Introduction to Source Coding/Data Compression

Course is about the Theory and Practice of Source Coding, a.k.a. Data Compression

Data compression is process of encoding data from some source into bits in such a way that it can be decoded back into a reproduction of the original data.

Source code = data compressor = data compression system = encoder + decoder

encoder creates bits, decoder creates reproduction from bits

Goals:
- efficiency: as few bits as possible
- accuracy, fidelity: reproduction as much like original as possible

Source is assumed to produce discrete-time samples or symbols
- e.g. text or samples of speech
- we won't spend significant time on sampling issues; just assume the source is already sampled
- source will be modelled as a random process, usually stationary and ergodic
- why assume random?? because if not, why encode it? because can exploit statistical characteristics (what occurs more frequently, what values, what combinations of values (correlation))
- autoregressive Gauss-Markov (AR) processes make nice tractable models of speech and image sources.
Rate is our measure of efficiency
rate = number of bits/sample
AVOID "compression ratio"
why encode into bits?? no big deal, just the most useful convention

Average Distortion wrt some distortion measure is our measure of fidelity
satisfactory human perception is usually the "ultimate" criteria
most commonly MSE

\[
\text{empirical distortion} = D = \frac{1}{n} \sum_{i=1}^{n} (X_i - Y_i)^2
\]

\[
\text{statistical distortion} = D = E \left[ \frac{1}{n} \sum_{i=1}^{n} (X_i - Y_i)^2 \right] \text{ or } E(X-Y)^2
\]

why MSE? pro's and cons
other distortion measures
usually empirical = statistical or else we're wasting our time with statistical

Summary: code performance on a given source is characterized by rate and distortion

Lossless coding is when \( \hat{X} \) must equal \( X \); i.e. \( D = 0 \)

Lossy coding is the other case
course is 3/4's lossy, 1/4 lossless, projects are mostly lossy, so we begin with lossy
Complexity is other big issue
implementation complexity
number of arithmetic operations per sample
bytes of auxiliary storage, e.g. for tables
these influence:

building cost -- cost of building or buying hardware for computing and storing

running/operating cost -- cost of operating (depreciation, power, heat, rental, or sharing of resources)

design complexity is a lesser issue
performance vs. complexity
We're not concerned with **channel errors**

Though it's possible to build source codes that are terribly sensitive to channel errors, it is also possible to build them that are not. Any source code can be "fixed" so it is not too sensitive to errors, with only small loss. Typically, \( p = 10^{-4} \) is small enough for speech and images, and even \( 10^{-3} \).

Shannon says that an optimum communication system can have a separation of source code and channel code.

But there are many situations where we're just storing data and the storage medium is so reliable that it doesn't make sense to model it as a noisy channel.

Working through lossy coding (quantization) with channel errors makes interesting exercises.
## Typical Examples of Lossy Compression

<table>
<thead>
<tr>
<th>Source</th>
<th>Uncompressed</th>
<th>Compressed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speech</td>
<td>64 Kbps</td>
<td>9.6K bps (CELP)</td>
</tr>
<tr>
<td>B/W Images</td>
<td>8 bpp</td>
<td>1 bpp (JPEG)</td>
</tr>
<tr>
<td>Color Images</td>
<td>24 bpp</td>
<td>1.25 bits/pixel</td>
</tr>
<tr>
<td>Video</td>
<td>100M bps</td>
<td>.01-20M bps</td>
</tr>
<tr>
<td>Audio</td>
<td>1.4M bps</td>
<td>256K bps (MPEG)</td>
</tr>
</tbody>
</table>

## Typical Example of Lossless Compression

| English Text    | 7 bits/symbol | 3 bits/symbol |

January 6, 2003