Background on Compression of Speech Signals

Speech, and other audio, signals represented by sample data \( Y^N = \{ y(n), n = 1, 2, \ldots, N \} \), are often quantized to a low bit rate during data transmission in order to obtain faster data transfer rates. Unfortunately, direct quantization of the speech signal, resulting in \( y_q(n) \), will have an undesirable degrading effect on the quality of the speech signal.

One way to reduce the degradation is to first fit an autoregressive (aka, all pole; aka, infinite impulse response (IIR)) digital filter model to the original audio signal \( y(n) \) of the form

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

Note that this can be written equivalently as the moving average (aka, all zero; aka, finite impulse response (FIR)) digital filter model

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

Equation (2) shows that given the filter coefficients \( a(k) \) one can easily compute the residuals \( e(n) \) via an FIR moving average digital filtering of the audio signal \( y(n) \). On the other hand, equation (1) shows that one can easily reconstruct the audio signal \( y(n) \) via an autoregressive (AR) filtering of the residuals \( e(n) \). Thus the audio signal and the residuals contain exactly the same information.

More generally, one could have a time-varying model where the filter coefficients, \( a(n, k) \) depend upon both \( n \) and \( k \). Such models are generally intractable for data-driven learning purposes and one tries to work in domains where the so-called stationary model shown in eq. (1) is valid. In order to ensure a reasonably good approximation to the stationarity assumption, it will be necessary to break the entire set of audio data \( Y^N \) into smaller blocks of data such that within each individual block the stationarity assumption will be assumed to hold. However, within each block a different set of filter coefficients values will be the case. A key goal of this assignment is to learn the filter coefficients within each block by determining data-block dependent least squares estimates of the digital filter coefficients.

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 \( \pm 0.1 \) (relative to the signal magnitude). To minimize the degradation effects of quantization, instead of quantizing the audio signal \( y(n) \) one can quantize the residuals \( e(n) \), then transmit only the quantized residuals, \( e_q(n) \), the coefficients.

\[\text{Assuming that one has values of the initial conditions } y(0), y(-1), \ldots, y(-9). \text{ More on this later.}\]
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 directly quantized signal $y_q(n)$ was sent, $\text{MSE} = \frac{1}{N} \sum_n (y(n) - y_q(n))^2$.

**Assignment Description** 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 least-squares estimates of the coefficients, $a(k)$, of the model (1) and residuals, $e(n)$, for a given speech signal sample data set $Y^N = \{y(n), n = 1, \cdots N\}$, then calculate the difference in the MSE for the directly quantized speech signal $y_q(n)$ and the signal $\hat{y}(n)$ reconstructed from the quantized residuals. You are to listen to the original signal, $y(n)$, the directly quantized signal, $y_q(n)$, and the reconstructed signal $\hat{y}(n)$ and described any subjectively perceived differences.

Because the stationary 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 separately for each block. This means that you will have to decompose the entire data sample into sequential, non-overlapping blocks of 160, process each block separately (to compute $y_q(n)$ and $\hat{y}(n)$ within each separate block) and then reassemble the blocks to get the entire set of signal samples $y_q(n)$ and $\hat{y}(n)$, for $n = 1, \cdots N$.

**Procedural Details**

**Step 1.** 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 `wavread`,

```matlab
>> y = 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 `sound` command in Matlab,

```matlab
>> 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,

```matlab
>> q = (max(y)-min(y))/L;
>> yq = round(y/q)*q;
```

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,

```matlab
>> MSE = (y-yq)'*(y-yq)/N;
```
Is there any relationship between MSE (a mathematical measure of distortion) and your subjective perceptual opinion of quality based on listening to the sound?\(^2\)

**Step 3.** Estimate the filter coefficients \(a(k), k = 1, \cdots, 10\) for each separate block of 160 data speech data points using the least squares optimization technique discussed in class by setting an \(Aa = b\) linear inverse problem. The Matlab command `toeplitz` might be useful in constructing the matrix \(A\). Type ‘help toeplitz’ in Matlab to see how it is used. You will have to use the last 10 elements of the previous block (beginning with the second block) in order to construct the \(A\) matrix needed to estimate the vector of filter coefficients \(a\).\(^3\) The the first block can be initialized with zeros.

**Step 4.** Since there will not be an exact solution to \(Aa = b\), you can calculate the residual error within each block as,

\[
>> \quad e = b - A*a;
\]

**Step 5.** Write a small FIR digital filter program that computes the residuals directly from eq. (2) and the audio data \(y(n)\) using the filter coefficients learned in Step 3. For each data block, form the MSE between these residuals and the residuals computed in Step 4. *This MSE should be identically zero.*

**Step 6.** Write a small AR digital filter program that reconstructs the audio data directly from eq. (1) and the residuals computed in Steps 4 and 5 above using the filter coefficients learned in Step 4.\(^4\) For each data block, form the MSE between the original audio signal samples and the reconstructed audio signal samples. *This MSE should be identically zero.*

**Step 7.** Quantize the residuals using the same quantization rates as in Step 2, but dependent now, however, on the maximum of the residuals in each block of data. Then reconstruct an estimate of the original audio signal samples from the quantized residuals using the AR digital filter you wrote in Step 6. Listen to the signal \(\hat{y}\) constructed for the various quantization rates and compare to the original signal \(y\) and the directly quantized signal \(y_q\).

**Step 8.** Calculate the MSEs for the reconstructed (quantization-rate dependent) signals \(\hat{y}(n)\) and compare it with the previously calculated MSEs for the directly quantized signal. You should plot the errors \((y(n) - y_q(n))\) and \((y(n) - \hat{y}(n))\) as a function of quantization rate 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.

**Step 9.** In reality, quantized filter coefficients, \(a_q(k)\), are sent and received. Quantize the vector of filter coefficients determined in Step 3,

\[
>> \quad q = (\max(a) - \min(a))/L;
\]

\[
>> \quad a_q = \text{round}(a/q)*q;
\]

---

\(^2\)One criticism of the MSE measure of distortion is that it is an arbitrary mathematical criterion which might not correspond to human subjective judgement.

\(^3\)In order to obtain the initial condition values \(y(0), y(-1), \cdots, y(-9)\) needed in (1).

\(^4\)As you sequentially reconstruct the blocks, save the last 10 reconstructed audio samples in each block to use as initial conditions in the subsequent block.
and then repeat Steps 7 and 8, replacing the coefficients, $a(k)$, by the quantized coefficients, $a_q(k)$.

Do not be surprised if the quantized AR filter is unstable (in which case you will not be able to do this Step). Stability of AR filters is a very difficult subject. Unlike FIR filters which are always guaranteed to be stable (which is one reason FIR filters are liked so much by the signal processing and communications engineering communities). An advanced course in speech and audio compression describes how to obtain stable discretized AR digital filters.