Numerical Analysis

FFT (Fast Fourier Transformation)

목 차

Introduction to the Periodic Function

- A periodic function *f(t)* is one for which

$$f(t) = f(t+T)$$

where T is a constant called the period that is the smallest value for which this equation holds.

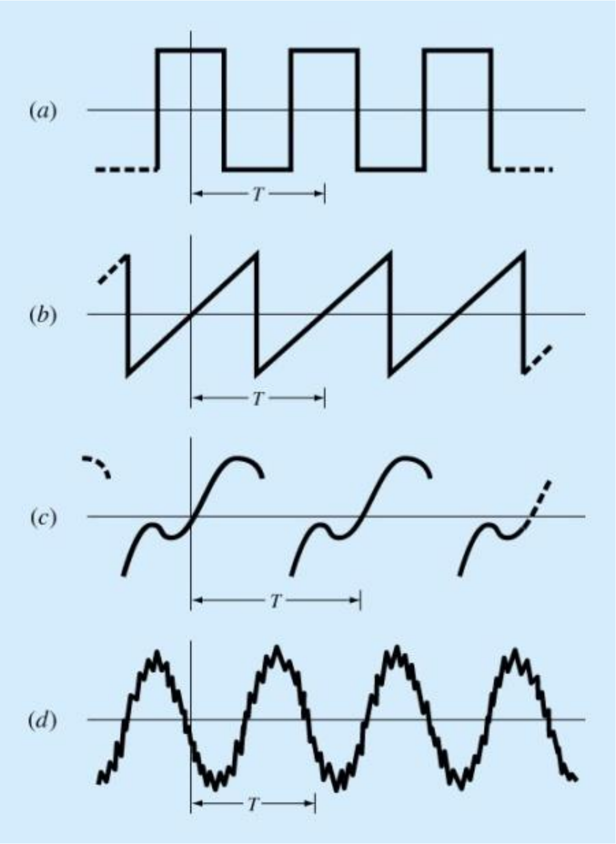
(Examples)

(a) square wave

(b) saw-tooth wave

(c) non-ideal wave

(d) wave contaminated by noise



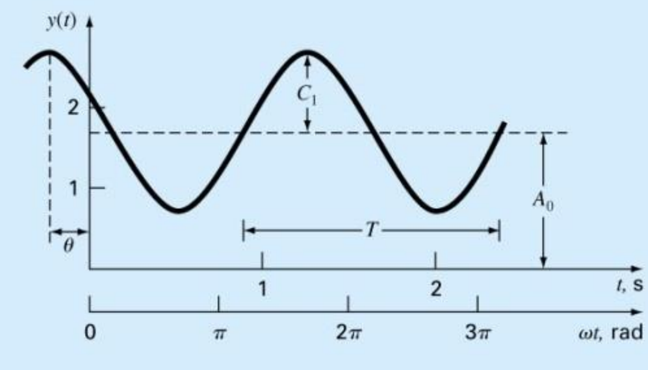
- Sinusoidal function
  - Any waveform that can be described as a sine or cosine is called sinusoid:

$$
\begin{aligned}
f(t) &= A_0 + C_1 \cos(\omega_0 t + \theta) \\
&= A_0 + C_1 \{\cos(\omega_0 t)\cos(\theta) - \sin(\omega_0 t)\sin(\theta)\} \\
&= A_0 + A_1 \cos(\omega_0 t) + B_1 \sin(\omega_0 t) \qquad \leftarrow A_1 = C_1 \cos(\theta),\ B_1 = -C_1 \sin(\theta)
\end{aligned}
$$

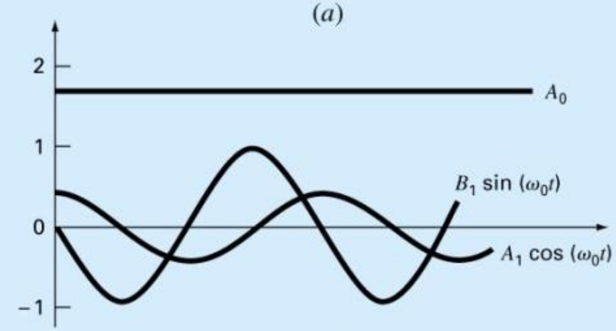
4-parameters

- A_0 : mean value
- C_1 : amplitude ($= \sqrt{A_1^2 + B_1^2}$)
- ω_0 :angular frequency
- θ : phase angle

$$\tan(\theta) = -\frac{B_1}{A_1} \rightarrow \theta = \arctan\left(-\frac{B_1}{A_1}\right)$$



(a)



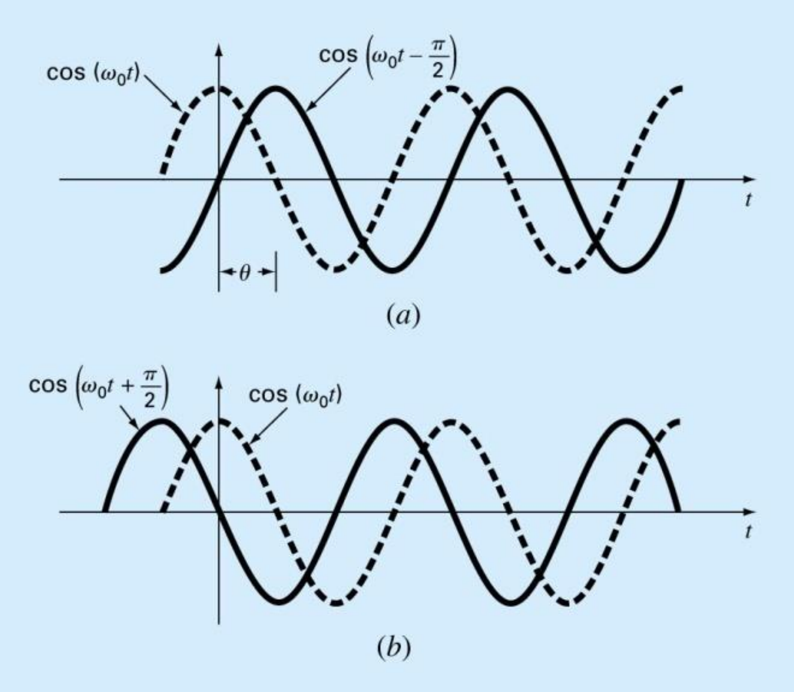
(b)

Introduction to the Periodic Function

- Definition of period: T

$$\omega_0 T = 2\pi \rightarrow T = \frac{2\pi}{\omega_0}$$

- Definition of the phase angle



○ Definition of Complex Exponent Function

$$e^{ix} = \cos(x) + i\sin(x)$$

$$e^{-ix} = \cos(x) + i\sin(-x) = \cos(x) - i\sin(x)$$

$$\cos(x) = \frac{1}{2}\left(e^{ix} + e^{-ix}\right)$$

$$\sin(x) = \frac{1}{2i}\left(e^{ix} - e^{-ix}\right) = \frac{i}{2}\left(e^{-ix} - e^{ix}\right)$$

→

$$e^{i(k\omega_0 t)} = \cos(k\omega_0 t) + i\sin(k\omega_0 t)$$

$$e^{-i(k\omega_0 t)} = \cos(k\omega_0 t) - i\sin(k\omega_0 t)$$

$$\cos(k\omega_0 t) = \frac{1}{2}\left(e^{i(k\omega_0 t)} + e^{-i(k\omega_0 t)}\right)$$

$$\sin(k\omega_0 t) = \frac{i}{2}\left(e^{-i(k\omega_0 t)} - e^{i(k\omega_0 t)}\right)$$

Example: Least Squares Approximation with Sinusoidal Functions

- Given N data points : $\{(t_1, y_1), (t_2, y_2), (t_3, y_3), \cdots, (t_j, y_j), \cdots, (t_N, y_N)\}$
- Regression: find a curve fitting best to the points $(t_j, y_j), \quad j = 1,2,\cdots,N$

$$y(t) = A_0 + A_1 \cos(\omega_0 t) + B_1 \sin(\omega_0 t) + e$$

$$e_j = A_0 + A_1 \cos(\omega_0 t_j) + B_1 \sin(\omega_0 t_j) - y_j, \quad j = 1,2,\cdots,N$$

$$\min J(A_0, A_1, B_1) = \sum_{j=1}^{N} e_j^{\,2} = \sum_{j=1}^{N} \left\{A_0 + A_1 \cos(\omega_0 t_j) + B_1 \sin(\omega_0 t_j) - y_j\right\}^2$$

optimality condition

$$\frac{\partial J}{\partial A_0} = 2\sum_{j=1}^{N} \left\{A_0 + A_1 \cos(\omega_0 t_j) + B_1 \sin(\omega_0 t_j) - y_j\right\} = 0$$

$$\frac{\partial J}{\partial A_1} = 2\sum_{j=1}^{N} \left\{A_0 + A_1 \cos(\omega_0 t_j) + B_1 \sin(\omega_0 t_j) - y_j\right\} \cos(\omega_0 t_j) = 0$$

$$\frac{\partial J}{\partial B_1} = 2\sum_{j=1}^{N} \left\{A_0 + A_1 \cos(\omega_0 t_j) + B_1 \sin(\omega_0 t_j) - y_j\right\} \sin(\omega_0 t_j) = 0$$

Example: Least Squares Approximation with Sinusoidal Functions

$$NA_0 + \left\{\sum \cos(\omega_0 t_j)\right\}A_1 + \left\{\sum \sin(\omega_0 t_j)\right\}B_1 = \sum y_j$$

$$\left\{\sum \cos(\omega_0 t_j)\right\}A_0 + \left\{\sum \cos^2(\omega_0 t_j)\right\}A_1 + \left\{\sum \sin(\omega_0 t_j)\cos(\omega_0 t_j)\right\}B_1 = \sum_{j=0}^{n}\left\{y_j \cos(\omega_0 t_j)\right\}$$

$$\left\{\sum \sin(\omega_0 t_j)\right\}A_0 + \left\{\sum \sin(\omega_0 t_j)\cos(\omega_0 t_j)\right\}A_1 + \left\{\sum \sin^2(\omega_0 t_j)\right\}B_1 = \sum_{j=0}^{n}\left\{y_j \sin\omega_0 t_j)\right\}$$

$$\boxed{\begin{pmatrix} N & \left\{\sum \cos(\omega_0 t_j)\right\} & \left\{\sum \sin(\omega_0 t_j)\right\} \\ \left\{\sum \cos(\omega_0 t_j)\right\} & \left\{\sum \cos^2(\omega_0 t_j)\right\} & \left\{\sum \sin(\omega_0 t_j)\cos(\omega_0 t_j)\right\} \\ \left\{\sum \sin(\omega_0 t_j)\right\} & \left\{\sum \sin(\omega_0 t_j)\cos(\omega_0 t_j)\right\} & \left\{\sum \sin^2(\omega_0 t_j)\right\} \end{pmatrix}\begin{pmatrix} A_0 \\ A_1 \\ B_1 \end{pmatrix} = \begin{pmatrix} \sum y_j \\ \sum_{j=0}^{n}\left\{y_j \cos(\omega_0 t_j)\right\} \\ \sum_{j=0}^{n}\left\{y_j \sin\omega_0 t_j)\right\} \end{pmatrix}}$$

- For a special case with equal spacing in t_j with the period T

$$h = \frac{T}{N-1}$$

$$t_j = t_1 + h(j-1), \quad j = 1,2,\cdots,N$$

where

$$T = \frac{2\pi}{\omega_0} \qquad \rightarrow \omega_0 T = 2\pi$$

$$f_0 = 2\pi\omega_0 = \frac{1}{T}\ (Hz) \rightarrow \text{frequency in Hertz}$$

Example: Least Squares Approximation with Sinusoidal Functions

$$\omega_0 t_j = \omega_0 \{t_1 + h(j-1)\}$$

$$e^{i\{\omega_0 t_j\}} = e^{i\omega_0 \{t_1 + h(j-1)\}} = e^{i\omega_0 \{t_1 + h(j-1)\}} = e^{i\omega_0 (t_1 - h)} e^{i\omega_0 jh} = e^{i\omega_0 (t_1 - h)} \left(e^{i\omega_0 h}\right)^j$$

$$\sum_{j=1}^{N} e^{i\{\omega_0 t_j\}} = \sum_{j=1}^{N} \left\{ e^{i\omega_0 (t_1 - h)} \left(e^{i\omega_0 h}\right)^j \right\} = e^{i\omega_0 (t_1 - h)} \sum_{j=1}^{N} \left(e^{i\omega_0 h}\right)^j = \frac{e^{i\omega_0 t_1}}{\Delta} \sum_{j=1}^{N} (\Delta)^j \leftarrow \Delta = \left(e^{i\omega_0 h}\right)$$

$$= e^{i\omega_0 t_1} \frac{(1 - \Delta^N)}{1 - \Delta} \leftarrow \Delta^N = e^{i\omega_0 N h}$$

If Nh is a integer multiple of T such as $Nh = kT$ for some positive integer k ,then

$$\omega_0 Nh = k\omega_0 T = 2\pi k$$

Therefore, $\Delta^N = e^{i\omega_0 N h} = e^{i2\pi k} = \cos(2\pi k) + i\sin(2\pi k) = 1$ and $\sum_{j=1}^{N} e^{i\{\omega_0 t_j\}} = 0$

Furthermore,

$$\sum_{j=1}^{N} e^{i\{\omega_0 t_j\}} = \sum_{j=1}^{N} \{\cos(\omega_0 t_j) + i\sin(\omega_0 t_j)\} = \sum_{j=1}^{N} \{\cos(\omega_0 t_j)\} + i\sum_{j=1}^{N} \{\sin(\omega_0 t_j)\} = 0$$

$$\rightarrow \sum_{j=1}^{N} \{\cos(\omega_0 t_j)\} = 0$$

$$\sum_{j=1}^{N} \{\sin(\omega_0 t_j)\} = 0$$

Example: Least Squares Approximation with Sinusoidal Functions

If we repeat above procedure for other terms using

$$\sin(\omega_0 t_j)\cos(\omega_0 t_j) = \frac{1}{2}\sin(2\omega_0 t_j) \rightarrow \sum \sin(\omega_0 t_j)\cos(\omega_0 t_j) = 0$$

$$\cos^2(\omega_0 t_j) = \frac{1}{2} + \frac{1}{2}\cos(2\omega_0 t_j) \rightarrow \sum \cos^2(\omega_0 t_j) = \frac{N}{2}$$

$$\sin^2(\omega_0 t_j) = \frac{1}{2} - \frac{1}{2}\cos(2\omega_0 t_j) \rightarrow \sum \sin^2(\omega_0 t_j) = \frac{N}{2}$$

Finally, we can get

$$\begin{pmatrix} N & 0 & 0 \\ 0 & N/2 & 0 \\ 0 & 0 & N/2 \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ B_1 \end{pmatrix} = \begin{pmatrix} \sum y_j \\ \sum \{y_j \cos(\omega_0 t_j)\} \\ \sum \{y_j \sin(\omega_0 t_j)\} \end{pmatrix} \rightarrow \begin{cases} A_0 = \dfrac{1}{N}\sum y_j \\ A_1 = \dfrac{2}{N}\sum \{y_j \cos(\omega_0 t_j)\} \\ B_1 = \dfrac{2}{N}\sum \{y_j \sin(\omega_0 t_j)\} \end{cases}$$

목　차

Continuous Fourier Series

- Fourier Series Expansion of a Continuous Function with the period T

$$f(t) = a_0 + a_1\cos(\omega_0 t) + b_1\sin(\omega_0 t) + a_2\cos(2\omega_0 t) + b_2\sin(2\omega_0 t) + \cdots$$

or more concisely

$$f(t) = f(t) = a_0 + \sum_{k=1}^{\infty}\left[a_k\cos(k\omega_0 t) + b_k\sin(k\omega_0 t)\right]$$

$$\boxed{T = \frac{2\pi}{\omega_0} \quad \rightarrow \omega_0 = \frac{2\pi}{T}}$$

ω_0 is called as the fundamental frequency

ω_0 : 1st harmonic, 1st harmonic amplitude → $A(\omega_0) = \sqrt{a_1^{\,2} + b_1^{\,2}}$

$2\omega_0$: 2nd harmonic, 2nd harmonic amplitude → $A(2\omega_0) = \sqrt{a_2^{\,2} + b_2^{\,2}}$

$3\omega_0$: 3rd harmonic, 3rd harmonic amplitude → $A(3\omega_0) = \sqrt{a_3^{\,2} + b_3^{\,2}}$

$4\omega_0$: 4th harmonic, 4th harmonic amplitude → $A(4\omega_0) = \sqrt{a_4^{\,2} + b_4^{\,2}}$

etc

Example: Least Squares Approximation with Sinusoidal Functions

- Important properties in the sinusoidal expansion of a continuous function with the period T

For two positive integers j,k

$$\cos(k\omega_0 t)\cos(j\omega_0 t) = \frac{1}{2}\{\cos(k-j)\omega_0 t + \cos(k+j)\omega_0 t\}$$

$$\sin(k\omega_0 t)\sin(j\omega_0 t) = \frac{1}{2}\{\cos(k-j)\omega_0 t - \cos(k+j)\omega_0 t\}$$

$$\sin(k\omega_0 t)\cos(j\omega_0 t) = \frac{1}{2}\{\sin(k+j)\omega_0 t + \sin(k-j)\omega_0 t\}$$

$$\int_0^T \cos(m\omega_0 t)dt = \begin{cases} T, & \text{if } m=0 \\ \left.\dfrac{\sin(m\omega_0 t)}{m\omega_0}\right|_0^T = \dfrac{\sin(m\omega_0 T)}{m\omega_0} = \dfrac{\sin(2\pi m)}{m\omega_0} = 0, & \text{if } m \neq 0 \end{cases}$$

$$\int_0^T \sin(m\omega_0 t)dt = \begin{cases} T, & \text{if } m=0 \\ 0, & \text{if } m \neq 0 \end{cases}$$

Example: Least Squares Approximation with Sinusoidal Functions

- Important properties in the sinusoidal expansion of a continuous function with the period T

Therefore,

$$\int_0^T \{\cos(k\omega_0 t)\cos(j\omega_0 t)\}dt = \begin{cases} \frac{T}{2}, & \text{if } k = j \\ 0, & \text{if } k \neq j \end{cases}$$

$$\int_0^T \{\sin(k\omega_0 t)\sin(j\omega_0 t)\}dt = \begin{cases} \frac{T}{2}, & \text{if } k = j \\ 0, & \text{if } k \neq j \end{cases}$$

$$\int_0^T \{\sin(k\omega_0 t)\cos(j\omega_0 t)\}dt = \begin{cases} \frac{T}{2}, & \text{if } k = j \\ 0, & \text{if } k \neq j \end{cases}$$

- Calculation of Fourier coefficients $a_0, a_k, b_k, \quad k = 1,2,3,\cdots$

$$f(t) = f(t) = a_0 + \sum_{k=1}^{\infty}\left[a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)\right]$$

$$f(t)\cos(j\omega_0 t) = a_0 \cos(j\omega_0 t) + \sum_{k=1}^{\infty}\left[a_k \cos(k\omega_0 t)\cos(j\omega_0 t) + b_k \sin(k\omega_0 t)\cos(j\omega_0 t)\right]$$

$$f(t)\sin(j\omega_0 t) = a_0 \sin(j\omega_0 t) + \sum_{k=1}^{\infty}\left[a_k \cos(k\omega_0 t)\sin(j\omega_0 t) + b_k \sin(k\omega_0 t)\sin(j\omega_0 t)\right]$$

Example: Least Squares Approximation with Sinusoidal Functions

(i) $j = 0$

$$\int_0^T f(t)dt = \int_0^T \left\{ a_0 + \sum_{k=1}^{\infty} [a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)] \right\} = a_0 T$$

(i) $j = 1,2,3,\cdots$

$$\int_0^T \{f(t)\cos(j\omega_0 t)\}dt = \int_0^T \left\{ a_0 \cos(j\omega_0 t) + \sum_{k=1}^{\infty} [a_k \cos(k\omega_0 t)\cos(j\omega_0 t) + b_k \sin(k\omega_0 t)\cos(j\omega_0 t)] \right\} = \frac{a_j T}{2}$$

$$\int_0^T \{f(t)\sin(j\omega_0 t)\}dt = \int_0^T \left\{ a_0 \sin(j\omega_0 t) + \sum_{k=1}^{\infty} [a_k \cos(k\omega_0 t)\sin(j\omega_0 t) + b_k \sin(k\omega_0 t)\sin(j\omega_0 t)] \right\} = \frac{b_j T}{2}$$

Therefore,

$$a_0 = \frac{1}{T}\int_0^T f(t)dt$$

$$a_j = \frac{2}{T}\int_0^T \{f(t)\cos(j\omega_0 t)\}dt \rightarrow$$

$$b_j = \frac{2}{T}\int_0^T \{f(t)\sin(j\omega_0 t)\}dt$$

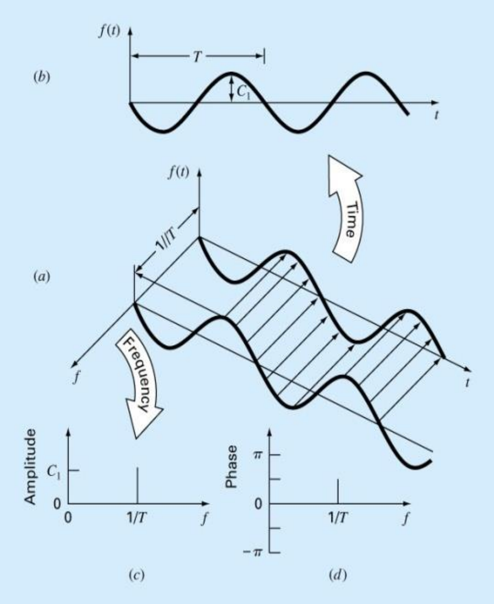
$$\boxed{\begin{aligned} a_0 &= \frac{1}{T}\int_0^T f(t)dt \\ a_k &= \frac{2}{T}\int_0^T \{f(t)\cos(k\omega_0 t)\}dt \\ b_k &= \frac{2}{T}\int_0^T \{f(t)\sin(k\omega_0 t)\}dt \end{aligned}}$$

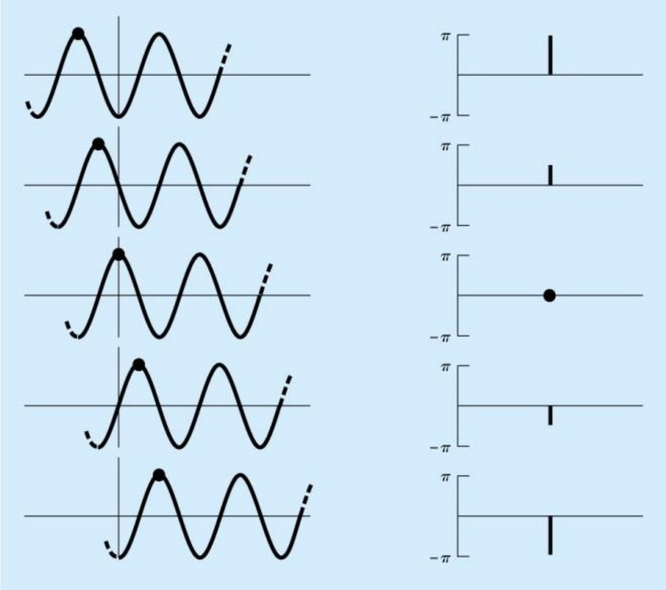
$k = 1,2,3,\cdots$

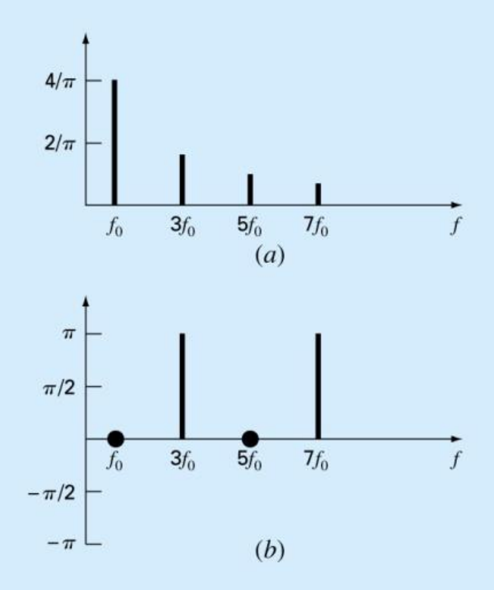
Frequency Domain Concept Compared to Time Domain Record

- Time domain Plot
  - x-axis: time t
  - y-axis: time varying amplitude $f(t)$
- Frequency domain Plot
  - x-axis: frequency $f = 2\pi\omega = \dfrac{1}{T}\,(Hz)$
  - y-axis: amplitude at each frequency

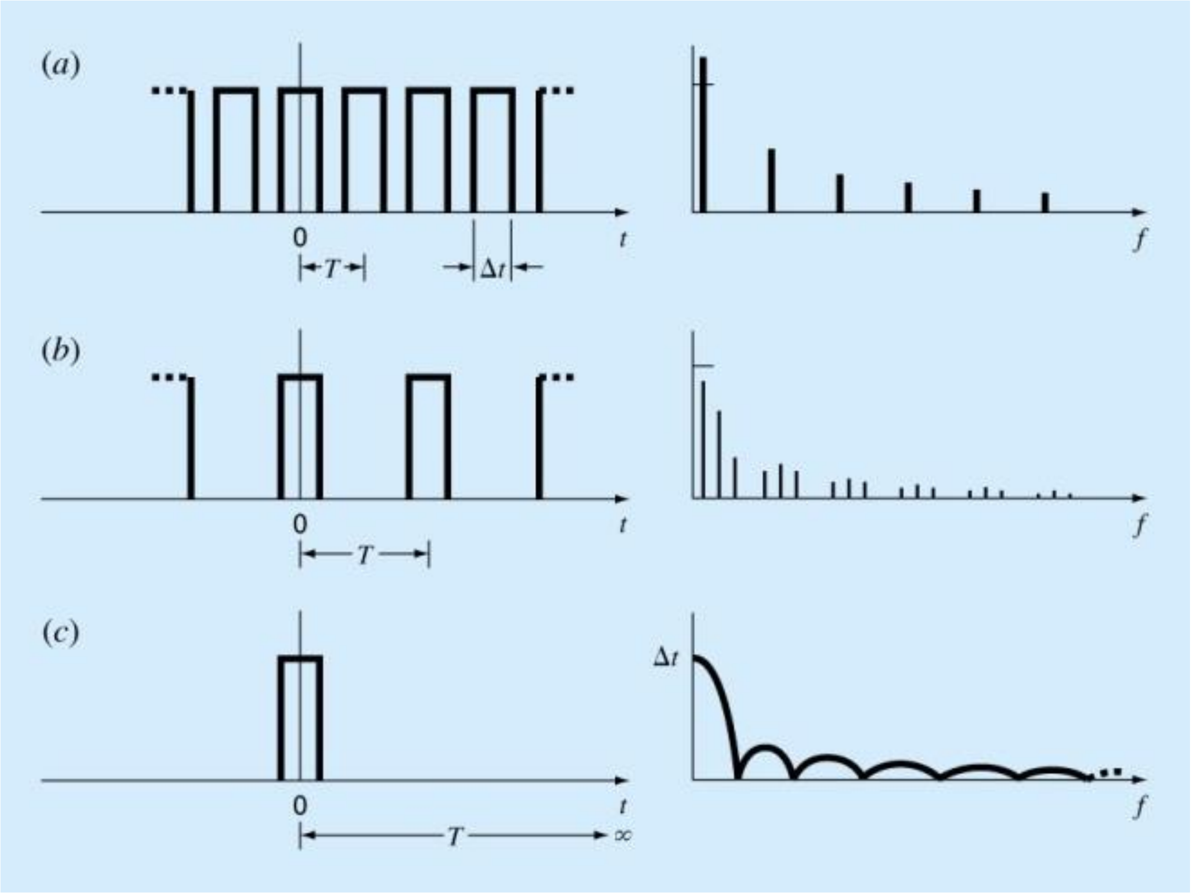
    phase angle at each frequency







Frequency Domain Concept Compared to Time Domain Record



Complex Form of the Fourier Series

- Using the following relations

$$\cos(k\omega_0 t) = \frac{1}{2}\left(e^{i(k\omega_0 t)} + e^{-i(k\omega_0 t)}\right)$$

$$\sin(k\omega_0 t) = \frac{i}{2}\left(e^{-i(k\omega_0 t)} - e^{i(k\omega_0 t)}\right)$$

$$\begin{aligned} f(t) &= a_0 + \sum_{k=1}^{\infty}\left[a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)\right] \\ &= a_0 + \sum_{k=1}^{\infty}\left[\frac{1}{2}a_k\left(e^{i(k\omega_0 t)} + e^{-i(k\omega_0 t)}\right) + \frac{i}{2}b_k\left(e^{-i(k\omega_0 t)} - e^{i(k\omega_0 t)}\right)\right] \\ &= a_0 + \sum_{k=1}^{\infty}\left[\frac{1}{2}(a_k - i\,b_k)e^{i(k\omega_0 t)} + \frac{1}{2}(a_k + i\,b_k)e^{-i(k\omega_0 t)}\right] \end{aligned}$$

Let's define

$$a_{-k} = a_k$$

$$b_{-k} = -b_k$$

$$b_0 = 0$$

Complex Form of the Fourier Series

Then

$$f(t) = a_0 + \sum_{k=1}^{\infty}\left[\frac{1}{2}(a_k - i\,b_k)e^{i(k\omega_0 t)}\right] + \sum_{k=1}^{\infty}\left[\frac{1}{2}(a_{-k} - i\,b_{-k})e^{-i(k\omega_0 t)}\right]$$

$$= (a_0 - i\,b_0)e^{i(0\times\omega_0 t)} + \sum_{k=1}^{\infty}\left[\frac{1}{2}(a_k - i\,b_k)e^{i(k\omega_0 t)}\right] + \sum_{k=-1}^{-\infty}\left[\frac{1}{2}(a_k - i\,b_k)e^{i(k\omega_0 t)}\right]$$

$$= \sum_{k=-\infty}^{\infty}\left[\frac{1}{2}(a_k - i\,b_k)e^{i(k\omega_0 t)}\right]$$

$$= \sum_{k=-\infty}^{\infty} c_k e^{i(k\omega_0 t)} \quad \leftarrow c_k = \frac{1}{2}(a_k - i\,b_k)$$

$$\rightarrow \boxed{\begin{aligned} f(t) &= \sum_{k=-\infty}^{\infty} c_k e^{i(k\omega_0 t)} \\ c_k &= \frac{1}{T}\int_{-T/2}^{T/2}\left\{f(t)e^{-k\omega_0 t}\right\}dt \end{aligned}}$$

k-th harmonic amplitude $= \boxed{A(k\omega_0) = |\,c_k\,|}$

k-th harmonic phase $= \boxed{\theta(k\omega_0) = \arctan\left(\frac{\mathrm{Im}(c_k)}{\mathrm{Re}(c_k)}\right)}$

Fourier Integral and Inverse Transform

- Complex Form of the Fourier Series

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{i(k\omega_0 t)}$$

$$c_k = \frac{1}{T}\int_{-T/2}^{T/2}\left\{f(t)e^{-i(k\omega_0 t)}\right\}dt$$

- Fourier Integral

$$F(i\omega_0) = \int_{-\infty}^{\infty} f(t)e^{-i\omega_0 t}d\omega$$

- Inverse Fourier Transform

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(i\omega_0)^{i\omega_0 t}d\omega$$

Inverse Fourier transform of $F(i\omega_0)$

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(i\omega_0)e^{i\omega_0 t}d\omega_0$$

$$F(i\omega_0) = \int_{-\infty}^{\infty} f(t)e^{i\omega_0 t}dt$$

Fourier transform pair

Fourier integral of $f(t)$, or Fourier transform of $f(t)$

목 차

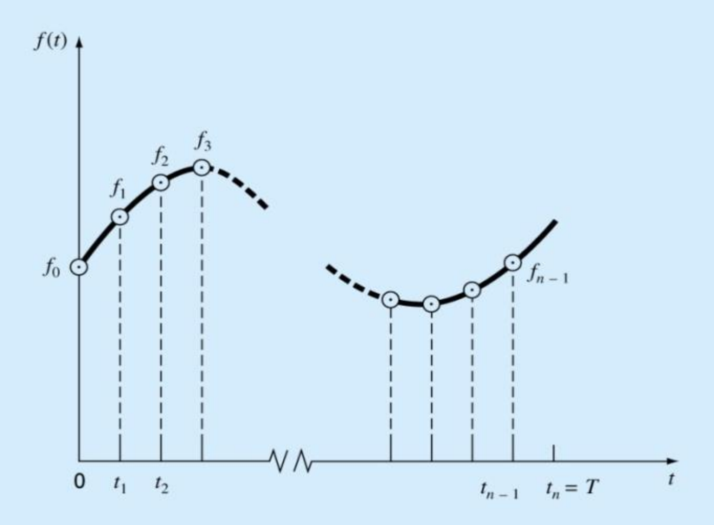
Discrete Fourier Transform (DFT)

- In engineering, functions are often represented by finite sets of discrete values and data is often collected in or converted to such a discrete format.

- An interval from 0 to T can be divided into N equi-spaced subintervals with widths of

$$\Delta t = \frac{T}{N} \quad \rightarrow \quad \begin{aligned} t_k &= k\Delta t, \quad k = 0,1,2,\cdots \\ t_N &= T \end{aligned}$$

$$\omega_0 = \frac{2\pi}{T}$$



- DFT formula

$$\boxed{\begin{aligned} F_k &= \sum_{n=0}^{N-1} f_n e^{-i(k\omega_0 n)} \quad \text{for } k = 0,1,2,\cdots,N-1 \\ &= \sum_{n=0}^{N-1} f_n \{\cos(k\omega_0 n) - i\sin(k\omega_0 n)\} \end{aligned}}$$

- Inverse DFT formula

$$\boxed{\begin{aligned} f_n &= \frac{1}{2\pi}\sum_{k=0}^{N-1} F_k e^{i(k\omega_0 n)} \quad \text{for } n = 0,1,2,\cdots,N-1 \\ &= \frac{1}{2\pi}\sum_{k=0}^{N-1} F_k \{\cos(k\omega_0 n) + \sin(k\omega_0 n)\} \end{aligned}}$$

Fast Fourier Transform (FFT)

- FFT is an algorithm that has been developed to compute the DFT in an extremely economical (fast) fashion by using the results of previous computations to reduce the number of operations.
- FFT exploits the periodicity and symmetry of trigonometric functions to compute the transform with approximately $N \log_2 N$ operations. Thus for N=50 samples, the FFT is 10 times faster than the standard DFT. For N=1000, it is about 100 times faster.

(1) Sande-Tukey Algorithm for the 1st stage

$$
\begin{aligned}
F_k &= \sum_{n=0}^{N-1} f_n e^{-i(k\omega_0 n)} \quad \text{for } k = 0,1,2,\cdots,N-1 \\
&= \sum_{n=0}^{N-1} f_n \{\cos(k\omega_0 n) - i\sin(k\omega_0 n)\}
\end{aligned}
$$

- Normalize time with Δt as

$$\hat{t}_k = \frac{t_k}{\Delta t} = k$$

$$\hat{T} = \frac{T}{\Delta t} = N$$

$\leftarrow$

$$\boxed{\begin{aligned} t_k &= k\Delta t, \quad k = 0,1,2,\cdots \\ t_N &= T = N\Delta t \end{aligned}}$$

Fast Fourier Transform (FFT)

$$\hat{\omega}_0 = \frac{2\pi}{\hat{T}} = \frac{2\pi\Delta t}{T} = \frac{2\pi}{N}$$

$$\frac{\hat{\omega}_0 kN}{2} = \pi k$$

$$e^{-\left(\frac{\hat{\omega}_0 kN}{2}\right)} = e^{-\pi k} = (-1)^k$$

- Let's define a complex-valued weight function W as

$$W = e^{-i\hat{\omega}}$$

$$F_k = \sum_{n=0}^{N-1} f_n W^{nk}$$

$$= \sum_{n=0}^{(N/2)-1} f_n W^{nk} + \sum_{n=N/2}^{N-1} f_n W^{nk}$$

$$= \sum_{n=0}^{(N/2)-1} f_n W^{nk} + \sum_{m=0}^{(N/2)-1} f_{m+N/2} W^{(m+N/2)k}$$

$$= \sum_{n=0}^{(N/2)-1} \left(f_n + f_{n+N/2} W^{(N/2)k}\right) W^{nk} \leftarrow W^{(N/2)k} = e^{-i\hat{\omega}_0 (N/2)k} = (-1)^k$$

$$= \sum_{n=0}^{(N/2)-1} \left(f_n + f_{n+N/2} (-1)^k\right) W^{nk}$$

Fast Fourier Transform (FFT)

Therefore,

$$F_{2k+1} = \sum_{n=0}^{(N/2)-1} \left(f_n - f_{n+N/2}\right) W^{n(2k+1)}, \quad k = 0,1,2,\cdots,(N/2)-1$$

for odd number 2k+1 →

$$= \sum_{n=0}^{(N/2)-1} \left(f_n - f_{n+N/2}\right) W^{n} W^{2kn}$$

$$= \sum_{n=0}^{(N/2)-1} h_n W^{2kn} \leftarrow h_n = \left(f_n - f_{n+N/2}\right) W^{n}$$

$$F_{2k} = \sum_{n=0}^{(N/2)-1} \left(f_n + f_{n+N/2}\right) W^{2kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

for even number 2k →

$$= \sum_{n=0}^{(N/2)-1} g_n W^{2kn} \leftarrow g_n = f_n + f_{n+N/2}$$

Let's define

$$G_k = \sum_{n=0}^{(N/2)-1} g_n W^{2kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

$$H_k = \sum_{n=0}^{(N/2)-1} h_n W^{2kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

Fast Fourier Transform (FFT)

Then the computation of $F_k = \sum_{n=0}^{N-1} f_n W^{kn}$ can be written as

$$F_{2k} = G_k = \sum_{n=0}^{(N/2)-1} g_n W^{2kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

$$F_{2k+1} = H_k = \sum_{n=0}^{(N/2)-1} h_n W^{2kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

with

$$g_n = f_n + f_{n+N/2}$$

$$h_n = \left(f_n - f_{n+N/2}\right) W^n$$

And the resultant computing time has been halved.

Fast Fourier Transform (FFT)

(2) Sande-Tukey Algorithm for the 2nd stage

Using $F_k = \sum_{n=0}^{N-1} f_n W^{kn}$

$$F_{2k} = G_k = \sum_{n=0}^{(N/2)-1} g_n W^{2kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

$$F_{2k+1} = H_k = \sum_{n=0}^{(N/2)-1} h_n W^{2kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

with

$$g_n = f_n + f_{n+N/2}$$

$$h_n = \left(f_n - f_{n+N/2}\right) W^n$$

$$F_{4k} = \sum_{n=0}^{(N/4)-1} g_{2n} W^{4kn}, \quad k = 0,1,2,\cdots,(N/4)-1$$

$$F_{4k+2} = \sum_{n=0}^{(N/2)-1} g_{2n+1} W^{2(2k+1)n}, \quad k = 0,1,2,\cdots,(N/4)-1$$

$$F_{4k+1} = \sum_{n=0}^{(N/2)-1} h_{2n} W^{4kn}, \quad k = 0,1,2,\cdots,(N/4)-1$$

$$F_{4k+3} = \sum_{n=0}^{(N/2)-1} h_{2n+1} W^{4kn}, \quad k = 0,1,2,\cdots,(N/2)-1$$

with

$$g_{2n} = g_n + g_{n+N/2} = f_n + f_{n+N/2} + f_{n+N/2} + f_{n+N/4}$$

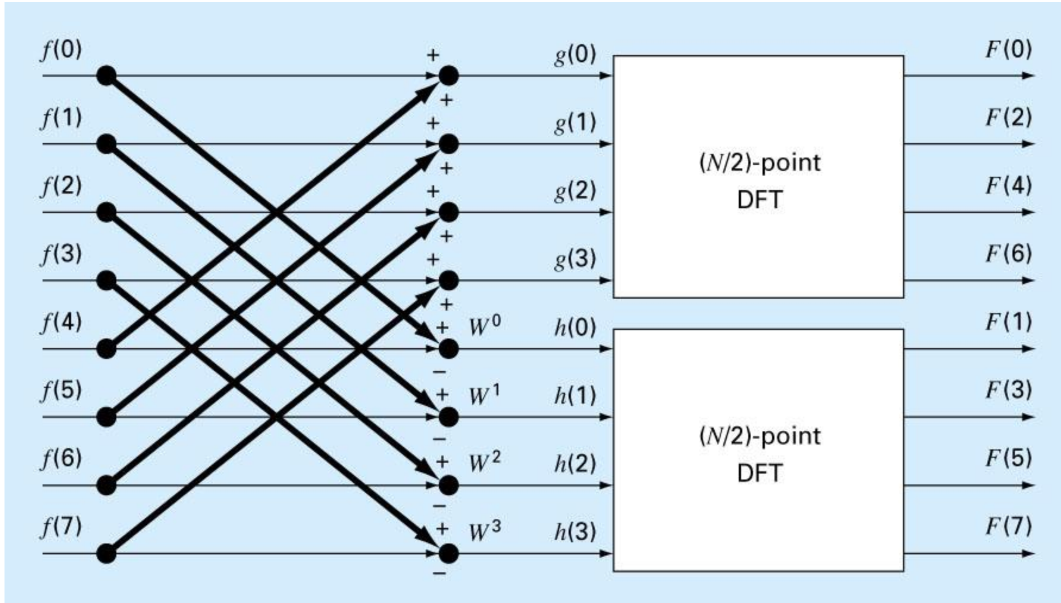
$$g_{2n+1} = (g_n - g_{n+N/2}) W^{2n} = (f_n + f_{n+N/2} - f_{n+N/2} - f_{n+N/4}) W^{2n}$$

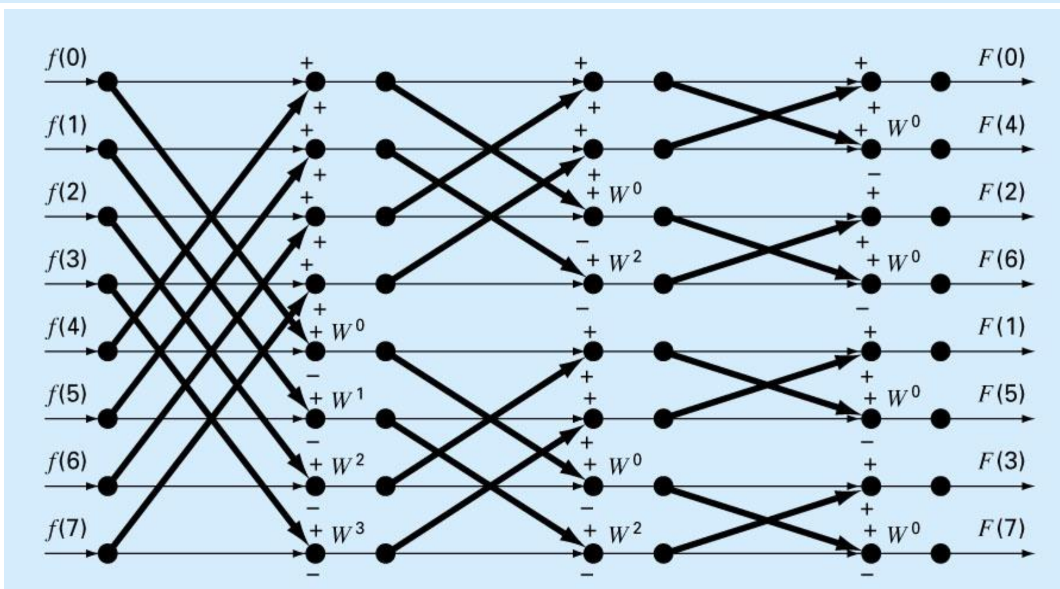
$$h_{2n} = \left(h_n + h_{n+N/2}\right) = \left(f_n - f_{n+N/2}\right) W^n + \left(f_n - f_{n+N/4}\right) W^n$$

$$h_{2n+1} = \left(h_n - h_{n+N/2}\right) W^{2n} = f_n + f_{n+N/2}$$

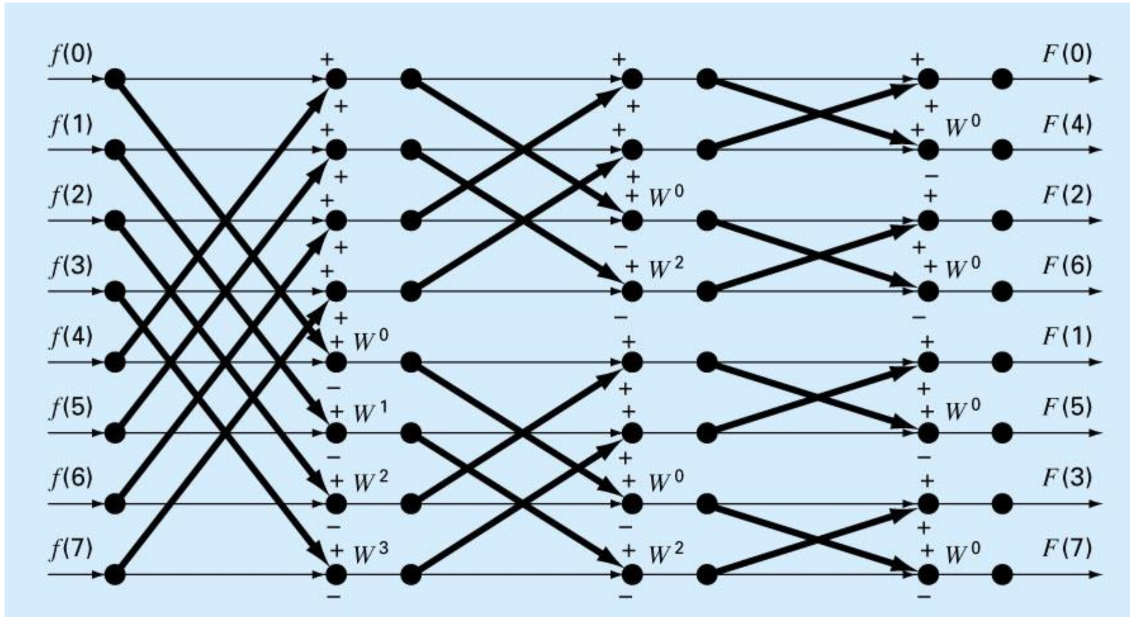
$$h_n = \left(f_n - f_{n+N/2}\right) W^n$$

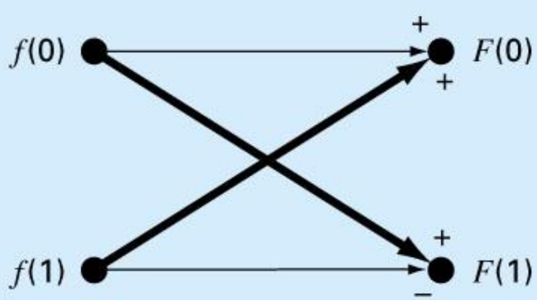
Fast Fourier Transform (FFT)





Fast Fourier Transform (FFT)





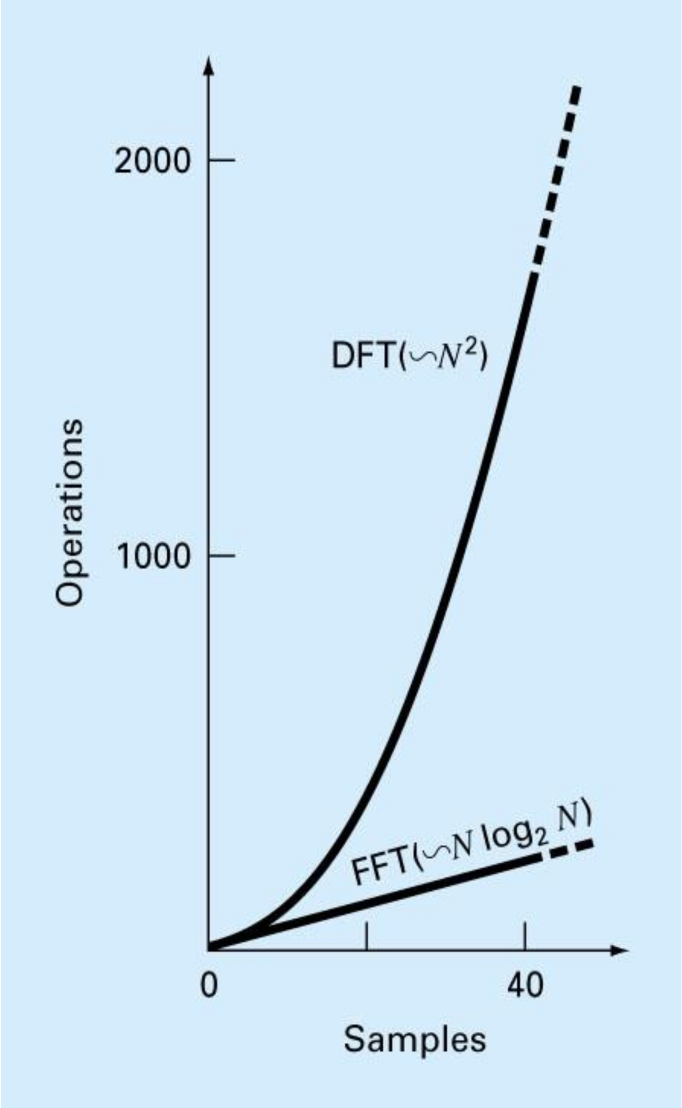
(a)

temporary = real (0) + real (1)
real (1) = real (0) – real (1)
real (0) = temporary
temporary = imaginary (0) + imaginary (1)
imaginary (1) = imaginary (0) – imaginary (1)
imaginary (0) = temporary

(b)

Fast Fourier Transform (FFT)

- Comparison of the Computational Time between DFT and FFT



목 차

Important topics in Discrete Fourier Transform (I): Aliasing

1. Aliasing : Folding the frequency contents with varying sampling intervals

Function: y(t) = cos(60*t);

Sampling intervals: Dt= 0.001 sec, 0.01 sec, 0.05 sec, 0.1 sec

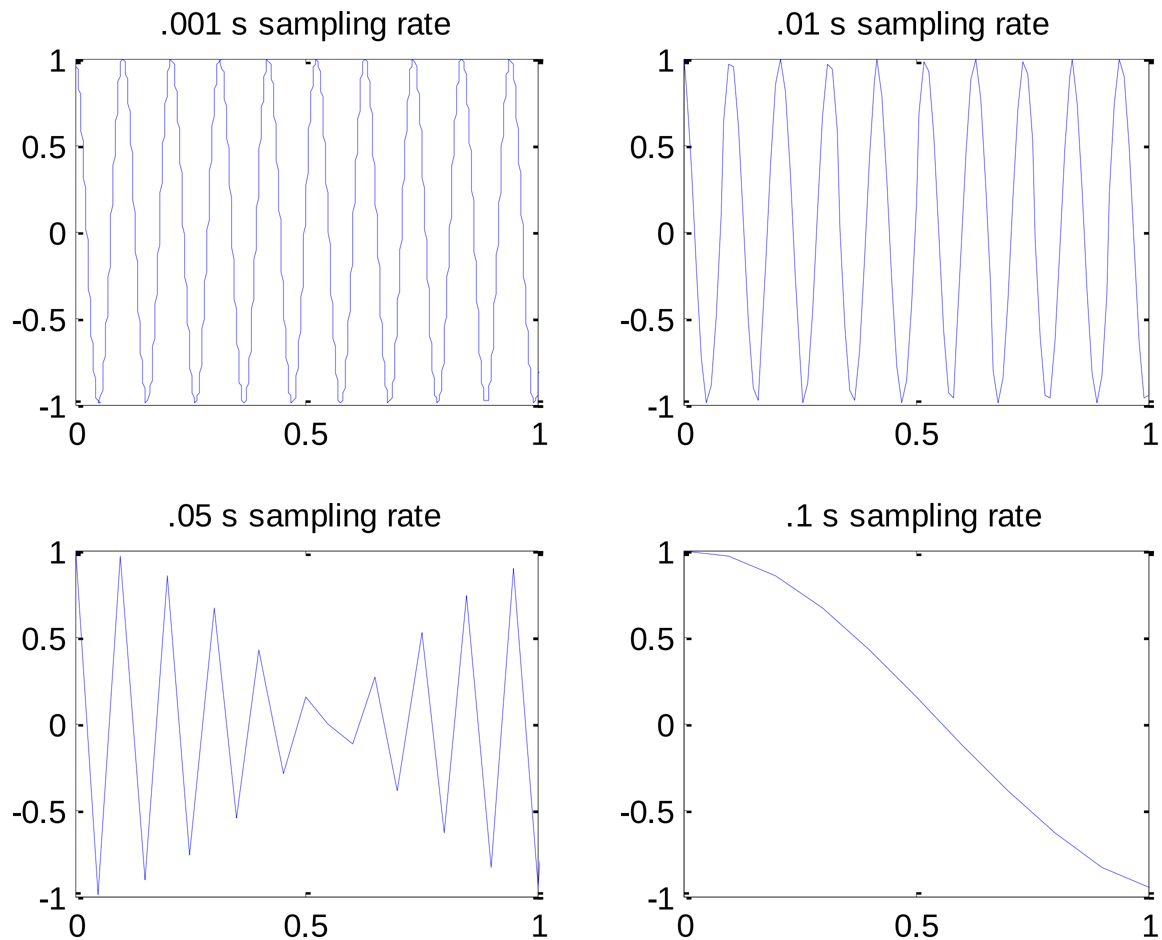
One would expect that if the signal has significant variation then Ts must be small enough to provide an accurate approximation of the signal x(t). Significant signal variation usually implies that high frequency components are present in the signal. It could therefore be inferred that the higher the frequency of the components present in the signal, the higher the sampling rate should be. If the sampling rate is not high enough to sample the signal correctly then a phenomenon called aliasing occurs.

The term aliasing refers to the distortion that occurs when a continuous time signal has frequencies larger than half of the sampling rate. The process of aliasing describes the phenomenon in which components of the signal at high frequencies are mistaken for components at lower frequencies.

Important topics in Discrete Fourier Transform (I): Aliasing

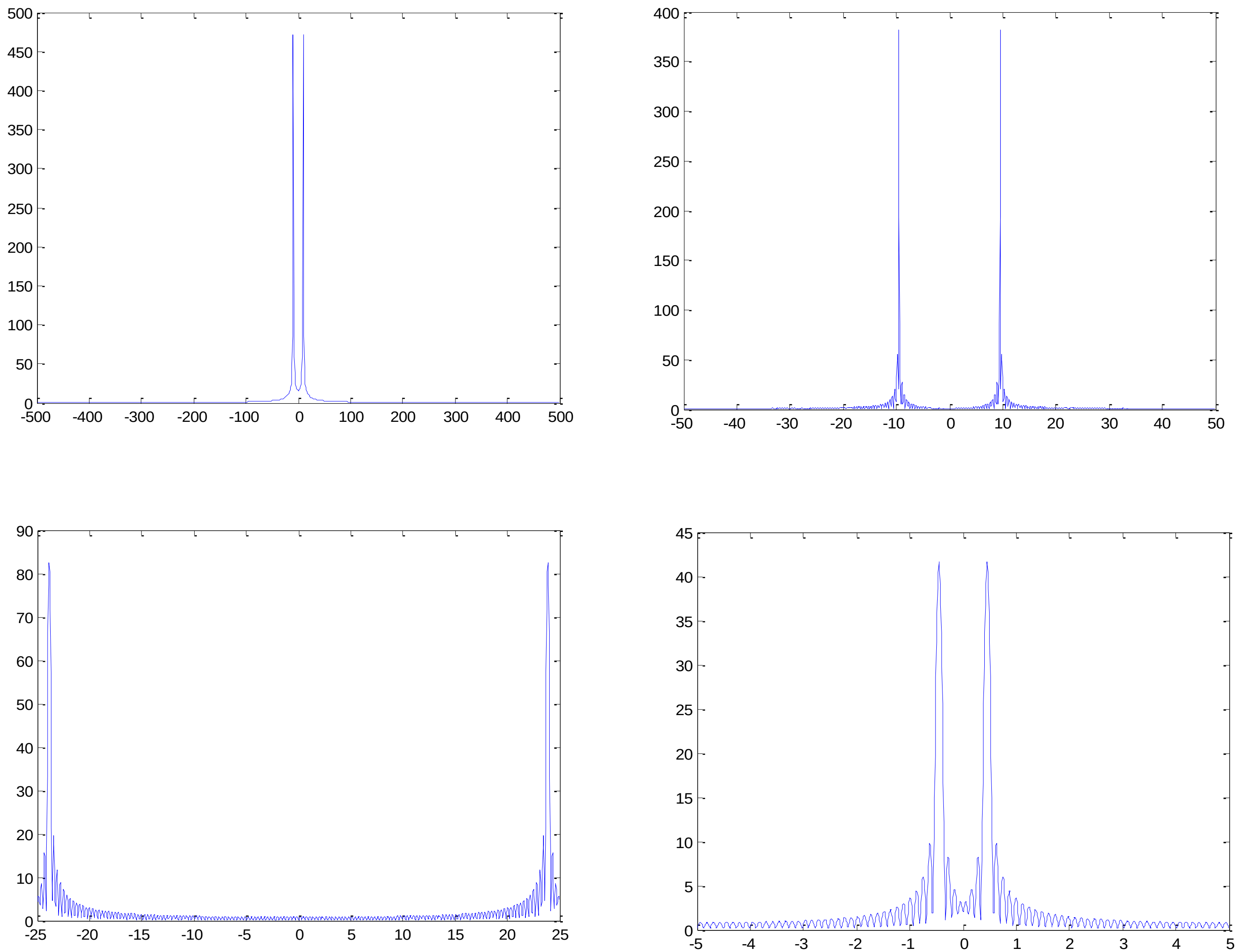
The Nyquist Sampling Theorem states that to avoid aliasing occuring in the sampling of a signal the samping rate should be greater than or equal to twice the highest frequency present in the signal. This is refered to as the Nyquist sampling rate.

(a) Sampled Data



Important topics in Discrete Fourier Transform (I): Aliasing

(b) Frequency contents



(c) Matlab Program

```
% aliasing.m
% This m-file illustrates the effects of aliasing in time and frequency domains
%
% Four time bases, at four sampling rates.
t1 = 0:.001:8;
t2 = 0:.01:8;
t3 = 0:.05:8;
t4 = 0:.1:8;

% Four samples of a cosine at 60 rad/sec
s1 = cos(60*t1);
s2 = cos(60*t2);
s3 = cos(60*t3);
s4 = cos(60*t4);

% Plots of the four samples
subplot(221);  plot(t1,s1); axis([0 1 -1 1]); title('.001 s sampling rate');
subplot(222);  plot(t2,s2); axis([0 1 -1 1]); title('.01 s sampling rate');
subplot(223);  plot(t3,s3); axis([0 1 -1 1]); title('.05 s sampling rate');
subplot(224);  plot(t4,s4); axis([0 1 -1 1]); title('.1 s sampling rate');
clc
```

Important topics in Discrete Fourier Transform (I): Aliasing

```
fprintf('Results show the effect of different sampling frequencies\n');
fprintf('and aliasing.\n\nThe sampled function is cos(omega*t)\n');
fprintf('where omega is 60 rad/sec\n\n');
fprintf('Note that if the time between samples is more than half the period of\n');
fprintf('the sampled function then an error occurs in the sampling.\n');
fprintf('This error is called aliasing.\n\n');
fprintf('In other words, to avoid aliasing when sampling a signal,\n');
fprintf('the sampling frequency must be greater than twice the highest\n');
fprintf('frequency component in the signal. This is the Nyquist Sampling
  Theorem.\n\n');
fprintf('Press any key to continue.\n\n\n\n');
pause

 clf
 clc

 fprintf('Aliasing effects are also evident in the frequency spectrum\n\n');
 fprintf('If the sampling frequency is less than the Nyquist frequency then the\n');
 fprintf('frequency components higher than the Nyquist frequency appear
   erroneously\n');
 fprintf('as lower frequencies.\n\n');
 fprintf('The higher frequencies "fold over" around the Nyquist frequency\n\n\n');
 fprintf('This series of spectrum plots illustrate the effect of changing the\n');
 fprintf('sampling rate on the frequency domain representation of the signal\n\n');
```

Important topics in Discrete Fourier Transform (I): Aliasing

```
S1 = fftshift(abs(fft(s1,1024)));
S2 = fftshift(abs(fft(s2,1024)));
S3 = fftshift(abs(fft(s3,1024)));
S4 = fftshift(abs(fft(s4,1024)));

w1 = -500:1000/1024:500-1/1024;
plot(w1,S1);

fprintf('Frequency spectrum: First sampling rate.\n');
fprintf('Peak is at 60 rad/sec.\n\n');

fprintf('Press any key to continue.\n\n');
pause
w2 = -50:100/1024:50-1/1024;
plot(w2,S2);
fprintf('Frequency spectrum: Sampling rate halved.\n');
fprintf('Peak is still at 60 rad/sec.\n\n');
fprintf('Press any key to continue.\n\n');
pause

w3 = -25:50/1024:25-1/1024;
plot(w3,S3);
fprintf('Frequency spectrum: Sampling rate again halved.\n');
fprintf('Peak is still at 60 rad/sec, but it''s close to the edge.\n\n');
fprintf('Press any key to continue.\n\n');
pause
```

Important topics in Discrete Fourier Transform (I): Aliasing

```
w4 = -5:10/1024:5-1/1024;
plot(w4,S4);
fprintf('Frequency spectrum: Sampling rate again halved.\n');
fprintf('60 rad/sec peak now appears to be much lower. Aliasing has
  occurred.\n\n');
```

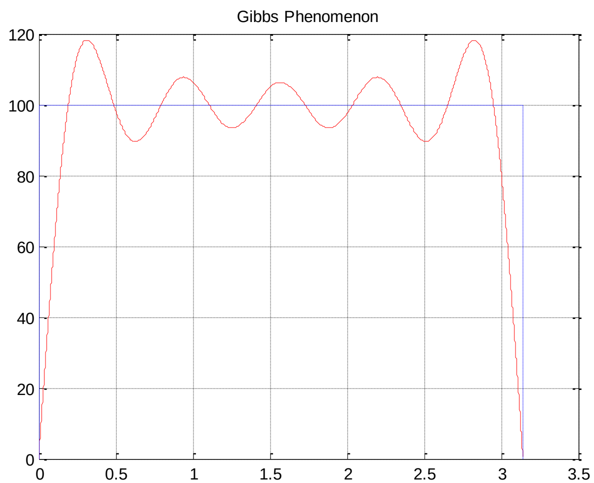
2. Gibbs Phenomena for Partial Fourier Transform for f(t) = 100 (0<t<3.14159 (pi))

With varying number of harmonics

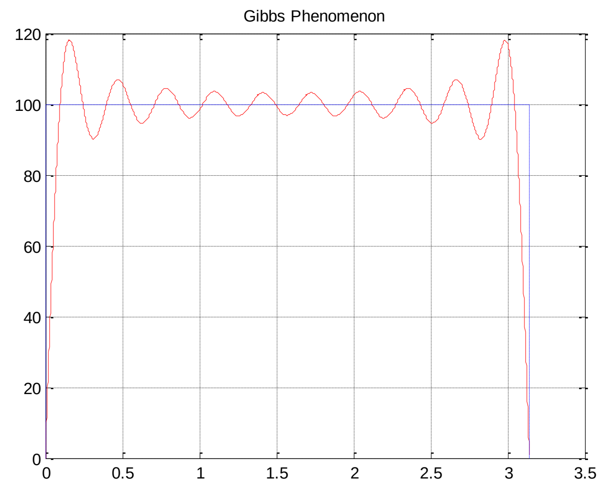
One shortcoming of Fourier series today known as the Gibbs phenomenon was first observed by H. Wilbraham in 1848 and then analyzed in detail by Josiah W. Gibbs (1839-1903)

(a) Phenomena

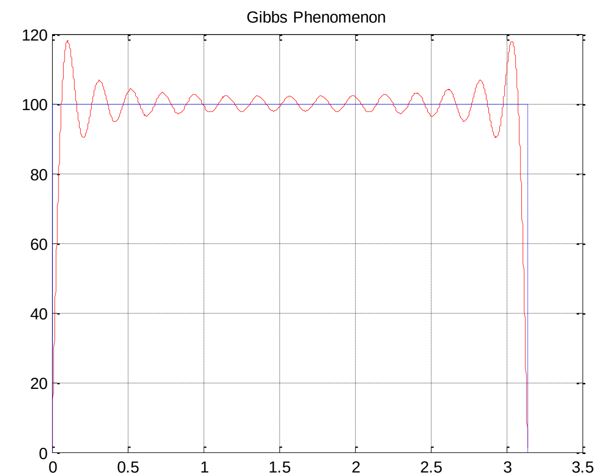
N = 5



N =10

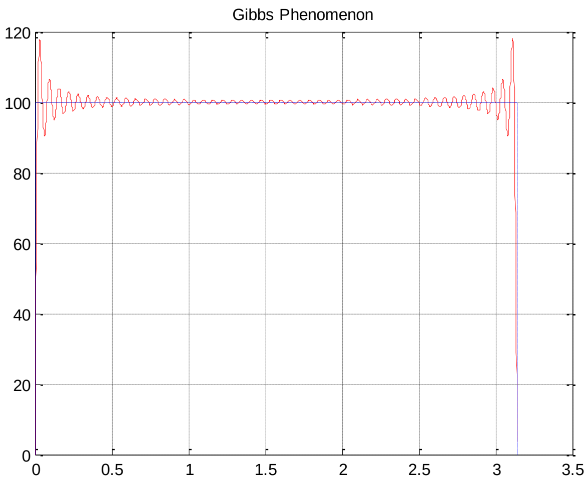


N = 15

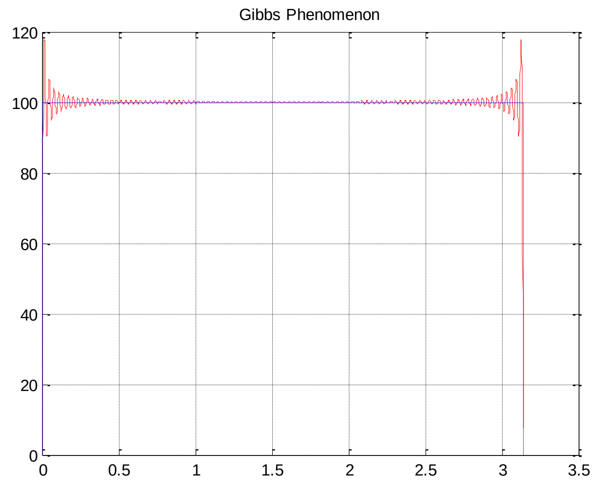


Important topics in Discrete Fourier Transform (II): Gibbs Phenomena

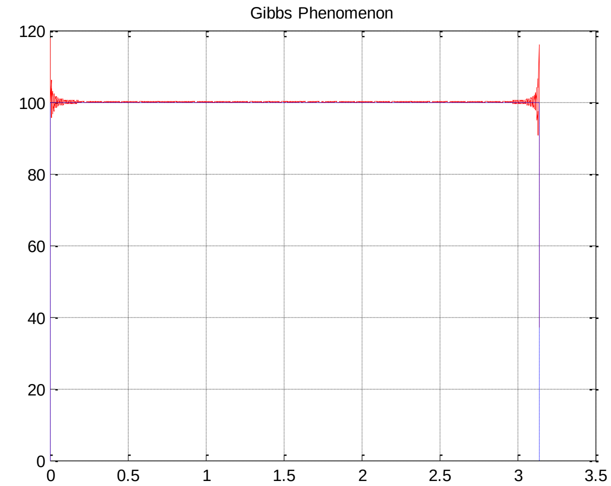
N = 50



N = 100



N = 500



Even with large number of harmonic terms, Oscillatory behavior never disappears

Window concept can be used to reduced high frequency content of oscillations

(b) Matlab Program

```
%function gibbs(n)
% An illustration of Gibbs phenomenon.
% This function computes the partial Fourier series sum of a square wave, to illustrate the peaks that
%  occur  at jump discontinuities when using the Fourier series.
%
% The function plots a square wave. and asks the user for the number of terms to use in the Fourier
%  series sum. The partial Fourier series approximation is then superimposed upon the square wave.
%
% variable n is the number of terms to use in the partial sum.
% create the square wave
dt = 0.001;
t = 0:dt:pi;
f = 100 * ones(size(t));
f(1) = 0;
f(length(f)) = 0;

%create the partial Fourier series approximation
s = zeros(size(t));
for i = 1:n
  s = s + 1/(2*i - 1)*sin((2*i - 1) * pi * t / pi);
end
s = 400 / pi * s;
 %plot the approximation and the square wave
plot(t, s, '-r', t, f, '--b');grid
 title('Gibbs Phenomenon');
```

목 차

3 Windowing ideal reconstruction filters

The windows investigated in this work are

- The rectangular window or box function

$$\Pi_\tau(x) = \begin{cases} 1 & \text{if } |x| \leq \tau \\ 0 & \text{else} \end{cases} \tag{3}$$

  which performs pure truncation.

- The Bartlett window, which is actually just a tent function:

$$\text{Bartlett}_\tau(x) = \begin{cases} 1 - \frac{|x|}{\tau} & \text{if } |x| < \tau \\ 0 & \text{else} \end{cases} \tag{4}$$

- The Welch window:

$$\text{Welch}_\tau(x) = \begin{cases} 1 - \left(\frac{x}{\tau}\right)^2 & |x| < \tau \\ 0 & \text{else} \end{cases} \tag{5}$$

- The Parzen window

$$\text{Parzen}(x) = \frac{1}{4} \begin{cases} 4 - 6|x|^2 + 3|x|^3 & 0 \leq |x| < 1 \\ (2 - |x|)^3 & 1 \leq |x| < 2 \\ 0 & \text{else} \end{cases} \tag{6}$$

Important topics in Discrete Fourier Transform (III): Windowing

is a piece-wise cubic approximation of the Gaussian window with extend two. Although its width is not directly adjustable it can, of course, be scaled to every desired extend.

- The Hann window (due to Julius van Hann, often wrongly referred to as Hanning window [17], sometimes just cosine bell window) and Hamming window, which are quite similar, they only differ in the choice of one parameter α:

$$\mathrm{H}_{\tau,\alpha}(x) = \begin{cases} \alpha + (1-\alpha)\cos(\pi\frac{x}{\tau}) & |x| < \tau \\ 0 & \text{else} \end{cases} \tag{7}$$

with $\alpha = \frac{1}{2}$ being the Hann window and $\alpha = 0.54$ the Hamming Window.

- The Blackman window, which has one additional cosine term as compared to the Hann and Hamming window.

$$\mathrm{Blackman}_{\tau}(x) = \begin{cases} 0.42 + \frac{1}{2}\cos(\pi\frac{x}{\tau}) + \\ 0.08\cos(2\pi\frac{x}{\tau}) & |x| < \tau \\ 0 & \text{else} \end{cases} \tag{8}$$

- The Lanczos window, which is the central lobe of a sinc function scaled to a certain extend.

$$\mathrm{Lanczos}_{\tau}(x) = \begin{cases} \frac{\sin(\pi\frac{x}{\tau})}{\pi\frac{x}{\tau}} & |x| < \tau \\ 0 & \text{else} \end{cases} \tag{9}$$

Important topics in Discrete Fourier Transform (III): Windowing

- The Kaiser window [3], which has an adjustable parameter α which controls how steeply it approaches zero at the edges. It is defined by

$$\mathrm{Kaiser}_{\tau,\alpha}(x) = \begin{cases} \frac{I_0(\alpha\sqrt{1-(x/\tau)^2})}{I_0(\alpha)} & |x| \leq \tau \\ 0 & \text{else} \end{cases} \tag{10}$$

where $I_0(x)$ is the zeroth order modified Bessel function [13]. The higher α gets the narrower becomes the Kaiser window.

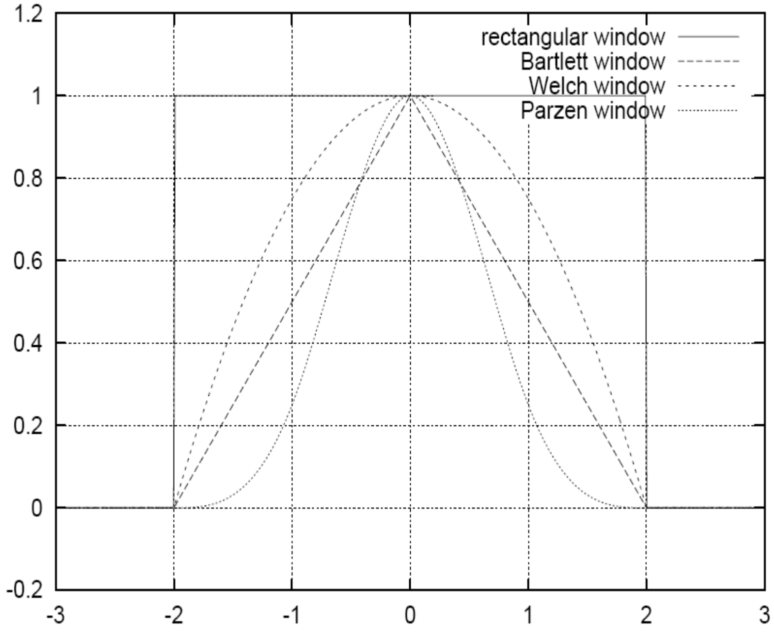
- and we can also use a truncated Gaussian function as window. It is in its general form defined by

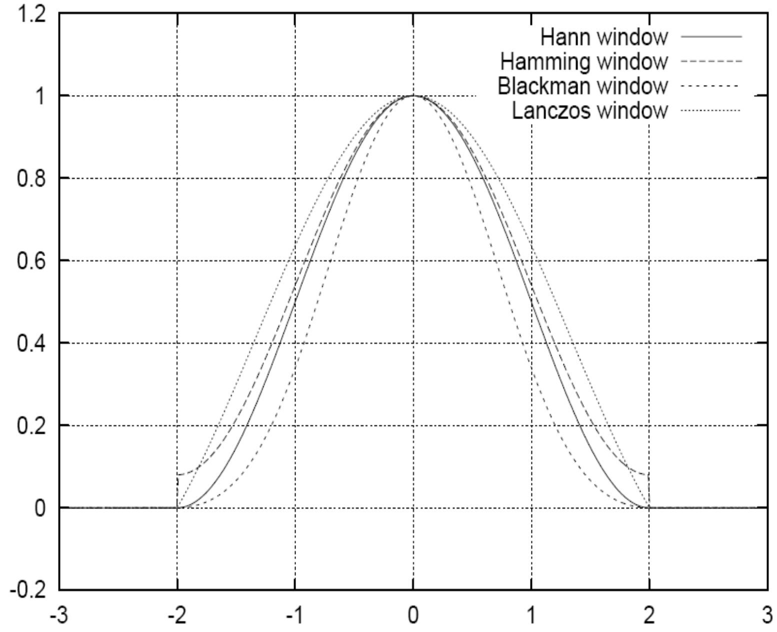
$$\mathrm{Gauss}_{\tau,\sigma}(x) = \begin{cases} 2^{-\left(\frac{x}{\sigma}\right)^2} & |x| < \tau \\ 0 & \text{else} \end{cases} \tag{11}$$

with σ being the standard deviation. The higher σ gets, the wider the Gaussian window becomes and, on the other hand, the more severe gets the truncation.

All these windows, except Kaiser and Gaussian windows, are depicted in Fig. 2 on top, the frequency responses of correspondingly windowed sinc (with window width two) in the middle row and windowed cosc in the bottom row. Since function reconstruction filters are even functions and first derivative filters are odd functions, the power spectra, as

Important topics in Discrete Fourier Transform (III): Windowing





spatial domain

Important topics in Discrete Fourier Transform (III): Windowing

frequency domain

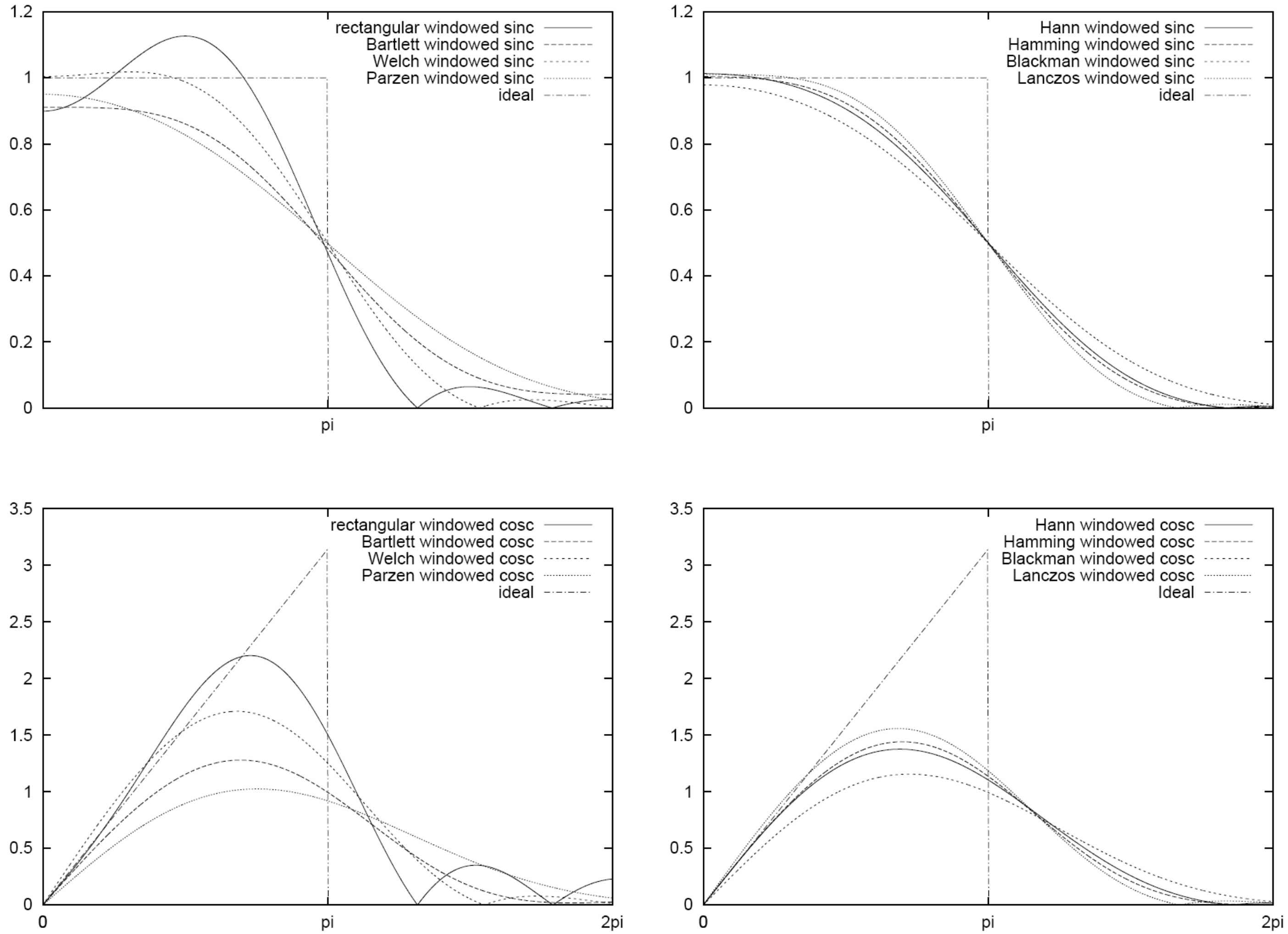


Figure 2: Rectangular, Bartlett, Welch, Parzen, Hann, Hamming, Blackman and Lanczos windows of width two on top, below the frequency responses of correspondingly windowed sinc and cosc functions.

Important topics in Discrete Fourier Transform (III): Windowing

Useful Information at signal processing toolbox at MATLAB

>>help signal

>> help window

>> help fdesing

See also bartlett, barthannwin, blackman, blackmanharris, bohmanwin,

chebwin, gausswin, hamming, hann, kaiser, nuttallwin, parzenwin,

rectwin, triang, tukeywin, wintool.

=======================================

>>d = fdesign.nyquist(5,'n',150); hd = window(d,hamming(151)); fvtool(hd)

목 차

FFT example of a signal: MATLAB Program (for f(t) = cos(3*2*pi*t) + cos(1*2*pi*t);

```
% fft_ex.m
% This is an example of computing the FFT of the addition of two cos functions.
%
clc
fprintf('***********************************************************\n\n');
fprintf('This file illustrates how to perform the FFT on a time\n');
fprintf('signal using MATLAB\n\n\n');
fprintf('In this case, the time signal is the sum of two cosines\n\n');

echo on
% Sample at 10 Hz, and use a 500 second long sample -> 5001 samples
t = 0:.1:500;
% Calculate the sum of the two cosines
x = cos(3*2*pi*t) + cos(1*2*pi*t);

% Take an 8192-point FFT (power of 2 greater than 5000)
f = fft(x,8192);

% Construct a frequency axis
Freq = -5:10/8192:5-1/8192;

% Plot frequency, magnitude.  fftshift centers around zero.
plot(Freq, abs(fftshift(f)));

echo off
```

End of Lecture