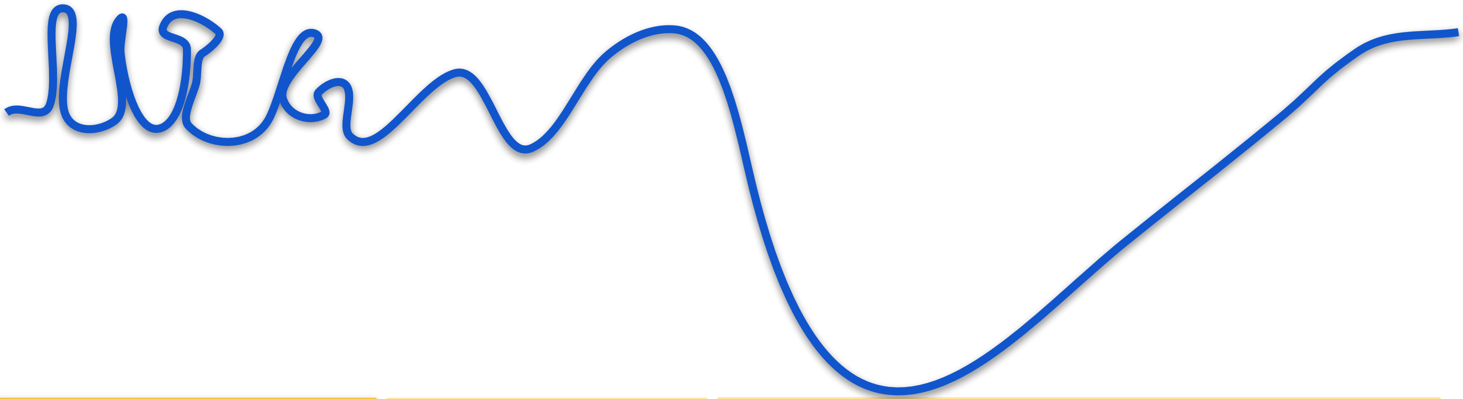
Computing with Signals



DA 623

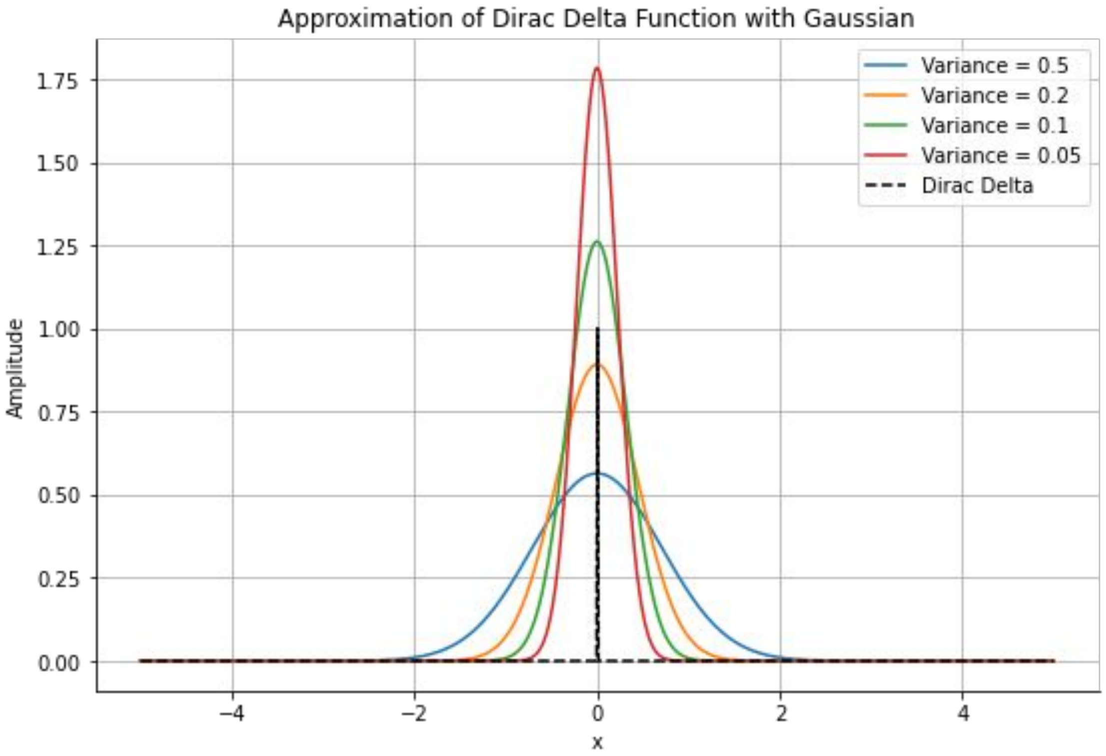
Jan - May 2024

IIT Guwahati

Instructors: Neeraj Sharma

Lecture-12-13
(and more)

$$g(x,t) = \frac{1}{\sqrt{2\pi t}} e^{\frac{-x^2}{2t}}, \ t > 0$$



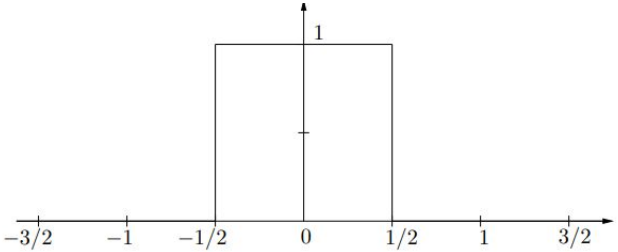
Dirac Delta

$$g(x,t) = \frac{1}{\sqrt{2\pi t}} e^{\frac{-x^2}{2t}}, \quad t > 0$$

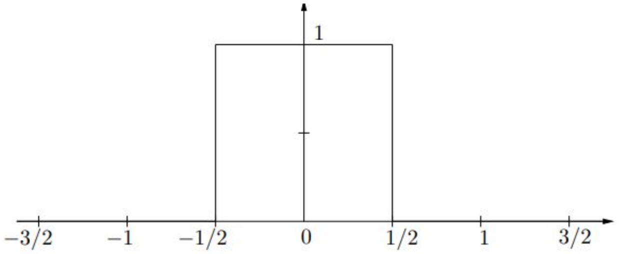
As t tends to 0, g(x,t) approximates a pulse of infinite height and infinitesimal width.
Such a signal is referred to as the Dirac Delta.

Fourier Transform of some typical signals

$$\Pi(t) = \begin{cases} 1 & |t| < 1/2 \\ 0 & |t| \geq 1/2 \end{cases}$$

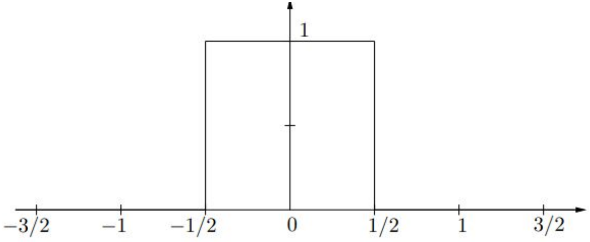


$$\Pi(t) = \begin{cases} 1 & |t| < 1/2 \\ 0 & |t| \geq 1/2 \end{cases}$$

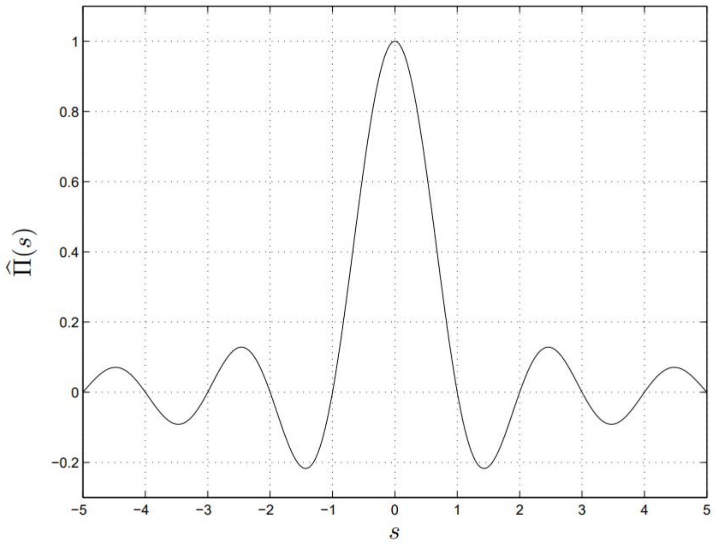


$$\hat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t} f(t)\, dt$$

$$\Pi(t) = \begin{cases} 1 & |t| < 1/2 \\ 0 & |t| \geq 1/2 \end{cases}$$

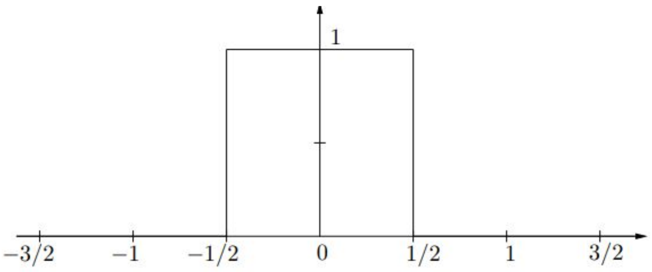


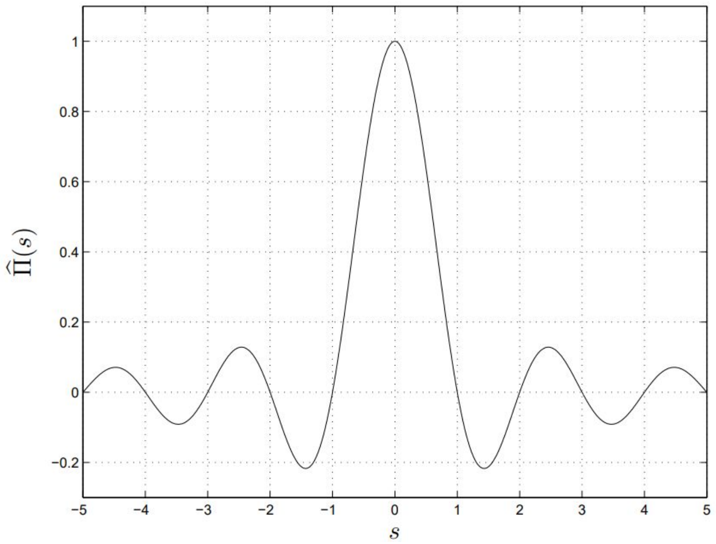
$$\widehat{\Pi}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t}\Pi(t)\,dt = \int_{-1/2}^{1/2} e^{-2\pi i s t} \cdot 1\,dt = \frac{\sin \pi s}{\pi s}$$



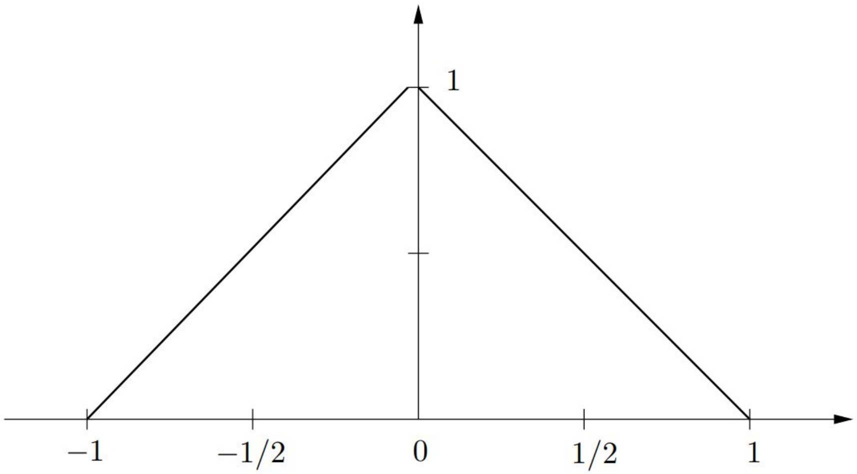
$$\Pi(t) = \begin{cases} 1 & |t| < 1/2 \\ 0 & |t| \geq 1/2 \end{cases}$$

$$\widehat{\Pi}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t}\,\Pi(t)\,dt = \int_{-1/2}^{1/2} e^{-2\pi i s t} \cdot 1\,dt = \frac{\sin \pi s}{\pi s}$$

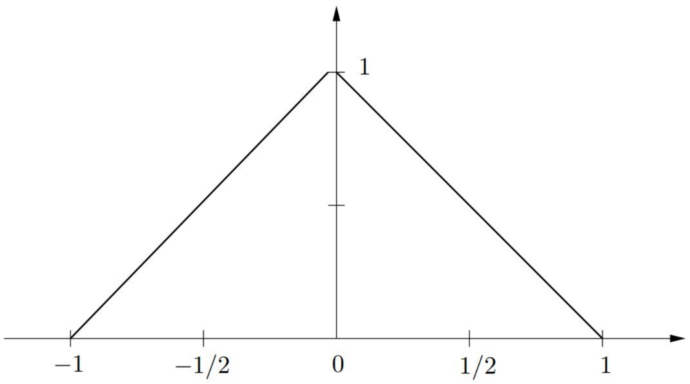




$$\Lambda(x) = \begin{cases} 1 - |x| & |x| \le 1 \\ 0 & \text{otherwise} \end{cases}$$

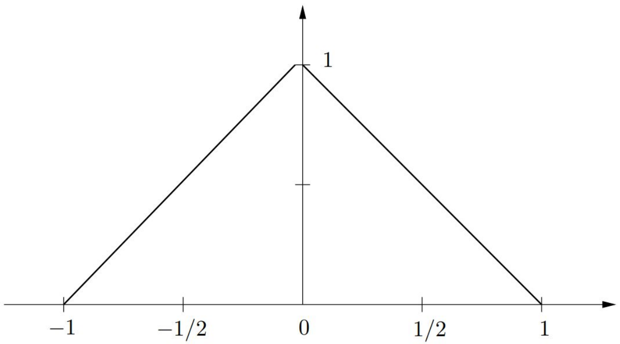


$$\Lambda(x) = \begin{cases} 1 - |x| & |x| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



$$\hat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t} f(t)\, dt$$

$$\Lambda(x) = \begin{cases} 1 - |x| & |x| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



$$\hat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t} f(t)\, dt$$

$$\begin{aligned}
\mathcal{F}\Lambda(s) &= \int_{-\infty}^{\infty} \Lambda(x) e^{-2\pi i s x}\, dx = \int_{-1}^{0} (1+x) e^{-2\pi i s x}\, dx + \int_{0}^{1} (1-x) e^{-2\pi i s x}\, dx \\
&= \left(\frac{1 + 2i\pi s}{4\pi^2 s^2} - \frac{e^{2\pi i s}}{4\pi^2 s^2} \right) - \left(\frac{2i\pi s - 1}{4\pi^2 s^2} + \frac{e^{-2\pi i s}}{4\pi^2 s^2} \right) \\
&= -\frac{e^{-2\pi i s}(e^{2\pi i s} - 1)^2}{4\pi^2 s^2} = -\frac{e^{-2\pi i s}(e^{\pi i s}(e^{\pi i s} - e^{-\pi i s}))^2}{4\pi^2 s^2} \\
&= -\frac{e^{-2\pi i s} e^{2\pi i s} (2i)^2 \sin^2 \pi s}{4\pi^2 s^2} = \left(\frac{\sin \pi s}{\pi s} \right)^2 = \operatorname{sinc}^2 s.
\end{aligned}$$

Linearity

$$\begin{aligned}\mathcal{F}(f+g)(s) &= \int_{-\infty}^{\infty} (f(x)+g(x))e^{-2\pi isx}\,dx \\ &= \int_{-\infty}^{\infty} f(x)e^{-2\pi isx}\,dx + \int_{-\infty}^{\infty} g(x)e^{-2\pi isx}\,dx = \mathcal{F}f(s) + \mathcal{F}g(s)\,.\end{aligned}$$

Shifting the signal

$$
\begin{aligned}
\int_{-\infty}^{\infty} f(t+b)e^{-2\pi i s t}\,dt &= \int_{-\infty}^{\infty} f(u)e^{-2\pi i s(u-b)}\,du \\
&\qquad \text{(substituting } u = t+b\text{; the limits still go from } -\infty \text{ to } \infty\text{)} \\
&= \int_{-\infty}^{\infty} f(u)e^{-2\pi i s u}e^{2\pi i s b}\,du \\
&= e^{2\pi i s b}\int_{-\infty}^{\infty} f(u)e^{-2\pi i s u}\,du = e^{2\pi i s b}\hat{f}(s).
\end{aligned}
$$

Scaling the signal

$$\int_{-\infty}^{\infty} f(at)e^{-2\pi ist}\,dt = \int_{-\infty}^{\infty} f(u)e^{-2\pi is(u/a)}\frac{1}{a}\,du$$

$$\text{(substituting } u = at\text{; the limits go the same way because } a > 0\text{)}$$

$$= \frac{1}{a}\int_{-\infty}^{\infty} f(u)e^{-2\pi i(s/a)u}\,du = \frac{1}{a}\mathcal{F}f\left(\frac{s}{a}\right)$$

Periodizing a function

$$\rho_p(x) = \sum_{k=-\infty}^{\infty} \rho(x - kp)$$

Periodizing a function using Diracs

$$\rho_p(x) = \sum_{k=-\infty}^{\infty} \rho(x - pk)$$

Periodizing a function using Diracs

$$\rho_p(x) = \sum_{k=-\infty}^{\infty} \rho(x - pk)$$

$$= \sum_{k=-\infty}^{\infty} \delta(x - kp) * \rho(x)$$

Periodizing a function using Diracs

$$\begin{aligned}
\rho_p(x) &= \sum_{k=-\infty}^{\infty} \rho(x - pk) \\
&= \sum_{k=-\infty}^{\infty} \delta(x - kp) * \rho(x) \\
&= \left(\sum_{k=-\infty}^{\infty} \delta(x - kp) \right) * \rho(x)
\end{aligned}$$

Periodizing a function

$$\rho_p(x) = \sum_{k=-\infty}^{\infty} \rho(x - kp)$$

- Shah function
- Comb function
- Train of Diracs

$$\text{Ш}_p(x) = \sum_{k=-\infty}^{\infty} \delta(x - kp)$$

$$\rho_p = \text{Ш}_p * \rho\,.$$

$$\text{Ш}_p(x) = \sum_{k=-\infty}^{\infty} \delta(x - kp) \quad \text{or} \quad \text{Ш}_p = \sum_{k=-\infty}^{\infty} \delta_{kp}$$

$$\langle Ш_p, \varphi \rangle = \left\langle \sum_{k=-\infty}^{\infty} \delta_{kp}, \varphi \right\rangle = \sum_{k=-\infty}^{\infty} \langle \delta_{kp}, \varphi \rangle = \sum_{k=-\infty}^{\infty} \varphi(kp)$$

$$Ш_p(x) = \sum_{k=-\infty}^{\infty} \delta(x - kp) \quad \text{or} \quad Ш_p = \sum_{k=-\infty}^{\infty} \delta_{kp}$$

Periodizing any function using the Shah function:

$$(f * \text{Ш}_p)(t) = \sum_{k=-\infty}^{\infty} f(t - pk)$$

Periodizing any function using the Shah function:

$$(f * \text{III}_p)(t) = \sum_{k=-\infty}^{\infty} f(t - pk)$$

Of special interest when f is zero for $|t| \geq p/2$ as then,

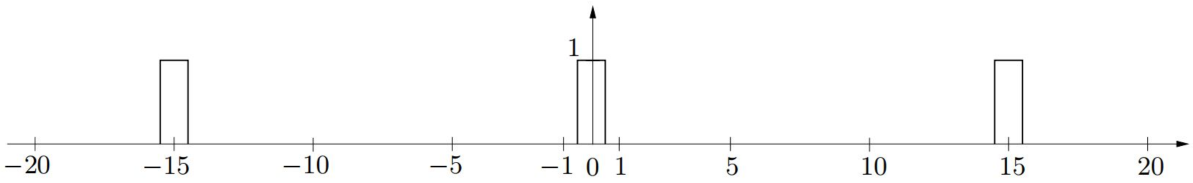
$$\boxed{\begin{gathered} \Pi_p f = f \\ \\ f = \Pi_p(f * \text{III}_p) \end{gathered}}$$

Example

$$(f * \text{Ш}_p)(t) = \sum_{k=-\infty}^{\infty} f(t - pk)$$

Can you recall we used this approach in earlier classes?

$$\Pi(t) = \begin{cases} 1 & |t| < 1/2 \\ 0 & |t| \geq 1/2 \end{cases}$$



The Shah function also provides one way sampling a function

$$f(x)\text{Ш}(x) = \sum_{k=-\infty}^{\infty} f(x)\delta(x-k) = \sum_{k=-\infty}^{\infty} f(k)\delta(x-k)$$

The Shah function provides one way to periodizing a function

$$(f * \text{Ш}_p)(t) = \sum_{k=-\infty}^{\infty} f(t-pk)$$

Sampling at arbitrary but regularly spaced points

$$f(x)\text{Ш}_p(x) = \sum_{k=-\infty}^{\infty} f(kp)\delta(x - kp)$$

Scaling the Shah function

$$\text{Ш}_p(x) = \sum_{k=-\infty}^{\infty} \delta(x - kp)$$

$$\text{Ш}(px) = \sum_{k=-\infty}^{\infty} \delta(px - k)$$

Fourier Transform of Shah function

$$\text{Ш}(x) = \sum_{k=-\infty}^{\infty} \delta(x-k)$$

$$\mathcal{F}\text{Ш}(s) = \sum_{k=-\infty}^{\infty} e^{-2\pi iks} = \sum_{k=-\infty}^{\infty} e^{2\pi iks} = \text{Ш}$$

Fourier Transform of Shah function

$$\text{Ш}(x) = \sum_{k=-\infty}^{\infty} \delta(x-k)$$

$$\mathcal{F}\text{Ш}(s) = \sum_{k=-\infty}^{\infty} e^{-2\pi iks} = \sum_{k=-\infty}^{\infty} e^{2\pi iks} = \text{Ш}$$

$$\boxed{\sum_{n=-N}^{N} e^{2\pi int}}$$

$$\text{Ш}(x) = \sum_{k=-\infty}^{\infty} \delta(x - k)$$

$$\boxed{\begin{aligned} \mathcal{F}\text{Ш}_p(s) &= \frac{1}{p}\mathcal{F}\left(\text{Ш}\left(\frac{x}{p}\right)\right) \\ &= \frac{1}{p}p\mathcal{F}\text{Ш}(ps) \quad \text{(stretch theorem)} \\ &= \text{Ш}(ps) \\ &= \frac{1}{p}\text{Ш}_{1/p}(s) \end{aligned}}$$

$$\mathcal{F}\text{Ш}(s) = \sum_{k=-\infty}^{\infty} e^{-2\pi iks} = \sum_{k=-\infty}^{\infty} e^{2\pi iks} = \text{Ш}$$

$$\mathcal{F}f = \Pi_p(\mathcal{F}f * Ш_p)$$

$$\mathcal{F}f = \Pi_p(\mathcal{F}f * \text{III}_p)$$

$f(t) = \mathcal{F}^{-1}\mathcal{F}f(t)$

$$f(t) = \mathcal{F}^{-1}\mathcal{F}f(t) = \mathcal{F}^{-1}(\Pi_p(\mathcal{F}f * \text{III}_p))(t)$$

$$
\begin{aligned}
f(t) = \mathcal{F}^{-1}\mathcal{F}f(t) &= \mathcal{F}^{-1}(\Pi_p(\mathcal{F}f * \text{III}_p))(t) \\
&= \mathcal{F}^{-1}\Pi_p(t) * \mathcal{F}^{-1}(\mathcal{F}f * \text{III}_p)(t) \\
&\qquad \text{(taking } \mathcal{F}^{-1} \text{ turns multiplication into convolution)}
\end{aligned}
$$

$$
\begin{aligned}
f(t) = \mathcal{F}^{-1}\mathcal{F}f(t) &= \mathcal{F}^{-1}(\Pi_p(\mathcal{F}f * \text{Ш}_p))(t) \\
&= \mathcal{F}^{-1}\Pi_p(t) * \mathcal{F}^{-1}(\mathcal{F}f * \text{Ш}_p)(t) \\
&\qquad \text{(taking } \mathcal{F}^{-1} \text{ turns multiplication into convolution)} \\
&= \mathcal{F}^{-1}\Pi_p(t) * (\mathcal{F}^{-1}\mathcal{F}f(t) \cdot \mathcal{F}^{-1}\text{Ш}_p(t)) \\
&\qquad \text{(ditto, except it's convolution turning into multiplication)}
\end{aligned}
$$

$$
\begin{aligned}
f(t) = \mathcal{F}^{-1}\mathcal{F}f(t) &= \mathcal{F}^{-1}(\Pi_p(\mathcal{F}f * \text{III}_p))(t) \\
&= \mathcal{F}^{-1}\Pi_p(t) * \mathcal{F}^{-1}(\mathcal{F}f * \text{III}_p)(t) \\
&\qquad \text{(taking } \mathcal{F}^{-1} \text{ turns multiplication into convolution)} \\
&= \mathcal{F}^{-1}\Pi_p(t) * (\mathcal{F}^{-1}\mathcal{F}f(t) \cdot \mathcal{F}^{-1}\text{III}_p(t)) \\
&\qquad \text{(ditto, except it's convolution turning into multiplication)} \\
&= p \operatorname{sinc} pt * (f(t) \cdot \frac{1}{p}\text{III}_{1/p}(t))
\end{aligned}
$$

$$
\begin{aligned}
f(t) = \mathcal{F}^{-1}\mathcal{F}f(t) &= \mathcal{F}^{-1}(\Pi_p(\mathcal{F}f * \text{III}_p))(t) \\
&= \mathcal{F}^{-1}\Pi_p(t) * \mathcal{F}^{-1}(\mathcal{F}f * \text{III}_p)(t) \\
&\qquad \text{(taking } \mathcal{F}^{-1} \text{ turns multiplication into convolution)} \\
&= \mathcal{F}^{-1}\Pi_p(t) * (\mathcal{F}^{-1}\mathcal{F}f(t) \cdot \mathcal{F}^{-1}\text{III}_p(t)) \\
&\qquad \text{(ditto, except it's convolution turning into multiplication)} \\
&= p \operatorname{sinc} pt * \left(f(t) \cdot \frac{1}{p}\text{III}_{1/p}(t)\right) \\
&= \operatorname{sinc} pt * \sum_{k=-\infty}^{\infty} f\left(\frac{k}{p}\right) \delta\left(t - \frac{k}{p}\right) \quad \text{(the sampling property of } \text{III}_p\text{)}
\end{aligned}
$$

$$
\begin{aligned}
f(t) = \mathcal{F}^{-1}\mathcal{F}f(t) &= \mathcal{F}^{-1}(\Pi_p(\mathcal{F}f * Ш_p))(t) \\
&= \mathcal{F}^{-1}\Pi_p(t) * \mathcal{F}^{-1}(\mathcal{F}f * Ш_p)(t) \\
&\qquad \text{(taking } \mathcal{F}^{-1} \text{ turns multiplication into convolution)} \\
&= \mathcal{F}^{-1}\Pi_p(t) * (\mathcal{F}^{-1}\mathcal{F}f(t) \cdot \mathcal{F}^{-1}Ш_p(t)) \\
&\qquad \text{(ditto, except it's convolution turning into multiplication)} \\
&= p \operatorname{sinc} pt * \left(f(t) \cdot \frac{1}{p}Ш_{1/p}(t)\right) \\
&= \operatorname{sinc} pt * \sum_{k=-\infty}^{\infty} f\left(\frac{k}{p}\right) \delta\left(t - \frac{k}{p}\right) \quad \text{(the sampling property of } Ш_p\text{)} \\
&= \sum_{k=-\infty}^{\infty} f\left(\frac{k}{p}\right) \operatorname{sinc} pt * \delta\left(t - \frac{k}{p}\right)
\end{aligned}
$$

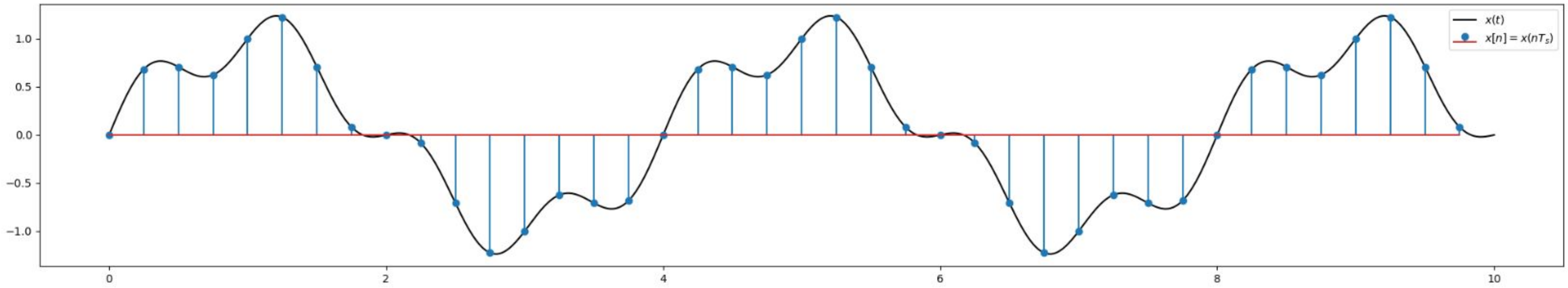
$$
\begin{aligned}
f(t) = \mathcal{F}^{-1}\mathcal{F}f(t) &= \mathcal{F}^{-1}(\Pi_p(\mathcal{F}f * \text{III}_p))(t) \\
&= \mathcal{F}^{-1}\Pi_p(t) * \mathcal{F}^{-1}(\mathcal{F}f * \text{III}_p)(t) \\
&\qquad \text{(taking } \mathcal{F}^{-1} \text{ turns multiplication into convolution)} \\
&= \mathcal{F}^{-1}\Pi_p(t) * (\mathcal{F}^{-1}\mathcal{F}f(t) \cdot \mathcal{F}^{-1}\text{III}_p(t)) \\
&\qquad \text{(ditto, except it's convolution turning into multiplication)} \\
&= p \operatorname{sinc} pt * (f(t) \cdot \frac{1}{p}\text{III}_{1/p}(t)) \\
&= \operatorname{sinc} pt * \sum_{k=-\infty}^{\infty} f\left(\frac{k}{p}\right) \delta\left(t - \frac{k}{p}\right) \quad \text{(the sampling property of } \text{III}_p\text{)} \\
&= \sum_{k=-\infty}^{\infty} f\left(\frac{k}{p}\right) \operatorname{sinc} pt * \delta\left(t - \frac{k}{p}\right) \\
&= \sum_{k=-\infty}^{\infty} f\left(\frac{k}{p}\right) \operatorname{sinc} p\left(t - \frac{k}{p}\right) \quad \text{(the sifting property of } \delta\text{)}
\end{aligned}
$$

$$f(t) = \sum_{k=-\infty}^{\infty} f\left(\frac{k}{p}\right) \operatorname{sinc} p \left(t - \frac{k}{p}\right) \quad \text{(the sifting property of } \delta\text{)}$$

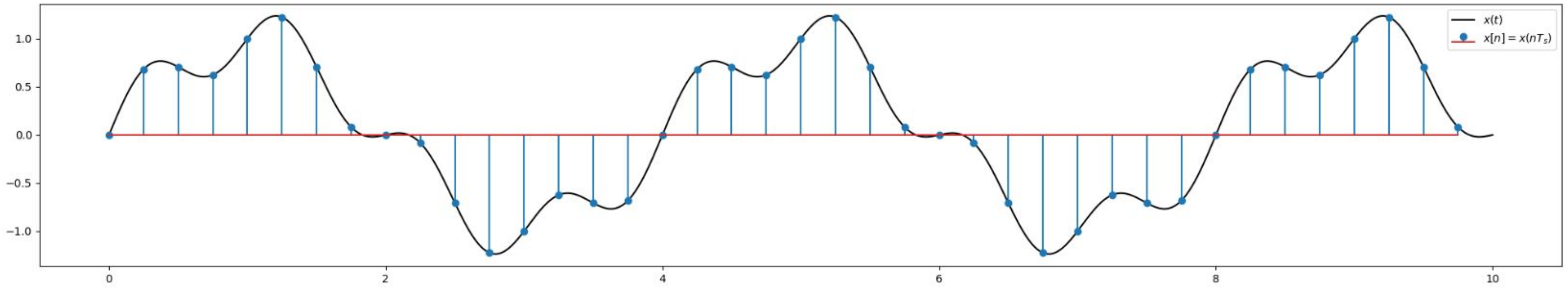
$$\downarrow$$

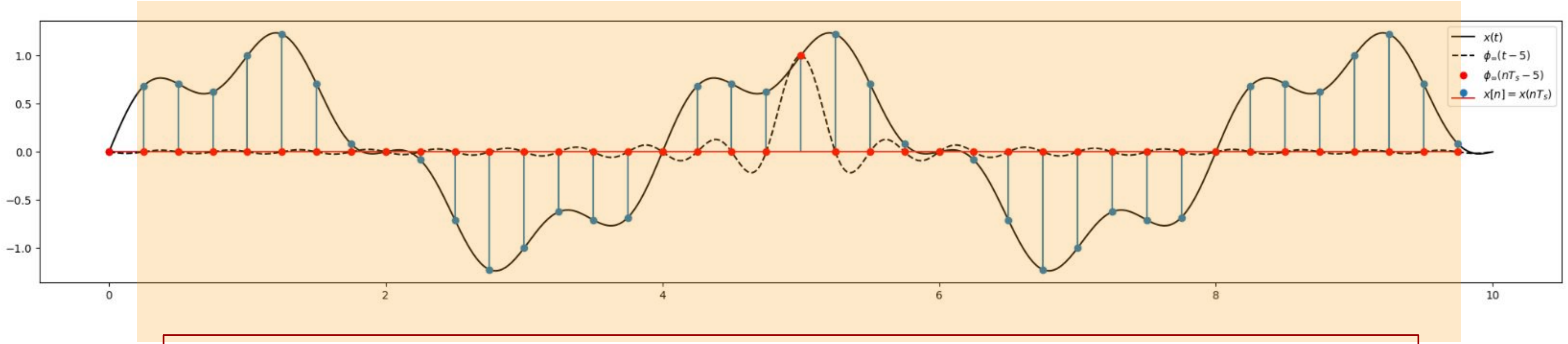
$$\boxed{f(t) = \sum_{k=-\infty}^{\infty} f(t_k) \operatorname{sinc} p(t - t_k) \quad , \quad t_k = \frac{k}{p}}$$

- Shannon Sampling Theorem (1949)
- Whittaker Sampling Formula (1915, 1935)

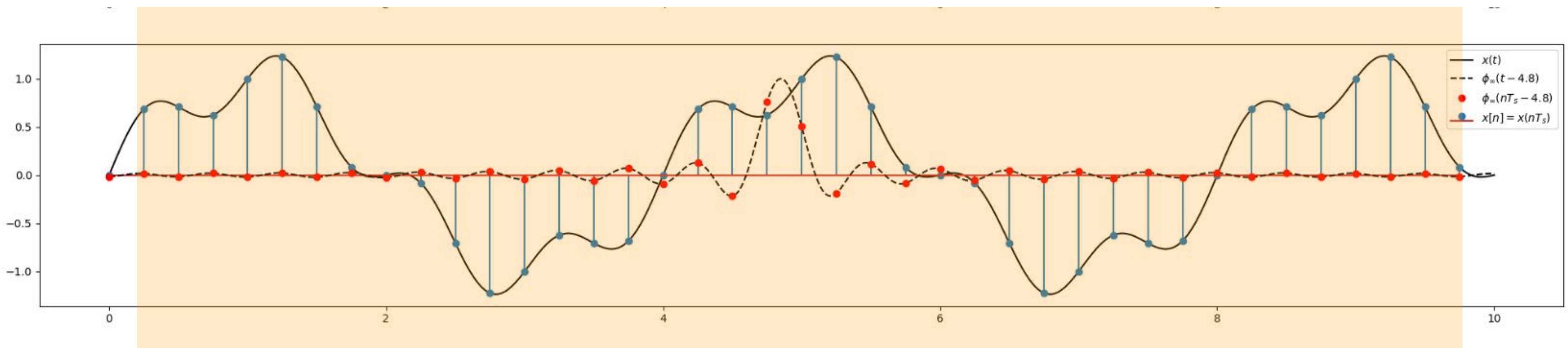


Source: https://staff.fnwi.uva.nl/r.vandenboomgaard/SignalProcessing/Applications/Sampling/interpolation.html





$$f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k)$$



$$f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k) \quad , \quad t_k = \frac{k}{p}$$

$$\boxed{f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k)\quad , \quad t_k = \frac{k}{p}}$$

Discrete samples of f(t)

Interpolating function

Bandwidth of f(t)

Discrete instants at which f(t) was sampled

$$f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k) \quad , \quad t_k = \frac{k}{p}$$

Discrete samples of f(t)

Interpolating function

Bandwidth of f(t)

Discrete instants at which f(t) was sampled

Our Discrete world.

Shannon sampling and reconstruction formula

$$f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k) \quad , \quad t_k = \frac{k}{p}$$

Continuous world

Discrete samples of f(t)

Interpolating function

Bandwidth of f(t)

Discrete instants at which f(t) was sampled

Our Discrete world.

Enables switching between the discrete, $f(t_k)$, and continuous world, $f(t)$. Without any error!

$$\boxed{f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k)\,, \qquad t_k = \frac{k}{p}}$$

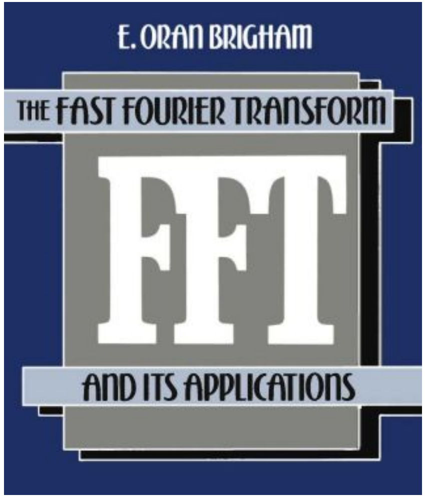
- Nyquist Sampling
- Shannon Sampling Theorem, 1940s
- Whittaker Sampling Formula

$$\boxed{f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k)\,, \qquad t_k = \frac{k}{p}}$$

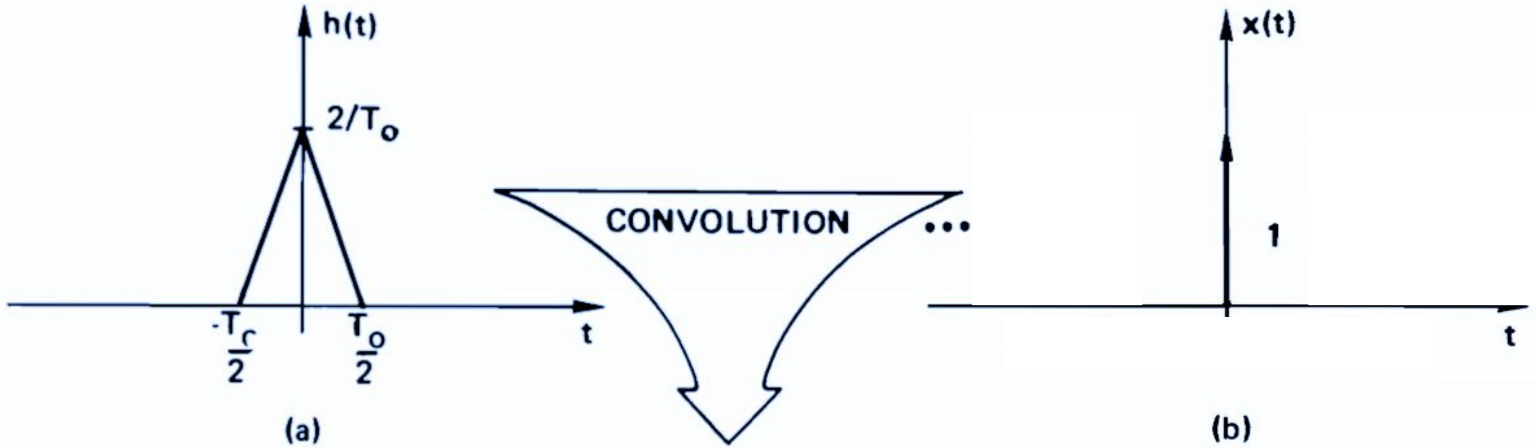
- Nyquist Sampling
- Shannon Sampling Theorem, 1940s
- Whittaker Sampling Formula

Sampling and Interpolation

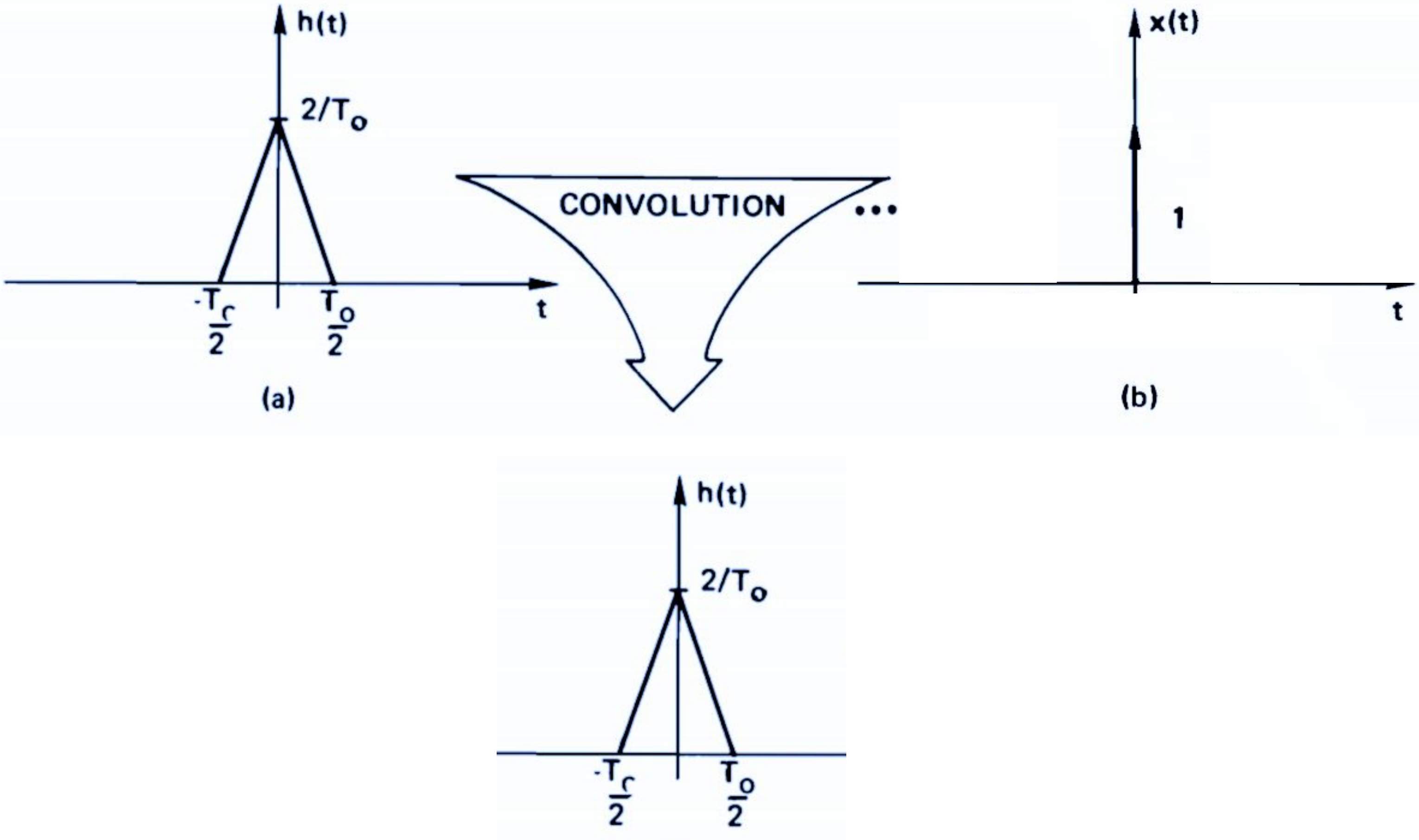
through visualizations



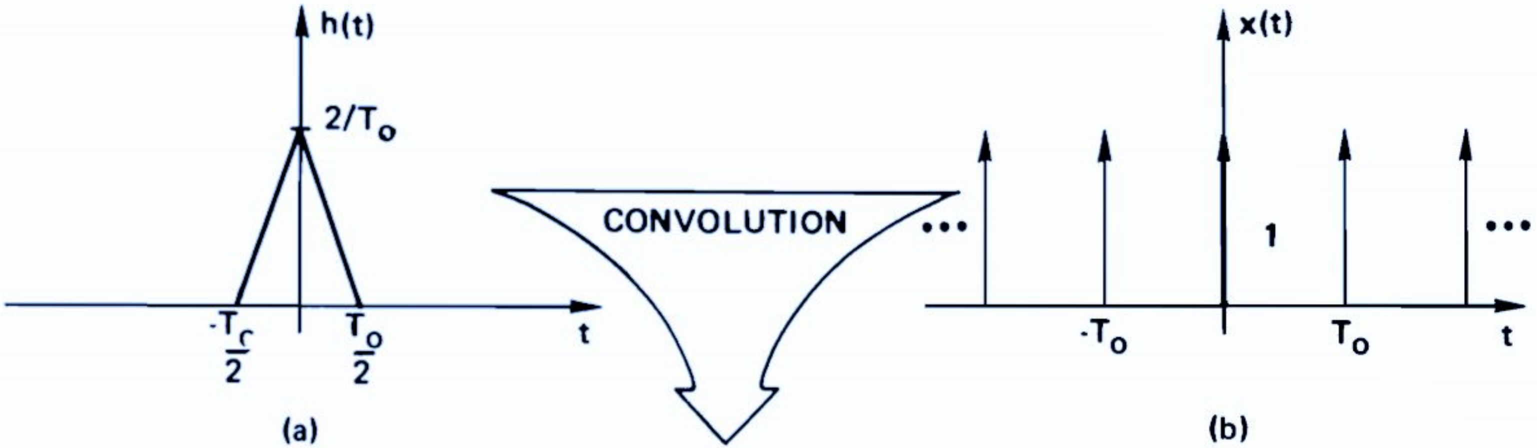
time-domain



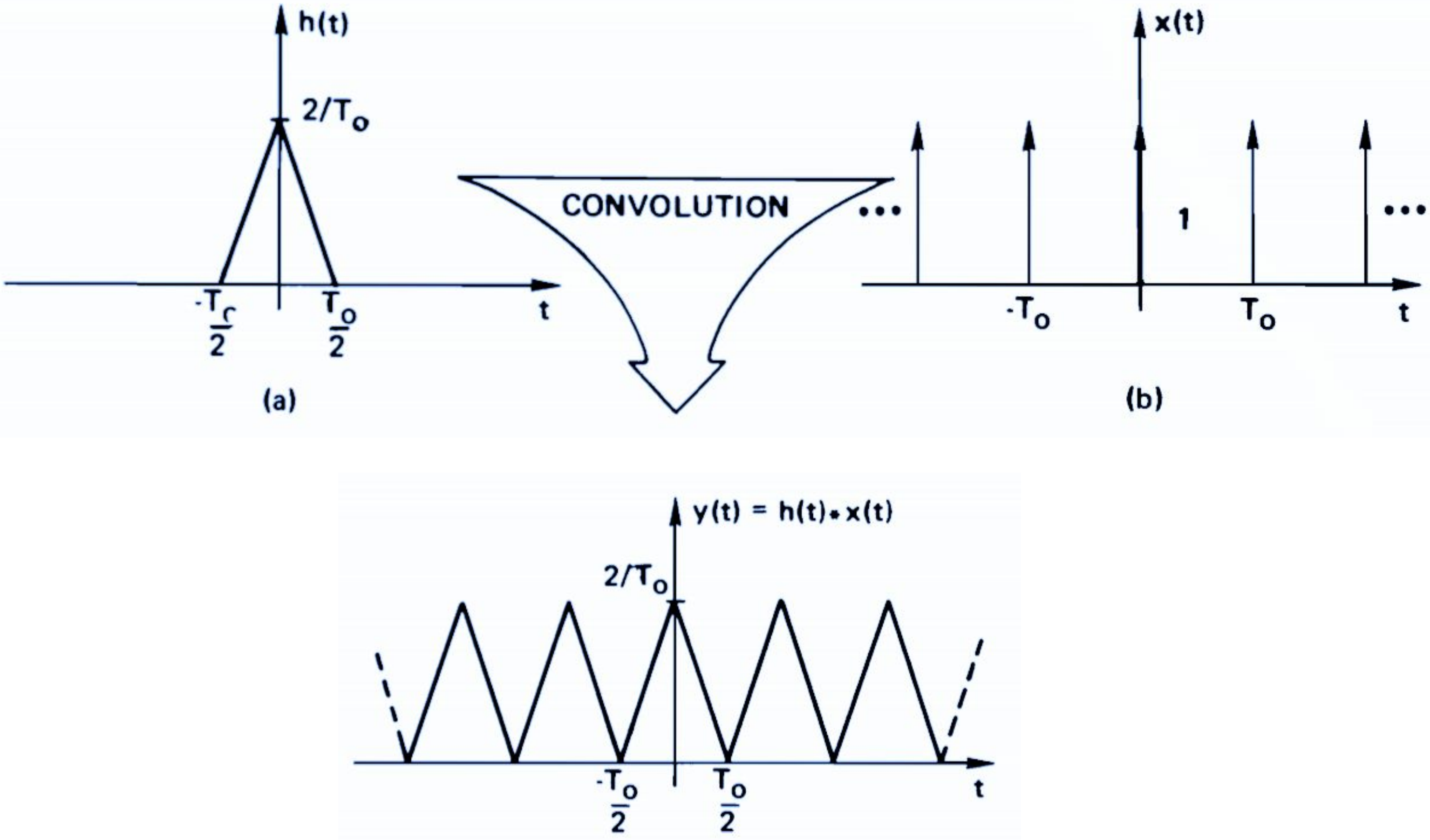
time-domain



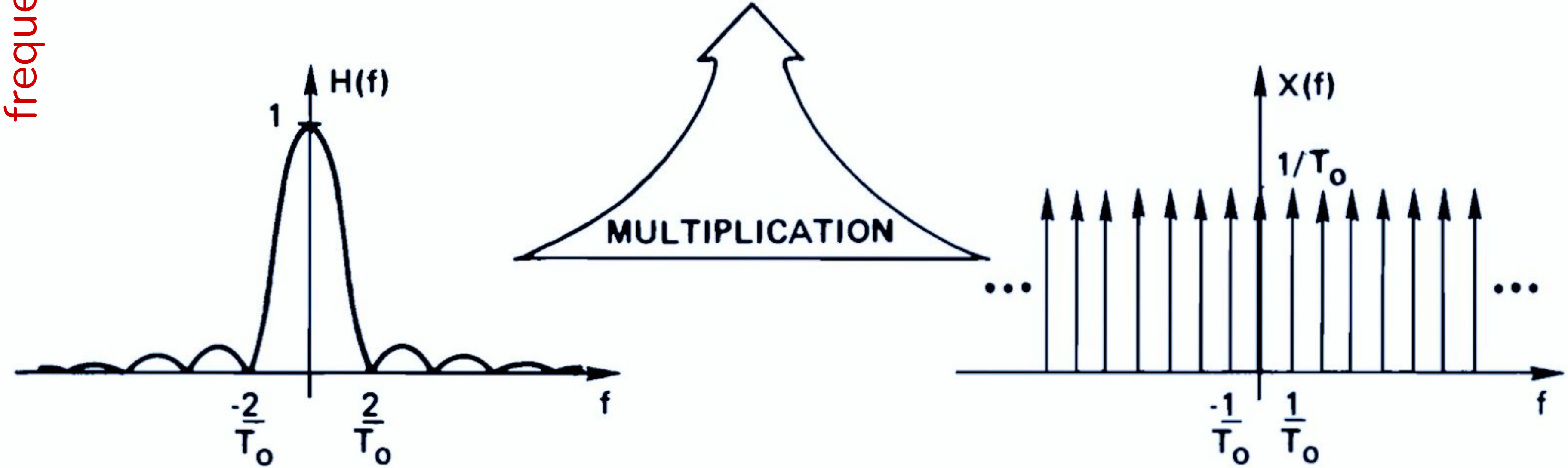
time-domain



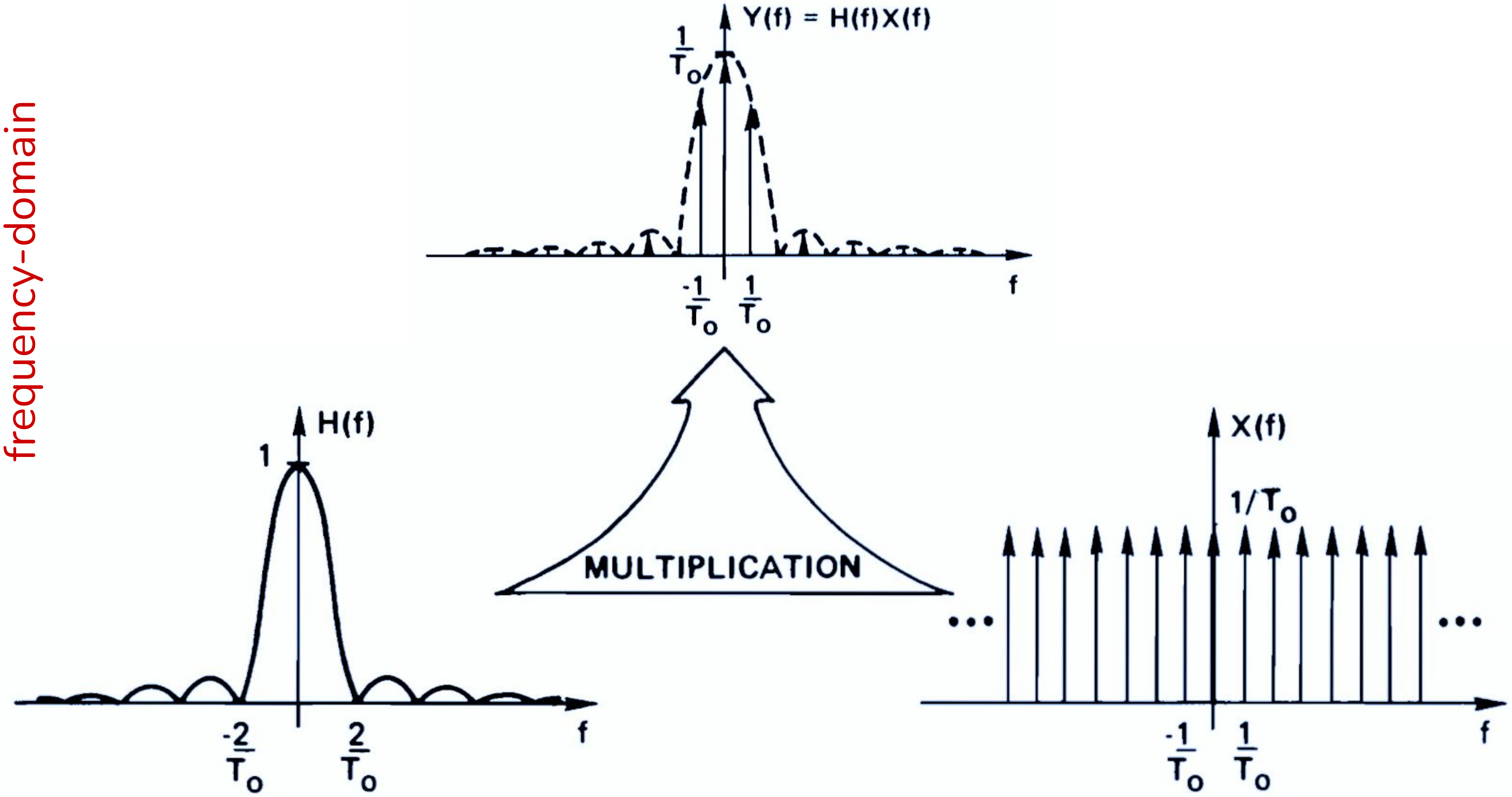
time-domain

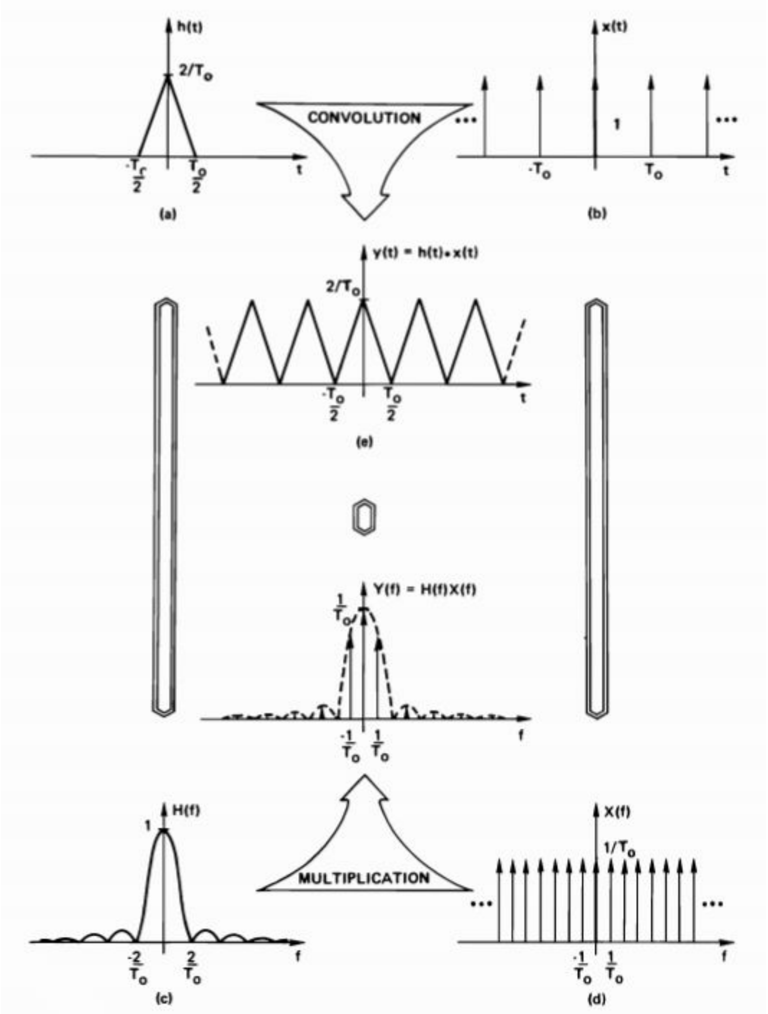


frequency-domain



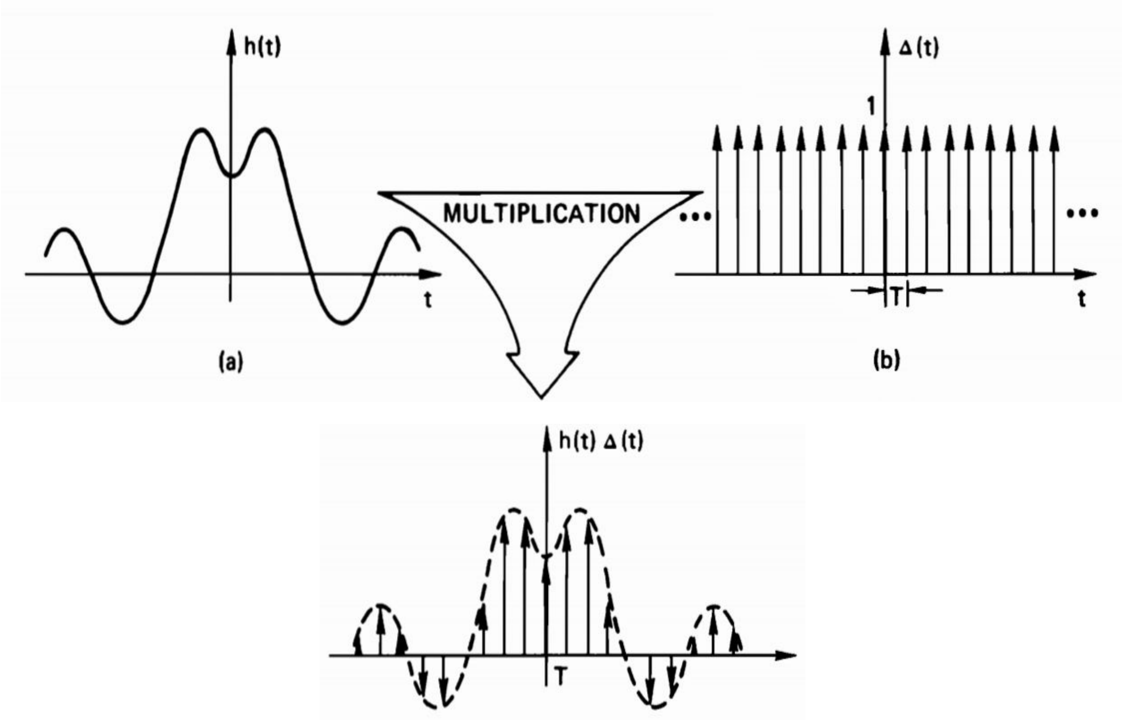
frequency-domain



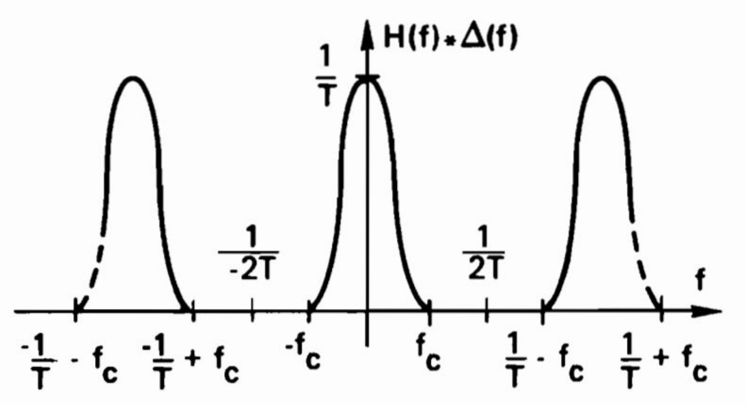


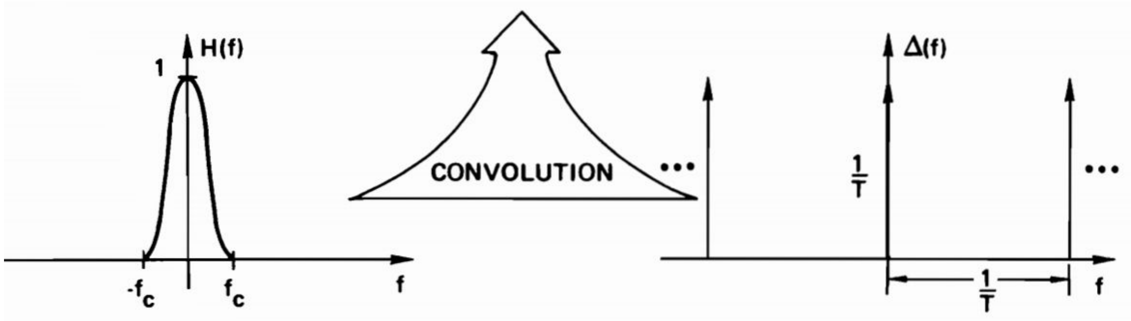
time-domain
frequency-domain
h(t)
2/T_o
CONVOLUTION
x(t)
1
(a)
(b)
y(t) = h(t)*x(t)
2/T_o
(e)
Y(f) = H(f)X(f)
H(f)
1
MULTIPLICATION
X(f)
1/T_o
(c)
(d)

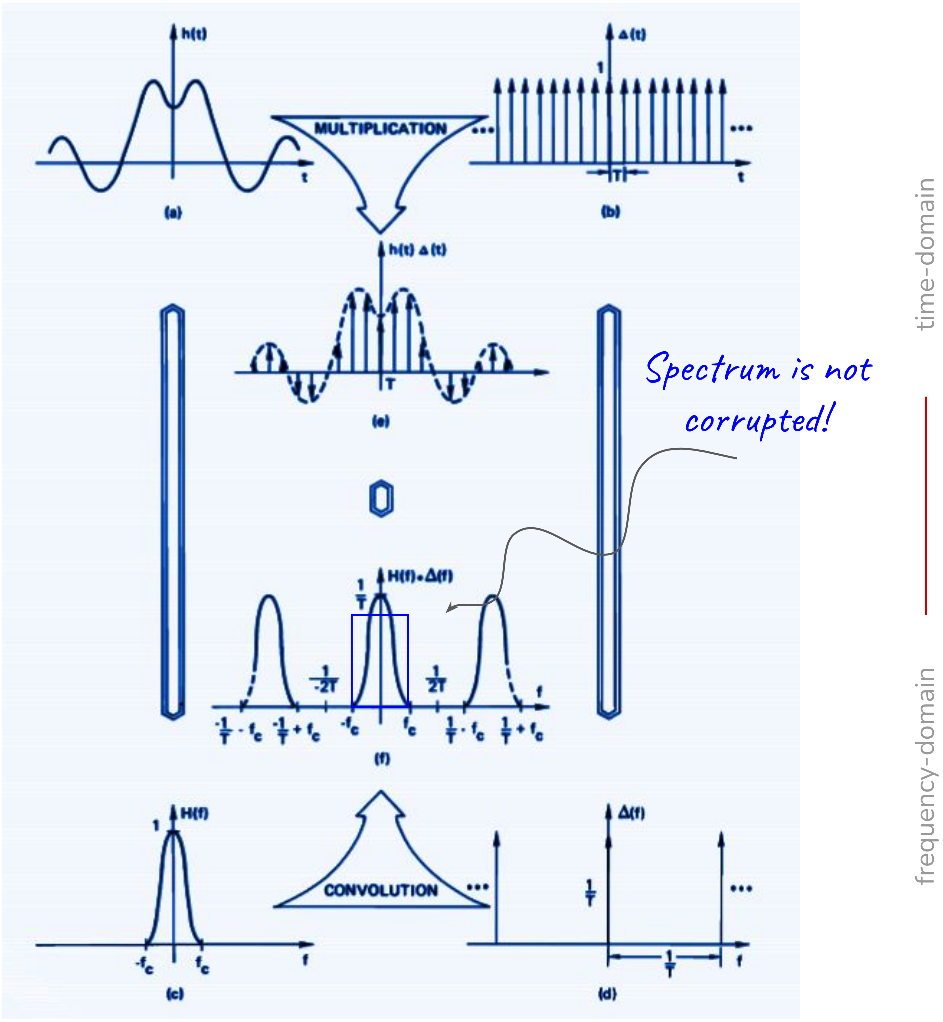
time-domain



frequency-domain







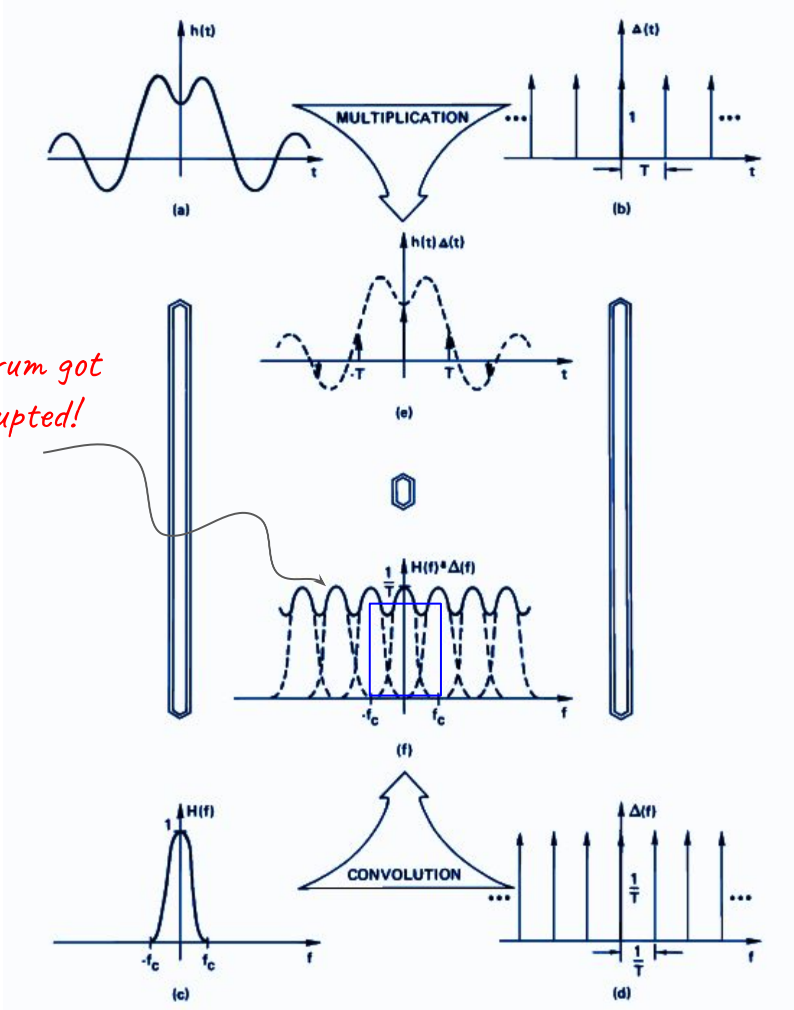
Perfect reconstruction

Aliasing (undersampling)

time-domain

frequency-domain

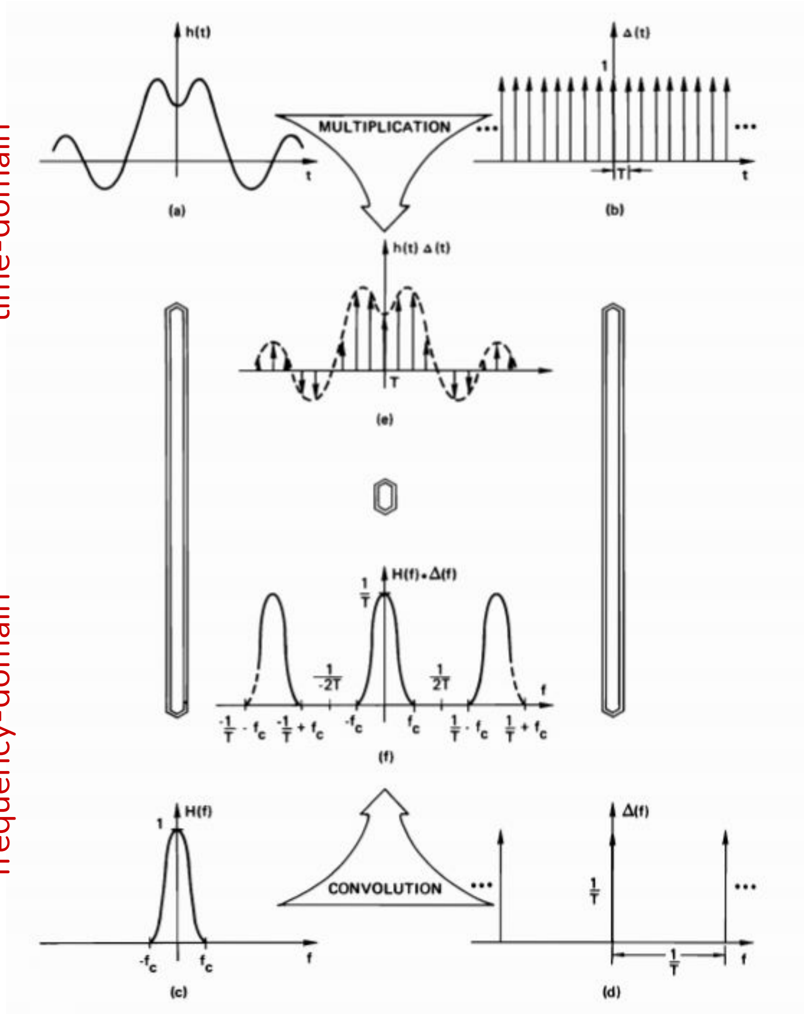
Spectrum got corrupted!



Oversampling

time-domain

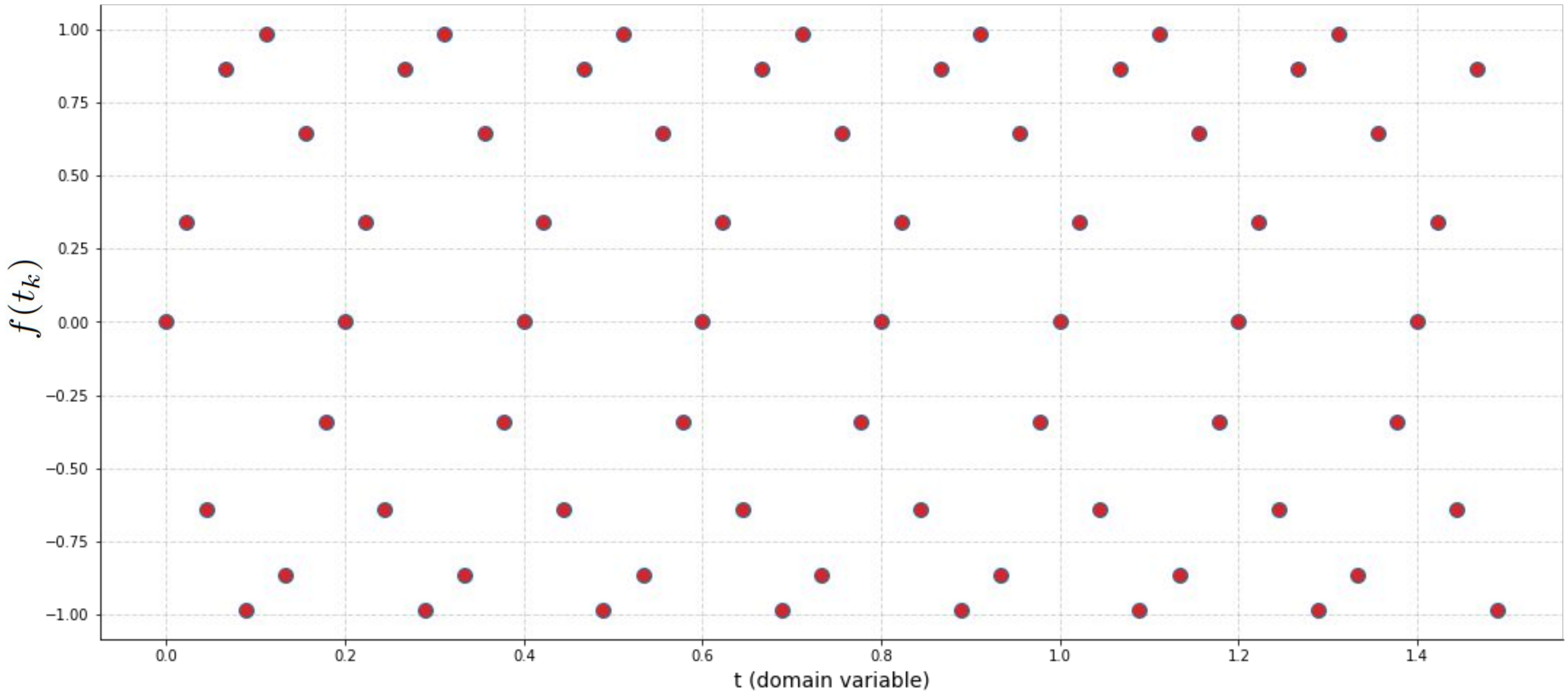
frequency-domain



$$\boxed{f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k)\,, \qquad t_k = \frac{k}{p}}$$

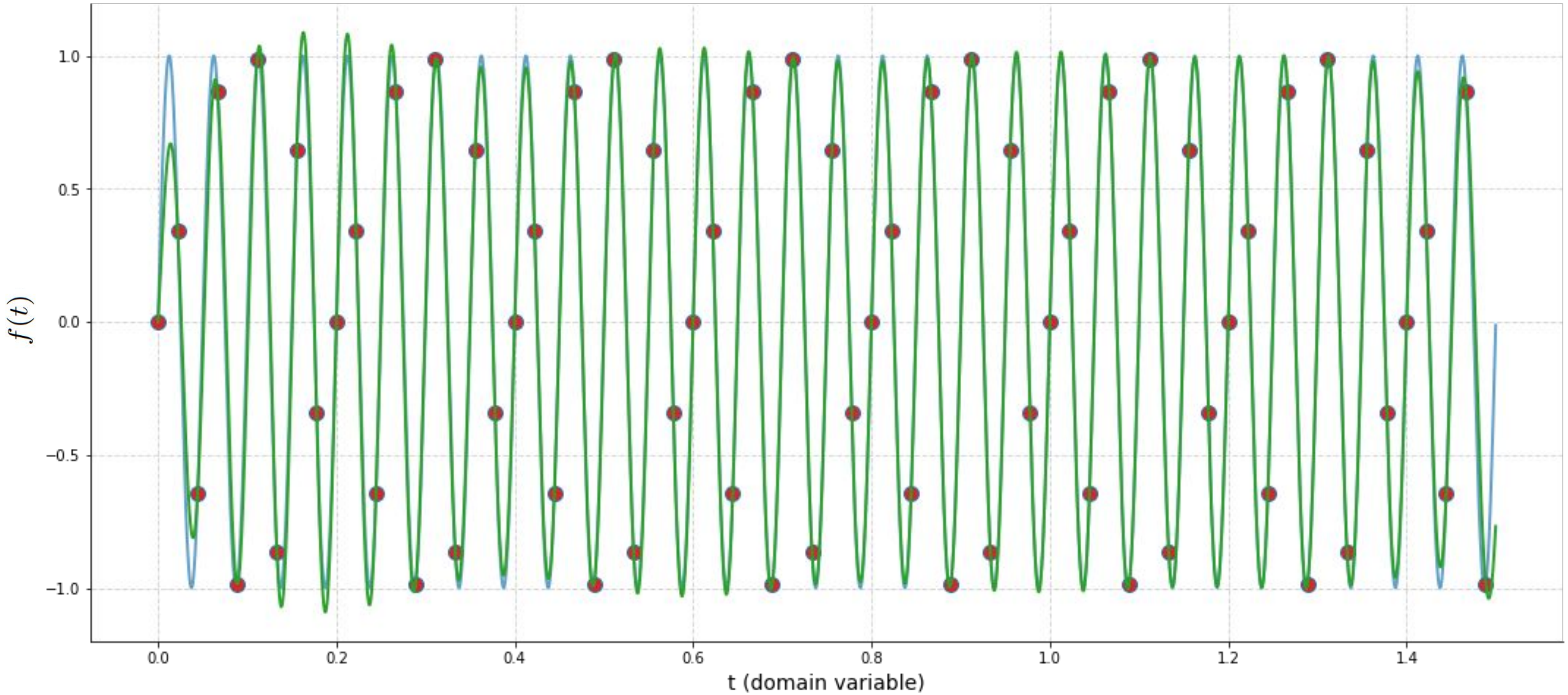
- Nyquist Sampling
- Shannon Sampling Theorem, 1940s
- Whittaker Sampling Formula

Example: we captured only its samples



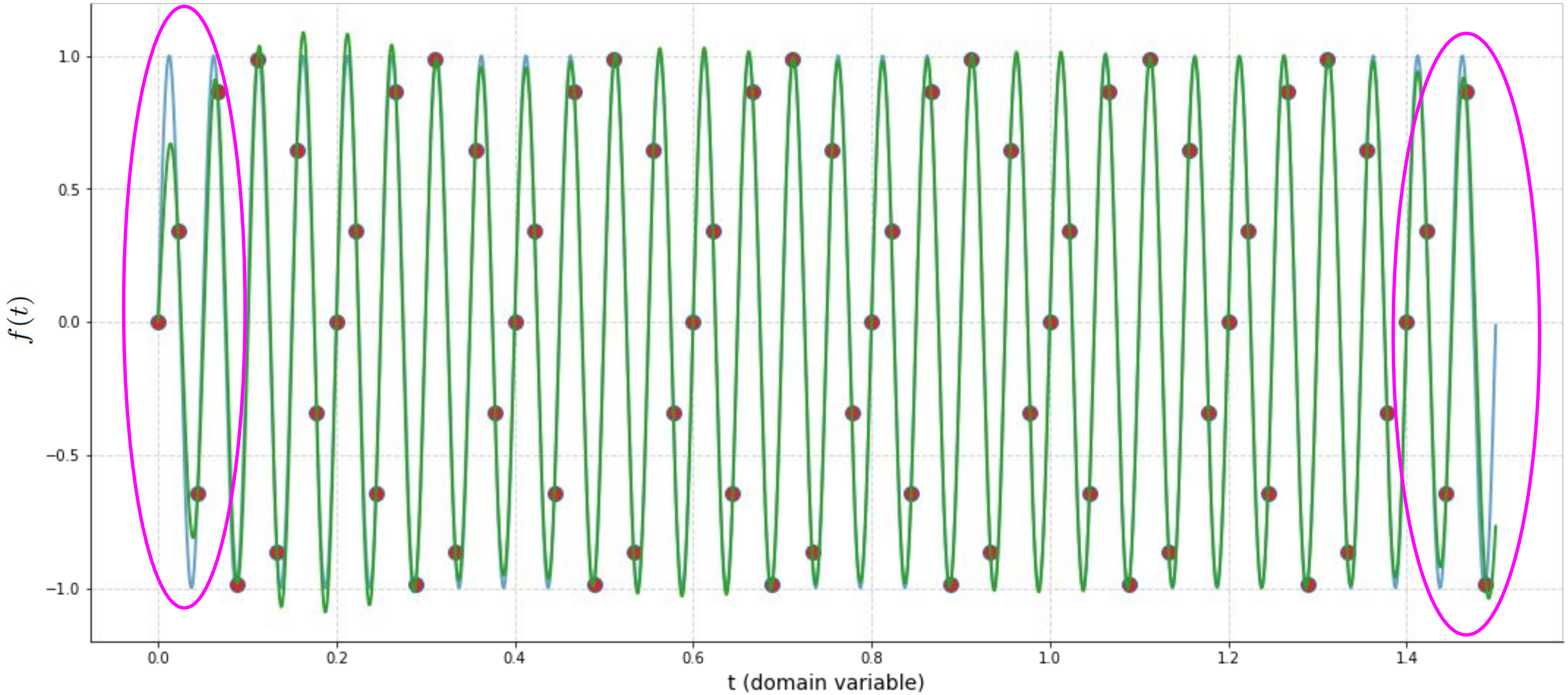
Example: reconstruct using sinc()

$$f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\,\mathrm{sinc}\,p(t - t_k) \quad , \quad t_k = \frac{k}{p}$$



Example: reconstruct using sinc()

$$f(t) = \sum_{k=-\infty}^{\infty} f(t_k)\, \mathrm{sinc}\, p(t - t_k) \quad , \quad t_k = \frac{k}{p}$$



Summary

- Sampling story
  - uniform sampling at lower rate leads to incorrect interpretation of captured data

- Mathematical concepts:
  - Dirac impulse
  - Train of Diracs
  - Spectrum of Diracs
  - Periodization and sampling using train of Diracs

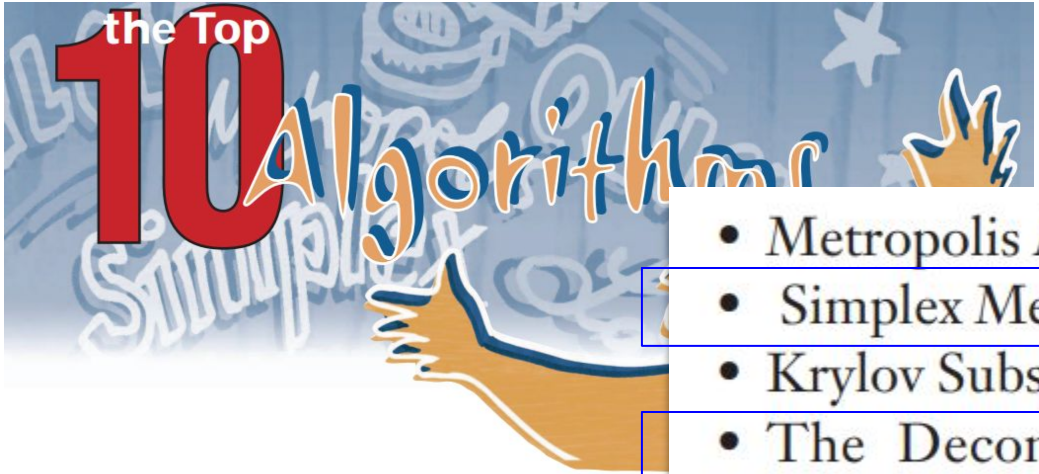
Summary

- Shannon sampling and reconstruction formula
  - Math and visualization

- Ways to avoid aliasing
  - Sample at higher and higher frequency (oversampling)
    - How do you decide? Physics? Experiments? Domain knowledge can help.
  - Use anti-aliasing filter
    - Don’t capture the full spectrum but what whatever is captured does not exist


the Top
10
Algorithms

- Metropolis Algorithm for Monte Carlo
- Simplex Method for Linear Programming
- Krylov Subspace Iteration Methods
- The Decompositional Approach to Matrix Computations
- The Fortran Optimizing Compiler
- QR Algorithm for Computing Eigenvalues
- Quicksort Algorithm for Sorting
- Fast Fourier Transform
- Integer Relation Detection
- Fast Multipole Method



- Metropolis Algorithm for Monte Carlo
- Simplex Method for Linear Programming
- Krylov Subspace Iteration Methods
- The Decompositional Approach to Matrix Computations
- The Fortran Optimizing Compiler
- QR Algorithm for Computing Eigenvalues
- Quicksort Algorithm for Sorting
- Fast Fourier Transform
- Integer Relation Detection
- Fast Multipole Method

Fourier Transform

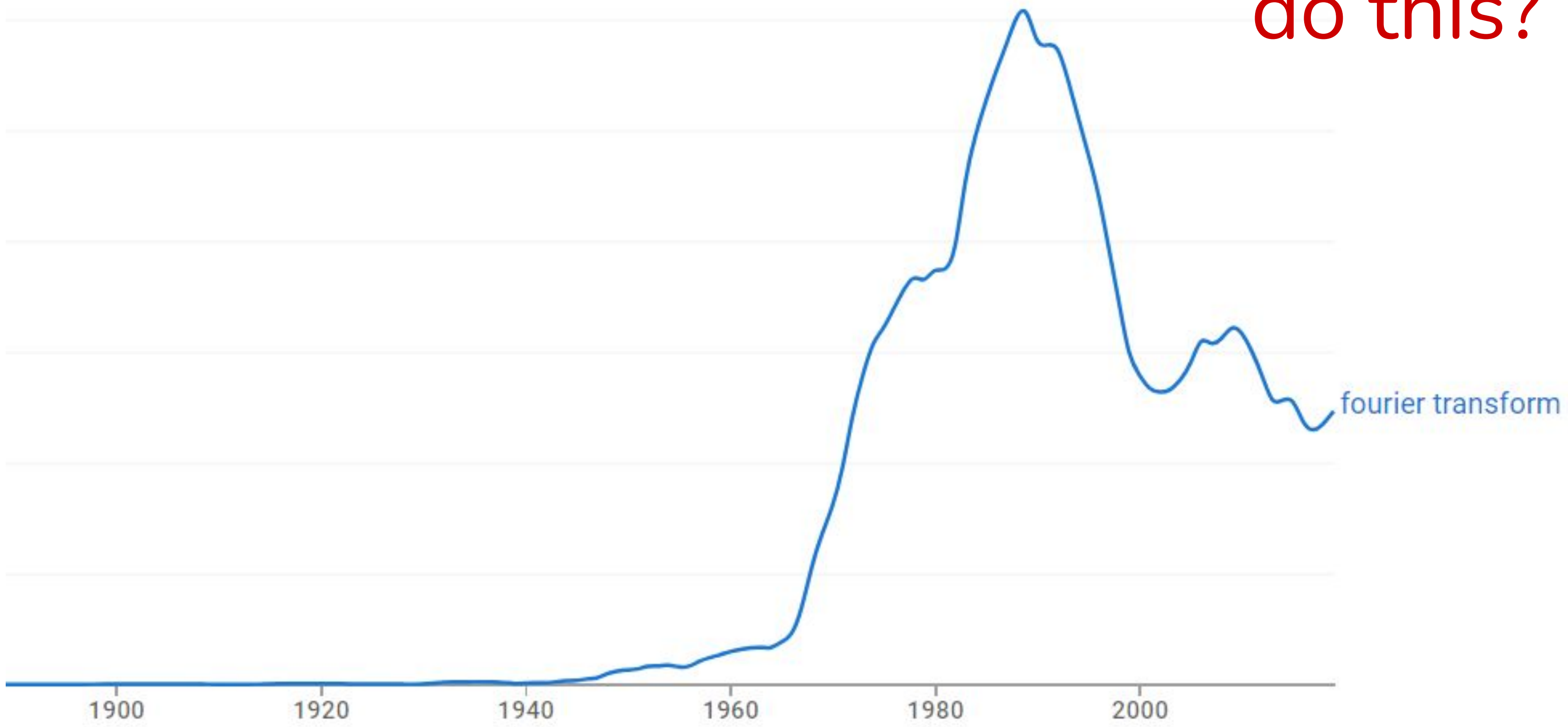
$$\hat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi ist} f(t)\, dt$$

Fourier Transform

$$\hat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t} f(t)\, dt$$

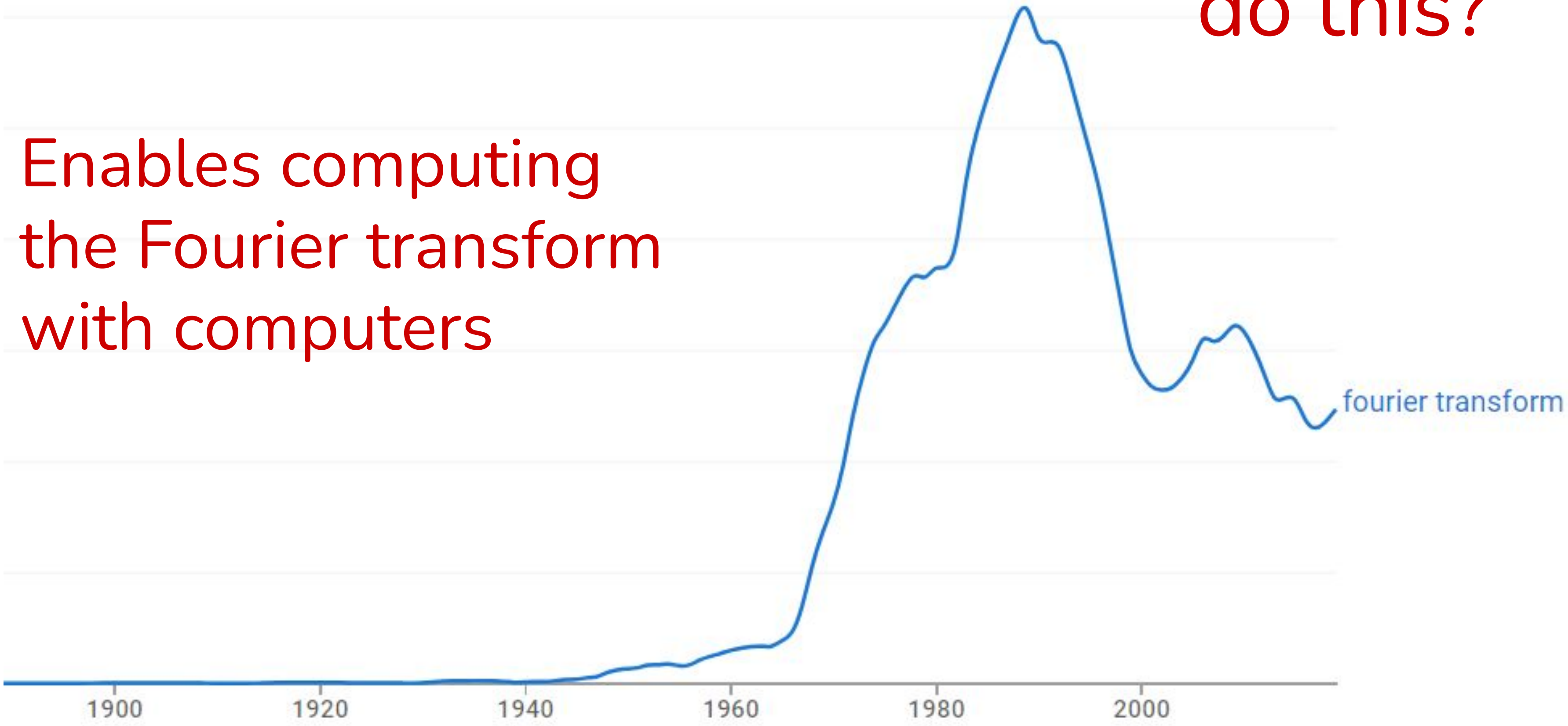
Can we discretize $f(t)$ and $\hat{f}(s)$?

Can we discretize $f(t)$ and $\hat{f}(s)$? Why to do this?



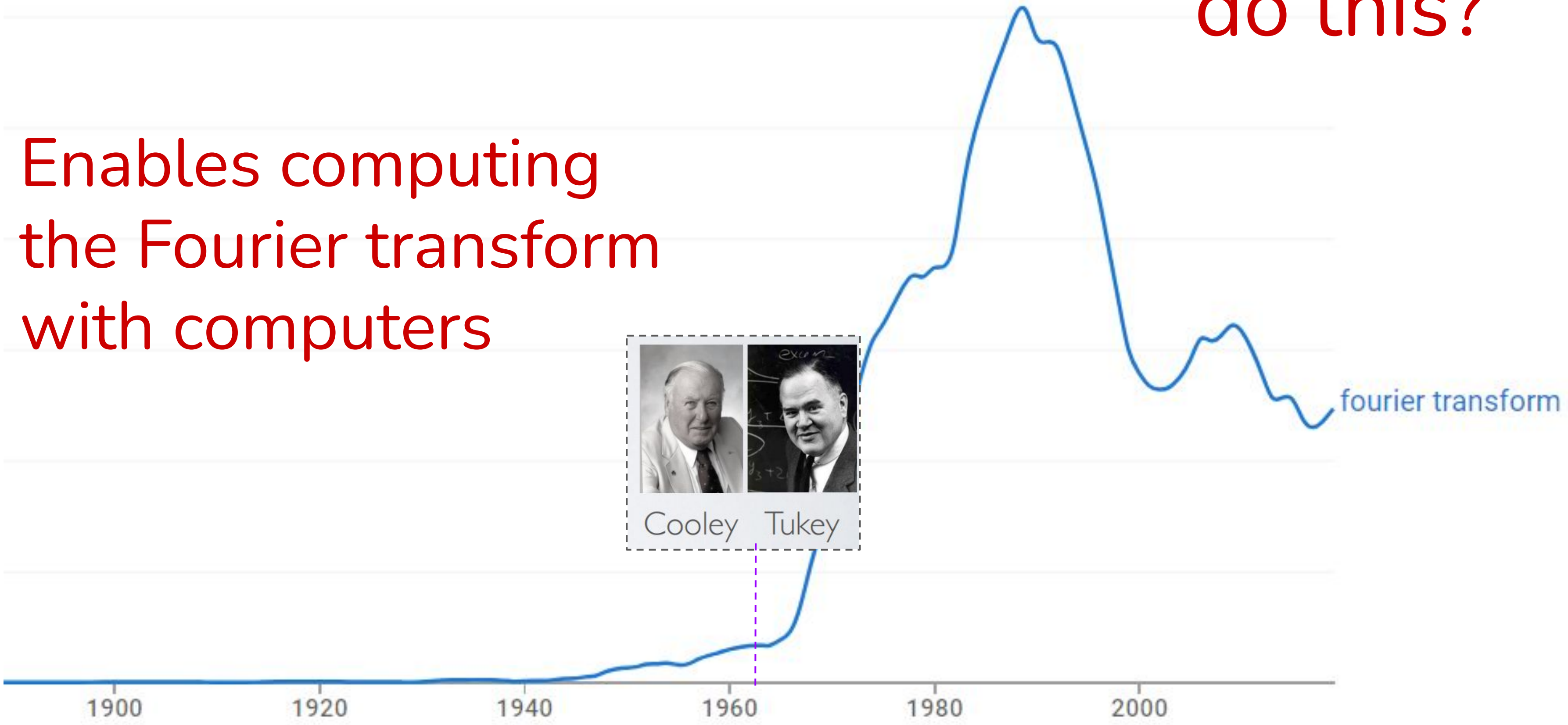
Can we discretize $f(t)$ and $\hat{f}(s)$? Why to do this?

Enables computing the Fourier transform with computers

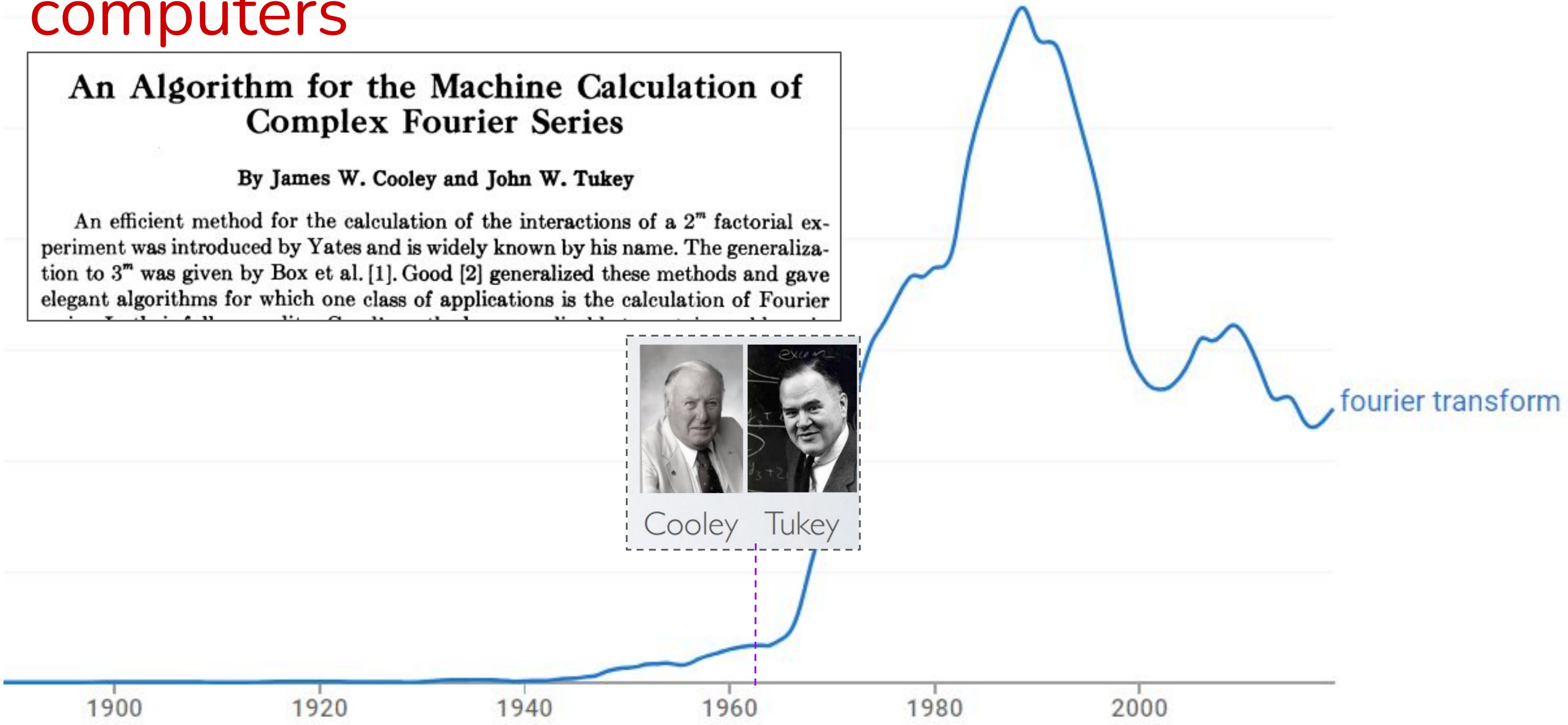


Can we discretize $f(t)$ and $\hat{f}(s)$? Why to do this?

Enables computing the Fourier transform with computers



Enables computing the Fourier transform with computers



An Algorithm for the Machine Calculation of Complex Fourier Series

By James W. Cooley and John W. Tukey

An efficient method for the calculation of the interactions of a 2^m factorial experiment was introduced by Yates and is widely known by his name. The generalization to 3^m was given by Box et al. [1]. Good [2] generalized these methods and gave elegant algorithms for which one class of applications is the calculation of Fourier

Discrete Fourier Transform

How do we proceed?

(1)
This will be discretized?

$$\hat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t} f(t)\, dt$$

(4)
Can this be discretized?

(3)
Integral to summation?

(2)
Use Finite samples?

Discrete Fourier Transform

How do we proceed?

(1)

This will be discretized?

$$\hat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi i s t} f(t)\, dt$$

(4)

Can this be discretized?

(3)

Integral to summation?

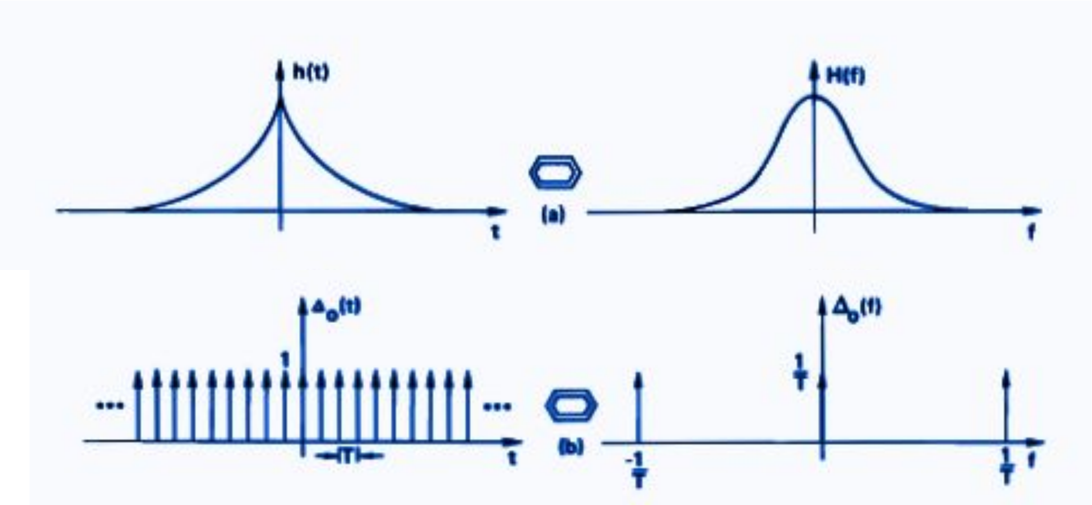
(2)

Use Finite samples?

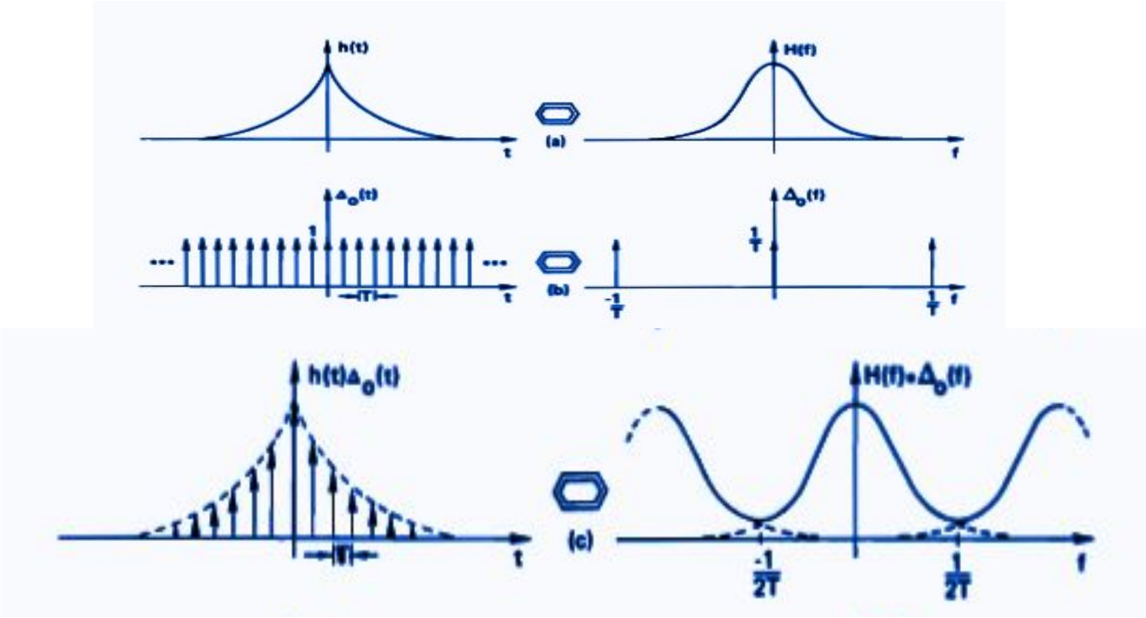
Let's proceed through visualization

- Continuous-time signal

- sampling

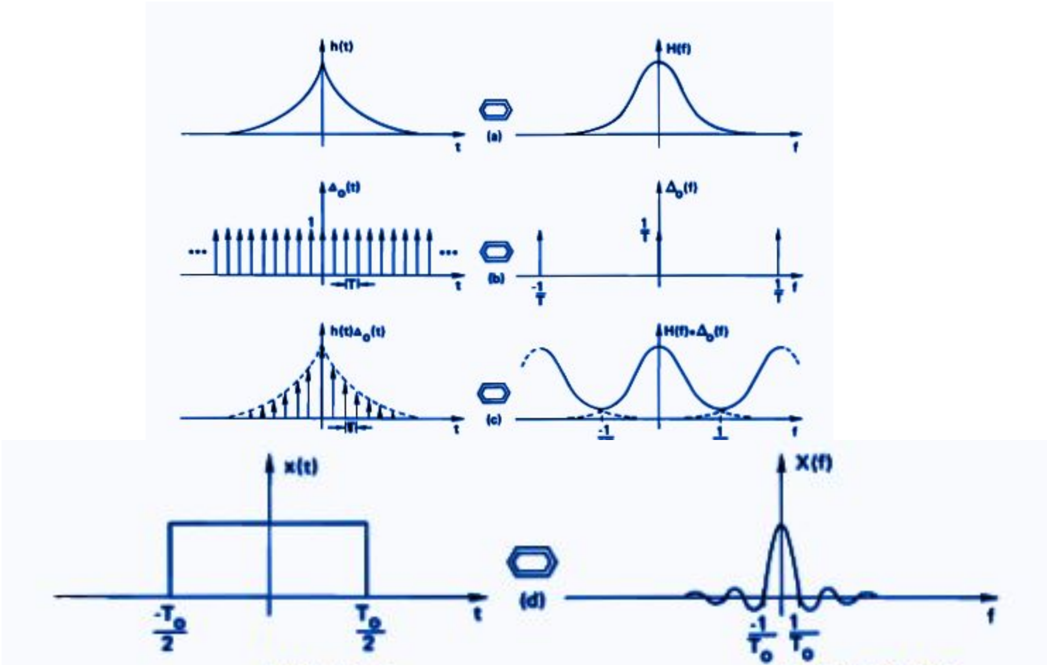


- sampling

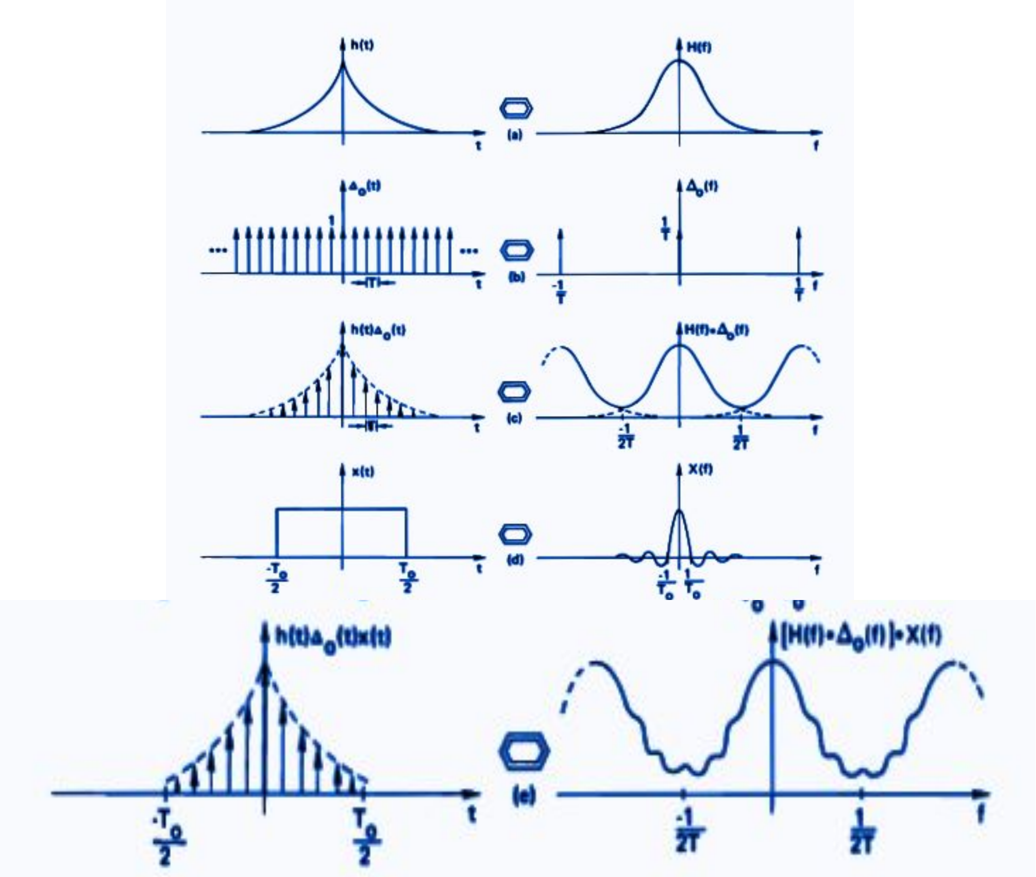


* sampling

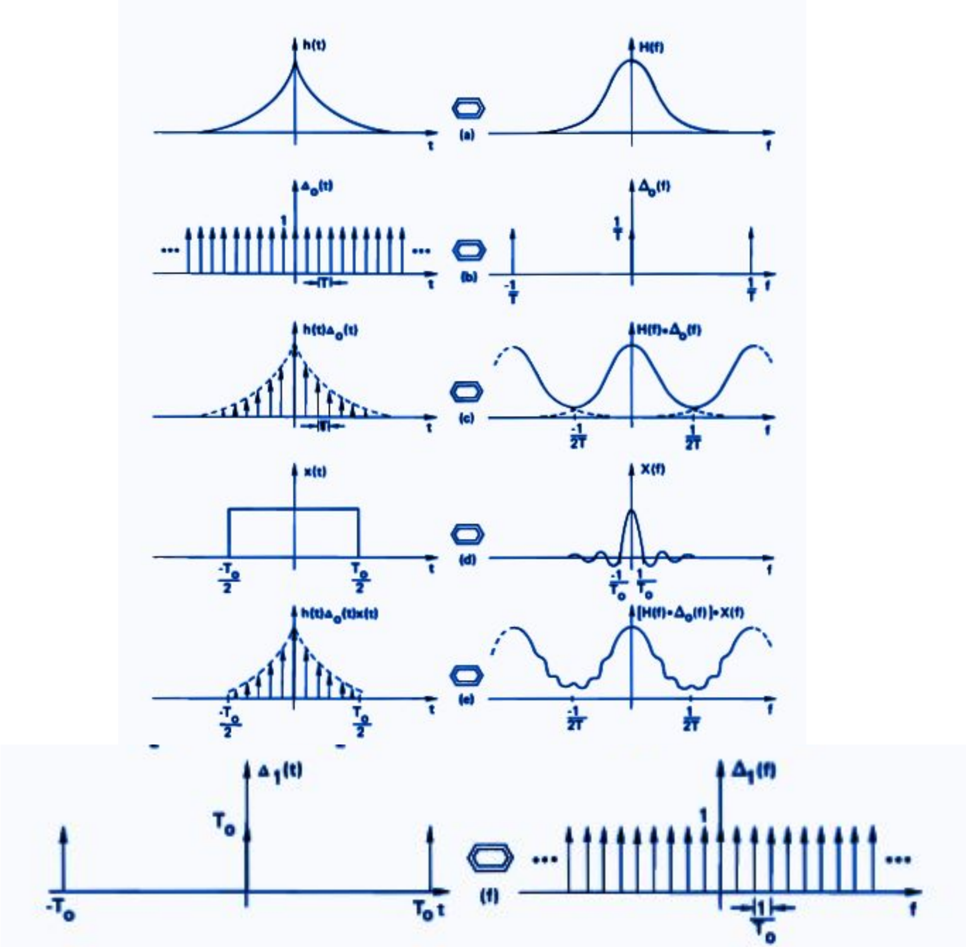
* truncation



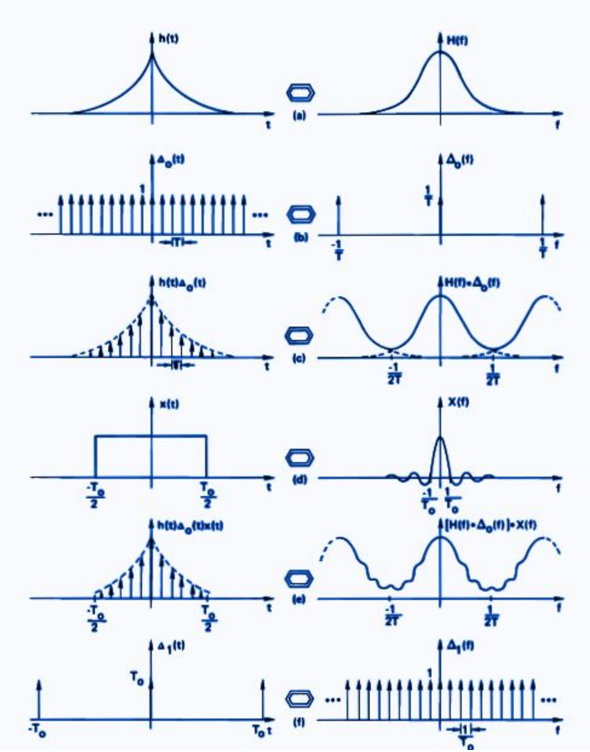
- sampling
- truncation



- sampling
- truncation
- periodization

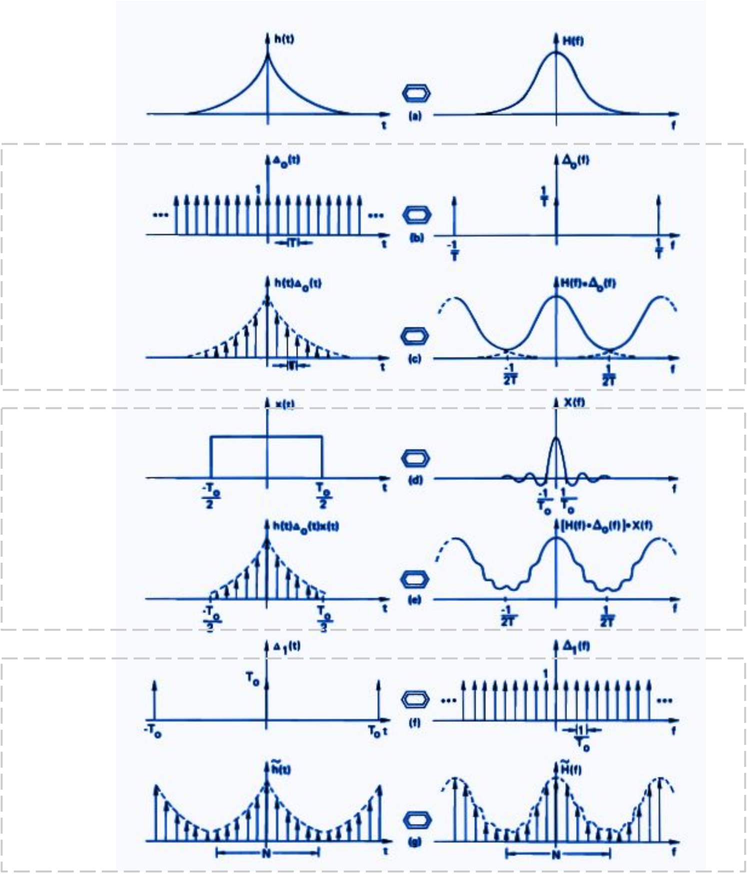


- sampling
- truncation
- periodization



Steps

- sampling
- truncation
- periodization



Result

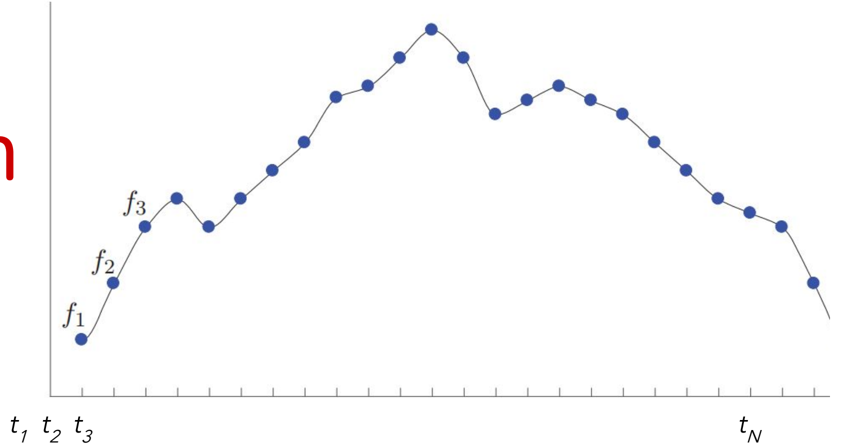
- Discrete Fourier Transform (DFT)

Discrete Fourier Transform (DFT)

$$F(s_m) = \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i s_m t_n}$$

$$= \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i n m / 2BL}$$

$$= \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i n m / N}$$



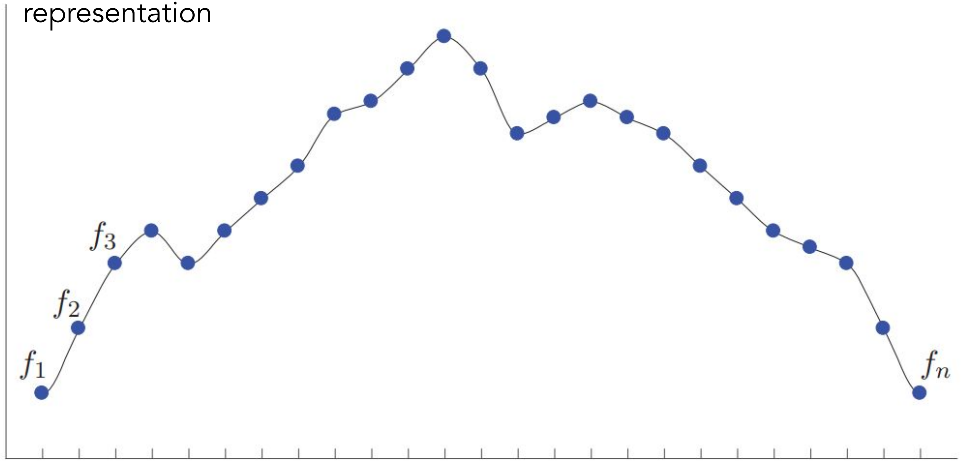
Assumptions:

- *f(t)* is (effectively) finite length in time and frequency
- Duration (length): *L*
- Bandwidth = *2B* (*-B* to *+B*)

$t_n = nT_s$, $T_s = L/N = 1/2B$

Signal

Function representation



$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ \vdots \\ f_N \end{bmatrix}$$

Data

Vector representation

DFT

$$F(s_m) = \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i s_m t_n}$$

$$= \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i n m / N}$$

Notation,

$$\omega = e^{2\pi i / N}$$

DFT

$$F(s_m) = \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i s_m t_n}$$

$$= \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i n m / N}$$

Notation,

$$\omega = e^{2\pi i/N}$$

$$\boldsymbol{\omega}^k = (1, \omega^k, \omega^{2k}, \ldots, \omega^{(N-1)k})$$

Additional notation,

$$\boldsymbol{\omega}^{-k} = (1, \omega^{-k}, \omega^{-2k}, \ldots, \omega^{-(N-1)k})$$

DFT

$$F(s_m) = \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i s_m t_n}$$

$$= \sum_{n=0}^{N-1} f(t_n) e^{-2\pi i n m/N}$$

$$\boxed{\mathbf{F}[m] = \sum_{k=0}^{N-1} \mathbf{f}[k] \omega^{-km} = \sum_{k=0}^{N-1} \mathbf{f}[k] e^{-2\pi i k m/N}}$$

Notation,

$$\omega = e^{2\pi i/N}$$

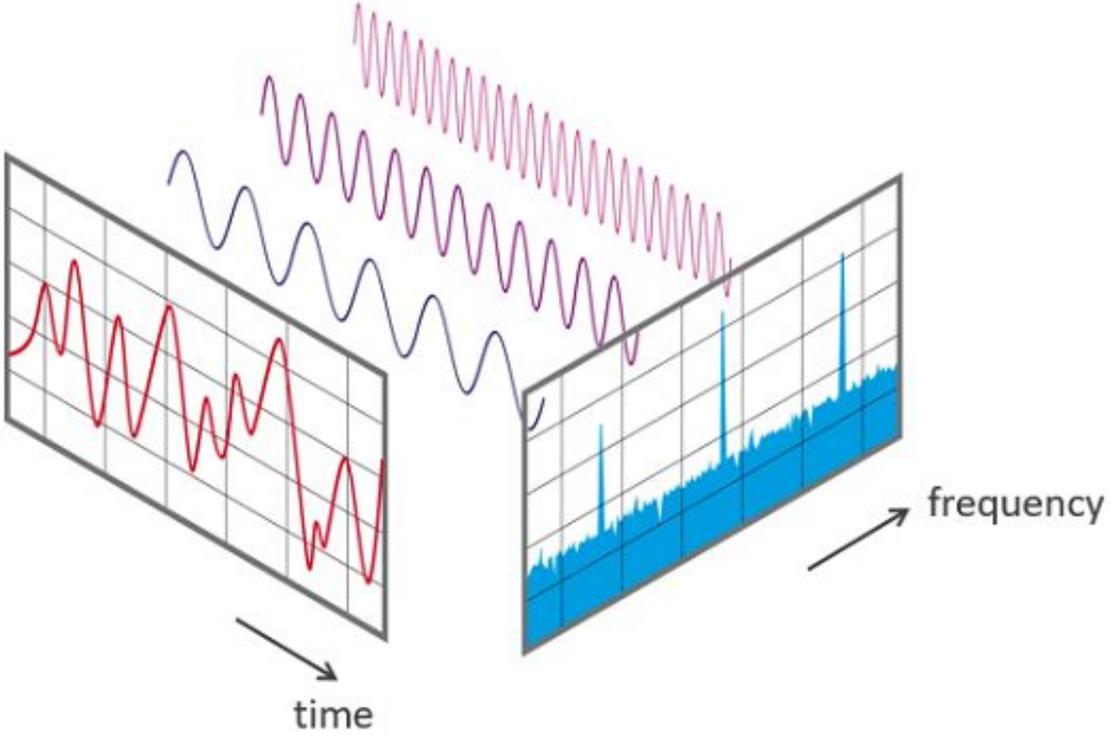
$$\boldsymbol{\omega}^k = (1, \omega^k, \omega^{2k}, \ldots, \omega^{(N-1)k})$$

Additional notation,

$$\boldsymbol{\omega}^{-k} = (1, \omega^{-k}, \omega^{-2k}, \ldots, \omega^{-(N-1)k})$$

DFT

$$\boxed{\mathbf{F}[m] = \sum_{k=0}^{N-1} \mathbf{f}[k]\omega^{-km} = \sum_{k=0}^{N-1} \mathbf{f}[k]e^{-2\pi ikm/N}}$$


frequency
time

DFT

$$\mathbf{F}[m] = \sum_{k=0}^{N-1} \mathbf{f}[k]\omega^{-km} = \sum_{k=0}^{N-1} \mathbf{f}[k]e^{-2\pi ikm/N}$$

$$\begin{pmatrix} F[0] \\ F[1] \\ F[2] \\ \vdots \\ F[N-1] \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^{-1\cdot 1} & \omega^{-1\cdot 2} & \cdots & \omega^{-(N-1)} \\ 1 & \omega^{-2\cdot 1} & \omega^{-2\cdot 2} & \cdots & \omega^{-2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(N-1)\cdot 1} & \omega^{-(N-1)\cdot 2} & \cdots & \omega^{-(N-1)^2} \end{pmatrix} \begin{pmatrix} f[0] \\ f[1] \\ f[2] \\ \vdots \\ f[N-1] \end{pmatrix}.$$

Discrete Fourier Transform
vector

DFT
Square Matrix

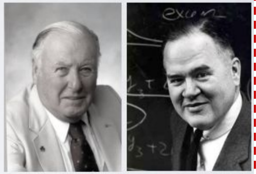
Data
vector

DFT Computation

N	DFT (real multiplies)	DFT (real additions)
	$4N^2$	$2N(2N-1)$
2	16	12
4	64	56
8	256	240
16	1,024	992
32	4,096	4,032
64	16,384	16,256
128	65,536	65,280
256	262,144	261,632
512	1,048,576	1,047,552
1,024	4,194,304	4,192,256

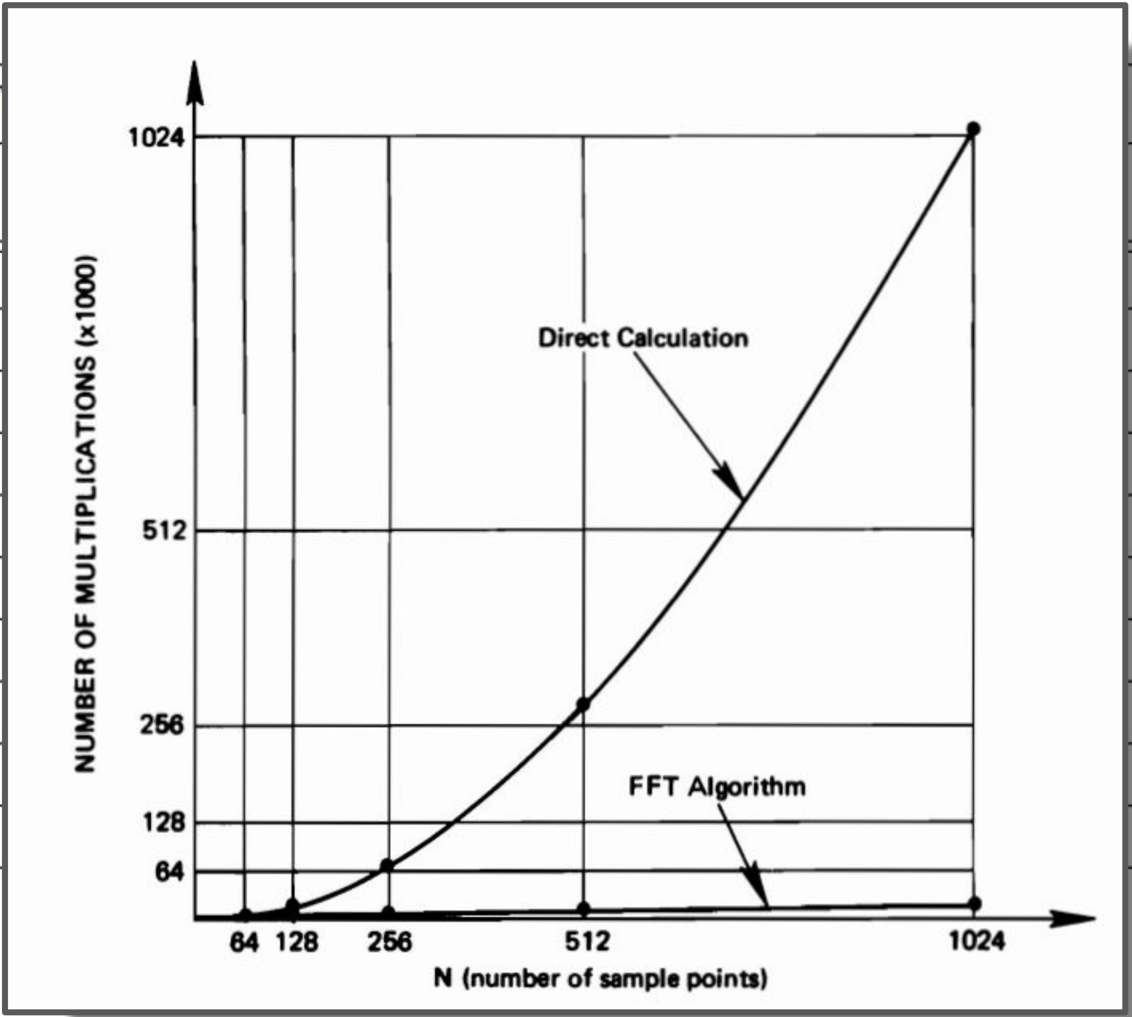
- A lot of multiplications and additions as the signal length increase.
- Is there a way to compute it efficiently?

DFT Computation using FFT

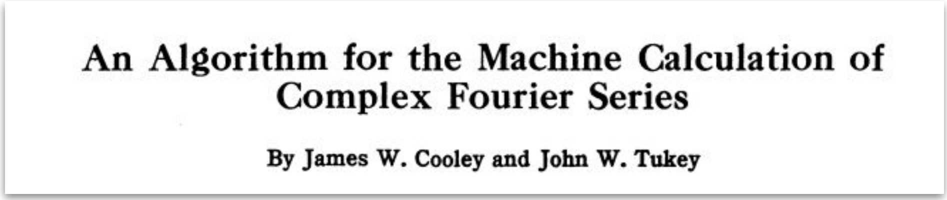


N	DFT (real multiplies) $4N^2$	DFT (real additions) $2N(2N-1)$	FFT (real multiplies) $2N\log_2 N$	FFT (real additions) $3N\log_2 N$
2	16	12	4	6
4	64	56	16	24
8	256	240	48	72
16	1,024	992	128	192
32	4,096	4,032	320	480
64	16,384	16,256	768	1,152
128	65,536	65,280	1,792	2,688
256	262,144	261,632	4,096	6,144
512	1,048,576	1,047,552	9,216	13,824
1,024	4,194,304	4,192,256	20,480	30,720

N	DFT	FFT (real additions)
		$3N \log_2 N$
2		6
4		24
8		72
16		192
32		480
64		1,152
128		2,688
256		6,144
512		13,824
1,024		30,720



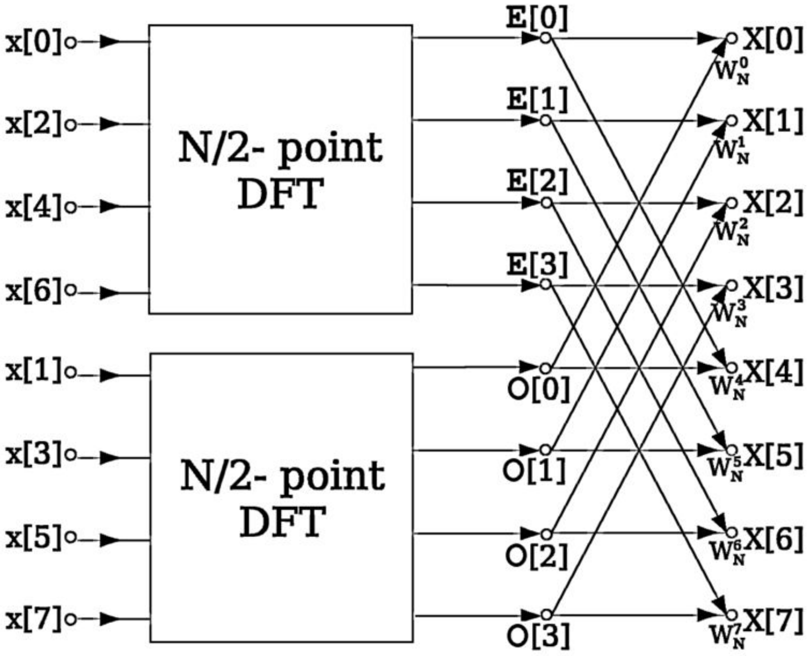
FFT: Fast Fourier Transform

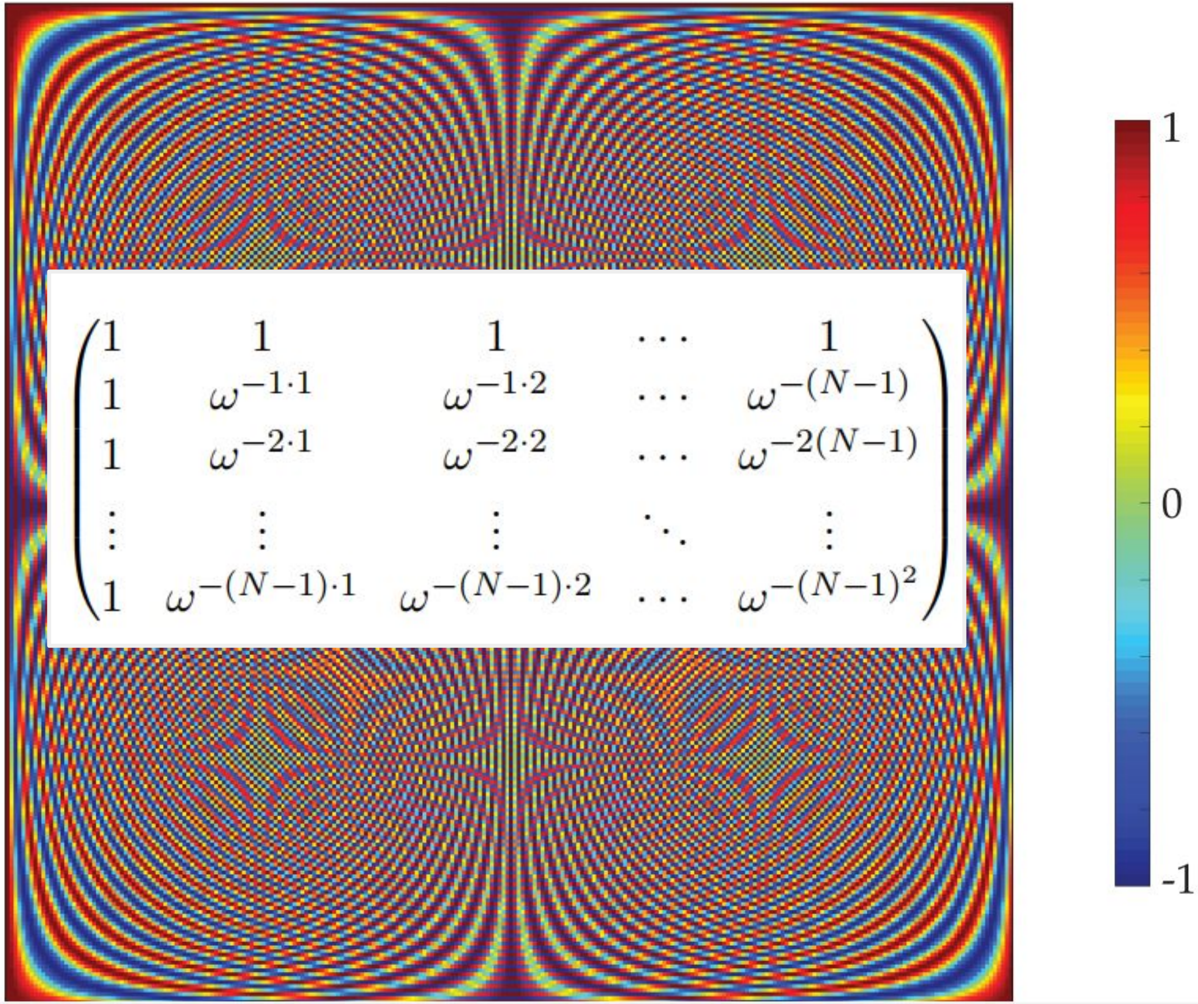

An Algorithm for the Machine Calculation of Complex Fourier Series

By James W. Cooley and John W. Tukey

An algorithm for faster computation of DFT.

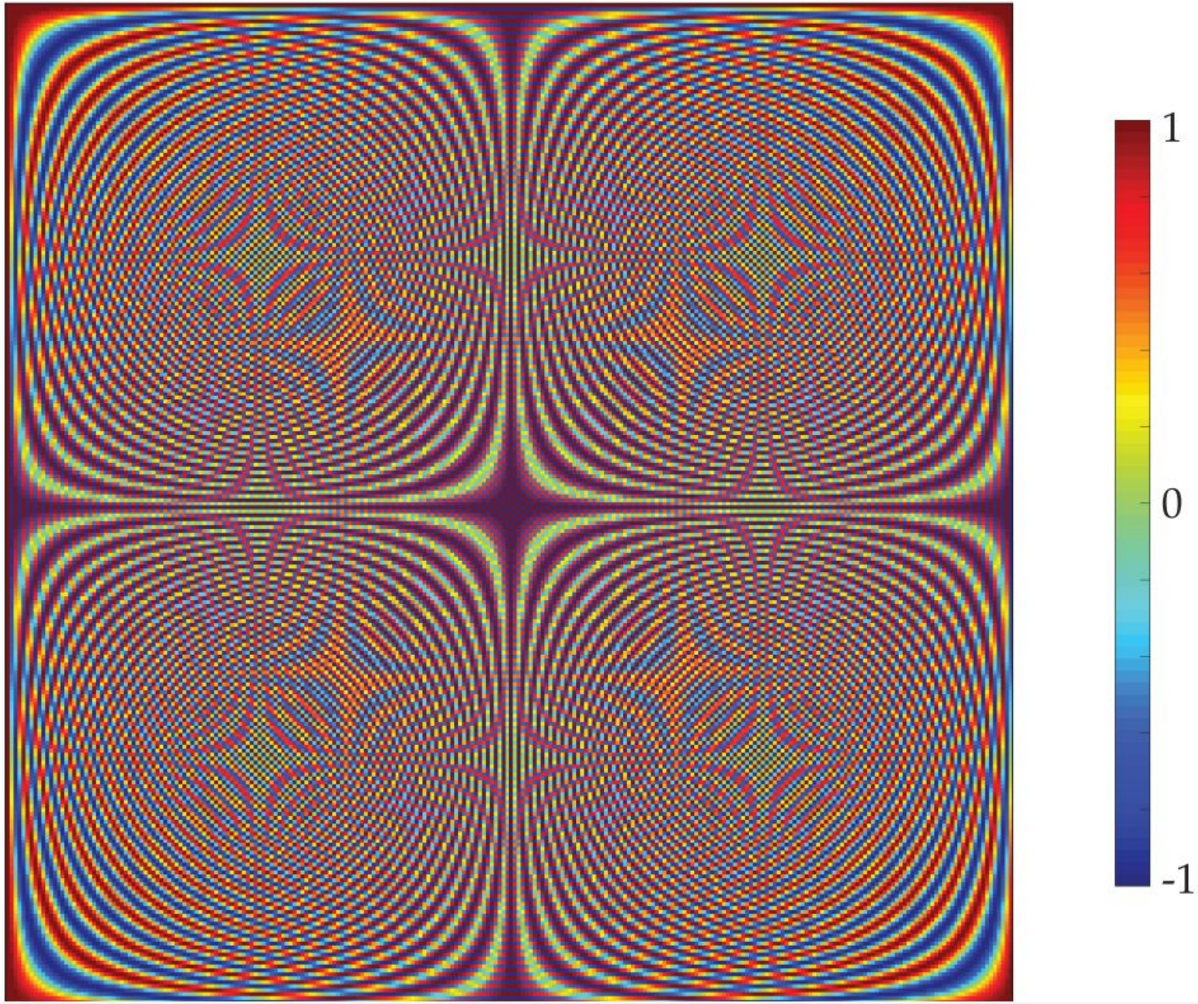
Proceeds by making group of even and odd indices in the input





$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^{-1\cdot 1} & \omega^{-1\cdot 2} & \cdots & \omega^{-(N-1)} \\ 1 & \omega^{-2\cdot 1} & \omega^{-2\cdot 2} & \cdots & \omega^{-2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(N-1)\cdot 1} & \omega^{-(N-1)\cdot 2} & \cdots & \omega^{-(N-1)^2} \end{pmatrix}$$

- Visualizing the Real [DFT]



- Cooley-Tukey algorithm calculates the DFT directly with fewer summations
- The trick to the Cooley-Tukey algorithm is recursion.
- Split the matrix we wish to perform the FFT on into two parts: one for all elements with even indices and another for all odd indices.
- We then proceed to split the array again and again until we have a manageable array size to perform a DFT (or similar FFT) on.
- We can also perform a similar re-ordering by using a bit reversal scheme, where we output each array index's integer value in binary and flip it to find the new location of that element.
- Complexity to ~O(N logN)

https://vanhunteradams.com/FFT/FFT.html

Summary

$$\begin{pmatrix} F[0] \\ F[1] \\ F[2] \\ \vdots \\ F[N-1] \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^{-1\cdot 1} & \omega^{-1\cdot 2} & \cdots & \omega^{-(N-1)} \\ 1 & \omega^{-2\cdot 1} & \omega^{-2\cdot 2} & \cdots & \omega^{-2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(N-1)\cdot 1} & \omega^{-(N-1)\cdot 2} & \cdots & \omega^{-(N-1)^2} \end{pmatrix} \begin{pmatrix} f[0] \\ f[1] \\ f[2] \\ \vdots \\ f[N-1] \end{pmatrix}.$$

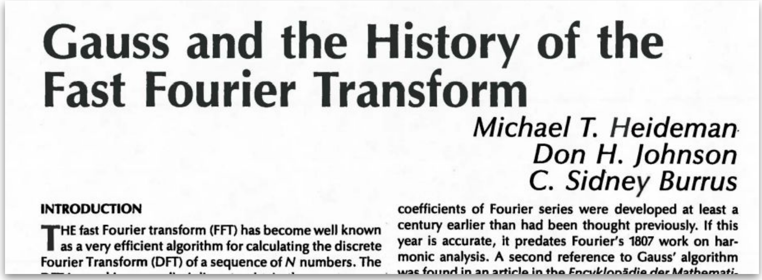
Fourier Transform
vector

DFT
Square Matrix

Data
vector

- Matrix multiplication is $O(N^2)$ computations
- This is a lot of computation for N>>1, as usual case
- Linear scaling is desired in most applications
- Fast Fourier Transform (FFT) algorithm enables computing DFT in O(N logN)

Resources



Gauss and the History of the Fast Fourier Transform

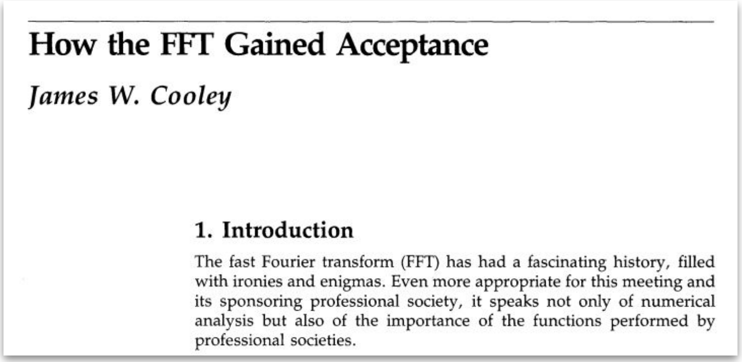
Michael T. Heideman
Don H. Johnson
C. Sidney Burrus

INTRODUCTION

THE fast Fourier transform (FFT) has become well known as a very efficient algorithm for calculating the discrete Fourier Transform (DFT) of a sequence of *N* numbers. The

coefficients of Fourier series were developed at least a century earlier than had been thought previously. If this year is accurate, it predates Fourier's 1807 work on harmonic analysis. A second reference to Gauss' algorithm was found in an article in the *Encyklopädie der Mathemati-*

Paper



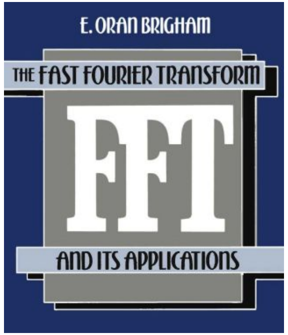
How the FFT Gained Acceptance

James W. Cooley

1. Introduction

The fast Fourier transform (FFT) has had a fascinating history, filled with ironies and enigmas. Even more appropriate for this meeting and its sponsoring professional society, it speaks not only of numerical analysis but also of the importance of the functions performed by professional societies.

Paper



Book