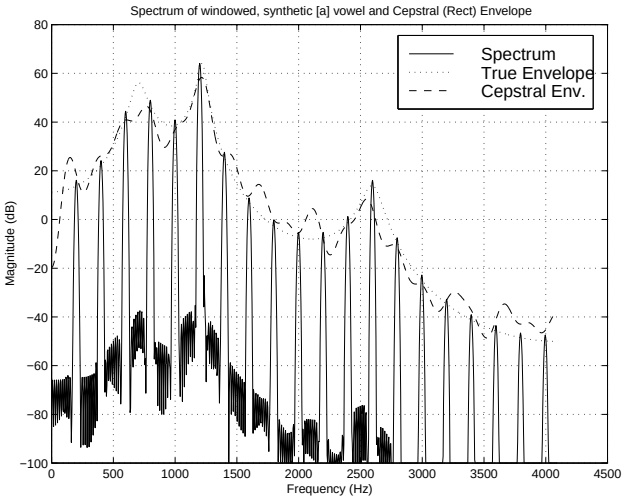


Spectral Envelope via Windowed Cepstrum



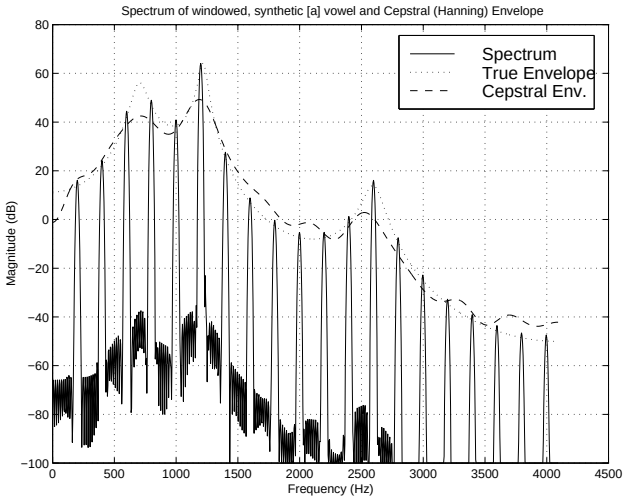

```
rcep = ifft(dbsspecfull);  % real cepstrum
imagerr = norm(imag(rcep))/norm(rcep) % check
rcep = real(rcep); % imag part is just roundoff error
period = round(fs/f0) % 41

aliasing = norm(rcep(nspec-10:nspec+10))/norm(rcep) % 0.0229

nw = 2*period-4; % almost 1 period left and right
if floor(nw/2) == nw/2, nw=nw-1; end; % make it odd
w = boxcar(nw)'; % rectangular window
wzp = [w(((nw+1)/2):nw),zeros(1,nfft-nw), ...
       w(1:(nw-1)/2)];  % zero-centered version
wrcep = wzp .* rcep;
```


Rectangular-Windowed Cepstrum




```
% Display cepstral envelope
rcepenv = fft(wrcep);
imagerr = norm(imag(rcepenv)) % should be zero
rcepenvp = real(rcepenv(1:nspec));
rcepenvp = rcepenvp + ones(1,nspec)*(mean(dbenv)...
                    - mean(rcepenvp)); % normalize
plot(f,[max(dbsspecn,-100); dbenv; rcepenvp]);
```

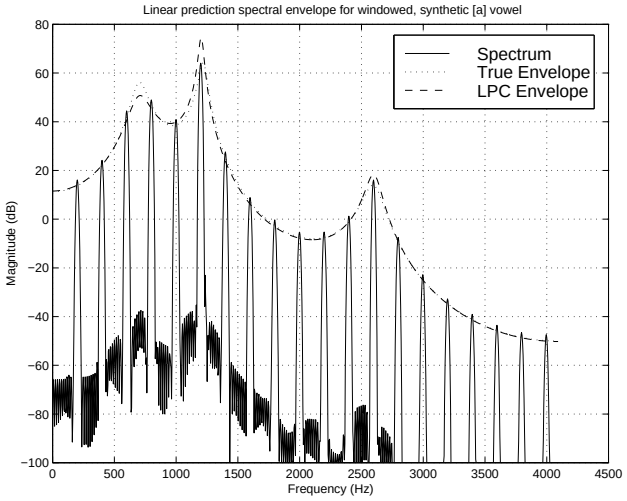

Hanning-Windowed Cepstrum



Spectral Envelope via Linear Prediction

Finally, let's do an LPC window. It had better be good because the LPC model is exact for this example.

```
M = 6; % three formants

% compute Mth-order autocorrelation function:
rx = zeros(1,M+1)';
for i=1:M+1,
  rx(i) = rx(i) + speech(1:nsamps-i+1) ...
                * speech(1+i-1:nsamps)';
end

% prepare the M by M Toeplitx covariance matrix:
covmatrix = zeros(M,M);
for i=1:M,
  covmatrix(i,i:M) = rx(1:M-i+1)';
  covmatrix(i:M,i) = rx(1:M-i+1);
end

% solve "normal equations" for prediction coeffs

Acoeffs = - covmatrix \ rx(2:M+1)

Alp = [1,Acoeffs']; % LP polynomial A(z)

dbenvlp = 20*log(abs(freqz(1,Alp,nspec)'));
dbsspecn = dbsspec + ones(1,nspec)*(max(dbenvlp) ...
                   - max(dbsspec)); % normalize
plot(f,[max(dbsspecn,-100);dbenv;dbenvlp]); grid;
```

Sound Example for LPC Speech Vowel "ah"

Sounds:

- Impulse Train
- Synthetic "Ah" Vowel