

1. Longitudes de los filtros FIR (DSPFirst 5.5)

2. Señales en tiempo discreto

Considere la señal en tiempo discreto $y[n]$, definida como:

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

- (a) Represente y clasifique $y[3-n]$.
- (b) Represente y clasifique $y[n]u[-n]$, donde $u[-n]$ es la señal de paso unitario:

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

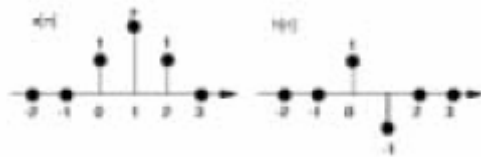
- (c) Exprese $y[n]$ en términos de $\delta[n]$ y en términos de $u[-n]$.

3. Paso unitario y promedio de ejecución (DSPFirst 5.5)

4. Convolución

Para cada uno de los siguientes conjuntos de señales, calcule su convolución (1) gráficamente a mano, (2) con MATLAB (puede utilizar la función **conv**) y (3) expresando las señales en términos de $\delta[n]$ y calculando la suma de convolución. En MATLAB, represente los resultados con la función **stem**, pero no olvide ajustar el eje x adecuadamente (utilice **stem(n,y)**, donde n es el vector del rango correspondiente).

(a)



(b)



Para cada una de las siguientes señales, calcule su convolución con $x[n] = \cos(2\pi(1/15)n)$ utilizando MATLAB (puede utilizar la función **conv**). Utilice **stem** para representar el resultado en el rango [0:99] asumiendo que el senoide existe en todo el intervalo de tiempo. Compare cada convolución con $x[n]$.

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

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

5. Procesos de Markoff, entropía y asignación de notas

Un profesor adjunto especialmente vago se enfrenta a la tarea de asignar calificaciones a los estudiantes. Para asignar la primera nota decide que el estudiante tiene un 30% de posibilidades de obtener una A, un 40% de posibilidades de obtener una B y un 30% de posibilidades de obtener una C (no califica con notas que no sean A (10/10), B (7,5/10) y C (5/10)). Sin embargo, a medida que va poniendo notas, éstas le van influyendo sobre las siguientes que debe asignar. Así, si la nota que acaba de dar es una A, no se siente lo suficientemente generoso para dar una nota semejante al siguiente alumno. Por su parte, si da una C, se siente culpable y tiende a dar una nota mejor al siguiente estudiante. A continuación se indica la tendencia que tiene a dar una nota según la que haya dado anteriormente:

Si acaba de dar una A, la siguiente nota será: A (20% de las veces), B (30%), C (50%)

Si acaba de dar una B, la siguiente nota será: A (30%), B (40%), C (30%)

Si acaba de dar una C, la siguiente nota será: A (40%), B (50%), C (10%)

- Dibuje el gráfico de Markoff de este proceso inusual de calificación
- Calcule la probabilidad combinada de todos los pares sucesivos de notas (p.ej., AA, AB, AC)
- Calcule la entropía, H, de dos notas sucesivas dadas

6. Respuesta de filtros FIR en el dominio del tiempo (*DSPFirst 5.6*)

7. Codificación de entropía

A menudo nos encontramos con casos en los que la probabilidad de que exista un conjunto de símbolos que queremos transmitir no es la misma. Si conocemos las probabilidades, parecería lógico representar los símbolos más comunes con cadenas de bits más cortas en lugar de utilizar el mismo número de dígitos binarios para todos los símbolos. Éste es el principio que se esconde tras los codificadores de longitud variable.

Un codificador de longitud variable fácil de entender es el codificador de Shannon-Fano. El procedimiento para crear uno consiste en organizar todos los símbolos en orden decreciente de probabilidad y, después, dividirlos en dos grupos de probabilidad aproximada (de la mejor forma posible teniendo en cuenta las probabilidades con las que tengamos que trabajar). Asignamos 0 como dígito inicial del código a las entradas del primer grupo y 1 a las del segundo. A continuación, y conservando el orden de los símbolos, aplicamos repetidamente el mismo algoritmo a los dos grupos hasta que no nos queden más espacios por dividir. El patrón de unos y ceros se convierte entonces en el código para cada símbolo. Por ejemplo, supongamos un alfabeto de seis símbolos:

Símb.	% prob.	Cod. bin	Cod. Shannon-Fano
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

Veamos qué ahorro nos supone este método. Si queremos enviar un centenar de estos símbolos, el código binario normal nos obligará a enviar 3 bits 100 veces, o lo que es lo mismo, 300 bits. En el caso del S-F, el 75 por ciento de los símbolos se transmitirán como códigos de 2 bits, un 12,5% como códigos de 3 bits, y un 12,5%

como códigos de 4 bits, por lo que el total, de media, será sólo de 237,5 bits. De esta forma, el código binario requiere 3 bits por símbolo, mientras que el código S-F, por su parte, requiere 2,375.

La entropía (o contenido de información) limita en menor medida el número de bits por símbolo que podemos obtener.

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

Si su calculadora no puede trabajar con logaritmos en base 2 (la mayoría no lo hacen), necesitará la siguiente relación que todos aprendimos en el colegio y que siempre olvidamos:

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

por lo que

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

Así, la entropía obtiene un valor de 2,375 bits por símbolo y hemos conseguido esta velocidad teórica. El codificador S-F no siempre funciona así de bien y otros métodos más complejos, como el codificador de Huffman, arrojan mejores resultados en otros casos (pero son demasiado laboriosos para utilizarlos en un boletín de problemas).

Ahora es su turno. Abajo se incluye una tabla de frecuencias y letras (también disponible en la dirección <http://ssi.www.media.mit.edu/courses/ssi/y01/ps4.freq.txt>):

Letra	% prob (número por cada cien letras)
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) Veintiséis letras necesitan cinco bits de código binario. ¿Cuál es la entropía en bits/letra codificada como letras por separado, obviando (para mayor simplicidad) las mayúsculas, los espacios y los signos de puntuación?

- (b) Escriba un código de Shannon-Fano para las letras del alfabeto. ¿Cuántos bits/letra necesitará para hacerlo?
- (c) Obviando (tal como hicimos en el caso anterior) las mayúsculas, los espacios y los signos de puntuación, ¿qué número total de bits se necesitan para enviar el siguiente mensaje redactado en inglés como código binario? ¿Y en el código que ha redactado usted mismo? [No necesita escribir el mensaje en código, sino simplemente añadir los bits.]

“There is too much signals and systems homework”

- (d) Repita el apartado (c) para la siguiente frase escrita en el dialecto indígena Clackamas-Chinook.

“nugwagimx lga dayaxbt, aga danmax wilxba diqelpxix”