Fourier transform

Advanced Mathematics in Geodesy

09.10.2024.

Overview

- Fourier series
- CFT (continuous transform)
- DFT (discrete transform)
- FFT (fast transform)
- Example: FFT of wheel mounted accelerometry
- Applications

1D Fourier series

- Expansion of *f*(*t*) in trigonometric basis (*t* denotes either time or distance)

$$f(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty}\left(a_n \cos n\omega_0 t + b_n \sin n\omega_0 t\right)$$

- Fundamental circular frequency (rad/s, rad/km) $\omega_0 = \frac{2\pi}{T}$

Fourier series in complex form

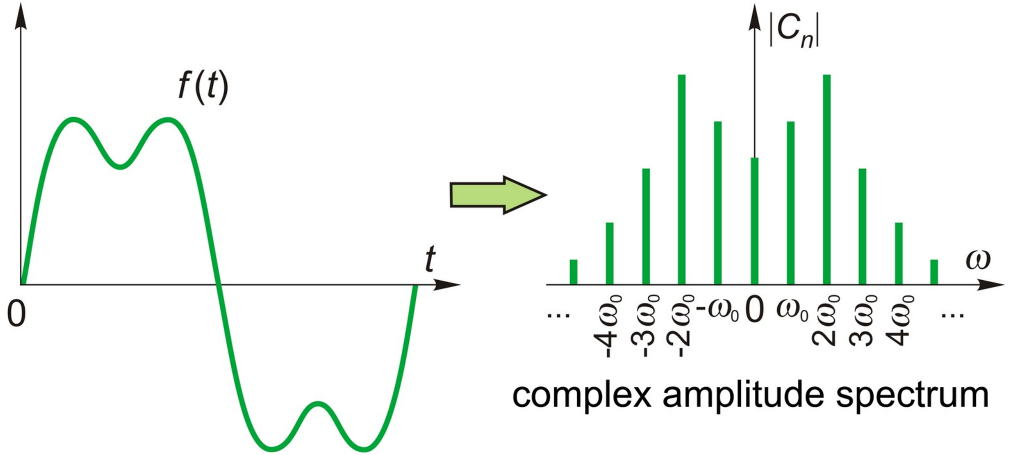
- Euler's identity: $e^{in\omega_0 t} = \cos n\omega_0 t + i \sin n\omega_0 t$
- in complex form:

$$f(t) = \sum_{-\infty}^{\infty} c_n e^{in\omega_0 t}$$

- complex coefficients:

$$c_{\pm n} = \frac{1}{2}\left(a_n \mp i b_n\right)$$

Periodic function mapping to coefficients



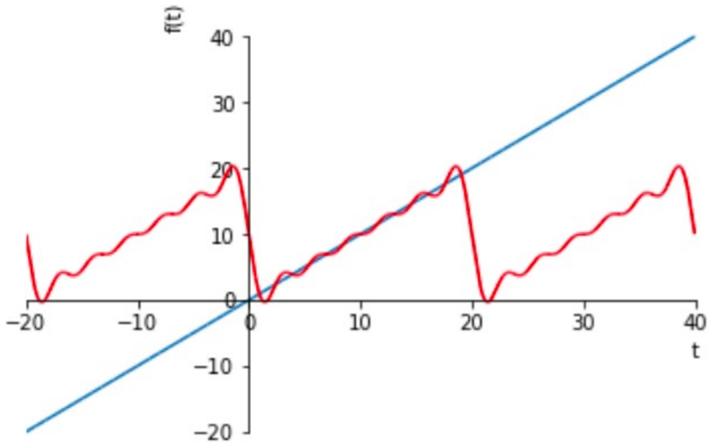
- calculation of coefficients (period T):

$$c_n = \frac{1}{T} \int_{d}^{d+T} f(t)\, e^{-in\omega_0 t}\, dt$$

1D Fourier series (Jupyter notebook)

https://nbviewer.jupyter.org/github/gyulat/Fourier/blob/master/1DFourier_en.ipynb

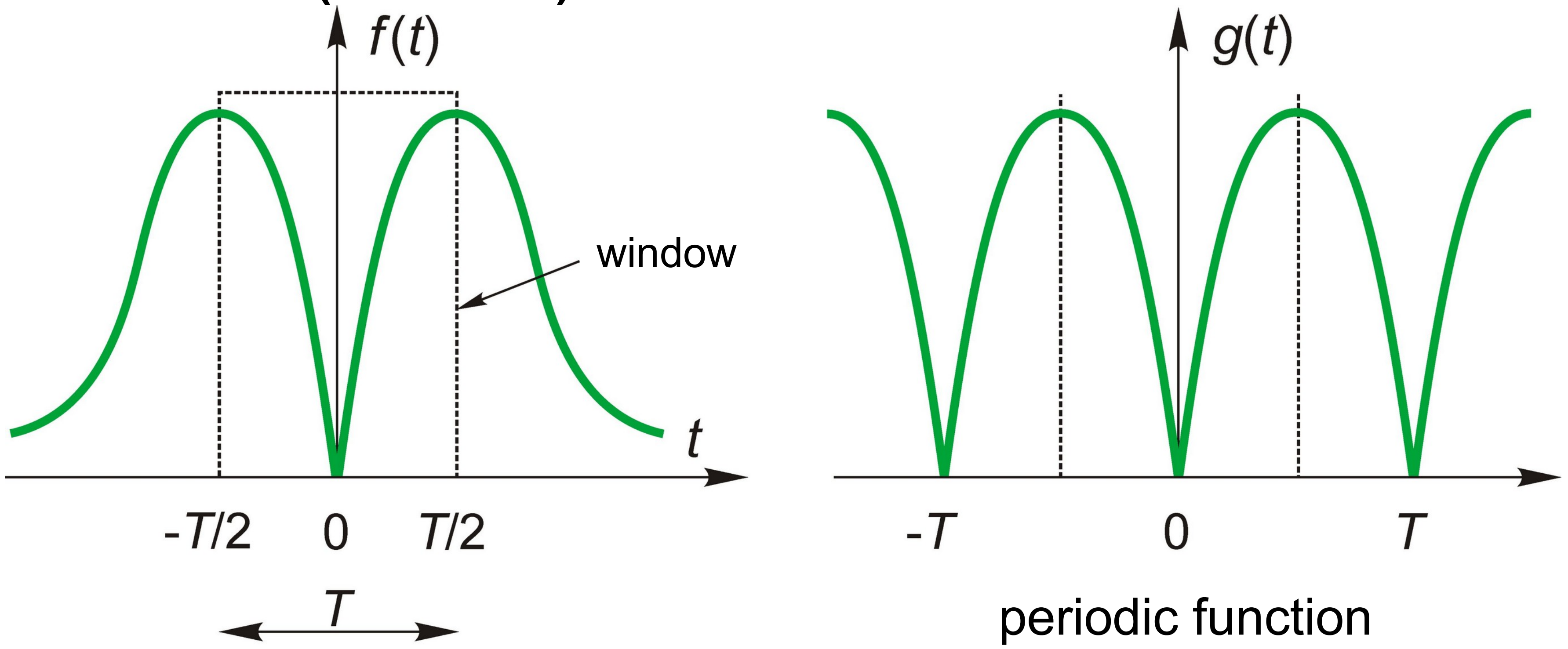
In [4]:
```
interval = (t, t0-T, t0+2*T)
p1 = sympy.plot(f(t), interval, show=False)
p2 = sympy.plot(analytic_approx, interval, show=False)
p2[0].line_color = 'red'
p1.extend(p2)
p1.show()
```



Insted of a symbolic solution a numerically equivalent approximation may also be given with SymPy's `mpmath` module.

Continuous Fourier transform (CFT)

- a periodic $g(t)$ function becomes nonperiodic for the limit ($T \to \infty$):



Fourier Transform (CFT)

- direct and inverse transform:

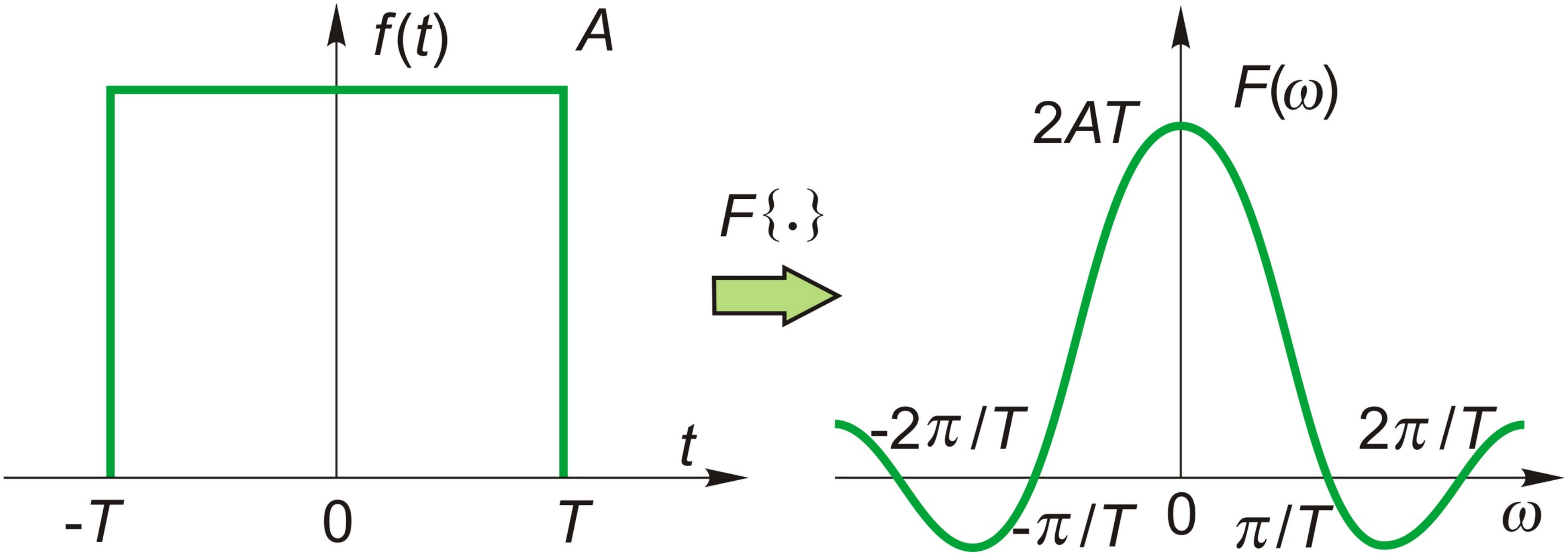
$$\boldsymbol{F}(\omega)=\int_{-\infty}^{\infty} f(t)\, e^{-i\omega t}\, dt=\boldsymbol{F}\{f(t)\}$$

$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty} \boldsymbol{F}(\omega)\, e^{i\omega t}\, d\omega=\boldsymbol{F}^{-1}\{\boldsymbol{F}(\omega)\}$$

$e^{i\omega t}=\cos\omega t+i\sin\omega t$ **periodic** function

Properties of the CFT

- transform *F(ω)* of an **even** function *f(t)* is **real**



sinc(ω) function

- **linearity**

$$\boldsymbol{F}\left\{\alpha f(t)+\beta g(t)\right\}=\alpha\,\boldsymbol{F}\left\{f(t)\right\}+\beta\,\boldsymbol{F}\left\{g(t)\right\}=\alpha\,\boldsymbol{F}(\omega)+\beta\,\boldsymbol{G}(\omega)$$

- **shift (phase change)**

$$\boldsymbol{F}\left\{f(t-t_0)\right\}=\boldsymbol{F}\left\{f(t)\right\}e^{-i\omega t_0}=\boldsymbol{F}(\omega)\,e^{-i\omega t_0}$$

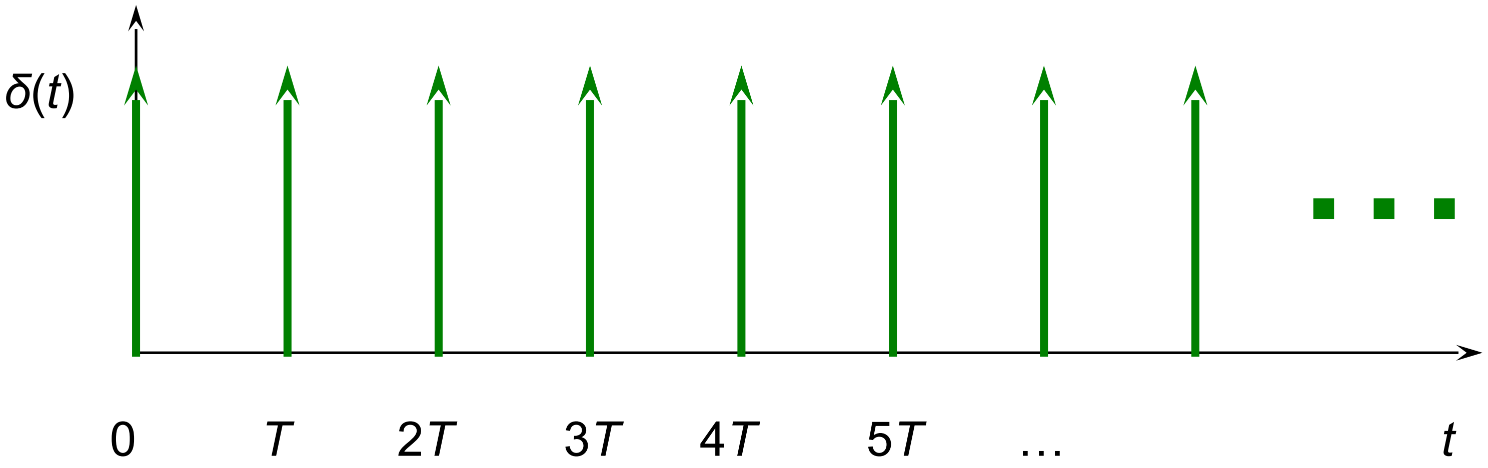
- **convolution**

$$\boldsymbol{F}\left\{f(t)*h(t)\right\}=\boldsymbol{F}(\omega)\,\boldsymbol{H}(\omega)$$

Fourier transform of the convolution is the **product** of the transforms of its component functions

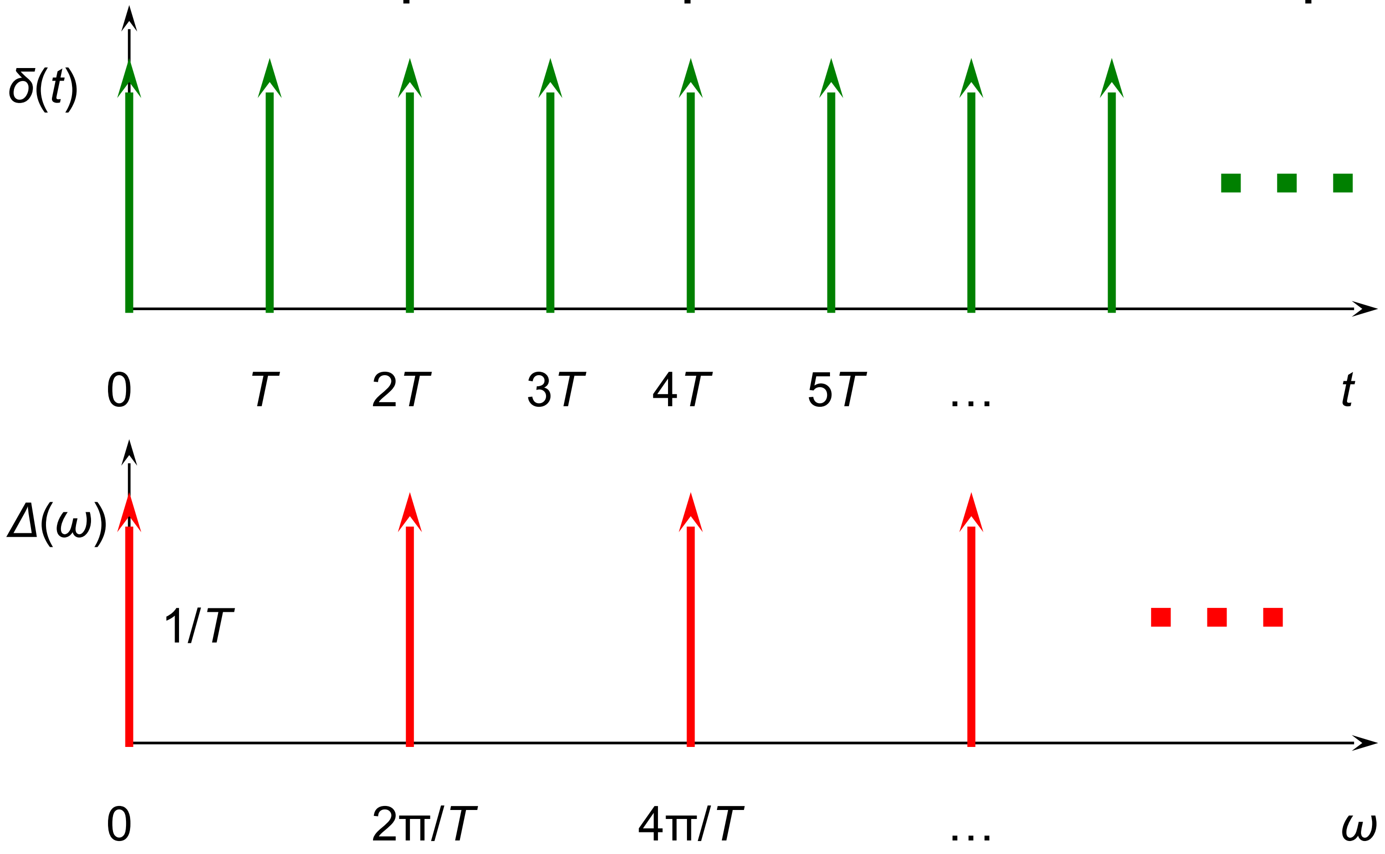
Dirac delta impulse function

- Diract delta „function” $\delta(t)$: zero everywhere except at t where it is infinitely large, but its integral is finite
- Useful abstraction like point mass
- sequence of impulses



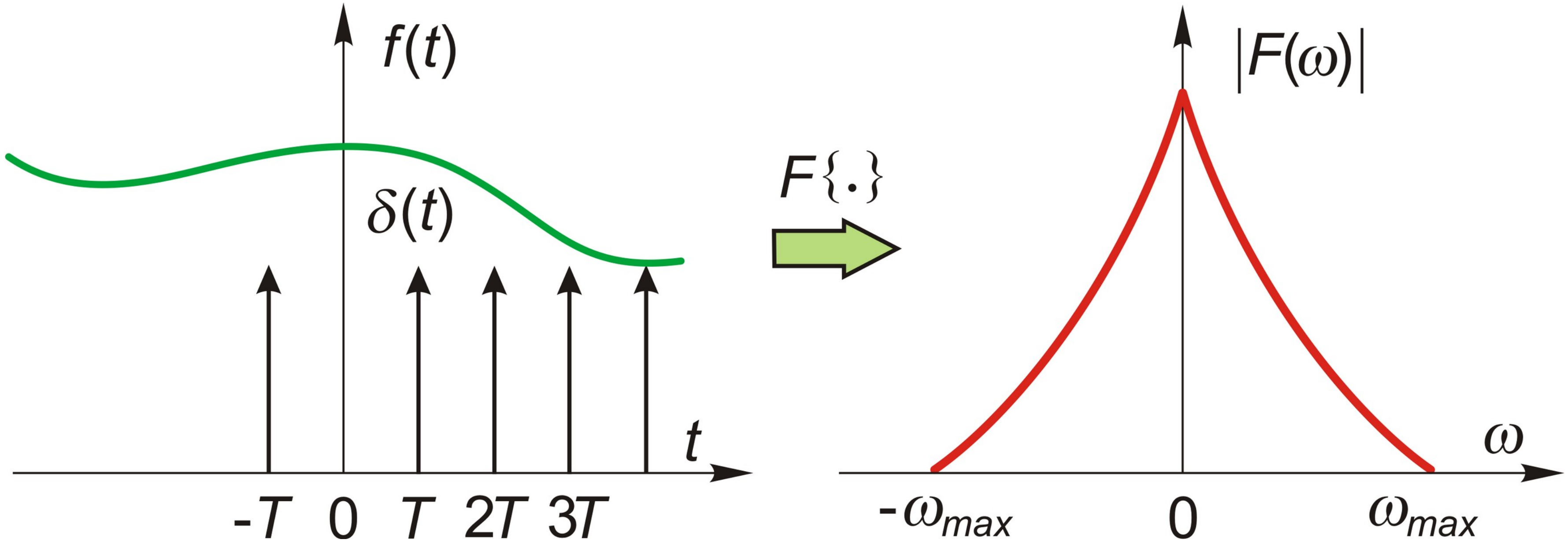
Fourier transform of impulse sequence

- CFT of an impulse sequence is another impulse sequence



Sampling

- sampling of $f(t)$ is achieved by multiplication with a sequence of impulses (sampling sequence):

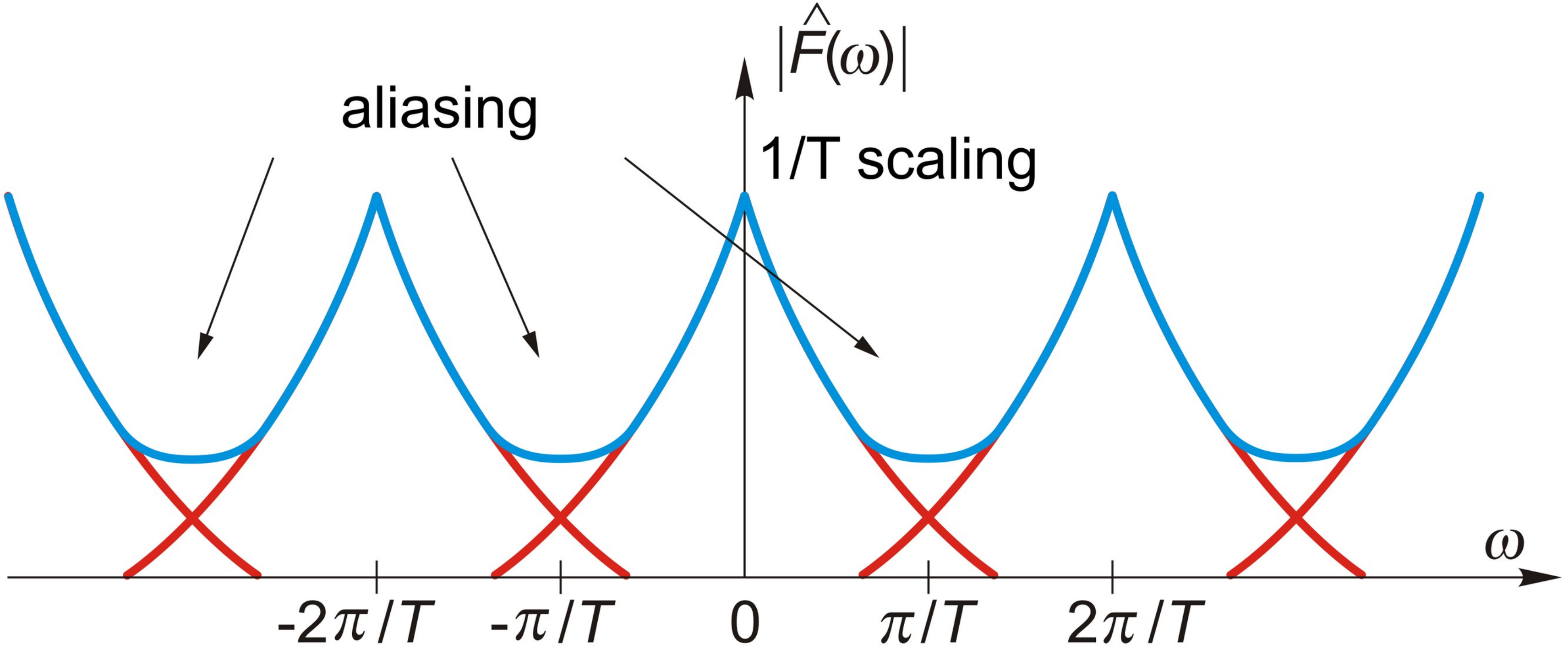


Fourier transform (CFT) of a sampled function

- CFT of a sampled function (according to the inverse convolution theorem) is the convolution of $f(t)$ with the Fourier transform of the sampling sequence
- copies of the transform $F(\omega)$ of $f(t)$ (scaled by $1/T$) are placed with the frequency interval $2\pi/T$ corresponding to the sampling (circular) frequency and added
- this is called the Discrete Fourier Transform (**DFT**)

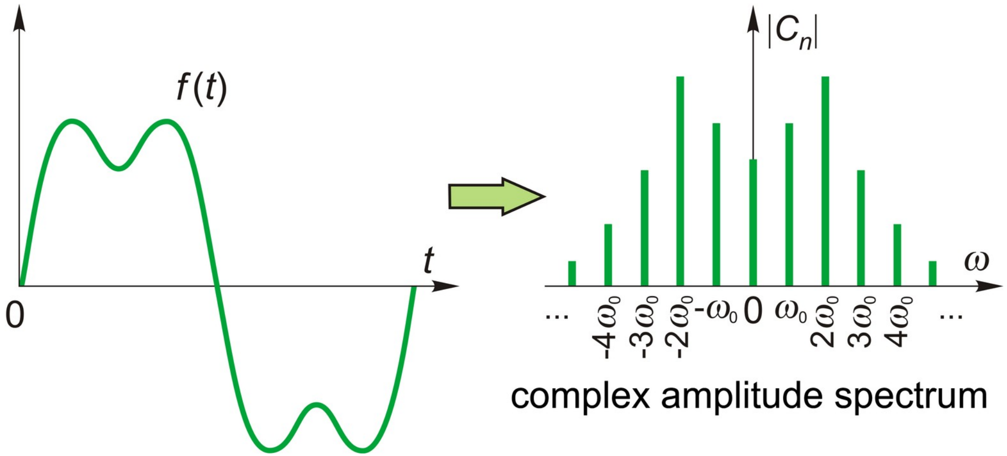
Discrete Fourier Transform (DFT)

- DFT is periodic with circular frequency $2\pi/T$
- neighbouring copies may overlap:



Discrete ↔ periodic

- Principle: discreteness in one domain induces periodicity in the other ($t \leftrightarrow \omega$)

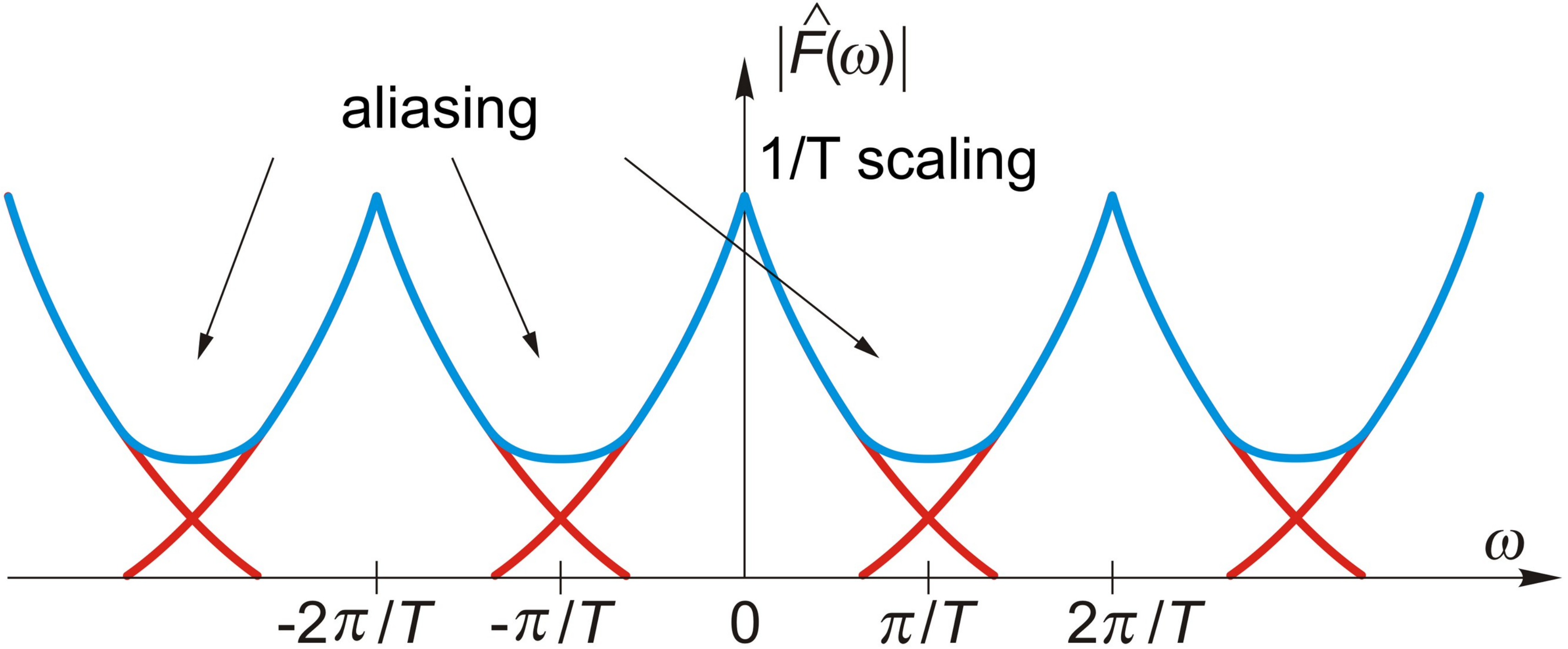


- Example: a sphere is doubly periodic, hence spherical harmonic expansion is doubly discrete: $\sum\sum$

Sampling theorem

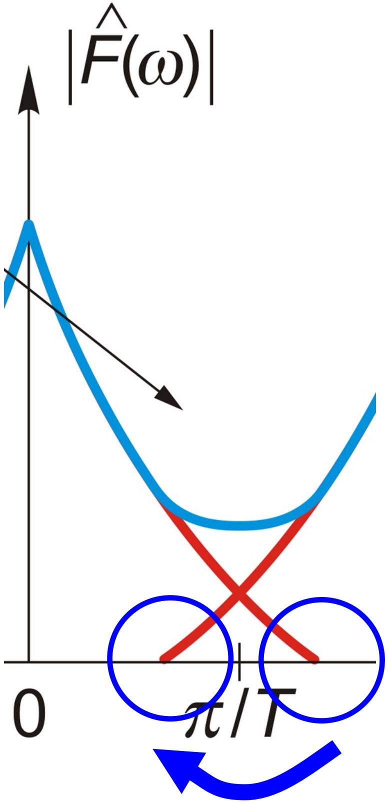
- No aliasing occurs if

$$\omega_{\max} \leq \frac{\pi}{T}$$



Aliasing

- Aliased high frequencies appear as low frequency bias.



original image

wavy (Moiré) pattern

Nyquist-Shannon theorem

- minimal sampling (Nyquist) period: $T_{\min} = \dfrac{\pi}{\omega_{\max}}$

- minimal sampling (Nyquist) circular frequency:

$$\omega_{\text{Nyquist}} = \frac{2\pi}{T_{\min}} = 2\omega_{\max}$$

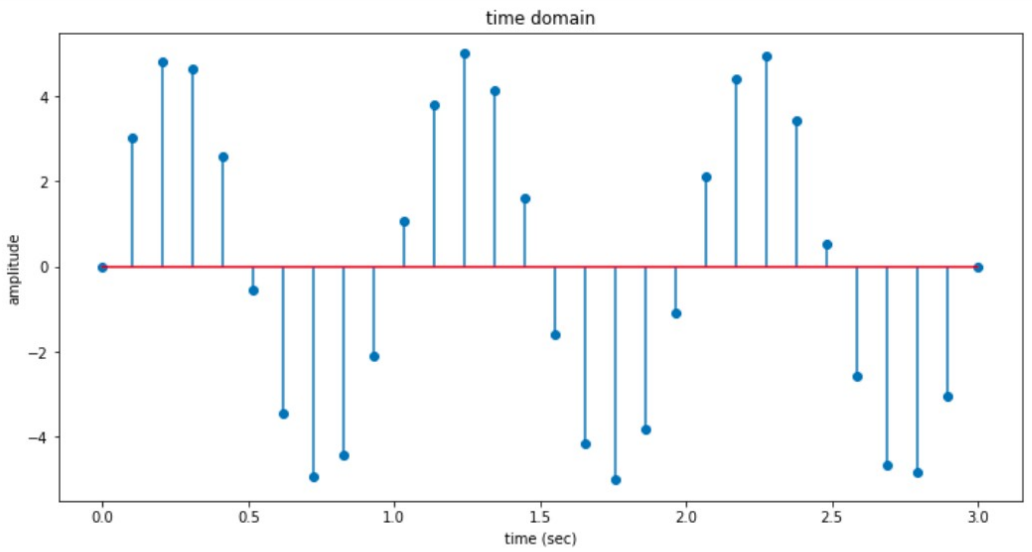
any signal must be sampled with a frequency that is at least **twice** of the highest frequency in the signal

Consequences (DFT)

- If the Nyquist-Shannon theorem is fulfilled, i.e sampling frequency is at least twice of the maximum signal frequency, then copies of the transforms in DFT do not overlap and there is no aliasing, hence one copy is enough (this copy will be scaled by T)

Example (Jupyter notebook)

https://nbviewer.jupyter.org/github/gyulat/Fourier/blob/master/FFT_en.ipynb



FFT calculation

We use NumPy function for calculation. This function evaluates the following sum:

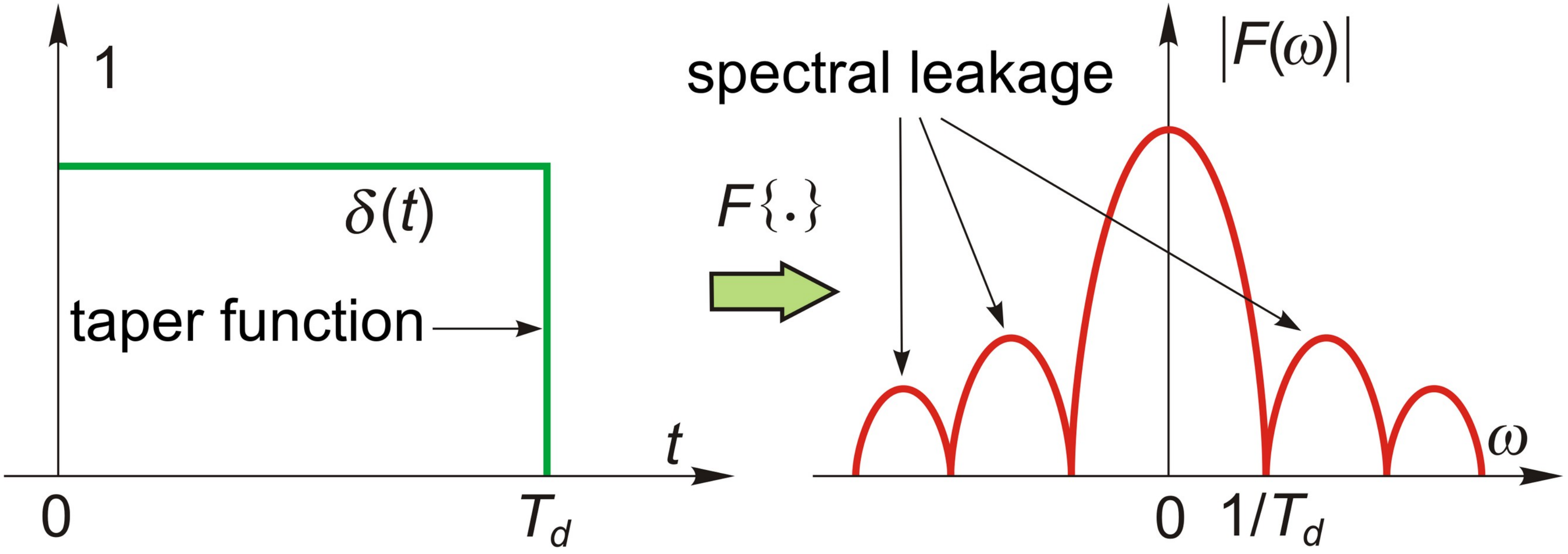
$$y[k] = \sum_{n=0}^{N-1} e^{-2\pi j \frac{kn}{N}} x[n]$$

Only a finite number of samples

- theoretically an infinity of samples are required
- in practice our data are restricted to the domain $t \in (t_{\mathrm{min}}, t_{\mathrm{max}})$
- what will be the effect in case of a continuous signal $f(t)$?

Spectral leakage

- product with a taper function (→convolution!)



- signal power ‘leaks’ to neighbouring frequencies

DFT transform pair

$$F_k = \sum_{n=0}^{N-1} f_n e^{-ink\,\Delta\omega T}$$

$k = 0, \ldots, N - 1$

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} F_k e^{ink\,\Delta\omega T}$$

$n = 0, \ldots, N - 1$

- operation count of DFT is N^2 complex multiplications and $N(N-1)$ complex additions

Fast Fourier Transform (FFT)

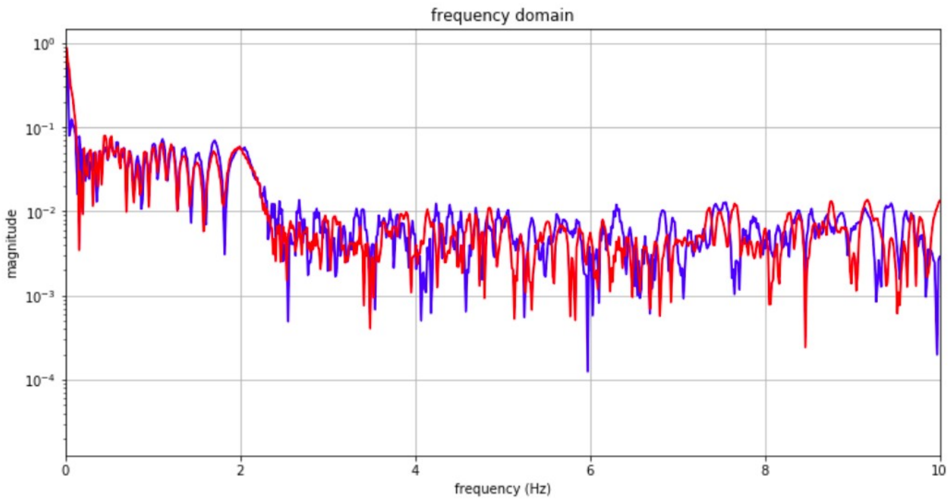
- operation count of DFT can be reduced
  - operation count for a binary number ($N=2^k$) of data can be decreased from N^2 to N $\log_2 N$
- Cooley and Tukey (1965)
  - time series analysis of seismic data
- Danielson and Lánczos (1942)
- Gauss (1805)
  - interpolation of asteroid orbits

Reduction of processing time by FFT

- FFT works for non-binary (2^k) data as well (mixed-radix FFT, which is effective when the number of data is a product of small prime factors, e.g. 2, 3, 5)
- There can be an enormous difference between the operation counts of DFT and FFT!
  - $N=10^9$ data, 1 GFLOPS processon (1 ns cycle time)
    - FFT: 45 seconds
    - DFT: 63 years

FFT spectrum of acceleration sensor data (Jupyter notebook)

https://nbviewer.jupyter.org/github/gyulat/Fourier/blob/master/FFT_demo_en.ipynb



Let us see the effect of Blackman tapering:

In [5]:
```
from scipy.signal import blackman
N = len(t)
w = blackman(N)
Ax = np.fft.rfft(ax*w)
Ay = np.fft.rfft(ay*w)
```

Applications

There are many applications of FFT, just a few are:

- digital signal and image processing, filtering, optical systems, telecommunication
- linear system analysis
- discrete signal convolution, deconvolution
- differential equations
- antenna design (interferometry, SAR, InSAR), image referencing
- matching DNA sequences (MAFFT)
- analyisis of closed curves, surfaces (medical image processing, optical character recognition, OCR)

More interactive Jupyter notebooks

Digital signal processing:

http://nbviewer.jupyter.org/github/spatialaudio/digital-signal-processing-lecture/blob/master/index.ipynb

Signal processing with Python:

https://github.com/unpingco/Python-for-Signal-Processing/blob/master/README.md