**Instituto Tecnológico de Buenos Aires**

93.54 MÉTODOS NUMÉRICOS

# Trabajo práctico N°1

*Grupo 3*

| | |
|---|---|
| FONTECHA, María Eugenia | 58138 |
| LAMBERTUCCI, Guido Enrique | 58009 |
| POUTHIER, Florian | 61337 |
| LONDERO BONAPARTE, Tomás Guillermo | 58150 |

*Profesor*

FIERENS, Pablo Ignacio

Presentado: 11/04/19

# 1. Introducción

## 1.1. Sistema de comunicación

Una forma simple de emular un sistema de comunicación es la presentada a continuación:

1. Un emisor transmite un paquete $s_k$ con un período $T$, es decir, dicha información es enviada en el instante $t = nT$, con $n \in \mathbb{N}_0$.
2. La señal es modificada por el canal en la cual se transmite, dependiendo de la respuesta impulsiva del sistema $\{h_k\}_{k=0}^{L}$, siendo $L$ la longitud de la respuesta al impulso.
3. A lo mencionado anteriormente, se le suma un ruido blanco Gaussiano aditivo $N_k \sim cN(0, \sigma)$.

Una vez establecido lo anteriormente mencionado, se pude modelar el sistema de la siguiente forma:

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

o también, de la forma matricial:

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

con

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

$$H = \begin{pmatrix} h_0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & 0 \\ h_1 & h_0 & 0 & \dots & 0 & 0 & \dots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & 0 & 0 & \dots & 0 & 0 \\ h_{L-1} & h_{L-2} & \dots & h_1 & h_0 & 0 & \dots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & h_1 & h_0 \end{pmatrix}$$

De manera equivalente, se puede escribir la ecuación anterior de la forma:

$$\vec{r} = S\vec{h} + \vec{N} \tag{3}$$

con

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

$$\vec{S} = \begin{pmatrix} s_0 & 0 & \dots & \dots & 0 \\ s_1 & s_0 & \dots & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ s_{L-2} & s_{L-3} & \dots & s_0 & 0 \\ s_{L-1} & s_{L-2} & \dots & s_1 & s_0 \\ s_L & s_{L-1} & \dots & s_2 & s_1 \\ \vdots & s_{M-1} & s_{M-2} & \dots & s_{M-L} \end{pmatrix}$$

Finalmente, el objetivo del trabajo es recuperar la señal emitida $\vec{s}$ partiendo de la recibida $\vec{r}$. Generalmente no se conoce ni $L$ ni $\{h_k\}$, por lo que estos deben ser estimados.

Si $L$ es conocido, y utilizando la ecuación (3) y el método de cuadrados mínimos, se puede encontrar $\vec{h} \in \mathbb{R}^L$ que minimice

$$\|S\vec{h} - \vec{r}\|_2^2 \tag{4}$$

## 1.2. Descomposición en valores singulares

El método de descomposición en valores singulares permite resolver ecuaciones de la forma de la ecuación (4), cuando la matriz $S$ no es de rango completo. Este algoritmo busca expresar una matriz $A \in \mathbb{R}^{mxn}$ (con $m \geq n$) como el producto de tres matrices:

$$A = U \cdot \Sigma \cdot V^T \tag{5}$$

siendo $U$ y $V$ matrices ortogonales y $\Sigma$ una matriz diagonal de valores singulares.

Para poder obtener dicha igualdad, primero se buscan los auto-valores y auto-vectores de $B = A^T \cdot A$. Por un lado, con los auto-valores de $B$ se obtienen los valores singulares y por consiguiente, la matriz $\Sigma$, colocandolos en la diagonal de manera decreciente. Por otro lado, $V$ queda definida como la matriz de de $B$. Luego se definen los vectores columna de la matriz $U$ de la siguiente manera:

$$\vec{u}_i = A\frac{\vec{v}_i}{\sigma_i} \tag{6}$$

Por convención, los valores singulares se ordenan de foma tal que

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > \sigma_{r+1} = \sigma_{r+2} = \ldots = \sigma_n = 0$$

se inventan $m - r$ vectores más, que sean ortonormales con los anteriores $\vec{u}_i$. Por consiguiente, queda definida la matriz $U$ como:

$$U = \begin{pmatrix} \vec{u}_0 & \vec{u}_1 & \dots & \vec{u}_m \end{pmatrix}$$

Retomando el problema planteado para la ecuación (3) y utilizando el metodo explicado, se puede dar como solución:

$$\vec{h} = \sum_{k=1}^{r} \frac{u_k^T \cdot \vec{r} \cdot v_k}{\sigma} \tag{7}$$

# 2. Código empleado

El código elaborado para este trabajo fue hecho en **Matlab**.

- **Main:**

```
function MAIN () L = 5;
ganancia = 0.1;
h = ganancia*(1+randn(L,1));
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); %Noise
s = double(a(:,:));
r = H*s + N; % lo que se recibe
b = uint8(r);
figure(1),title('Ejercicio 1,2')
subplot(1,2,1), imshow(a),title('Original') % la imagen original
subplot(1,2,2), imshow(b),title('Recibida') % muestro la imagen recibida
E = 512;
sE= zeros(E,1);
for i=1:E
    sE(i,1) = rand()*256-1;
end
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 = estimate(S,rE);
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
 %simbolos 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),title('Ejercicio 3')
subplot(1,3,1), imshow(a),title('Original') % muestro la imagen original
subplot(1,3,2), imshow(b),title('Recibida') % muestro la imagen recibida
subplot(1,3,3), imshow(d),title('Estimada') % muestro la imagen estimada
```

- **Función "estimate":**

```
function he = estimate (S, rE)
[U,S,V] = DVS(S);
he = zeros(size(V,2),1);
for k=1:size(V,2)
    he= he+(U(:,k)'*rE*V(:,k))/S(k,k);
end
```

- **Función "DVS":**

```
function [U,S,V] = DVS (A)
m = size(A,1);
n = size(A,2);
Uprov = zeros(m,m);
V = zeros(n,n);
S = zeros(m,n);
B=A'*A;
[V,sigma] = eig(B);
sigma2 = sort(diag(sigma),'descend');
V=fliplr(V);
Sig=sqrt(sigma2);
for k = 1:size(Sig)
    S(k,k)=Sig(k);
end
for k=1:n
    Uprov(:,k)=(A * V(:,k))/Sig(k);
end
if size(Uprov,2) ¡m
    U = completar(Uprov);
else
    U = Uprov;
end
```

- **Función "completar":**

```
function [Ucompleta] = completar(U) %completar U toma una matriz U de m*n
%y devuelve otra matriz de m*m ortonormal
[ m,n ] = size(U);
U = [ U zeros(m, m-n) ];
for i = 1:(m-n)
    vec = rand([m 1]);
```

```
        vecOrt = vec - sum(((vec'*U).*U),2); %ortogonalización de vec
        U(:,n+i) = vecOrt/norm(vecOrt); %normalización del vector
    end
    Ucompleta = U;
    end
```

# 3. Resultados obtenidos

A continuación, se presentan los resultados obtenidos de la simulación. Cabe aclarar que, para poder realizar la estimación, se establecieron valores tales como la longitud de la respuesta impulsiva $L = 5$ y la ganancia $h = 0{,}1$. Para los parámetros de longitud $E$ y el desvío estándar $\sigma$, se repitió el proceso con distintos valores:

- $E = 32$


Figura 1: Iteración con $E = 32$ y $\sigma = 0$.


Figura 2: Iteración con $E = 32$ y $\sigma = 0{,}1$.


Figura 3: Iteración con $E = 32$ y $\sigma = 1$.


Figura 4: Iteración con $E = 32$ y $\sigma = 10$.


Figura 5: Iteración con $E = 32$ y $\sigma = 20$.


Figura 6: Iteración con $E = 32$ y $\sigma = 50$.

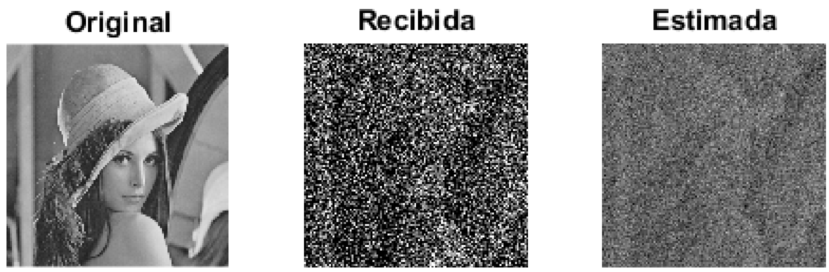


Figura 7: Iteración con $E = 32$ y $\sigma = 100$.

- $E = 512$


Figura 8: Iteración con $E = 512$ y $\sigma = 0$.


Figura 9: Iteración con $E = 512$ y $\sigma = 0{,}1$.


Figura 10: Iteración con $E = 512$ y $\sigma = 1$.


Figura 11: Iteración con $E = 512$ y $\sigma = 10$.


Figura 12: Iteración con $E = 512$ y $\sigma = 20$.


Figura 13: Iteración con $E = 512$ y $\sigma = 50$.

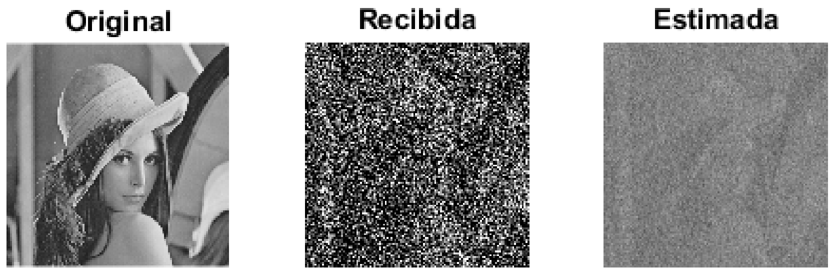


Figura 14: Iteración con $E = 512$ y $\sigma = 100$.

- $E = 1024$


Figura 15: Iteración con $E = 1024$ y $\sigma = 0$.


Figura 16: Iteración con $E = 1024$ y $\sigma = 0{,}1$.


Figura 17: Iteración con $E = 1024$ y $\sigma = 1$.


Figura 18: Iteración con $E = 1024$ y $\sigma = 10$.


Figura 19: Iteración con $E = 1024$ y $\sigma = 20$.


Figura 20: Iteración con $E = 1024$ y $\sigma = 50$.

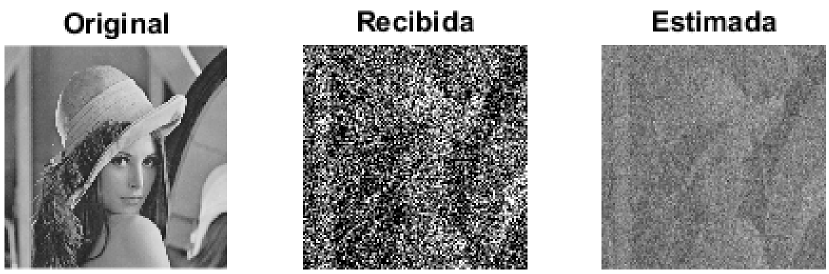


Figura 21: Iteración con $E = 1024$ y $\sigma = 100$.

# 4. Conclusión

Luego de analizar los resultados obtenidos para cada valor de $E$ y de $\sigma$, se llegó a las siguientes conclusiones.

Por un lado, queda evidenciado que, para una longitud E fija, a medida que aumenta el desvío estándar en el ruido Gaussiano, la calidad de la imagen recibida disminuye considerablemente, lo que se traduce a su vez, en una imagen estimada que cada vez se aproxima menos a la original.

Por otro lado, si se varía la longitud y se deja fijo el valor de $\sigma$, se observa que a mayor longitud, mayor calidad de las figuras analizadas, tanto de la recibida como de la estimada.

Se puede concluir que a mayor longitud, el efecto del aumento del desvío estándar es menor que a menor longitud. Por ejemplo, se puede observar que para una longitud de 32, un $\sigma$ de 50 implica una pérdida total de la imagen original, mientras que con una longitud de 512, para el mismo desvío estándar todavía se puede apreciar cierta similitud entre la imagen estimada y la original, a pesar del alto grado de distorsión.

De la misma manera, a menor valor de $\sigma$, las diferencias de la calidad de las figuras estimadas para cada valor de longitud son cada vez más despreciables. Se puede observar cómo, para desvío estándar de 0 y 0.1, las figuras correspondientes a cada longitud parecen ser idénticas entre sí.