Speech Analysis

Tony Robinson

Lent Term 1998

Contents

1 Introduction

These notes are available as:
`http://svr-www.eng.cam.ac.uk/~ajr/SpeechAnalysis`

1.1 What is Speech Analysis?

- Analysis of speech sounds taking into consideration their method of production
- The level of processing between the digitised acoustic waveform and the acoustic feature vectors.
- The extraction of "interesting" information as an acoustic vector.

1.1.1 So what is an acoustic vector?

A representation of the speech sound at that time. For example:

- The short-term power spectra
- A representation of the vocal tract shape
- An estimation of the formant frequencies and bandwidths

These exist as there are limitations on the rate of speech production, (thus the fundamental information transfer rate) and this is less than a general signal at the same sampling rate. Speech analysis deals with time-scales of around 20ms.

1.2 Why Speech Analysis?

- Established layer in recognition, synthesis and coding
- Model estimation: needed low dimensional vectors
- Psychoacoustic results - don't need full waveform
- Analysis for analysis sake - learning about speech
- Linguistics
- Pathology

For example, to make sense of speech waveforms such as figure 1 – or zooming in, figure 2, which are the waveforms for the words "She had your dark".

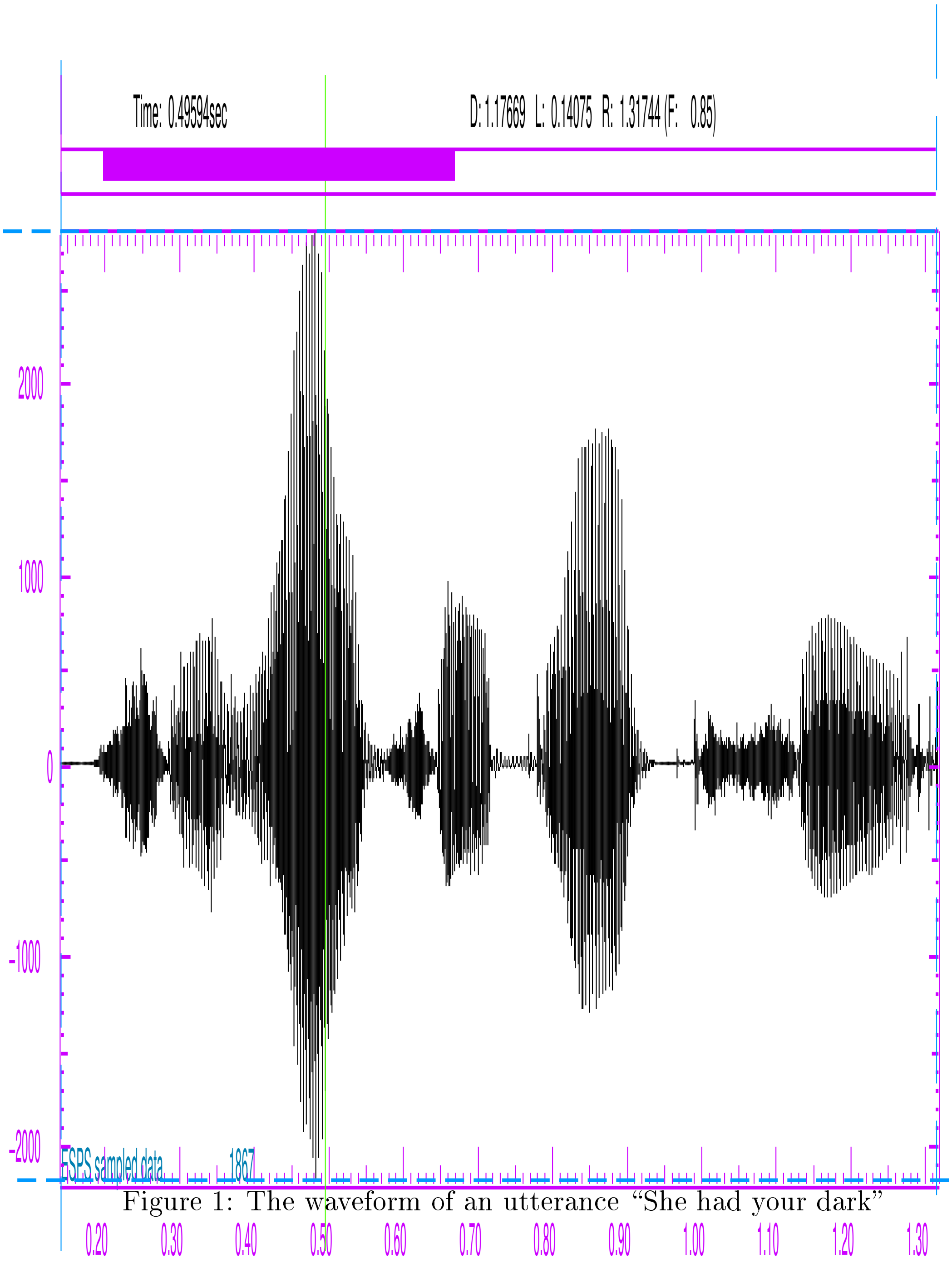

Figure 1: The waveform of an utterance “She had your dark”

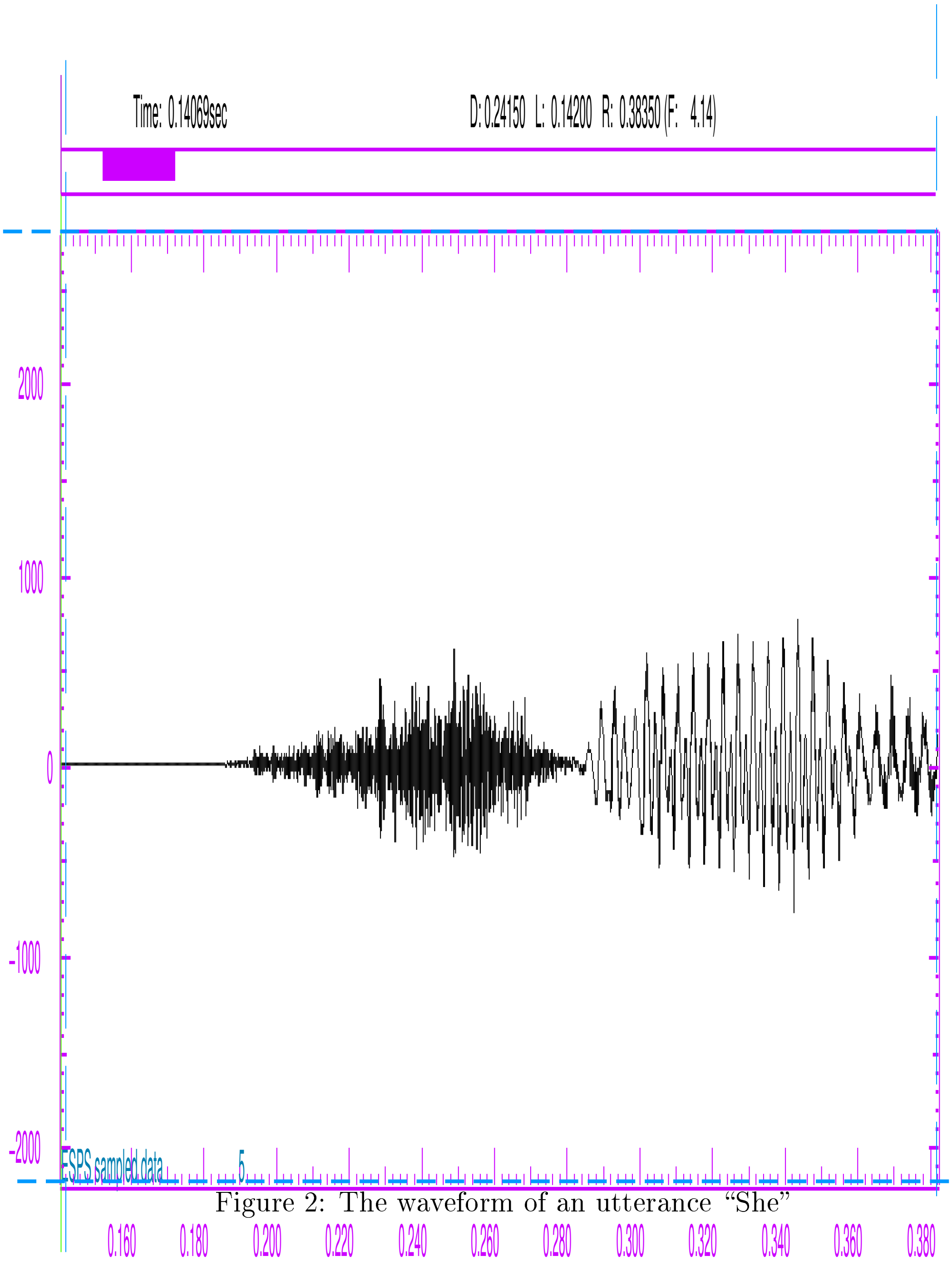

Figure 2: The waveform of an utterance "She"

1.3 The problems of speech analysis

- The assumption that speech is short time stationary
- The formulation of a feature vector representation that captures the important information in the speech signal for future processing

1.4 Standard references for this course

The published books on speech analysis tend to provide a lot of mathematical detail. An old, established and still very useful book is [1]. Two chapters of [2] cover this material well. More recently published is [3] which both covers this course and is useful for other courses. The book [4] covers low level signal processing, filtering, FFTs etc.

2 Background

2.1 Sampling theory

To convert an analogue signal to a digital form it must first be band-limited then sampled:

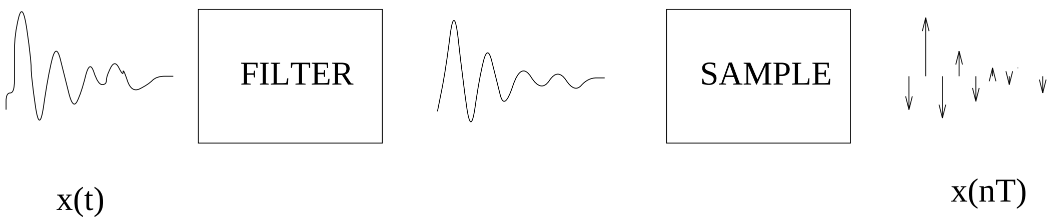

Figure 3: The digitisation process

2.1.1 Sampling frequency

Signals must be filtered prior to sampling. Theortically the maximum frequency that can be represented is half the sampling frequency. In practice a higher sample rate is used to allow for non-ideal filters.

The signal is now represented at multiples of the sampling period, T, as $s(nT)$ which is also written s_n.

Telephone speech is sampled at 8 kHz. 16 kHz is generally regarded as sufficient for speech recognition and synthesis. The audio standard is a sample rate of 44.1 kHz (Compact Disc) or 48 kHz (Digital Audio Tape) to represent frequencies up to 20 kHz.

2.1.2 Sampling resolution

- The dynamic range of the ear (inaudible to painful) is about 20 bits (i.e. ± 1 to $\pm 10^6$).

Sound level/ dB SPL	Power ratio	Amplitude ratio	Typical example
140	10^{14}	10^7	Gunshot at close range
120	10^{12}	10^6	Loud rock group
100	10^{10}	10^5	Shouting at close range
80	10^8	10^4	Busy street
70	10^7	3160	Normal conversation
50	10^5	316	Quiet conversation
30	10^3	31.6	Soft whisper
20	10^2	10	Country area at night
6.5	4.5	2.1	Mean absolute threshold @ 1kHz
3	1	1	Reference level

Table 1: Taken from [5] page 9

- Normally stored in 16 bits, of which 12 are significant. Telephony quality speech is sampled at 8kHz with a 12 bit dynamic range (stored in 8 bits with a non-linear squashing function).

2.2 Linear filters

2.2.1 Finite Impulse Response filters

A Finite Impulse Response (FIR) filter produces an output, $y(n)$, that is the weighted sum of the current and past inputs, $x(n)$.

$$\begin{aligned} y_n &= b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} + \ldots + b_q x_{n-q} && (1) \\ &= \sum_{j=0}^{q} b_j x_{n-j} && (2) \end{aligned}$$

This is shown in figure 4 with z^{-1} representing a unit delay.
Consider supplying this filter with a sine wave, $x_n = \sin(\omega n T)$:

$$y_n = \sum_{j=0}^{q} b_j \sin(\omega(n-j)T) \qquad (3)$$

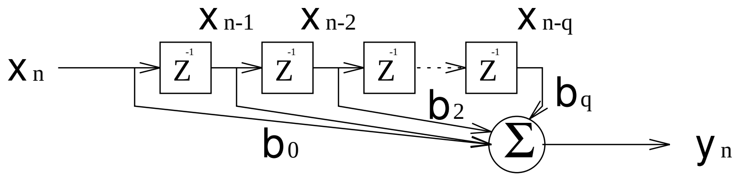

Figure 4: A FIR filter

Using the identity $\sin(\theta+\phi) = \sin\theta\cos\phi + \cos\theta\sin\phi$:

$$
\begin{aligned}
y_n &= \sum_{j=0}^{q} b_j \left(\sin(\omega nT)\cos(-\omega jT) + \cos(\omega nT)\sin(-\omega jT)\right) && (4)\\
&= \left(\sum_{j=0}^{q} b_j \cos(-\omega jT)\right)\sin(\omega nT) + \left(\sum_{j=0}^{q} b_j \sin(-\omega jT)\right)\cos(\omega nT) && (5)
\end{aligned}
$$

The terms in parantheses are independent of time and hence the output is a sinusoid with amplitude:

$$\sqrt{\left(\left(\sum_{j=0}^{q} b_j \cos(-\omega jT)\right)^2 + \left(\sum_{j=0}^{q} b_j \sin(-\omega jT)\right)^2\right)}$$

and phase:

$$\tan^{-1}\left(\sum_{j=0}^{q} b_j \sin(-\omega jT) / \sum_{j=0}^{q} b_j \cos(-\omega jT)\right)$$

This method may be used to provide the amplitude and phase response for any FIR filter. The transform is called the Fourier transform (defined in section 4), and it has a simple inverse. Conversely the filter coefficents may be obtained from the desired filter response using the same technique.

As a simple example, consider a low pass filter where the desired response, $H(\omega)$ is:

$$H(\omega) = \begin{cases} 1 & 0\omega \leq \omega_c \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

$$b_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(\omega) e^{in\omega} d\omega \quad (7)$$

$$= \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{in\omega} d\omega \quad (8)$$

$$= \frac{1}{2\pi}\left[\frac{1}{in}e^{in\omega}\right]_{-\omega_c}^{\omega_c} \quad (9)$$

$$= \frac{1}{\pi}\sin(n\omega_c)/n \quad (10)$$

But this means we need an infinite number of filter coefficients! True enough – real "brick wall" filters are impossible and sharp filters are hard to design. Some solutions:

- truncate: simple and effective if cycles are short
- window: use the Hamming window - minimises the power in the side-lobes
- Use a more complex filter design package, for example Parks-McClelland Remez Exchange algorithm, which designs optimal zero-phase FIR filters with arbitrary frequency responses.

Matlab implements all these, for example "fir1(14, 0.5)" is a 15 tap low pass filter that has a cutoff at half the maximum frequency. The filter coefficents are a windowed sinc funtion, plotted in figure 5 and the amplitude response is plotted in figure 6.

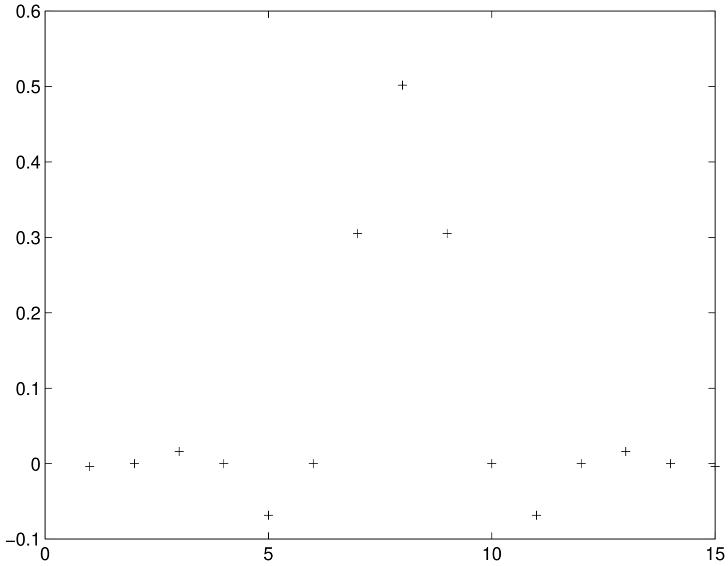

Figure 5: Example filter coefficiants: plot(fir1(14, 0.5), '+')

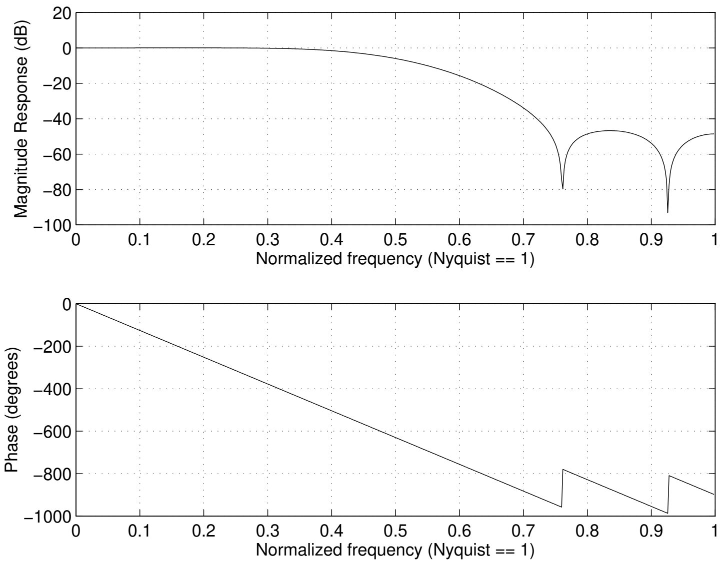

Figure 6: Example filter response: freqz(fir1(14, 0.5))

FIR filters are computationally expensive to implement but need not introduce phase distortions - useful in processing high quality speech.

IIR filters are often much more efficient, but can not be designed to have exact linear phase.

2.2.2 Infinite Impulse Response filters

An Infinite Impulse Response (IIR) filter produces an output, $y(n)$, that is the weighted sum of the current and past inputs, $x(n)$, and past outputs.

The Linear Predictive model is a specical case of an IIR filter and shown in figure 7.

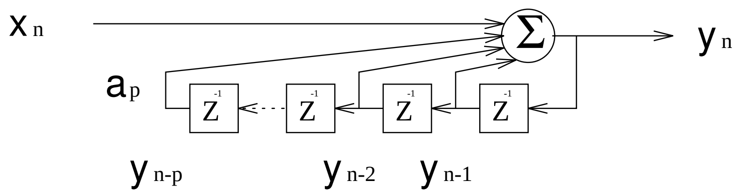

Figure 7: An IIR filter

The general IIR filter (figure 8) is given by:

$$y_n = \sum_{i=1}^{p} a_i y_{n-i} + \sum_{j=0}^{q} b_j x_{n-j} \tag{11}$$

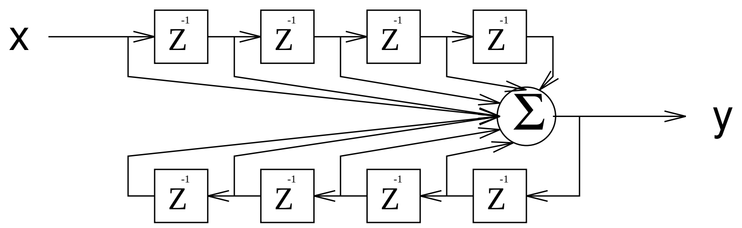

Figure 8: The general linear filter

If $p = 0$ then the system represents a finite impulse response (FIR) filter. If p is not zero, then the system is an infinite impulse response (IIR) filter.

An example is the two pole resonator with center frequency ω and bandwidth related to r is:

$$y_n = 2r\cos(\omega T)y_{n-1} - r^2 y_{n-2} + x_n - \cos(\omega T)x_{n-1} \tag{12}$$

Common types of IIR filter:

Type	Characteristics
Butterworth	maximally flat amplitude
Bessel	maximally flat group delay
Chebyshev	equiripple in passband or stop-band
Elliptic	equiripple in passband and stop-band

For more details see [4].

2.3 The source filter model of speech

Assume a Source Filter model of speech as in figure 9.

Speech is split into a rapidly varying excitation signal and a slowly varying filter. The envelope of the power spectra contains the vocal tract information.

3 Filter bank Analysis

We can now assemble a set of band pass filters to analyse speech. These need to be covering - that is every frequency is covered by one filter so no information is lost.

An example of one filter is figure 10:

The output is a waveform, but as the phase is not so important we need the magnitude or the energy - how?

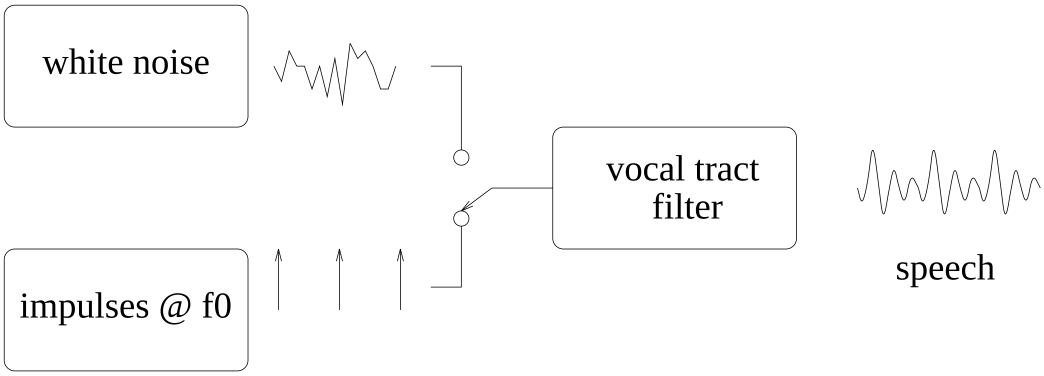

Figure 9: The source-filter model of speech production

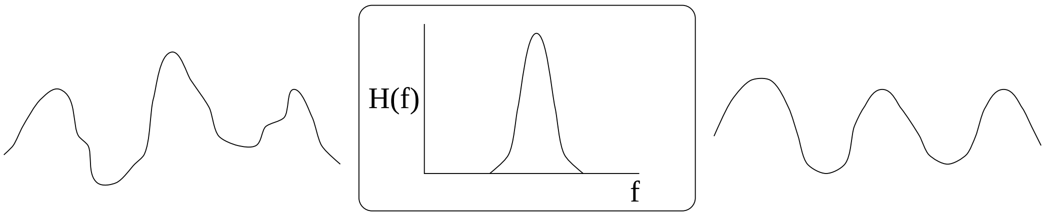

Figure 10: A band pass filter

- Rectify and smooth - convenient for hardware implementations
- Square and smooth - gives a better estimation of the power
- Hilbert transforms - generate a signal that is 90 degrees out of phase, then square both signals and add:

$$
\begin{aligned}
& (a\sin(\omega t + \phi))^2 + (a\sin(\omega t + \phi + \pi/2))^2 && (13)\\
= \; & a^2(\sin(\omega t + \phi))^2 + a^2(\cos(\omega t + \phi))^2) && (14)\\
= \; & a^2 && (15)
\end{aligned}
$$

If the filter bank is implemented as an FIR filter, a simple transform of the coefficients yields the phase shifted signal (use sine coefficients on a_i instead of cosine coefficients).

3.1 Spectrograms

Can plot the short-term power in different frequency bands as a function of time - called the spectrogram. For example, figure 11.

Originally implemented by recording onto magnetic tape, then passing the tape over a rapidly spinning drum contain the playback head (one loop

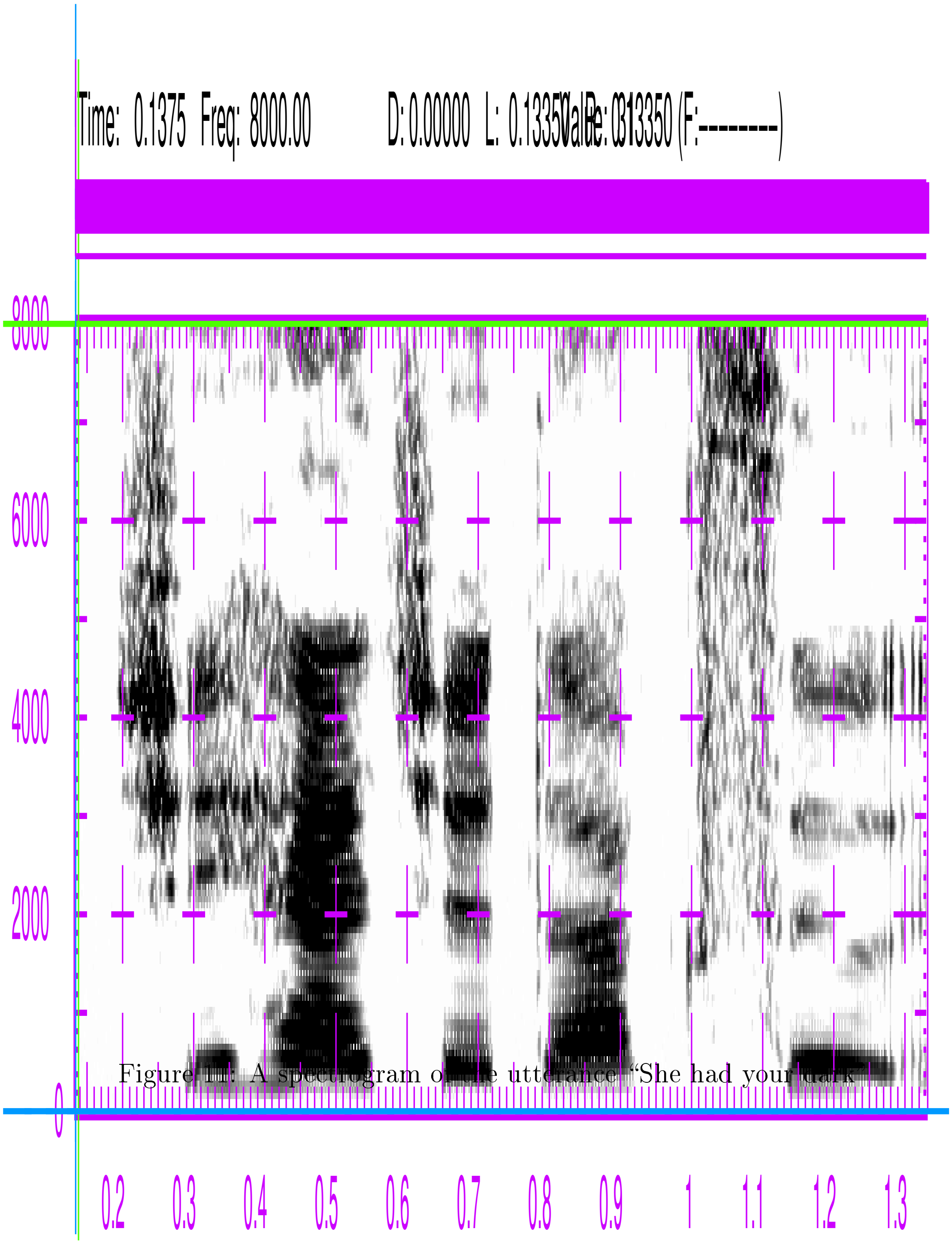

Figure 1.1: A spectrogram of the utterance "She had your dark"

around was the "window"), the resulting signal was fed to a bandpass filter whose centre frequency was determined by the position of a stylus on head sensitive paper - the resulting signal being used to mark the paper with the intensity of the signal at that frequency.

Serious choice of filter bandwidth:

- Narrow band analysis: resolves harmonics but blurs temporal detail such as burst onsets.
- Wide band analysis: resolves fine temporal detail but looses fine frequency detail.

In most applications this tradeoff is manifested as the choice of window size. Good rule of thumb is to include a few pitch periods, say 20ms to 32ms.

3.2 Non-linear frequency scales

There are many non-linear frequency scales that approximate the sensitivity of the human ear. For example:

- Constant Q: Q is the ratio of filter bandwidth over centre frequency, hence this implies a exponential form.
- Equivalent Rectangular bandwidth (ERB): The bandwidths of the auditory filters are measured
- Bark: Also derived from perception experiments
- Basilar membrane: The distance measured along the basilar membrane.
- Mel: The engineers solution

In practice, there is not much to chose from, unless you are interested in auditory model research.

The mel scale is common amongst engineers as it has a simple analytical form:

$$m = 1125 \log(0.0016 f + 1) \tag{16}$$

where f is the frequency in Hz and m is the resulting mel scaled frequency.

For example, this may look like: Filter specifications for a 19 - channel vocoder (Holmes 1980).

Another reason to use a non-linear frequency analysis is to get around the frequency/time resolution tradeoff. Using a narrow bandwidth at low frequencies enables harmonics to be resolved but gives poor onset information. Using a larger bandwidth at higher frequencies allows for high temporal resolution of bursts, etc.

Channel number	Centre frequency/Hz	Bandwidth
1	240	120
2	360	120
3	480	120
4	600	120
5	720	120
6	840	120
7	1000	150
8	1150	150
9	1300	150
10	1450	150
11	1600	150
12	1800	200
13	2000	200
14	2200	200
15	2400	200
16	2700	300
17	3000	300
18	3300	300
19	3700	500

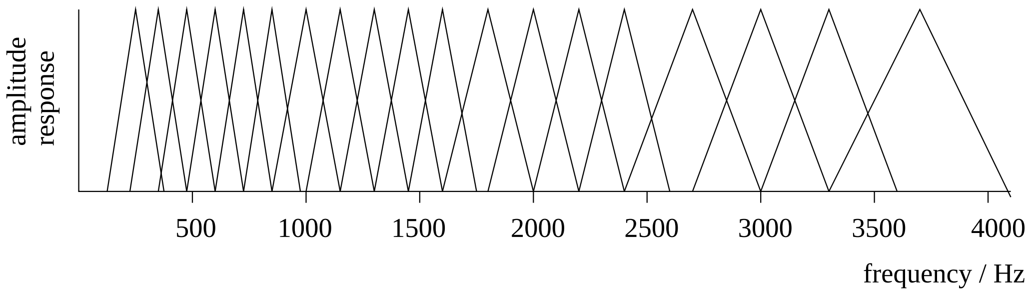

Figure 12: The Holmes filter bank

4 Short-Term Fourier Analysis

The discrete Fourier transform (DFT) is defined as:

$$X(e^{i\theta}) = \sum_{n=-\infty}^{n=\infty} x(n)e^{-in\theta} \tag{17}$$

where:

$$\begin{aligned}\theta &= 2\pi fT &(18)\\ &= 2\pi f/f_s &(19)\end{aligned}$$

Where T is the sampling period and f_s is the sampling frequency.

The inverse transform is defined by:

$$x(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{i\theta})e^{in\theta}d\theta \tag{20}$$

Note that $X(e^{i\theta})$ is continuous - that is θ can take on any real value in the range 0 to 2π.

$X(e^{i\theta}) = X(e^{i(\theta+2\pi)})$, the DFT is periodic in θ with period 2π – and therefore periodic in f with period f_s.

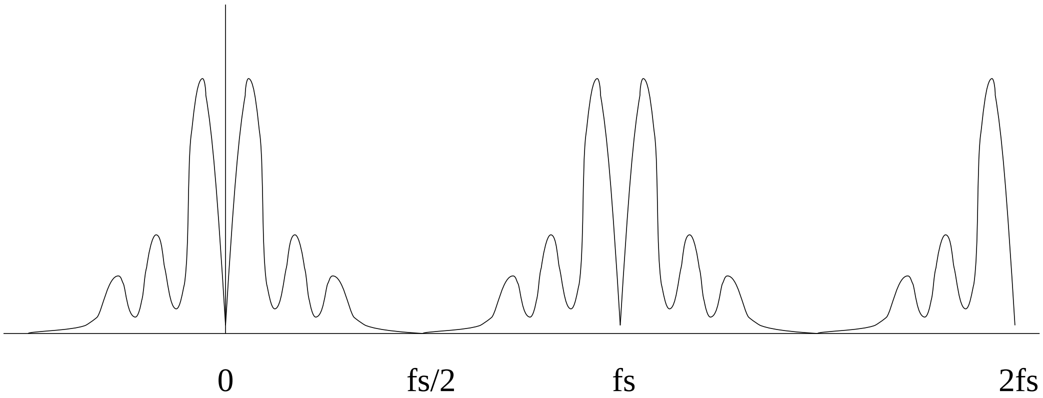

Figure 13: A DFT illustrating the periodic nature

The amplitude spectrum is the magnitude of each component in the DFT, $|X(e^{i\theta})|$. The power spectrum is the square of the components in the amplitude spectrum:

$$S(e^{i\theta}) = |X(e^{i\theta})|^2 \tag{21}$$

4.1 Properties of the DFT

4.1.1 Linearity

$$\alpha h(n) + \beta g(n) \quad \Leftrightarrow \quad \alpha H(e^{i\theta}) + \beta G(e^{i\theta}) \tag{22}$$

That is, the Fourier transform of a signal $h(n)$ multiplied by a scalar α is the Fourier transform of the signal, $H(e^{i\theta})$, multiplied by α. Also, the Fourier transform of the sum of two signals, $h(n)$ and $g(n)$, is the sum of the Fourier transforms of both signals.

4.1.2 Time Shift

$$h(n+k) \quad \Leftrightarrow \quad H(e^{i\theta})e^{i\theta k} \tag{23}$$

That is, if a signal is shifted by k, the discrete Fourier transform is phase shifted by θk.

4.1.3 Frequency Shift

$$h(n)e^{in\theta_0} \quad \Leftrightarrow \quad H(e^{i(\theta-\theta_0)}) \tag{24}$$

Similarly, multiplying the signal by $e^{in\theta_0}$ introduces a frequency shift of θ_0 in the Fourier transform.

4.1.4 Convolution

$$h(n) * g(n) \quad \Leftrightarrow \quad H(e^{i\theta})\ G(e^{i\theta}) \tag{25}$$

$$h(n)\ g(n) \quad \Leftrightarrow \quad H(e^{i\theta}) * G(e^{i\theta}) \tag{26}$$

Convolving two signals in the time domain results in a Fourier transform that is the multiplication of the Fourier transform of the two original signals. Similarly, multiplication of two signals in the time domain results in a convolution in the frequency domain.

This is important later when $g(n)$ is a windowing function.

4.1.5 Real valued input

if $h(n)$ is real, as in most speech processing, $H(e^{i\theta})$ is symmetric. Most useful.

Similarly, if $h(n)$ is even, i.e. $h(n) = h(-n)$ then $H(e^{i\theta})$ is real.

Fourier transforms of real valued data can be done roughly twice as fast (only half the complex numbers are present) – look for realft in Numerical recipes.

4.2 Windowing

For speech processing we want to assume the signal is short-time stationary and perform a Fourier transform on these small blocks. Solution: multiple the signal by a window function that is zero outside some defined range.

The rectangular window is defined as:

$$w_n = \begin{cases} 1 & 0 \leq n < N \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

But consider the discontinuities this can generate, as illustrated in figure 14. One way to avoid disconitinuities at the ends is to taper the signal

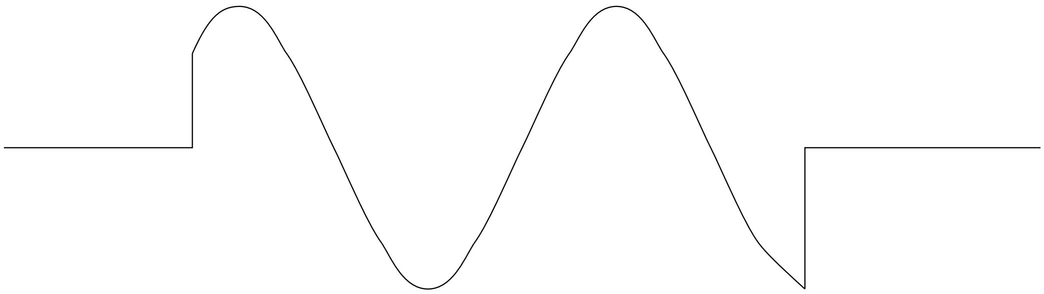

Figure 14: A waveform truncated with a rectangular window

to zero or near zero and hence reduce the mismatch.

The most common in speech analysis is the Hamming window:

$$w_n = \begin{cases} 0.54 - 0.46\cos(2\pi n/(N-1)) & 0 \leq n < N \\ 0 & \text{otherwise} \end{cases} \tag{28}$$

This is simply a raised cosine and is plotted out in figure 15.

As stated in section 4.1.4, multiplication of the signal by a window function in the time domain is the same as convolving the signal in the frequency domain.

Rectangular window gives maximum sharpness but large side-lobes (ripples) - hamming window blurs in frequency but produces much less leakage. For example:

```
n = 512;
m = 128;
h = hamming(m);
sr = zeros(size(1:n));
sh = zeros(size(1:n));
```

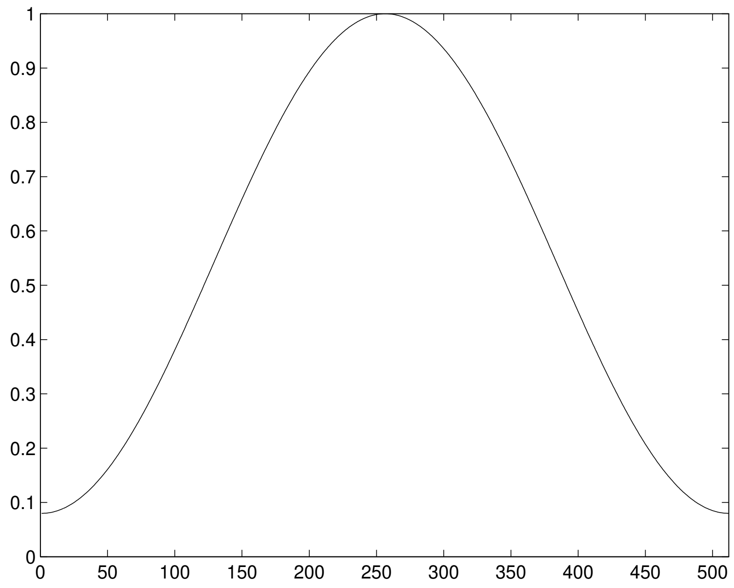

Figure 15: The hamming window

```
f = pi / 3;

for i = 1:m ; sr(i) =        sin(f * i) ; end
for i = 1:m ; sh(i) = h(i) * sin(f * i) ; end
ar=abs(fft(sr));
ah=abs(fft(sh));

plot(ar(1:n/2))
plot(20 * log10(ar(1:n/2)))
plot(20 * log10(ah(1:n/2)))
```

Figure 16 shows the reactangular and hamming windowed sine wave. Figure 17 shows the magnitude of the FFT and Figure /reffigwinLog the magnitude on a dB scale.

4.3 The short-term Fourier transform

First extract N samples of speech starting at N_0 and then window:

$$x'_n = \begin{cases} w_n x_{N_0+n} & 0 \leq n < N \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

Now apply the DFT.

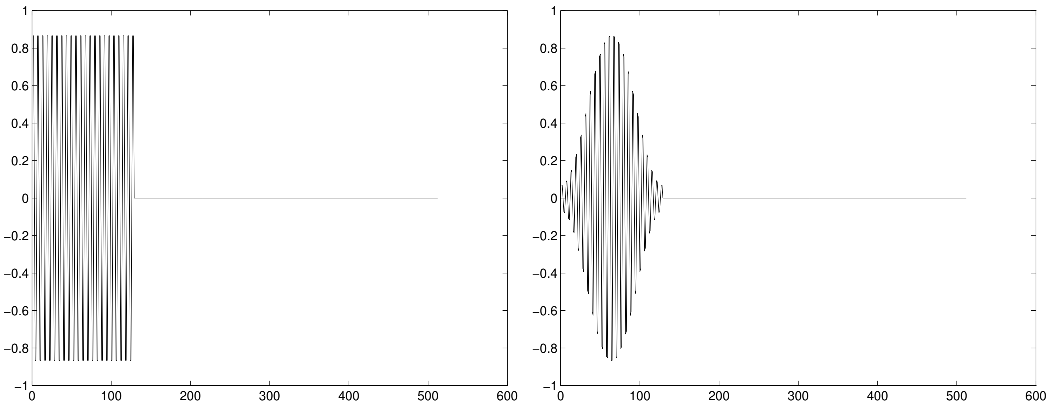

Figure 16: rectangular and Hamming windowed sine wave

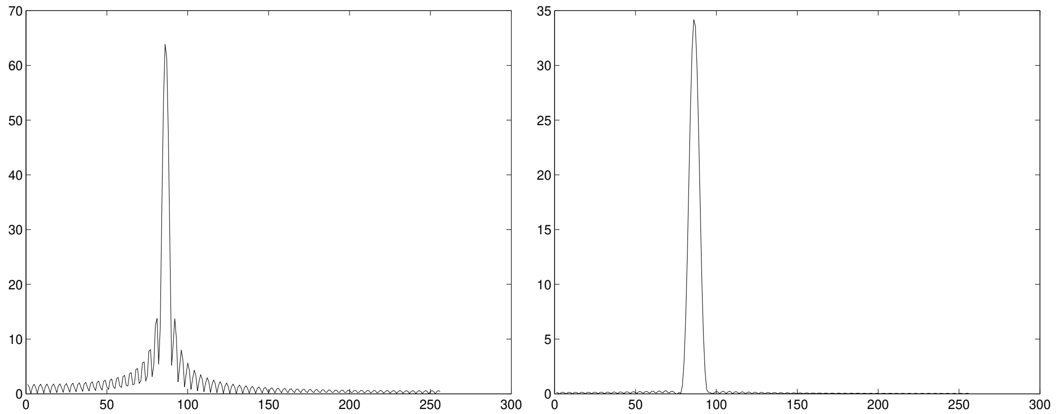

Figure 17: FFT of rectangular and Hamming windowed sine wave

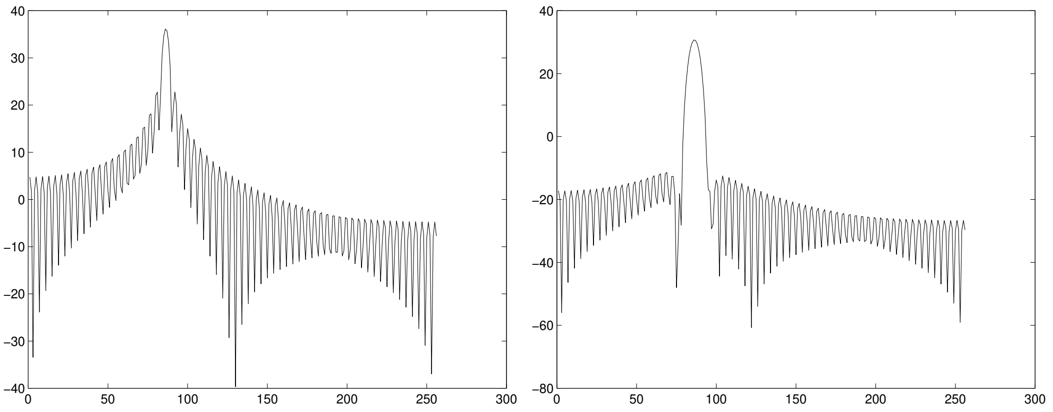

Figure 18: FFT of rectangular and Hamming windowed sine wave in dB

The result is continuous, but it can be sampled at n points without loss of information.

$$
\begin{aligned}
\theta_k &= \frac{2\pi k}{N \qquad 0 \leq k \leq N} && (30)\\
X_k &= X(e^{i\frac{2\pi k}{N}}) && (31)\\
&= \sum_{n=0}^{N-1} x_n e^{-i\frac{2\pi n k}{N}} && (32)
\end{aligned}
$$

This equation defines an algorithm that takes an array of N complex numbers (or equivalently an array of N real numbers and N imaginary numbers) and returns an array of N complex numbers.

The inverse short-term DFT can be similarly defined:

$$
x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{i\frac{2\pi n k}{N}} \qquad (33)
$$

4.4 Zero padding

- We force the signal to be zero outside the window
- Can explicity add zeros
- More frequency domain points (but not more resolution)

4.5 Fast Fourier transforms

A simple interpretation requires $O(N^2)$ operations. The FFT requires $O(N \log N)$ if N is a factor of two. In general, FFT style techniques can be used for any N – firstly N is decomposed into prime factors and then a recursive "butterfly" operation is performed for each factor.

The "Decimation in Time" algorithm (Cooley and Turkey, 1965):

Start with the top level Fourier transform described by:

$$
X_1(k) = \sum_{n=0}^{N-1} x_n e^{-i2\pi nk/N}, \qquad k = 0, \ldots, N-1 \qquad (34)
$$

Now write $X_1(k)$ as a summation of odd terms and even terms:

$$
X_1(k) = \sum_{n=0}^{N/2-1} x_{2n} e^{-i2\pi(2n)k/N} + \sum_{n=0}^{N/2-1} x_{2n+1} e^{-i2\pi(2n+1)k/N} \qquad (35)
$$

$$= \sum_{n=0}^{N/2-1} x_{2n} e^{-i2\pi(2n)k/N} + e^{-i2\pi k/N} \sum_{n=0}^{N/2-1} x_{2n+1} e^{-i2\pi(2n)k/N} \quad (36)$$

$$= \sum_{n=0}^{N/2-1} x_{2n} e^{-i2\pi nk/(N/2)} + e^{-i2\pi k/N} \sum_{n=0}^{N/2-1} x_{2n+1} e^{-i2\pi nk/(N/2)} \quad (37)$$

$$= X_{11}(k) + e^{-i2\pi k/N} X_{12}(k) \quad (38)$$

That is, the original Fourier transform has been written in terms of two Fourier transforms operating on the odd and even portions of the data.

This recursion can be continued until the trivial case of one point is reached.

4.6 Practical application of the short-term Fourier transform

Take 512 samples of speech, as in figure 19:

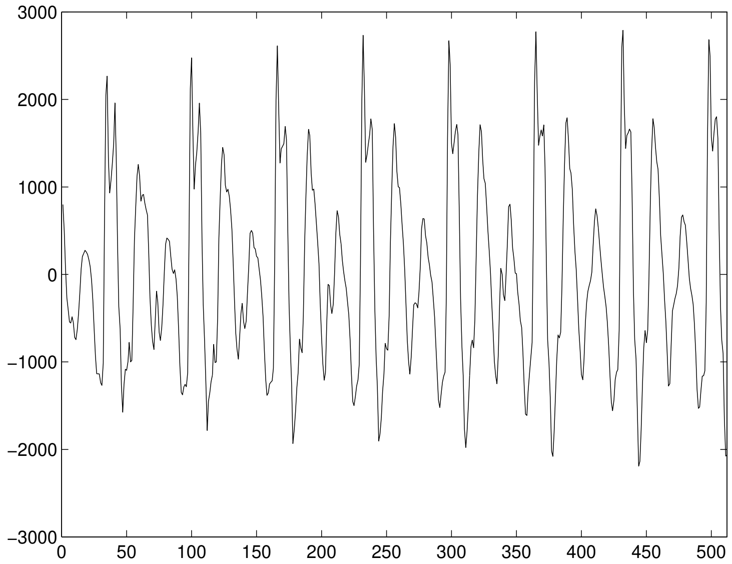

Figure 19: A segment of a vowel extracted with a rectangular window

Perform FFT and plot the magnitude of the result:

As can be seen from figure 19, there is a sharp discontinuity at the ends. Application of the Hamming window reduces this discontinuity:

And as a result the harmonic structure of the speech is more visible:

This is the basis for most computer generated spectrograms (display pixel intensity on a log scale by limiting the dynamic range to about 60-80 dB).

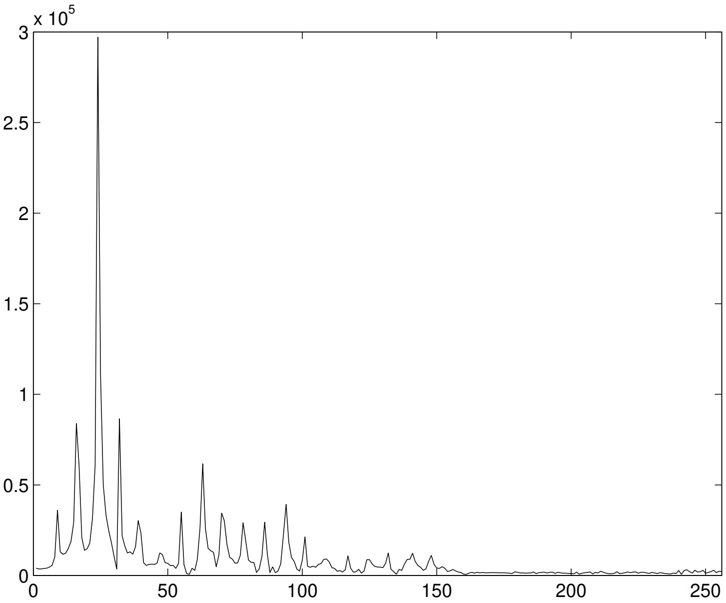

Figure 20: The amplitude spectrum using a rectangular window Calculated using Matlab: `abs(fft(sig))`

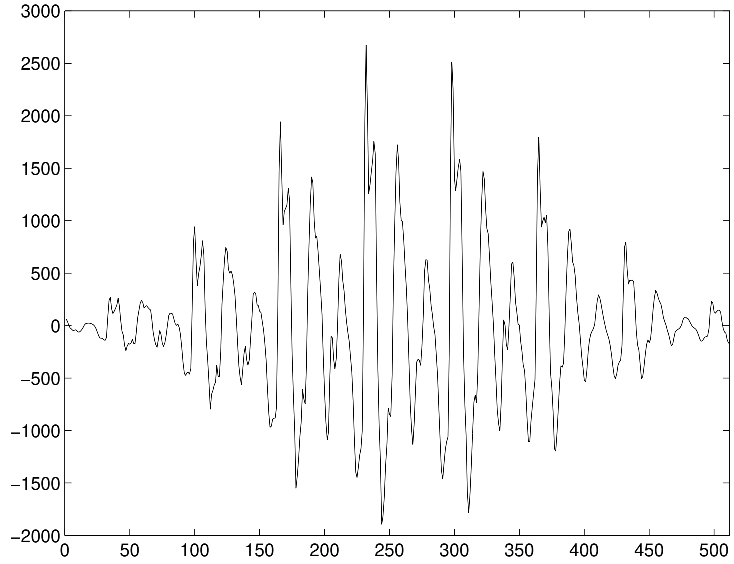

Figure 21: A segment of a vowel extracted with a hamming window. Calculated using Matlab: `hamming(512) .* sig`

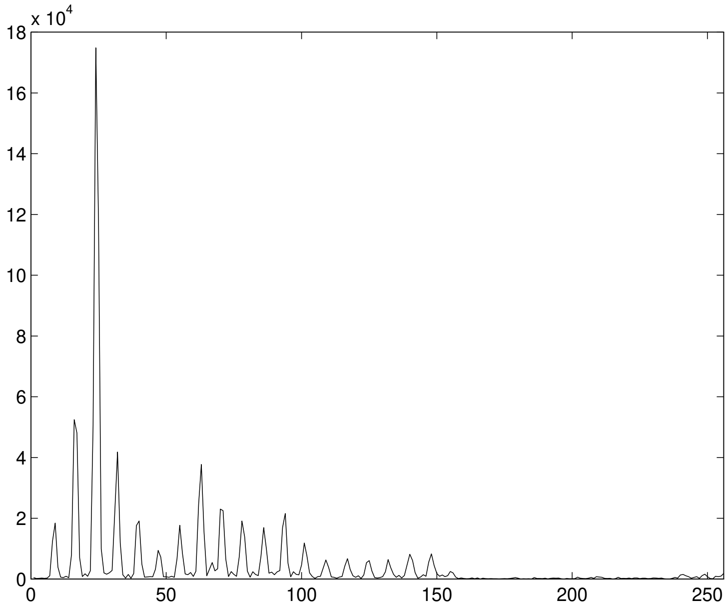

Figure 22: The amplitude spectrum using a hamming window. Calculated using Matlab: `abs(fft(hamming(512) .* sig))`

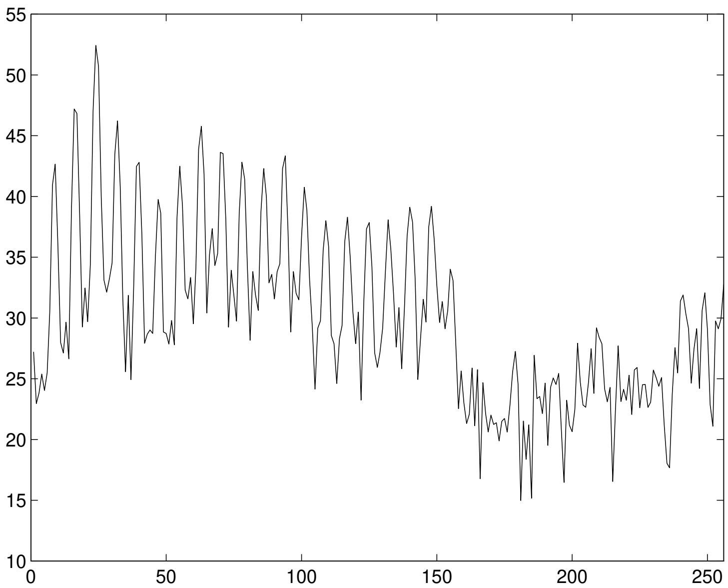

Figure 23: The power spectrum displayed on a dB scale. Calculated using Matlab: `10 log10(abs(fft(hamming(512) .* sig)))`

To illustrate the effect of window size the above analysis has been repeated using a hamming window of length 64. The speech is illustrated in figure reffig:hammsigshort which shows that less than one pitch period is included in the analysis window. Hence it is not possible to resolve the harmonics in the DFT, but only display the broad formant patterns of the speech (figure 25).

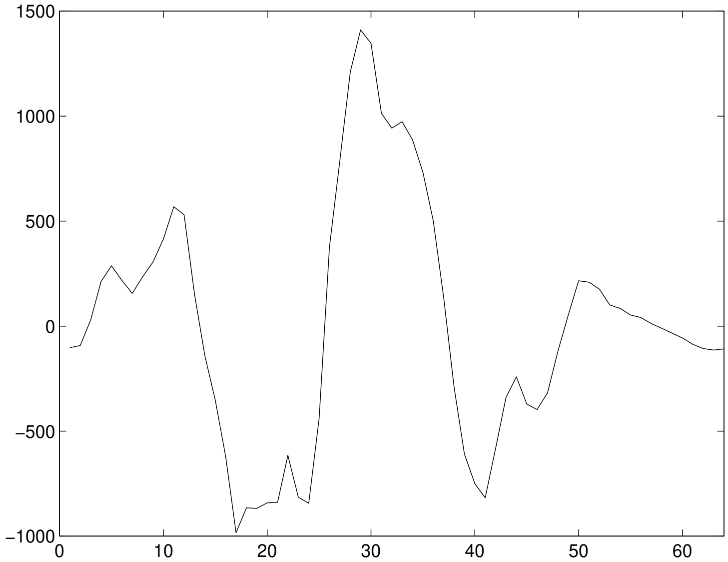

Figure 24: A segment of a vowel extracted with a hamming window of length 64

4.7 Overlap and add for linear filtering

The DFT followed by the inverse DFT results in an exact reconstruction of the original signal. This provides an easy way to implement a digital filter, as illustrated in figures 26 and 27.

A FIR linear filter would apply a constant multiplicative weighting in the frequency domain. For filter lengths greater than about 60 samples it is faster to use the FFT than direct convolution. Note that zero padding of the input is needed to ensure that there are no wrap-around effects.

This framework also allows for more general filtering where the filter may be time dependent or dependent on the analysed signal.

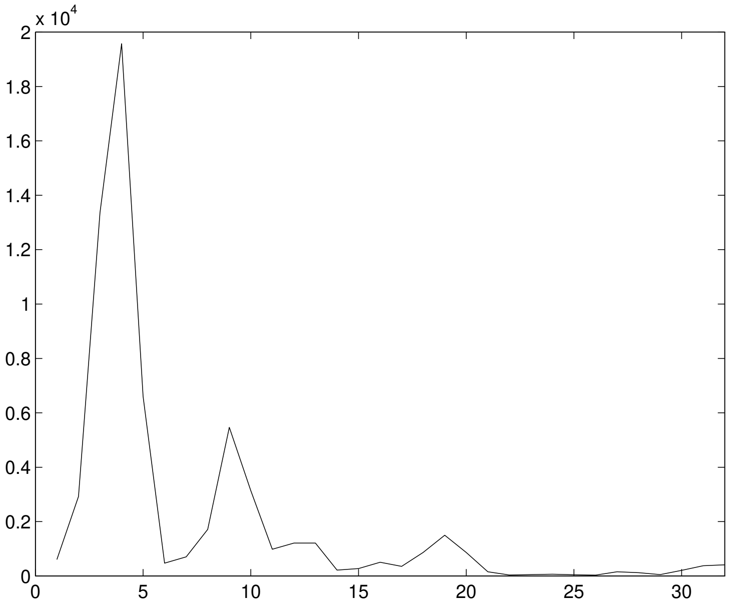

Figure 25: The amplitude spectrum using a hamming window of length 64

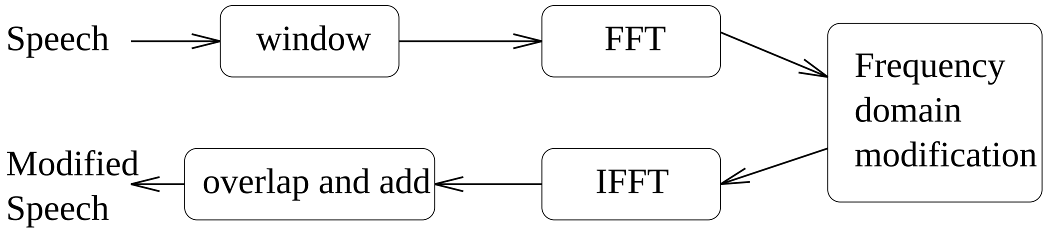

Figure 26: The overlap and add method for linear filtering

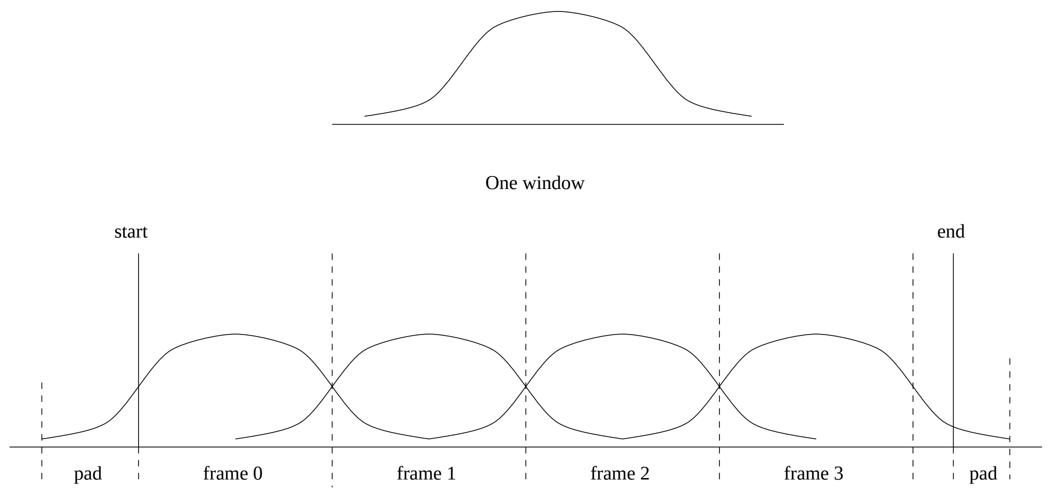

Figure 27: overlap and add method in the time domain

4.7.1 Example: Spectral subtraction

A speech signal, s, is corrupted by additive noise, n. The combined signal, $s+n$, is observed and an estimate of the noise amplitude spectrum, $|N(e^{i\theta})|$ is also available. An estimate of the original signal amplitude spectrum, $|\hat{S}(e^{i\theta})|$, is given by:

$$|\hat{S}(e^{i\theta})| = |S(e^{i\theta}) + N(e^{i\theta})| - |\hat{N}(e^{i\theta})| \tag{39}$$

That is, the estimated amplitude of the signal at a given frequency is the real amplitude of the signal plus noise at that frequency minus the estimated noise amplitude. The phase of the signal is assumed to be the same as that of the signal plus noise. This modification of the frequency domain information can be placed within the overlap and add framework to give the technique know as spectral subtraction [6, 7].

There are several problems with this method.

- It is quite likely that equation 39 results in a negative estimate of $|\hat{S}(e^{i\theta})|$. This is not physical and a simple solution is to set such values to zero.
- There are invariably large differences between $|N(e^{i\theta})|$ and $|\hat{N}(e^{i\theta})|$. If $|N(e^{i\theta})| >> |\hat{N}(e^{i\theta})|$ then considerable noise remains.
- All channels are processed independently: This often leads to one channel of one frame having a significant spuriously high amplitude. The resulting signal has a short tonal burst and this problem is referred to as that of "musical tones".

Lots of fancy tricks (i.e. tweaks) can be added to this basic method, for example the subtraction of an overestimate of the noise signal or the use of median filtering to help suppress the musical tones [6, 7]

4.8 Cepstral analysis

The source filter model of speech production decomposes the speech signal, s_n, into an excitation, e_n, and a linear filter, $H(e^{i\theta})$. In the frequency domain:

$$S(e^{i\theta}) = H(e^{i\theta})E(e^{i\theta}) \tag{40}$$

We wish $H(e^{i\theta})$ to represent the envelope of the speech power spectra and $E(e^{i\theta})$ to represent the fine detail of the excitation. For example, see figure 22 and figure 23. With a suitable definition of the log of a complex number ($\log z = \log |z| + i \arg\{z\}$) this may be achieved with:

$$\log(S(e^{i\theta})) = \log(H(e^{i\theta})) + \log(E(e^{i\theta})) \tag{41}$$

For most speech processing applications we require only the amplitude spectra, hence the equation is written:

$$\log(|S(e^{i\theta})|) \quad = \quad \log(|H(e^{i\theta})|) + \log(|E(e^{i\theta})|) \tag{42}$$

The slowly varying components of $\log(|S(e^{i\theta})|)$ are represented by the low frequencies and the fine detail by the high frequencies. Hence another Fourier transform is the natural way to separate the components of $H(e^{i\theta})$ and $E(e^{i\theta})$. This produces the cepstral analysis, shown diagrammatically in figure 28.

Speech → window → DFT → Log → IDFT → Cepstrum

Figure 28: Cepstral analysis

For the example speech of figures 21,22,23 the resulting (real) cepstral analysis is shown in figure 29.

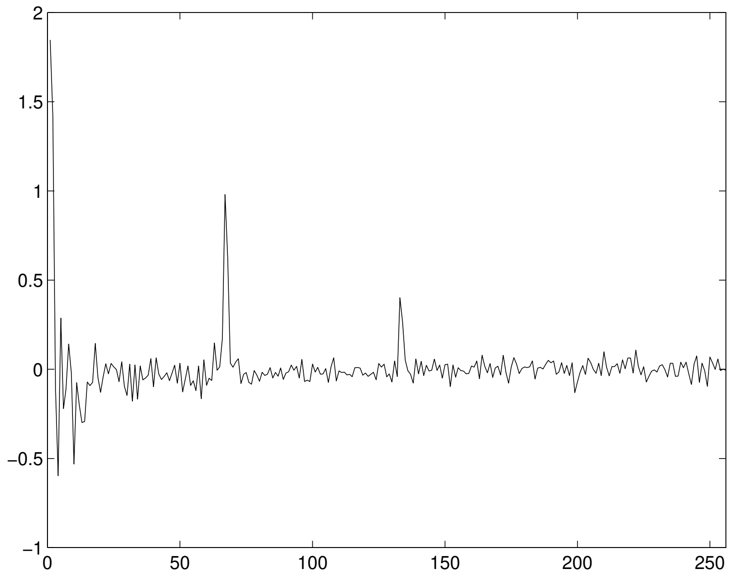

Figure 29: The full real cepstrum. Calculated using Matlab: `ifft(log(abs(fft(hamming(512) .* sig))))`

It can be seen that most of the detail occurs near the origin and in peaks higher up the cepstrum. Thus the lower numbered coefficients provide the envelope information. The remainder of the detail is mostly contained in the peaks which are separated by the pitch period (in this case about 70 sample) and provide the fine detail pitch information.

An enlargement of the few samples is shown in figure 30

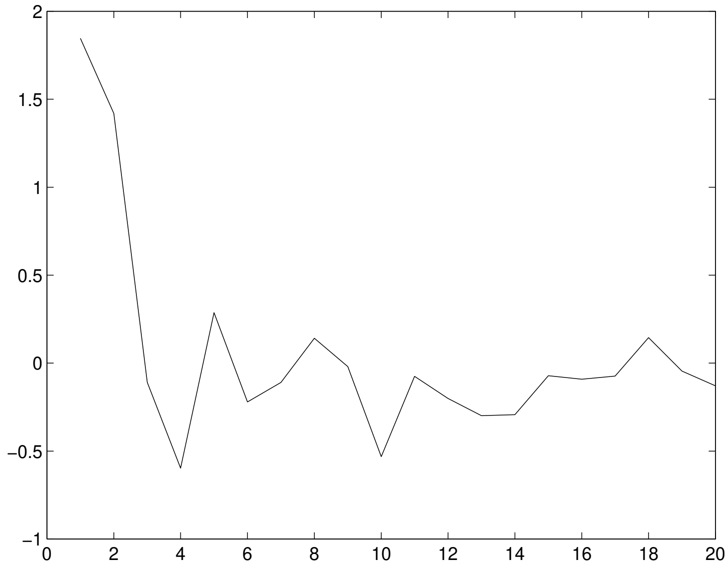

Figure 30: The first 20 cepstral coefficients

4.9 Homomorphic filtering

Cepstral analysis provides a method for separating out the vocal tract information from the excitation information. Thus the reverse transformation can be carried out to provide a smoothed power spectrum. This process is known as Homomorphic filtering and the result is illustrated in figure 31.

4.10 Mel scaled analysis

All the above analysis techniques place equal emphasis on every part of the frequency scale from zero up to the maximum representable frequency. As discussed in section 3, it is often better to place the bins on a non-linear frequency scale. Figure 32 shows the result of warping 23 onto the mel scale.

Note that these plots have only shown part of the spectrum – as illustrating in figure 13 the DFT is periodic. To transform a plot such as figure 32 to the cepstral domain involves either reflecting about zero frequency or using the Discrete Cosine Transform.

4.11 The Autocorrelation from the FFT

It can be shown that the Fourier transform of the power spectrum is the autocorrelation function. In applications where the full autocorrelation function is needed it may be faster to use this method that the direct computation.

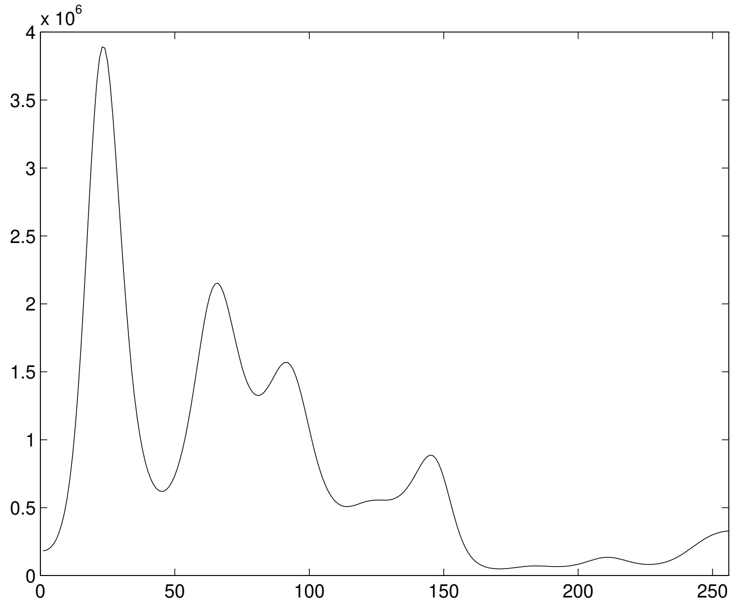

Figure 31: Homomorphic filtering of the amplitude spectrum

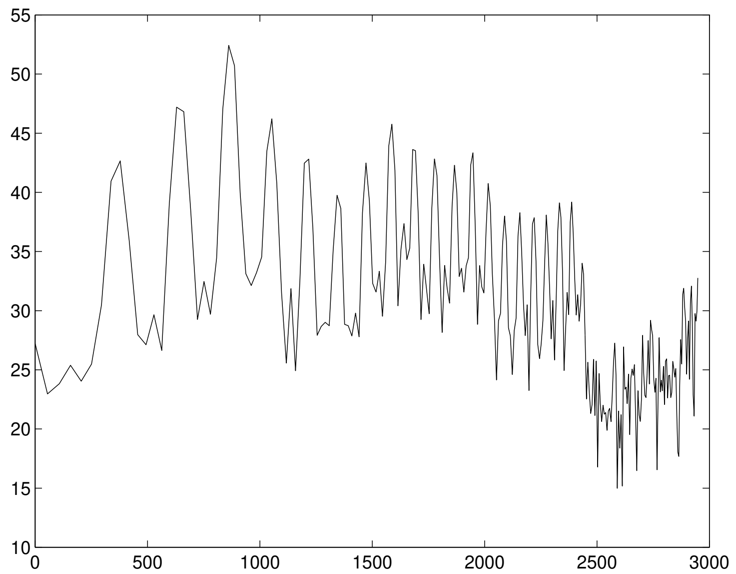

Figure 32: Mel scaled log power spectrum

The result for the example speech is shown in figure 33.

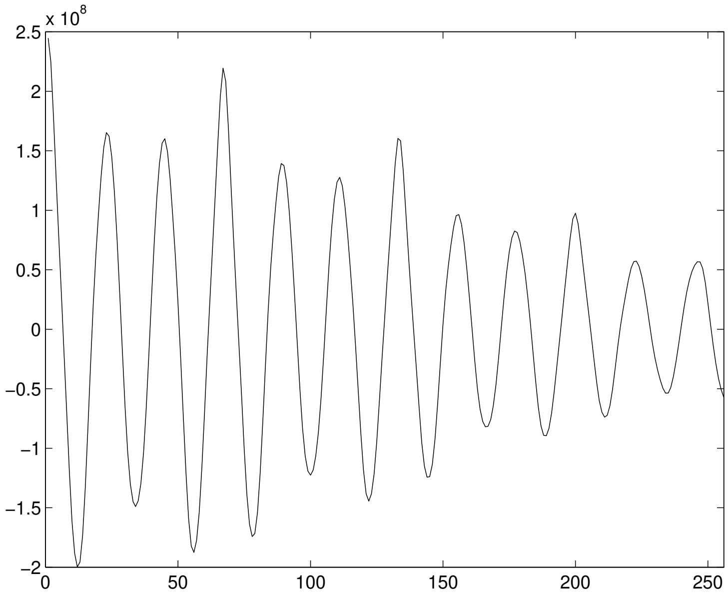

Figure 33: The autocorrelation function via the DFT

5 Z transforms

The z-transform is defined by:

$$X(z) = \sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{43}$$

The sequence, $x(n)$ is known and z is a complex number. Hence $X(z)$ is just a weighted sum. For example, for the sequence: $x(0) = 1, x(1) = 3, x(2) = 3, x(3) = 1$ and $x(n) = 0$ otherwise

$$X(z) = 1 + 3z^{-1} + 3z^{-2} + z^{-3} \tag{44}$$

and evaluating this at a particular point, e.g. $z = i/2$

$$\begin{aligned} X(i/2) &= 1 + 3(i/2)^{-1} + 3(i/2)^{-2} + (i/2)^{-3} && (45) \\ &= 1 + 3(-2i) + 3(-4) + (8i) && (46) \\ &= -11 + 2i && (47) \end{aligned}$$

Only defined for values of z where the series converges.

That is, z-transform is the general version of the discrete Fourier transform. To obtain the Fourier restrict z to lie on the unit circle $z = e^{i\theta} = \cos\theta + i\sin\theta$.

There are several ways of obtaining the inverse z transform:

a) By inspection: if $X(z)$ can be written as a simple polynomial in z then the time domain sequence is the coeffients of the polynomial

b) By expansion: expanding $X(z)$ as a polynomial in z

c) By decomposition: breaking up $X(z)$ into parts whose inverse z transforms are known (e.g. see table 3.1 in [4])

d) By definition: the inverse transform is defined by:

$$x(n) = \frac{1}{2\pi i} \oint_C X(z) z^{n-1} dz \tag{48}$$

Where C is a closed contour that includes $z = 0$.

The z transform is a linear transform, i.e.

$$y(n) = \alpha x_1(n) + \beta x_2(n) \tag{49}$$

$$Y(z) = \alpha X_1(z) + \beta X_2(z) \tag{50}$$

So, if $y(n)$ is the convolution of two signals, $h(n)$ and $x(n)$, i.e.:

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k) \tag{51}$$

$$= x(n) * h(n) \tag{52}$$

then

$$Y(z) = H(z)X(z) \tag{53}$$

The linear filters of section 2.2 can now be expressed in terms of z-transforms.

The general linear filter is expressed as:

$$Y(z) = H(z)X(z) \tag{54}$$

where $H(z)$ is called the "system function" and is the z-transform of the unit sample response.

For the FIR filter of order q:

$$y(n) = \sum_{r=1}^{q} b_r x_{n-r} \tag{55}$$

$$Y(z) = H(z)X(z) \tag{56}$$

$$H(z) = \sum_{r=0}^{q} b_k z^{-k} \tag{57}$$

Similarly for the IIR filter:

$$H(z) = \frac{\sum_{r=0}^{q} b_r z^{-r}}{\sum_{k=0}^{p} a_k z^{-k}} \tag{58}$$

This is useful as $H(z)$ can be factored:

$$H(z) = \frac{A \prod_{r=1}^{q}(1 - c_r z^{-1})}{\prod_{k=1}^{p}(1 - d_k z^{-1})} \tag{59}$$

From this equation it can be seen that if $(1 - c_r z^{-1}) = 0$ then the filter will have zero response – these are the "zeros" of the linear system.

Similarly, $(1 - d_k z^{-1}) = 0$ defines the "poles" of the linear system. When $q = 0$, as in linear prediction, we have an "all pole" filter.

For a stable system, all the poles must lie within the unit circle. An un-

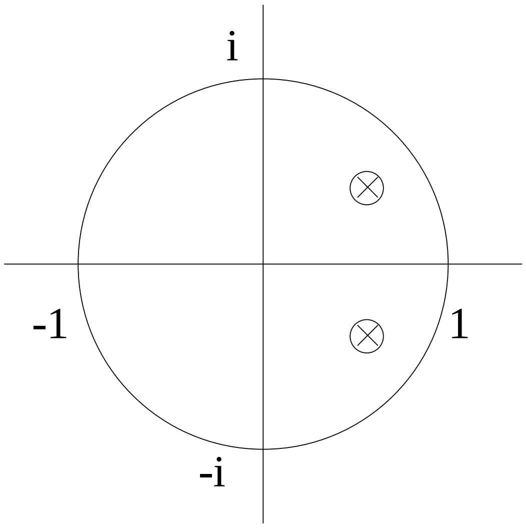

Figure 34: An argand diagram showing a stable pole-pair within the unit circle

stable system is one whose output is unbounded in response the unit impulse.

Manipulation of the form of $H(z)$ allows many different implementations. For example, as the coefficients a_i and b_i are real, the poles and zeros occur in complex conjugate pairs. By grouping these together $H(z)$ can be expressed in terms of second order sections:

$$H(z) \quad = \quad A \prod_{k=1}^{p} \frac{1 + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}} \tag{60}$$

This "cascade form" is illustrated in figure 35.

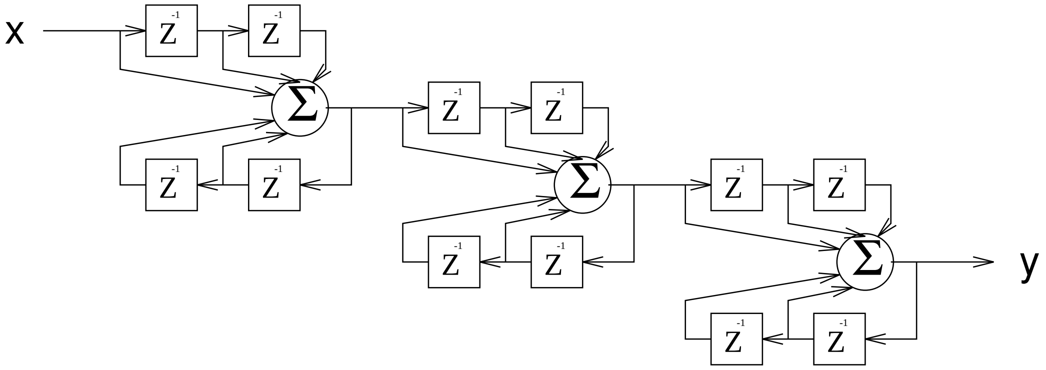

Figure 35: The cascade form for a linear filter

It is also possible to expand $H(z)$ in terms of partial fractions:

$$H(z) \quad = \quad \sum_{k=1}^{p} \frac{c_{0k} + c_{1k} z^{-1}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}} \tag{61}$$

This "parallel form" is illustrated in figure 36.

Both forms are popular in speech synthesis – indeed the Klatt synthesiser has both a parallel and a cascade path (for ease of specifying the coefficients I assume).

6 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:

$$\hat{s}_n \quad = \quad \sum_{i=1}^{p} a_i s_{n-i} \tag{62}$$

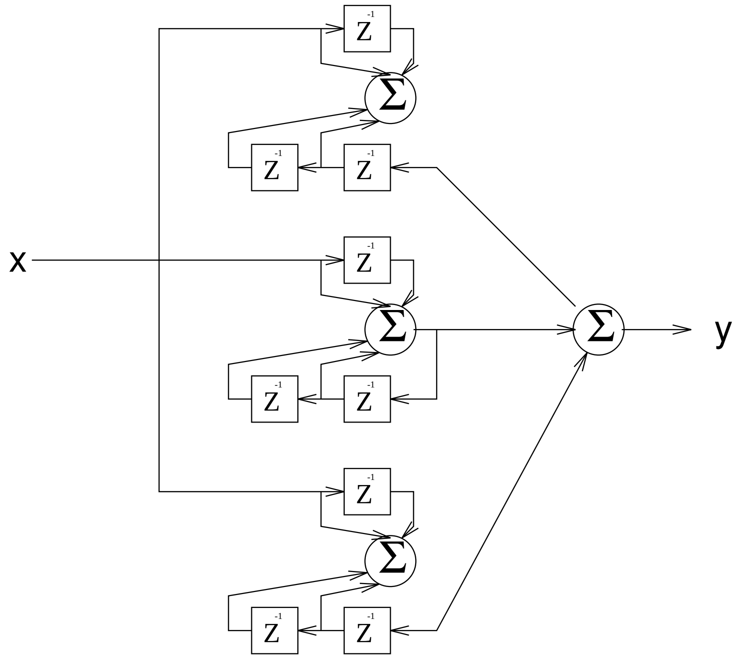

Figure 36: The parallel form for a linear filter

This linear filter has the transfer function:

$$H(z) \quad = \quad \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} \tag{63}$$

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

6.1 Motivation from lossless tubes

The transfer function of a lossless tube can be described by an all pole model. This is also a reasonable approximation to speech formed by the excitation of the vocal tract by glottal pulses (although the glottal pulses are not spectrally flat).

But:

- The vocal tract is not built of cylinders
- The vocal tract is not lossless
- The vocal tract has a side passage (the nasal cavity)
- fricatives (e.g. /s/ and /sh/) are generated near the lips

Nevertheless, with sufficient parameters the LP model can make a reasonable approximation to the spectral envelope for all speech sounds.

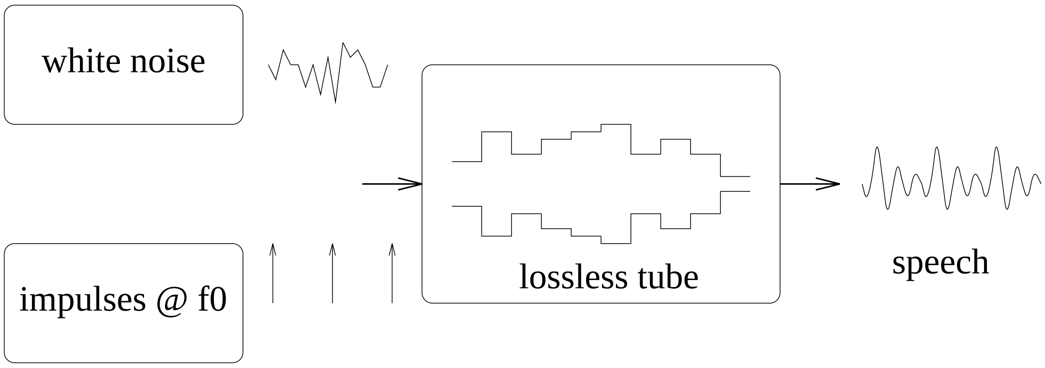

Figure 37: The lossless tube model of speech production

6.2 Parameter estimation

Given N samples of speech, we would like to compute estimates to a_i that result in the best fit. One reasonable way to define "best fit" is in terms of mean squared error. These can also be regarded as "most probable" parameters if it is assumed the distribution of errors is Gaussian and a priori there were no restrictions on the values of a_i.

The error at any time, e_n, is:

$$\begin{aligned} e_n &= s_n - \hat{s}_n && (64) \\ &= s_n - \sum_{i=1}^{p} a_i s_{n-i} && (65) \end{aligned}$$

Hence the summed squared error, E, over a finite window of length N is:

$$\begin{aligned} E &= \sum_{n=0}^{N-1} e_n^2 && (66) \\ &= \sum_{n=0}^{N-1} (s_n - \sum_{k=1}^{p} a_k s_{n-k})^2 && (67) \end{aligned}$$

The minimum of E occurs when the derivative is zero with respect to each of the parameters, a_k. As can be seen from equation 67 the value of E is quadratic in each of the a_k therefore there is a single solution. Very large positive or negative values of a_k must lead to poor prediction and hence the solution to $\partial E / \partial a_k = 0$ must be a minimum.

Hence differentiating equation 67 with respect to a_j and setting equal to zero gives the set of p equations:

$$\frac{\partial E}{\partial a_j} = 0 \quad = \quad - \sum_{n=0}^{N-1} \left(2(s_n - \sum_{k=1}^{p} a_k s_{n-k}) s_{n-j} \right) \qquad (68)$$

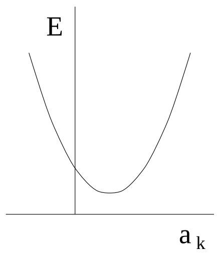

Figure 38: Schematic showing single minimum of a quadratic

$$= \quad -2\sum_{n=0}^{N-1} s_n s_{n-j} + 2\sum_{n=0}^{N-1}\sum_{k=1}^{p} a_k s_{n-k} s_{n-j} \tag{69}$$

rearranging equation 69 gives:

$$\sum_{n=0}^{N-1} s_n s_{n-j} \quad = \quad \sum_{k=1}^{p} a_k \sum_{n=0}^{N-1} s_{n-k} s_{n-j} \tag{70}$$

Define the covariance matrix, $\mathbf{\Phi}$ with elements $\phi_{i,k}$:

$$\phi_{i,k} \quad = \quad \sum_{n=0}^{N-1} s_{n-i} s_{n-k} \tag{71}$$

Now we can write equation 70 as:

$$\phi_{i,0} \quad = \quad \sum_{k=1}^{p} \phi_{i,k} a_k \tag{72}$$

or in matrix form:

$$\begin{pmatrix} \phi_{1,0} \\ \phi_{2,0} \\ \phi_{3,0} \\ \cdots \\ \phi_{p,0} \end{pmatrix} = \begin{pmatrix} \phi_{1,1} & \phi_{1,2} & \phi_{1,3} & \cdots & \phi_{1,p} \\ \phi_{2,1} & \phi_{2,2} & \phi_{2,3} & \cdots & \phi_{2,p} \\ \phi_{3,1} & \phi_{3,2} & \phi_{3,3} & \cdots & \phi_{3,p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_{p,1} & \phi_{p,2} & \phi_{p,3} & \cdots & \phi_{p,p} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \cdots \\ a_p \end{pmatrix} \tag{73}$$

or simply:

$$\mathbf{\Phi_0} \quad = \quad \mathbf{\Phi a} \tag{74}$$

Hence the *Covariance method* solution is obtained by matrix inverse:

$$\mathbf{a} = \mathbf{\Phi}^{-1}\mathbf{\Phi_0} \tag{75}$$

Note that $\mathbf{\Phi}$ is symmetric, i.e. $\phi_{i,k} = \phi_{k,i}$, and that this symmetry can be expoited in inverting $\mathbf{\Phi}$ (see [9]).

These equations reference the samples $s_{-p} \ldots s_{-1}$.

6.3 The autocorrelation method

When dealing with windowed speech we need to take into account the boundary effects in order to avoid large prediction errors at the edges. Can refine the area over which we perform least squares minimisation in equation 67 and make use of the fact that samples are zero outside of the window to rewrite $\phi_{i,j}$ as:

$$\phi_{i,j} = \sum_{n=0}^{N-1-(i-j)} s_n s_{n+(i-j)} \tag{76}$$

Now $\phi_{i,j}$ is only dependent on the difference, $i - j$, and may be written in terms of the autocorrelation function, $\phi_{i,j} = r_{i-j}$:

$$r_k = \sum_{n=0}^{N-1-k} s_n s_{n+k} \tag{77}$$

Now $\mathbf{\Phi}$ is Toeplitz:

$$\begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ \cdots \\ r_p \end{pmatrix} = \begin{pmatrix} r_0 & r_1 & r_2 & \cdots & r_{p-1} \\ r_1 & r_0 & r_1 & \cdots & r_{p-2} \\ r_2 & r_1 & r_0 & \cdots & r_{p-3} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{p-1} & r_{p-2} & r_{p-3} & \cdots & r_0 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \cdots \\ a_p \end{pmatrix} \tag{78}$$

Efficient methods exist to invert such matrices, one of which is Durbin's algorithm. Denoting the values of the LP parameters at iteration i by $a_k^{(i)}$ and the residual energy by $E^{(i)}$ ($E^{(0)} = r_0$) for i = 1, 2, ...

$$k_i = \left(r_i - \sum_{j=1}^{i-1} a_j^{(i-1)} r_{i-j} \right) / E^{(i-1)} \tag{79}$$

$$a_i^{(i)} = k_i \tag{80}$$

$$a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)} \qquad 1 \leq j < i \tag{81}$$

$$E^{(i)} = (1 - k_i^2) E^{(i-1)} \tag{82}$$

For example, for the signal of figure 21:

$$\begin{aligned} r_0 &= 2.4470\ 10^8 \\ r_1 &= 2.2466\ 10^8 \\ r_2 &= 1.7823\ 10^8 \end{aligned}$$

Therefore on the first iteration:

$$\begin{aligned} k_1 &= r_1/E^{(0)} \\ &= 0.9181 \\ a_1^{(1)} &= k_1 \\ &= 0.9181 \\ E^{(1)} &= (1-k_1^2)E^{(0)} \\ &= (1-0.9181 * 0.9181)2.4470\ 10^8 \\ &= 0.38442\ 10^8 \end{aligned}$$

And on the second iteration:

$$\begin{aligned} k_2 &= \left(r_2 - a_1^{(1)} r_1\right)/E^{(1)} \\ &= \left(1.7823\ 10^8 - 0.9181 * 2.2466\ 10^8\right)/0.38442\ 10^8 \\ &= -0.72915 \\ a_2^{(2)} &= k_2 \\ &= -0.72915 \\ a_1^{(2)} &= a_1^{(1)} - k_2 a_1^{(1)} \\ &= 0.9181 - -0.729150.9181 \\ &= 1.58753 \\ E^{(2)} &= (1-k_2^2)E^{(1)} \\ &= (1 - -0.72915 * -0.72915)0.38442\ 10^8 \\ &= 0.18004\ 10^8 \end{aligned}$$

The parameters k_i are known as the reflection parameters. Note that:

- All intermediate solutions are calculated
- This method also provides the reflection coefficients

- The resulting filter is guaranteed to be stable
- The value of the squared prediction residual, $E^{(i)}$ is also computed and is guaranteed to decrease (or remain constant) on each iteration

6.4 The covariance method

The covariance method uses the real values of $s_{-p} \ldots s_{-1}$.

Note that:

- This method can be used on much smaller sample sequences (as end discontinuities are less of a problem)
- There is no guarantee of stability (but you can check for instability)
- Commonly used in "open" and "closed" phase analysis
- This is just a special case of the general least squares problem.

6.5 Pre-emphasis

- The LP filter so far presented attempts to fit an all-pole model using the least-mean-squares distance metric.
- The lower formants contain more energy and therefore are preferentially modeled with respect to the higher formants
- A pre-emphasis filter,

$$s'_n = s_n - a_1 s_{n-1} \tag{83}$$

  is often used to boost the higher frequencies. Typically $0.96 \leq a_1 \leq 0.99$, or the optimal pre-emphasis $a_1 = r_1/r_0$ is used.
- If reconstructing the speech the inverse filter should be used:

$$s_n = s'_n + a_1 s_{n-1} \tag{84}$$

6.6 The LP spectrum

The transfer function $1/H(z)$ is a FIR whitening filter for the speech. The frequency response for this can be computed as the FT of the filter coefficients, then inverted to give the frequency response of $H(Z)$. Figure 39 shows the LP spectrum for the example segmetn of speech.

Matlab:

```
a = lpc(hamming(512) .* sig, 16);
h = (1 ./ fft([a zeros(1,512 - 17)])).';
```

Note:

- The formants are very peaked

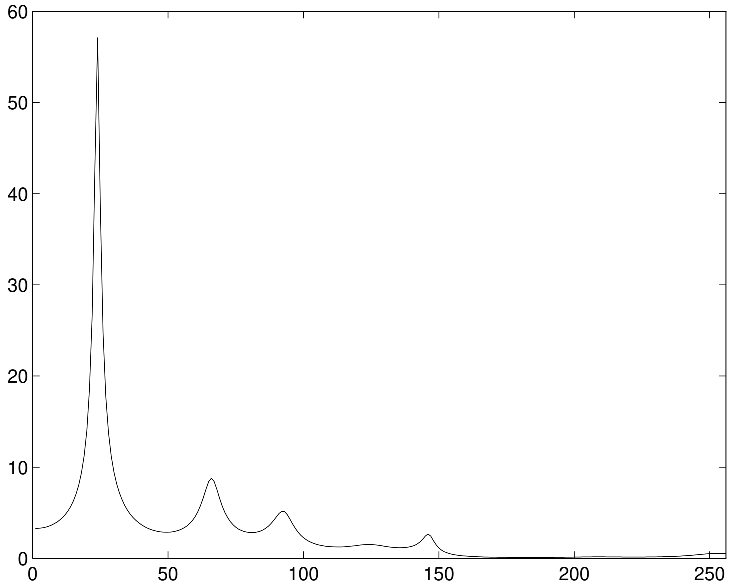

Figure 39: The LP spectrum

6.7 Gain computation

Figure 39 gives the amplitude response of a LP filter for any single frequency. The source-filter excitations are flat, hence the overall gain can be computed.

Durbin's algorithm gives the gain as $G = \sqrt{E^{(0)}/E^{(p)}}$.

This may also be calculated as:

$$G^2 = 1 - \frac{1}{r_0}\sum_{k=1}^{p} a_k r_k \tag{85}$$

- Power in an impulse train separated by N samples is $1/N$
- Power is a Gaussian distributed variable with zero mean and unit variance is 1.
- Power in a random number uniformly distributed on $[-1, 1]$ is:

$$\frac{1}{2}\int_{-1}^{1} x^2 dx = \frac{1}{2}\left[\frac{1}{3}x^3\right]_{-1}^{1} \tag{86}$$

$$= \frac{1}{3} \tag{87}$$

6.8 The lattice filter implementation

Direct implementation of the IIR filter can lead to instabilities if a_i is quantised. The filter is stable provided $1 < k_i < 1 \quad \forall i$ – hence k_i can be quantised and the result is guaranteed to be stable.

Can either convert back from k_i to a_i or implement the IIR filter as a lattice and use the k_i values directly – useful if working on a limited precision DSP chip (e.g. a GSM phone).

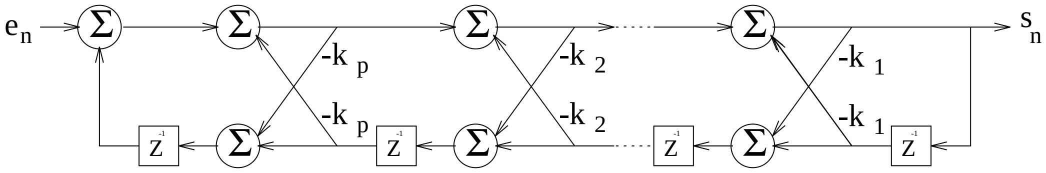

Figure 40: The lattice filter

This is analogous with the lossless tube model:

- each filter section is one section of the tube
- The forward wave is partially reflected backwards
- The backward wave is partially reflected forwards

Hence the terminology of k_i.

6.9 The Itakura distance measure

Consider the case where a speech, s_n, is passed through the linear predictor, a, corresponding to a template. The residual mean squared error, E, is:

$$E = \sum_{n=0}^{N-1} e_n^2 \tag{88}$$

$$= \sum_{n=0}^{N-1} \left(s_n - \sum_{i=1}^{p} a_i s_{n-i} \right)^2 \tag{89}$$

$$= \sum_{n=0}^{N-1} \left(s_n^2 - 2\sum_{i=1}^{p} a_i s_n s_{n-i} + \sum_{i=1}^{p}\sum_{j=1}^{p} a_i a_j s_{n-i} s_{n-j} \right) \tag{90}$$

$$= \sum_{n=0}^{N-1} s_n^2 - 2\sum_{i=1}^{p} a_i \sum_{n=0}^{N-1} s_n s_{n-i} + \sum_{i=1}^{p}\sum_{j=1}^{p} a_i a_j \sum_{n=0}^{N-1} s_{n-i} s_{n-j} \tag{91}$$

$$= \sum_{n=0}^{p} \phi_{00} - 2\sum_{i=1}^{p} a_i \phi_{0i} \sum_{i=1}^{p}\sum_{j=1}^{p} a_i a_j \phi_{ij} \tag{92}$$

$$= \begin{bmatrix} -1 & a_1 & a_2 & \cdots & a_p \end{bmatrix} \begin{bmatrix} \phi_{00} & \phi_{01} & \phi_{02} & \cdots & \phi_{0p} \\ \phi_{10} & \phi_{11} & \phi_{12} & \cdots & \phi_{1p} \\ \phi_{20} & \phi_{21} & \phi_{22} & \cdots & \phi_{2p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_{p0} & \phi_{p1} & \phi_{p2} & \cdots & \phi_{pp} \end{bmatrix} \begin{bmatrix} -1 \\ a_1 \\ a_2 \\ \cdots \\ a_p \end{bmatrix} \tag{93}$$

So can define

$$\mathbf{R} = \begin{bmatrix} \phi_{00} & \phi_{01} & \phi_{02} & \cdots & \phi_{0p} \\ \phi_{10} & \phi_{11} & \phi_{12} & \cdots & \phi_{1p} \\ \phi_{20} & \phi_{21} & \phi_{22} & \cdots & \phi_{2p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_{p0} & \phi_{p1} & \phi_{p2} & \cdots & \phi_{pp} \end{bmatrix} \tag{94}$$

Or in the autocorrelation case:

$$\mathbf{R} = \begin{bmatrix} r_0 & r_1 & r_2 & \cdots & r_0 \\ r_1 & r_0 & r_1 & \cdots & r_1 \\ r_2 & r_1 & r_0 & \cdots & r_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ r_p & r_{p-1} & r_{p-2} & \cdots & r_0 \end{bmatrix} \tag{95}$$

Let $\mathbf{y}$ be the augmented vector of "template" or "reference" LP coefficients $\begin{bmatrix} -1 & a_1 & a_2 & \cdots & a_p \end{bmatrix}$ and $\mathbf{x}$ the augmented vector of "observed" or "unknown" LP coefficients $\begin{bmatrix} -1 & a'_1 & a'_2 & \cdots & a'_p \end{bmatrix}$ then:

$$\mathbf{xRx}^T = \text{Energy at the output of the inverse filter formed by reference with input speech} \tag{96}$$

$$\mathbf{yRy}^T = \text{Minimum possible output energy from LP filter with input speech} \tag{97}$$

In the autocorrelation case this can be computed with similar complexity to the Euclidean distance.

$$d(\mathbf{x}, \mathbf{y}) = \log\left(\frac{E_x}{E_y}\right) \tag{98}$$

$$= \log\left(\frac{\mathbf{xRx}^T}{\mathbf{yRy}^T}\right) \tag{99}$$

6.10 The LP cepstrum

The cepstrum parameters may be computed directly from the LP parameters using the following recursion:

$$c_k = a_k + \frac{1}{k}\sum_{i=1}^{k-1} i c_i a_{k-i} \qquad 1 \le k \le p \tag{100}$$

6.11 Log area ratios

The ratio of the areas, A_i, of the sections in the lossless tube may be directly computed from the reflection coefficients:

$$\frac{A_{i+1}}{A_i} = \frac{1-k_i}{1+k_i} \tag{101}$$

Hence, assuming constant areas at the glottis the total cross section of the vocal tract could be calculated:

- The method is subject to all the LP assumptions
- better normalization methods need to be employed so as to get realistic cross sections
- with further fiddling this methods can be useful – for example as an aid to the deaf

Alternatively this provides the reflection coefficients if we knew the areas.

6.12 The roots of the predictor polynomial

The denominator of the transfer function may be factored:

$$1 - \sum_{k=1}^{p} a_k z^{-k} = \prod_{k=1}^{p} (1 - z^{-1}/c_k) \tag{102}$$

Where c_k is a set of complex numbers defining the roots with angular frequency:

$$\theta_i = \tan^{-1}\left(\frac{\mathrm{Imag}\{z_i\}}{\mathrm{Real}\{z_i\}}\right) \tag{103}$$

and amplitude:

$$r_i = \sqrt{\mathrm{Imag}\{z_i\}^2 + \mathrm{Real}\{z_i\}^2} \tag{104}$$

If close to the unit circle then the root represents a formant.

6.13 Line spectral pairs

The LP polynomial $A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k}$ can be decomposed into:

$$P(z) = A(z) + z^{-(p+1)} A(z^{-1}) \tag{105}$$

$$Q(z) = A(z) - z^{-(p+1)} A(z^{-1}) \tag{106}$$

Now:

- All the roots of $P(z)$ and $Q(z)$ lie on the unit circle
- The roots of $P(z)$ and $Q(z)$ are interspersed
- $P(z)$ corresponds to the vocal tract with the glottis closed and $Q(z)$ with the glottis open
- Very useful in speech coding

7 Perceptual Linear Prediction

A combination of DFT and LP techniques is perceptual linear prediction (PLP) [10].

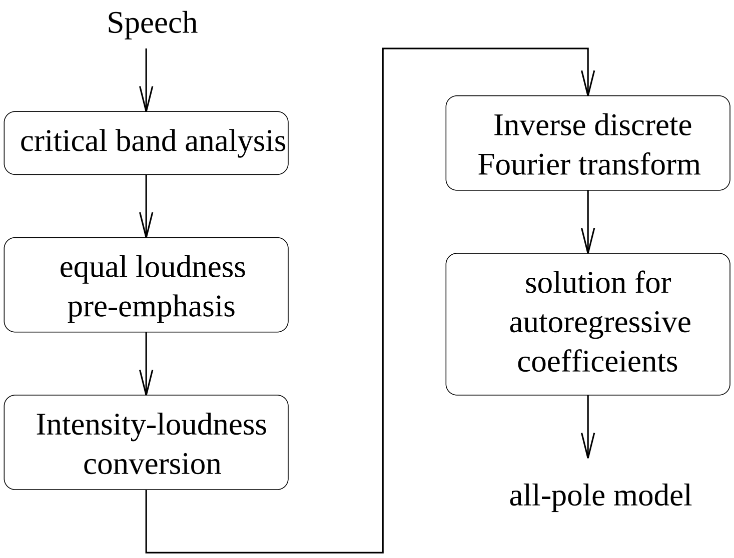

Figure 41: Perceptual Linear Prediction

7.1 Spectral Analysis

The typical window length is 20ms. For 10 kHz sampling frequency, 200 speech samples are used, padded with 56 zeros, hamming windowed, FFT and converted to a power spectral density.

7.2 Critical-band spectral resolution

- The power spectrum is warped onto a Bark scale using the approximation:

$$\Omega(\omega) \quad = \quad 6 \ln \left(\omega/1200\pi + \sqrt{(\omega/1200\pi)^2 + 1} \right) \tag{107}$$

- The bark scaled spectra is convolved with the power spectra of the critical band filter. This simulates the frequency resolution of the ear which is approximately constant on the Bark scale.

$$\begin{aligned}
\Phi(\omega) = \quad & 0 & \Omega < -1.3 \qquad & (108)\\
& 10^{2.5(\Omega+0.5)} & -1.3 \leq \Omega < -0.5 \qquad & (109)\\
& 1 & -0.5 \leq \Omega < 0.5 \qquad & (110)\\
& 10^{-1.0(\Omega-0.5)} & 0.5 \leq \Omega < 2.5 \qquad & (111)\\
& 0 & \Omega > 2.5 \qquad & (112)
\end{aligned}$$

  This convolution reduces the spectral resolution

- The smoothed Bark scale spectrum is down-sampled by resampling every 1 Bark (0 - 5 kHz maps to 0- 16.9 Bark).

7.3 Equal loudness preemphasis

- Need to compensate for the non-equal perception of loudness at different frequencies
- Preemphasise by an equal-loudness curve

$$E(\omega) \quad = \quad \frac{(\omega^2 + 56.8\ 10^6)\omega^4}{(\omega^2 + 6.3\ 10^6)(\omega^2 + 0.38\ 10^9)(\omega^6 + 9.58\ 10^{26})} \tag{113}$$

- The inverse if this curve is plotted in figure 42 on a dB scale.

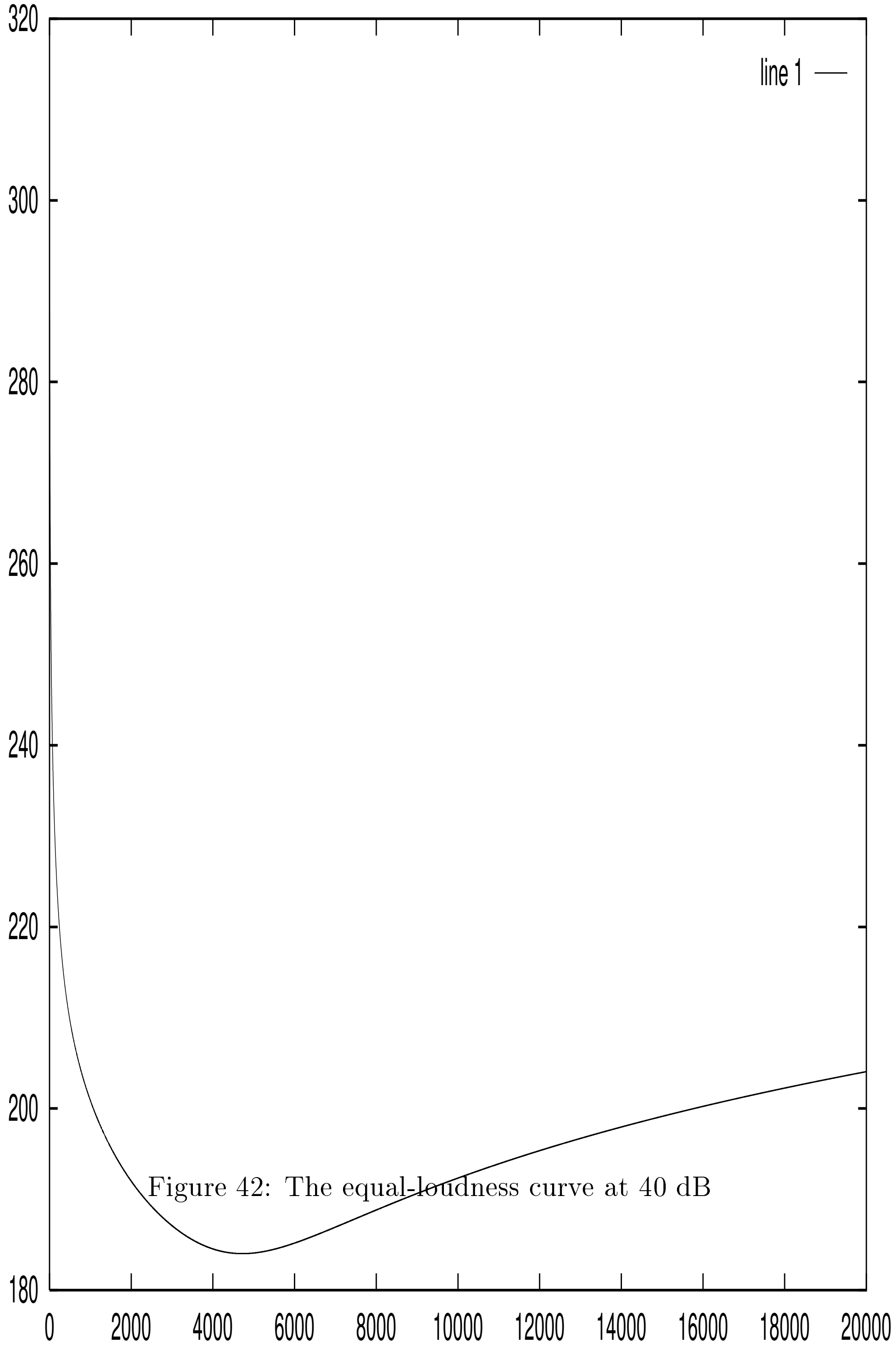

Figure 42: The equal-loudness curve at 40 dB

7.4 Intensity-loudness power law

- Perceived loudness, $L(\omega)$, is approximately the cube root of the intensity, $I(\omega)$

$$L(\omega) = I(\omega)^{1/3} \tag{114}$$

- Not true for very loud or very quiet sounds
- A reasonable approximation for speech

7.5 Autoregressive modelling

- Apply the IDFT to get the equivalent of the autocorrelation function
- Fit a LP model
- Optionally transform into cepstral coefficients

7.6 Discussion

- Computes a simple auditory spectrum
- Computational requirements similar to LP or FFT analysis
- Better spectral modelling than LP
- Mel scaled cepstral coefficients (MFCC) are historically more popular
- Provides a more robust representation than MFCC.

8 Formant analysis

- Formants are perceptually defined.
- The corresponding physical property is the frequencies of resonances of the vocal tract.
- Formant analysis is useful as the position of the first two formants pretty much identifies a vowel.

8.1 Motivation

- Base level analysis
- The development of speech synthesis technology
- To study phonetics

8.2 Obtaining candidate values

Can employ both DFT and LP methods.

8.2.1 Peak picking on the smoothed spectrum

- Use cepstral techniques to get the envelope (section 4.9)
- Pick the N largest peaks/the N lowest frequency peaks/all peaks

8.2.2 Peak picking on the LP spectrum

- Use cepstral techniques to get the envelope (section 6.6)
- Pick the N largest peaks/the N lowest frequency peaks/all peaks
- *or* pick points where the second derivative is most negative - i.e. the sharpest peaks

8.2.3 Factoring for the LP roots

As described by equation 116, the LP polynomial may be factored and therefore the method of partial fractions may be used to express the transfer function as:

$$\frac{1}{1-\sum_{k=1}^{p} a_k z^{-k}} = \frac{1}{\prod_{k=1}^{p}(1-z_k z^{-1})} \tag{115}$$

$$= \sum_{k=1}^{p} \frac{C_k}{(1-z_k z^{-1})} \tag{116}$$

- The roots occur in complex conjugate pairs (as the polynomial in a_i is real)
- The angle defines the formant frequency
- As with the LP spectrum, can choose all peaks or those closest to the unit circle.

8.3 Fitting "bumps"

- Each formant is approximately Gaussian
- Can treat the power spectral density as a probability density function
- Can model a probability density function as mixture of Gaussians

$$b_j(\underline{x}) = \sum_{m=1}^{M} c_{jm} \frac{1}{\sqrt{2\pi}\underline{\sigma}_{jm}} e^{-\frac{1}{2}\left[\frac{\underline{x}-\underline{\mu}_{jm}}{\underline{\sigma}_{jm}}\right]^2} \tag{117}$$

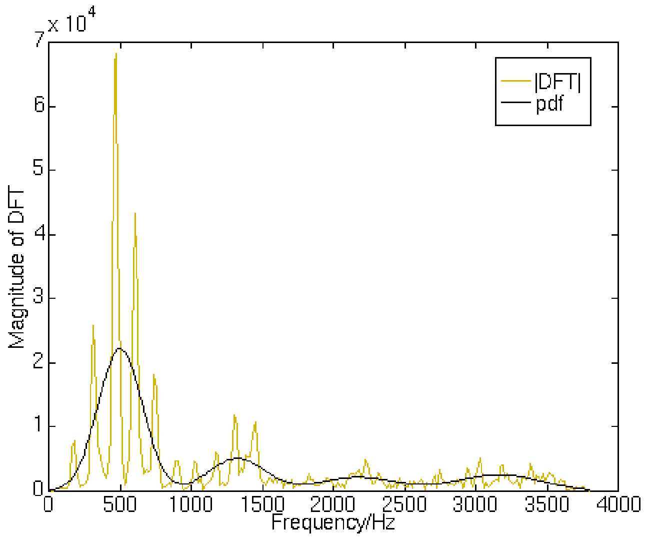

Figure 43: A Mixture of Gaussians fit to a DFT magnitude spectrum

8.4 Combining candidates

- Every method produces dud peaks.
- Formants have continuity and rate of change constraints
- Impose constraints heuristically and throw away some candidates
- Express constraints formally use dynamic programming

9 Voicing analysis

- Need both:
  - Pitch
  - Degree of voicing
- Pitch is perceptually defined – the acoustic correlate is fundamental frequency
- The fundamental frequency may be missing – e.g. telephone speech
- Even on band limited speech, pitch is still clear

Standard reference is Hess [11].

9.1 Pitch synchronous analysis

The discontinuities associated with windowing can be reduced if the analysis is carried out pitch synchronously.

Mostly applied to LP based techniques as it is easier to work with these on arbitrary frame lengths.

Again, need a set of potential candidates:

9.2 Zero-crossing points

- Periodic sounds have periodic zero-crossings
- For significant harmonics a range of pitch values are obtained

9.3 Peak in the autocorrelation function

- For a periodic signal of period P the autocorrelation function repeats with period P: $r_0 = r_{nP}$.
- In practice, $r_0 \geq r_P$ (n.b. can't be greater)
- For example, see figure 33
- Like previous method – hard to find right peak

9.4 Peak in the autocorrelation of the LP residual

- Similar to previous method
- Has advantage that a weak fundamental is boosted to the same level as the other harmonics before the autocorrelation
- DRAW DIAGRAM

9.5 The average magnitude difference function

- Same as autocorrelation but faster to implement in integer arithmetic as no multiplies.

$$\frac{1}{N}\sum_{i=0}^{N} |s_i - s_{i-n}|$$

9.6 Peak in the cepstrum

- The cepstrum can often show a very clear pitch peak - for example, figure 29
- One of the most robust frame based methods

9.7 Combining candidates

- Every method produces dud candidates.
- Pitch has continuity and rate of change constraints
- Impose constraints heuristically and throw away some candidates
- Express constraints formally use dynamic programming

Pitch doubling and pitch halving errors are the most common type.

10 Degree of voicing

10.1 Voicing determination

- zero-crossing rate
- r_1/r_0 or the fraction of the power at low frequencies
- Degree of strength of "pitch peak"
- Surprisingly difficult. Makes a huge difference to the subjective quality of a LP coder.
- In practice, combine many methods (using statistical pattern recognition techniques)

10.2 Usage

- Most commonly binary: voiced or unvoiced
- Real valued is useful:
  - Mix voiced and unvoiced excitation with appropriate powers
  - Apply a high pass filter to the unvoiced excitation, low pass filter to the voiced excitation, and the degree of voicing defines the single cutoff frequency

11 Speech coding

- Speech can be coded at many levels.
- Lower bit-rates are achieved by imposing more constraints of the speech production mechanism are applied
- Failure becomes more catastrophic as the bit rate is reduced

11.1 Waveform coders

- Simple to implement in hardware
- Low delay
- Contain little speech specific information and so are very general
- data rates about 32 kbps

11.1.1 Pulse Code Modulation (PCM)

- Needs the sampling frequency, f_s, to be greater than the Nyquist frequency (twice the maximum frequency in the signal)
- For n bits per sample, the dynamic range is $\pm 2^{n-1}$ and the quantisation noise is $1/12$
- Total bit rate: nf_s
- Can use non-uniform quantisation or variable length codes

11.1.2 Differential Pulse Code Modulation (DPCM)

- Predict the next sample based on the last few *decoded* samples
- Minimise mean squared error of prediction residual - use LP coding
- Good prediction results in a reduction in the dynamic range needed to code the prediction residual and hence a reduction in the bit rate
- Can use non-uniform quantisation or variable length codes

11.1.3 Adaptive Differential Code Modulation (ADPCM)

- Speech is quasi-stationary
- Adapt the predictor
- Forward adaptation: send new predictor values
- Backward adaptation: use predictor values computed from recently decoded signal
- Can use non-uniform quantisation or variable length codes

11.1.4 Example: lossless waveform coding

- Lossless compression of a speech database using ADPCM-like techniques with a forward predictor (`shorten`)
- Speech sampled at 16 kHz and 16 bits per sample (TIMIT database)

The variation of compression ratio with predictor order is shown in figure 44.

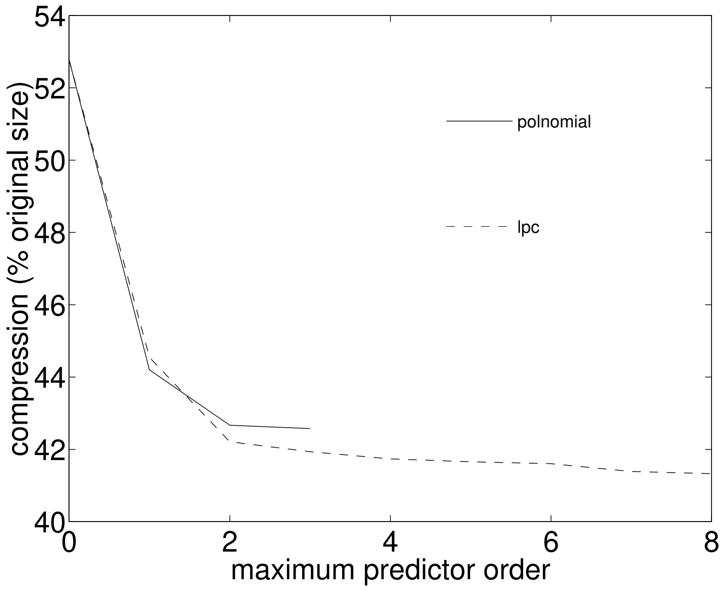

Figure 44: Variation of compression ratio with predictor order for a polynomial and general linear predictor

Note that there is considerable compression without a predictor. This is because the prediction error is non-uniform and therefore may be entropy coded. The distribution of prediction errors is shown in figure 45.

11.2 Sub-band coders

- Exploit the frequency sensitivity of the auditory system
- Split the signal into sub-band using band pass filters, such as the example of figure 12
- Code each subband at an appropriate resolution - e.g. 4 bits per sample in the lower subbands and 2 bits per sample in the upper subbands
- Can also exploit auditory masking - use fewer bits if a neighboring subband is much louder
- Basis for the MPEG audio standard (5:1 compression of CD quality audio with no perceptual degradation)

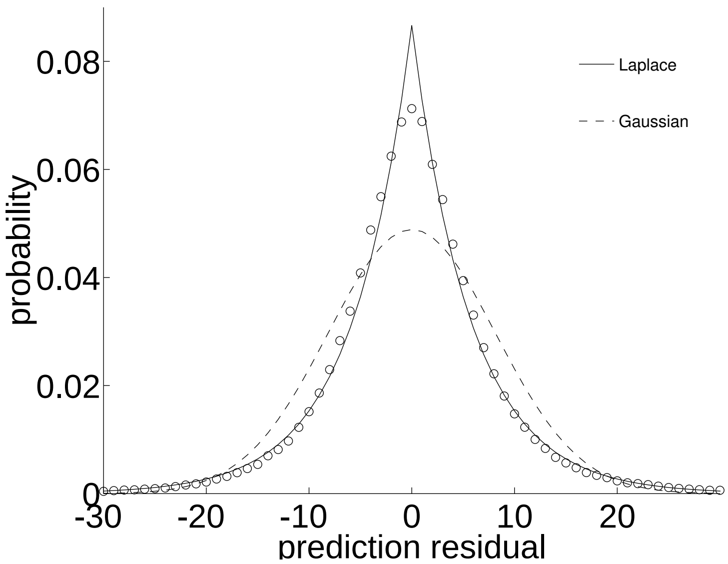

Figure 45: The probability density function for the prediction error and the Gaussian and Laplacian distributions

11.3 Linear prediction vocoders

- Employ the source filter model (e.g. figure 9).
- For ever frame, need to code:
    - A representation of the LP filter
    - Power
    - Degree of voicing
    - Pitch (if voiced)
- Most bits go into the LP parameters

Commonly used representations of the LP parameters:

- LP coefficients: when quantisation is not a problem
- Reflection coefficients: robust but not very efficient
- Line spectral pairs: most efficient

Quantisation:

- Independent: about 50 bits per frame
- Vector quantisation: about 25 bits per frame

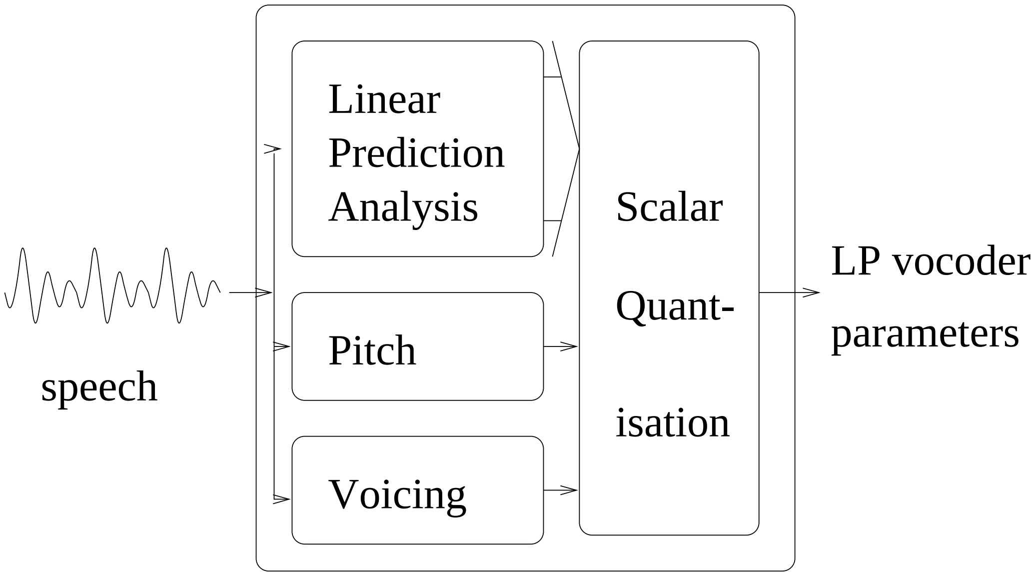

Figure 46: Simple LPC vocoder

11.3.1 Parameter updating

- LP paramters/gain/voicing/pitch determined for equally spaced frames
- Not feasible to do interpolate on every sample
- Infrequent interpolation introduces discontinuities
- Discontinuities less noticable on pithc period boundaries
- Practical solution:
  - For unvoiced speech use original analysis timings
  - For voiced speech update on pitch pulse injection
  - Use linear interpolation to get LP paramters

11.3.2 Example: LPC10

So called because 10 LP coefficients are used.

- 8 kHz sample rate, 180 samples/frame, 44.44 frames/second
- Order 10 LP analysis: First two coefficients are quantised as log area ratios with five bits each, last 8 as reflection coefficients. Number of bits per coefficient decreases with index down to two bits
- 7 bits used for pitch and voicing decision
- 5 bits used for gain

- Total: 54 bits per frame, 2400 bps

Disadvantages:

- Can sound very "buzzy" - noise through parameter updates
- Poor LP modelling results in wide bandwidths and rapid decay of the pulse excitation
- Regularly voiced excitation is unnatural - normally some jitter
- Voicing errors produce significant distortions
- Binary voicing decision is sometimes poor
- Not suited to model nasals - although okay in practice
- Only models speech - doesn't work if background noise (i.e. not suited to mobile phone applications without further work).

Items which have to be addressed:

- Glottal pulse shaping
- Pitch synchronous parameter updating
- Fine tuning of the voicing decision
- Separation of speech and noise
- Exploitation of temporal correlations of acoustic vectors

11.4 Formant coders

- Formant vocoders are like LP vocoders but the transmitted parameters is the formant information.
- Lower information rate
- Harder to get right $\Rightarrow$ greater distortion

11.5 Multi-pulse coders

- Like LP coders but the excitation is modelled as a number of pulses
- Pulse position is found by an exhaustive search to minimise mean squared error
- About 1 pulse per millisecond is enough to generate high quality speech
- No longer very popular
- Typical bit-rate is 9600 bps

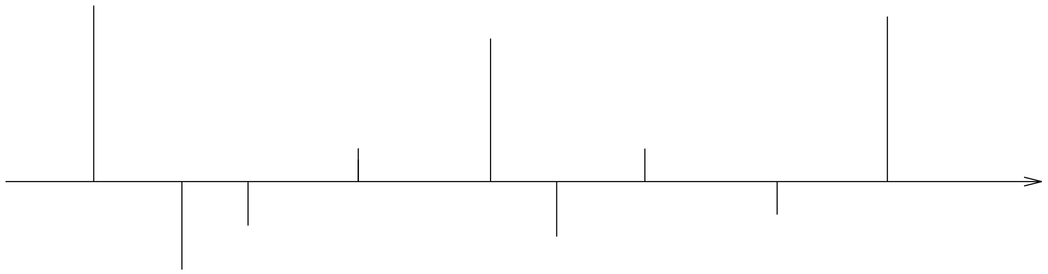

Figure 47: Typical Multi-pulse excitation sequence

11.6 Backwards adaptation

The basic waveform coders presented so far compute and transmit LPC and residual parameters on a frame by frame basis. Backwards adaptation uses past decoded speech to estimate the LP model. Issues are:

- No need to transmit LP paramterers
  - LP Model is poorer - trained on delayed data
  - LP paramters are hard to quantise - consume appreciable bandwidth
  - Simpler bitstream format
- Longer model order is possible - not limited by quantisation error
- Decoder needs more power to generate the LP model

Works best for low delay systems.

11.7 CELP coders

Code Excited Linear Prediction (CELP) coders are state-of-the-art in telephony coding:

- Start from basic LP vocoder
- Apply a long term predictor to remove redundancies through periodicity
- Vector quantise the LP parameters - as before
- Vector quantise the LP residual - typically 40 samples (5ms) using 1024 codebook entries
- Coding is much more computationally expensive than decoding (need codebook searches)

- Resultant bit rate is about 4 kbps

The theory says that after all the linear correlations have been removed, the prediction residual is uncorrelated values with a Gaussian distribution. Therefore generate the excitation codebook as a set of random numbers.

Complete system looks like figure 48:

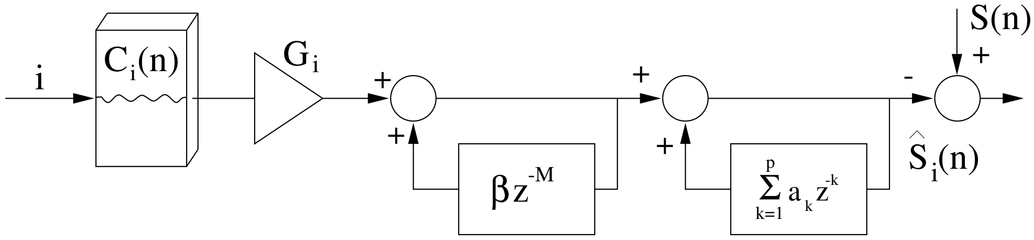

Figure 48: Typical CELP coder

Where all parameters of the system are obtained by minimising the final mean squared error. Sequential parameter estimation is non-optimal.

12 Audio demonstration

	Female "She had your..."	Male "An apple..."	S. Vega "I am sitting..."
Original	1	2	3
Simple LPC @ 200Hz	4	5	6
Simple LPC @ 100Hz	7	8	9
LPC: all voiced	10	11	12
LPC: all unvoiced	13	14	15
LPC: inverse filter	16	17	18
CELP @ 4800 bps	19	20	-
LPC10e 2400 bps	21	22	23
HP, 16kHz, 1800bps	24	25	26
HP, 8kHz, 950 bps	27	28	29
HP, 8kHz, 550 bps	30	31	32

References

[1] L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals.* Prentice Hall, Englewood Cliffs, New Jersey, 1978.

[2] Frank Fallside and William A. Woods. *Computer Speech Processing.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[3] John R. Deller, John G. Proakis, and John H. L. Hansen. *Discrete-Time Processing of Speech Signals.* Maxwell Macmillan International, 1993.

[4] Emmanuel C. Ifeachor and Barrie W. Jervis. *Digital Signal Processing: A practical approach.* Addison-Wesley, 1993.

[5] B. C. J. Moore. *An Introduction to the Psychology of Hearing.* Academic Press, London, second edition, 1982.

[6] Steven F. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-27(2):113–120, April 1979.

[7] M. Berouti, R. Schwartz, and J. Makhoul. Enhancement of speech corrupted by acoustic noise. In *Proc. ICASSP*, pages 208–211, 1979.

[8] J. Makhoul. Linear prediction: A tutorial review. *Proceedings of the IEEE*, 63(4):561–580, April 1975.

[9] Willaim H. Press, Brian P. Flannery, Saul A. Teukolsky, and Willaim T. Vetterling. *Numerical Recipies in C: The art of Scientific Programming.* Cambridge University Press, Cambridge, England, 1988.

[10] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *Journal of the Acoustical Society of America*, 87:1738–1752, 1990.

[11] W. Hess. *Pitch Determination of Speech Signals.* Springer-Verlag, 1983.