ELEC-A7200

Signals and Systems

Professor Riku Jäntti
Fall 2021

Aalto University
School of Electrical
Engineering

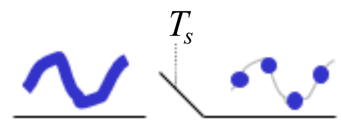
Lecture 7
Sampling and DFT

Content

- **Sampling**
- **Discrete Time Fourier Transform**
- **Discrete Fourier Transform & Fast Fourier Transform**

Sampling

In signal processing, *sampling* is the reduction of a continuous-time signal to a discrete-time signal.

$x(t), t \geq 0$  $x(kTs),\ k=0,1,2,..$

- T_s **denotes sampling interval**
- $f_s=1/T_s$ **denotes sampling frequency**

Nyquist sampling theorem

A bandlimited continuous-time signal $x(t)$ having bandwidth B can be sampled and perfectly reconstructed from its samples $x(kT_s)$ if the waveform is sampled with rate

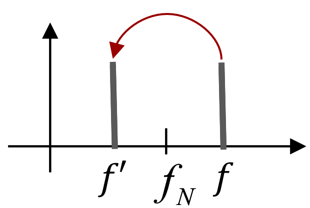
$$f_s = \frac{1}{T_s} > 2B.$$

The minimum sampling rate f_s that produces a signal that still contains all of the original signal's information is known as the Nyquist rate (a.k.a. Nyquist limit frequency)

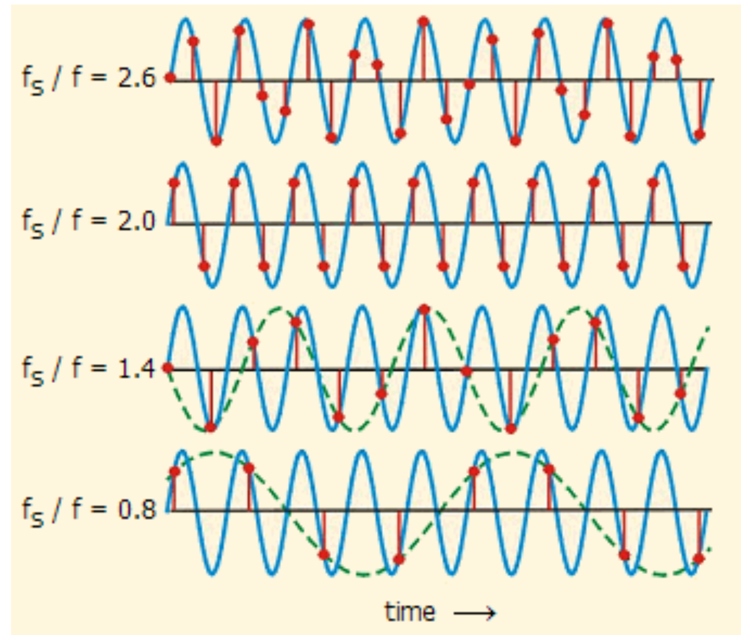
$$f_N = \frac{f_s}{2}$$

Aliasing

Any sinusoidal component of the signal of frequency f higher than f_N is not only lost, but it is reintroduced in the sampled signal by folding at frequency f_N as an **alias** (false name) sinusoidal component of frequency f'



$$f' = |f - k\,fs|, k = 1,2,3, \ldots$$
$$0 \leq f' \leq f_N$$



Aliasing example

A time domain signal

$$x(t) = A_0 \cos(2\pi f_0 t + \theta_0) + A_1 \cos(2\pi f_1 t + \theta_1)$$

f_0=50 Hz
f_1=100 Hz

is sampled using sampling frequency f_s=180 Hz.

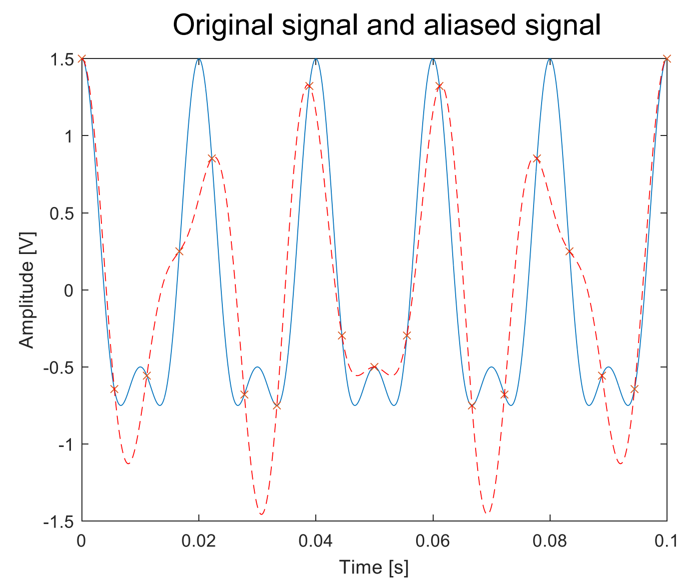
What frequencies are present in the sampled signal?

Answer:

$$f_N = \frac{f_s}{2} = \frac{180\text{ Hz}}{2} = 90\text{ Hz}$$

f_0=50 Hz $< f_N \Rightarrow f_0$=50 Hz will be present without folding

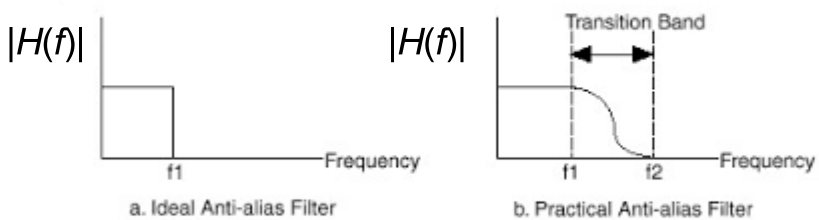
f_1=100 Hz $> f_1'$= |100 Hz – 180 Hz| = 80 Hz



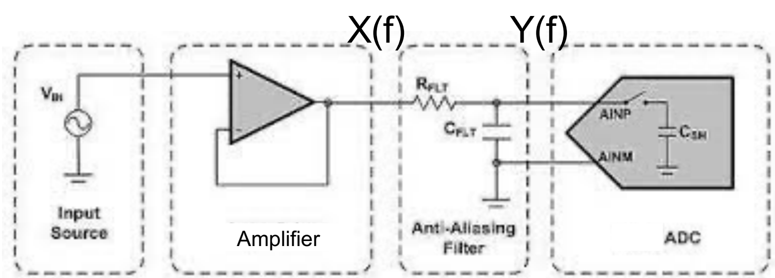
Aalto University
School of Electrical Engineering

Anti-aliasing filter

An anti-aliasing filter (AAF) is a filter used before a signal sampler to restrict the bandwidth of a signal to satisfy the Nyquist–Shannon sampling theorem over the band of interest.



$$Y(f) = H(f)X(f)$$



$H(f)$

Ideal sampling

Multiplication with Dirac's delta function samples a signal

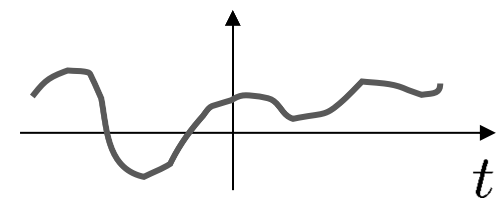
$$x(t)\delta(t - nT_s) = x(nT_s)\delta(t - nT_s)$$

$$\int_{-\infty}^{\infty} x(t)\, \delta(t - nT_s)dt = x(nT_s)$$

Ideal sampling

- **Continuous time signal**

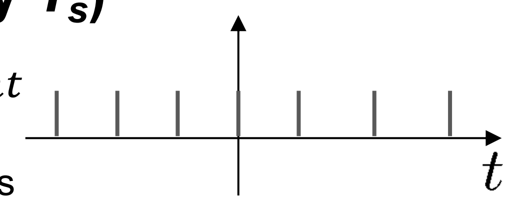
  $x(t)$



- **Sampling signal (periodic with periodicity T_s)**

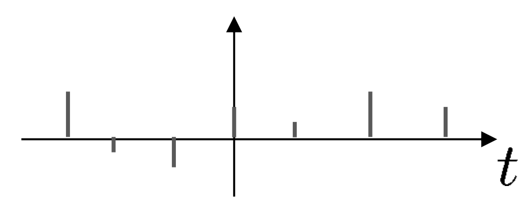
  $$s(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT_s) = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} e^{j\frac{2\pi}{T_s}nt}$$

  Exponential Fourier series



- **Sampled signal**

  $x_s(t) = x(t)s(t)$



Ideal sampling

Time domain signals

$x(t)$

$$x_s(t) = x(t)s(t)$$

$$= \sum_{k=-\infty}^{\infty} x(t)\delta(t - kT_s)$$

$$= x(t)\frac{1}{T_s}\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi}{T_s}nt}$$

Frequency domain signals

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft}dt$$

Fourier transform

$$X_s(f) = X(f) \otimes S(f)$$

Convolution

$$= \sum_{n=-\infty}^{\infty} x(nT_s)e^{-j2\pi fkT_s}$$

Discrete Time Fourier transform

$$= \frac{1}{T_s}\sum_{n=-\infty}^{\infty} X\left(f - \frac{n}{T_s}\right)$$

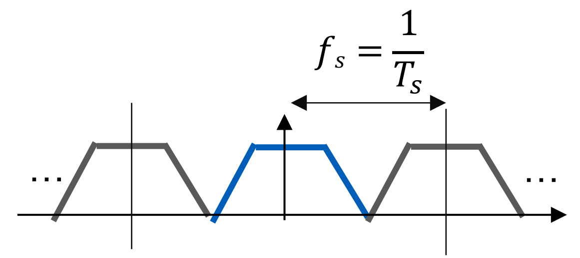
Discrete time Fourier transform (DTFT)

Discrete Time Fourier Transform (DTFT)

$$\text{DTFT}[\{x(kT_s), \ldots, -1, k, 1, ..\}] = \sum_{n=-\infty}^{\infty} x(nT_s) e^{-j2\pi f n T_s}$$

Poisson's sum formula $$\sum_{n=-\infty}^{\infty} x(nT_s) e^{-j2\pi f n T_s} = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} X\left(f - \frac{n}{T_s}\right)$$

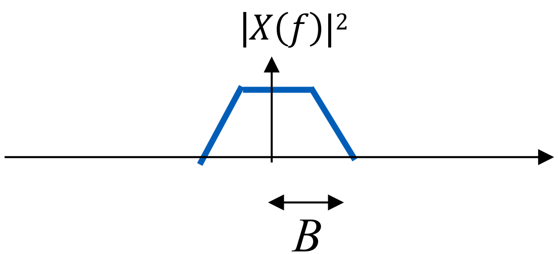
DTFT is periodic in frequency domain



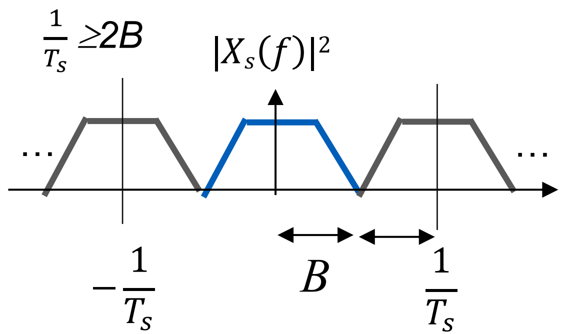
Discrete time Fourier transform (DTFT)

Fourier transform of the sampled signal

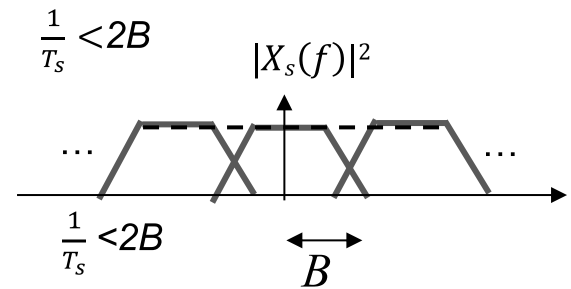
$$X_s(f) = \sum_{n=-\infty}^{\infty} x(nT_s)e^{-j2\pi f n T_s} = \frac{1}{T_s}\sum_{n=-\infty}^{\infty} X\left(f - \frac{n}{T_s}\right)$$



Spectrum of the original continuous time signal



Original signal can be Reconstructed from the sampled signal



Original signal cannot be reconstructed from the sampled signal.
Aliasing happens!

DTFT example

Sinusoidal signal

$$x(t) = A\cos(2\pi f_0 t) \Leftrightarrow \frac{A}{2}\left(\delta(f+f_0) + \delta(f-f_0)\right)$$

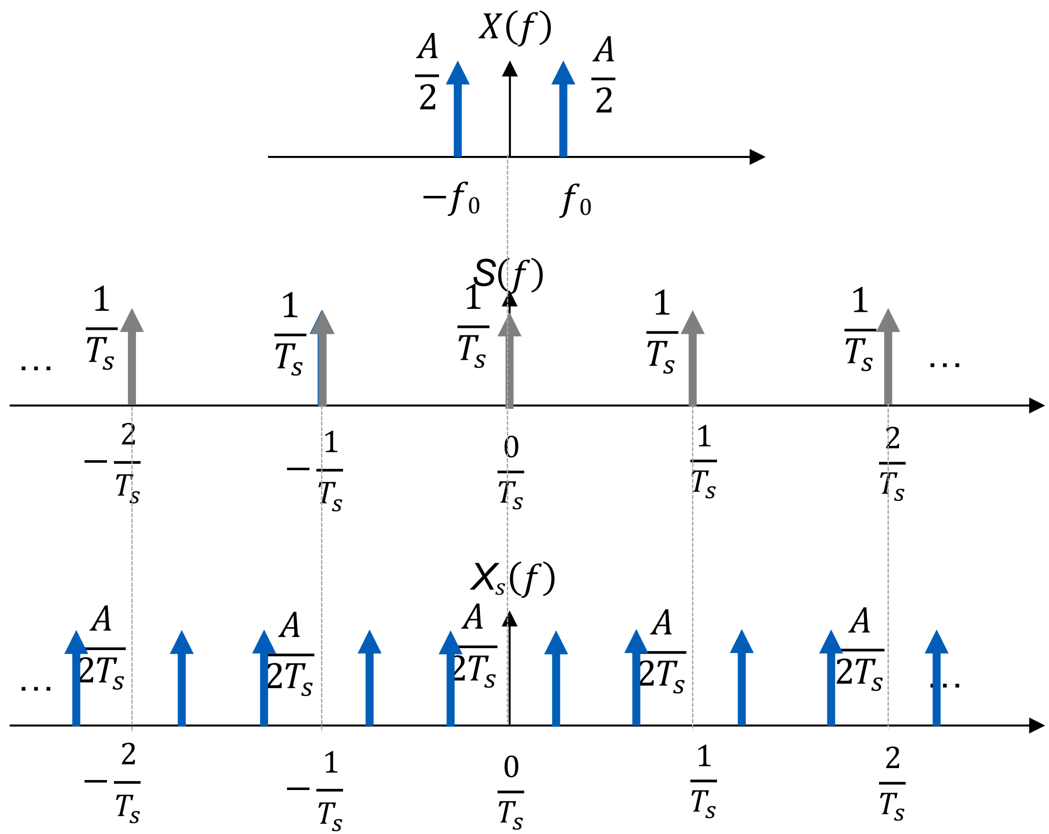
Sampling signal

$$s(t) = \frac{1}{T_s}\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi}{T_s}nt} \Leftrightarrow \frac{1}{T_s}\sum_{n=-\infty}^{\infty} \delta\left(f - \frac{1}{T_s}n\right)$$

Sampled signal

$$x_s(t) = x(t) \Leftrightarrow X(f) \otimes S(f)$$

Convolution



Discrete Fourier Transform DFT

DFT transforms a sequence of complex numers $\{x_0, x_1, x_2 \ldots, x_{N-1}\}$ into another sequence of complex numbers $\{X(0), X(1), X(2) \ldots, X(N-1)\}$

$$X(k) = \sum_{n=0}^{N-1} x_n e^{\frac{-j2\pi}{N}kn}$$

Inverse Discrete Fourier Transform

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X(k) e^{\frac{j2\pi}{N}kn}$$

DFT

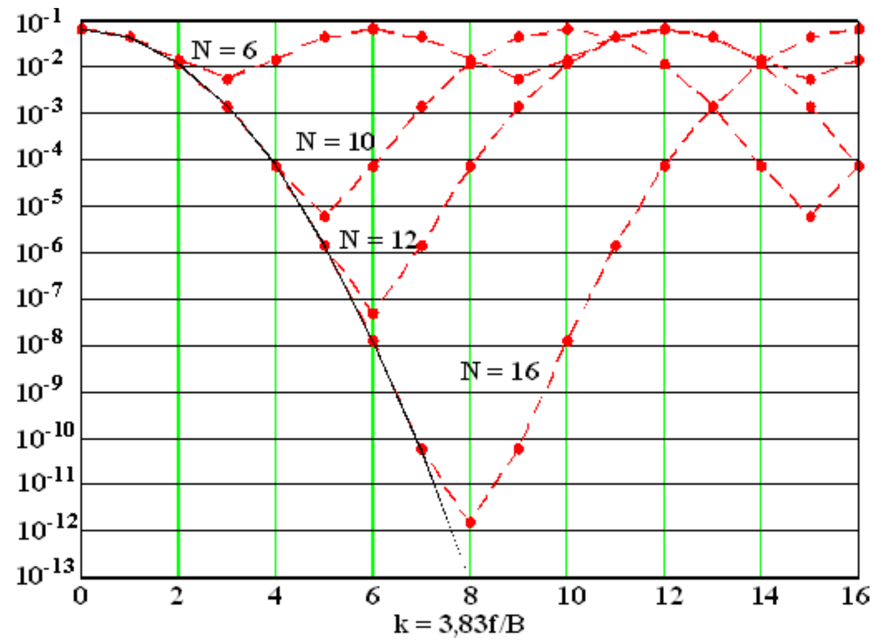
DFT is periodic

$$X(k+N) = X(k)$$

and assumes also the discrete sequence to be periodic

$$x_{n+N} = x_n$$

Fourier transform of a Gauss pulse



Fast Fourier Transform

Fast Fourier Transform (FFT) is a computationally efficient method to calculate DFT.

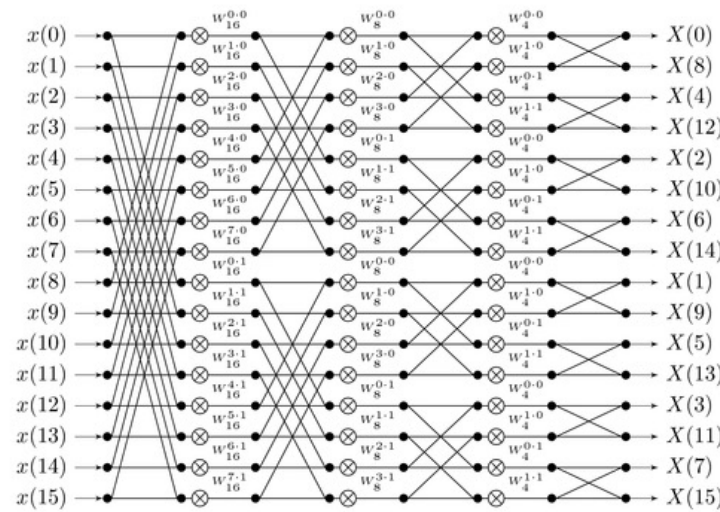
- **DFT is used as defined has complexity $O(N^2)$**
- **FFT has complexity $O(N\log(N))$**

FFT libraries are available for (almost) all programming languages

Matlab: `fft(X,N,DIM)`
Phyton NumPy: `fft(a[, n, axis, norm])`

Similarly to FFT, there also exist Inverse Fast Fourier Transform

16 point FFT



$$W_N = e^{\frac{-j2\pi}{N}}$$

DFT vs DTFT

Consider N samples of a continuous time signal $\{x(nT_s), n = 0,1, \dots N\}$

$X_s(f)$=DTFT$[x(kT_s), k = 0,1, \dots, N-1]$ = $\sum_{k=0}^{N-1} x(nT_s) e^{-j2\pi f T_s n}$

DFT of the samples

Fourier transform and DFT of a Gaussian pulse

$X(k)$=DFT$[x(kT_s), k = 0,1, \dots, N-1]$ = $\sum_{k=0}^{N-1} x(nT_s) e^{\frac{-j2\pi}{N} kn}$

Equivalent if

$$fT_s = \frac{k}{N} \Rightarrow f = \frac{k}{N}\frac{1}{T_s} = \frac{k}{N} f_s$$

We can use DFT (FFT)
to calculate DTFT!

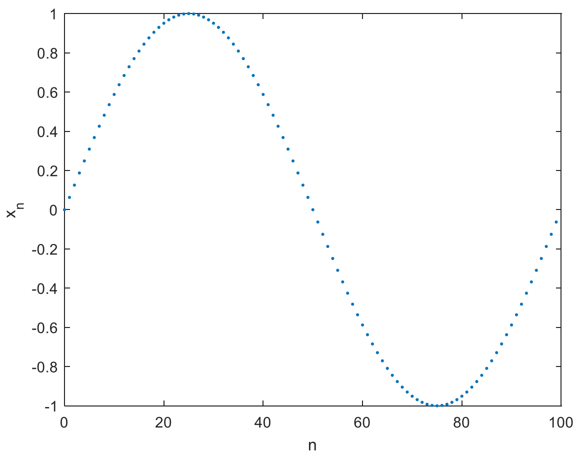
Frequency granularity $\Delta f = \frac{1}{N} f_s$

Parseval's theorem

Parseval's theorem for DFT

$$\sum_{n=0}^{N-1} |x_n|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

Example: Sinusoid



$$x_n = \sin\left(\frac{2\pi n}{N}\right)$$

$$X(k) = \begin{cases} -j\frac{N}{2}, k = 1 \\ j\frac{N}{2}, k = N \\ 0, \text{otherwise} \end{cases}$$

$$\sum_{n=0}^{N-1} |x_n|^2 = \frac{N}{2}$$

$$\sum_{k=0}^{N-1} |X(k)|^2 = \frac{N^2}{4} + \frac{N^2}{4} = \frac{N^2}{2}$$

DFT vs Fourier transform

Fourier transform of a pulse defined on an interval [0, *T*]

$$X(f) = \int_0^T x(t)e^{-j2\pi f t} \approx T_s \underbrace{\sum_{k=0}^{N-1} x(nT_s)e^{\frac{-j2\pi}{N}kn}}_{\text{DFT}}$$

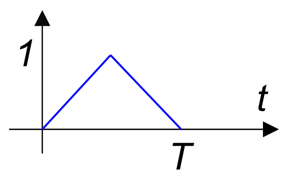
Euler numerical integration (T_s)

$$T = (N-1)T_s$$

$$f = \frac{k}{N} f_s$$

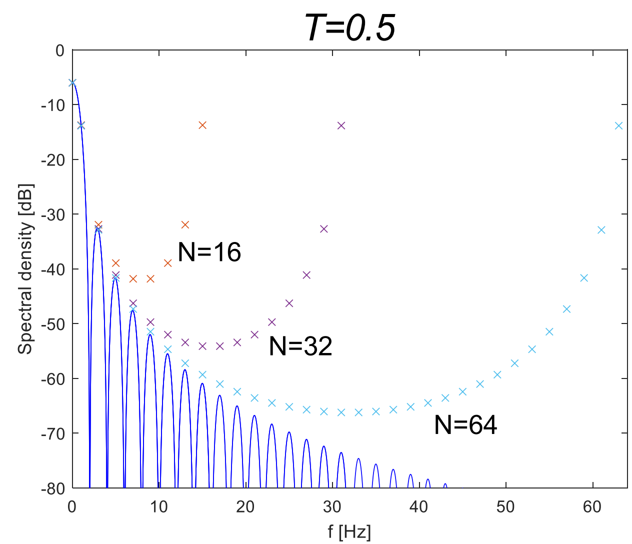
Triangle pulse

$$x(t) = \mathrm{tria}\left(\frac{t}{T/2} - 1\right)$$



Spectrum

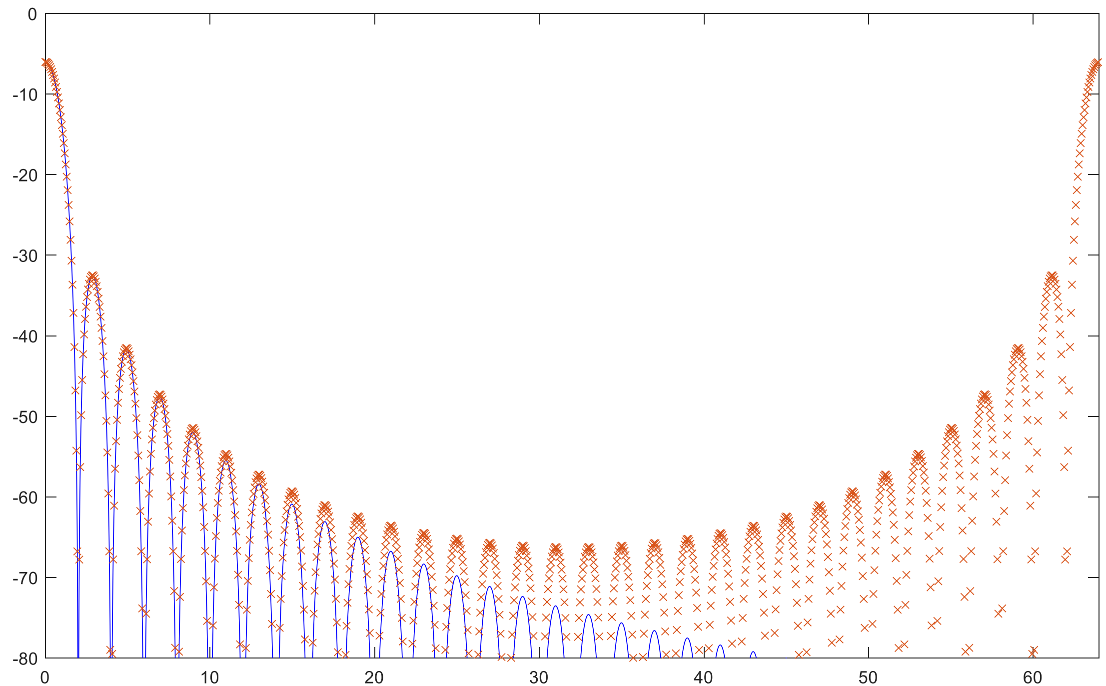
$$|X(f)|^2 = \frac{T^2}{4}\mathrm{sinc}^4\left(\frac{fT}{2}\right)$$



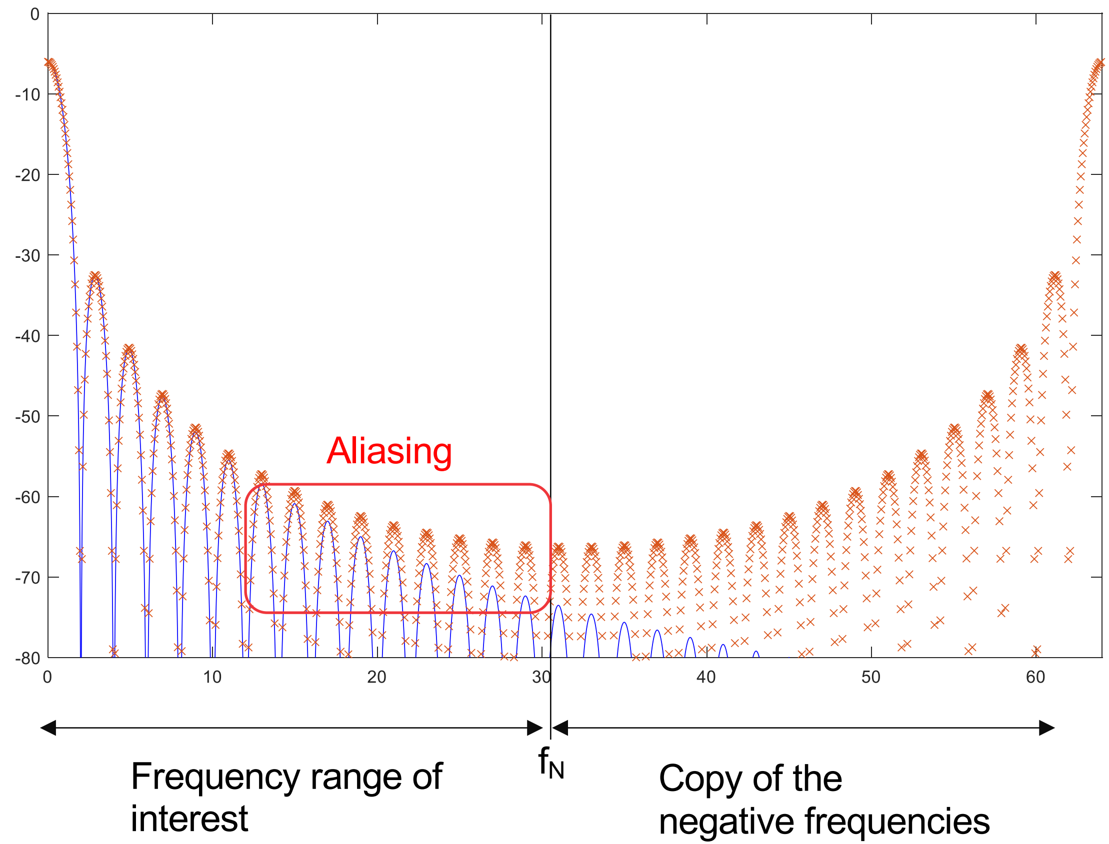
Zero padding

Adding N_z zeros in the end of the sequence makes DFT to interpolate more frequencies. The frequency granularity becomes

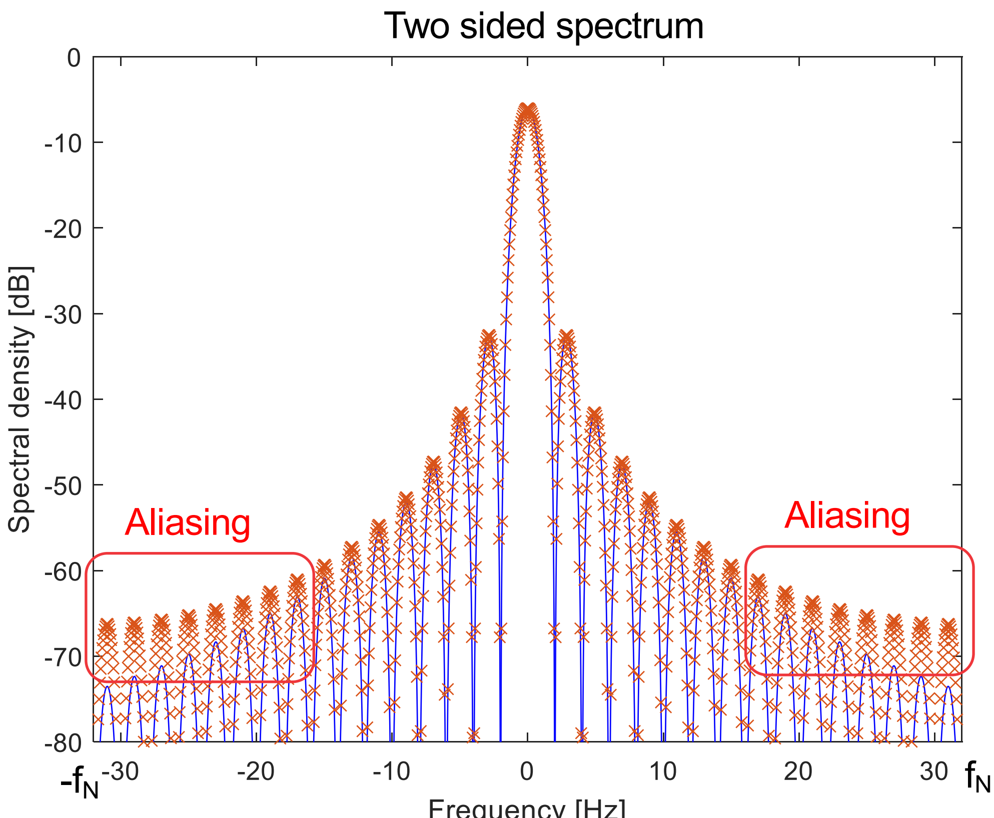
$$\Delta f = \frac{1}{N + N_z} f_s$$



Spectral density



Spectral density



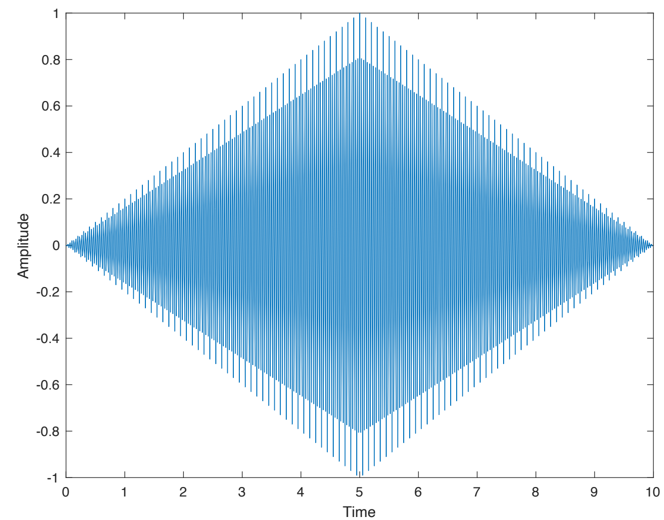
For better estimate of the spectrum we should apply anti-aliasing filter before sampling

Modulated signal

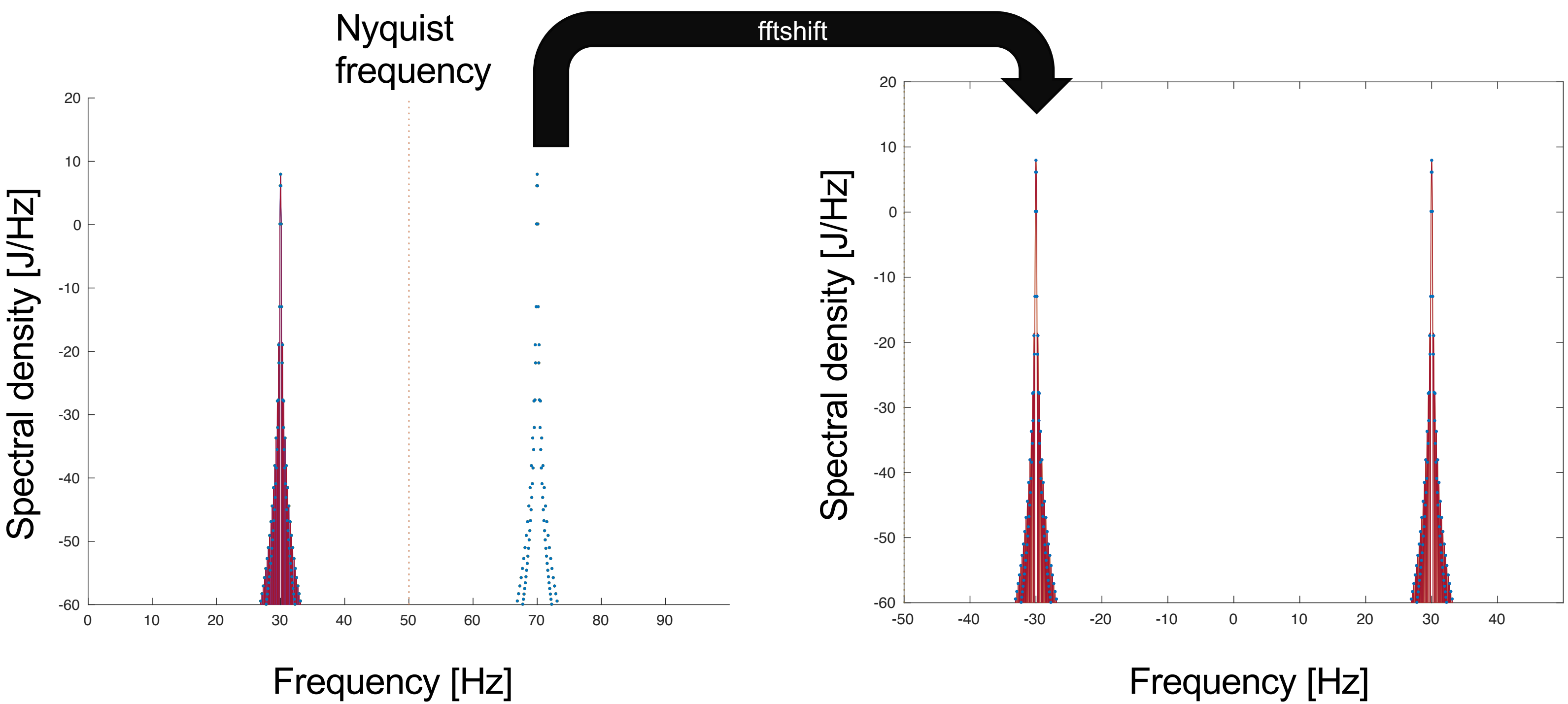
Example: Modulated tria pulse

$$x(t) = \text{tria}\left(\frac{t - \frac{1}{2}T}{T}\right)\cos(2\pi f_c t)$$

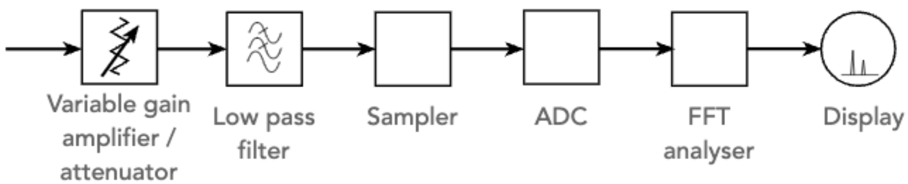
- Pulse width T=10s
- Carrier Frequency fc=30 Hz
- Sampling frequency fs=100 Hz



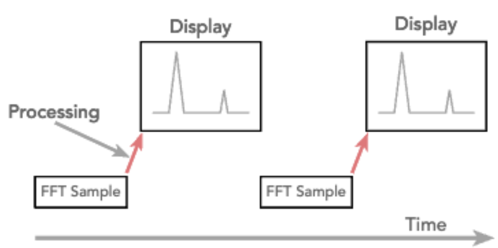
Spectral density

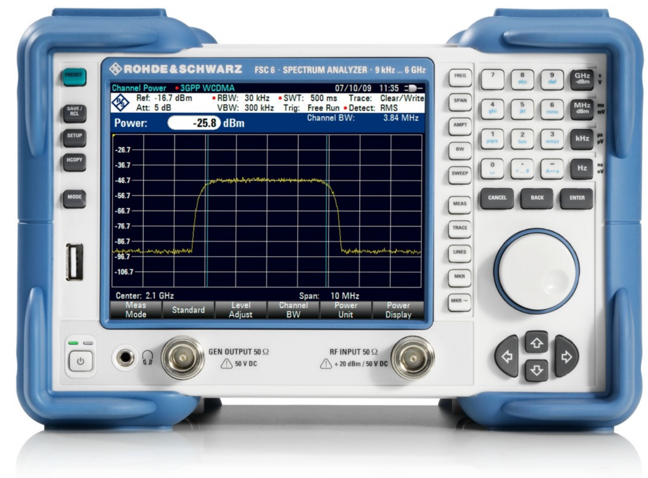


FFT based spectrum analyzer



FFT Spectrum Analyser Block Diagram





Periodic convolution

Periodic convolution is defined for periodic sequences

$$x_n = x_{n+N}$$

$$\ldots \{x_0, x_1, x_2, \ldots, x_{N-1}\} \{x_0, x_1, x_2, \ldots, x_{N-1}\} \{x_0, x_1, x_2, \ldots, x_{N-1}\} \ldots$$

-N,-N+1,-N+2,…,-1, 0,1,2 …., N-1, N, N+1,N+2,…2N-1,…

$$y_n = h_n \overset{P}{\oplus} x_n = \sum_{m=o}^{N-1} h_m x_{n-m}$$

DFT of periodic convolution

$$Y(k) = \text{DFT}[h_n \overset{P}{\oplus} x_n] = H(k)X(k)$$

Discrete linear convolution

Discrete linear convolution between pulse sequences h_n **and** x_n

1. Add zeros

$$h_{a,n} = \begin{cases} h_n & n = 0,1,\dots,N_h - 1 \\ 0 & n = N_h + 1, N_h + 2, \dots, N_h + N_x - 1 \end{cases}$$

$$x_{a,n} = \begin{cases} x_n & n = 0,1,\dots,N_x - 1 \\ 0 & n = N_x + 1, N_x + 2, \dots, N_h + N_x - 1 \end{cases}$$

2. Calculate the sum

$$y_n = \sum_{m=0}^{N-1} h_{a,n} x_{a,(m-n) \bmod N} \qquad N = N_h + N_x - 1$$

Discrete linear convolution using FFT and IFFT

Discrete linear convolution between pulse sequences h_n **and** x_n

1. Add zeros

$$h_{a,n} = \begin{cases} h_n & n = 0,1, \ldots, N_h - 1 \\ 0 & n = N_h + 1, N_h + 2, \ldots, N_h + N_x - 1 \end{cases}$$

$$x_{a,n} = \begin{cases} x_n & n = 0,1, \ldots, N_x - 1 \\ 0 & n = N_x + 1, N_x + 2, \ldots, N_h + N_x - 1 \end{cases}$$

2. Calulate $H(k) = \mathrm{FFT}[h_{a,n}]$ and $X(\mathrm{k}) = \mathrm{FFT}[x_{a,n}]$

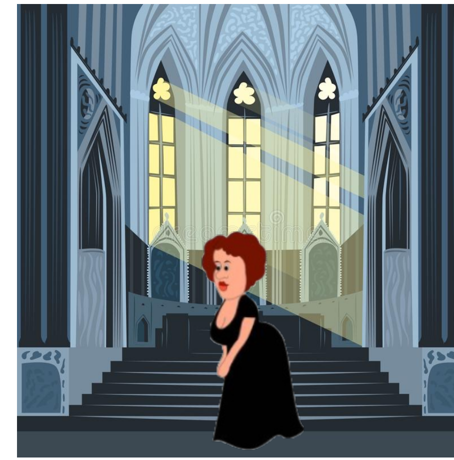
3. Obtain the convolution by applying IFFT on $Y(k)=H(k)X(\mathrm{k})$ to obtain

$$y_n = IFFT[Y(k)]$$

Discrete linear convolution example

$\otimes$

=



Singing in a studio

Time domain sound signal x_n
Zero padded signal $x_{a,n}$
Frequency domain signal
$X(k)=FFT[x_{a,n}]$

Impulse response model of the church hall

Acoustic impulse response h_n
Zero signal $h_{a,n}$
Frequency response:
$H(k)=FFT[h_{a,n}]$

Singing in the church

Time domain sound signal
$y_n = x_{a,n} \otimes h_{a,n} = IFFT[\ X(k) \cdot H(k)\ ]$

Aalto University
School of Electrical
Engineering