

## MAS 160/510 Problem Set Four

### 1. Lengths of the FIR filters (*DSP First 5.5*)

### 2. Discrete-Time Signals

Consider the discrete-time signal  $y[n]$ , defined as:

$$y[n] = \begin{cases} -\frac{2}{3}, & n = -2 \\ \frac{1}{2}, & n = -1 \\ 2, & n = 0 \\ \frac{2}{3}, & 1 \leq n \leq 2 \\ 0, & \text{elsewhere} \end{cases}$$

- (a) Sketch and label  $y[3 - n]$ .  
 (b) Sketch and label  $y[n]u[-n]$ , where  $u[n]$  is the unit-step signal:

$$u[n] = \begin{cases} 0, & n < 0 \\ 1, & n \geq 0 \end{cases}$$

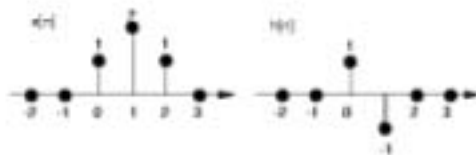
- (c) Express  $y[n]$  in terms of  $\delta[n]$  and in terms of  $u[n]$ .

### 3. Unit-step and running average (*DSP First 5.5*)

### 4. Convolution

For each of the following sets of signals, compute their convolution (1) graphically by hand, (2) with MATLAB (you may use the `conv` function), and (3) by expressing the signals in terms of  $\delta[n]$  and computing the convolution sum. In MATLAB, plot your results with `stem`, but be sure to fix the  $n$ -axis appropriately (use `stem(n,y)` where  $n$  is a vector of the appropriate range).

(a)



(b)



For each of the following of signals, compute their convolution with  $x[n] = \cos(2\pi(\frac{1}{16})n)$  using MATLAB (you may use the `conv` function). Use `stem` to plot your result over the range  $[0:99]$ , assuming the sinusoid exists for all time. Compare each convolution with  $x[n]$ .

(c)  $h[n] = \frac{1}{2}\delta[n] + \frac{1}{2}\delta[n - 1]$

(d)  $h[n] = \delta[n] - \delta[n - 1]$

5. **Markoff processes, entropy, and grading ;)**

A particularly lazy teaching assistant is faced with the task of assigning student grades. In assigning the first grade, he decides that the student has a 30% chance of getting an A, a 40% chance of getting a B, and a 30% chance of getting a C (he doesn't give grades other than A, B, or C). However, as he continues to grade, he is affected by the grade he has just given. If the grade he just gave was an A, he starts to feel stingy and there is less chance he will give a good grade to the next student. If he gives a C, he starts to feel guilty and will tend to give the next student a better grade. Here is how he is likely to grade given the previous grade:

If he just gave an A, the next grade will be: A (20% of the time), B (30%), C (50%).

If he just gave a B, the next grade will be: A (30%), B (40%), C(30%).

If he just gave a C, the next grade will be: A (40%), B (50%), C(10%).

- (a) Draw a Markoff graph of this unusual grading process.
- (b) Calculate the joint probability of all successive *pairs* of grades (i.e. AA, AB, AC, etc.)
- (c) Calculate the entropy,  $H$ , of two successive grades given.

6. **Time-domain response of FIR filters (*DSP First 5.6*)**

7. **Entropy Coding**

Often it is the case that a set of symbols we want to transmit are not equally likely to occur. If we know the probabilities, then it makes sense to represent the most common symbols with shorter bit strings, rather than using an equal number of binary digits for all symbols. This is the principle behind variable-length coders.

An easy-to-understand variable-length coder is the Shannon-Fano code. The way we make a Shannon-Fano code is to arrange all the symbols in decreasing order of probability, then to split them into two groups with approximately equal probability totals (as best we can, given the probabilities we have to work with), assigning 0 as an initial code digit to the entries in the first group and 1 to those in the second. Then, keeping the symbols in the same order, we recursively apply the same algorithm to the two groups till we've run out of places to divide. The pattern of ones and zeros then becomes the code for each symbol. For example, suppose we have an alphabet of six symbols:

Symbol	% probability	Binary code	Shannon-Fano code
A	25	000	00
B	25	001	01
C	25	010	10
D	12.5	011	110
E	6.25	100	1110
F	6.25	101	1111

Let's see how much of a savings this method gives us. If we want to send a hundred of these symbols, ordinary binary code will require us to send 100 times 3 bits, or 300 bits. In the S-F case, 75 percent

of the symbols will be transmitted as 2-bit codes, 12.5 as 3-bit codes, and 12.5 as 4-bit codes, so the total is only 237.5 bits, on average. Thus the binary code requires 3 bits per symbol, while the S-F code takes 2.375.

The entropy, or “information content” expression gives us a lower limit on the number of bits per symbol we might achieve.

$$\begin{aligned}
 H &= - \sum_{i=1}^m p_i \log_2(p_i) \\
 &= -[0.25 \log_2(0.25) + 0.25 \log_2(0.25) + 0.25 \log_2(0.25) + 0.125 \log_2(0.125) \\
 &\quad + 0.0625 \log_2(0.0625) + 0.0625 \log_2(0.0625)]
 \end{aligned}$$

If your calculator doesn’t do base-two logs (most don’t), you’ll need the following high-school relation that many people forget:

$$\log_a(x) = \log_{10}(x) / \log_{10}(a),$$

so

$$\log_2(x) = \log_{10}(x) / 0.30103.$$

And the entropy works out to 2.375 bits/symbol. So we’ve achieved the theoretical rate this time. The S-F coder doesn’t always do this well, and more complex methods like the Huffman coder will work better in those cases (but are too time-consuming to assign on a problem set!).

Now it’s your turn to do some coding. The below is a letter-frequency table for the English language (also available at <http://ssi.www.media.mit.edu/courses/ssi/y01/ps4.freq.txt>) :

<b>Letter</b>	<b>% probability (number per hundred letters)</b>
E	13.105
T	10.468
A	8.151
O	7.995
N	7.098
R	6.832
I	6.345
S	6.101
H	5.259
D	3.788
L	3.389
F	2.924
C	2.758
M	2.536
U	2.459
G	1.994
Y	1.982
P	1.982
W	1.539
B	1.440
V	0.919
K	0.420
X	0.166
J	0.132
Q	0.121
Z	0.077

- (a) Twenty-six letters require five bits of binary. What’s the entropy in bits/letter of English text coded as individual letters, ignoring (for simplicity) capitalization, spaces, and punctuation?

- (b) Write a Shannon-Fano code for English letters. How many bits/letter does your code require?
- (c) Ignoring (as above) case, spaces, and punctuation, how many total bits does it take to send the following English message as binary? As your code? [You don't need to write out the coded message, just add up the bits.]

*"There is too much signals and systems homework"*

- (d) Repeat (c) for the following Clackamas-Chinook sentence (forgive our lack of the necessary Native American diacritical marks!).

*"nugwagimx lga dayaxbt, aga danmax wilxba diqelpxix."*