LECTURE 16: LINEAR PREDICTION-BASED REPRESENTATIONS

- Objectives:
  - Introduce analysis/synthesis systems
  - Discuss error analysis and gain-matching
  - Relate linear prediction coefficients to other spectral representations
  - Introduce reflection and prediction coefficent recursions
  - Lattice/ladder filter implementations

There is a classic textbook on this subject:

> J.D. Markel and A.H. Gray, *Linear Prediction of Speech*, Springer-Verlag, New York, New York, USA, ISBN: 0-13-007444-6, 1976.

This lecture also includes material from two other textbooks:

> J. Deller, et. al., *Discrete-Time Processing of Speech Signals*, MacMillan Publishing Co., ISBN: 0-7803-5386-2, 2000.

and,

> L.R. Rabiner and B.W. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-015157-2, 1993.

ECE 8463: FUNDAMENTALS OF SPEECH RECOGNITION

Professor Joseph Picone
Department of Electrical and Computer Engineering
Mississippi State University

email: picone@isip.msstate.edu
phone/fax: 601-325-3149; office: 413 Simrall
URL: http://www.isip.msstate.edu/resources/courses/ece_8463

Modern speech understanding systems merge interdisciplinary technologies from Signal Processing, Pattern Recognition, Natural Language, and Linguistics into a unified statistical framework. These systems, which have applications in a wide range of signal processing problems, represent a revolution in Digital Signal Processing (DSP). Once a field dominated by vector-oriented processors and linear algebra-based mathematics, the current generation of DSP-based systems rely on sophisticated statistical models implemented using a complex software paradigm. Such systems are now capable of understanding continuous speech input for vocabularies of hundreds of thousands of words in operational environments.

In this course, we will explore the core components of modern statistically-based speech recognition systems. We will view speech recognition problem in terms of three tasks: signal modeling, network searching, and language understanding. We will conclude our discussion with an overview of state-of-the-art systems, and a review of available resources to support further research and technology development.

Tar files containing a compilation of all the notes are available. However, these files are large and will require a substantial amount of time to download. A tar file of the html version of the notes is available here. These were generated using wget:

```
wget -np -k -m http://www.isip.msstate.edu/publications/courses/ece_8463/lectures/current
```

A pdf file containing the entire set of lecture notes is available here. These were generated using Adobe Acrobat.

Questions or comments about the material presented here can be directed to help@isip.msstate.edu.

LECTURE 16: LINEAR PREDICTION-BASED REPRESENTATIONS

- Objectives:
    - Introduce analysis/synthesis systems
    - Discuss error analysis and gain-matching
    - Relate linear prediction coefficients to other spectral representations
    - Introduce reflection and prediction coefficent recursions
    - Lattice/ladder filter implementations

There is a classic textbook on this subject:

> J.D. Markel and A.H. Gray, *Linear Prediction of Speech*, Springer-Verlag, New York, New York, USA, ISBN: 0-13-007444-6, 1976.

This lecture also includes material from two other textbooks:

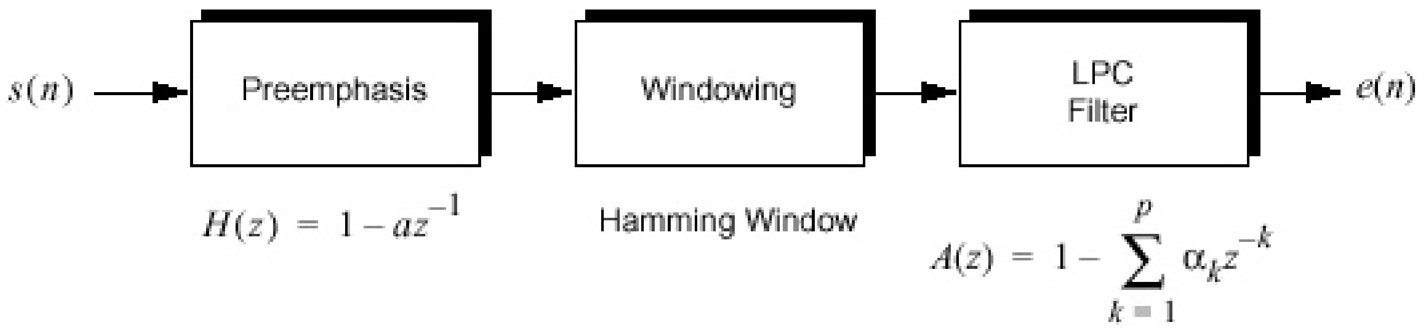
> J. Deller, et. al., *Discrete-Time Processing of Speech Signals*, MacMillan Publishing Co., ISBN: 0-7803-5386-2, 2000.

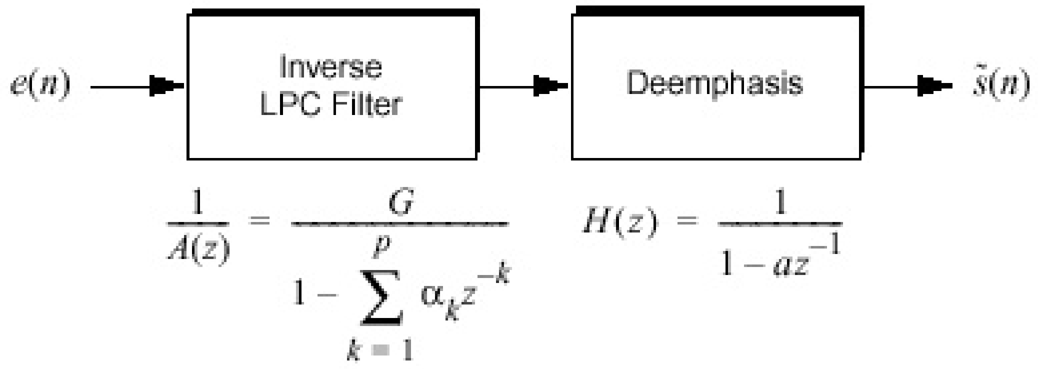
and,

> L.R. Rabiner and B.W. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-015157-2, 1993.

AN LPC-BASED ANALYSIS SYNTHESIS SYSTEM

We can use the linear prediction model to create an analysis/synthesis system:



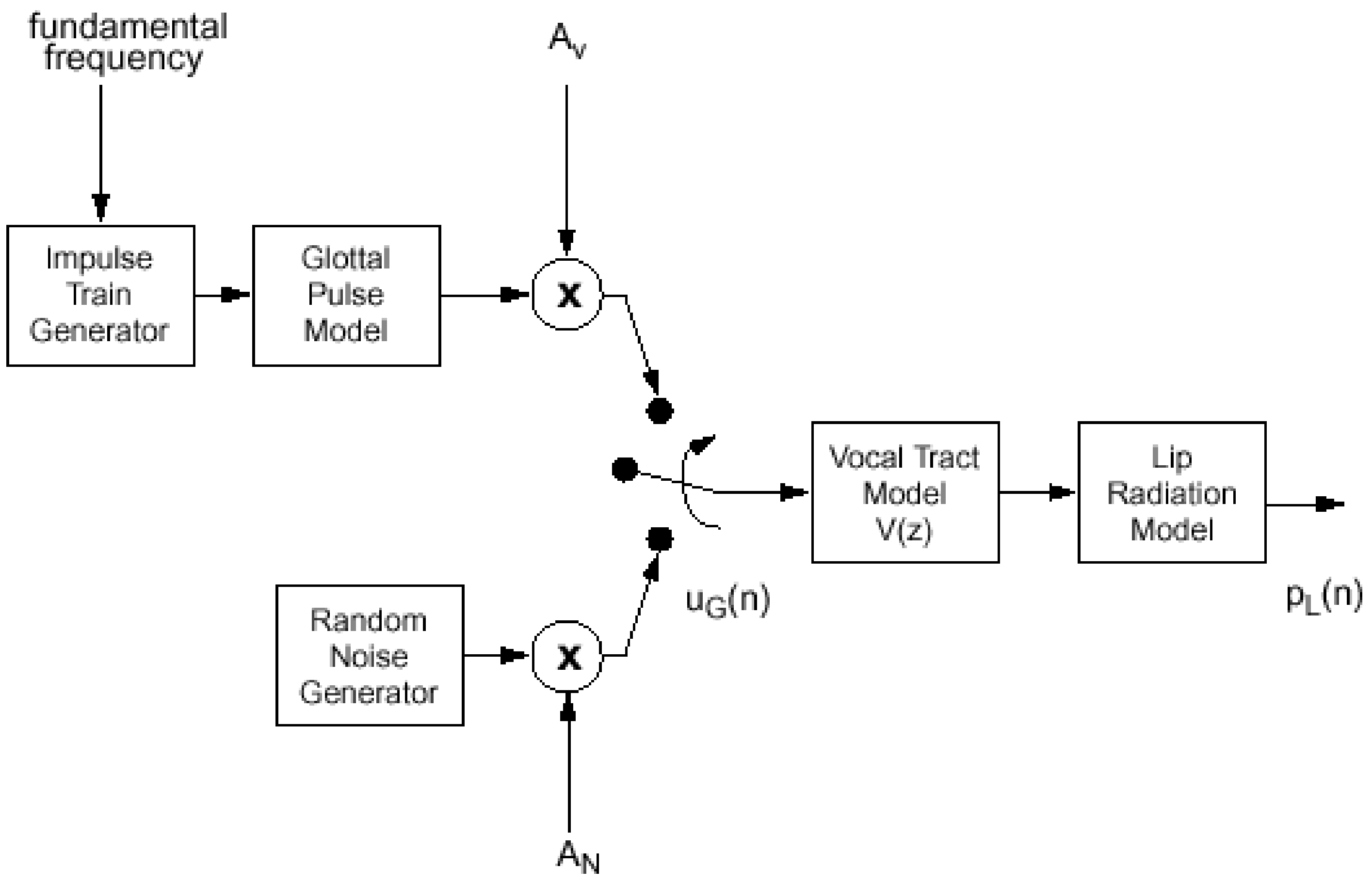

$e(n)$
Inverse LPC Filter
Deemphasis
$\tilde{s}(n)$
$\frac{1}{A(z)} = \frac{G}{1 - \sum_{k=1}^{p} \alpha_k z^{-k}}$
$H(z) = \frac{1}{1 - az^{-1}}$

Recall that the linear prediction model, A(z), takes on the form:

$$A(z) = 1 - \sum_{k=1}^{p} \alpha_k z^{-k}$$

This is an all-zero filter which is minimum phase. The inverse of this filter, which is used to synthesize the signal, is an all-pole filter. If the autocorrelation method is used to compute the LP coefficients, the filter is guaranteed to be stable.

Note the similarities of this system to our digital model of speech production:



THE LPC ERROR SIGNAL

The LPC error signal can be computed using:

$$E(z) = S(z)A(z) \qquad A(z) = 1 - \sum_{k=1}^{p} \alpha_k z^{-k}$$

Below are some examples of the LPC error signal:

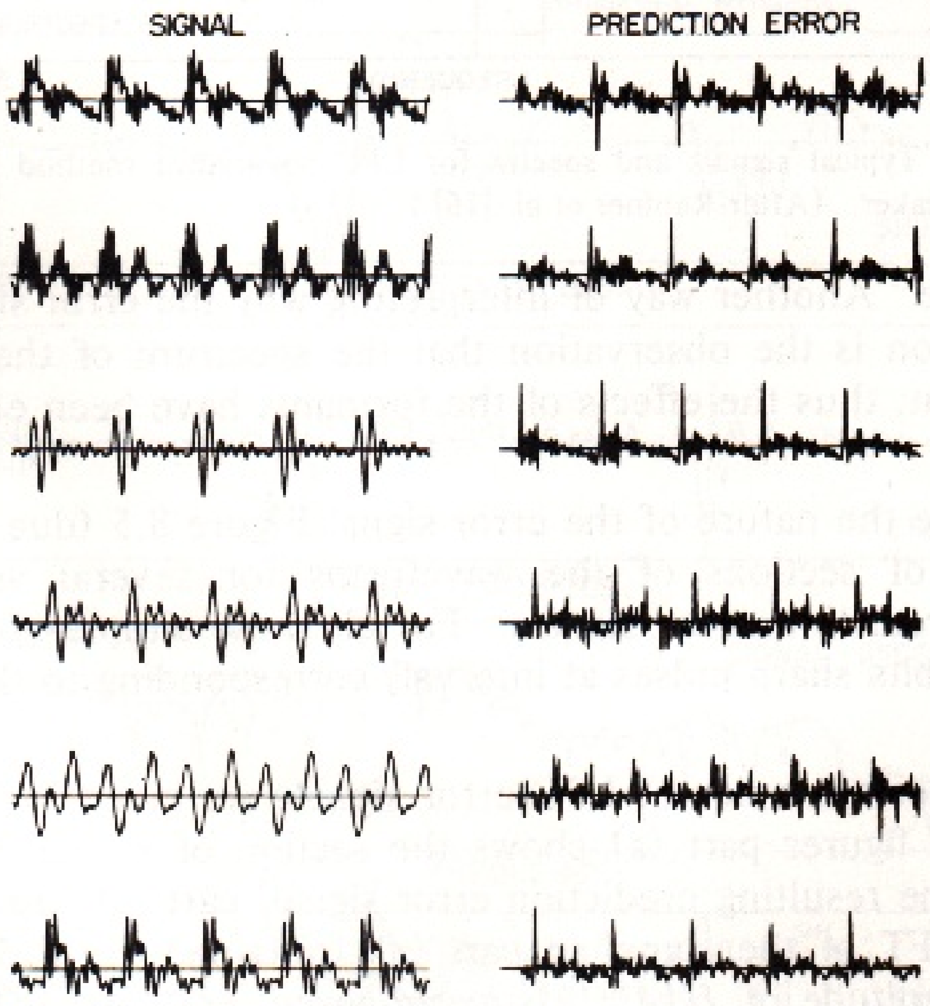


Fig. 8.5 Examples of signal (differentiated) and prediction error for vowels (*i*, *e*, *a*, *o*, *u*, *y*). (After Strube [14].)

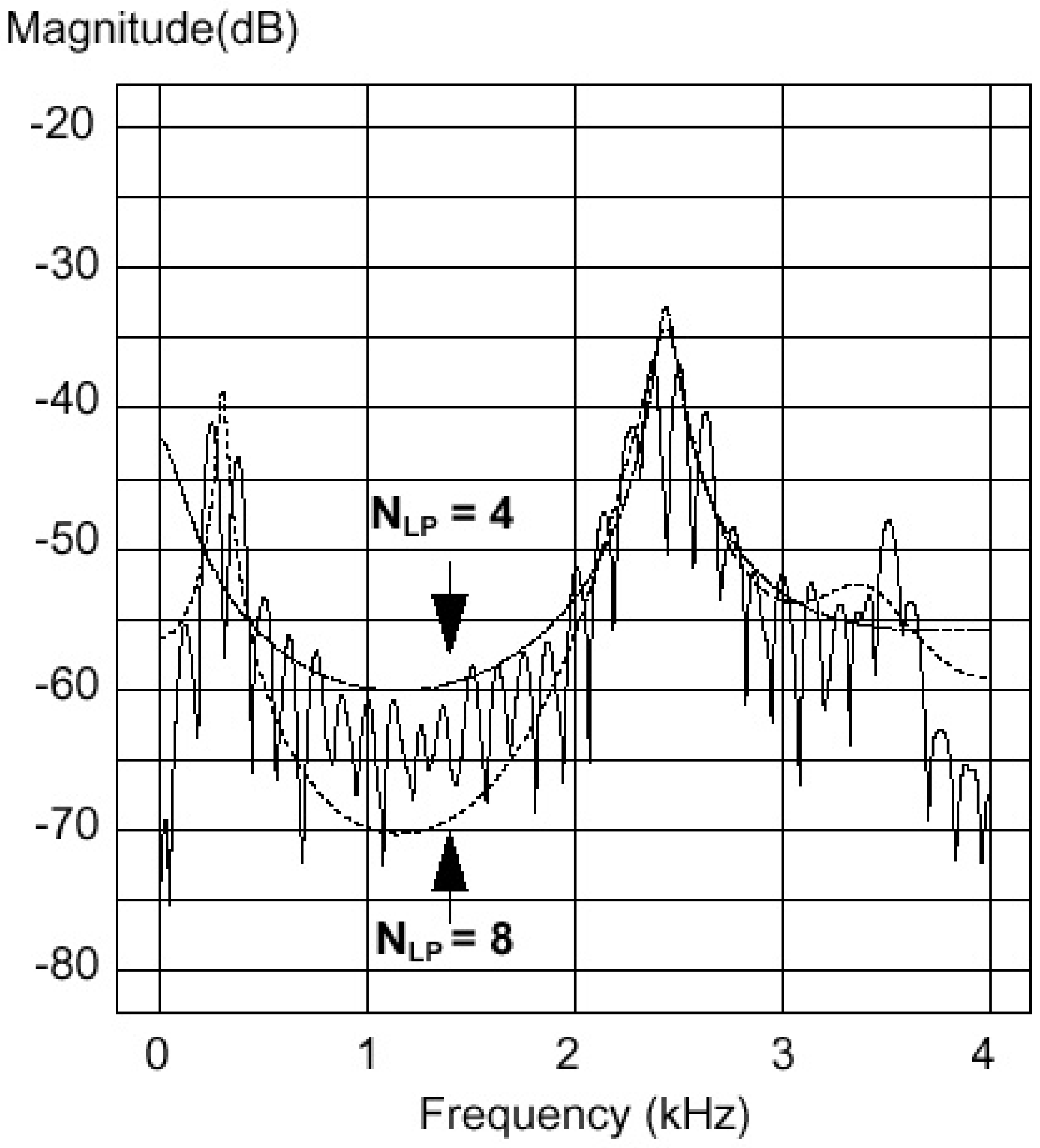
The error energy (related to accuracy) can be computed from the LPC reflection coefficients:

$$E = R_0 \prod_{j=1}^{p} 1 - k_i^2$$

Note that if the reflection coefficients have a magnitude less than one, the error energy is monotonically decreasing with the linear prediction order.

GAIN MATCHING THE LPC MODEL

Recall the spectral matching interpretation of the LP model:



LP analysis is invariant to energy (the same signal at different energy levels produces the same LP coefficients). How do we match the spectrum implied by the LP model to the signal:

- Gain matching:

$$S(z) = \frac{1}{A(z)} = \frac{G}{1 - \sum_{k=1}^{p} \alpha_k z^{-k}} \qquad G^2 = R(0) - \sum_{k=1}^{p} \alpha_k R(k)$$

- Error energy:

$$G = R_0 \prod_{j=1}^{p} 1 - k_i^2$$

Either techniques produces the same result. The latter technique is popular in speech compression systems.

LEVINSON-DURBIN RECURSION

The prediction coefficients can be efficiently computed for the autocorrelation method using the Levinson-Durbin recursion:

```
$for\ i = 1, 2, \ldots, p$
 $E_0 = r(0)$
 $k_i = \left( r(i) - \sum_{j=1}^{i-1} a_{i-1}(j) r(i-j) \right) / E_{i-1}$
 $for\ \mathrm{j} = 1, 2, \ldots, i-1$
    $a_i(i) = k_i$
    $a_i(j) = a_{i-1}(j) - k_i a_{i-1}(i-j)$
 $E_i = (1 - k_i^2) E_{i-1}$
```

This recursion gives us great insight into the linear prediction process. Let us examine a simple example in which we compute a second order model:

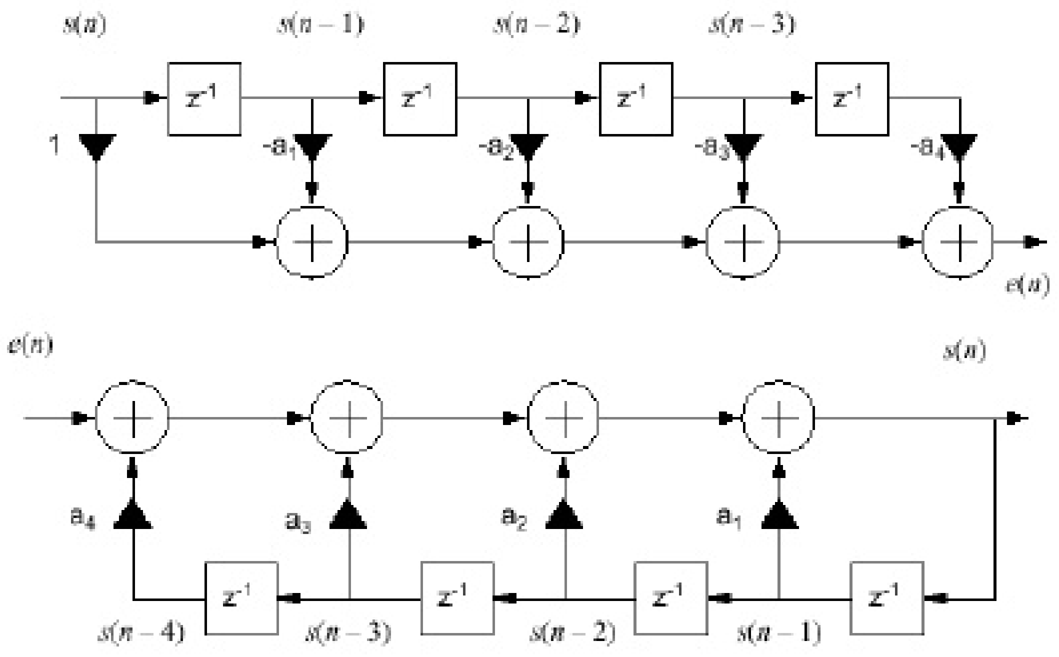
$$
\begin{aligned}
E_0 &= r(0) \\
k_1 &= r(1)/r(0) \\
a_1(1) &= k_1 = r(1)/r(0) \\
E_1 &= (1 - k_1^2) E_0 \\
&= \frac{r^2(0) - r^2(1)}{r(0)} \\
k_2 &= \frac{r(2)r(0) - r^2(1)}{r^2(0) - r^2(1)}
\end{aligned}
$$

$$
\begin{aligned}
a_2(2) &= k_2 = \frac{r(2)r(0) - r^2(1)}{r^2(0) - r^2(1)} \\
a_2(1) &= a_1(1) - k_2 a_1(1) \\
&= \frac{r(1)r(0) - r(1)r(2)}{r^2(0) - r^2(1)} \\
\alpha_1 &= a_2(1) \\
\alpha_2 &= a_2(2)
\end{aligned}
$$

This reduces the LP problem to O(p) complexity and saves an order of magnitude in computational complexity. This calculation also makes the process amenable to fixed-pint digital signal processors and microprocessors.

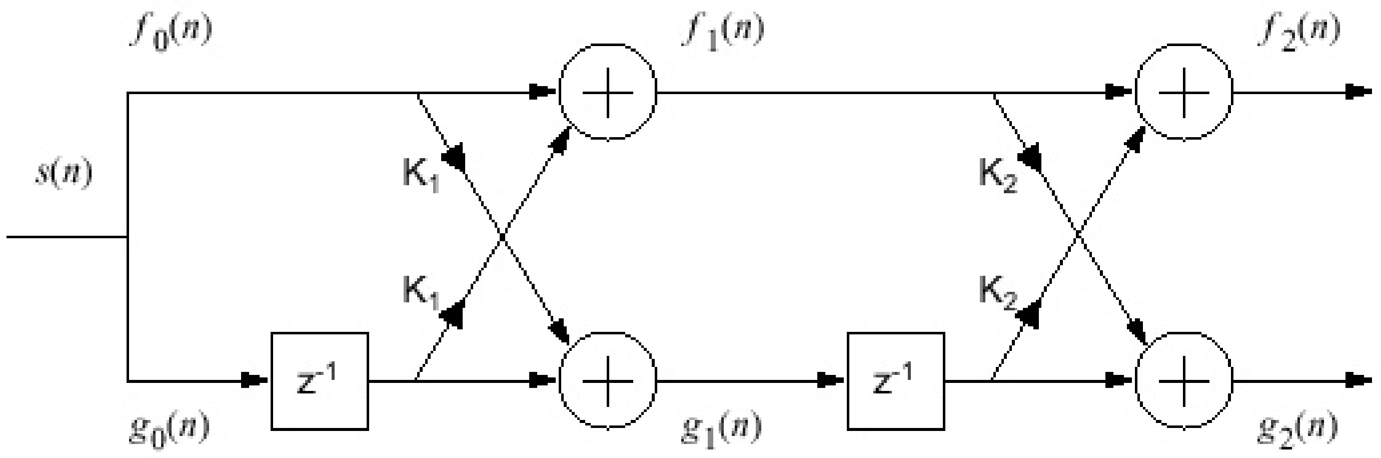
PREDICTION COEFFICIENTS

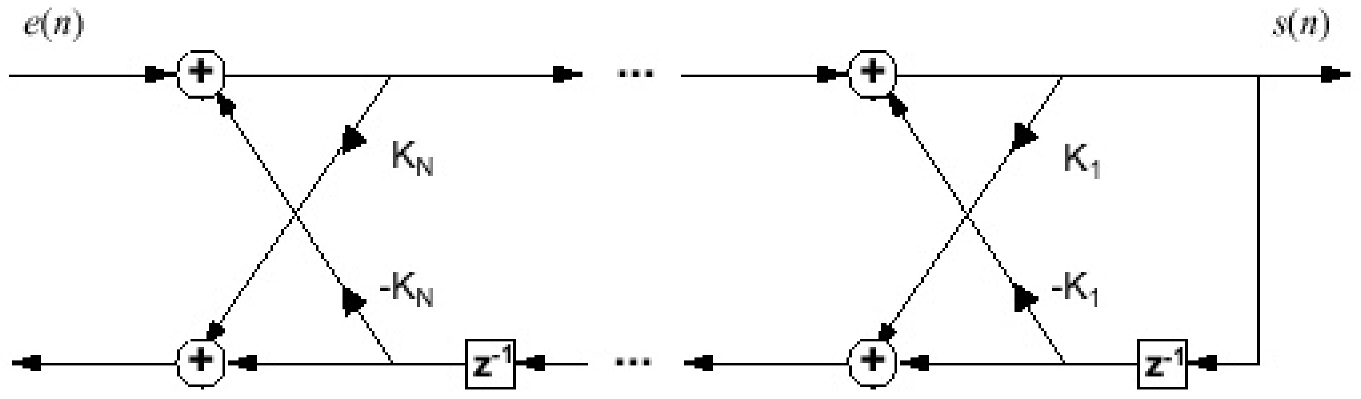
The LP model can be implemented as a direct form filter:



The coefficients of this model are referred to as *prediction coefficients*.

This same filter can be implemented using a lattice filter:





The coefficients of this filter are referred to as *reflection coefficients*. They can be related directly to the lossless concatenated tube model for the vocal tract (log area ratios).

LINEAR PREDICTION COEFFICIENT TRANSFORMATIONS

The prediction and reflection coefficients can be transformed back and forth with no loss of information:

Prediction to reflection coefficient transformation:

```
for $i = p, p-1, \ldots, 1$
    $k_i = \alpha_i(i)$
    $\alpha_{i-1}(j) = \dfrac{\alpha_i(j) + k_i\alpha_i(i-j)}{1-k_i^2} \qquad 1 \le j \le i-1$
```

Reflection to prediction coefficient transformation:

```
for $i = 1, 2, \ldots, p$
    $\alpha_i(i) = k_i$
    $\alpha_i(j) = \alpha_{i-1}(j) - k_i\alpha_{i-1}(i-j) \qquad 1 \le j \le i-1$
```

Also, note that these recursions require intermediate storage for $\{\alpha_i\}$.

From the above recursions, it is clear that $|k_i| \ne 1$. In fact, there are several important results related to k_i:

(1) $|k_i| < 1$

(2) $|k_i| = 1$, implies a harmonic process (poles on the unit circle).

(3) $|k_i| > 1$ implies an unstable synthesis filter (poles outside the unit circle).

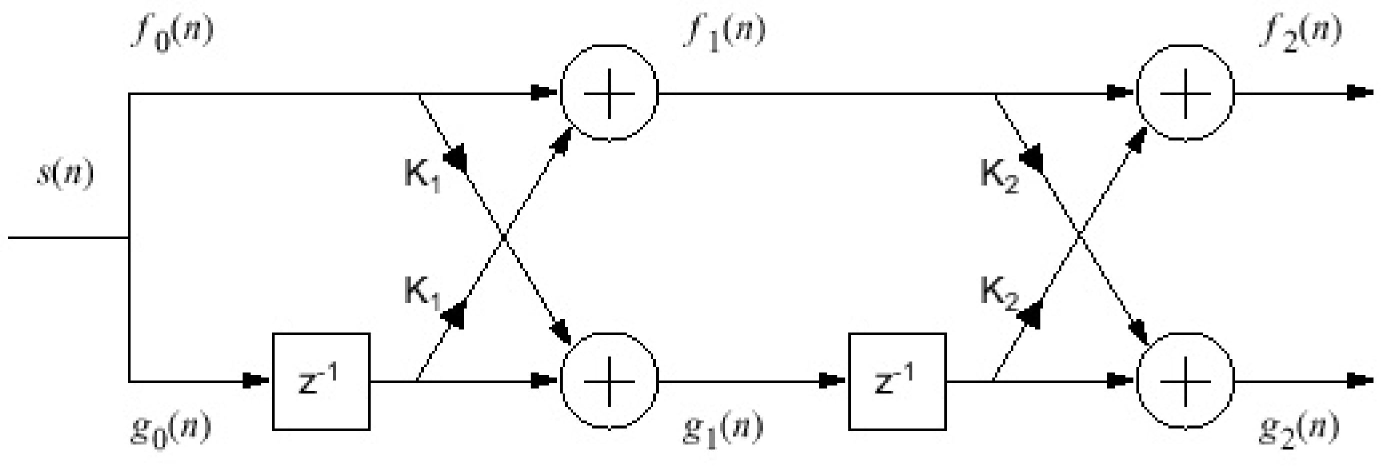
(4) $E_0 > E_1 > \ldots > E_p$

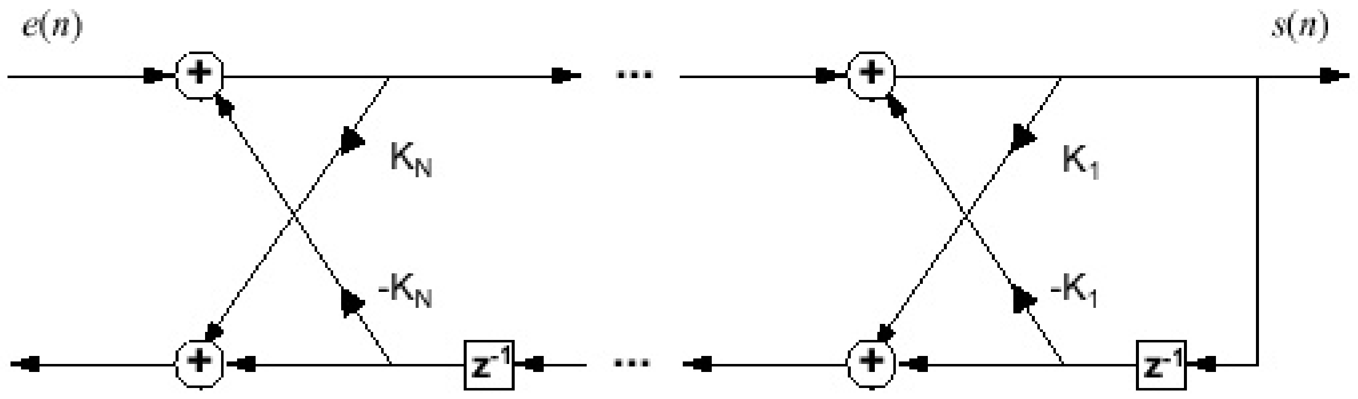
This gives us insight into how to determine the LP order during the calculations. We also see that reflection coefficients are orthogonal in the sense that the best order "p" model is also the first "p" coefficients in the order "p+1" LP model (very important!).

We can also convert LP coefficients to/from autocorrelation coefficients, cepstral coefficients, and log area ratios. See Signal Modeling Techniques for further details.

THE BURG METHOD

The standard direct-form FIR filter can be implemented in a lattice structure:





The reflection coefficients can be computed directly using this equation:

$$K_i = \frac{\sum_{m=0}^{N-1} f_{i-1}(m) g_{i-1}(m-1)}{\left\{\left(\sum_{m=0}^{N-1} (f_{i-1}(m))^2\right)\left(\sum_{m=0}^{N-1} (f_{i-1}(m-1))^2\right)\right\}^2}$$

For this filter to be stable, the reflection coefficients must be bounded by 1.

This equation represents one solution from a family of solutions based on lattice methods.

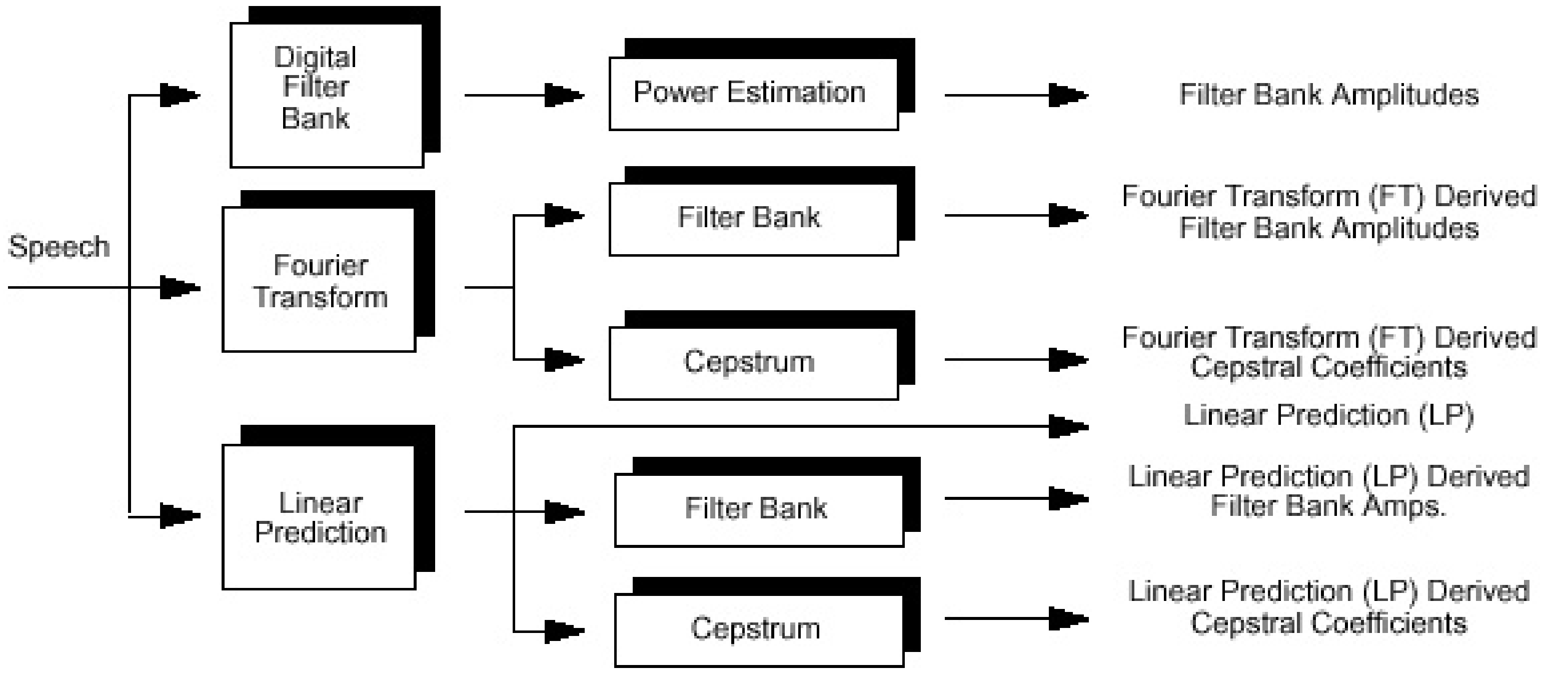
One of the most famous lattice formulations in this family is the *Burg algorithm*, originally introduced in the mid-60's prior to the introduction of LP in speech. There are actually a family of lattice solutions of a similar form. The Burg algorithm requires the coefficients to be computed using:

$$K_i = \frac{\sum_{m=0}^{N-1} f_{i-1}(m) g_{i-1}(m-1)}{\frac{1}{2}\left\{\sum_{m=0}^{N-1} f_{i-1}(m)^2 + \sum_{m=0}^{N-1} f_{i-1}(m-1)^2\right\}}$$

We see this is a weighted average of the forward and backward error terms, and that the reflection coefficients are FORCED to be bounded.

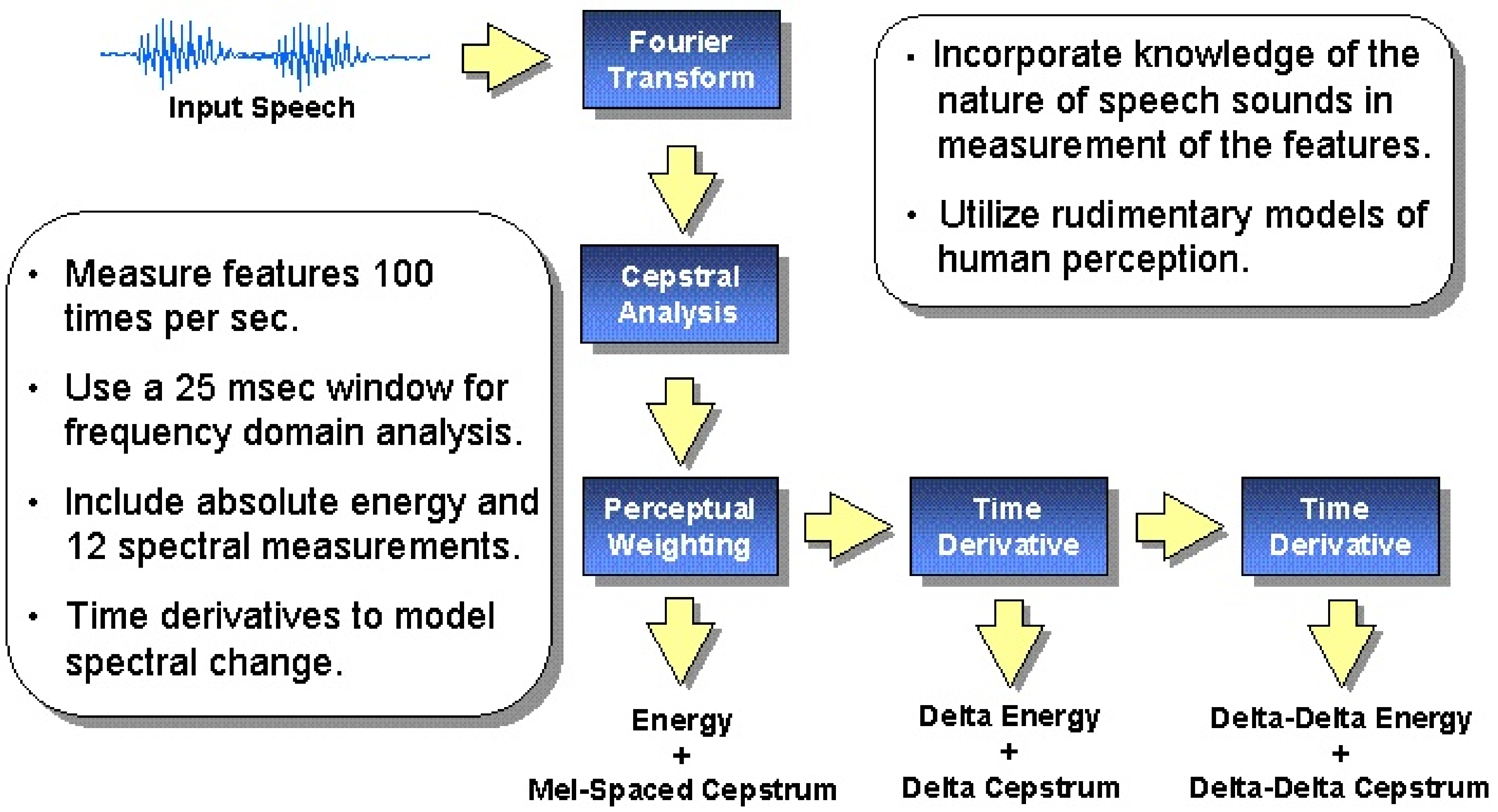
ALTERNATIVE METHODS FOR FREQUENCY DOMAIN ANALYSIS

We have now established two different ways to perform a filterbank analysis of the speech signal (temporal and spectral):


Speech
Digital Filter Bank
Power Estimation
Filter Bank Amplitudes
Fourier Transform
Filter Bank
Fourier Transform (FT) Derived Filter Bank Amplitudes
Cepstrum
Fourier Transform (FT) Derived Cepstral Coefficients
Linear Prediction (LP)
Linear Prediction
Filter Bank
Linear Prediction (LP) Derived Filter Bank Amps.
Cepstrum
Linear Prediction (LP) Derived Cepstral Coefficients

The most popular front ends are those that use cepstral coefficients dervied from the Fourier transform. Why?

A TYPICAL SPEECH RECOGNITION FRONT END


Input Speech
Fourier Transform
Cepstral Analysis
Perceptual Weighting
Time Derivative
Time Derivative
Energy + Mel-Spaced Cepstrum
Delta Energy + Delta Cepstrum
Delta-Delta Energy + Delta-Delta Cepstrum

- Incorporate knowledge of the nature of speech sounds in measurement of the features.
- Utilize rudimentary models of human perception.

- Measure features 100 times per sec.
- Use a 25 msec window for frequency domain analysis.
- Include absolute energy and 12 spectral measurements.
- Time derivatives to model spectral change.

Linear Prediction analysis

Linear prediction analysis of speech is historically one of the most important speech analysis techniques. The basis is the source-filter model where the filter is constrained to be an all-pole linear filter. This amounts to performing a linear prediction of the next sample as a weighted sum of past samples:

$$\hat{s}_n = \sum_{i=1}^{p} a_i s_{n-i} \tag{61}$$

This linear filter has the transfer function:

$$H(z) = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} \tag{62}$$

A good introductory article is [8], and this subject is also covered well in [1, 2, 3].

LPC10E (2.4 Kbps) - Linear Prediction Coder

Since most coders used today are based on LPC10E (while additionally coding the residual), I have chosen to include a little history on it. This was taken from [DEL93], pp. 473-474:

"The earliest LPC-based vocoders followed the classic all-pole model" [figure references omited] "directly in the use of a pitch-pulse excitation for synthesis of voiced speech, and a noise source excitation for unvoiced. Such a vocoder first caught the public's attention in the educational toy "Speak-and-Spell" produced by Texas Instruments in the 1970s. The basic method is still in use in many vocoding applications and is employed by the U.S. Government as one speech coding standard. The algorithm is usually called LPC-10, a reference to the fact that 10 coefficients are typically employed."

"One version of LPC-10 is described in the papers by Kang et al. (1979) and Tremain (1982). LPC-10 partitions the speech into 180 sample frames, resulting in a frame rate of 44.44 frames/sec when an 8-KHz sample rate is used on the data. Pitch and a voicing decision are determined by using the AMDF and zero crossing measures [reference to section deleted]." "The covariance method is used to compute the LP solution, and a set of "generalized" reflection coefficients are encoded for storage or transmission. Ten coefficients are computed; the first two are encoded as [Log-Area Ratios] LARs and the others are encoded linearly. When an unvoiced decision is made, only four coefficients are used. In the stored or transmitted bit stream, 41 bits are used for the reflection coefficients, 7 for pitch and the voiced/unvoiced bit, and 5 for the gain. One additional bit is used for synchronization. Accordingly, a total of 54 bits per frame are sent, yielding a bit rate of 2400 bps."

The LPC10E and CELP software implementation may be purchased from NTIS U.S. Department of Commerce, 5285 Port Royal Road, Springfield, VA 22161, USA (703) 487-4650, or obtained for free from CMU ftp.cs.cmu.edu, or 139.169.31.12

In case you are going through NTIS, the "AD" ordering number for the CELP software is AD M000 118 (US$ 90.00) and for the TIB it's AD A256 629 (US$ 17.50). The LPC-10 standard is FIPS Pub 137 (US$ 12.50). There is a $3.00 shipping charge on all U.S. orders. The telephone number for their automated system is 703-487-4650, or 703-487-4600 if you'd prefer to talk with a real person.

The source code was obtained from Carnegie Mellon University, in Pittsburg, PA, FTP: ftp.cs.cmu.edu, or 139.169.31.12, in the /project/fgdata/speech-compression/ directory in July 1995.

Environment

Quiet Room

Shuttle: STS-62

Shuttle: STS-62 (group)

Shuttle: STS-41 (cabin) 10dB

Shuttle: STS-41 (cabin) 5dB

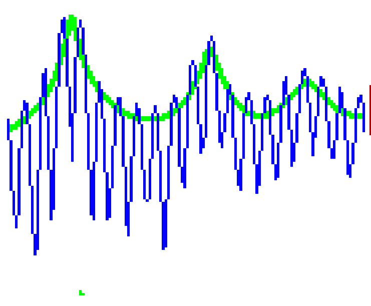
Street Noise: 10dB

Street Noise: 5dB

Music: RUSH 2

Music: Pink Floyd 1

DTMF Tones

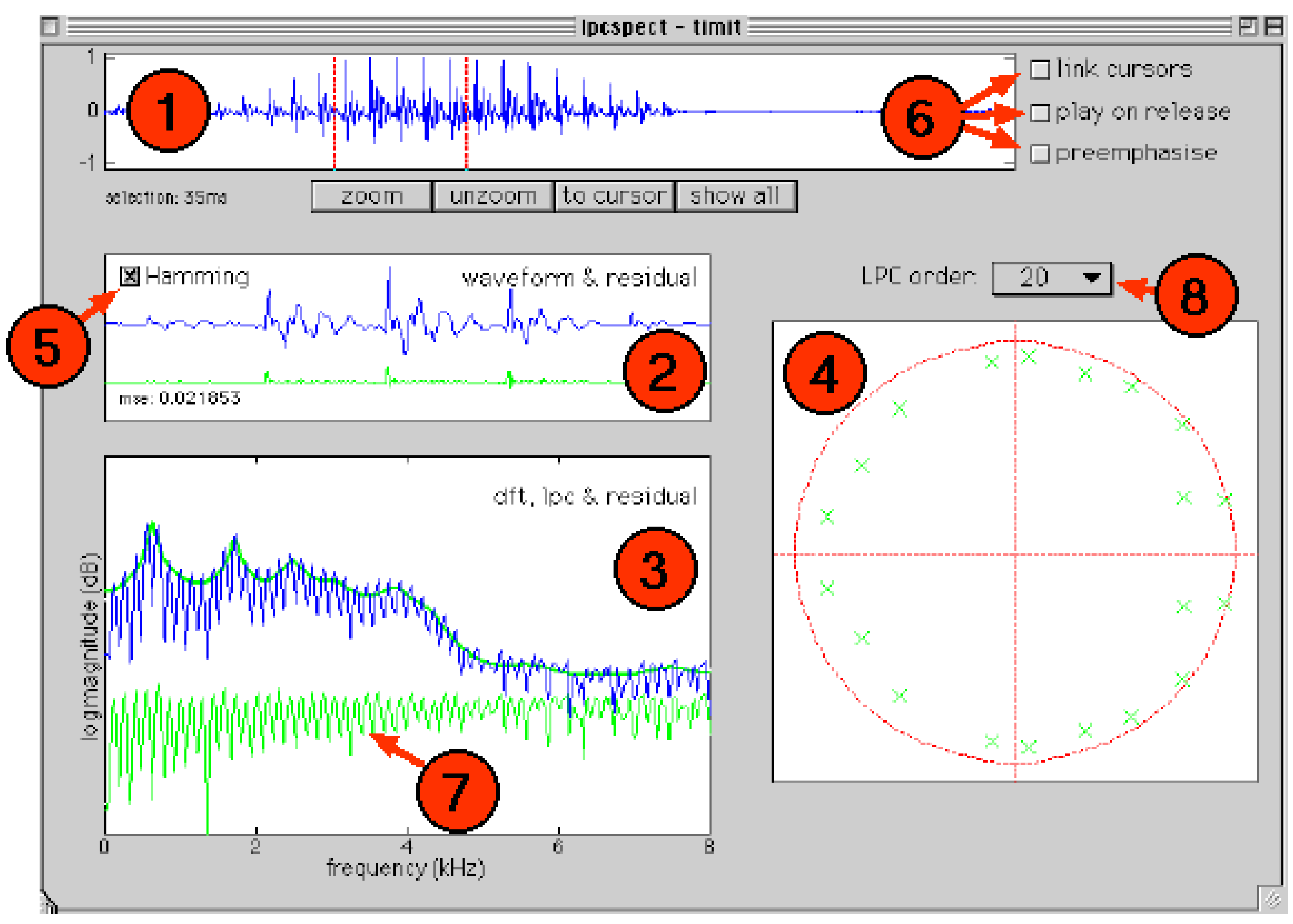


linear prediction of speech

Introduction

This tool allows users to explore linear prediction of speech and other signals. It also reinforces the concepts of windowing, preemphasis and frame size selection in speech analysis. Users can select regions for analysis and see the results as lpc-smoothed spectra and poles. The residual signal and its spectrum is also displayed. This tool is not a substitute for learning about linear prediction. To understand what the various panels mean, you should read one of the texts listed below. The demo requires the MATLAB signal processing toolbox.

The tool



Type 'lpcspect' to launch the demo. The *file* menu is used to load an existing signal file or to create a new one using the createsig tool. Supported formats currently include .wav, .snd and .au sound files. The signal will appear in panel (1), and associated linear prediction signals for the region between the cursors will be displayed in the other panels. Use the cursors to select any segment. On cursor release, the segment appears in panel (2) along with linear prediction residual (error signal). Panel (3) shows the DFT spectrum of this segment, overlaid with the LPC-smoothed spectrum and the DFT of the residual (7). The poles corresponding to the linear predictor are shown in panel (4).

The LPC order can be changed using menu (8). The signal can be preemphasised by checking the relevant box in cluster (6). Cursors can be linked (so that a constant segment length is maintained). The waveform segment chosen will be played on release if the associated checkbox is checked.

Clicking on any signal waveform (incuding the residual) causes it to be played. Holding the mouse button down in the z-plane panel (4) will cause the frequency at the selected z-plane angle to be displayed in the top left corner of the panel.

The chosen waveform segment can be Hamming-windowed (5). Unchecked, a rectangular window is applied.

Things to investigate

1. Choose a vowel segment of speech. Examine what happens to the residual signal (and its mean square error) as you increase the number of poles. What happens to the LPC-smoothed spectrum? How many poles provide a comfortable fit to a vowel segment.
2. Now repeat this analysis for unvoiced sounds such as fricatives. How does the error compare?
3. Again, perform this analysis for a nasal sound.
4. Can you predict where the poles will appear for a given analysis order, and how they will change when you modify the order?
5. Explore the effect of preemphasis on the LPC spectrum and the number of poles required to obtain a given residual error.
6. Listen to the waveform segment and the residual signal in sequence. Can you detect a change in timbre? How does this change manifest itself as the LPC order increases?
7. What is the effect of changing LPC order on the spectrum of the residual signal?
8. Choose a vowel segment and a relatively low LPC order (10, say). Read off the pole locations (frequencies) by holding the mouse down in panel (4). Now use the polezero tool to recreate just these pole locations and listen to the resulting sound.
9. Use the *create* option on the file menu to create a harmonic series with, say, 5 components. Do you expect to be able to fit it well with a 10th order model? Try it.
10. Now add some noise (using *create*) to your harmonic series. Do the pole locations change? (Read about the peak-hugging properties of LPC).

Further reading

- A good introduction to LPC is provided by Makhoul, J. (19??)
- This tool was written partly as an example of the interface issues in speech and hearing demonstrations, and is described further in Cooke et al (1999) The interactive auditory demonstrations project, Eurospeech'99, Budapest, September.

Credits etc

Produced by: Martin Cooke, April 1999.

FILTER DESIGN

"All applets on our web site now require the java plugin to run. This is necessary so we can bring state-of-the-art features to you which are not currently supported by browser vendors. You can find the plugin at http://java.sun.com/products/plugin. For additional information or suggestions please contact help@isip.msstate.edu"
No JDK 1.2 support for APPLET!!

- Source Code: Download the source code for this applet.
- Tutorial: Learn how to use this applet.