Medios de Transmisión

Práctica Final

Simulación de un Sistema de Transmisión Digital Banda Base

Curso 2008-2009

1. Introducción

El objetivo de esta práctica es realizar un programa en Matlab que simule el funcionamiento de un sistema de transmisión digital banda base. El canal se supone que tiene un ancho de banda limitado e introduce un ruido blanco gaussiano. Matlab permite generar de forma bastante fidedigna las señales que aparecen en el sistema de transmisión. Mediante la simulación del sistema se persigue evaluar su calidad en función del ancho de banda del canal y del nivel de ruido.

En la figura 1 está representado el modelo de comunicaciones que se va a utilizar para el desarrollo de la práctica.

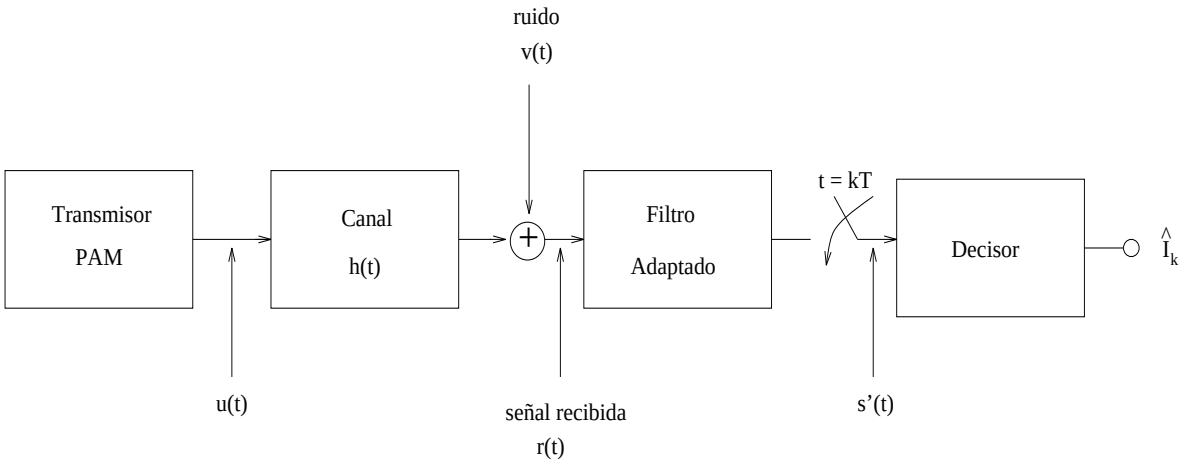

Figura 1: Modelo de un Sistema de Comunicaciones en Banda Base

2. Generación de las señales

1. Se supone que los bits a transmitir son una secuencia aleatoria de 0's y 1's. La probabilidad de transmitir un 0 o un 1 es la misma, es decir, 1/2. Para generar

esta secuencia se puede utilizar la función *randn* de Matlab que genera números aleatoriamente siguiendo una distribución normal de media 0 y varianza 1. Observe que la probabilidad de que los números generados con esta función sean positivos es la misma que la de que sean negativos, es decir, 1/2. Por tanto, cuando el número aleatorio generado es positivo, se supone que se transmite un 1 y cuando es negativo se transmite un 0.

2. Se supone que el transmisor envía una señal PAM binaria que consiste en un tren de pulsos modulados en amplitud. La amplitud será +1 cuando se transmita un cero binario y −1 cuando se transmita un uno binario. La señal transmitida $u(t)$ se representa matemáticamente de la forma

$$u(t) = \sum_{k=-\infty}^{\infty} I_k p(t - kT) \tag{1}$$

donde $p(t)$ es la forma de pulso transmitido, T es el período de símbolo y I_k son las amplitudes discretas (+1 o −1) correspondientes a los símbolos transmitidos. Esta señal se puede aproximar con el Matlab a través de

$$u(n) = \sum_{k=0}^{K-1} I_k p(n - kN) \tag{2}$$

donde K es el número de símbolos que se transmiten, N es el período de símbolo y $p(n)$ es la forma del pulso empleado en la transmisión.

Para generar $u(n)$ se recomienda hacerlo como la convolución de las dos señales siguientes

$$u(n) = \left[\sum_{k=0}^{K-1} I_k \delta(n - kN)\right] * p(n) = s(n) * p(n) \tag{3}$$

La señal $p(n)$ debe almacenarse en un vector llamado `pulso`. El programa debe funcionar indistintamente tanto para pulsos de duracion limitada como ilimitada (pulsos de Nyquist). En todos los casos debe definirse el vector `pulso` en el intervalo $[-LN, LN]$ donde L es el número de períodos de símbolo que se utilizan para representar $p(n)$. Observe que, si el pulso es de duracion limitada, el vector `pulso` es todo ceros excepto en el intervalo $[0, N-1]$. Una de las formas de pulso escogidas debe ser el Pulso de Nyquist:

$$p(n) = \frac{sen(\pi n/N)}{(\pi n/N)} \tag{4}$$

donde N es el período de símbolo.

La energía de bit E_b es la energía que se consume cuando se transmite un bit. E_b puede calcularse de la forma

$$E_b = \sum_{n=LN}^{LN} |p(n)|^2 \tag{5}$$

que es la versión en discreto del cálculo de la energía de una señal contínua [1].

[1] Para calcular E_b le puede ser de utilidad la instrucción `sum`

3. Para generar la señal recibida $r(n)$ debe de hacerse la convolución entre la señal transmitida $u(n)$ y la respuesta al impulso del canal $h(n)$ y sumar una señal de ruido $v(n)$

$$r(n) = u(n) * h(n) + v(n) \tag{6}$$

Como $h(n)$ escoja la respuesta al impluso de un filtro paso bajo ideal

$$h(n) = \frac{sen(Wn)}{\pi n} \tag{7}$$

donde W es el ancho de banda del canal. La respuesta en frecuencia del canal, $H(\omega)$ tiene la forma que se muestra en la figura 2.

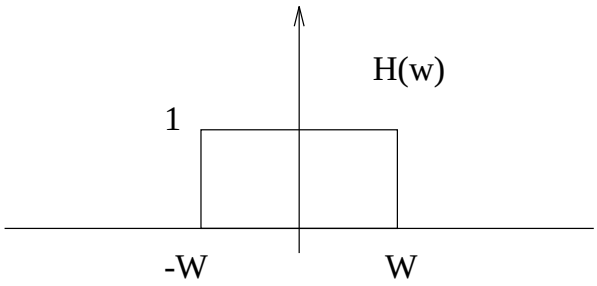

Figura 2: Respuesta en Frecuencia del canal

El ruido blanco gaussiano introducido por el canal puede generarse empleando la instrucción `randn`. Haciendo `randn(1,L)` se genera un vector de ruido blanco gaussiano de longitud L, media nula y densidad espectral de potencia 1. Si se quiere modificar la densidad espectral de potencia $N_0/2$ hay que multiplicar la secuencia anterior por $\sqrt{N_0/2}$.

4. A continuación debe construirse la salida del filtro adaptado lo cual puede hacerse convolucionando la señal recibida con la respuesta al impulso del filtro. Esta respuesta al impulso debe construirse a partir de la forma del pulso de modo que al cambiar la forma del pulso se modifique automáticamente la respuesta al impulso.

5. El siguiente paso es obtener las observaciones a partir de las cuales se toman las decisiones. Para ello deben tomarse muestras de la señal recibida en los puntos $n = kN$ para $k = 0, \cdots, K-1$, obteniéndose la siguiente señal

$$s'(n) = \sum_{k=0}^{K-1} \hat{I}_k \delta(n - kN) \tag{8}$$

donde $\hat{I}_k = r(n = kN)$ son las amplitudes observadas. Tenga en cuenta a la hora de tomar estas muestras el efecto que sobre las longitudes de las señales tiene la operación de convolución.

6. Finalmente, la operación de decisión la realizaremos de la siguiente forma: cuando la amplitud observada es positiva, supondremos que el bit recibido es 1 y cuando

es negativa que el bit recibido es 0. Comparando los bits transmitidos con los bits recibidos, cuente el número de errores que se ha producido en la transmisión y estime la probabilidad de error dividiendo el número de errores entre el número de símbolos transmitidos.

Uno de los objetivos de la simulación es comprobar, en ausencia de ruido, como la transmisión no tiene errores cuando $W > \pi/N$, es decir, cuando el ancho de banda del canal es mayor que el de la señal transmitida. Cuando esta condición no se cumple, compruébese cómo el número de errores aumenta cuanto menor sea el ancho de banda del canal.

3. Datos de entrada del programa

El programa ha de aceptar, al menos, los siguientes parámetros de entrada:

1. **N**: Período de símbolo.
2. **L**: Número de períodos de símbolo utilizados para definir $h(n)$ y $p(n)$, es decir, el dominio a utilizar para definir estas señales será $n = -L * N : L * N$.
3. **K**: Número de símbolos a transmitir.
4. El vector `pulso`: vector de longitud $2LN + 1$ que contiene la forma del pulso a utilizar.
5. **W**: Ancho de banda del canal. ($W < \pi$)
6. **M**: Número de períodos de símbolo a visualizar en pantalla.
7. `EbNo`: es el cociente $\frac{E_b}{N_0}$ expresado en decibelios.

No es necesario construir una función que permita al usuario introducir los valores por teclado cuando se llame al programa, es suficiente definir las variables al principo del programa y modificarlas con el editor cuando se desee.

4. Requisitos del programa

Teniendo en cuenta la forma de generar las señales explicada en el apartado 2, se pide simular un sistema de transmisión digital binario PAM que se ajuste al modelo de la figura 1. El programa ha de ir presentando paulatinamente los siguientes resultados:

- Dibujar los M primeros símbolos del tren de deltas a transmitir.
- Dibujar la forma del pulso transmitido y su transformada de Fourier[2]
- Dibujar los M primeros símbolos de la señal transmitida y superponer sobre los mismos ejes la señal $s(n)$.

[2]Utilice la instrucción `dtft` de las prácticas anteriores

- Dibujar la respuesta en frecuencia del canal $H(\omega)$ y sobre los mismos ejes la transformada de Fourier del pulso utilizado $P(\omega)$.
- Dibujar los M primeros símbolos de la señal recibida y superponer sobre los mismos ejes la señal $s(n)$.
- Dibujar los M primeros símbolos de la salida del filtro adaptado.
- Dibujar el 'Diagrama de Ojo' correspondiente a la señal de salida del filtro adaptado.
- Dibujar los M primeros símbolos de la secuencia de las observaciones.
- Contar el número de símbolos que han sido recibidos de forma errónea.
- Estimar la probabilidad de error (es el cociente entre el número de símbolos erróneos y el número de símbolos transmitidos).
- Calcular la probabilidad de error del sistema teniendo en cuenta la $\frac{E_b}{N_0}$ que está siendo utilizada. Dado que estamos suponiendo que el sistema utiliza señalización antipodal, la probabilidad de error viene dada por la expresión

$$P_{error} = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) \tag{9}$$

donde $Q(x)$ es la función error complementario definida de la forma

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty exp\left(-\frac{u^2}{2}\right)\, du \tag{10}$$

Dado que la $\frac{E_b}{N_0}$ es un dato de entrada la probabilidad de error puede calcularse de forma inmediata utilizando la función `erfc` ya definida por MATLAB. No obstante, tenga en cuenta que MATLAB define la función `erfc(x)` de la siguiente forma

$$erfc(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty exp(-u^2)\, du \tag{11}$$

lo cual debe tenerlo en cuenta para calcular la probabilidad de error.

El eje horizontal de la presentación de señales debe estar normalizado respecto al periodo de símbolo.

5. Comentarios

- Esta práctica debe hacerse sin utilizar bucles **for** o **while**. La inclusión de alguno de estos bucles en cualquier parte del programa, exceptuando la construcción del 'Diagrama de Ojo', será motivo de anulación de la práctica.
- Comente adecuadamente el programa.
- Cualquier mejora en la entrada de los datos o en la presentación de los resultados NO será tenida en cuenta.

6. Ejemplo

En esta sección mostramos el resultado de una ejecución del programa que realiza la simulación del sistema descrito. Se consideraron pulsos de Nyquist. Los parámetros de entrada que se tomaron fueron los siguientes:

1. N=10
2. L=10
3. K=1000
4. M=10
5. W=pi/2
6. EbNo=6

Para este ejemplo el número de bits que se recibieron de forma errónea fueron 2 y por tanto, la probabilidad de error estimada es 2×10^{-3}. La probabilidad de error teorica es $2{,}4 \times 10^{-3}$

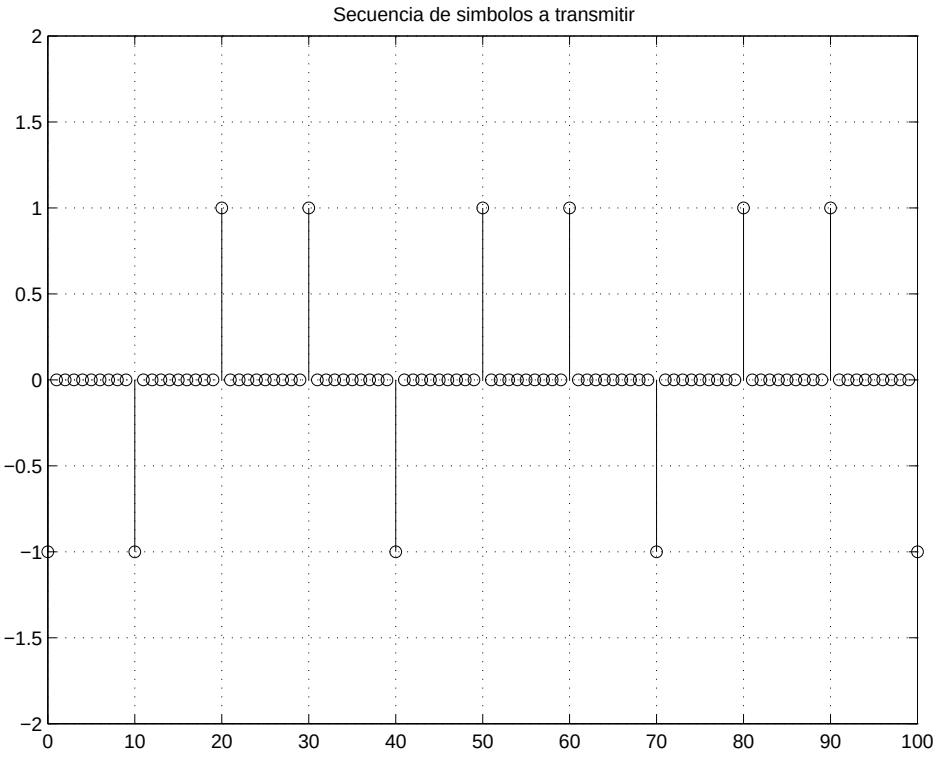

Figura 3: Tren de deltas a transmitir.

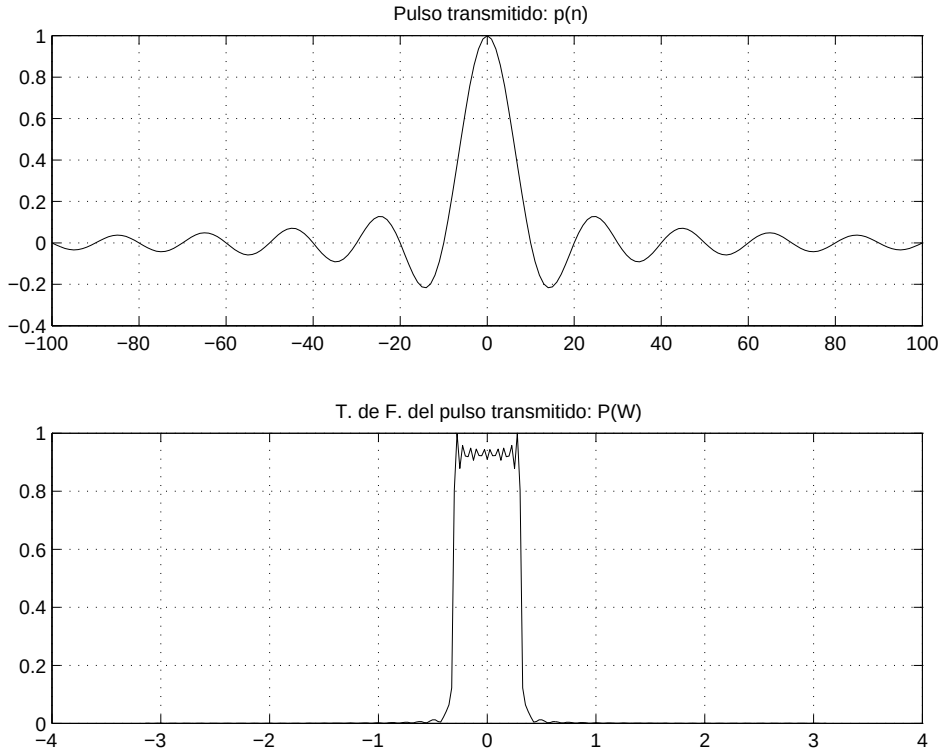

Figura 4: Pulso utilizado y su T. Fourier

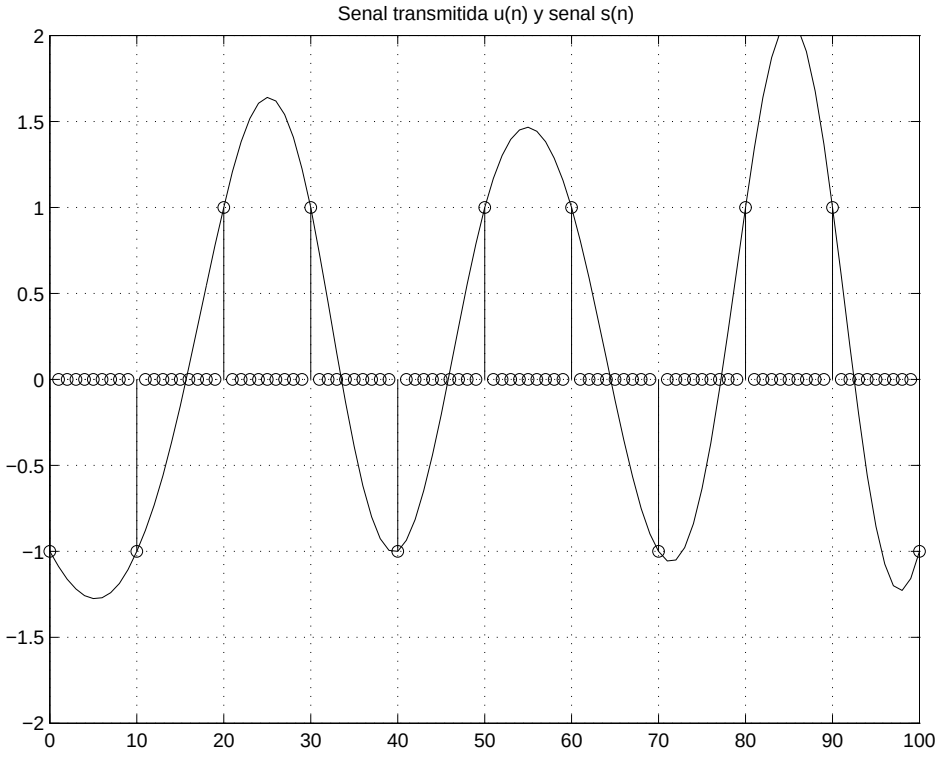

Figura 5: Señal transmitida u(n) y señal s(n)

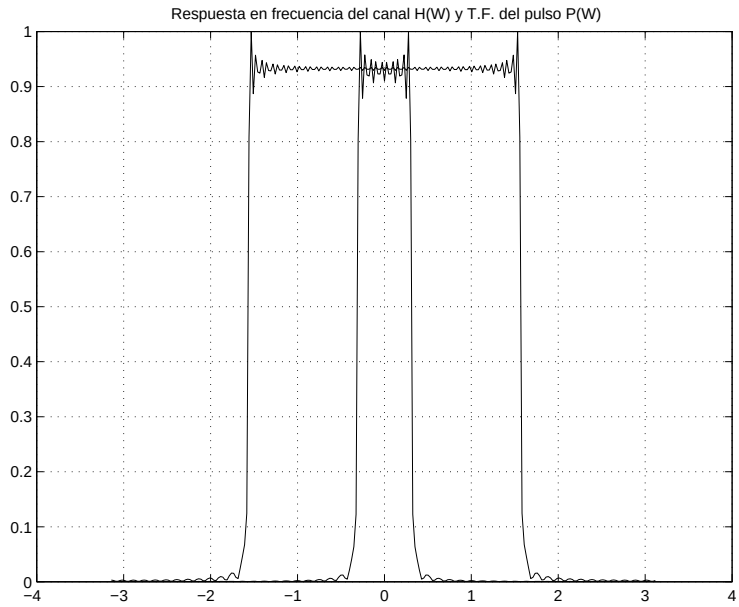

Figura 6: Respuesta en frecuencia del canal $H(\omega)$ y T.F. del pulso $P(\omega)$

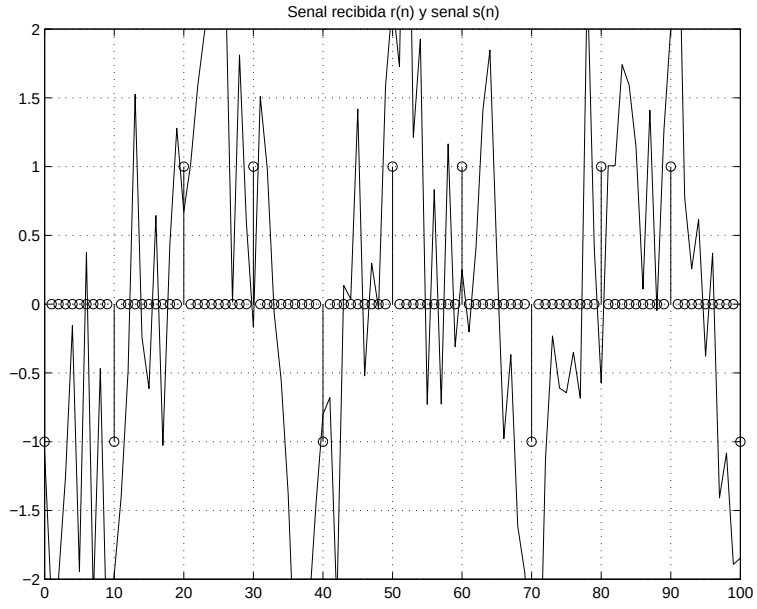

Figura 7: Señal recibida

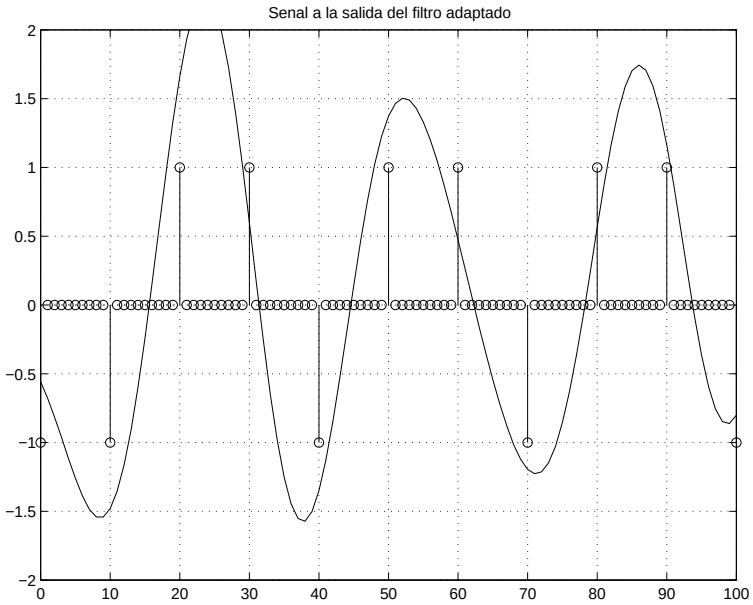

Figura 8: Señal a la salida del filtro adaptado

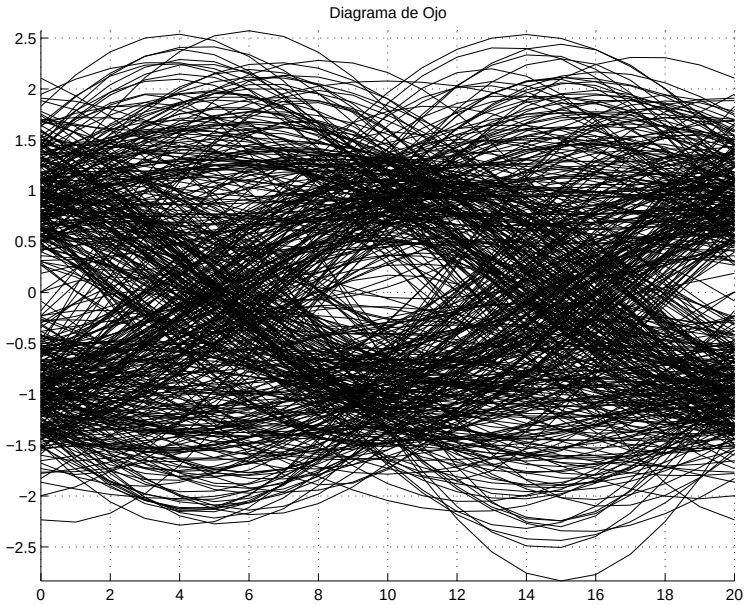

Figura 9: Diagrama de Ojo de la salida del filtro adaptado

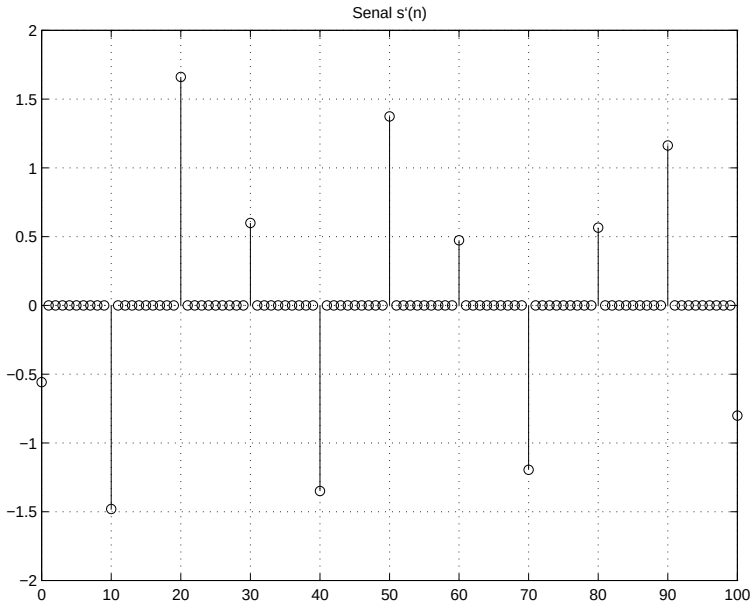

Figura 10: Observaciones