Fast Fourier transform (FFT)

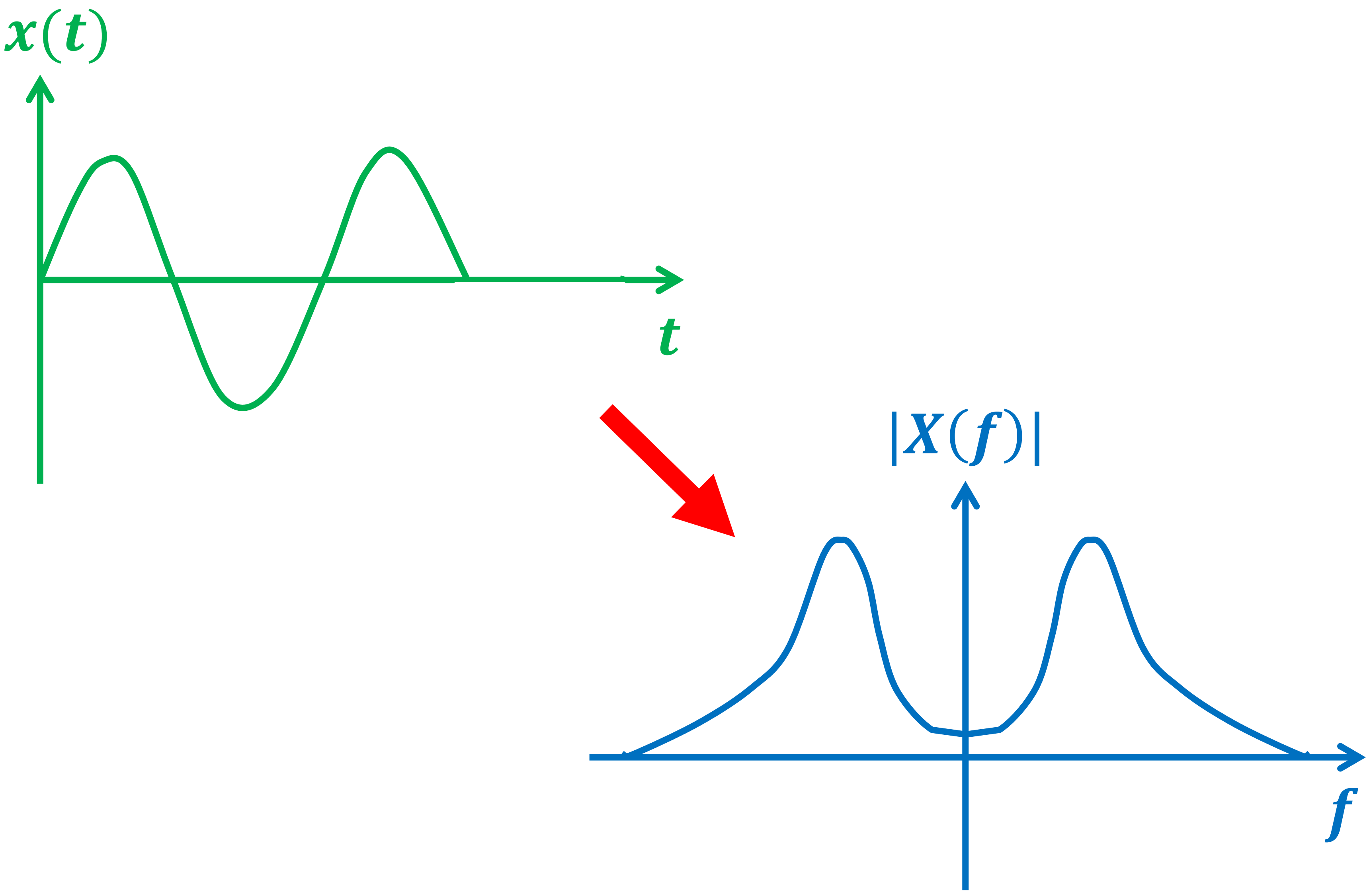
MATLAB tutorial series (Part 2.1)

Pouyan Ebrahimbabaie

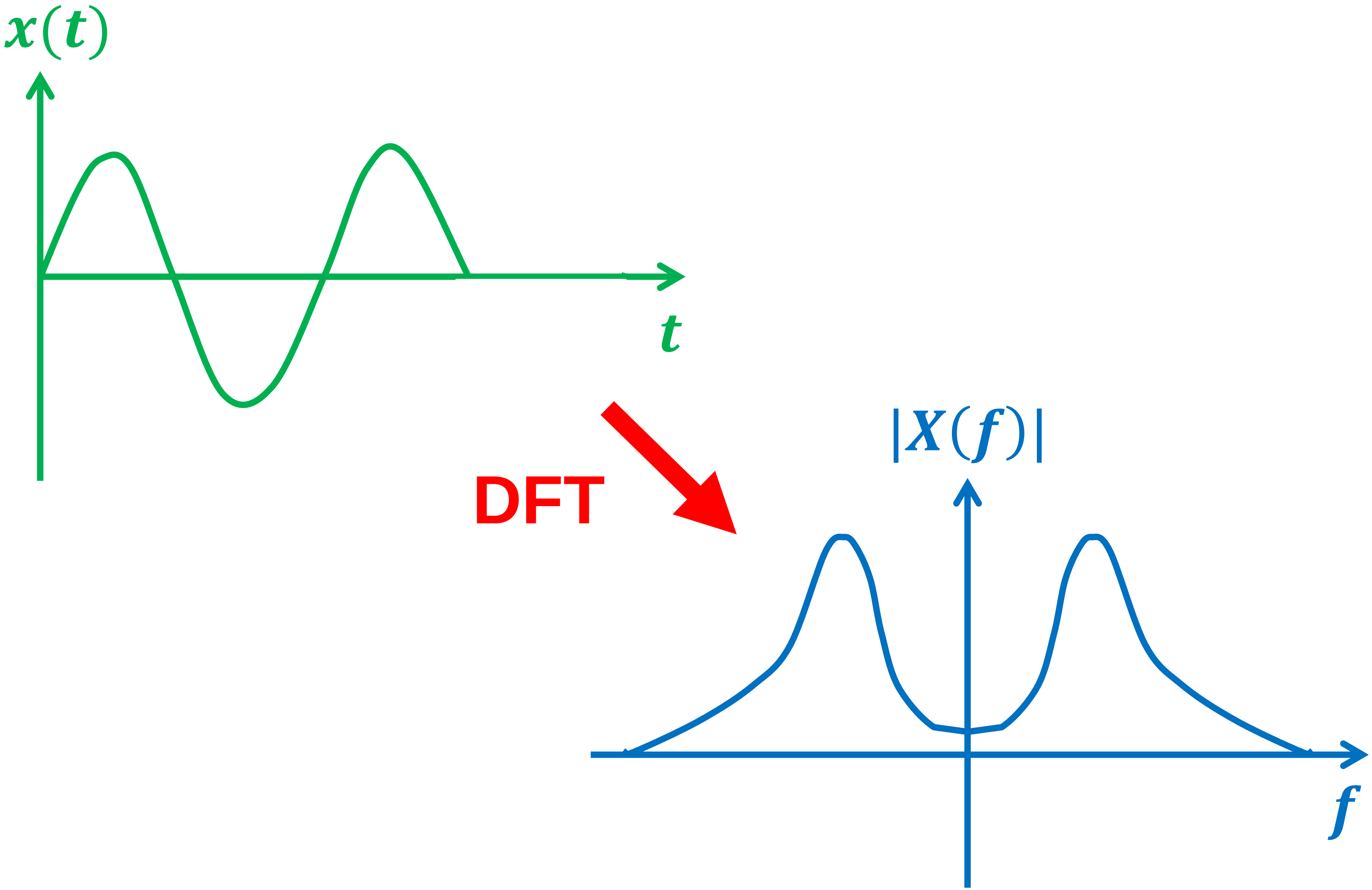
Laboratory for Signal and Image Exploitation (INTELSIG)
Dept. of Electrical Engineering and Computer Science
University of Liège
Liège, Belgium

Applied digital signal processing (ELEN0071-1)
31 March 2021

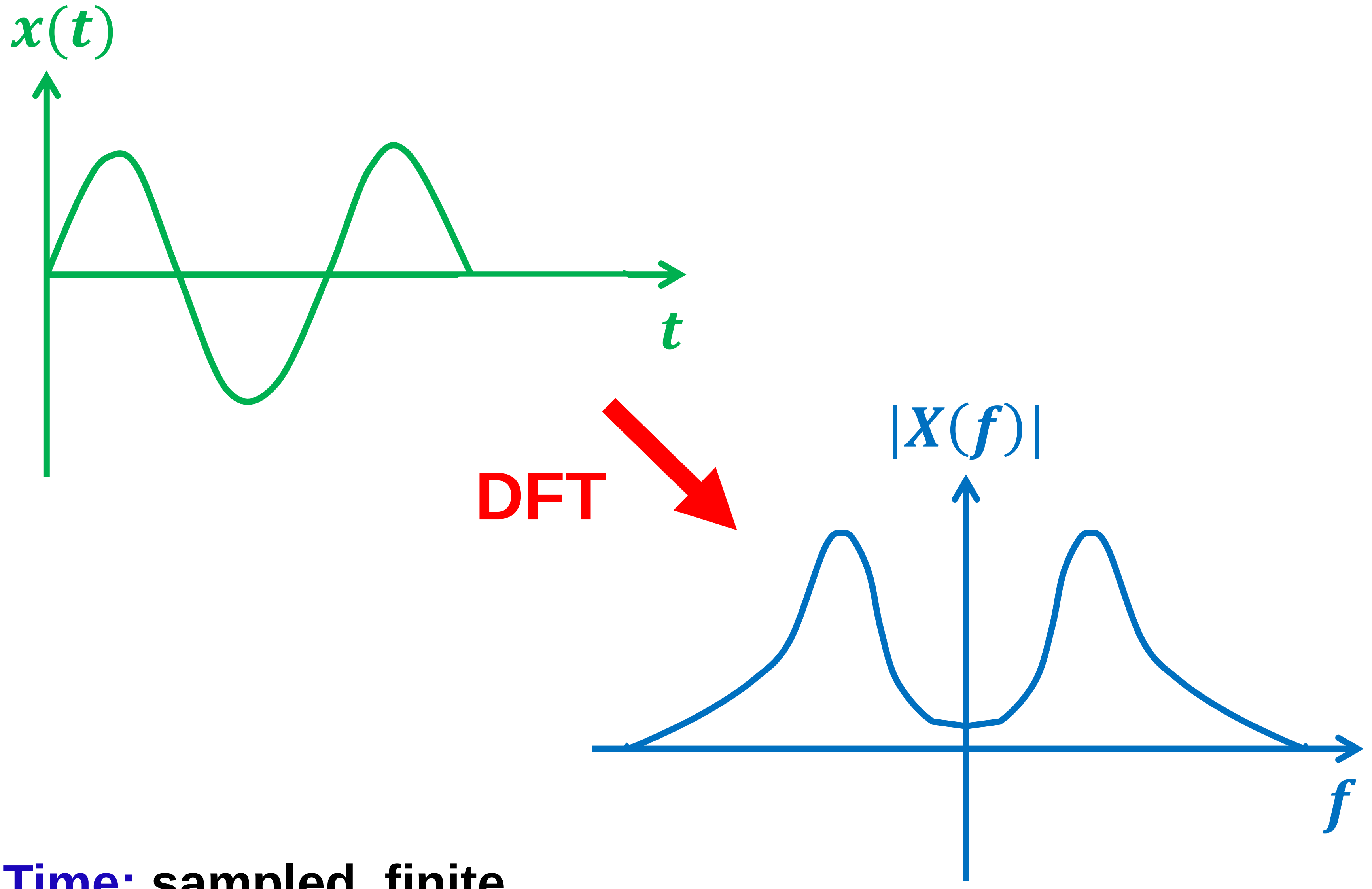
Introduction



Introduction



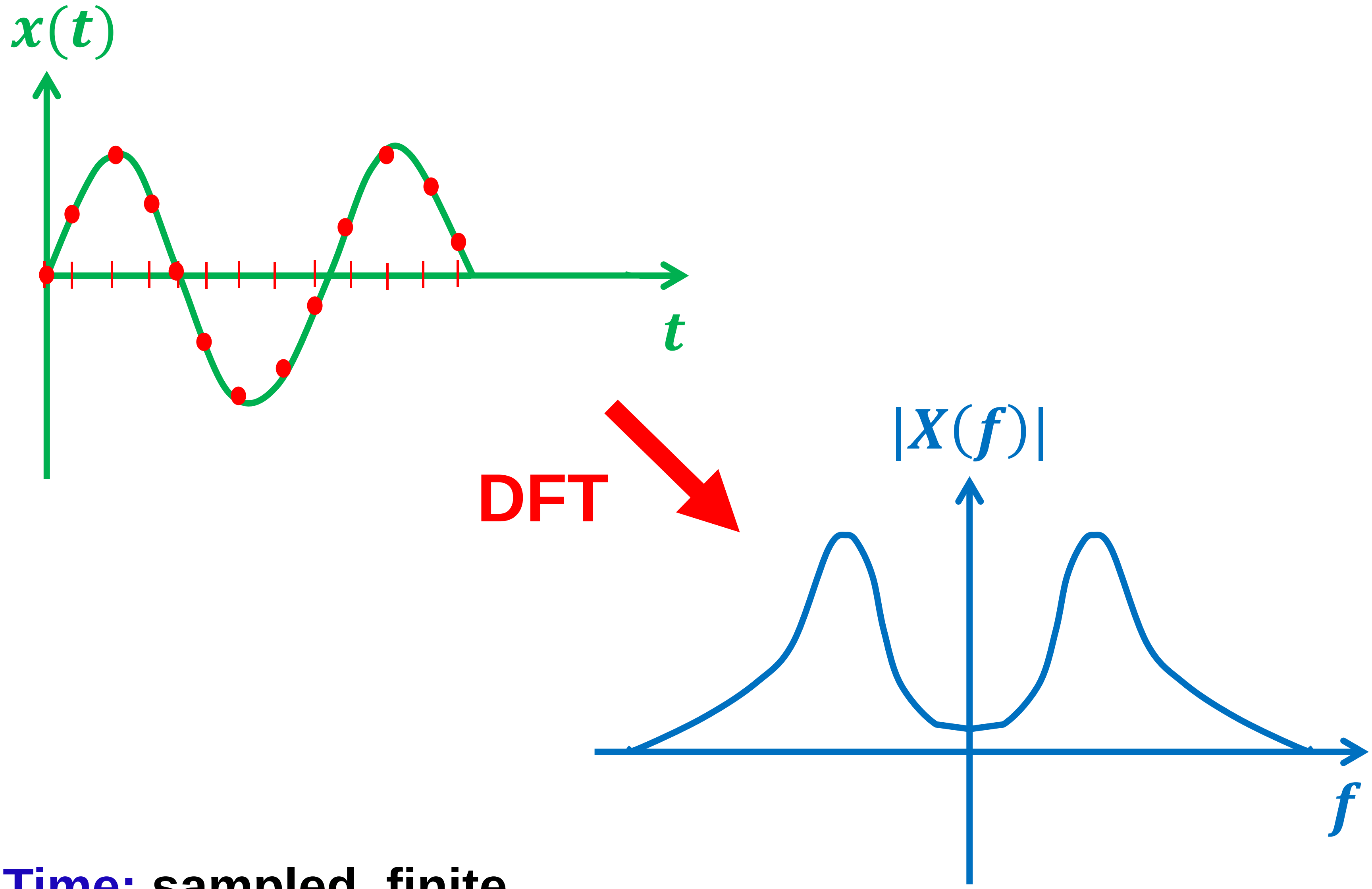
Introduction



Time: **sampled, finite**

Frequency: **sampled, finite**

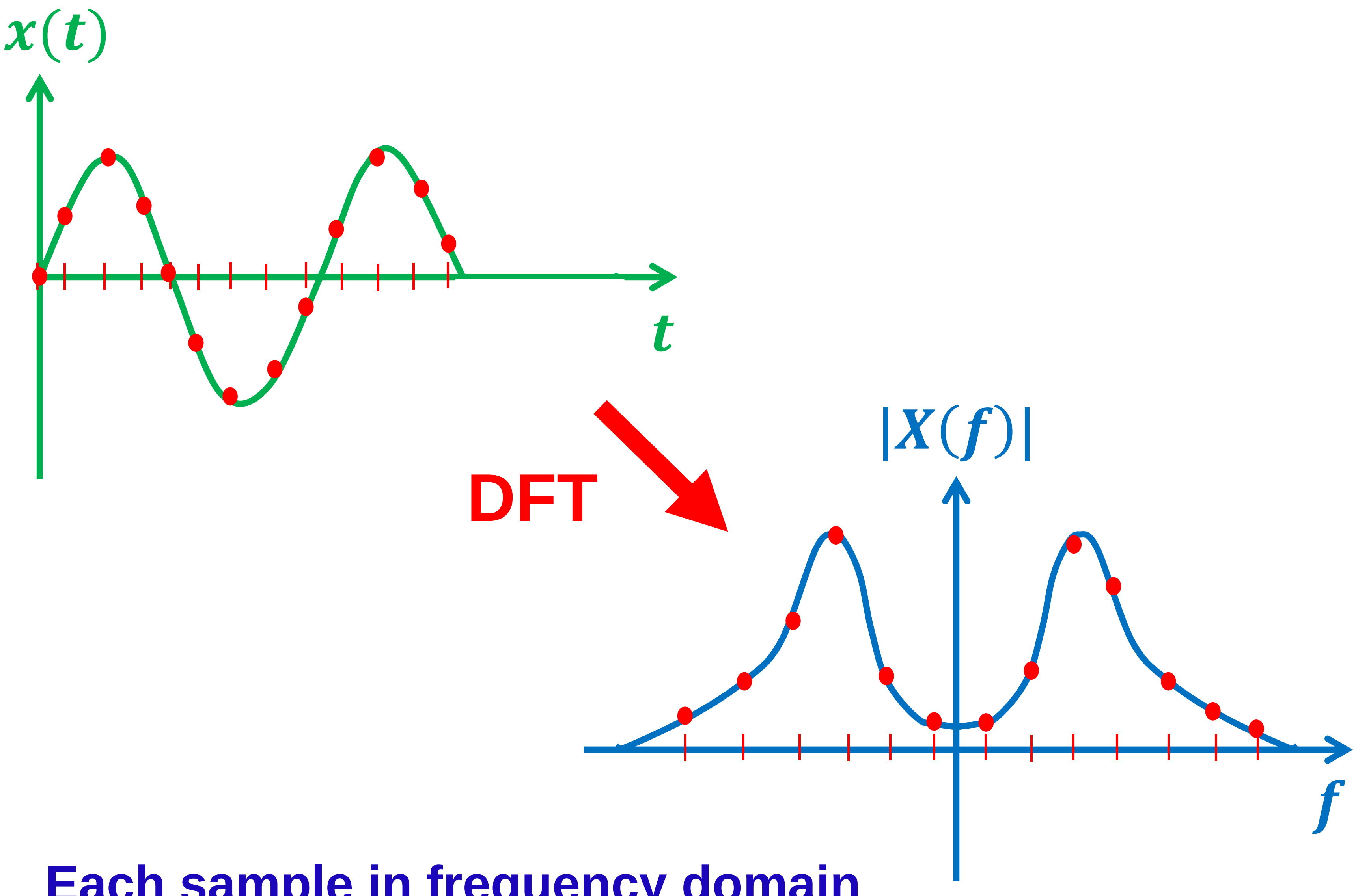
Introduction



Time: **sampled, finite**

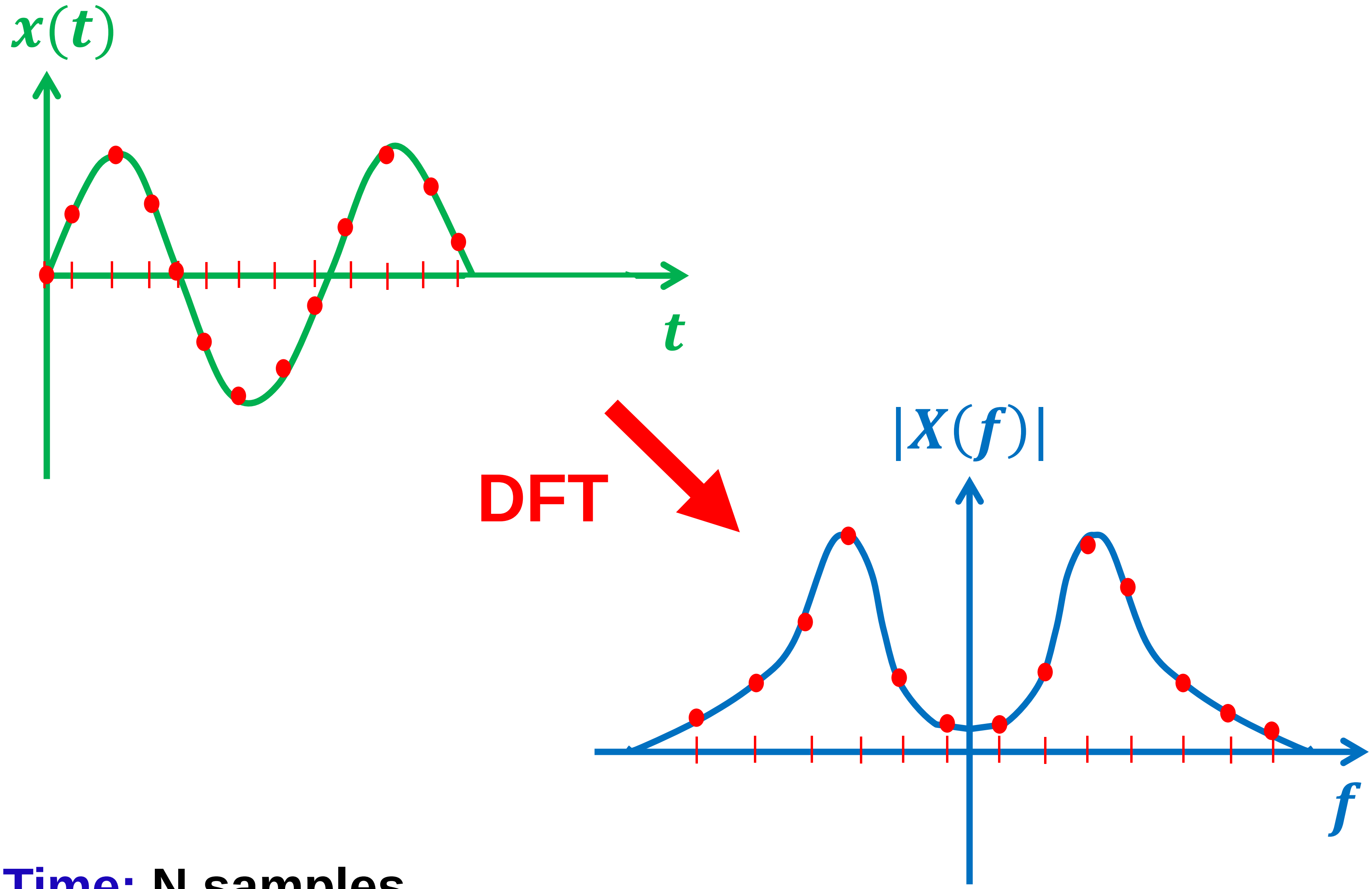
Frequency: **sampled, finite**

Introduction



Each sample in frequency domain corresponds to a sample in time domain !

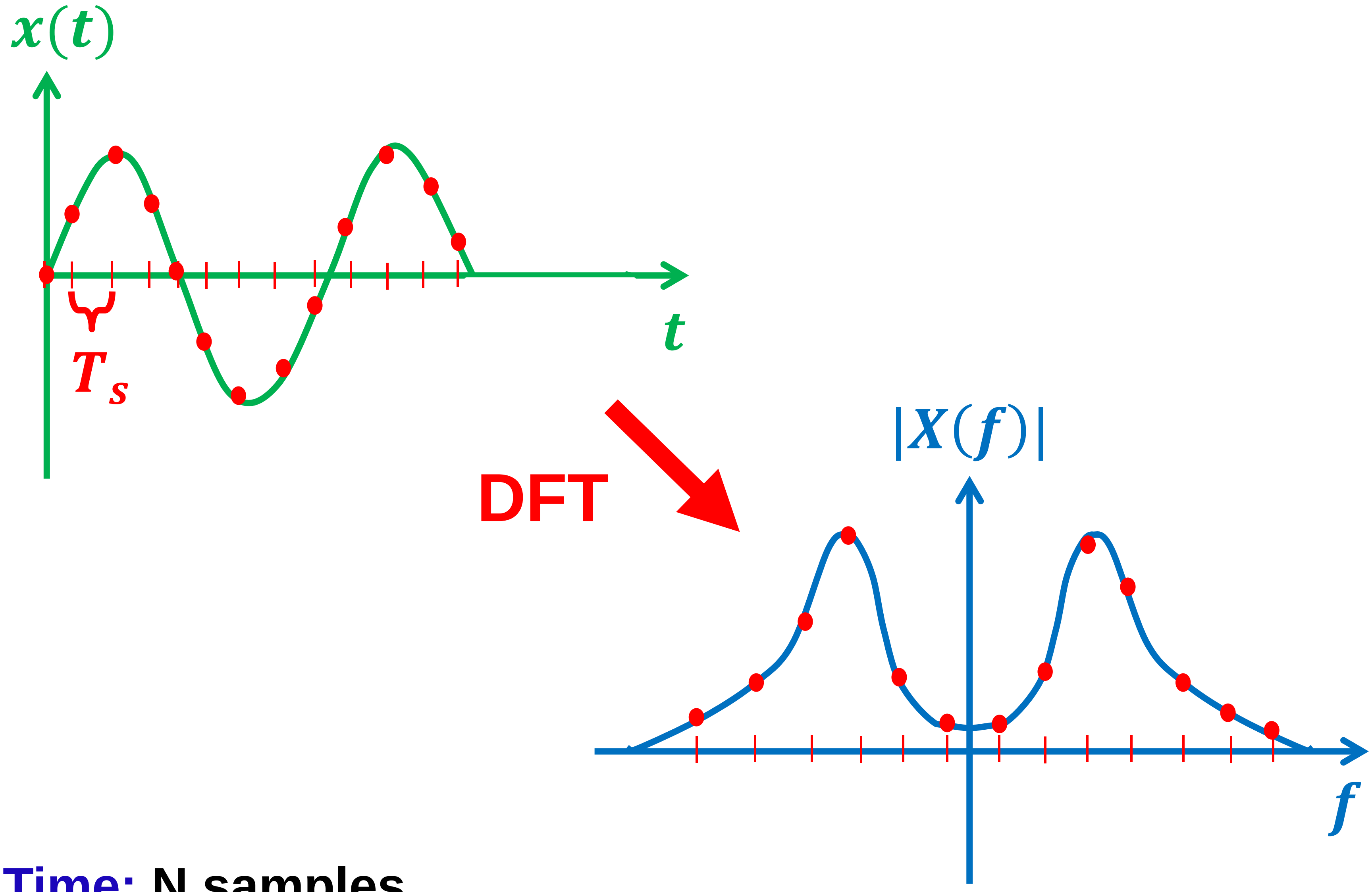
Introduction



Time: N samples

Frequency: N samples

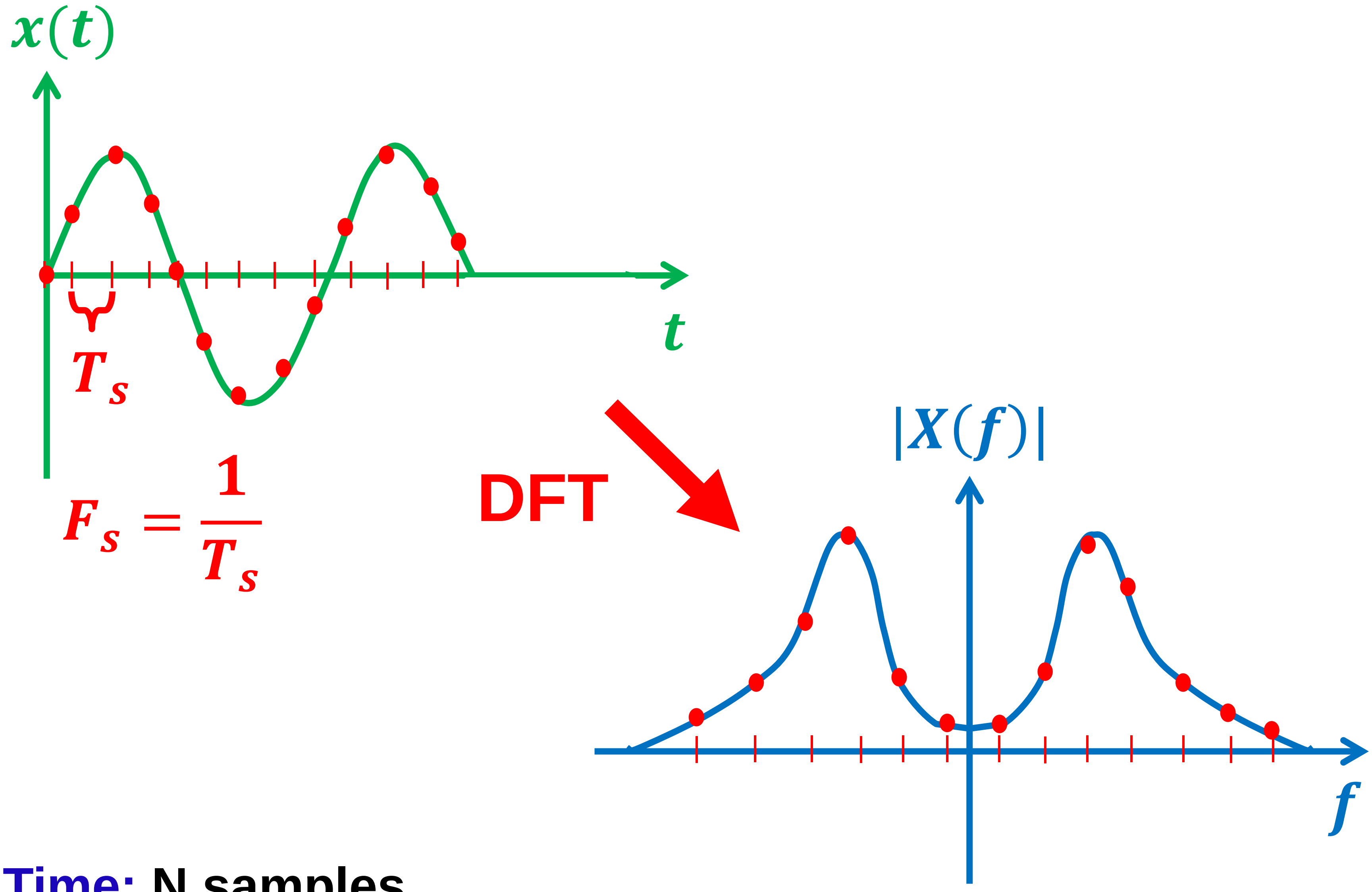
Introduction



Time: N samples

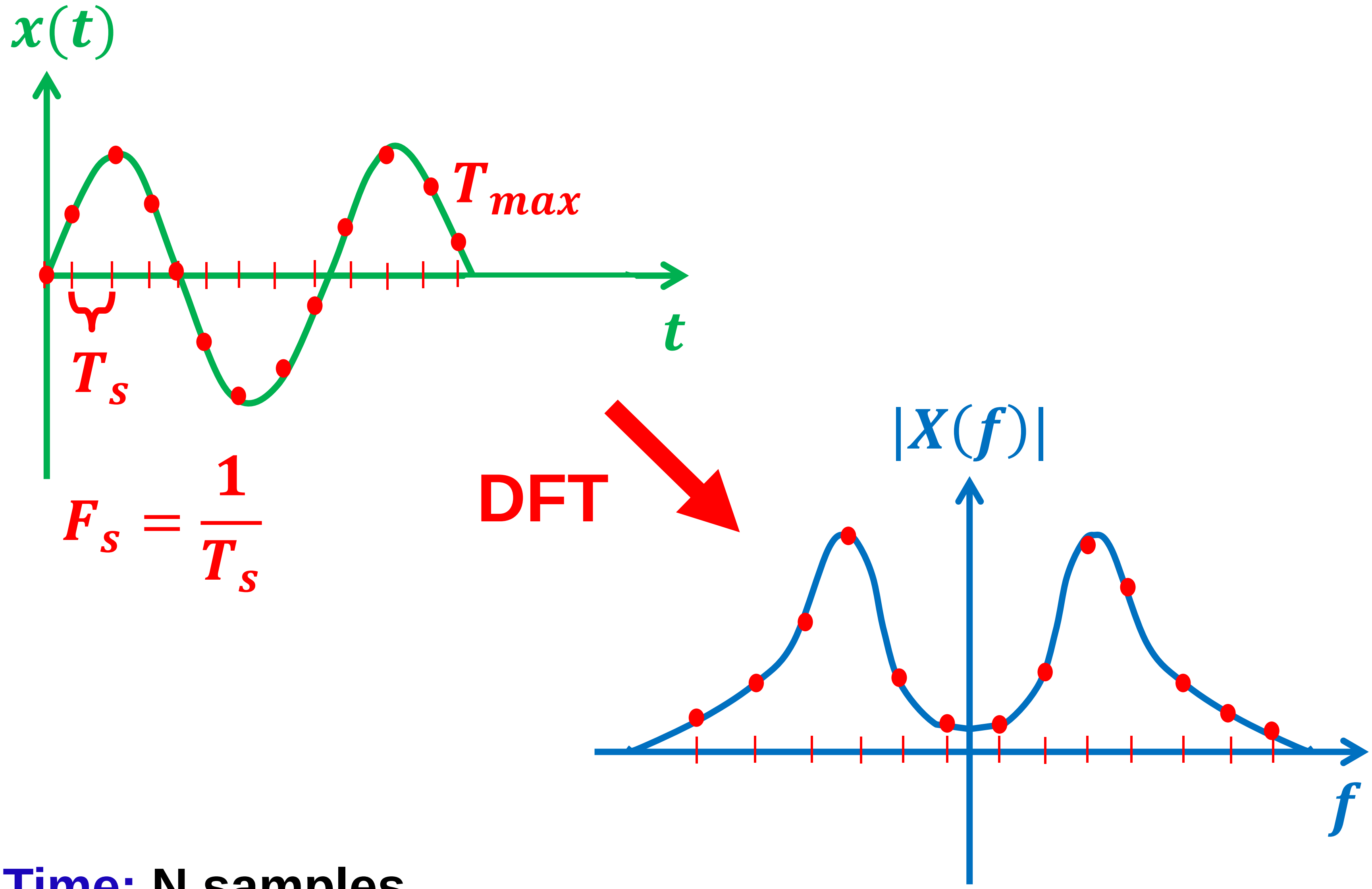
Frequency: N samples

Introduction



Time: N samples
Frequency: N samples

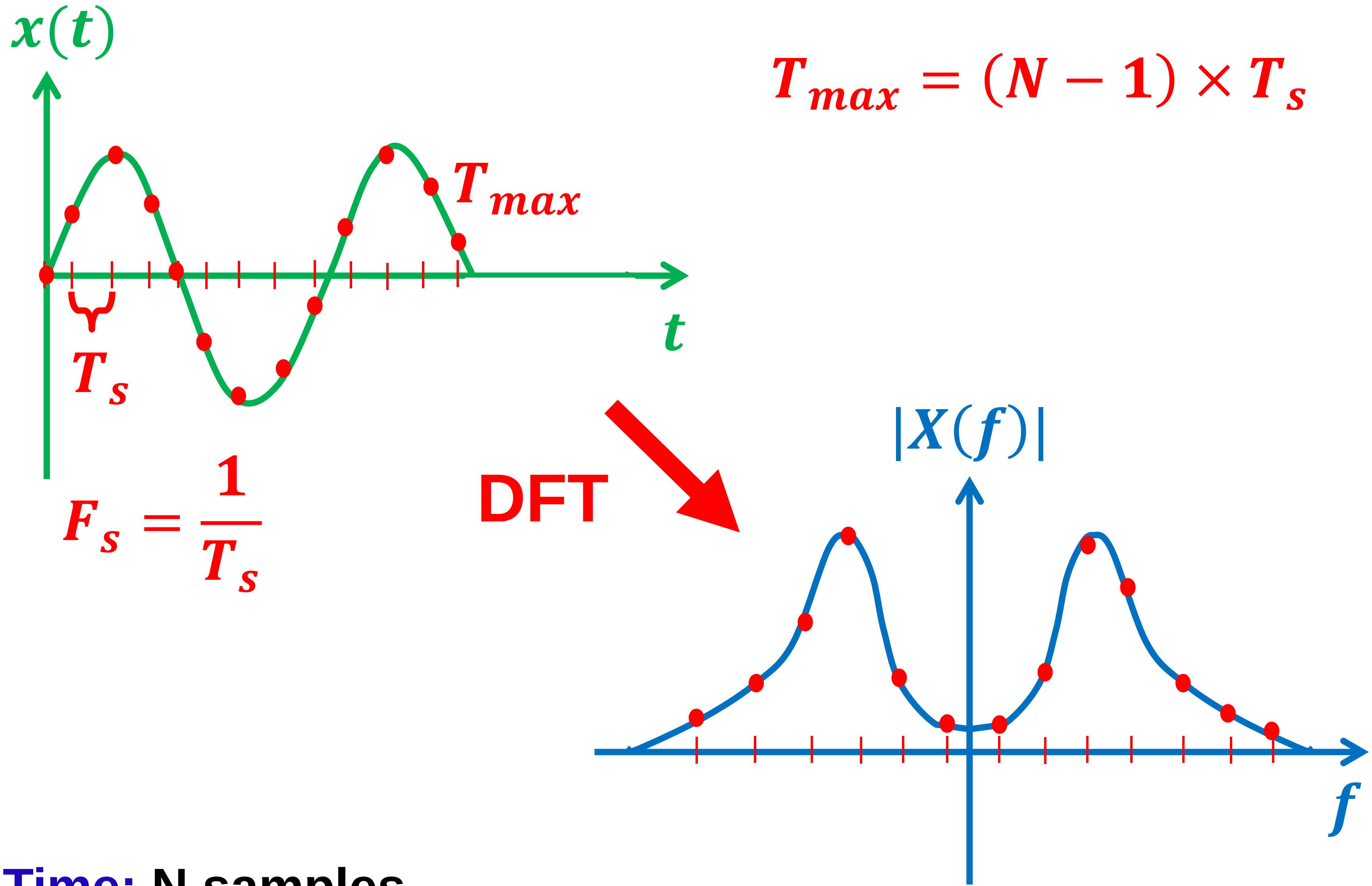
Introduction



Time: N samples

Frequency: N samples

Introduction



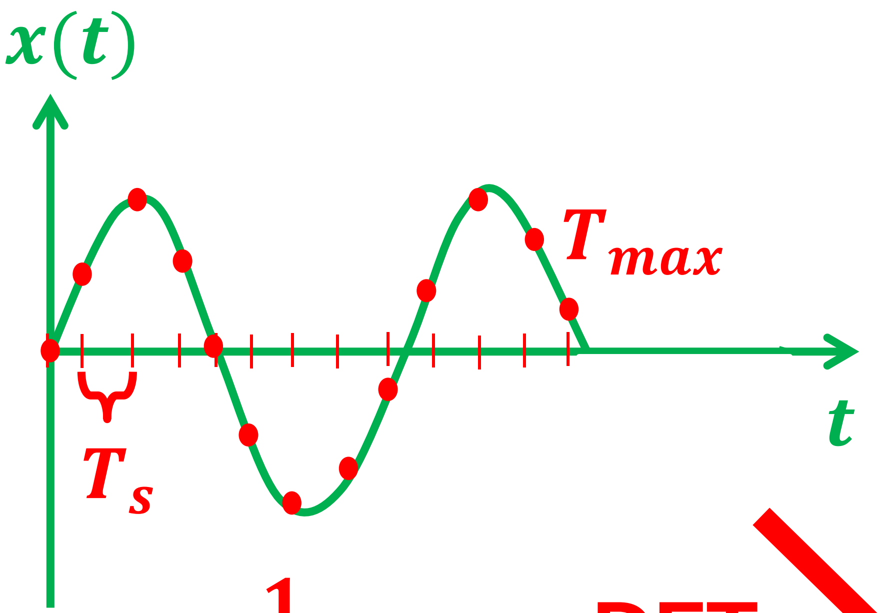
$$T_{max} = (N-1) \times T_s$$

$$F_s = \frac{1}{T_s}$$

Time: **N samples**

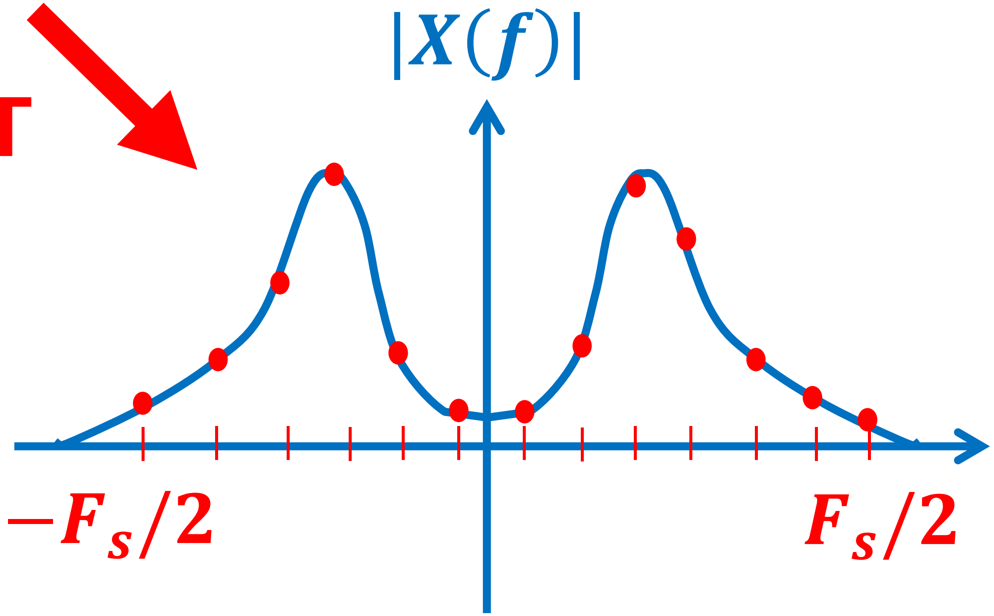
Frequency: **N samples**

Introduction



$$T_{max} = (N - 1) \times T_s$$

DFT

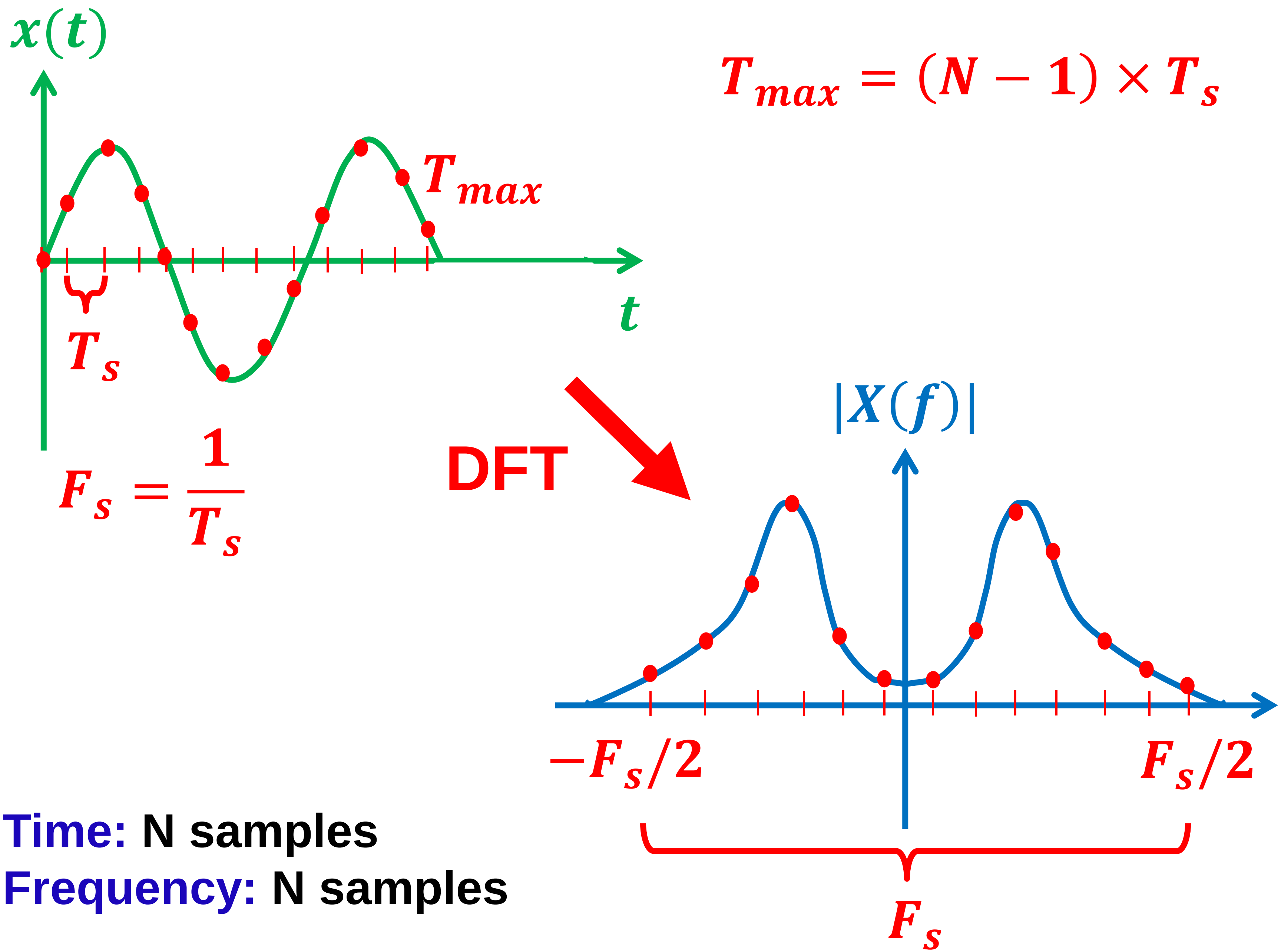


Time: **N samples**

Frequency: **N samples**

From sampling theorem

Introduction

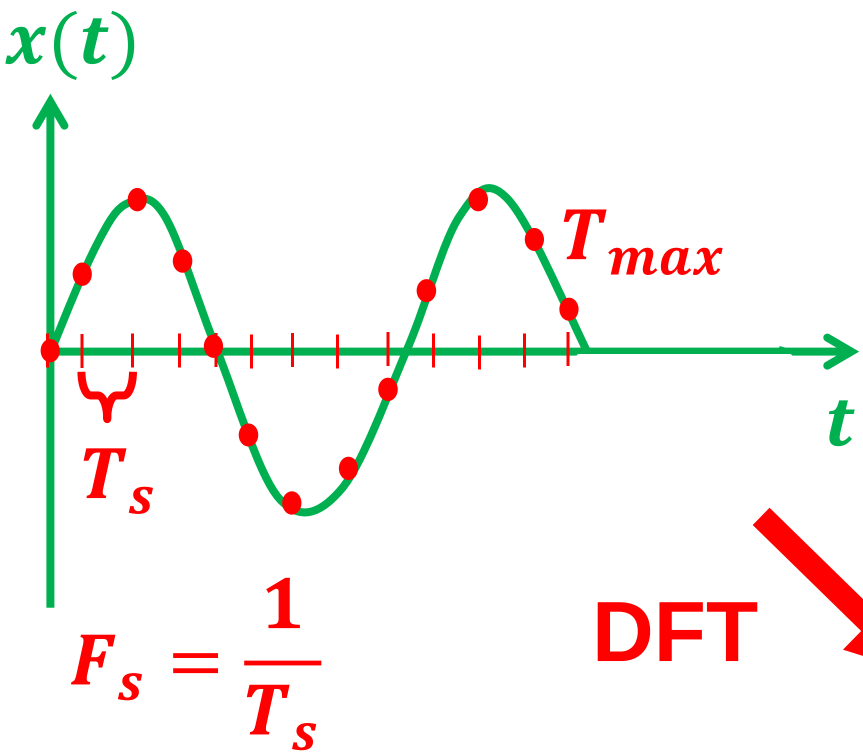


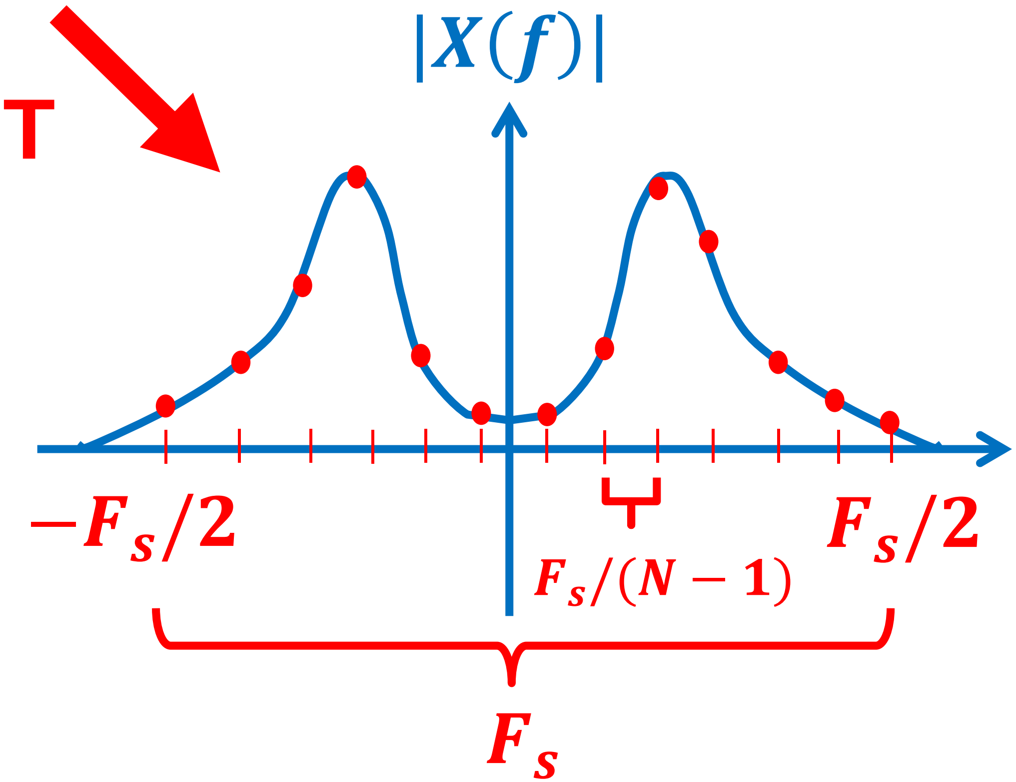
$$T_{max} = (N-1) \times T_s$$

Time: N samples
Frequency: N samples

Introduction

$$T_{max} = (N - 1) \times T_s$$





Time: N samples

Frequency: N samples

FFT is just an algorithm to compute DFT efficiently !

In MATLAB:

```
X=fft(x)
or
X=fft(x,n)
```

Example 2.1: pure tone

```
% Sampling frequency
Fs=44100;
```

Example 2.1: pure tone

```
% Sampling frequency
Fs=44100;
% Sampling period
Ts=1/Fs;
```

Example 2.1: pure tone

```
% Sampling frequency
Fs=44100;
% Sampling period
Ts=1/Fs;
% Signal length (in second)
N_sec=5;
% Siganal length (in sample)
N=N_sec*Fs;
```

Example 2.1: pure tone

```
% Sampling frequency
Fs=44100;
% Sampling period
Ts=1/Fs;
% Signal length (in second)
N_sec=5;
% Siganal length (in sample)
N=N_sec*Fs;
% Maximum time
Tmax=(N-1)*Ts;
% Time vector
t=0:Ts:Tmax;
```

Example 2.1: pure tone

```
% Signal
x=sin(2*pi*F0.*t);
% Play the sound
sound(x,Fs)
% Plot the sound (show from zero to 60 msec)
figure(1)
plot(t,x,'LineWidth',2.5)
xlim([0 0.06])
```

Example 2.1: pure tone

```
% Signal
x=sin(2*pi*F0.*t);
% Play the sound
sound(x,Fs)
% Plot the sound (show from zero to 60 msec)
figure(1)
plot(t,x,'LineWidth',2.5)
xlim([0 0.06])
% take FFT (without shift)
X1=fft(x);
```

Example 2.1: pure tone

```
% Signal
x=sin(2*pi*F0.*t);
% Play the sound
sound(x,Fs)
% Plot the sound (show from zero to 60 msec)
figure(1)
plot(t,x,'LineWidth',2.5)
xlim([0 0.06])
% take FFT (without shift)
X1=fft(x);
% plot result
figure(2)
plot(abs(X1),'LineWidth',2.5);
```

Example 2.1: pure tone

```
% Signal
x=sin(2*pi*F0.*t);
% Play the sound
sound(x,Fs)
% Plot the sound (show from zero to 60 msec)
figure(1)
plot(t,x,'LineWidth',2.5)
xlim([0 0.06])
% take FFT (without shift)
X1=fft(x);
% plot result
figure(2)
plot(abs(X1),'LineWidth',2.5);
```

You should always perform some post processing operations (shifting, scaling, etc.) to be able to present the results of fft !

Example 2.1: pure tone

```
% take FFT (with shift)
X2=fftshift(fft(x));
```

Example 2.1: pure tone

```
% take FFT (with shift)
X2=fftshift(fft(x));
% Frequency range
F=-Fs/2:Fs/(N-1):Fs/2;
```

Example 2.1: pure tone

```
% take FFT (with shift)
X2=fftshift(fft(x));
% Frequency range
F=-Fs/2:Fs/(N-1):Fs/2;
figure(3)
plot(F,abs(X2)/N,'LineWidth',2.5);
xlabel('Frequency (Hz)')
title('Double sided magnitude response')
```

Example 2.1: pure tone

fftshift

```
% take FFT (with shift)
X2=fftshift(fft(x));
% Frequency range
F=-Fs/2:Fs/(N-1):Fs/2;
figure(3)
plot(F,abs(X2)/N,'LineWidth',2.5);
xlabel('Frequency (Hz)')
title('Double sided magnitude response')
```

Scaling

Example 2.1: pure tone

```
% take FFT (with shift)
X2=fftshift(fft(x));
% Frequency range
F=-Fs/2:Fs/(N-1):Fs/2;
figure(3)
plot(F,abs(X2)/N,'LineWidth',2.5);
xlabel('Frequency (Hz)')
title('Double sided magnitude response')
```

This is the correct method to graph a "double-sided" (negative and positive) frequency spectrum ☺

Example 2.2: single sided spectrum

```
% Sampling frequency
Fs=44100;
% Sampling period
Ts=1/Fs;
% Signal length (in second)
N_sec=5;
% Signal length (in sample)
N=N_sec*Fs;
% Maximum time
Tmax=(N-1)*Ts;
% Time vector
t=0:Ts:Tmax;
```

Example 2.2: single sided spectrum

```
% pure F0, F1 and F3
F0=600;
F1=1300;
F2=2000;
% Signal
x=sin(2*pi*F0.*t)+...
...0.5*sin(2*pi*F1.*t)+0.2*sin(2*pi*F2.*t);
% Play the sound
sound(x,Fs)
% Plot the sound (show from zero to 60 msec)
figure(1)
plot(t,x,'LineWidth',2.5)
xlim([0 0.06])
```

Example 2.2: single sided spectrum

```
% Compute fft
X=fft(x);
```

Example 2.2: single sided spectrum

```
% Compute fft
X=fft(x);
% Take abs and scale it
X2=abs(X/N);
```

Example 2.2: single sided spectrum

```
% Compute fft
X=fft(x);
% Take abs and scale it
X2=abs(X/N);
% Pick the first half
X1=X2(1:N/2+1);
```

Example 2.2: single sided spectrum

```
% Compute fft
X=fft(x);
% Take abs and scale it
X2=abs(X/N);
% Pick the first half
X1=X2(1:N/2+1);
% Multiply by 2 (except the DC part), to compensate
% the removed side from the spectrum.
X1(2:end-1) = 2*X1(2:end-1);
```

Example 2.2: single sided spectrum

```
% Compute fft
X=fft(x);
% Take abs and scale it
X2=abs(X/N);
% Pick the first half
X1=X2(1:N/2+1);
% Multiply by 2 (except the DC part), to compensate
% the removed side from the spectrum.
X1(2:end-1) = 2*X1(2:end-1);
% Frequency range
F = Fs*(0:(N/2))/N;
```

Example 2.2: single sided spectrum

```
% Plot single-sided spectrum
plot(F,X1,'LineWidth',2.5)
title('Single-Sided Amplitude Spectrum')
xlabel('f (Hz)')
```

Most of the time we use “single-sided” amplitude or phase spectrum !

Example 2.3: siren

```
F0=1300;
F1=200;
F2=1400;
B0=100;
B1=100;
B2=500;
% Signal
x=sin(2*pi*F0.*t+B0*pi*t.^2)+...
sin(2*pi*F1.*t+B1*pi*t.^2)...
+sin(2*pi*F2.*t+B2*pi*t.^2);
```

Example 2.4: voice

```
% read audio file .wav
[x,Fs]=audioread('adult_female_speech.wav');
% play the sound
sound(x,Fs)
% Sampling period
Ts=1/Fs;
% Length of signal
N=length(x);
% Maximum time
Tmax=(N-1)*Ts;
% Time vector
t=0:Ts:Tmax;
…
```

Usable voice frequency band in telephony:

~ 300 Hz to 3400 Hz

How fast is it?

DTFT:

$$X(e^{j\omega}) = \sum_{n=0}^{N-1} x[n]e^{-j\omega n}$$

How fast is it?

DTFT: $$X(e^{j\omega}) = \sum_{n=0}^{N-1} x[n]e^{-j\omega n}$$

Sample ω at $\omega_k = (2\pi/N)k,\ \ k = 0, 1, 2, \cdots N$

How fast is it?

DTFT:

$$X(e^{j\omega}) = \sum_{n=0}^{N-1} x[n]e^{-j\omega n}$$

Sample ω **at** $\omega_k = (2\pi/N)k, \;\; k = 0, 1, 2, \cdots N$

DFT:

$$X[k] = X(e^{j\frac{2\pi}{N}k}) = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn}$$

How fast is it?

DTFT:

$$X(e^{j\omega}) = \sum_{n=0}^{N-1} x[n]e^{-j\omega n}$$

Sample ω at $\omega_k = (2\pi/N)k, \;\; k = 0, 1, 2, \cdots N$

DFT:

$$X[k] = X(e^{j\frac{2\pi}{N}k}) = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn}$$

For each k:

N complex multiplications,

$N - 1$ complex adds

How fast is it?

$O(N^2)$ computations for direct DFT

How fast is it?

$O(N^2)$ computations for direct DFT

vs.

$O(N \log_2 N)$ computations for fft !

How fast is it?

$O(N^2)$ computations for direct DFT

vs.

$O(N \log_2 N)$ computations for fft !

N	1000	10^6	10^9
N^2	10^6	10^{12}	10^{18}
$N \log_2 N$	10^4	20×10^6	30×10^9

How fast is it?

$O(N^2)$ computations for direct DFT

vs.

$O(N \log_2 N)$ computations for fft !

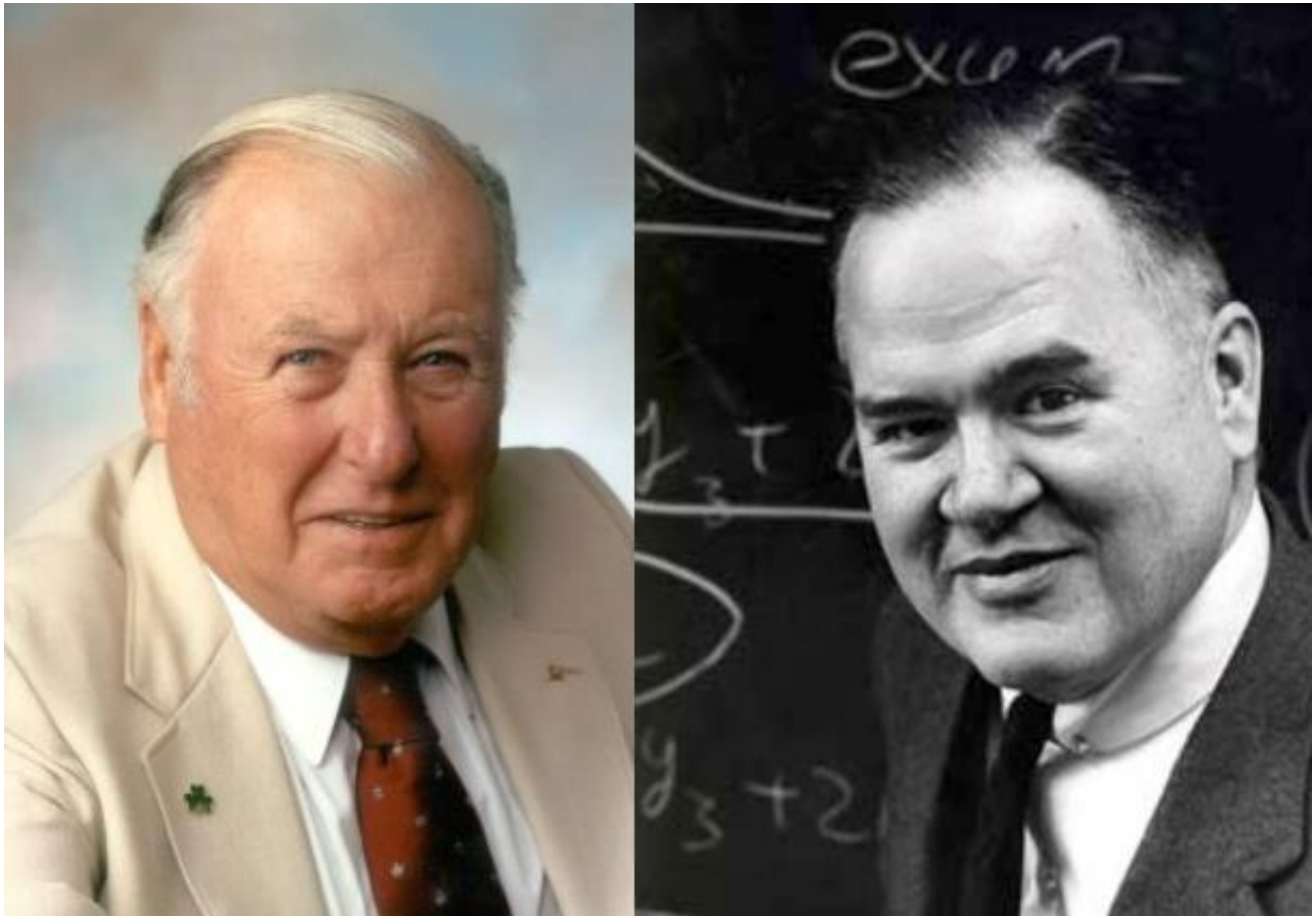
N	1000	10^6	10^9
N^2	10^6	10^{12}	10^{18}
$N \log_2 N$	10^4	20×10^6	30×10^9

$10^{18}\, ns \sim 31.2\ years$

vs.

$30 \times 10^9\, ns \sim 30\ seconds$

First time presented by …



Cooley and Tukey (1965)

First time presented by …

Gauss (1805)

Real-Time application

…

Useful links

- https://www.youtube.com/watch?v=iTMn0Kt18tg
- https://allsignalprocessing.com/fast-fourier-transform-fft-algorithm/
- https://allsignalprocessing.com/discrete-fourier-transform-sampling-the-dtft/
- https://nl.mathworks.com/help/matlab/ref/fft.html