

# Discrete Fourier Transform and filters

Eugeniy E. Mikhailov

The College of William & Mary



Lecture 26

# DFT vs. Matlab FFT

## DFT

$$y_k = \frac{1}{N} \sum_{n=0}^{N-1} c_n \exp(i \frac{2\pi(k-1)n}{N}) \quad \text{inverse Fourier transform}$$

$$c_n = \sum_{k=1}^N y_k \exp(-i \frac{2\pi(k-1)n}{N}) \quad \text{Fourier transform}$$

$$n = 0, 1, 2, \dots, N-1$$

## Matlab FFT

$$y_k = \frac{1}{N} \sum_{n=1}^N c_n \exp(i \frac{2\pi(k-1)(n-1)}{N}) \quad \text{inverse Fourier transform}$$

$$c_n = \sum_{k=1}^N y_k \exp(-i \frac{2\pi(k-1)(n-1)}{N}) \quad \text{Fourier transform}$$

$$n = 1, 2, \dots, N$$

So do DFT  $\rightarrow$  Matlab FFT is equivalent of  $n \rightarrow n+1$  and vice versa

# Warning about notation

$c_0$  has a special meaning. It is the 0 frequency (i.e., DC) amplitude of the signal. Thus, I will always use the **DFT notation** unless mentioned otherwise.

People often denote the forward Fourier transform as  $\mathcal{F}$

$$Y = \mathcal{F}y$$

So  $Y = (Y_0, Y_1, Y_2, \dots, Y_{N-1}) = (c_0, c_1, c_2, \dots, c_{N-1})$  is the spectrum of the time domain signal  $y$

Inverse Fourier transform is denoted as  $\mathcal{F}^{-1}$

$$y = \mathcal{F}^{-1}Y$$

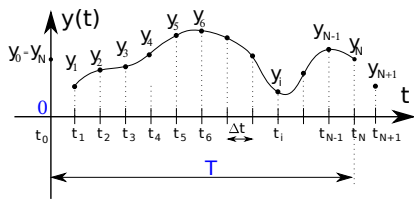
Instead of using  $c_n$  coefficients, we refer in this notation to  $Y_n$

# Sampling rate and important physics relationship

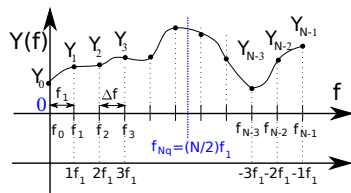
For DFT we need to have equidistant points and the signal repeating itself. We consider signals which start at time 0 and take  $N$  points over the period time  $T$ , thus,  $y_k = y_{k+N}$ . To deduce the time of a data point, we just multiply its index by the time spacing  $\Delta t = T/N$ . I.e.,  $y_i$  is taken at time  $t_i = i\Delta t = i/f_s$

The sampling rate  $f_s$  is defined as  $f_s = 1/\Delta t = f_1 N$ , and  $f_1 = f_s/N = 1/T$  is the frequency spacing in the spectrum, sometimes it is referred as the resolution bandwidth (RBW).

Time series



Spectrum



In Matlab `fft`,  $Y_n$  has the frequency  $f_n = f_1 \times (n - 1) = f_s \times (n - 1)/N$ .

## Nyquist frequency

If we take  $N$  data points with the sampling rate  $f_s$ , what is the maximum frequency which we can expect to see in our spectrum?

**Naively**, we can say  $(N - 1) \times f_1 \approx f_s$ , since in the DFT spectrum all points are separated by the fundamental frequency  $f_1 = 1/T = f_s/N$ . However, recall that

$$Y_n = c_n = \sum_{k=1}^N y_k \exp(-i \frac{2\pi(k-1)n}{N})$$

Thus,  $Y_{N-n} = Y_{-n}$ , i.e., the higher half of the vector  $Y$  contains negative frequency. So at most, we can hope to obtain a spectrum with the highest frequency **smaller than**

## Nyquist frequency

$$F_{Nq} = f_1 \frac{N}{2} = \frac{f_s}{2}$$

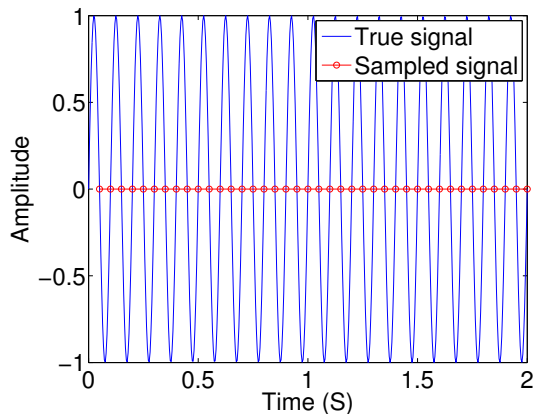
# Nyquist criteria

$$f_s > 2f_{\text{signal}}$$

You must sample your signal twice faster than the highest frequency component of it. I.e., the Nyquist frequency of your sample should be  $>$  than the highest signal frequency.

# Aliasing: wrong/slow sampling frequency

Sampling with  
 $f_s = 2f_{\text{signal}}$   
i.e.  
 $f_{Nq} = f_{\text{signal}}$   
Sampled signal  
appeared to be DC

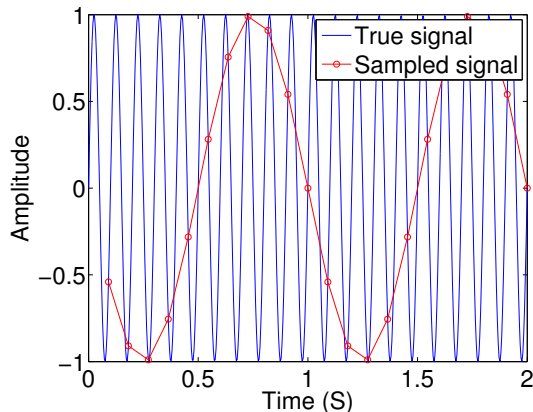


# Aliasing: too slow sampling frequency - reflection

Under sampling

$$f_s = 1.1 f_{\text{signal}}$$

The sampled signal seems to have a lower frequency.



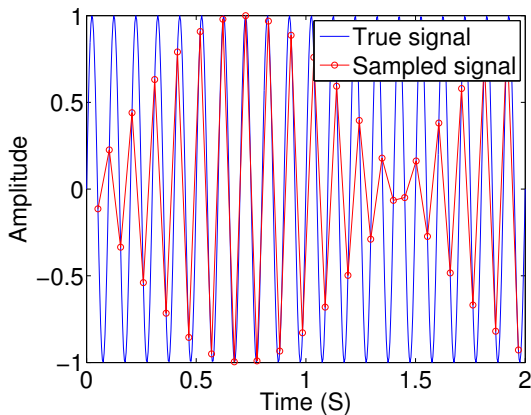
The sampled signal appears to have a slower frequency. This is case of the reflection/folding where the signal frequency is slightly higher than the sampling frequency.

$$f_{\text{apparent signal}} = (f_{\text{signal}} - 2f_{Nq}) \approx f_{\text{signal}} - f_s$$



# Aliasing: too slow sampling frequency - ghosts

Under sampling  
 $f_s = 1.93f_{\text{signal}}$   
The sampled signal  
looks very different.



# DFT filters

Once you get a signal, you can filter the unwanted frequencies out of it. The recipe is the following

- sample the signal
- calculate DFT (use Matlab `fft`)
- have a look at the spectrum and decide which frequencies are unwanted
- apply a filter which attenuate unwanted frequencies amplitudes
  - If you attenuate the component of the frequency  $f$  by  $g_f$ , you need to attenuate the component at  $-f$  by  $g_f^*$ . Otherwise, the inverse Fourier transform will have non zero imaginary part.
- calculate inverse DFT (`ifft`) of the filtered spectrum
- repeat if needed

## Applications

- Noise reduction
- Compression