# Procesamiento Digital de Señales
# Lab. 4 - Parte I: Transformada Discreta de Fourier en MatLab

**Preparado por**

Dr. Matías Zañartu, e-mail: Matias.Zanartu@usm.cl

## I. Introducción

Este laboratorio está compuesto por dos sesiones en la cuales se estudiará la transformada discreta de Fourier (DFT), sus consideraciones, implementaciones, y aplicaciones. En esta primera parte, se trabajará con conceptos básicos de DFT y herramientas de diseño y análisis de MATLAB.

La transformada $Z$ permite analizar la respuesta de frecuencia tanto de señales como filtros. Cuando la transformada $Z$ de una señal $X(z)$ se evalúa en $X(z = e^{j\omega})$ se obtiene la transformada en tiempo discreto de Fourier (DTFT). La transformación inversa de la DTFT lleva la señal en frecuencia al dominio del tiempo discreto nuevamente.

$$Transformada \;\; \mathrm{Z} \quad X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}, \tag{1}$$

$$DTFT \quad X(e^{jw}) = \sum_{n=-\infty}^{\infty} x(n) e^{-jwn}, \tag{2}$$

$$DTFT \;\; inversa \quad x(n) = \tfrac{1}{2\pi} \textstyle\int_{-\pi}^{\pi} X(e^{jw}) e^{jwn} dw. \tag{3}$$

La DTFT es la transformación más general de Fourier para señales discretas, la cual tiene una gran utilidad analítica, pero resulta imposible de calcular en general, ya que contiene una suma infinita y la evaluación de una integral para su inversa. La transformada discreta de Fourier (DFT) es una versión muestreada de la DTFT que es factible de calcular e invertir en un computador, y se obtiene utilizando

$$DFT \quad X_N(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}, \tag{4}$$

$$DFT \;\; inversa \quad x(n) = \tfrac{1}{N} \sum_{n=0}^{N-1} X_N(k) e^{j2\pi kn/N}, \tag{5}$$

donde la representación en frecuencia es discreta (por lo cual se usa el índice $k = 0,\ldots,N-1$ en vez de $\omega$ para representar a las frecuencias, y a dichas muestras se les llama $bins$ de frecuencia), y donde la resolución espectral está dada por la cantidad de puntos ($N$) utilizados en el cálculo.

Es posible demostrar que la DFT de una señal es una versión muestreada en frecuencia (es decir calcula solo algunas frecuencias) de la DTFT de dicha señal luego que ésta es truncada por una ventana rectangular. Esto implica que, en general, la DFT es solo una aproximación de la DTFT, y que varía en su resolución y en los efectos que introduce la ventana. Tanto la DTFT como la DFT son representaciones periódicas del espectro cada $2\pi$, para frecuencias positivas y negativas.

La transformada rápida de Fourier (FFT) es una implementación muy eficiente de la DFT, que reduce su complejidad numérica de $N^2$ a $Nlog_2(N)$ (por ejemplo para $N = 1024$, es más de 100 más eficiente). Es importante tener presente que la FFT y DFT son la misma transformación. Algunas observaciones importantes sobre la DFT son:

- La DFT y su inversa se utilizan tanto para el análisis de señales como para su procesamiento, por ejemplo para realizar filtrado de señales en el dominio de la frecuencia (convolución en el tiempo equivale a multiplicación en frecuencia y viceversa), siendo este método mucho más eficiente.

- Si la DFT tiene un número de puntos espectrales $N$ menor al número de muestras de la señal $x(n)$, la DFT toma solamente sus $N$ primeros puntos, lo que genera variaciones con la DTFT por este truncamiento. Por otro lado, la transformación inversa sólo recupera los $N$ primeros puntos de la señal original.
- Si $N$ es mayor o igual al número de muestras de la señal $x(n)$, la DFT varía con respecto a la DTFT solamente en que el espectro es discreto en vez de continuo. La transformación inversa logra recuperar perfectamente la señal original $x(n)$.
- Si $N$ es igual al número de muestras, muchas líneas espectrales se ubican justo en ceros de la ventana cuadrada, lo que aparenta que ésta no existiese. Si $N$ es mayor al número de muestras se agregan ceros (zero-padding) al final de la señal para aumentar la resolución espectral.
- Una ventana rectangular puede generar cortes abruptos en la selección de señal y observaciones incompletas de la periodicidad de la señal, lo que produce distorsiones armónicas. Esto se corrige utilizando otras ventanas, las que deben ser suficientemente largas.
- La DFT se obtiene entre $[0, 2\pi]$ y la DTFT entre $[-\pi, \pi]$. En MATLAB, para poder desplazar los resultados de la DFT y ser mostrados como en la DTFT, se utiliza en comando $fftshift$.
- En MATLAB, la magnitud de la DFT debe ser normalizada por N para obtener amplitudes adecuadas. Si $N$ es mayor al largo de la señal, se normaliza por $min(N, length(x))$. Si se usa otra ventana, la normalización es por la suma de los valores de la ventana, en vez de $N$.

## II. Cálculo de la DFT para Señales Simples

Considere las señales $x_1 = \sin(\omega t)$ y $x_2 = \cos(\omega t)$, ambas con frecuencia de 100 Hz y muestreadas con $f_s = 5$ kHz durante 100 ms (500 muestras exactas).

1) Obtenga una expresión analítica para la DTFT de las señales suponiendo que tienen una duración infinita (recuerde que $e^{j\theta} = \cos(\theta) + j\sin(\theta)$). Obtenga la misma DTFT asumiendo la duración real de las señales. Grafique la magnitud del espectro (en dB) para cada caso. (3 Ptos.)
2) Obtenga la DFT de cada señal utilizando el comando MATLAB $fft$ con $N = 4096$. Muestre los resultados de magnitud entre a) $[-f_s/2, f_s/2]$ y b) $[0, f_s]$. Asegúrese que la amplitud de la señal sea la esperada. ¿Cómo se compara esta representación con lo esperado para una señal?. (3 ptos.)
3) Obtenga la DFT de cada señal para diversos valores de $N = 256$, $N = 500$, $N = 2048$. Muestre los siguientes gráficos: a) Magnitud v/s Frecuencia, y b) Parte Real y Parte Imaginaria v/s Frecuencia. Muestre los resultados de magnitud entre $[-\pi, \pi]$. ¿Qué valor de $N$ entrega resultados más cercanos a la DTFT de las señales muestreadas? ¿Qué sucede cuando $N$ es igual al largo de la señal? ¿Por qué las amplitudes no son exactas en algunos casos?. (4 ptos.)

## III. Estimación de Frecuencia en Presencia de Ruido

Genere una señal compuesta por dos tonos puros de 100 Hz y 500 Hz con amplitudes de 0.5 y 1.5, respectivamente. Sume a esta señal un ruido blanco con distribución normal (media 0, variancia 2). Utilice una frecuencia de muestreo adecuada y una duración de 100 ms. Grafique la señal resultante. ¿Es posible determinar la señal y sus frecuencias del gráfico temporal de la señal con ruido? Grafique la magnitud del espectro de la señal con ruido y compárela en el mismo gráfico con la señal limpia. Presente sus resultados entre [0, 1000] Hz y mostrando toda la energía de la señal en ese intervalo. ¿Por qué la amplitud de los tonos no es exactamente igual en estos casos? Genere el mismo gráfico pero con una escala de amplitud en decibeles ($dB = 20log_{10}(|X|)$). Normalice la magnitud de modo que la amplitud máxima sea 0 dB. ¿Qué diferencia de amplitud existe entre la componente de señal más elevada y el ruido de fondo? Aumente la duración de la señal a 1 s y repita este último punto. ¿A qué se debe esta variación?. (10 ptos.)

## IV. Filtrado de Señales en el Dominio de la Frecuencia

Utilice el archivo $nspeech.mat$ el cual contiene una señal de voz junto con la presencia no deseada de un tono puro. Obtenga la DFT de la señal e identifique la frecuencia del tono no deseado. Grafique la magnitud de la señal en el rango [0, 4000] Hz y con amplitud en dB. Diseñe un filtro que tenga dos ceros conjugados sobre el círculo unitario a dicha frecuencia (recuerde que este filtro se reduce a la expresión $H(\omega) = 1 - 2\cos(\theta)e^{-j\omega} + e^{-2j\omega}$). Grafique la magnitud de dicho filtro en MATLAB. Obtenga el parámetro $\theta$ y realice el filtrado mediante la multiplicación

punto a punto entre la DFT de la señal y $H(\omega)$. Grafique la magnitud de la señal resultante y compare. Mediante la DFT inversa, obtenga la señal en el dominio del tiempo. Para asegurarse que la señal resultante sea real, utilice la opción $symmetric$ en la DFT inversa. ¿Qué valor de $N$ se requiere para reconstruir la señal completa? Grafique la señal original y la filtrada. Escuche y comente la señal resultante. ¿Se elimina completamente el tono molesto?. (10 ptos.)

## V. Consideraciones de Implementación de la DFT

En esta sección de laboratorio se estudiarán las consideraciones de implementaci0́n de la transformada discreta de Fourier (DFT) y el algoritmo de la transformada rápida de Fourier (FFT).

Considere la DFT de $N$ puntos $X_N(k)$, la cual llamaremos también $X^{(N)}(k)$ por notación y donde $k = 0, \ldots, N-1$:

$$X_N(k) = X^{(N)}(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}. \tag{6}$$

La DFT se puede calcular directamente con la expresión de (6) como una suma utilizando $for - loops$ para cada $bins$ de frecuencia, o también en una representación matricial para los N bins simultáneamente. Aunque la complejidad numérica de ambas formas es la misma ($N^2$), este último método presenta ventajas significativas producto del trabajo en matrices. Estas dos formas directas de calcular la DFT se estudiarán en este laboratorio. El algoritmo FFT entrega los mismos resultados que (6), pero optimiza el cómputo dividiendo el problema en cálculos de DFT de menor orden y una estructura recursiva. Para derivar una de las versiones más comunes del algoritmo FFT llamado $Radix - 2$, asumimos que $N = 2^m$. De esta forma, podemos partir por dividir el cálculo para las muestras pares e impares.

$$X^{(N)}(k) = \sum_{n=0(n\ par)}^{N-1} x(n) e^{-j2\pi kn/N} + \sum_{n=0(n\ impar)}^{N-1} x(n) e^{-j2\pi kn/N}. \tag{7}$$

$$X^{(N)}(k) = \sum_{m=0}^{N/2-1} x(2m) e^{-j2\pi k2m/N} + \sum_{m=0}^{N/2-1} x(2m+1) e^{-j2\pi k(2m+1)/N}, \tag{8}$$

$$X^{(N)}(k) = \sum_{m=0}^{N/2-1} x(2m) e^{-j2\pi km/(N/2)} + e^{-j2\pi k/N} \sum_{m=0}^{N/2-1} x(2m+1) e^{-j2\pi km/(N/2)}. \tag{9}$$

Si llamamos a la serie de muestras par $x_0 = x(2m)$ y a la impar $x_1 = x(2m+1)$ entonces

$$X^{(N)}(k) = \sum_{m=0}^{N/2-1} x_0(m) e^{-j2\pi km/(N/2)} + e^{-j2\pi k/N} \sum_{m=0}^{N/2-1} x_1(m) e^{-j2\pi km/(N/2)}, \tag{10}$$

$$X^{(N)}(k) = X_0^{(N/2)}(k) + e^{-j2\pi k/N} X_1^{(N/2)}(k). \tag{11}$$

Esto indica que una DFT de $N$ puntos se puede obtener como la suma de dos DFT de $N/2$ puntos con un factor multiplicador especial. Aun así, es posible simplificar esta operación aun más observando las simetrías del sistema. Si llamamos a los coeficientes $W_N = e^{-j2\pi/N}$ entonces:

$$X^{(N)}(k) = \sum_{n=0}^{N-1} x(n) W_N^k = X_0^{(N/2)}(k) + W_N^k X_1^{(N/2)}(k). \tag{12}$$

Notando además que cada DFT de $N/2$ puntos es periódica cada $N/2$ puntos y que $W_N^{k+N/2} = -W_N^k$, se puede obtener que:

$$X^{(N)}(k) = X_0^{(N/2)}(k) + W_N^k X_1^{(N/2)}(k), \tag{13}$$

$$X^{(N)}(k + N/2) = X_0^{(N/2)}(k) - W_N^k X_1^{(N/2)}(k), \tag{14}$$

para $k = 0, \ldots, N/2 - 1$. Esta transformación reduce la complejidad numérica de la DFT de $N^2$ a $N^2/2 + N$. La representación gráfica de (13) y (14) se denota en la Fig. 1.

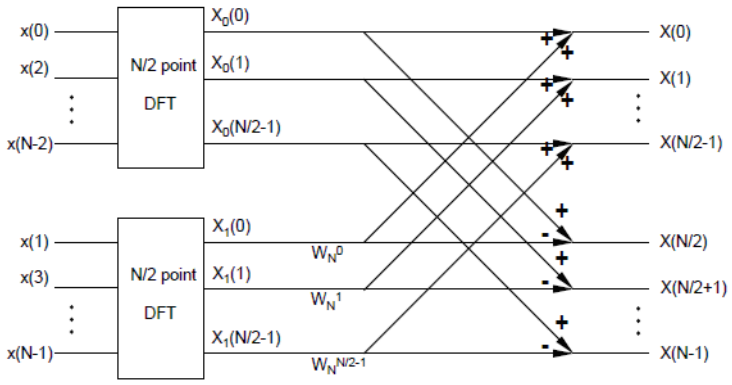


Figure 1. Cálculo de la DFT usando reducción de orden. La DFT de $N$ puntos ($X(k)$) se obtiene mediante dos $N/2$ DFTs ($X_0(k)$) y ($X_1(k)$).

Esta división en DFT de $N$ puntos a dos de $N/2$ se puede volver a realizar recursivamente en cada bloque de la Fig. 1. Para ilustrar este punto, la reducción de orden se vuelve a aplicar sobre cada DFT de $N/2$ puntos en la Fig. 2. Es posible seguir esta reducción hasta llegar al bloque más pequeño posible de una DFT de dos puntos, a cual se obtiene directamente de la ecuación (6).

$$X^{(2)}(0) = x(0) + x(1), \tag{15}$$

$$X^{(2)}(1) = x(0) - x(1). \tag{16}$$

Esta implementación se conoce como FFT $radix - 2$ y es una de las formas más comunes. La Fig. 2 presenta el esquema de dicha estructura.

## VI. Cálculo Directo de la DFT

Cree una función en MATLAB llamada $X = DFTsum(x)$ para obtener la DFT de $N$ puntos, donde la señal de entrada $x(n)$ es un vector de $N$ puntos. Su rutina debe calcular la DFT exactamente como se define en la ecuación (6) utilizando $for - loops$ y calculando los exponenciales directamente. Tenga cuidado de no usar $i$ y $j$ en sus contadores para evitar confusión con la variable compleja $j = \sqrt{-1}$. Incluya el código MATLAB de cada implementación.

1) Pruebe su implementación con las siguientes señales, todas las cuales tienen una duración $N = 8$: a) $x(n) = \delta(n)$, b) $x(n) = 1$, c) $x(n) = e^{-j2\pi n/N}$, y d) $x(n) = \cos(2\pi n/N)$. Grafique la magnitud de cada DFT entre $[-\pi, \pi]$. Compare los resultados anteriores con los obtenidos utilizando el comando $fft$ de MATLAB. (12 Ptos.)
2) Obtenga expresiones analíticas para cada DFT de las señales anteriores y compare sus resultados con estas expresiones. Considere que la DTFT de una ventana cuadrada de largo $N$ es: (8 Ptos.)

$$W(e^{jw}) = e^{-jw(N-1)/2}\frac{\sin(wN/2)}{\sin(w/2)}. \tag{17}$$

3) Considere una señal $x_1 = \cos(wt)$ con frecuencia de 100 Hz, muestreada a 5 kHz durante 1000 ms de modo de obtener 5000 muestras exactas. Compare la magnitud resultante de la DFT con la función DFTsum con $fft$ de MATLAB para los 500 primeros puntos de la señal. Grafique solo las diferencias (errores entre ambos métodos). Utilizando los comandos $tic$ y $toc$ de MATLAB, calcule el tiempo de procesamiento para cada función. ¿Como varía la razón entre estos dos tiempos a medida que la DFT considera más puntos?. Grafique el tiempo de procesamiento en función de N para cada caso (6 Ptos.)

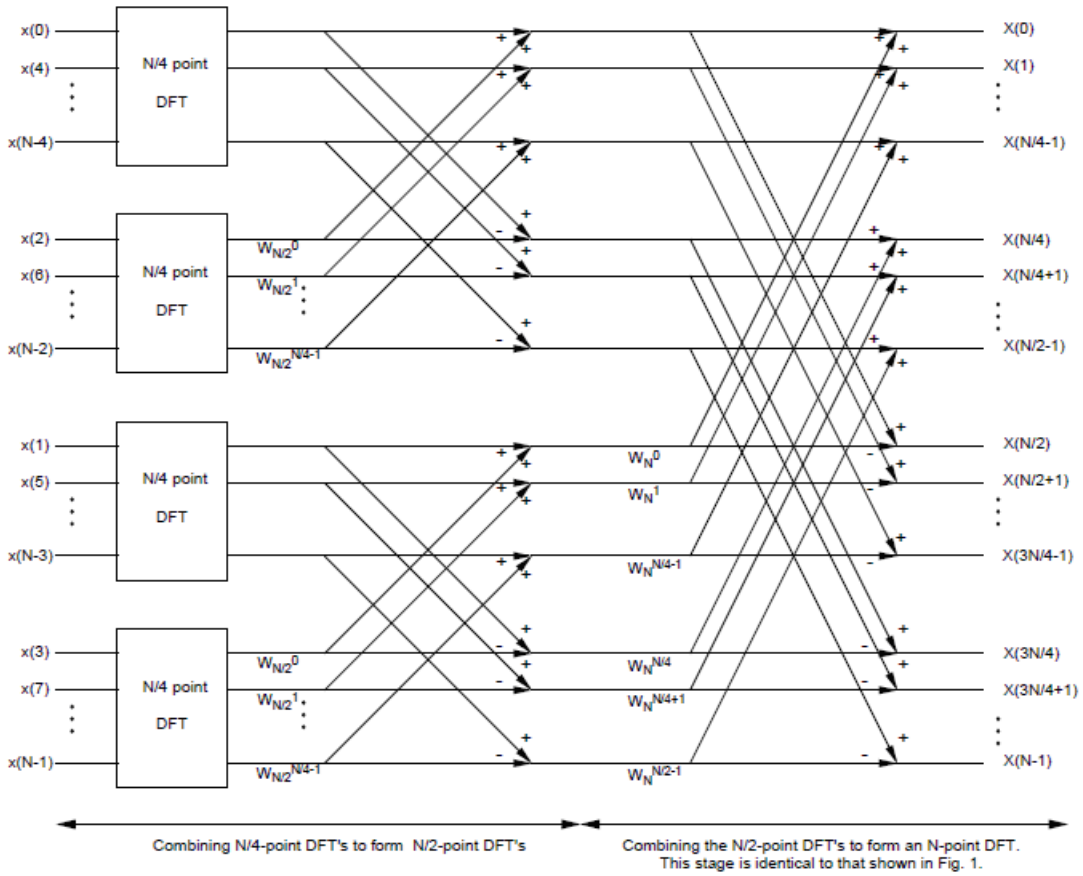


Figure 2. Cálculo de la DFT usando reducción de orden. La DFT de $N$ puntos ($X(k)$) se obtiene mediante cuatro $N/4$ DFTs.

## VII. Cálculo Matricial de la DFT

Es posible representar la ecuación (6) como un producto entre una matriz y un vector, de modo que $X = Ax$, donde $X$ y $x$ son vectores columna de $N \times 1$ y $A$ es una matriz cuadrada de $N \times 1N$. Este producto es equivalente a $X_k = \sum_{n=1}^{N} A_{kn} x_n$, donde $A_{kn}$ es el elemento $(k, n)$ de la matriz dado por $A_{kn} = e^{-j2\pi(k-1)(n-1)/N}$. Note que en MATLAB los índices se inician en 1 y no en 0, lo que explica el corrimiento en -1 de $A_{kn}$. Cree una función en MATLAB llamada $X = DFTmatrix(x)$ para obtener la DFT de $N$ puntos, donde la señal de entrada $x(n)$ es un vector de N puntos. Su rutina debe calcular la DFT como se explica en esta sección, sin utilizar $for - loops$ en el producto entre $x$ y $A$. Para abordar el problema se solicita lo siguiente:

1) Cree una función para obtener la matriz A utilizando $for - loops$ y compare sus resultados con el comando $dftmtx$ de MATLAB. Compare sus resultados para $N$ = 2, 4, y 8. (5 Ptos.)
2) Grafique la parte imaginaria y la parte real de la matriz A utilizando el comando $imagesc$ para $N$ =4, 32, y 256. ¿Qué simetrías se observan en la matriz A?. (5 Ptos.)
3) Pruebe nuevamente su implementación con las señales del punto VI.1, y confirme que los resultados son los mismos. Calcule el error cuadrático medio entre el comando $fft(x)$ y los métodos implementados: $DFTmatrix(x)$, $DFTsum(x)$. (6 Ptos.)
4) Utilizando la misma señal que en el problema VI.3, calcule el tiempo de procesamiento para los métodos $fft$, $DFTsum$ y $dftmtx$ a medida que aumenta el número de N. ¿Qué método es mas eficiente?. ¿Qué método utiliza mas recursos de memoria? (6 ptos.)

## VIII. Algoritmo Radix-2 de la FFT

El algoritmo Cooley-Turkey o Radix-2, es uno de los métodos más comunes para implementar la transformada rápida de Fourier (FFT). Este método se basa en expresar una DFT de $N = N_1 N_2$ puntos, en términos de DFTs

de tamaños $N_1$ y $N_2$ de forma recursiva. Esto se realiza para reducir el tiempo de cálculo de la DFT. La estructura Radix-2 para una FFT de 2 puntos se presenta en la Fig. 3 (ver archivo de ayuda: Cooley and Tukey (1965) - An algorithm for the machine calculation of complex Fourier series).

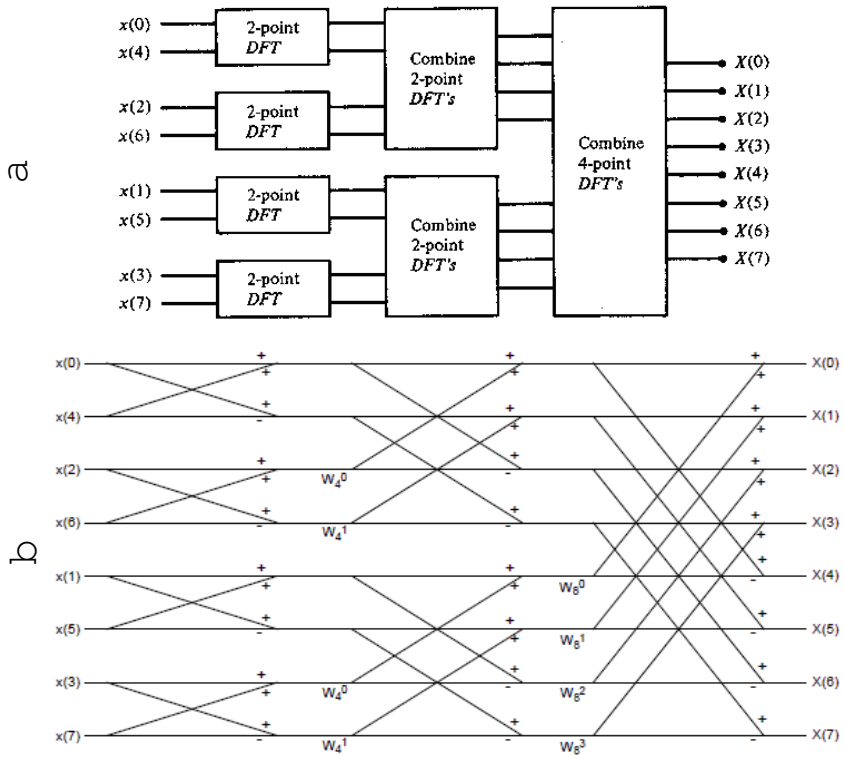


Figure 3. Cálculo de una FFT de 8 puntos usando el esquema Radix-2. (a) Esquema general; (b) Operaciones aritméticas involucradas.

1) Escriba una función $X = DFTdc(x)$ para implementar una reducción de orden en la implementación de la DFT. Separe la señal en las muestras pares e impares y utilice su función $DFTsum$ para calcular las dos DFT de $N/2$ puntos. Evalúe la DFT obtenida con $fft8$ para las señales: a) $x(n) = \delta(n)$, b) $x(n) = 1$, c) $x(n) = e^{-j2\pi n/8}$, todas con un largo $N = 8$. Grafique magnitud, calcule el tiempo de procesamiento de esta función y compárela con $fft(x)$. (5 ptos.)
2) Escriba tres funciones para calcular una $fft$ de 2, 4 y 8 puntos usando los nombres $X = FFT2(x)$, $X = FFT4(x)$, y $X = FFT8(x)$. La $FFT2$ calcula la $fft$ de 2 puntos utilizando las ecuaciones (15) y (16), mientras que las $FFT4$ y $FFT8$ utilizan las ecuaciones (13) y (14) de forma recursiva, es decir $FFT8$ llama a $FFT4$ y ésta llama a $FFT2$. Evalúe la DFT obtenida con $FFT8$ para las señales del punto VIII.1. Grafique magnitud y compare los tiempos de procesamiento con $fft(x)$. (5 ptos.)
3) Escriba una función $X = fft\_stage(x)$ que calcule la $fft$ de cualquier señal cuyo largo sea una potencia de dos, utilizando la misma idea que en el punto VIII.2, pero de forma recursiva (la función se llama a si misma). Si $N = 2$ la función calcula $FFT2$ y si $N > 2$ la estructura de la función debiese ser muy similar a $FFT8$, pero de forma recursiva. Compare resultados con su $FFT8$ para los casos anteriores. Compare sus resultados y con la función $fft$ de MATLAB de igual forma que en el punto VII.4. Comente sus resultados. (6 ptos.)

**Informe de Laboratorio**:
Para todos los puntos de este laboratorio: Presente las funciones de MATLAB requeridas, junto con sus cálculos, ecuaciones y gráficos. Comente sus observaciones y etiquete sus gráficos adecuadamente.