Arithmetic coding

1. (a) Under a first order Markov model:

$$
\begin{aligned}
p(\vec{x}) &= p(A_1 A_2 A_2 A_1) \\
&= p(X_4 = A_1 | X_3 = A_2)\, p(X_3 = A_2 | X_2 = A_2)\, p(X_2 = A_2 | X_1 = A_1)\, p(X_1 = A_1) \\
&= \frac{1}{4} \cdot \frac{3}{4} \cdot \frac{1}{8} \cdot \frac{5}{8} \\
&= \frac{15}{1024}
\end{aligned}
$$

(b) $pred(\vec{x}) = pred(A_1 A_2 A_2 A_1) = A_1 A_2 A_1 A_2$

(c)

$$
\begin{aligned}
l_0 &= 0 \\
u_0 &= 1 \\
l_1 &= 0 \\
u_1 &= \frac{5}{8} \\
l_2 &= \frac{35}{64} \\
u_2 &= \frac{5}{8} \\
l_3 &= \frac{145}{256} \\
u_3 &= \frac{5}{8} \\
l_4 &= \frac{145}{256} \\
u_4 &= \frac{595}{1024} \\
T(x) &= \frac{1175}{2048}
\end{aligned}
$$

2. The codeword begins $00110101001\ldots$. We can write

$$
\begin{aligned}
.00110101001 &= \frac{1}{8} + \frac{1}{16} + \frac{1}{64} + \frac{1}{256} + \frac{1}{2048} \\
&= \frac{256}{2048} + \frac{128}{2048} + \frac{32}{2048} + \frac{8}{2048} + \frac{1}{2048} \\
&= \frac{425}{2048}
\end{aligned}
$$

We need to choose an l_k, u_k sequence that contains this value.

k	l_k	u_k	symbol
0	0	1	
1	0	$\frac{3}{8}$	A_1
2	$\frac{9}{64}$	$\frac{15}{64}$	A_2
3	$\frac{51}{256}$	$\frac{60}{256}$	A_3
4	$\frac{408}{2048}$	$\frac{435}{2048}$	A_1
5	$\frac{424.875}{2048}$	$\frac{435}{2048}$	A_3

Indeed, we can continue in this way and show $k = 6$ and $k = 7$ symbols are both A_1. (Do this on your own if you have extra time on your hands. You may with to use a larger power of 2 for the denominator rather than fractions.)

Thus, we can conclude that

$$\vec{x} = A_1 A_2 A_3 A_1 A_3 A_1 A_1...$$

Since we assume that $X_0 = 0$, it follows that the original sequence is :

$$(X_0, X_1, X_2, ...) = (0, -1, -1, 0, -1, 0, -1, -2, ...)$$

3. (a) $pred(\vec{x}) = (0, 1, 1, 0)$

   (b) $p(\vec{x}) = (\frac{1}{4})(\frac{3}{4})^3 = \frac{27}{256}$

   (c)

k	X_k	l_k	u_k
1	0	0	$\frac{1}{4}$
2	1	$\frac{1}{16}$	$\frac{4}{16}$
3	1	$\frac{7}{64}$	$\frac{16}{64}$
4	1	$\frac{37}{256}$	$\frac{64}{256}$

   $T(\vec{x}) = \frac{u_4 + l_4}{2} = \frac{\frac{101}{256}}{2} = \frac{101}{512}$, which in binary is: .001100101

   (d) $\#bits = \lceil \log \frac{2}{p(\vec{x})} \rceil = \lceil \log \frac{512}{27} \rceil = 5$, $C(\vec{x}) = .00110$

   (e) $\lambda(\vec{x}) \le 4 \Leftrightarrow \lceil \log \frac{2}{p(\vec{x})} \rceil \le 4 \Leftrightarrow p(\vec{x}) \ge \frac{1}{8}$. The only sequence with $p(\vec{x}) \ge \frac{1}{8}$ is $(1, 1, 1, 1)$.

Transform and Predictive Coding

1. We choose a that minimizes,

$$\sum_{j=2}^{n}(x_j - ax_{j-1})^2$$

Taking the derivative of the above expression with respect to a, and setting it to 0 yields:

$$\sum_{j=2}^{n}(x_j - ax_{j-1})\, x_{j-1} = 0$$

$$\sum_{j=2}^{7} x_j x_{j-1} = 39.18$$

$$\sum_{j=2}^{7} x_{j-1} x_{j-1} = 54.85 \tag{1}$$

Thus, we set $a = 39.18/54.85 = .7143$.

The value of a depends on the correlation of one sample with its neighbor. If the correlation is high, then a will be close to 1. (e.g. if the sequence x_j were constant, then a would be 1.) If neighboring samples are independent, then the correlation would be 0 (or near zero, since we are estimating from a finite number of samples). If the correlation between neigbhoring samples were negative, then a would be negative.

2. We have a set of samples of (X_1, X_2). To decorrelate, we estimate correlations

$$\begin{vmatrix} E(X_1^2) & E(X_1 X_2) \\ E(X_1 X_2) & E(X_2^2) \end{vmatrix} \approx \frac{1}{8} \begin{vmatrix} 14 & 8 \\ 8 & 26 \end{vmatrix}$$

The eigenvvectors are $\frac{1}{\sqrt{5}}(1, 2)^T$ and $\frac{1}{\sqrt{5}}(2, -1)^T$ respectively.

The tricky part of the question is that I require the encoder and decoder to use integer operations only. From the eigenvectors above, we use the transform $Y = \mathbf{U}^T X$, where

$$\mathbf{U}^T = \begin{vmatrix} 1 & 2 \\ 2 & -1 \end{vmatrix}$$

and then quantize the Y values. But notice that the values would be a factor $\sqrt{5}$ bigger than if the $\mathbf{U}$ matrix were orthonormal. If we use a quantizer of $\Delta = 1$, then we will use too many bits.

The counteract the stretching of the same that is due to non-orthonormal $\mathbf{U}$, we use a larger Δ values. e.g. $\Delta = 2$ would be more appropriate since 2 is close to $\sqrt{5}$.

3. (a) 2^{16}

   (b) $2^{17} - 1$

(c) There is no single correct answer here, but the basic idea is that differential coding works better when the entropy of the intensity differences is less than the entropy of the raw intensities.

Audio signals are measurements of pressure with mean 0 and so the raw signal and the difference signal both have mean 0 as well. Thus, we are comparing two distributions with mean zero. If we don't have any information other than what's given in the question, we could check if $E(X_i - X_{i-1})^2 < E(X_i^2)$. If so, then differential coding is likely to use fewer bits (else, coding the raw X_i's is likely to use fewer bits).

4. (a)
$$\sum_{i_1=0}^{m-1}\sum_{i_2=0}^{m-1}(X_{j+1}(k_1 + i_1, k_2 + i_2) - \hat{X}_j(k_1 + i_1 - v_1, k_2 + i_2 - v_2))^2$$

(b) 1,3,2,5,4,7,6,8.

(c) One reason for the failure is that something might appear in only one frame (the B frame). Another possibility is that there is a block in a B frame in the middle of a long sequence of B frames such that it doesn't have anything in common with the corresponding I or P frames. To solve this problem one could encode that B frame block as an I frame block (i.e. no predition). Indeed, this is what MPEG does.

5. (a) The a_i's are used as weights for the previous m elements of the sequence when predicting the next value.

(b) $\sum_{j=m}^{n-1}(x_{j+1} - \sum_{i=0}^{m-1} x_{j-i}a_i)^2$

(c) We are comparing two values of m here, say m and m'. Suppose $m' > m$. Then the error for m' can be no greater than the error for m because the error for m is just what we would get if we set each a_i's to 0 for $i > m$.

(d) Linear predictive coding is based on the *decoder's estimate* of x_j's, not on the true values of x_j's, whereas the error formula is based only on the true values. Thus, there is no guarentee that the a_i's are optimal for linear prediction. Moreover, we don't use the a_i's exactly, but rather we use a quantized version of them. Finally, the error formula of (b) is somewhat arbitrary. Why least squares? There is no guarentee that minimizing this error means you use fewest bits, on average.

6. (a) With $N = 6$ and $\Delta = 2$, we have six levels: -5, -3, -1, 1, 3, 5.

actual	encoded	estimated
55	55	55
58	3	58
65	5	63
71	5	68
64	-3	65
61	-3	62

(b) • Δ too small. The problem here is that the decoder cannot keep up with the changes in the sequence, since the change for each element is at most $\pm\Delta$. For example, if we have $\Delta = 1$, a sequence (1,3,5,7,9) will be decoded as (1,2,3,4,5). Note also that the error is unbounded.

- Δ too large: The problem here is that most errors will be granular errors and, since $N = 3$, the decoder will be told that there is no change. So, many values will be quantized to the same level when the difference between neighbors is small. For example, if we have $\Delta = 20$, a sequence (1,3,5,7,9) will be decoded as (1,1,1,1,1). However, the error is bounded by $|Q(x) - x| < \frac{\Delta}{2}$.