Fast Fourier Transform (FFT)

Yarko Niño C. y Paulo Herrera R.

Departamento de Ingeniería Civil, Universidad de Chile

Semestre Primavera 2011

Transformada de Fourier

Un proceso físico puede ser descrito en el *dominio del tiempo* como una función $h(t)$ o en el *dominio de frecuencias* como $H(f)$. La relación entre las dos funciones está dada por,

$$H(f) = \int_{-\infty}^{\infty} h(t) e^{2\pi i f t} \, \mathrm{d}t$$

$$h(t) = \int_{-\infty}^{\infty} H(f) e^{-2\pi i f t} \, \mathrm{d}f$$

Si $h(t)$ es función del tiempo, entonces f tiene unidades de ciclos por segundo. También puede ser que h es función de la posición, entonces f es la inversa de la longitud de onda y tiene unidades de ciclos por metro. Otras unidades también son posibles.

Transformada de Fourier (cont.)

En física es común expresar la frecuencia como *frecuencia angular* [radianes/segundo]. En ese caso,

$$w = 2\pi f \qquad H(w) = [H(f)]_{f=w/2\pi}$$

y el par de transformadas se escribe como,

$$H(w) = \int_{-\infty}^{\infty} h(t) e^{iwt}\, \mathrm{d}t$$

$$h(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} H(w) e^{-iwt}\, \mathrm{d}w$$

Es decir, el uso de $H(w)$ **rompe la simetría del par de transformadas**. Por eso es mejor usar $H(f)$ para problemas computacionales.

Propiedades de transformada de Fourier

$h(t)$ es real	$H(-f) = [H(f)]^*$
$h(t)$ es imaginaria	$H(-f) = -[H(f)]^*$
$h(-t) = h(t)$ (par)	$H(-f) = H(f)$
$h(-t) = -h(t)$ (impar)	$H(-f) = -H(f)$
$h(t)$ es real y par	$H(f)$ es real y par
$h(t)$ es real e impar	$H(f)$ es imaginaria e impar
$h(t)$ es imaginaria y par	$H(f)$ es imaginaria y par
$h(t)$ es imaginaria e impar	$H(f)$ es real e impar

Propiedades de transformada de Fourier

Transformada de Fourier es una transformación lineal.

$h(at)$	$\frac{1}{\lvert a\rvert} H\left(\frac{f}{a}\right)$	time scaling
$\frac{1}{\lvert b\rvert} h\left(\frac{t}{b}\right)$	$H(bf)$	frequency scaling
$h(t-t_0)$	$\mathrm{H(f)}\ \mathrm{e}^{2\pi i f t_0}$	time shifting
$h(t)e^{-2\pi i f_0 t}$	$H(f-f_0)$	frequency shifting
$g * h$	$G(f)H(f)$	Convolution Theorem
$\mathrm{Corr}(g,h)$	$G(f)H(-f))$	Correlation Theorem

$$g * h \equiv \int_{-\infty}^{\infty} g(\tau)h(t-\tau)\,\mathrm{d}\tau$$

$$\mathrm{Corr}(g,h) \equiv \int_{-\infty}^{\infty} g(\tau+t)h(\tau)\,\mathrm{d}\tau$$

Propiedades de transformada de Fourier (cont.)

Teorema de Wiener-Kinchin (autocorrelación):

$$\mathrm{Corr}(g, g) \Leftrightarrow |G(f)|^2$$

donde usamos $G(-f) = [G(f)]^*$ si $g(t)$ es real.

Theorema de Parseval (Energía Total):

$$\mathrm{E} \equiv \int_{-\infty}^{\infty} |h(t)|^2 \, \mathrm{d}t = \int_{-\infty}^{\infty} |H(f)|^2 \, \mathrm{d}f$$

Power Spectral Density (PSD)

PREGUNTA: Cuánta energía está contenida entre f y $f + \mathrm{d}f$?

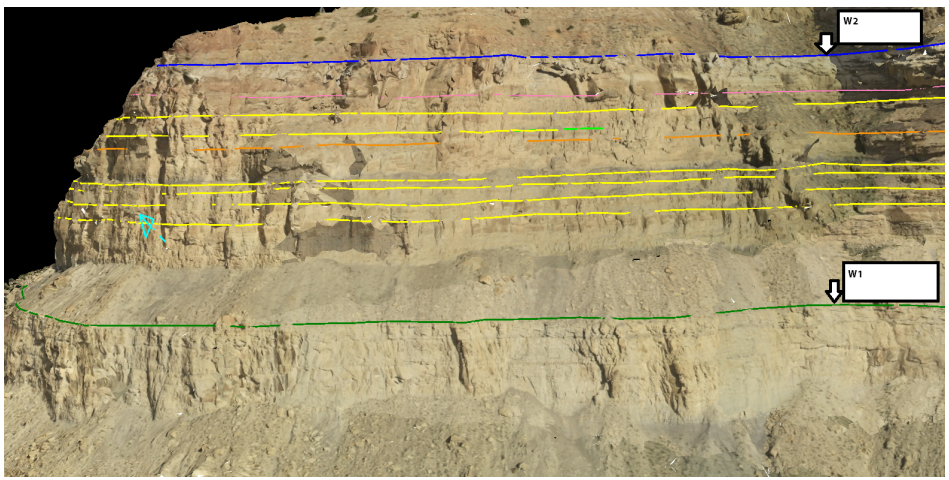
Si la función $h(t)$ es real entonces,

$$P_h(f) \equiv |H(f)|^2 + |H(-f)|^2 = 2|H(f)|^2 \qquad 0 \leq f < \infty$$

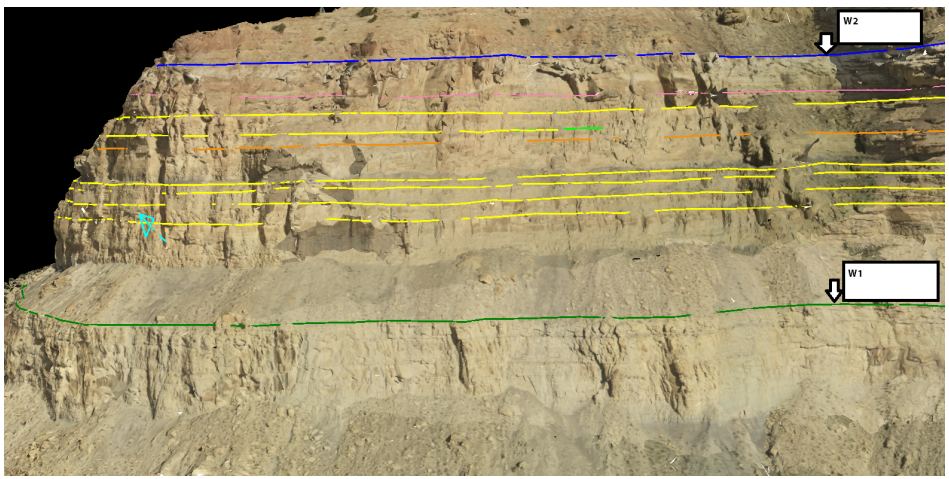
P_h es llamada *one-sided power spectral density (PSD)*.

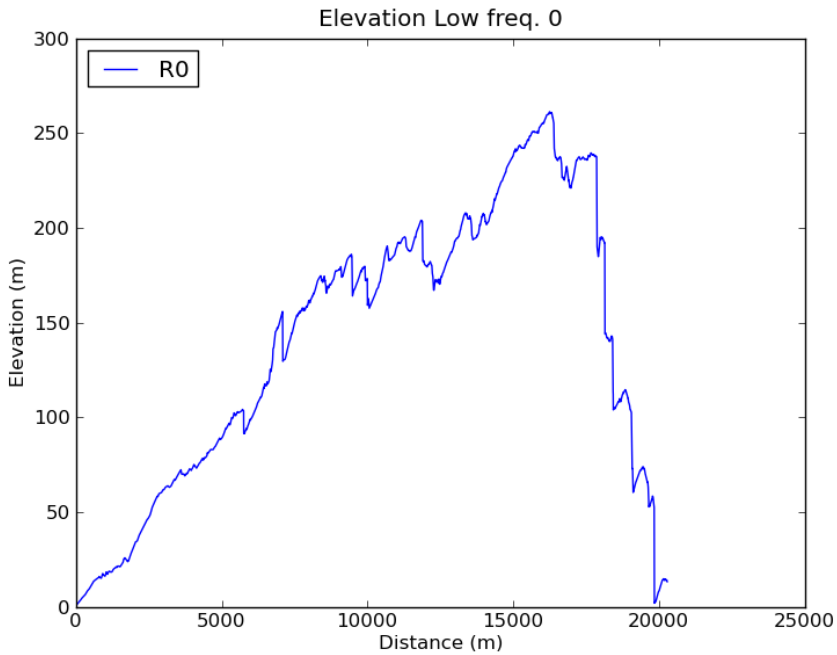
Normalmente, $h(t)$ es evaluada o medida en un intervalo acotado de tiempo Δt. En ese caso, se define la PSD por unidad de tiempo, dividiéndola por Δt.

Ejemplo: Topografía.

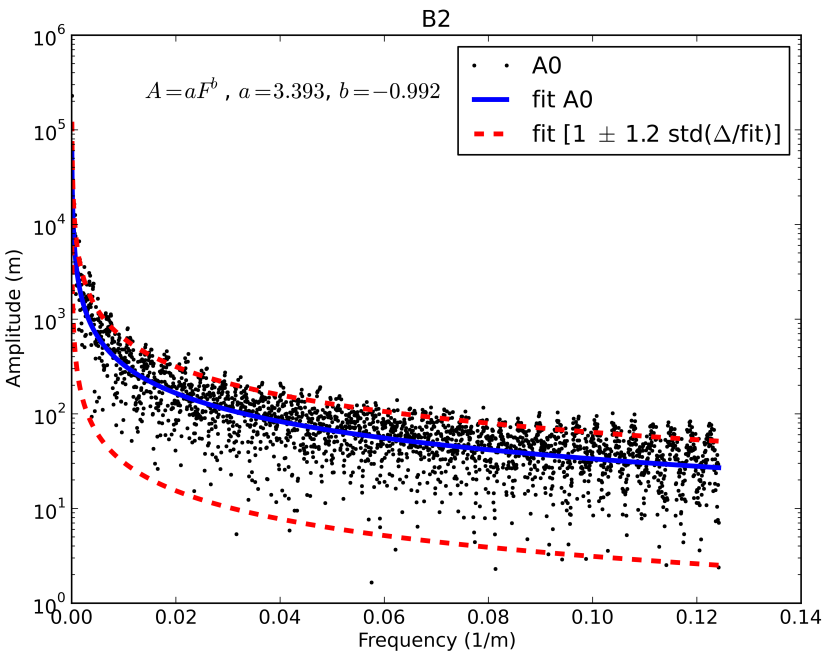


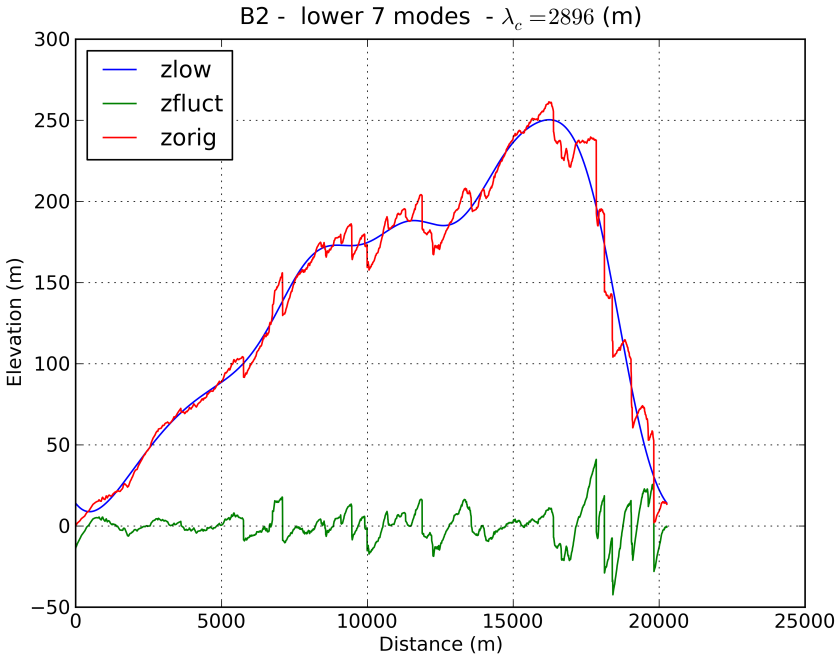
Ejemplo: Topografía.





Ejemplo: Topografía.





Series discretas de valores

En la mayoría de los casos $h(t)$ es medida o evaluada a intervalos regulares. Si el espaciamiento de las muestras es Δ, entonces los valores evaluados son:

$$h_n = h\left(n\Delta\right) \qquad n = \ldots, -2, -1, 0, 2, 3, \ldots$$

y la frecuencia de muestreo es $f = 1/\Delta$.

Teorema de Nyquist:
Sea,

$$f_c \equiv \frac{1}{2\Delta}$$

Si la función $h(t)$ contiene un espectro acotado de frecuencias, tal que $H(f) = 0, \quad \forall\, |f| \geq f_c$, entonces la función $h(t)$ **es completamente determinada por los valores medidos h_n.**

Teorema de Nyquist

Si la función $h(t)$ contiene un espectro acotado de frecuencias en el intervalo $-f_c < f < f_c$, entonces pueder ser reconstruida a partir de los valores medidos con intervalo Δ, usando,

$$h(t) = \Delta \sum_{n=-\infty}^{\infty} h_n \frac{\sin[2\pi f_c(t - n\Delta)]}{\pi(t - n\Delta)}$$

NOTAS:

- Este teorema establece que la *información* contenida en una función **continua** que tiene un espectro de frecuencia limitado, puede ser representada por un número **finito** de valores.
- También es conocido como **Nyquist-Shanon Sampling Theorem** en Teoría de la Información.

Teorema de Nyquist (cont.)

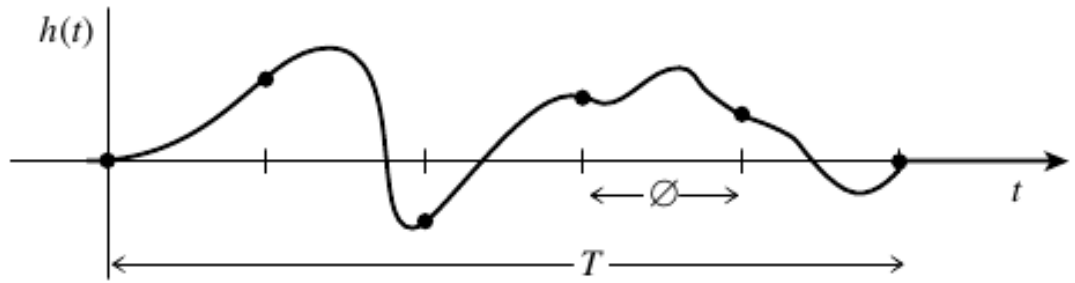
Uno de los corolarios más importantes del Teorema de Nyquist es que:

Si la función $h(t)$ contiene componentes de **frecuencia más alta que** f_c, la energía de esos componentes es **movida** al intervalo de frecuencias $-f_c < f < f_c$ (*aliasing*).

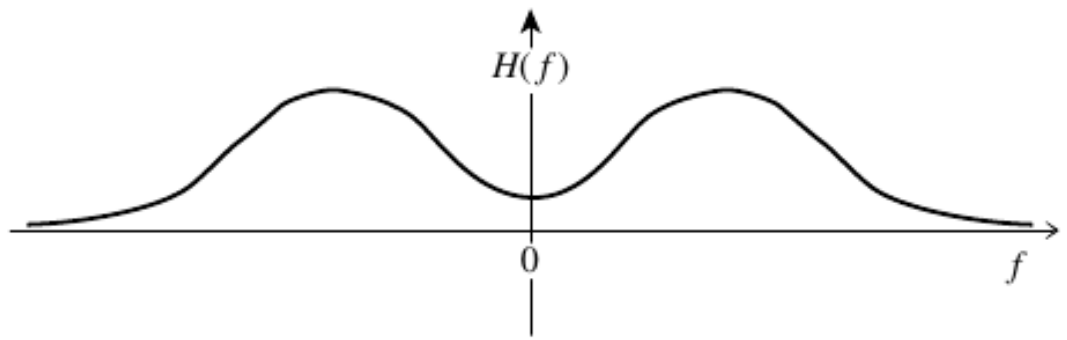
En general, aliasing no se puede remover.

Sin embargo podemos chequear si el contenido de la PSD se acerca a cero a media que las frecuencia se acerca a $-f_c$ o f_c. Si no lo hace, es probable que aliasing es importante.

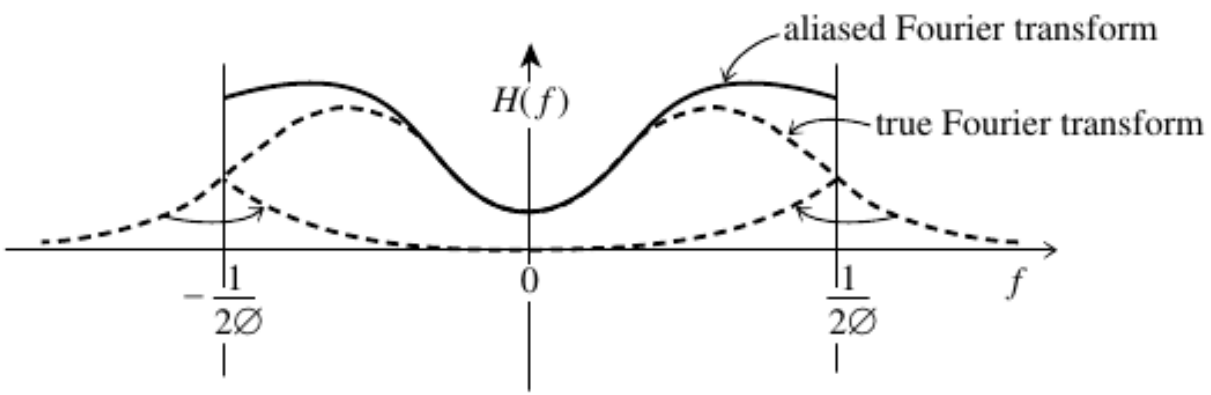
Aliasing



(a)



(b)



(c)

Transformada de Fourier Discreta

Considerando N valores consecutivos,

$$h_k \equiv h(t_k) = h(k\Delta) \qquad k = 0, 1, 2, \ldots, N-1$$

Supuestos:

- $h(t)$ es no cero sólo en los N puntos muestreados. Si no,
- $h(t)$ es *periódica*
- N es par (por ahora).

Tenemos N valores $\Rightarrow$ podemos producir N resultados. Definiendo,

$$f_n \equiv \frac{n}{N\Delta}, \qquad n = -\frac{N}{2}, \ldots, \frac{N}{2}$$

- $N+1$ valores en total, pero N valores independientes.
- Frecuencias extremas corresponden a las frecuencias de Nyquist.

Transformada de Fourier Discreta (cont.)

Aproximando la transformada de Fourier como una sumatoria,

$$H(f_n) = \int_{-\infty}^{\infty} h(t) e^{2\pi i f_n t} \, \mathrm{d}t \approx \sum_{k=0}^{N-1} h_k e^{2\pi i f_n t_k} \Delta = \Delta \sum_{k=0}^{N-1} h_k e^{2\pi i k n / N}$$

La transformada de Fourier discreta de los h_k valores es definida como,

$$H_n \equiv \sum_{k=0}^{N-1} h_k e^{2\pi i k n / N}$$

NOTA: H_n no depende de ningún parámetro dimensional.

La transformada de Fourier continua puede ser calculada a partir de la transformada discreta usando,

$$H(f_n) \approx \Delta H_n$$

Transformada de Fourier Discreta (cont.)

$$H(f_n) = \int_{-\infty}^{\infty} h(t) e^{2\pi i f_n t} \approx \sum_{k=0}^{N-1} h_k e^{2\pi i f_n t_k} \Delta = \Delta \sum_{k=0}^{N-1} h_k e^{2\pi i k n/N}$$

H_n es periódica en n con período N, entonces,

$$H_{-n} = H_{N-n} \qquad n = 1, 2, \ldots$$

Por lo tanto, más fácil usar $n = 0, 1, \ldots, N-1$. Con este sistema:

- $f = 0$ corresponde a $n = 0$.
- $f_c > f > 0$ corresponde a $1 \leq n \leq N/2 - 1$
- $-f_c < f < 0$ corresponde a $N/2 + 1 \leq n \leq N - 1$
- $-f_c$ y f_c corresponde a $N/2$

Transformada de Fourier Discreta (cont.)

Todas las propiedades de la transformada de Fourier también son válidas para H_n, con los cambios $H(f) \to H_n$ y $H(-f) \to H_{N-n}$.

La transformada inversa de H_n para recuperar los h_k es,

$$h_k = \frac{1}{N} \sum_{n=0}^{N-1} H_n e^{-2\pi i k n/N}$$

NOTA: En general, se usa la misma rutina para calcular H_n y recuperar h_k, por lo que, muchas veces hay que **normalizar** los resultados de la transformada inversa.

Transformada de Fourier Discreta (cont.)

La versión discreta del Teorema de Parseval es,

$$\sum_{k=0}^{N-1} |h_k|^2 = \frac{1}{N} \sum_{n=0}^{N-1} |H_n|^2$$

Transformada de Fourier Discreta (cont.)

Cuánto cuesta calcular H_n?

Procedimiento original: definir $W \equiv e^{2\pi i/N}$, entonces

$$H_n = \sum_{k=0}^{N-1} W^{nk} h_k$$

Equivalente a multiplicar $\mathbf{W} \cdot \mathbf{h}_k$, donde

$$W_{n,k} = W^{n \cdot k}$$

Multiplicación de la matriz requiere N^2 operaciones complejas + otras operaciones para generar $W_{n,k}$.

CONCLUSION: El cálculo ingenuo de la DFT es $O(N^2)$. **Esto es demasiado para valores grandes de N!!!**.

Fast Fourier Transform (FFT)

Danielson and Lanczos (1942), demostraron que:

$$
\begin{aligned}
F_k &= \sum_{j=0}^{N-1} e^{2\pi ijk/N} f_j \\
&= \sum_{j=0}^{N/2-1} e^{2\pi ik(2j)/N} f_{2j} + \sum_{j=0}^{N/2-1} e^{2\pi ik(2j+1)/N} f_{2j+1} \\
&= \sum_{j=0}^{N/2-1} e^{2\pi ikj/(N/2)} f_{2j} + W^k \sum_{j=0}^{N/2-1} e^{2\pi ikj/(N/2)} f_{2j+1} \quad = F_k^e + W^k F_k^o
\end{aligned}
$$

Es decir, el cálculo de la DFT se puede dividir en el cálculo de dos DFT: una sobre valores pares (F_k^o) y otra sobre valores impares (F_k^e).

Fast Fourier Transform (FFT) (cont.)

Si N es par, este procedimiento se puede aplicar recursivamente en $log_2 N$ niveles hasta llegar a calcular la DFT de un sólo elemento,

$$F_k^{eoeoeo...oee} = f_n \quad \text{for some } n$$

Por ejemplo, en un segundo nivel de recursión cada DFT es calculada sobre $N/4$ valores considerando elementos del tipo F_k^{ee} (impar-impar) y F_k^{eo} (impar-par).

PROBLEMA: Qué n corresponde a la combinación $eoeoeo \ldots oee$?

Fast Fourier Transform (FFT) (cont.)

Si N es par, este procedimiento se puede aplicar recursivamente en $log_2 N$ niveles hasta llegar a calcular la DFT de un sólo elemento,

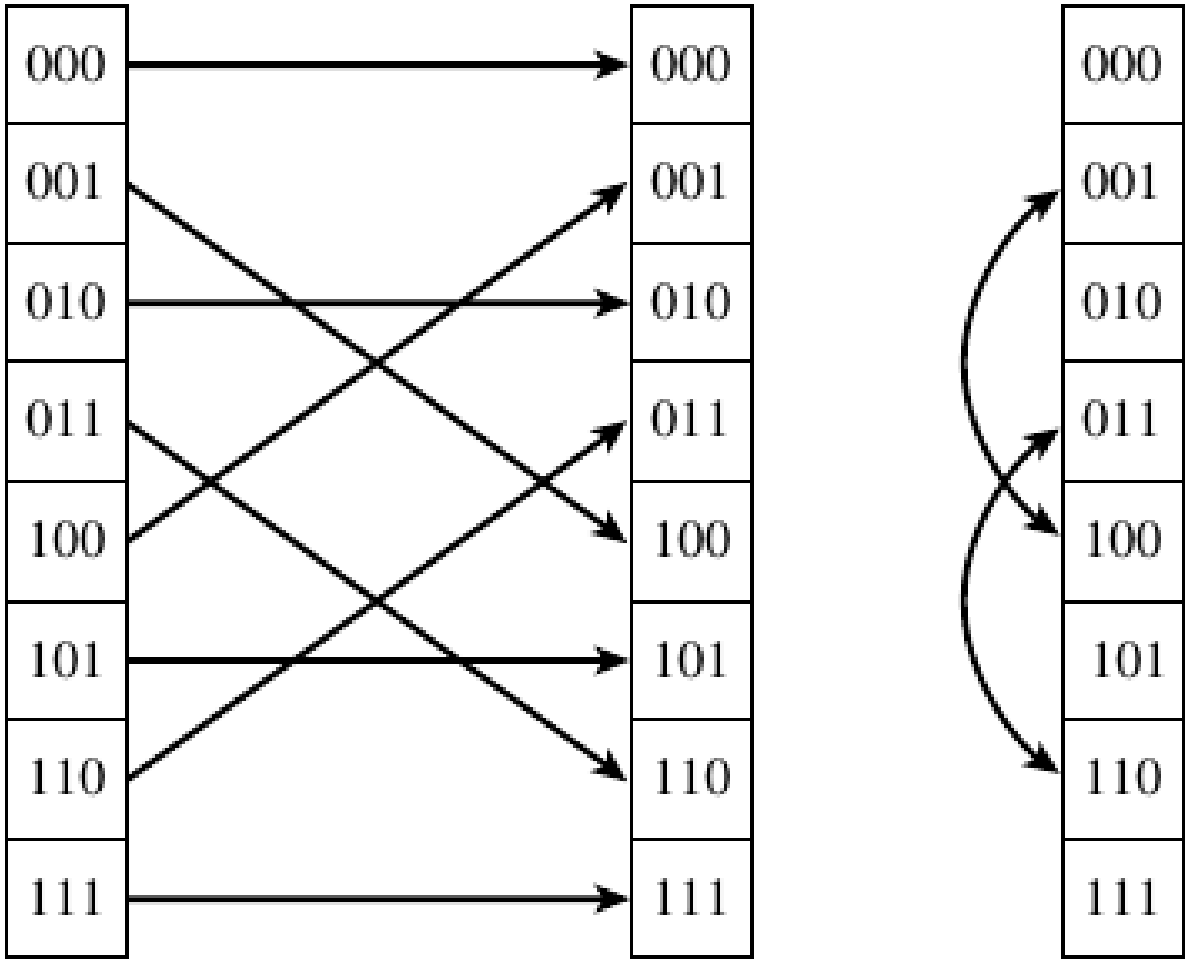
$$F_k^{eoeoeo...oee} = f_n \quad \text{for some } n$$

Por ejemplo, en un segundo nivel de recursión cada DFT es calculada sobre $N/4$ valores considerando elementos del tipo F_k^{ee} (impar-impar) y F_k^{eo} (impar-par).

PROBLEMA: Qué n corresponde a la combinación $eoeoeo \ldots oee$?

SOLUCION: Cambiar el sentido de la secuencia de es y os y asignar $e = 0$ y $o = 1$, entonces la representación binaria de la secuencia corresponde al valor de n.

Bit-reversing order



Valores pueden ser copiado y ordenados en un nuevo arreglo o en el mismo arreglo (*in-place*).

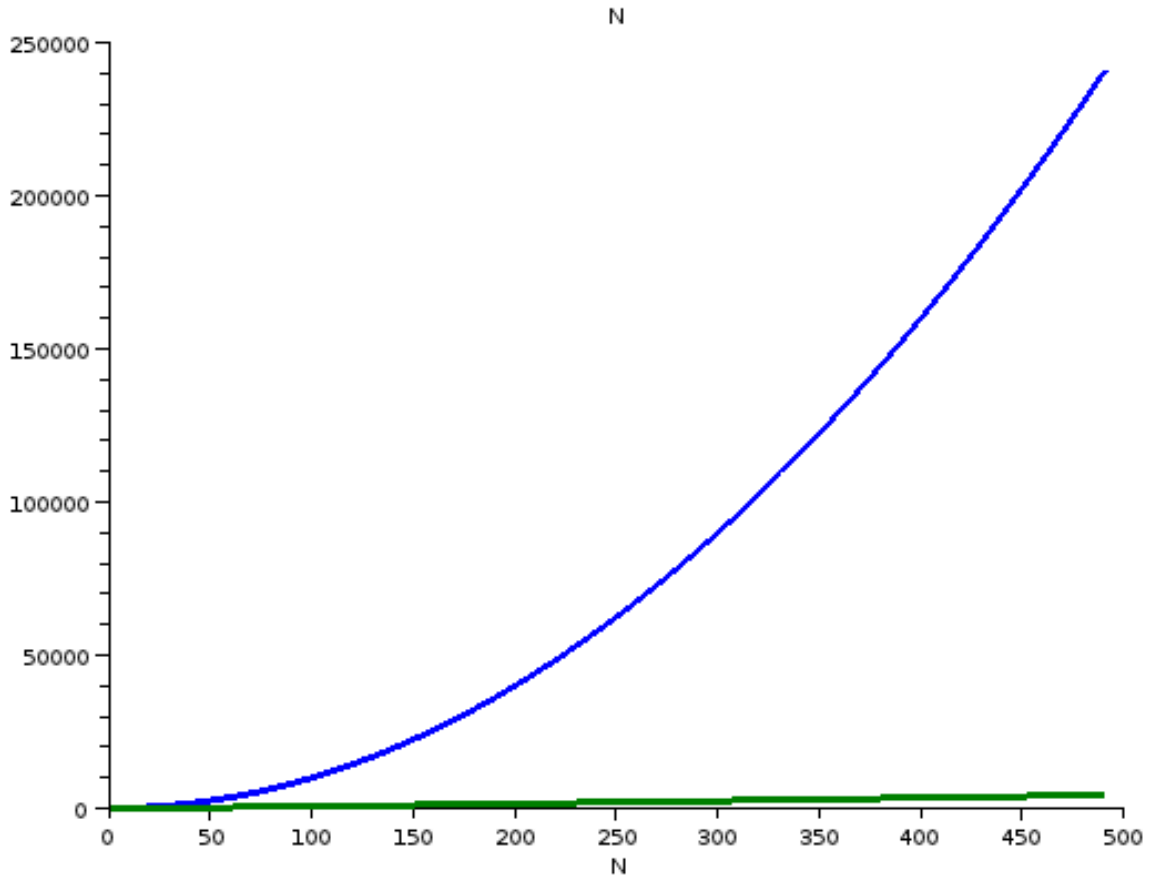
Bit-reversing order (cont.)

Depués de ordenar los valores, es fácil calcular la DFT a diferentes niveles, porque los coeficientes en cada nivel están almacenados en forma contigua.

Algoritmo:

1. Ordenar los valores de usando *bit-reversing*
2. Calcular DFTs para cada nivel. Esto requiere N operaciones por nivel, y tenemos $\log_2 N$ niveles $\Rightarrow N \log_2 N$ operaciones.

$O(N^2)$ vs $O(N \log_2 N)$



Para $N = 10^6$, $N^2 = 10^{12}$, $N \log_2 N \approx 2\,\mathtt{x}\,10^8$
Intel Core i7 980 XE puede alcanzar 109 GFLOPS (FLOPS = floating point operations per second)

Comentarios

- La FFT es considerada como uno (el) de los algoritmos más importantes en computación científica.
- Es una buena muestra del uso de ideas "ingeniosas" para acelerar un cálculo.
- La FFT es un buen ejemplo de algoritmos del tipo "Divide & Conquer".
- Aunque en principio la FFT fue desarrollada para N par, también hay implementanciones para N impar que utilizan soluciones híbridas (FFT + DFT).
- La FFT se aplica en muchos problemas prácticos: manipulación de imágenes (filtros), análisis de señales, métodos espectrales, estadística (correlación), evaluación de convolución, etc.