homework solutions for:

Homework #5: Spectral Estimation using Linear Prediction

submitted to:

Dr. Joseph Picone
ECE 8993 Fundamentals of Speech Recognition

May 3, 1998

submitted by:

Jonathan Hamaker

Institute for Signal and Information Processing
Department of Electrical and Computer Engineering
Mississippi State University
Box 9571, 216 Simrall, Hardy Rd.
Mississippi State, Mississippi 39762
Tel: 601-325-8335, Fax: 601-325-3149
Email: hamaker@isip.msstate.edu

Introduction

Spectral estimation is one of the most widely used concepts in signal processing technology. It is found in every speech recognition system as one tries to model the speech signal by a set of features corresponding to, among other things, the frequency content of the signal. Linear predictive (LP) analysis is one method for estimating the spectrum which provides an autoregressive all-pole model of the short-term power spectrum of the signal. One drawback to this approach is that human speech is definitely not an all-pole system. Nevertheless, LP-derived Cepstra models are the most popular form of spectral modeling used in speech front-ends today. In this assignment, we will explore the use of LP analysis for modeling of a speech signal. The results will be compared to a DFT of the same signal.

Problem Statement

Implement a capability to plot a signal's FFT spectrum, and the gain-matched spectrum produced by a linear prediction model. The tool must read speech from a binary file (assume 16-bit linear sampling), and allow the user to select the following:

- sample frequency of the signal
- pre-emphasis constant
- window duration in secs
- center time for the window in secs
- a rectangular or hamming window
- the linear prediction order

The net result should be a plot of the signal spectrum computed using the following parameters:

fs = 8 kHz
pre-emphasis = argv[1]
window_duration = argv[2]
center time of the window = argv[3]
hamming window = yes
lp_order = argv[4]

and plotted on a log amplitude vs. linear frequency scale. The spectra of the corresponding linear prediction model should be plotted as well.

Use the DFT to compute the spectrum of the signal.

For example,

```
my_prog.exe 8000.0 0.95 0.03 3.0 12 | xmgr -source stdin
```

should produce a signal and LP model spectrum for a 30 msec window of the signal centered at 3 secs. The LP analysis will be of order 12. A pre-emphasis filter 1 - 0.95z**-1 is applied to the data.

The resulting plots will typically have about a 60 dB dynamic range

Methodology

The autoregressive LP model is derived from the thought that a message-bearing signal is never completely random. Thus we can use the signal autocorrelation to predict samples in the signal. LP analysis of a signal produces an all-pole model of the signal. The LP coefficients are found from the autocorrelation vector using the popular Levinson-Durbin recursion. In Linear prediction, we determine the current sample value as a linear combination of previous sample values. Our goal is to define the linear combination coefficients (also known as predictor coefficients or LP coefficients) so as to minimize the error between the predicted sample value and the actual sample value.

Intuitively, we can define a signal $s(n)$. We wish to predict the next sample using the p previous samples by

$$\tilde{s}(n) = \sum_{k=1}^{p} a_k s(n-k) \ . \tag{1}$$

Since the ultimate goal is to minimize the error between the predicted value and the actual value for the entire signal, we will define a short-time average prediction error

$$E = \sum_n e^2 = \sum_n \left\{ s(n) - \sum_{k=1}^{p} a_k s(n-k) \right\}^2 \ . \tag{2}$$

We find the optimal a_k by setting the derivative of E to 0 and solving for each a_k. From this we derive the LP equation

$$\sum_n s(n)s(n-l) = \sum_{k=1}^{p} a_k \left(\sum_n s(n-k)s(n-l) \right) \ . \tag{3}$$

An efficient method for solving this for the a_k values uses the Levinson-Durbin recursion to implement an autocorrelation method. This method is described in the pseudocode below.

for $i = 1, 2, \ldots, p$

$E_0 = r(0)$, where $r(i)$ is the i^{th} autocorrelation coefficient

$$k_i = \left\{ r(i) - \sum_{j=1}^{i-1} a_{i-1}(j) r(i-j) \right\} / E_{i-1}$$

for $j = 1, 2, \ldots, i-1$

$a_i(i) = k_i$

$a_i(j) = a_{i-1}(j) - k_i a_{i-1}(i-j)$

$E_i = (1 - k_i^2) E_{i-1}$

Results

To observe the effect of LP model order on our ability to approximate the spectrum of a signal we have written the LP analysis program described in Figure 1. This program takes a portion of a speech file and produces the gain-matched LP-derived spectrum as well as the spectrum derived from a DFT calculation. The output from the program is suitable for plotting with xmgr. We show spectra generated from a 30msec window centered at 1 second in the file 710_s_8k.raw using a hamming window and a pre-emphasis filter with a coefficient of 0.95.

Figures [2-10] show the DFT spectrum (in black) in the LP-derived spectrum (in red). It is easy to guess that as the LP model order gets higher, the error between the DFT spectrum and LP-derived spectrum gets much smaller reaching close to zero at an LP order of 256. The primary use of the LP model, though, is to follow peaks in the spectrum. This property is shown even for low LP model orders. The LP model uses the coefficients available to it to model the most significant peaks (poles) in the signal. As the model order grows, the model begins to allocate coefficients for less significant peaks in the spectrum.

```
name: calculate_lpc
synopsis: calculate_lpc [options]
descr: produces the lp-derived spectrum and the dft spectrum in a format suitable for
        input to xmgr
example: calculate_lpc -sf 8000 -swap_bytes -input input_file.raw -use_hamming
          -use_pre_emph -pre_emph_coeff 0.95 -window_dur .03 -mid_time 5.6 -lp_order 12

arguments:
8000 : sample frequency in samples per second
input_file.raw : input file name - file is assumed to be in 16-bit linear format
0.95 : pre-emphasis coefficient of first-order highpass filter
.03 : this indicates a window length of 30 msecs
5.6 : this indicates that the window should be centered around 5.6 seconds.
12 : order of the all-pole LP model

options:
-sf : sample frequency of the data (default: 8000)
-swap_bytes :   flag indicating whether the byte ordering should be swapped
-input : input speech file
-use_hamming : tells program to hamming window the data (rectangular window otherwise)
-use_pre_emph : tells program to use a pre-emphasis filter of the form 1-a(z^(-1))
-pre_emph_coeff : the ‘a’ term in the pre-emphasis equation20 msec)
-window_dur : duration of the data window
-mid_time : time to center the window about
-lp_order : order of the LP model
```

Figure 1. Command line interface to LP spectrum analysis tool

Software

All software for this assignment was written using C++ and can be found at: *http://www.isip.msstate.edu/resources/courses/ece_8993_speech/1998/problem_05/hamaker/src/.*

References

[1] J. Picone, “Fundamentals of Speech Recognition: A Short Course,” Institute for Signal and Information Processing, Mississippi State University, 1996.

[2] J. Markel and A. Gray Jr., *Linear Prediction of Speech*, Springer-Verlag, New York, 1976.

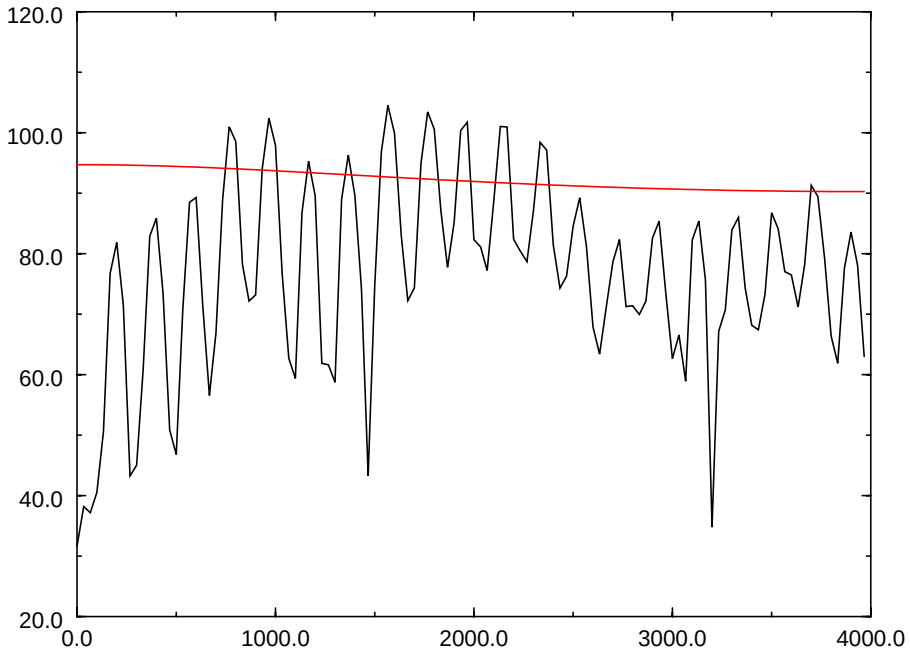

Figure 2. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 1 (red).

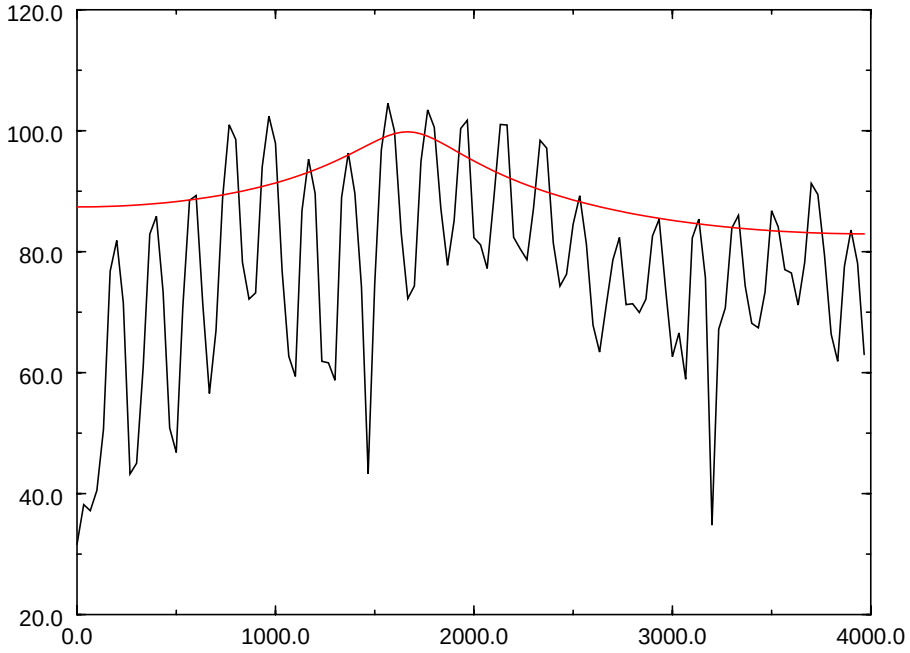

Figure 3. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 2 (red). Notice that the model now uses one pole to model the most significant peak in the spectrum (the other pole models the symmetric part of the spectrum.

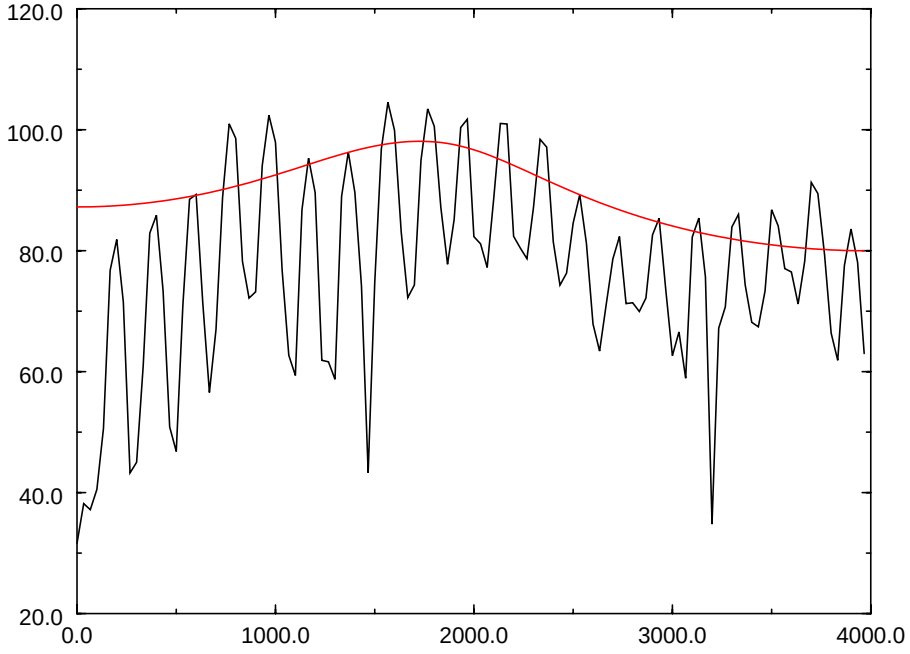

Figure 4. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 4 (red). Now the model is using the next two poles to widen the modeling of the primary peaks in the spectrum.

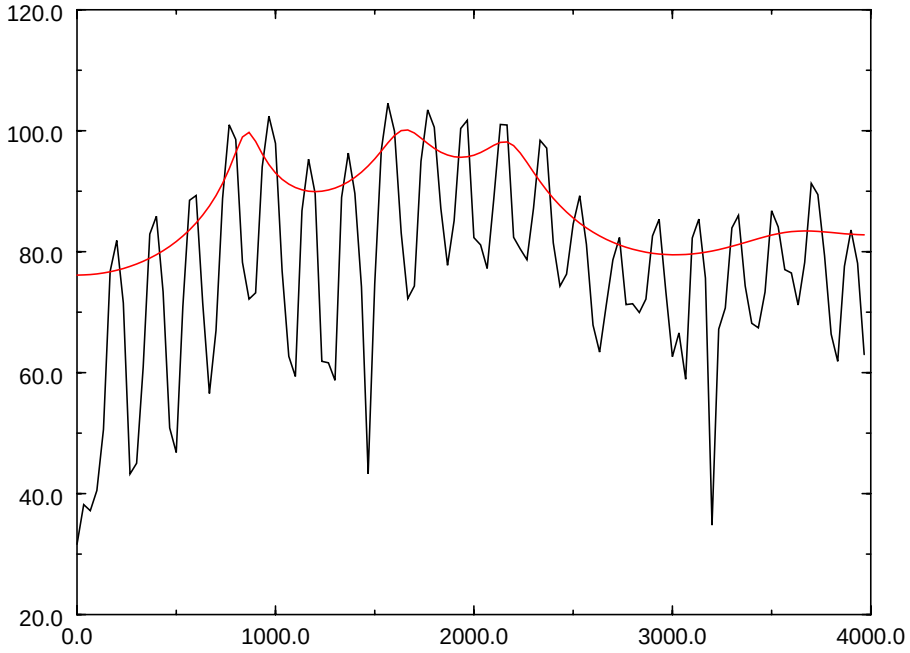

Figure 5. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 8 (red). We see that this model is now modeling some of the lower energy peaks in the spectrum

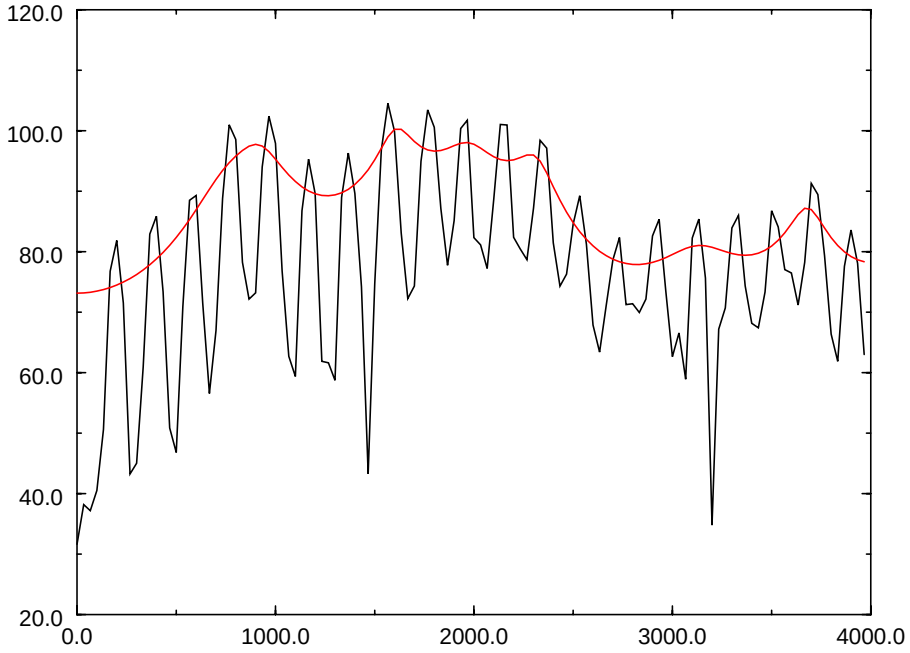

Figure 6. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 16 (red). We see that this model is now modeling some of the lower energy peaks in the spectrum

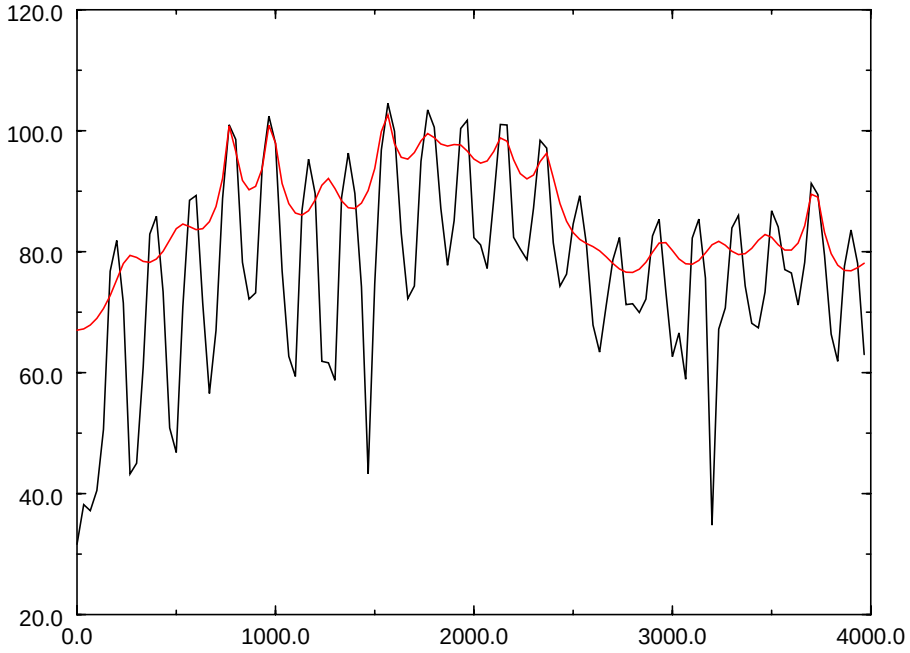

Figure 7. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 32 (red). With this size model, the LP analyzer is able to model almost every peak in the spectrum but still has a hard time modeling the troughs

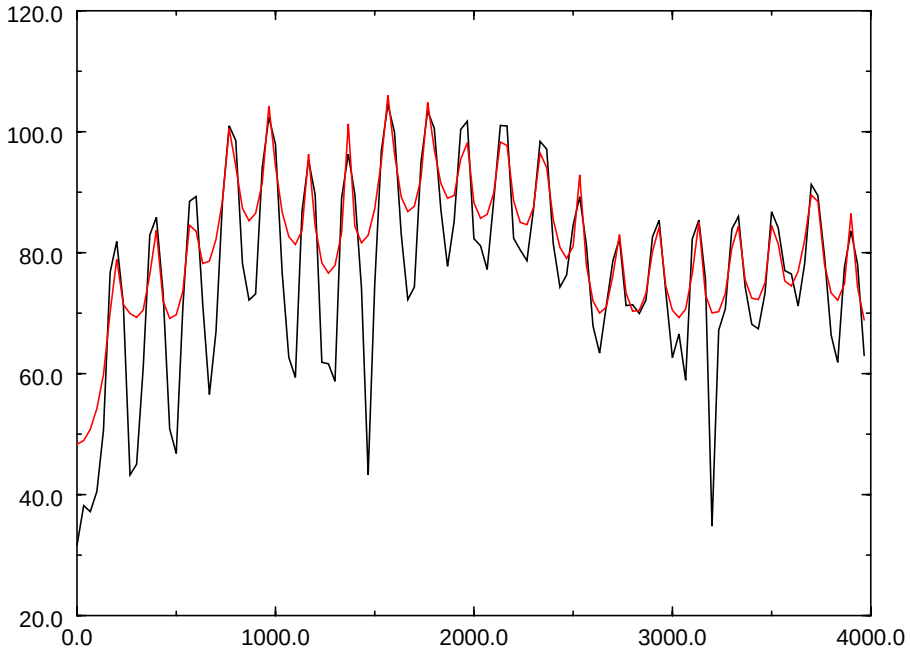

Figure 8. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 64 (red). Increasing the model to order 64 is allowing the LP all-pole model to fill in the troughs with poles thus better representing the actual spectrum.

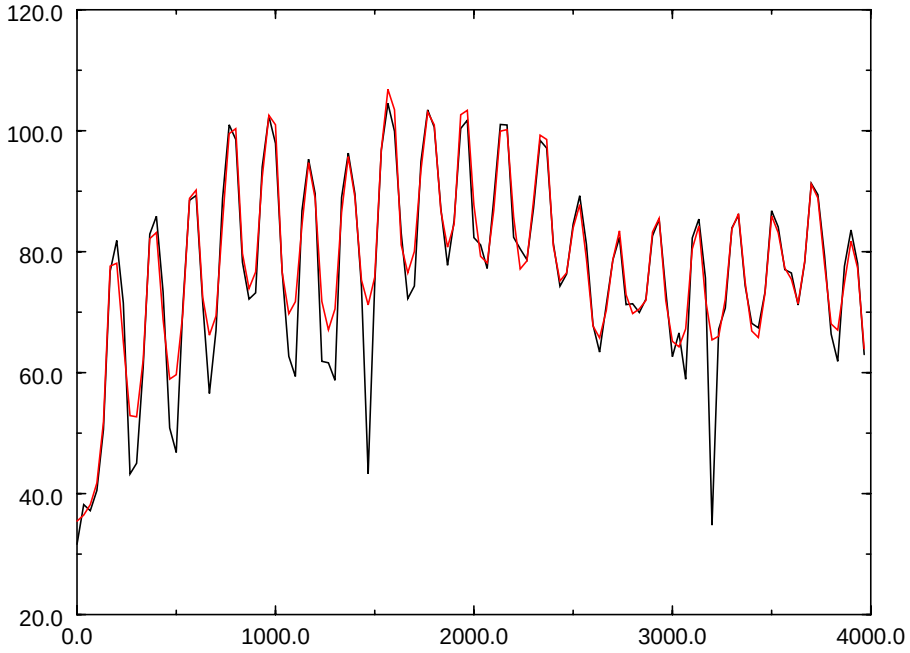

Figure 9. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 128 (red). As the LP order grows larger, the model error gets lower and lower as shown in this figure.

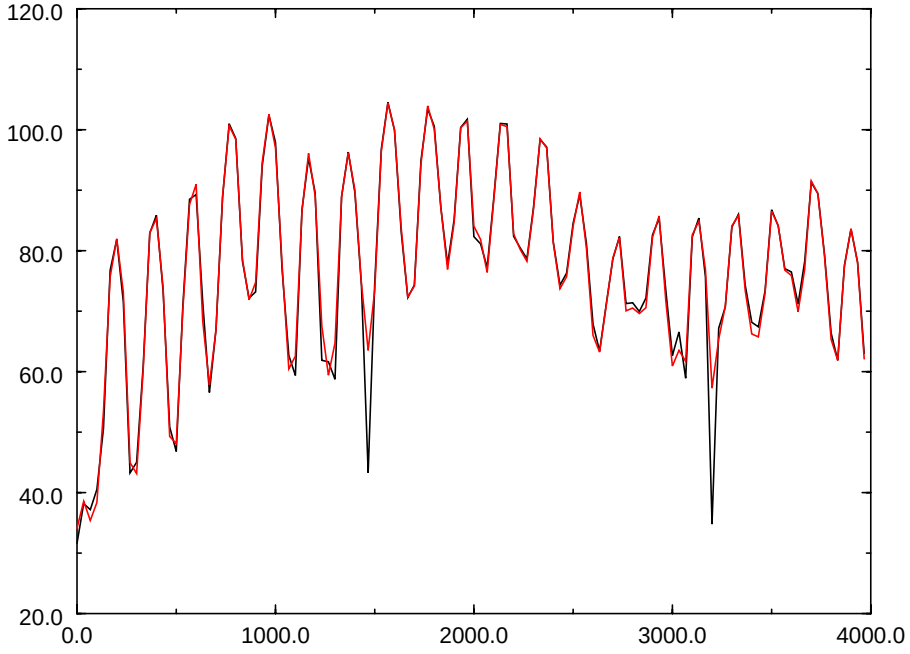

Figure 10. Spectra of 30msec window centered about 1 second using a DFT (black) and an LP model of order 256 (red).