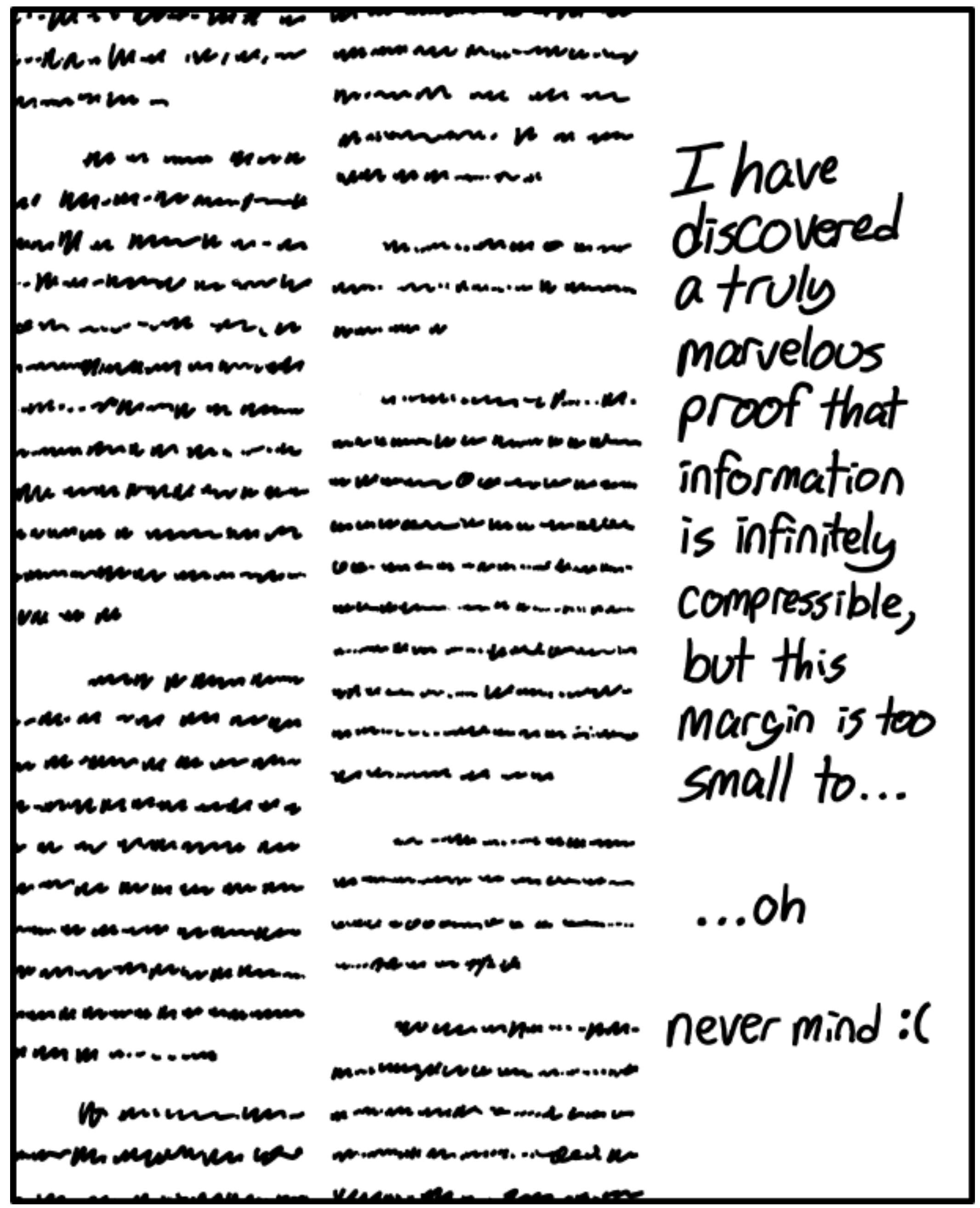
Applications of Mathematics in Computer Science (MACS)

LECTURE #5: FOURIER TRANSFORM



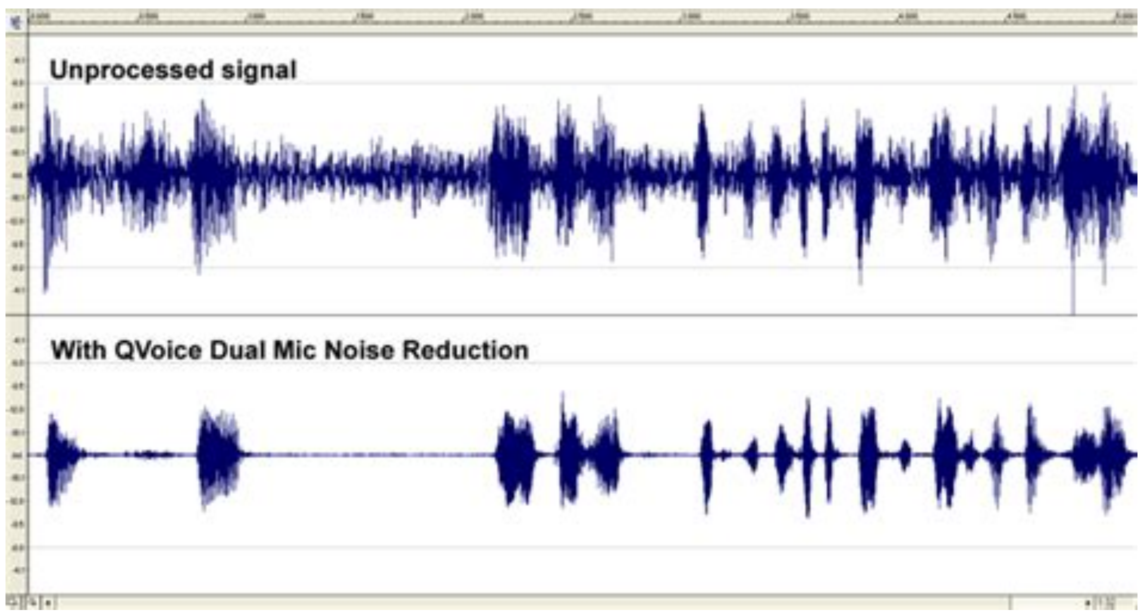
The CS Problem:

frequency filtering

- Time-series search and comparison (Shazam)
- Information compression (Zoom, Skype)
- Image processing

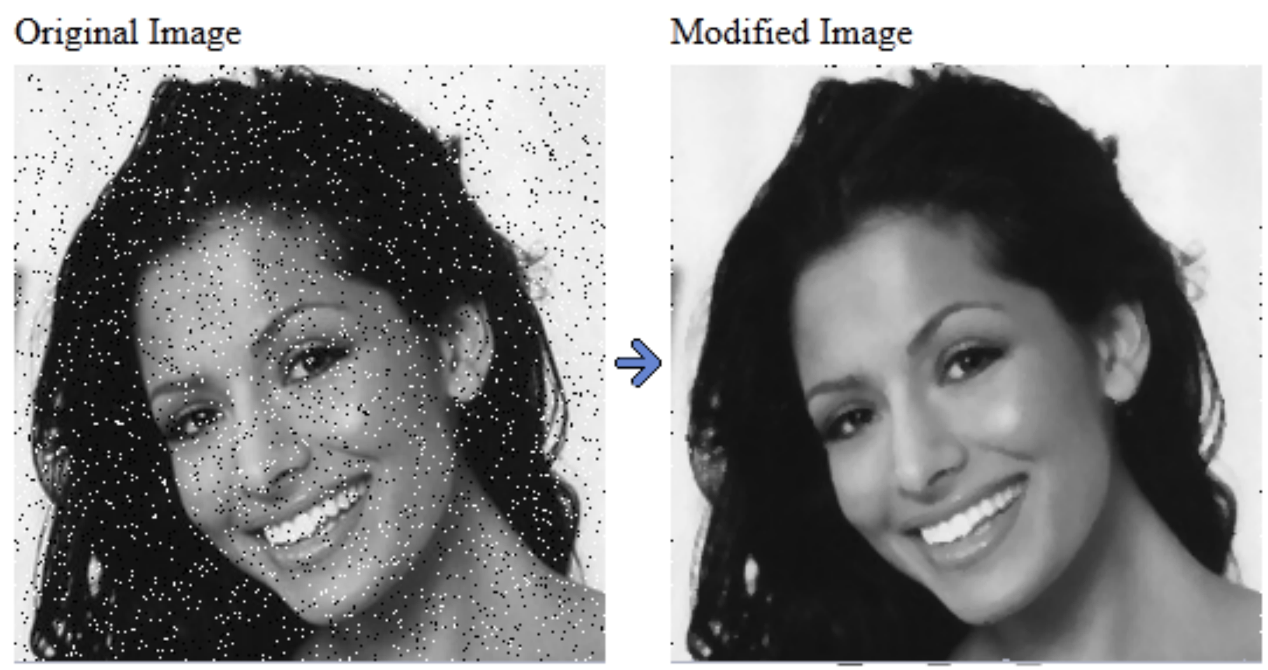
Frequency filtering

Sound



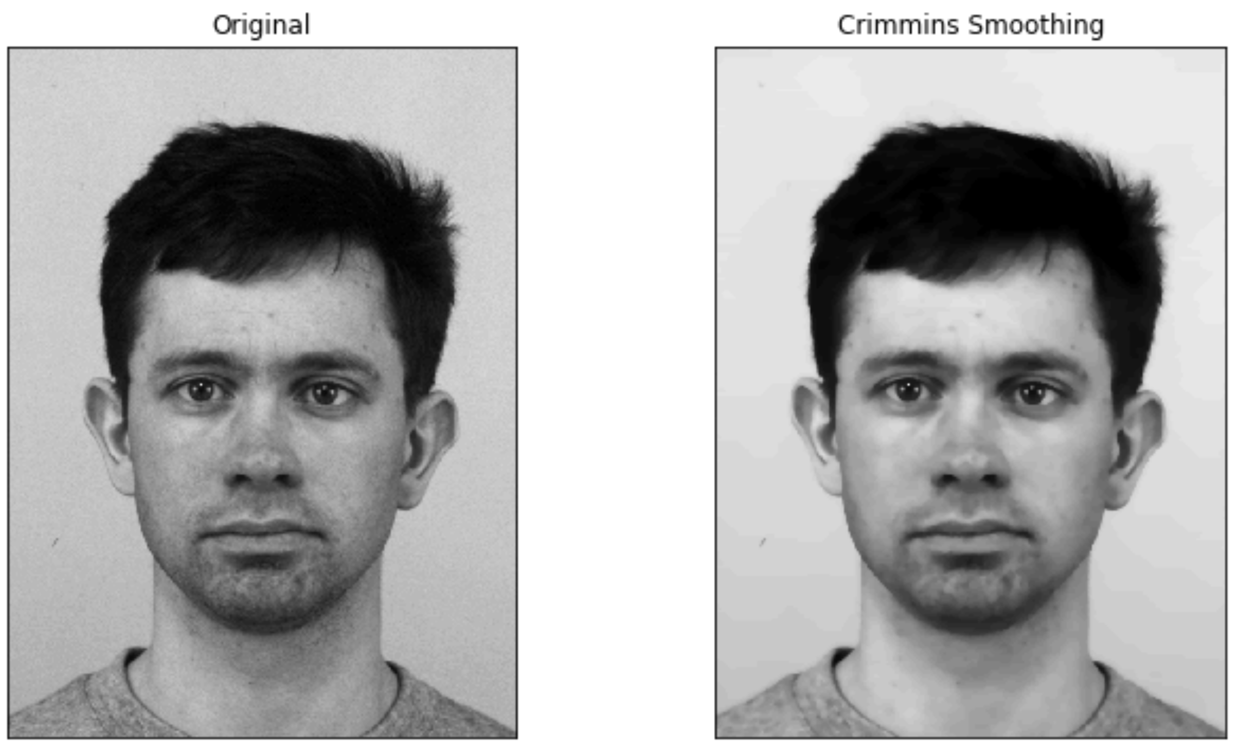
Image

Noise reduction



Types of filters

- Low-pass filter



- High-pass filter

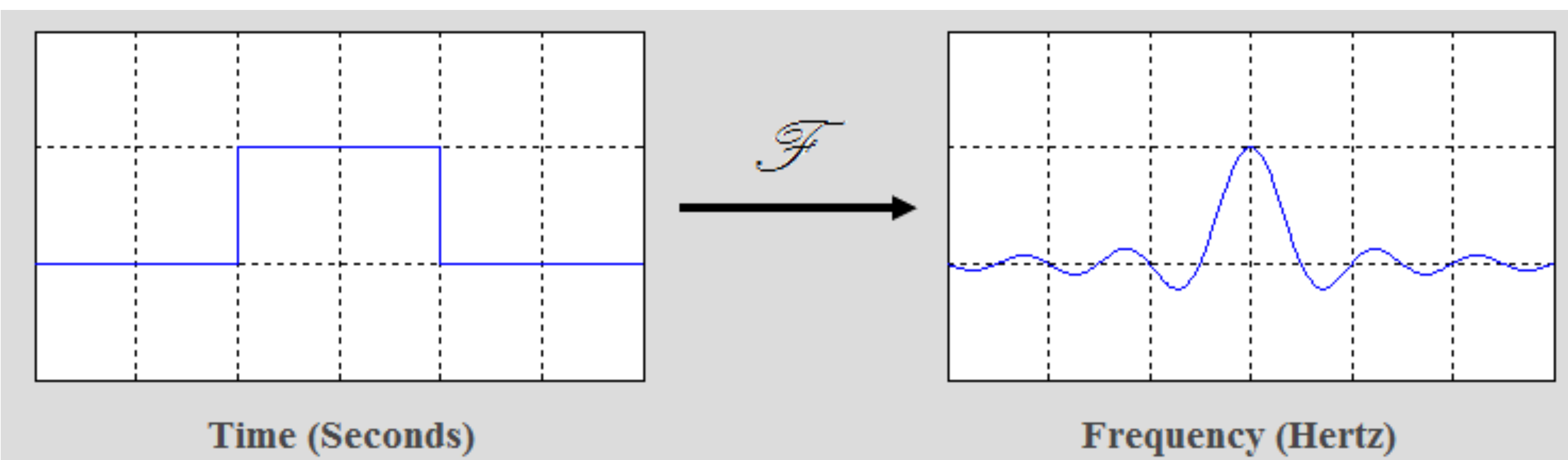
- Band-pass filter

Method: Fourier transform

spacial/temporal domain $y = f(x)$

↓

frequency domain $y' = f(\texttt{frequency})$



But how?

The math technique:

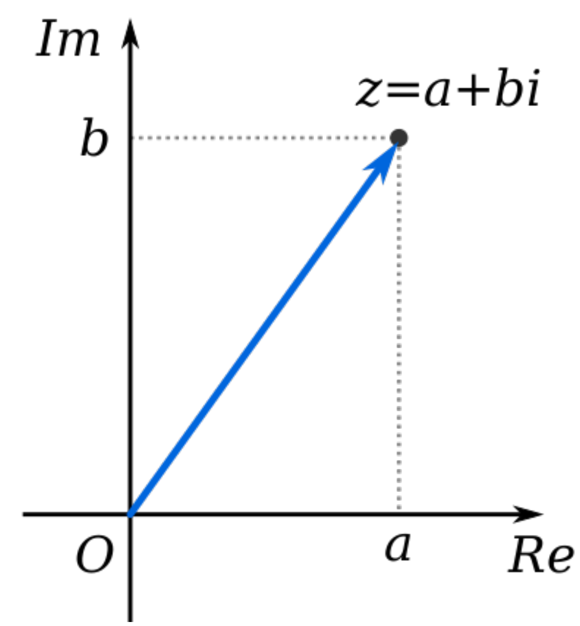
Complex Numbers & Fourier Transform

Complex Numbers

$$x^2 + 1 = 0$$

$$x = ?$$

$$x = \sqrt{-1} = i$$



Complex numbers: $x = a + bi$

Re(x)
real

Im(x)
imaginary

Complex Numbers: Operations

$$(a+bi)+(c+di)=(a+c)+(b+d)i$$

$$(a+bi)\cdot(c+di)=(ac-bd)+(ad+bc)i$$

$$\frac{1}{a+bi}=\frac{a}{a^2+b^2}-\frac{b}{a^2+b^2}i$$

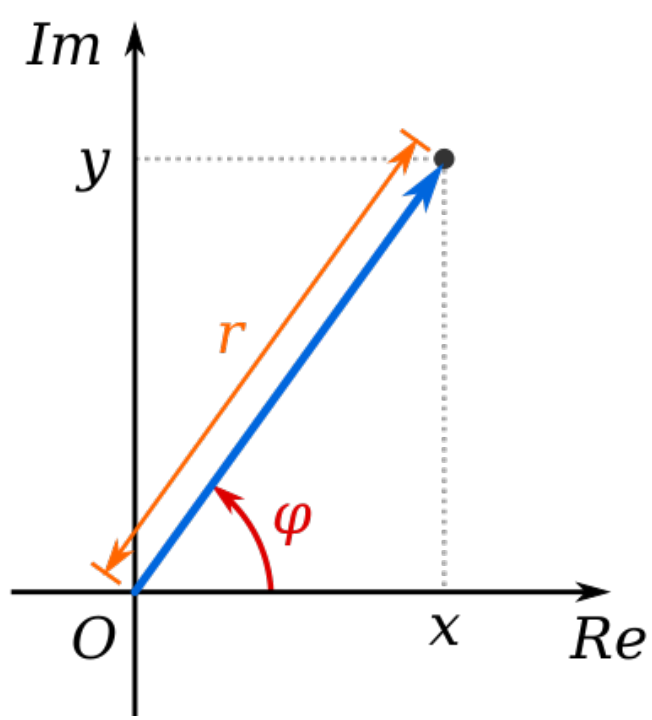
$$\vdots$$

Complex Numbers: Polar Form

$$x = a + bi = r \cdot e^{\varphi i}$$

$$r = \sqrt{a^2 + b^2}$$

$$\varphi = \arctan\left(\frac{b}{a}\right)$$

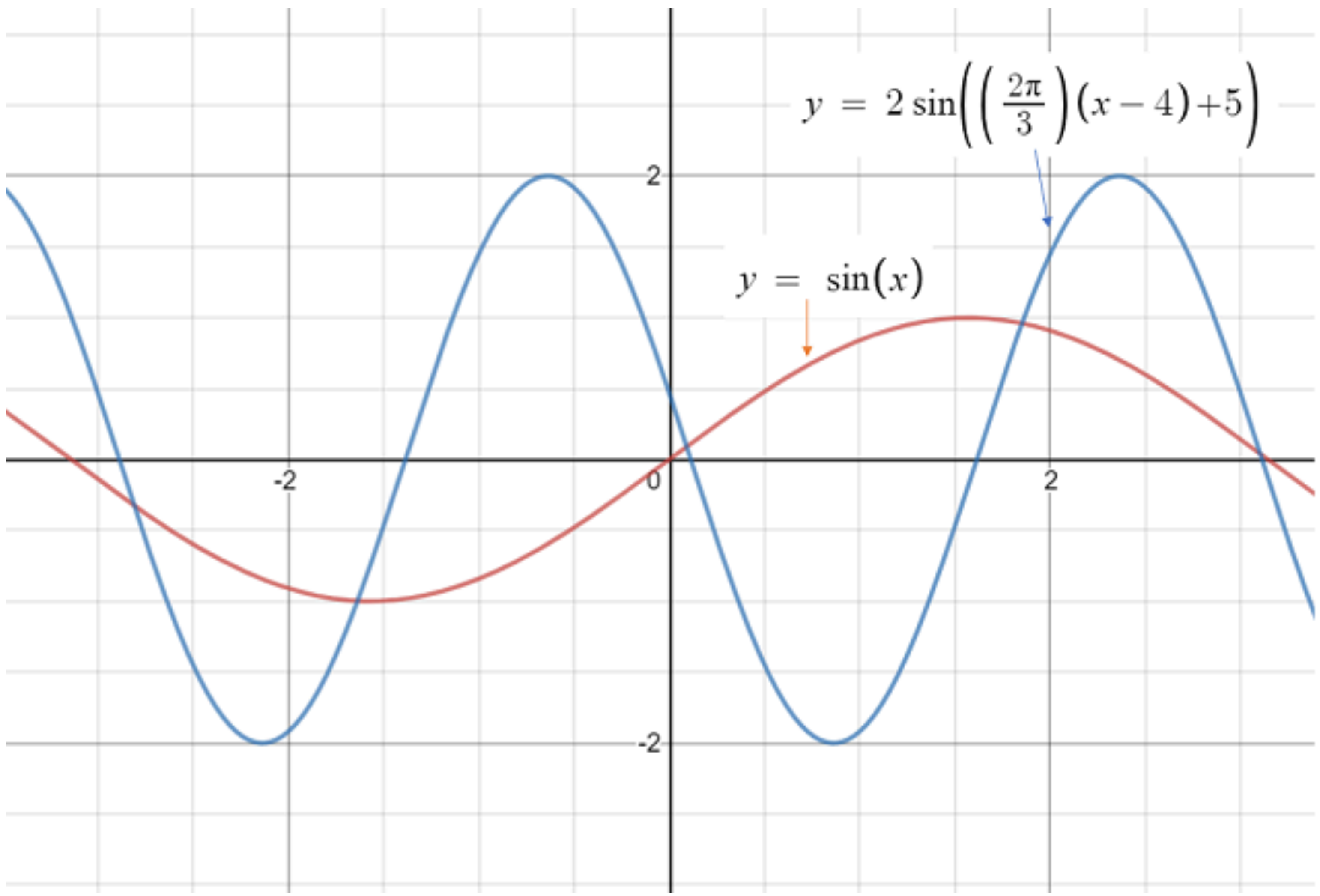


Euler's formula

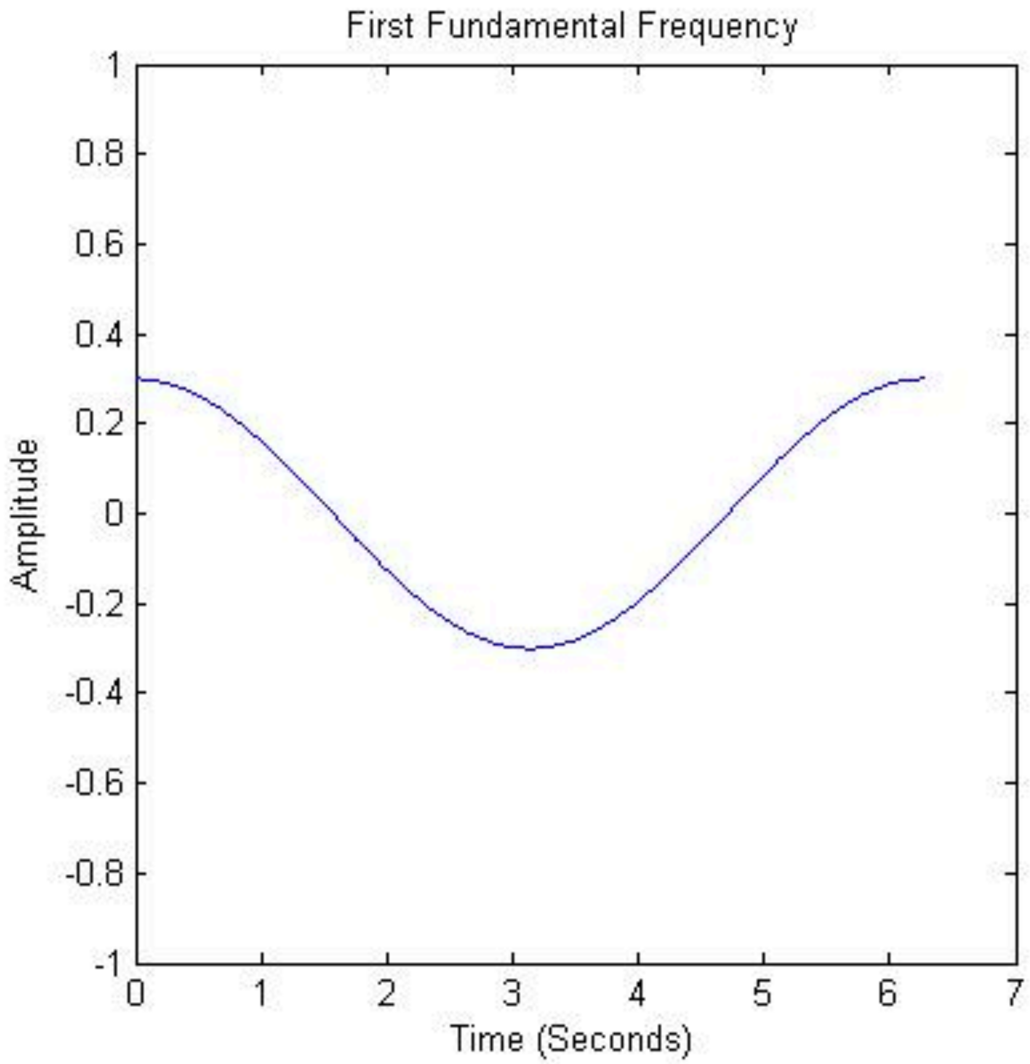
$$e^{a+bi} = e^a \cdot (\cos b + i \sin b)$$

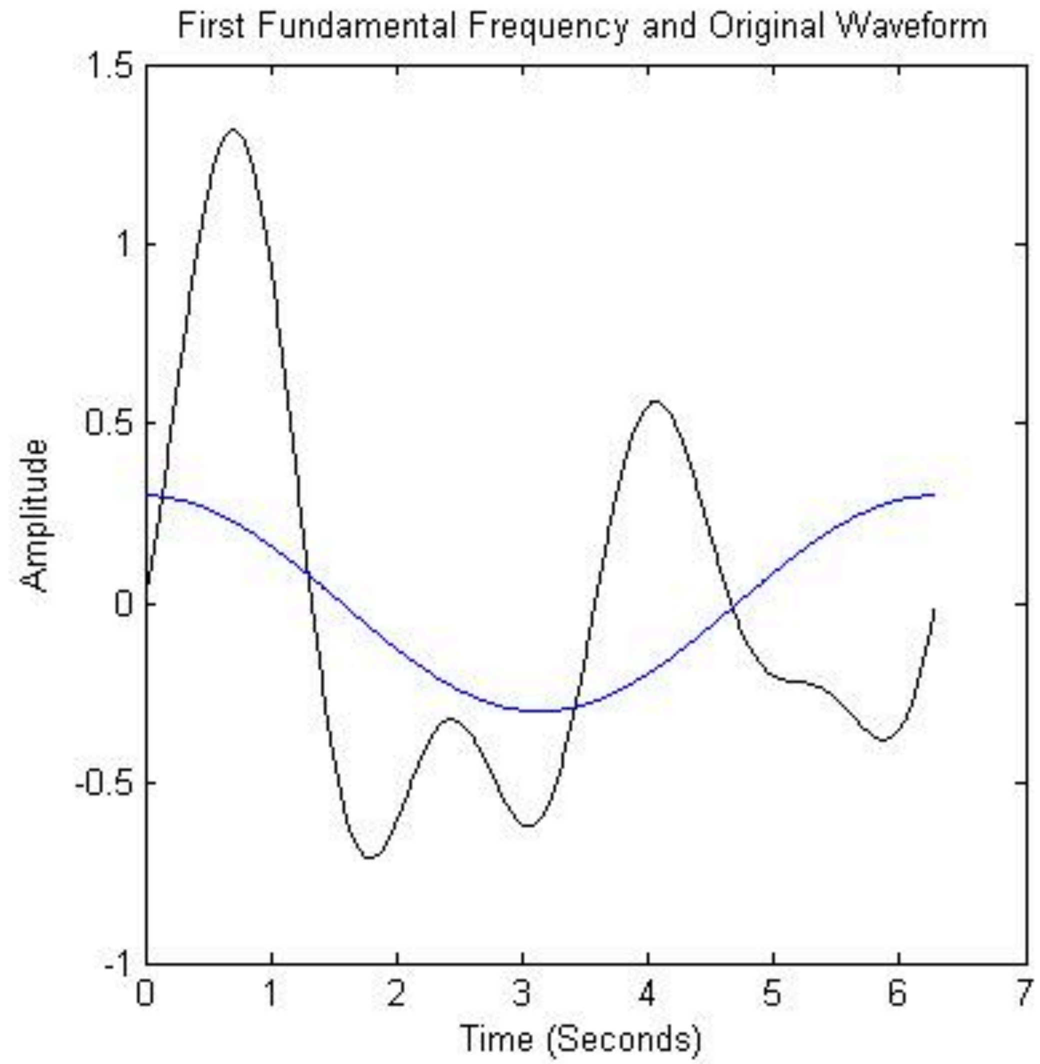
Sinusoidal functions

$$y = a\sin\left(\frac{2\pi}{b}(x-c)\right)+d$$

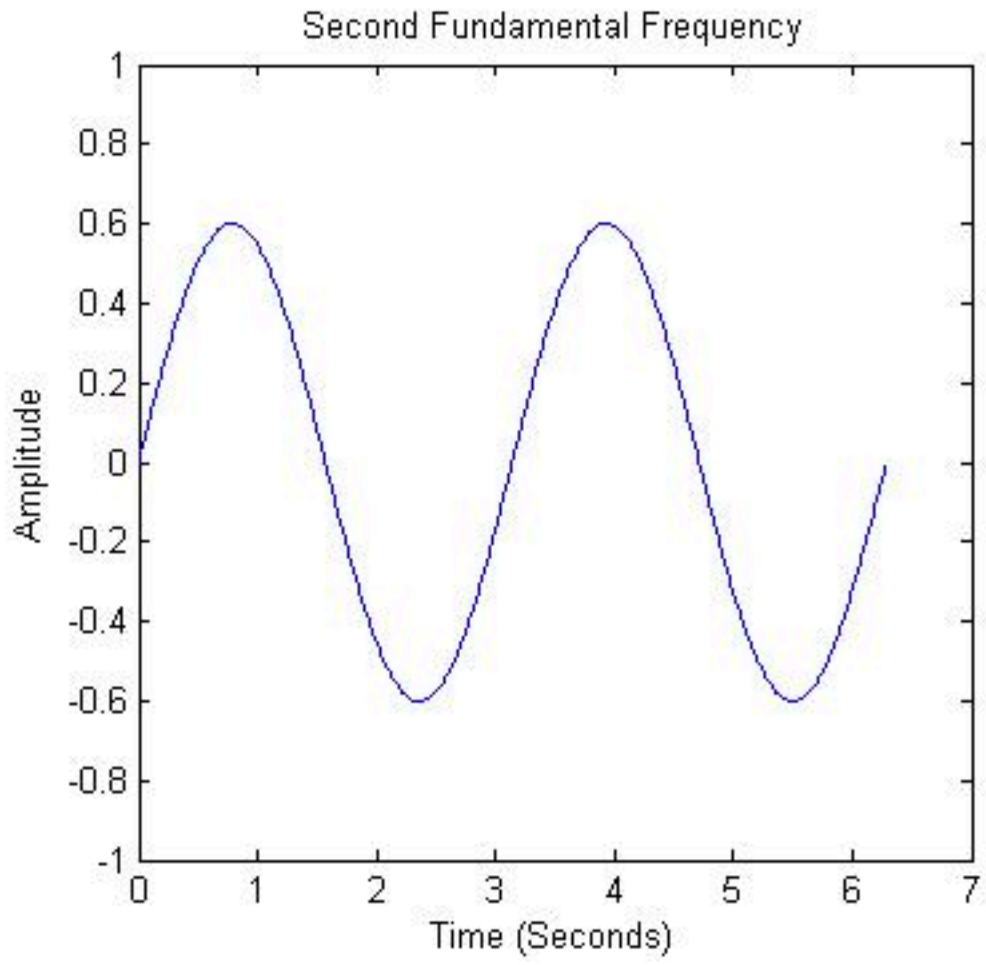


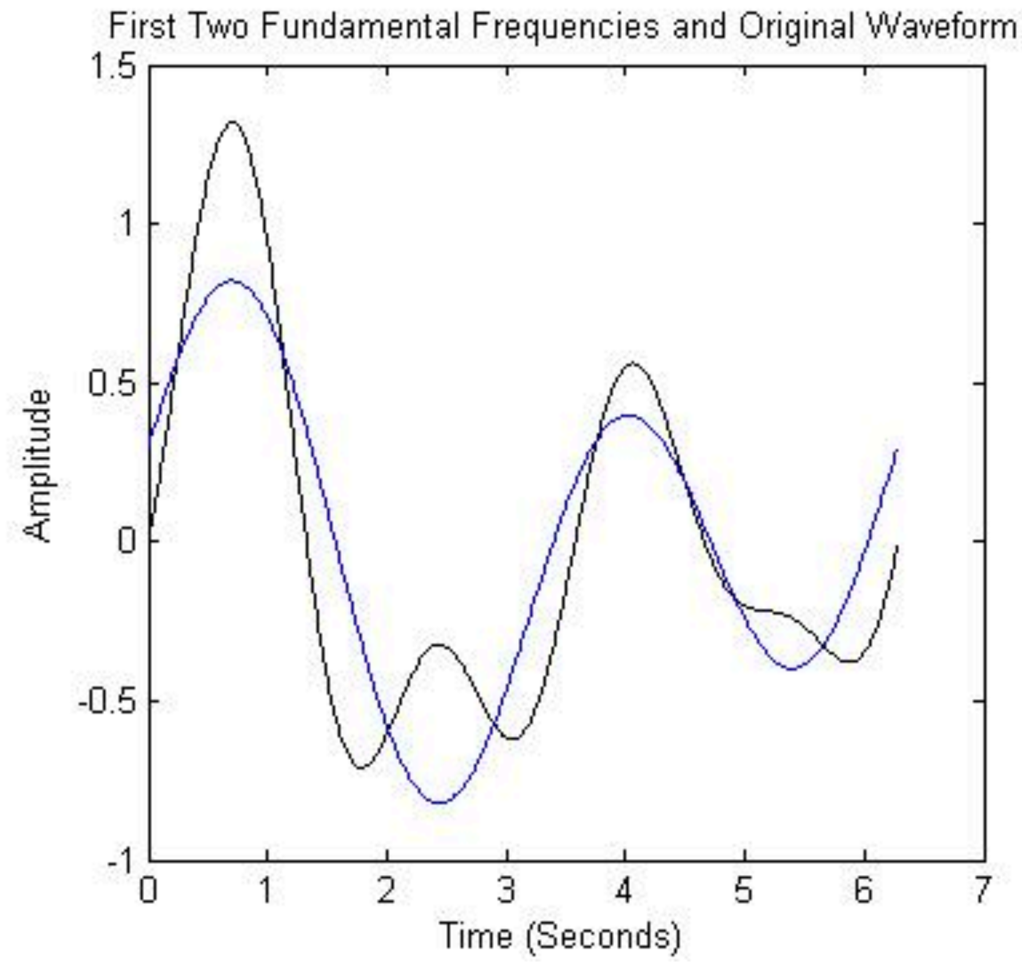
Sinusoidal functions



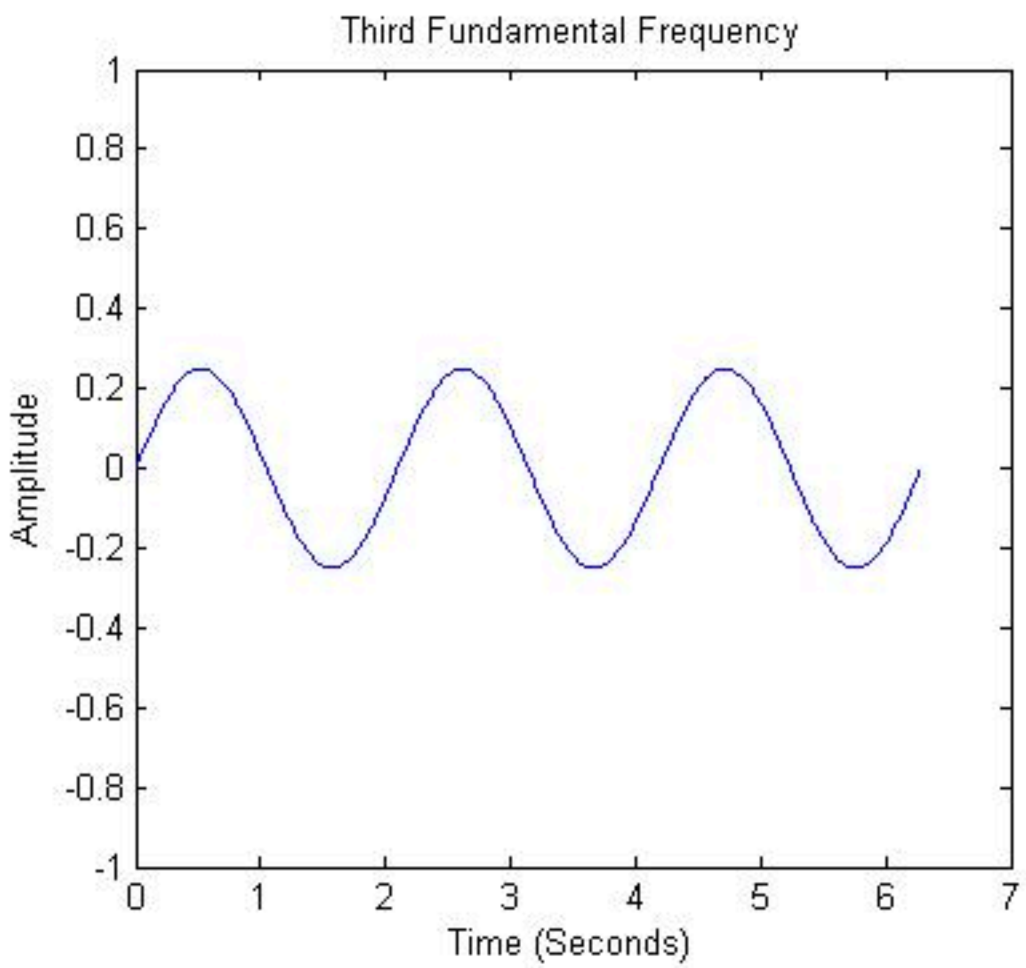


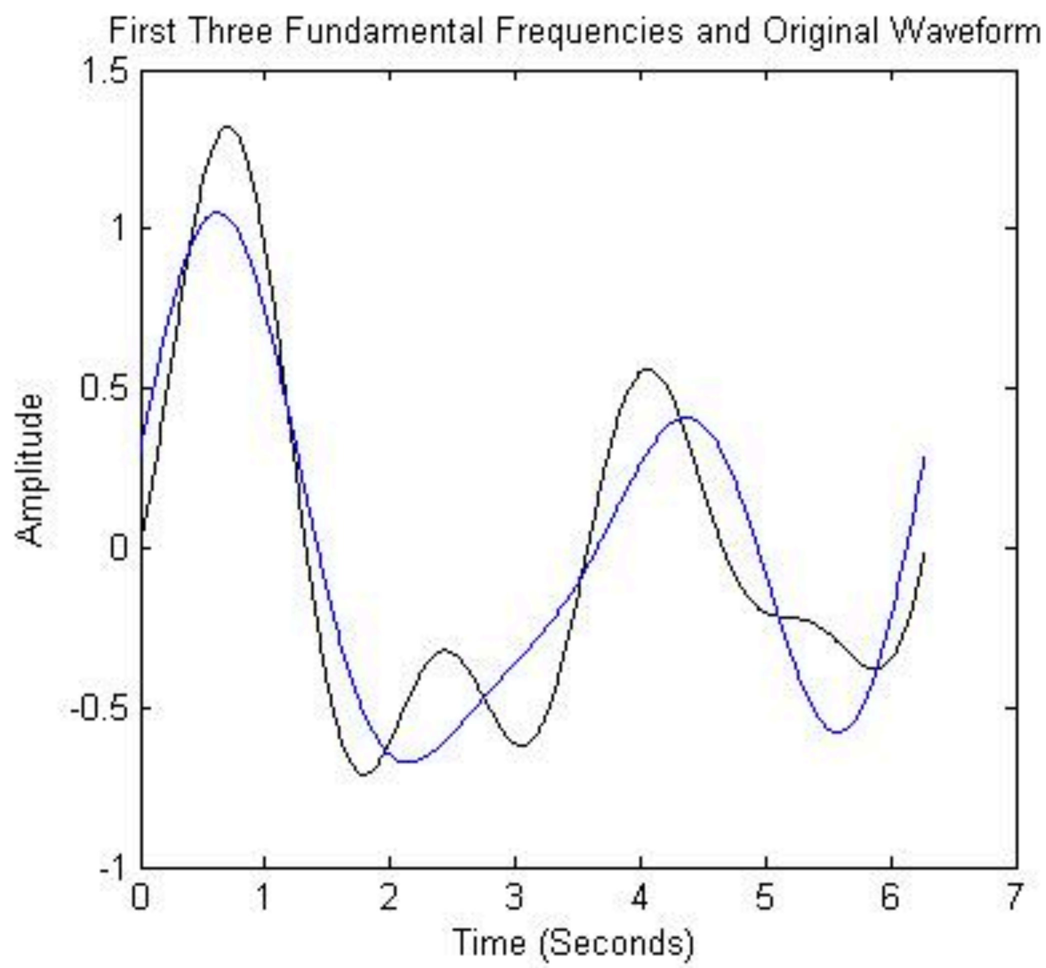
Sinusoidal functions





Sinusoidal functions





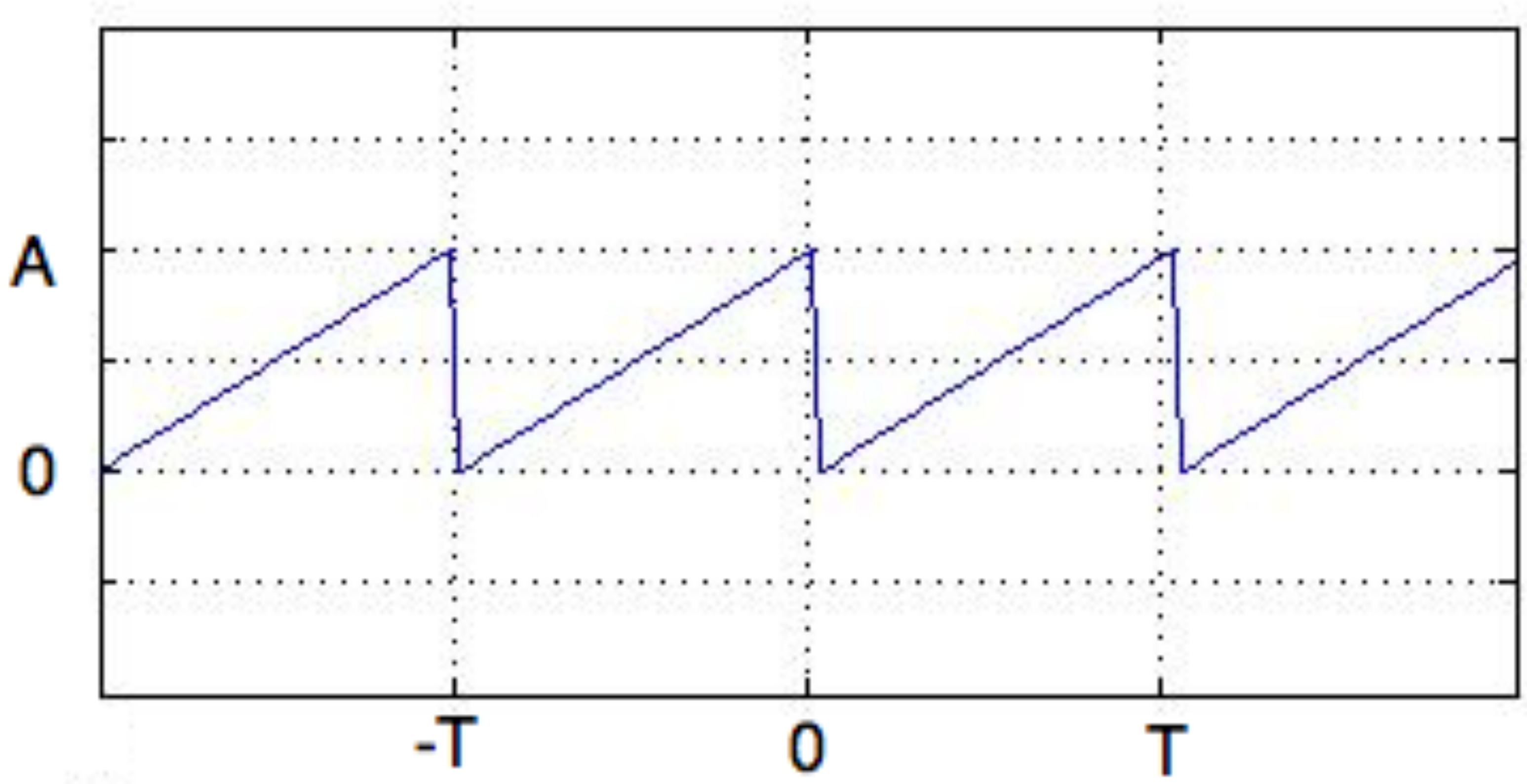
Fourier transform

$$f(t) = \sum_{n=-\infty}^{\infty} c_n \cdot e^{\frac{2\pi n}{T} ti}$$

$$c_n = \frac{1}{T} \int_0^T f(t) \cdot e^{-\frac{2\pi n}{T} ti} dt$$

Fourier transform of the saw function

$$f(t) = A \cdot \frac{t}{T} \quad , \quad 0 \leq t < T$$



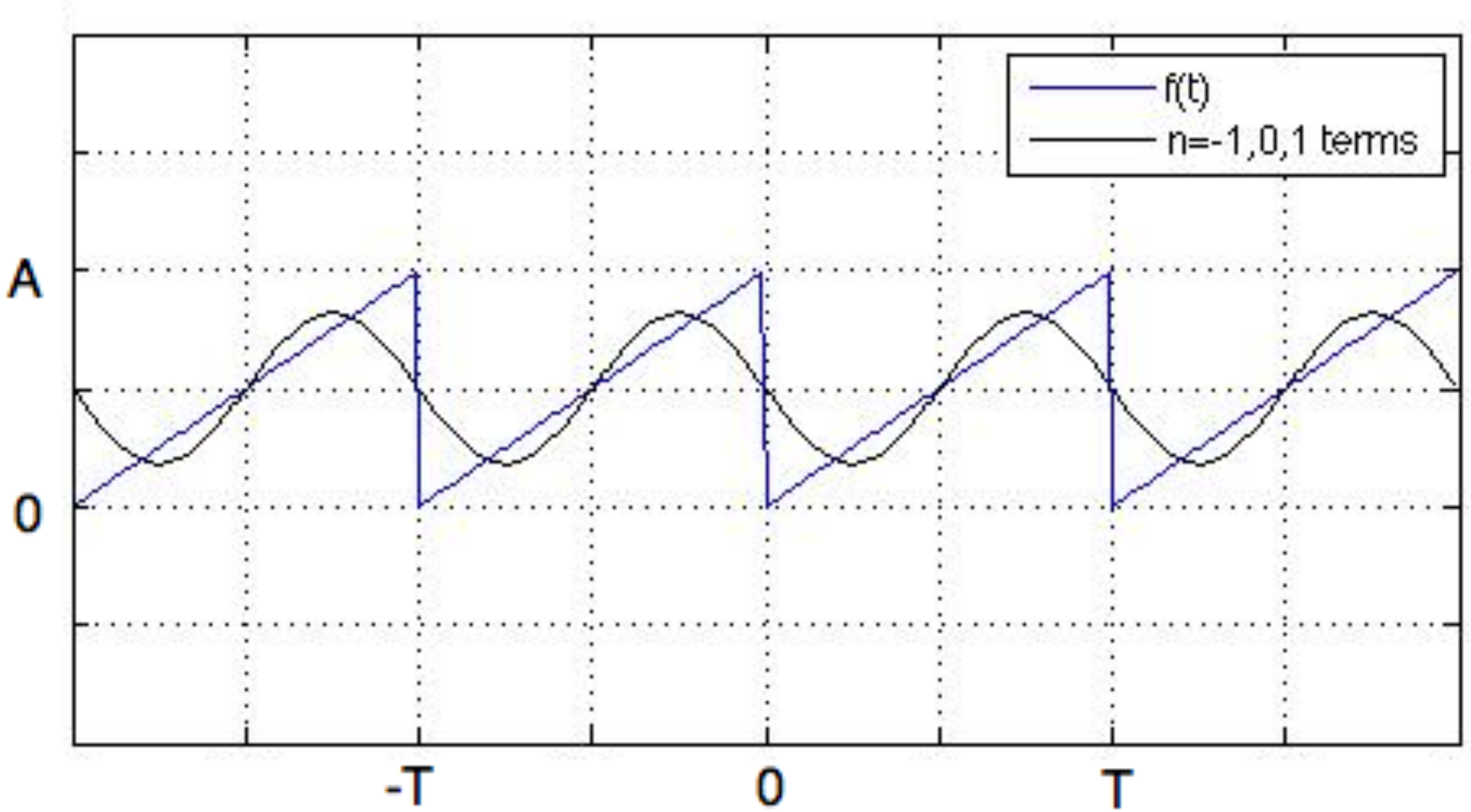
Coefficients:

$$c_0 = \frac{A}{2} \qquad c_n = \frac{A}{2\pi n} i$$

$$\boxed{c_n = \frac{1}{T} \int_0^T f(t) \cdot e^{-\frac{2\pi n}{T} t i} dt}$$

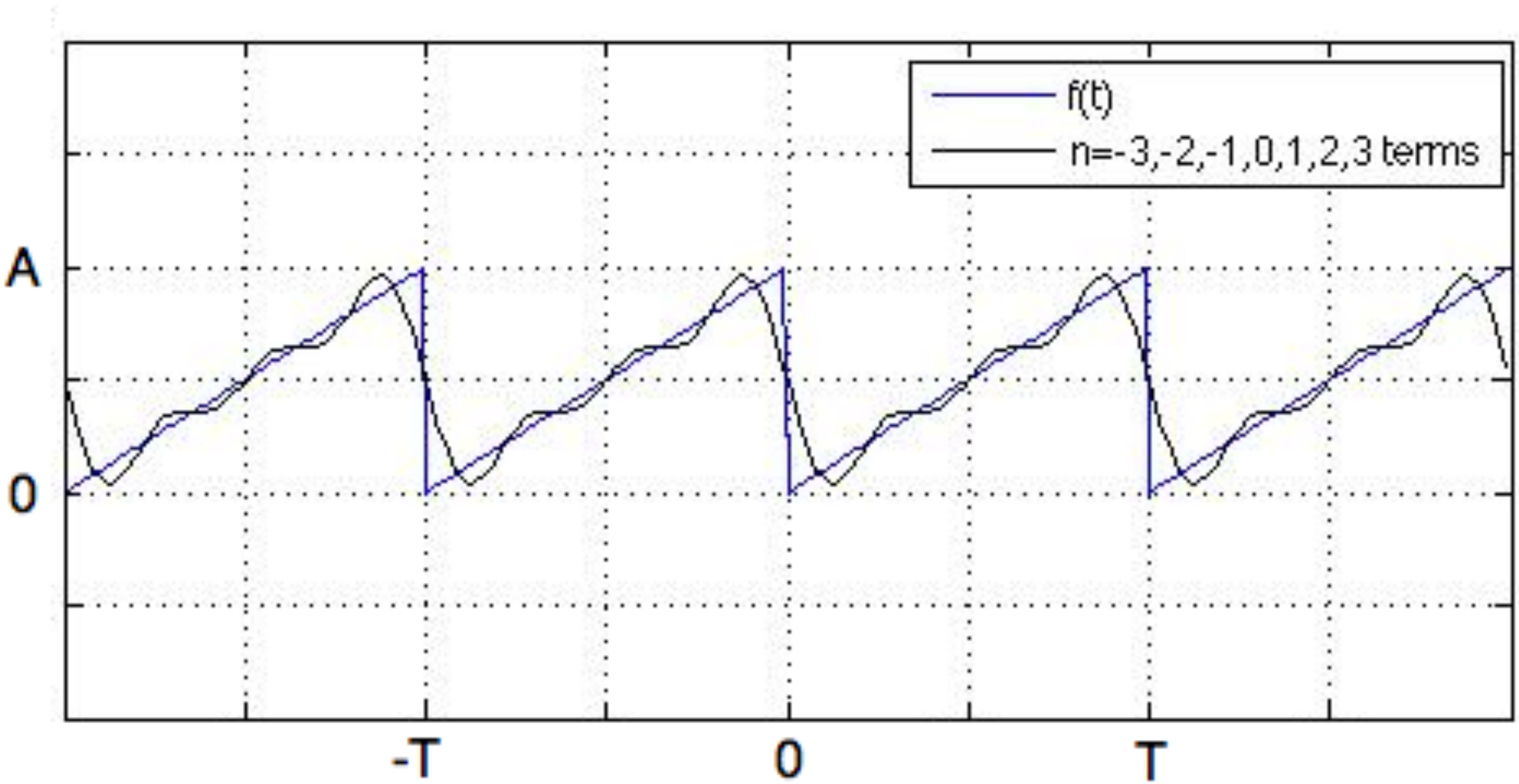
Fourier transform of the saw function

2 Terms



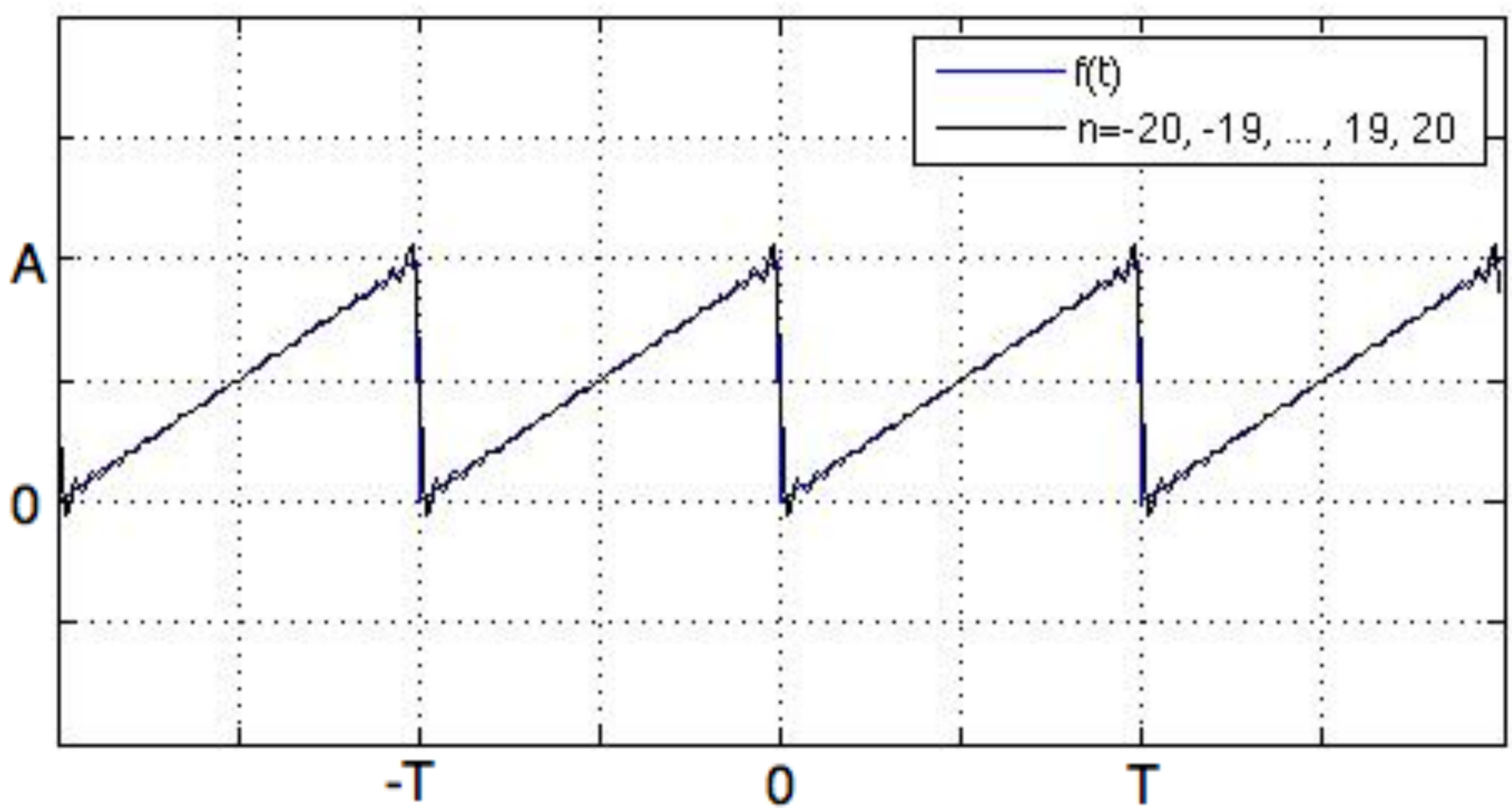
Fourier transform of the saw function

4 Terms



Fourier transform of the saw function

20 Terms



Discrete Fourier transform

When the data is discrete (pixels, measurements)

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{2\pi i}{N}kn}$$

Fast Fourier Transform (FFT) computes in $O(N \log N)$ instead of $O(N^2)$

Let's try...