Instituto Politécnico Nacional

Escuela Superior De Cómputo

Entregable 10

Filtinic/Filter

Nombre de los integrantes:

Hernandez Rodriguez Juan Uriel

Vergara Martinez Brenda

García Quiroz Gustavo Ivan

Gutiérrez Jiménez Cinthia Nayelli

Ramírez Carrillo José Emilio

Iturbide Serrano Uriel

Grupo: 5CV1

Procesamiento Digital de Señales

Nombre del profesor: José Antonio Flores Escobar

Índice de Contenidos

Introducción

En el campo del procesamiento digital de señales, la Transformada de Fourier Discreta (DFT) desempeña un papel fundamental al permitir el análisis de señales en el dominio de la frecuencia. Esta herramienta matemática convierte una secuencia de muestras de tiempo discreto en una secuencia de valores complejos, representando la amplitud y fase de las componentes de frecuencia que componen la señal original.

La presente práctica tiene como objetivo principal el cálculo de la DFT de una señal dada, utilizando diferentes métodos y herramientas computacionales. A lo largo de este ejercicio práctico, se explorará el marco teórico que sustenta la DFT, incluyendo su definición, propiedades y la forma unitaria de la misma. Además, se abordará el desarrollo detallado del cálculo de la DFT, empleando técnicas numéricas y recursos de programación.

Al finalizar esta práctica, se habrá adquirido una comprensión profunda de la DFT y su aplicación en el análisis de señales digitales, lo que resulta fundamental en diversas áreas de la ingeniería, como el procesamiento de audio, imágenes y señales biomédicas, entre otras. Asimismo, se habrán desarrollado habilidades prácticas en el uso de herramientas computacionales para el cálculo eficiente de la DFT, preparando al estudiante para su aplicación en problemas reales.

Marco teórico

Transformada de Fourier discreta

La **transformada discreta de Fourier** o **DFT** (Discrete Fourier Transform) es un tipo de transformada discreta utilizada en el análisis de Fourier. Transforma una función matemática en otra, obteniendo una representación en el dominio de la frecuencia, siendo la función original una función en el dominio del tiempo. Pero la DFT requiere que la función de entrada sea una secuencia discreta y de duración finita. Dichas secuencias se suelen generar a partir del muestreo de una función continua, como puede ser la voz humana. Al contrario que la transformada de Fourier en tiempo discreto (DTFT), esta transformación únicamente evalúa suficientes componentes frecuenciales para reconstruir el segmento finito que se analiza. Utilizar la DFT implica que el segmento que se analiza es un único período de una señal periódica que se extiende de forma infinita; si esto no se cumple, se debe utilizar una ventana para reducir los espurios del espectro. Por la misma razón, la DFT inversa (IDFT) no puede reproducir el dominio del tiempo completo, a no ser que la entrada sea periódica indefinidamente. Por estas razones, se dice que la DFT es una transformada de Fourier para análisis de señales de tiempo discreto y dominio finito. Las funciones sinusoidales base que surgen de la descomposición tienen las mismas propiedades.

Los algoritmos FFT se utilizan tan habitualmente para calcular DFTs que el término FFT muchas veces se utiliza en lugar de DFT en lenguaje coloquial. Formalmente, hay una diferencia clara: DFT hace alusión a una transformación o función matemática, independientemente de cómo se calcule, mientras que FFT se refiere a una familia específica de algoritmos para calcular DFTs.

Definición

La secuencia de N números complejos $x_0, ..., x_{N-1}$ se transforma en la secuencia de N números complejos $X_0, ..., X_{N-1}$ mediante la DFT con la fórmula:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \dots, N-1$$

donde i es la unidad imaginaria y $e^{-\frac{2\pi i}{N}}$ es la N-ésima raíz de la unidad. (Esta expresión se puede escribir también en términos de una matriz DFT; cuando se escala de forma apropiada se convierte en una matriz unitaria y X_k puede entonces ser interpretado como los coeficientes de x en una base ortonormal.)

La transformada se denota a veces por el símbolo F, igual que en $\mathbf{X} = \mathcal{F}\{\mathbf{x}\}$ o $\mathcal{F}(\mathbf{x})$ o $\mathcal{F}\mathbf{x}$.

La transformada inversa de Fourier discreta (IDFT) viene dada por

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{\frac{2\pi i}{N}kn} \qquad n = 0,\dots,N-1.$$

Una descripción simple de estas ecuaciones es que los números complejos X_k representan la amplitud y fase de diferentes componentes sinusoidales de la señal de entrada x_n. La DFT calcula X_k a partir de x_n, mientras que la IDFT muestra cómo calcular x_n como la suma de componentes sinusoidales $\left(\frac{1}{N}\right)X_k e^{\frac{2\pi i}{N}kn}$ con una frecuencia de $\frac{k}{N}$ ciclos por muestra.

Propiedades

Tabla 2: PROPERTIES OF THE DTFT

Propiedad	Secuencia	Transformada de Fourier
	$x[n]$	$X(\Omega)$
	$x_1[n]$	$X_1(\Omega)$
	$x_2[n]$	$X_2(\Omega)$
Periodicidad	$x[n]$	$X(\Omega + 2\pi) = X(\Omega)$
Linealidad	$a_1x_1[n] + a_2x_2[n]$	$a_1X_1(\Omega) + a_2X_2(\Omega)$
Desplazamiento en el tiempo	$x[n - n_0]$	$e^{-j\Omega n_0}X(\Omega)$
Desplazamiento en la frecuencia	$e^{j\Omega_0 n}x[n]$	$X(\Omega - \Omega_0)$
Conjugación	$x^*[n]$	$X^*(-\Omega)$
Inversión del tiempo	$x[-n]$	$X(-\Omega)$
Escalamiento en el tiempo	$x_{(m)}[n] = \begin{cases} x[n/m] & \text{si } n = km \\ 0 & \text{si } n \neq km \end{cases}$	$X(m\Omega)$
Diferenciación en la frecuencia	$nx[n]$	$j\dfrac{dX(\Omega)}{d\Omega}$

La DFT unitaria

Otra forma de interpretar la DFT es dándose cuenta de que puede expresarse como una matriz de Vandermonde:

$$\mathbf{F} = \begin{bmatrix} \omega_N^{0\cdot 0} & \omega_N^{0\cdot 1} & \ldots & \omega_N^{0\cdot (N-1)} \\ \omega_N^{1\cdot 0} & \omega_N^{1\cdot 1} & \ldots & \omega_N^{1\cdot (N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_N^{(N-1)\cdot 0} & \omega_N^{(N-1)\cdot 1} & \ldots & \omega_N^{(N-1)\cdot (N-1)} \end{bmatrix}$$

donde

$$\omega_N = e^{-2\pi i/N}$$

es una raíz de la unidad. La transformada inversa viene entonces dada por la inversa de la matriz anterior:

$$F^{-1} = \frac{1}{N} F^*$$

Desarrollo

En esta práctica, se implementaron dos códigos en MATLAB para analizar el comportamiento de sistemas de filtrado digital. El primer código se enfoca en la implementación manual de un filtro digital definido por una ecuación en diferencias, mientras que el segundo utiliza las funciones incorporadas de MATLAB para el filtrado.

En el primer código, se define un filtro digital con la ecuación en diferencias y(n) = 2x(n) - 4x(n-1) - 0.5y(n-1) - y(n-2). Se establecen condiciones iniciales específicas y se utiliza una señal de entrada x(n) = 0.8^n u(n). El código calcula la respuesta del sistema para 20 muestras utilizando un bucle for que aplica la ecuación en diferencias. Se generan tres gráficas: la señal de entrada, la respuesta del sistema con las condiciones iniciales dadas, y la respuesta del sistema con diferentes condiciones iniciales.

Filtro digital con condiciones iniciales

```matlab
% Entregable    10 – Filtinic/Filter
% Grupo         5CV1
% Equipo        7
% Profesor: José Antonio Flores Escobar
% Integrantes:
%               Hernandez Rodriguez Juan Uriel
%               Vergara Martinez Brenda
%               García Quiroz Gustavo Ivan
%               Gutiérrez Jiménez Cinthia Nayelli
%               Ramírez Carrillo José Emilio
%               Iturbide Serrano Uriel

% Definir la ecuación en diferencias del filtro digital
% y(n) = 2x(n) - 4x(n-1) - 0.5y(n-1) - y(n-2)

% Condiciones iniciales
%     y(-2) = 1
%     y(-1) = 0
%     x(-1) = -1
%     Señal de entrada x(n) = 0.8^n u(n)

% Calcular la respuesta del sistema para 20 muestras usando MATLAB

y = zeros(1,20); % Inicializar vector de salida
y = [1 0 -y]; % Establecer condiciones iniciales de y(n)

n = 0:1:19; % Vector de índices de tiempo

x = 0.8.^n; % Calcular señal de entrada
x = [0 -1 x]; % Establecer condición inicial de x(n)
```

```
for n=1:20
    y(n+2) = 2*x(n+2) - 4*x(n+1) - 0.5*y(n+1) - y(n); % Aplicar ecuación er
end

n = 0:1:19; % Vector de índices de tiempo

% Graficar señal de entrada x(n)
subplot(3, 1, 1);
stem(n, x(3:22));
title('Señal de entrada x(n)');
xlabel('Índice de tiempo');
ylabel('Amplitud');

% Graficar respuesta del sistema y(n)
subplot(3, 1, 2);
stem(n, y(3:22));
title('Respuesta del sistema y(n)');
xlabel('Índice de tiempo');
ylabel('Amplitud');

y = zeros(1,20); % Inicializar vector de salida
y = [0 0 y]; % Establecer condiciones iniciales de y(n)

n = 0:1:19; % Vector de índices de tiempo

x = 0.8.^n; % Calcular señal de entrada
x = [0 0 x]; % Establecer condiciones iniciales de x(n)

for n=1:20
    y(n+2) = 2*x(n+2) - 4*x(n+1) - 0.5*y(n+1) - y(n); % Aplicar ecuación er
end

n = 0:1:19; % Vector de índices de tiempo

% Graficar respuesta del sistema y(n) con diferentes condiciones iniciales
subplot(3, 1, 3);
stem(n, y(3:22));
title('Respuesta del sistema y(n) con diferentes condiciones iniciales');
xlabel('Índice de tiempo');
ylabel('Amplitud');
```

Figura 1 Filtro digital con condiciones iniciales

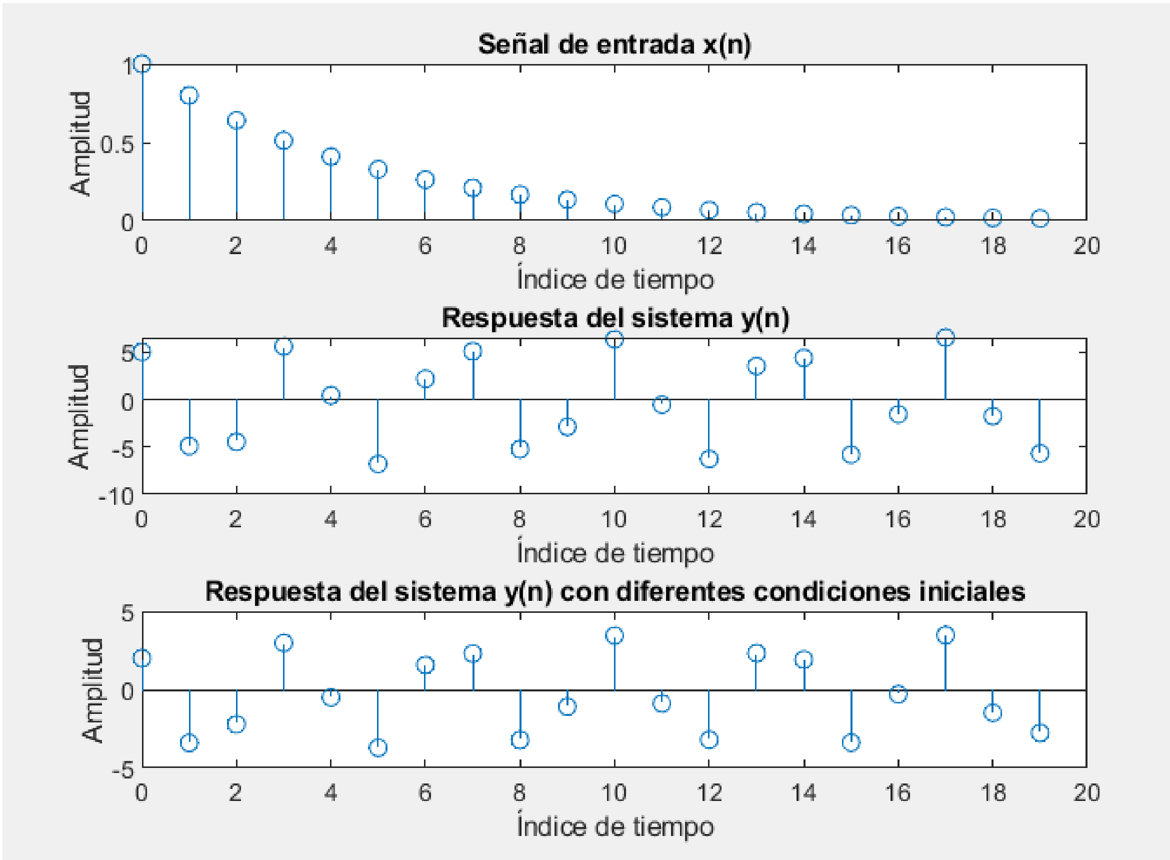

Figura 2 Simulación Filtro digital con condiciones iniciales

Filtro digital con Filtinic/Filter

El segundo código implementa un filtro digital más simple, definido por la ecuación y(n) = 0.5y(n-2) + x(n-1). En este caso, se utilizan las funciones incorporadas de MATLAB 'filtic' y 'filter' para calcular la salida del sistema. Se definen los coeficientes del filtro, la señal de entrada, y las condiciones iniciales. La función 'filtic' se usa para calcular las condiciones iniciales del filtro, mientras que 'filter' se emplea para aplicar el filtro a la señal de entrada.

Ambos códigos demuestran diferentes enfoques para implementar y analizar filtros digitales, proporcionando una comprensión práctica de cómo las condiciones iniciales y las ecuaciones en diferencias afectan la respuesta del sistema.

```
% Entregable    10 - Filtinic/Filter
% Grupo         5CV1
% Equipo        7
% Profesor: José Antonio Flores Escobar
% Integrantes:
%               Hernandez Rodriguez Juan Uriel
%               Vergara Martinez Brenda
%               García Quiroz Gustavo Ivan
%               Gutiérrez Jiménez Cinthia Nayelli
%               Ramírez Carrillo José Emilio
%               Iturbide Serrano Uriel

% Calcular la salida del sistema
%y(n) = 0.5y(n-2) + x(n-1)
%Para las primeras 4 muestras se usaran las siguientes condiciones inicial
%   y(-2) = 1
%   y(-1) = 0
%   x(-1) = -1
%   entrada x(n) = (0.5)^n u(n)

B = [0 1];
A = [1 0 -0.5];
x = [1 0.5 0.25 0.125];
Xi = [-1 0];
Yi = [0 1];
Zi = filtic(B, A, Yi, Xi);
y = filter(B, A, x, Zi);
```

Figura 3 Filtro digital con Filtinic/Filter

Simulación

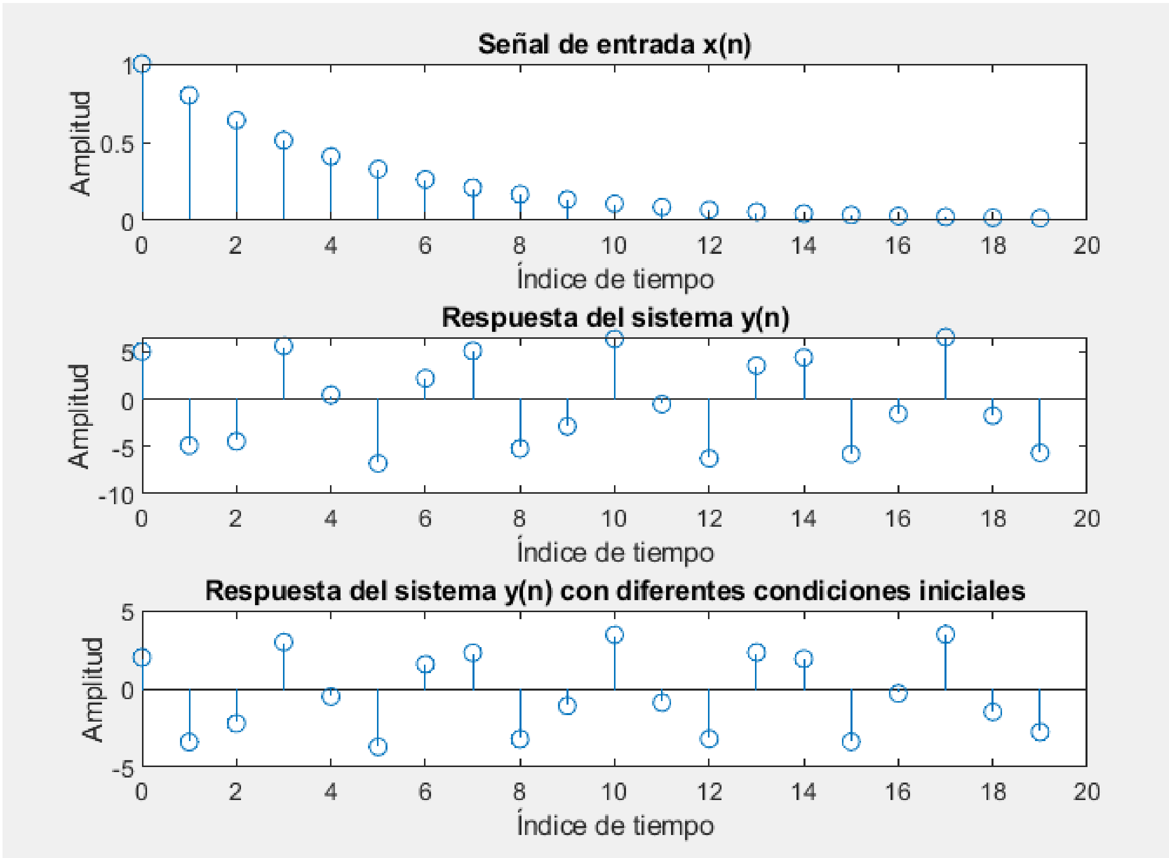

Figura 4 Simulación Filtro digital con condiciones iniciales

Conclusión

Esta práctica nos permitió profundizar en la implementación y análisis de filtros digitales utilizando MATLAB. A través de dos enfoques distintos uno manual y otro utilizando funciones incorporadas pudimos observar cómo las condiciones iniciales y las ecuaciones en diferencias afectan la respuesta de los sistemas de filtrado. El primer método nos brindó una comprensión detallada del proceso de cálculo, mientras que el segundo demostró la eficiencia de las herramientas integradas de MATLAB. Ambos enfoques resaltaron la importancia de entender los fundamentos teóricos detrás de los filtros digitales y cómo se traducen en implementaciones prácticas.

Referencias

[1] L. Tan, *Digital Signal Processing*. Academic Press, 2008.

[2] Wikipedia contributors, “Transformada de Fourier discreta”, *Wikipedia, The Free Encyclopedia*. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Transformada_de_Fourier_discreta&oldid=148495486.

Anexo. Código.

Código Filtinic/Filter

```
% Entregable    10 – Filtinic/Filter
% Grupo         5CV1
% Equipo        7
% Profesor: José Antonio Flores Escobar
% Integrantes:
%              Hernandez Rodriguez Juan Uriel
%              Vergara Martinez Brenda
%              García Quiroz Gustavo Ivan
%              Gutiérrez Jiménez Cinthia Nayelli
%              Ramírez Carrillo José Emilio
%              Iturbide Serrano Uriel

% Definir la ecuación en diferencias del filtro digital
% y(n) = 2x(n) - 4x(n-1) - 0.5y(n-1) - y(n-2)

% Condiciones iniciales
%     y(-2) = 1
%     y(-1) = 0
%     x(-1) = -1
%     Señal de entrada x(n) = 0.8^n u(n)

% Calcular la respuesta del sistema para 20 muestras usando MATLAB

y = zeros(1,20); % Inicializar vector de salida
y = [1 0 -y]; % Establecer condiciones iniciales de y(n)

n = 0:1:19; % Vector de índices de tiempo

x = 0.8.^n; % Calcular señal de entrada
x = [0 -1 x]; % Establecer condición inicial de x(n)

for n=1:20
    y(n+2) = 2*x(n+2) - 4*x(n+1) - 0.5*y(n+1) - y(n); % Aplicar ecuación en
diferencias
end

n = 0:1:19; % Vector de índices de tiempo

% Graficar señal de entrada x(n)
subplot(3, 1, 1);
stem(n, x(3:22));
title('Señal de entrada x(n)');
xlabel('Índice de tiempo');
ylabel('Amplitud');

% Graficar respuesta del sistema y(n)
subplot(3, 1, 2);
stem(n, y(3:22));
title('Respuesta del sistema y(n)');
```

```
xlabel('Índice de tiempo');
ylabel('Amplitud');

y = zeros(1,20); % Inicializar vector de salida
y = [0 0 y]; % Establecer condiciones iniciales de y(n)

n = 0:1:19; % Vector de índices de tiempo

x = 0.8.^n; % Calcular señal de entrada
x = [0 0 x]; % Establecer condiciones iniciales de x(n)

for n=1:20
    y(n+2) = 2*x(n+2) - 4*x(n+1) - 0.5*y(n+1) - y(n); % Aplicar ecuación en
diferencias
end

n = 0:1:19; % Vector de índices de tiempo

% Graficar respuesta del sistema y(n) con diferentes condiciones iniciales
subplot(3, 1, 3);
stem(n, y(3:22));
title('Respuesta del sistema y(n) con diferentes condiciones iniciales');
xlabel('Índice de tiempo');
ylabel('Amplitud');
```

Código Filtinic/Filter

```
% Entregable    10 – Filtinic/Filter
% Grupo         5CV1
% Equipo        7
% Profesor: José Antonio Flores Escobar
% Integrantes:
%              Hernandez Rodriguez Juan Uriel
%              Vergara Martinez Brenda
%              García Quiroz Gustavo Ivan
%              Gutiérrez Jiménez Cinthia Nayelli
%              Ramírez Carrillo José Emilio
%              Iturbide Serrano Uriel

% Calcular la salida del sistema
%y(n) = 0.5y(n-2) + x(n-1)
%Para las primeras 4 muestras se usaran las siguientes condiciones iniciales:
%      y(-2) = 1
%      y(-1) = 0
%      x(-1) = -1
%      entrada x(n) = (0.5)^n u(n)

B = [0 1];
A = [1 0 -0.5];
x = [1 0.5 0.25 0.125];
Xi = [-1 0];
Yi = [0 1];
Zi = filtic(B, A, Yi, Xi);
```

```
y = filter(B, A, x, Zi);
```