LEAST SQUARES SPEECH COMPRESSION
DIFFERENTIAL/PREDICTIVE CODING

Compression of Speech Signals.

Speech signals, represented by sample data \( \{y(n), n = 1, 2, \cdots, N\} \), are often quantized to a low bit rate during data transmission in order to obtain faster data transfer rates. Unfortunately, the quantization of the speech signal, resulting in \( y_q(n) \), will have a degrading effect on the quality of the speech signal. One method to reduce the degradation, however, is to fit an autoregressive (all pole) model to the signal of the form:

\[
y(n) = \sum_{k=1}^{10} a(k)y(n - k) + e(n).
\]

(1)

The residuals \( e(n) \) will generally have a much smaller dynamic range than the original speech signal values. The speech signal will vary (after normalization) from -1 to 1 while the residuals are mostly smaller than 0.1. To minimize the degradation effects of quantization, one can quantize the residuals \( e(n) \), then transmit only the quantized residuals, \( e_q(n) \), the coefficients for the model (1), and the quantization rates. Reconstructing the signal from the model and the quantized residuals, \( e_q(n) \), to obtain an estimate of the original speech signal, \( \hat{y}(n) \), one should have a smaller error Mean Squared-Error (MSE), \( \text{MSE} = \frac{1}{N} \sum_n (y(n) - \hat{y}(n))^2 \), than if the quantized original signal \( y_q(n) \) was sent, \( \text{MSE} = \frac{1}{N} \sum_n (y(n) - y_q(n))^2 \).

In Matlab, one may process the signals and listen to them as well. The assignment is to obtain speech data from the internet and to use Matlab to calculate the coefficients, \( a(k) \), of the model (1) and residuals, \( e(n) \), for a given speech signal, \( y(n) \), then calculate the difference in the MSE for the quantized speech signal and the signal reconstructed from the quantized residuals. You are to listen to the original signal, \( y(n) \), the quantized signal, \( y_q(n) \), and the reconstructed signal \( \hat{y}(n) \) and described any perceived differences. Because the linear model (1) cannot accurately represent the entire speech signal, you need to process the signal in blocks of 160 sample data points and calculate model coefficients and quantization levels for each block.

Procedure:

Step 1. Using a Web browser, such as Netscape, download a speech signal from the Web. Search engines allow you to limit your search to just speech files. These typically have identifying extensions, such as .wav. For a .wav file, you can load the speech signal using the Matlab command \texttt{wavread},

\[
>> y = \text{wavread}('file.wav');
\]

The speech signal may start with some nonzero constant value such as -1 before the actual speech starts. In the array containing the data \( y \), reset these values to 0 up to the point where the values in \( y \) begin to vary. You can then listen to the speech signal using the \texttt{sound} command in Matlab,

\[
>> \text{sound}(y);
\]
Step 2. Calculate the quantized speech signal $y_q(n)$ for a specified number of quantization quantization levels, $L = 2^r$, where $r$ is the quantization rate in bits–per–symbol. This can be done using the following two lines of Matlab code,

$$\text{>> } q = (\text{max}(y)-\text{min}(y))/L;$$
$$\text{>> } yq = \text{round}(y/q)*q;$$

(Alternatively, you might use the second (or third, etc.) largest and smallest values in $y$ to discount spurious outlier spikes in data value. This might be particularly efficacious when quantizing the residual values in $e$.) You can listen to the quantized speech signal. Try it for quantization rates $r = 1, 2, 3, 4, 5, 6, 7, 8$, and listen to the difference in quality. You can calculate the MSE by simple matrix multiplication,

$$\text{>> } \text{MSE} = (y-yq)'*(y-yq)/N;$$

Is there any relationship between MSE (a mathematical measure of distortion) and your subject opinion of quality based on listening to the sound?\(^1\)

Step 3. Estimate the filter coefficients $a(k)$ for each block of 160 data speech data points using the least squares optimization technique discussed in class by setting it up as an $Aa = b$ problem. The Matlab command $\text{toeplitz}$ should be useful in constructing the matrix $A$. Type ‘help toeplitz’ in Matlab to see how it may be used. You will have to ‘remember’ the last 10 elements of the previous block in estimating $a(k)$, while for the first block you can initialize it with zeros.

Step 4. Calculate the residual error, $e(n)$ in equation (1) via digital filtering of the sound samples $y(k)$. This can be done by writing code to implement equation (1) in its “moving average” (MA) mode, which takes $y(k)$ as inputs and produces the residuals $e(k)$ as outputs. The initial conditions required to do this can be taken to be the last ten samples of the previous block. (For the first block use zero values for the initial conditions.) Alternatively, you can perform the required digital filtering via use of the Matlab commands $\text{filter}$ and $\text{filtic}$. The command $\text{filtic}$ generates initial conditions for the beginning of each block for $\text{filter}$. However, the initial conditions used in $\text{filter}$ are not easily understood unless you’ve already taken a course in digital signal processing (DSP). If you have not had a course in DSP, it is recommended that you write your own filtering code.

Step 5. Quantize the residuals using the same quantization rates as in Step 2, but dependent, however, on the maximum and minimum of the residuals in each block (or on the second maximum and minimum, etc., as mentioned above) of data. Reconstruct the signal via digital filtering of the quantized residuals. This can be done by writing code to implement equation (1) in its recursive “autoregressive” (AR) mode, which takes residuals as inputs and produces sound samples as outputs. The initial conditions required to do this can be taken to be the last ten samples of the previously reconstructed sound block. (For the first block you use values for the initial conditions.) Note that you can implement this filter because you will have already reconstructed the past sound values needed to perform the next step of reconstruction. Alternatively, reconstruction of the sound signal be done using the Matlab $\text{filter}$ command to obtain the reconstructed signal $\hat{y}$ via digital filtering of the quantized residuals. However, as mentioned above, this requires understanding the type of initial conditions used in the $\text{filter}$ command.

\(^1\)One criticism of the MSE measure of distortion is that it is an arbitrary mathematical criterion which might not correspond to human subjective judgement.
Because an AR filter is recursive (i.e., has \textit{dynamics}), it can be \textit{unstable} if poor values of the filter coefficients are learned or constructed.\footnote{On the other hand, an MA filter (which is just a simple weighted running average of the inputs) is \textit{never} unstable, which is one reason that they are so widely used for DSP applications.} This will likely be more of an issue when you are asked to quantize the filter coefficients below, and you should not encounter this problem at this stage.

Listen to the signal $\hat{y}$ and compare to the original signal $y$ and the directly quantized signal $y_q$.

\textbf{Step 6.} Calculate the MSE for the reconstructed signal $\hat{y}(n)$ and compare it with the previously calculated MSE for the directly quantized signal. You should plot the errors $(y(n) - y_q(n))$ and $(y(n) - \hat{y}(n))$ to compare their sizes. In your written description of the project you should discuss your results objectively (e.g., is the MSE smaller for the reconstructed signals?) and subjectively (e.g., is the perceived quality of speech better?) Remember to discuss and compare the results for different quantization levels.

\textbf{Step 7.} In reality, \textit{quantized} filter coefficients, $a_q(k)$, are sent and received. Quantize the vector of filter coefficients determined in Step 3,

\begin{verbatim}
>> q = (max(a)-min(a))/L;
>> aq = round(a/q)*q;
\end{verbatim}

and then repeat Steps 4 and 5, replacing the coefficients, $a(k)$, by the quantized coefficients, $a_q(k)$. Quantizing the filter coefficients can lead to an unstable AR filter. How to avoid this problem is discussed in graduate courses in speech compression. If you encounter instabilities which you cannot resolve. Indicate this in your report write-up and suggest that further study of the problem is warranted.