Trabajo Práctico 1

1. Un sistema de comunicaciones

Toda la programación relativa a este trabajo debe ser realizada en Matlab, Octave o Python (con la biblioteca `numpy`, por ej.). De usar Python, por favor usar Python 3.x.

Un modelo muy simple (discreto en banda base) de un sistema de comunicaciones es el siguiente:

1. El transmisor envía un dato s_k cada T segundos, donde s_0 es enviado en $t = 0$, s_1 en $t = T$, s_2 en $t = 2T$, ...
2. Los datos son modificados por el canal, es decir, por el medio donde son transmitidos. Esa modificación está presentada por la así denominada respuesta al impulso del canal[1] $\{h_k\}_{k=0}^{L}$, donde L es la longitud de la respueta al impulso.
3. Además de ser modificados por el canal, los datos son afectados por ruido blanco Gaussiano aditivo $N_k \sim cN(0, \sigma)$.

Teniendo todo esto en cuenta, cada T segundos el receptor observa:

$$r_n = \sum_{k=0}^{L-1} h_k s_{n-k} + N_n. \tag{1}$$

También podemos expresar la Ec. 1 en forma matricial como

$$\vec{r} = \mathbf{H}\vec{s} + \vec{N}, \tag{2}$$

donde

$$\vec{r} = \begin{pmatrix} r_0 \\ r_1 \\ \vdots \\ r_M \end{pmatrix}, \ \vec{s} = \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{M-1} \end{pmatrix}, \ \vec{N} = \begin{pmatrix} N_0 \\ N_1 \\ \vdots \\ N_{M-1} \end{pmatrix}, \tag{3}$$

$$\mathbf{H} = \begin{pmatrix} h_0 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\ h_1 & h_0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & 0 & 0 & \cdots & 0 & 0 \\ h_{L-1} & h_{L-2} & \cdots & h_1 & h_0 & 0 & \cdots & 0 & 0 \\ 0 & h_{L-1} & \cdots & h_2 & h_1 & h_0 & \cdots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & h_1 & h_0 \end{pmatrix} \in \mathbb{R}^{M\times M}, \tag{4}$$

y $M \geq L$.

Otra forma equivalente es la siguiente:

$$\vec{r} = \mathbf{S}\vec{h} + \vec{N}, \tag{5}$$

[1] En la próxima guía vamos a explicar qué es una respuesta al impulso.

donde $\vec{r}$ y $\vec{N}$ son como antes y

$$\vec{h} = \begin{pmatrix} h_0 \\ h_1 \\ \vdots \\ h_L \end{pmatrix}, \tag{6}$$

$$\mathbf{S} = \begin{pmatrix} s_0 & 0 & \cdots & & 0 \\ s_1 & s_0 & \cdots & & 0 \\ \vdots & & \vdots & & 0 \\ s_{L-2} & s_{L-3} & \cdots & s_0 & 0 \\ s_{L-1} & s_{L-2} & \cdots & s_1 & s_0 \\ s_L & s_{L-1} & \cdots & s_2 & s_1 \\ \vdots & s_{M-1} & s_{M-2} & \cdots & s_{M-L} \end{pmatrix} \in \mathbb{R}^{M\times L}. \tag{7}$$

Lo que nos interesa es recuperar correctamente la señal enviada $\vec{s}$ a partir de la señal recibida $\vec{r}$. Esto no sería tan difícil si se conociesen L y los $\{h_k\}$. En efecto, tomando $M = L$ en la Ec. 2, podemos buscar $\vec{s}$ que minimice:

$$\|\mathbf{H}\vec{s} - \vec{r}\|_2^2 \,. \tag{8}$$

El problema es, sin embargo, que en general ni L ni $\{h_k\}$ son conocidos: estimarlos corresponde al problema de estimación del canal.

Si L es conocido, una forma de estimar el canal es mediante la Ec. 5 y el método de cuadrados mínimos. En efecto, tómese $M > L$ y envíese una señal conocida por el receptor $\vec{s} \in \mathbb{R}^M$ (denominada secuencia de entrenamiento). Luego, se estima el canal planteando el siguiente problema de cuadrados mínimos: encontrar $\vec{h} \in \mathbb{R}^L$ que minimice

$$\left\|\mathbf{S}\vec{h} - \vec{r}\right\|_2^2 \,. \tag{9}$$

Esta parte del trabajo práctico consiste en la estimación de un canal mediante cuadrados mínimos. Más específicamente:

1. Genere una respuesta al impulso del canal aleatoria, con $L = 5$. En Octave, esto se puede hacer de la siguiente manera:

```
L = 5;
ganancia = 1/10;
h = ganancia*(1+randn(L,1));
```

2. Usando ruido con desvío estándar $\sigma = 1$, transmita la imagen de Lena en escala de grises que se le provee y muestre los resultados. La forma de transmitir la imagen de Lena de izquierda a derecha y de arriba hacia abajo. Para facilitar el proceso, transmitiremos una línea a la vez. En Octave, esto se puede hacer de la siguiente manera:

```
sigma = 0.1;
a = imread('lena512.bmp');
M = size(a,2);
P = size(a,1);
H = toeplitz([h.' zeros(1,M-L)],zeros(1,M));
r = zeros(M,P);
N = sigma*randn(M,P); % ruido
% se transmite una línea a la vez
s = double(a(:,:)); % lo que se envía
r = H*s + N; % lo que se recibe
b = uint8(r);
figure(1)
subplot(1,2,1), imshow(a) % muestro la imagen original
subplot(1,2,2), imshow(b) % muestro la imagen recibida
```

3. Estime $\vec{h}$ usando una secuencia de entrenamiento conocida de longitud $E = 512$. Al estimar $\vec{h}$ suponga que $L = 5$. Estime la imagen enviada a partir de sus resultados. Repita para varios valores de $\vec{h}$ y varias realizaciones de ruido. *Se trata de dos problemas: la estimación del canal (de $\vec{h}$) y la estimación de la imagen original.* Para resolver el primero de esos problemas utilice cuadrados mínimos usando:

   - Grupo 1: Descomposición Cholesky.
   - Grupo 2: Descomposición QR.
   - Grupo 3: Descomposición en valores singulares.
   - Grupo n con $n > 3$: lo mismo que fue especificado para el grupo $\mathrm{mod}(n, 4) + 1$.

   Ustedes deben implementar sus propias versiones de los algoritmos de descomposición Cholesky, descomposición QR y Gauss-Seidel.

   El otro problema, la estimación de la imagen original, también se puede realizar haciendo cuadrados mínimos. Aquí utilizaremos otra alternativa sencilla que asume que se conoce el nivel de ruido: *LMMSE - Linear Minimum Mean Square Error*. Para simplificar aún más la resolución, asumiremos (erróneamente) que los símbolos enviados son independientes entre sí. Para comprender mejor lo que se hace, pueden consultar, por ej., Wikipedia (`https://en.wikipedia.org/wiki/Minimum_mean_square_error`, accedido 14 de septiembre de 2017).

   El siguiente código en Octave muestra todo el procedimiento en esta parte:

```
%%%%%%%%%
%Asumo h desconocido
%Entreno con una secuencia de longitud 1024
E = 1024;
sE= unidrnd(256,1024,1)-1; %genero una secuencia (conocida)
a enviar
ME= size(sE,1);
PE= size(sE,2);
HE= toeplitz([h.' zeros(1,ME-L)],zeros(1,ME));
rE= zeros(ME,PE);
NE= sigma*randn(ME,PE); % ruido
% se transmite
rE = HE*sE + NE; % lo que se recibe
% estimo h
S = toeplitz(sE');
S = S(:,1:L);
S = tril(S);
he= S\rE; %Esto lo tienen que implementar ustedes!!!!!
%comparo h con h estimado
[h he]
% estimo la imagen
He= toeplitz([he.' zeros(1,M-L)],zeros(1,M));
CN= sigma^2*eye(size(He)); % matriz de covarianza del ruido
sigmax= sqrt(1/256*(sum(((0:255)-mean(0:255)).^2))); %
dispersión de los símbolos enviados
mx= mean(0:255)*ones(M,1); % media de los símbolos enviados
CX= sigmax^2*eye(size(He)); % matriz de covarianza de los
símbolos enviados - asume independencia
W = CX*He'*inv(He*CX*He'+CN);% matriz de ecualización
d = zeros(M,P);
for k = 1:P
d(:,k) = W*(r(:,k)-He*mx)+mx;
end
d = uint8(d);
figure(2)
subplot(1,3,1), imshow(a) % muestro la imagen original
subplot(1,3,2), imshow(b) % muestro la imagen recibida
subplot(1,3,3), imshow(d) % muestro la imagen estimada
```

4. Repita el ejercicio 3 usando $E = 32$. Comente sus resultados.

5. Repita el ejercicio 3 usando $E = 1024$. Comente sus resultados.

6. Repita los ejercicios anteriores usando $\sigma = 0, 0{,}1, 10, 20, 50, 100$. Comente sus resultados.