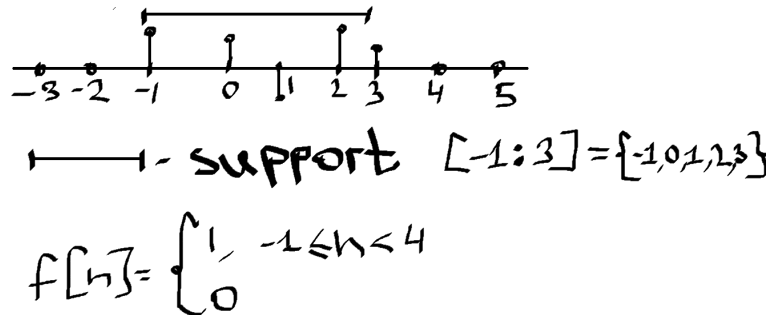


Discrete Signal Processing

A one-dimensional discrete signal is a sequence of real (or complex) numbers $f[n], n \in \mathbb{Z}$, also written $(f[n])_{n \in \mathbb{Z}}$

Support

The support of a discrete signal f defined on \mathbb{Z} is the smallest connected set where the signal has non zero values. If the support is compact, the signal is of finite length or finite.



Stable

A discrete signal is stable if

$$f \in L^1(\mathbb{Z}) \iff E_f = \sum_{n \in \mathbb{Z}} |f[n]| < \infty$$

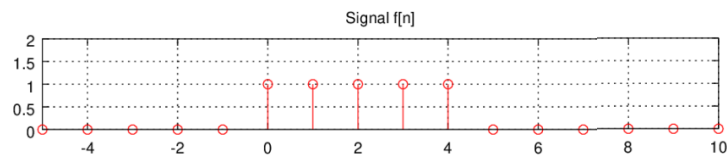
Finite Energy

A discrete signal is of finite energy if

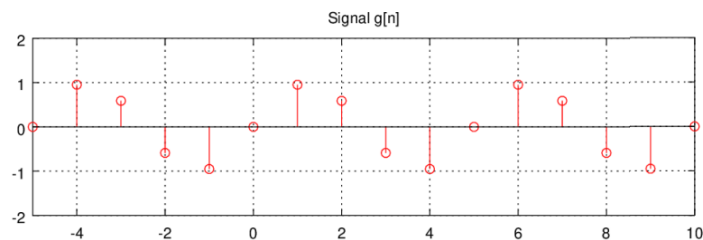
$$f \in L^2(\mathbb{Z}) \iff \sum_{n \in \mathbb{Z}} |f[n]|^2 < \infty$$

Examples

$$f[n] = \begin{cases} 1 & 0 \leq n \leq 4 \\ 0 & \text{otherwise} \end{cases}$$

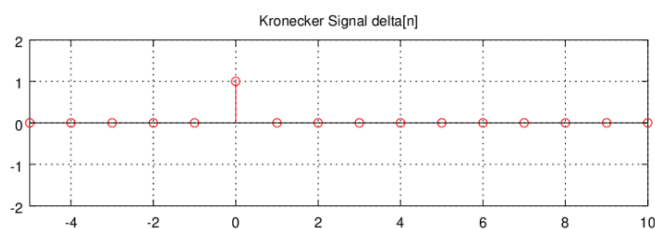


$$g[n] = \sin\left(\frac{2\pi}{5}n\right)$$



Kronecker signal:

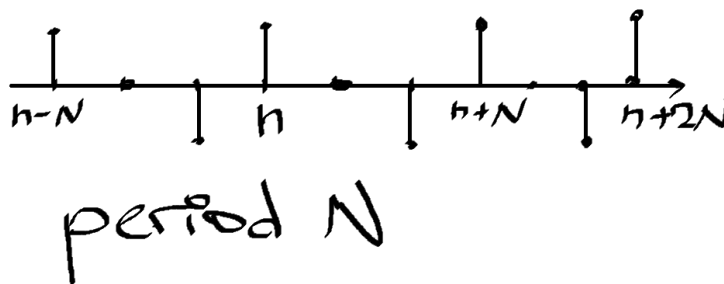
$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases}$$



Operations

Periodic

A one-dimensional discrete signal is periodic of period $N \in \mathbb{N}$ if $\forall n \in \mathbb{Z} : f[n] = f[n + kN], k \in \mathbb{Z}$



Examples

1. $f[n] = (-1)^n, n \in \mathbb{Z}, N = 2$
2. $g[n] = \sin\left(\frac{2\pi}{5}n\right), g[n] = g[n + N], \sin\left(\frac{2\pi}{5}n\right) = \sin\left(\frac{2\pi}{5}n + \frac{2\pi}{5}N\right), \frac{2\pi}{5}N = 2\pi \rightarrow N = 5$

Translation

The translation by $m \in \mathbb{Z}$, of a one-dimensional discrete signal is defined $\forall n \in \mathbb{Z}$ by $\tau_m f[n] = f[n - m]$

$$\tau_m \delta[n] = \delta[n - m] = \delta_m[n], n \in \mathbb{Z}$$

Convolution

The convolution $h = f \circ g$ is defined by

$$\forall n \in \mathbb{Z} : h[n] = \sum_{k=-\infty}^{+\infty} f[n-k]g[k]$$

If f or g are of finite length, h is well defined

In practice, all of the signals will be of finite length/duration.

Convolution is a fundamental mathematical operation in signal processing, used to combine two signals to produce a third signal.

Here's how the convolution operation works: 1. For each point in time (continuous time) or sample (discrete time) in the resulting signal “y,” you take a weighted sum of the product of the two input signals “f” and “g.” 2. The weights are determined by the values of the second signal “g,” which are typically flipped (reversed) in time or order before multiplication and summing. This reversal is what makes convolution different from other operations like cross-correlation. 3. The convolution operation is performed over a range of values, often extending to infinity in continuous time or to a suitable range in discrete time, depending on the problem.

The result of the convolution represents the combined effect of the two signals. In practical terms, convolution is used for various purposes, such as filtering, signal smoothing, and simulating the response of linear time-invariant systems. In image processing, for instance, it is used for tasks like blurring, edge detection, and feature extraction.

Fourier

The Fourier transform of discrete signal f is:

$$\forall v \in \mathbb{R}, \hat{f}(v) = \sum_{-\infty}^{+\infty} f[n]e^{-ivn} = A_f(v)e^{i\Phi_f(v)}$$

This is sometimes also referred to as Fourier Transform in Discrete Time (FTDT).

To a real sequence $(f[n])_{n \in \mathbb{Z}}$ is associated a 2π -**periodic function** $\hat{f} : \mathbb{R} \rightarrow \mathbb{C}; v \rightarrow \hat{f}(v)$

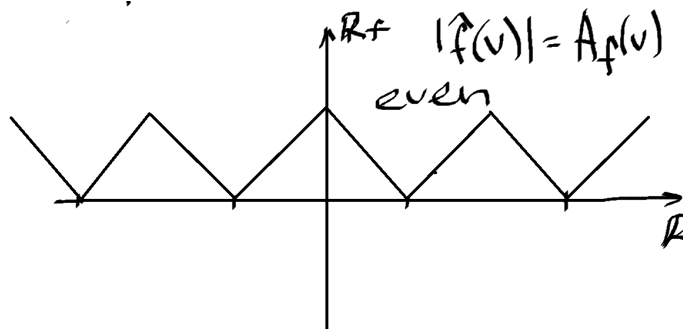
\hat{f} is 2π -periodic:

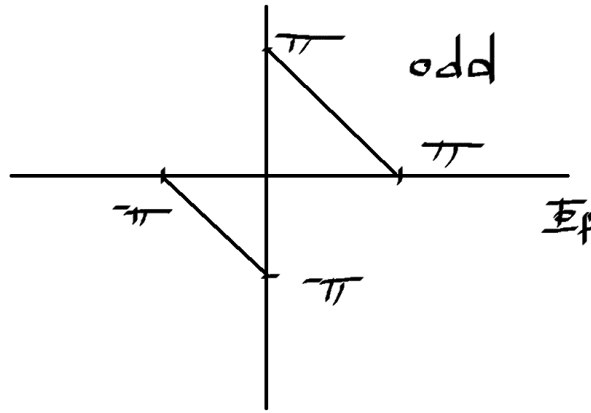
$$\hat{f}(v + 2\pi k) = \sum_{-\infty}^{+\infty} f[n]e^{-in(v+2\pi k)} = \hat{f}(v)$$

$$e^{-in(v+2\pi k)} = e^{-ivn} \cdot (e^{-2i\pi nk} = 1)(e^{xi} = \cos(x) + i \sin(x), e^{-2\pi i} = \cos(-2\pi) + i \sin(-2\pi) = 1)$$

*Remark: if $(f[n])_{n \in \mathbb{Z}}$ is **periodic**: $f_p[n] = f_p[n + mN], n \in \mathbb{Z}$ the FTDT is **not defined**. Periodic signals are **unstable**.*

The only difference between Fourier Transform in continuous case and discrete case is that in discrete case Fourier Transform, Amplitude and Phase are 2π -periodic.





Properties

$$\begin{aligned}\hat{f}(v) &= \overline{\hat{f}(-v)} \\ |\hat{f}(v)| &= \hat{f}(-v) \\ \Phi(\hat{f}(-v)) &= -\Phi(\hat{f}(v))\end{aligned}$$

The low frequency content at $\hat{f}(0)$ will repeat at $v = 2\pi k, k \in \mathbb{Z}$

The high frequencies are localized at $v = \pi + 2\pi k, k \in \mathbb{Z}$

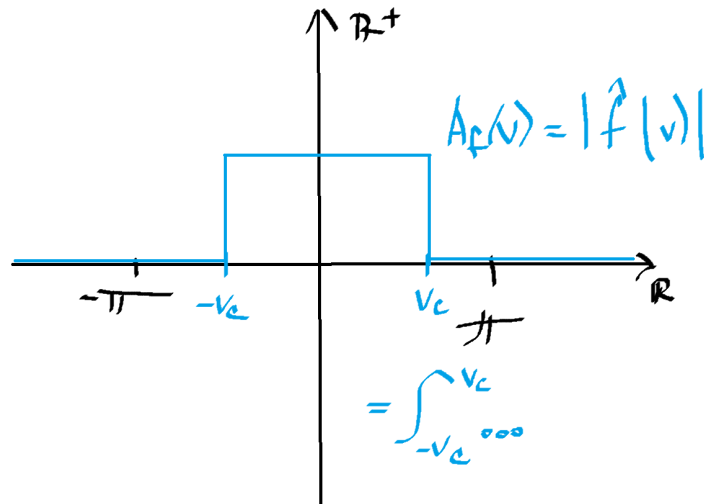
$$\begin{aligned}\widehat{f * g}(v) &= \hat{f}(v)\hat{g}(v) \\ \widehat{fg}(v) &= \frac{1}{2\pi}\hat{f}(v) * \hat{g}(v)\end{aligned}$$

$$\begin{aligned}\widehat{\tau_m f}(v) &= \hat{f}(v)e^{-imv} = A(v)e^{i[\Phi(v)-mv]} \\ \widehat{f[n]e^{i\alpha n}} &= \hat{f}(v - \alpha) = \tau_\alpha \hat{f}(v)\end{aligned}$$

Inverse Fourier Transform

$$\forall n \in \mathbb{Z}, f[n] = \frac{1}{2\pi} \int_{-\pi}^{+\pi} \hat{f}(v)e^{+ivn} dv = \frac{1}{2\pi} \int_{-\pi}^{+\pi} A_f(v)e^{i(nv+\Phi_f(v))} dv$$

If $A = 0 \rightarrow$ frequency is not present.



One period of \hat{f} , in the continuous frequency variable, allows reconstruction.

Amplitude

$$A_f(v) = |\hat{f}(v)|$$

Bandwidth

The bandwidth is the width of a subset of $[-\pi, \pi]$ centered at 0.

Periodic Discrete Signals

Restrict signals to N values (one period).

Discrete Fourier Transform of Periodic Signals (DFT)

$$\hat{f}_p[k] = \sum_{n=0}^{N-1} f_p[n] e^{-in(\frac{2\pi}{N}k)}, \quad k = 0, 1, \dots, N-1$$

$f_p[n]$ is N periodic

$$(\frac{2\pi}{N}k) \cong v$$

Inverse Discrete Fourier Transform

$$f_p[n] = \frac{1}{N} \sum_{k=0}^{N-1} \hat{f}_p[k] e^{in(\frac{2\pi}{N}k)}, \quad n = 0, 1, \dots, N-1$$

In the time and in the frequency domain, a period characterizes \hat{f} and f .

Convolution

For periodic signals the usual convolution is not defined (in general no convergence). The following definition uses the N -periodicity of the signals.

Circular Convolution If f_p and g_p have period N , we define $h_p = f_p *_c g_p$ with

$$h_p[n] = (f_p *_c g_p)[n] = \sum_{m=0}^{N-1} f_p[n-m]g_p[m], \quad n = 0, 1, \dots, N-1$$

The circular convolution becomes multiplication in frequency domain.

For DFT we have:

$$\widehat{h_p}[k] = \widehat{f_p}[k]\widehat{g_p}[k], \quad k = 0, 1, \dots, N-1$$

Example | Exercise 10 Let $f_1[n]$ and $f_2[n]$ be discrete signals of lengths N_1 and N_2 , respectively, $i = 1, 2$

1. What is the length of the convolution signal $g = f_1 * f_2$? The question can be restated as: for which values of $n \in \mathbb{Z}$: $f_1[n-k]f_2[k] \neq 0$?

Suppose $f_1[k] \neq 0$ for $k = n_1, n_1 + 1, \dots, N_1 + n_1 - 1$. In other words, f_1 starts at n_1 and ends at $N_1 + n_1 - 1$. $\text{len}(f_1) = (N_1 + n_1 - 1) - n_1 + 1$

For f_2 : $\text{supp}(f_2) = [n_2, N_2 + n_2 - 1]$

$f_2[k] \neq 0$ for k s.t. $n_2 \leq k \leq N_2 + n_2 - 1$

$f_1[n-k] \neq 0$ for $n-k$ s.t. $n_1 \leq n-k \leq N_1 + n_1 - 1$

\Rightarrow get by summing up: $n_1 + n_2 \leq n \leq N_1 + N_2 + n_1 + n_2 - 2$

Thus, $g[n] = \sum_{k \in \mathbb{Z}} f_1[n-k]f_2[k] \neq 0$ for $n \in [n_1 + n_2, N_1 + N_2 + n_1 + n_2 - 1]$

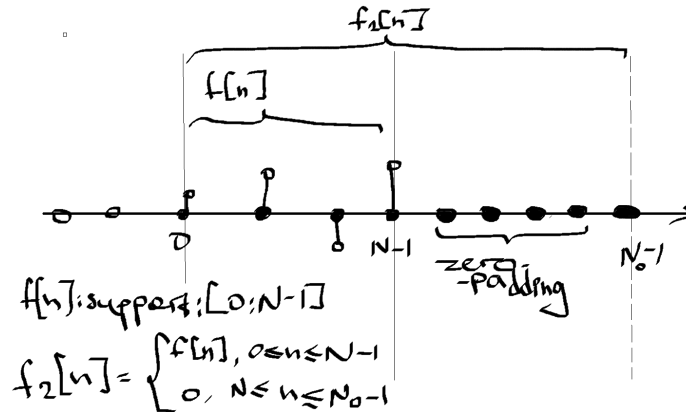
$\text{len}(g) = (N_1 + N_2 + n_1 + n_2 - 1) - (n_1 + n_2) + 1 = N_1 + N_2 - 1$

If $N_1 = N_2 = N$ then $\text{len}(g) = 2N - 1$

2. For the rest of the questions, suppose $N = N_1 = N_2$. What is the complexity in real scalar operations of a (full) computation of $g = f_1 * f_2$? $\text{len}(g) = 2N - 1$, thus we have $(2N - 1)$ coefficients to compute. For each coefficient we have N multiplications and $N - 1$ additions. Thus, we get $N + N - 1 = 2N - 1$ operations for each of the terms of $g[n]$. \Rightarrow total complexity is $(2N - 1)(2N - 1) = (2N - 1)^2$ or $O(N^2)$
3. We want to use the Fast Fourier Transform (FFT) and the inverse FFT (iFFT) to compute $g[n]$. To what length must the signals f_1 and f_2 be completed in order to recover the correct result, considering the respective lengths? $g = f_1 * f_2$
 $\hat{g} = \hat{f}_1 \cdot \hat{f}_2$
 $\text{len}(f_1) = N, \text{len}(f_2) = N, \text{len}(g) = 2N - 1$
DFT of f_1 N coefficients, DFT of f_2 N coefficients, but DFT of g $(2N - 1)$ coefficients
Thus we need to complete f_1 and f_2 with zeros up until we get a signal of length $(2N - 1)$:
 $\hat{g}[k] = \hat{f}_1[k] \cdot \hat{f}_2[k], 0 \leq k \leq 2N - 2$
 $\hat{g}[k] = \hat{g}(\frac{2\pi}{2N-1}k)$
4. What is the complexity of the method mentioned in question 3?
 1. By FFT: $2(2N - 1) \log(2N - 1)$ (f_1 and f_2)
 2. Multiply $(2N - 1)$ times two complex numbers
 3. Make inverse FFT of $(2N - 1)$ length signal: $(2N - 1) \log(2N - 1)$
 4. Thus we get: $3(2N - 1) \log(2N - 1) + (2N - 1) \approx_{N \rightarrow \infty} N \log(N)$
 5. Complexity is $O(N \log(N)) < O(N^2)$
 6. So, it is better to go to compute $FFT\hat{g} = \hat{f}_1\hat{f}_2 \rightarrow iFFT\hat{g}$ to get g than $g = f_1 * f_2$

Zero-padding

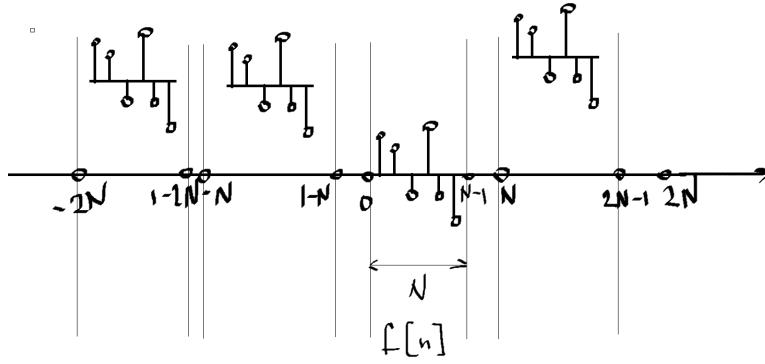
Zero-padding: We can complete a signal by adding zeros: artificially make signal longer in time. By zero-padding we obtain any length.



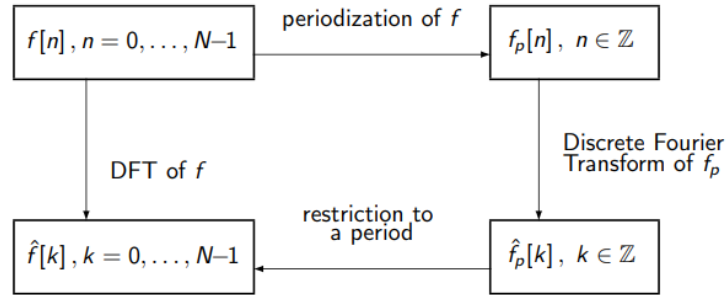
Periodization

Periodization: make signal periodic. By periodization we obtain an infinite-length, periodic signal.

$$f_p[n] = \sum_{m \in \mathbb{Z}} f[n + mN] = f[n \bmod N]$$



DFT and DFTP



Fast Fourier Transform

$$\hat{f}[k] = \sum_{n=0}^{N-1} f[n] e^{-in(\frac{2\pi}{N}k)}, \quad k = 0, 1, \dots, N-1$$

Fast Inverse Fourier Transform

$$f[n] = \frac{1}{N} \sum_{k=0}^{N-1} \hat{f}[k] e^{in(\frac{2\pi}{N}k)}, \quad n = 0, 1, \dots, N-1$$

- Direct computations are $O(N^2)$ real scalar operations.
- Reorganizing computations leads to the Fast Fourier Transform (FFT) which is $O(N \ln(N))$.

DFT and FTDT Properties

We have finite discrete signal. Then:

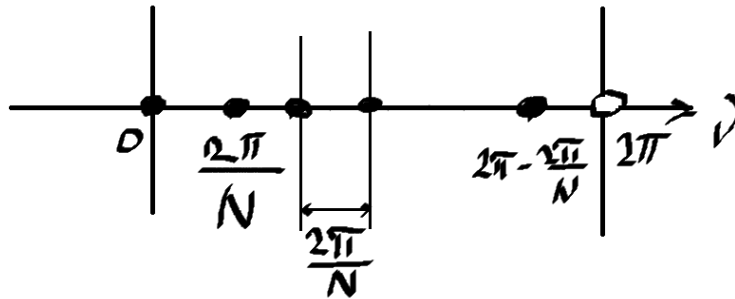
$$\hat{f}[k] = \hat{f}\left(\frac{2\pi}{N}k\right), \quad k = 0, 1, \dots, N-1$$

$$\begin{cases} \hat{f}(v) = \sum_{n=0}^{N-1} f[n] e^{-inv} & v \in [0, 2\pi) \\ \hat{f}[k] = \sum_{n=0}^{N-1} f[n] e^{-in\frac{2\pi}{N}k} \end{cases}$$

$$v = \frac{2\pi}{N}k, \quad 0 \leq k \leq N-1$$

$$\hat{f}[0] = \hat{f}(0)$$

$$\hat{f}[1] = \hat{f}\left(\frac{2\pi}{N}\right), \quad k = 1$$



We have information at 0 and at 2π the same, which is why the borders are $[0, 2\pi)$

The DFT coefficients are samples at rate $T = \frac{2\pi}{N}$ of the FTDT on $[0, 2\pi)$. In other words, the DFT is a sampling of the FTDT by $\frac{2\pi}{N}$: the larger N , the better the information.

Example

$$f[n] = \begin{cases} 1 & 0 \leq n \leq 4 \\ 0 & 5 \leq n \leq 9 \end{cases}$$

$$\text{supp}(f) = [0 : 9], N = 10$$

$$\hat{f}(v) = \sum_{n \in \mathbb{Z}} f[n] e^{-inv} = \sum_{n=0}^{N_0} f[n] e^{-inv}$$

$$\hat{f}[k] = \sum_{n=0}^{N-1} f[n] e^{-in \frac{2\pi}{N} k} = \sum_{n=0}^{N_0} f[n] e^{-in \frac{2\pi}{N} k}$$

Remark: summing up to N_0 , because 0's after N_0

Symmetries in DFT Coefficients for Real Finite Signals

- $\hat{f}[k]$ is N -periodic

$$\hat{f}[-k] = \hat{f}[-k + N]$$

$$\hat{f}[k] = \overline{\hat{f}[-k]} = \overline{\hat{f}[N - k]}$$

$$\hat{f}[1] = \hat{f}[N - 1]$$

- For $0 \leq k \leq N - 1$ real, we have $\hat{f}[k] = \overline{\hat{f}[-k]}$, thus

$$|\hat{f}[k]| = |\hat{f}[-k]| \text{ and } \arg(\hat{f}[k]) = -\arg(\hat{f}[-k]), \quad \arg = \Phi_f$$

- By N periodicity: for $1 \leq k \leq N-1$, $\hat{f}[k] = \overline{\hat{f}[N-k]}$, thus

$$|\hat{f}[k]| = |\hat{f}[N-k]| \text{ and } \arg(\hat{f}[k]) = -\arg(\hat{f}[N-k])$$

- We always have $\hat{f}[0] = \sum_{n=0}^{N-1} f[n] \in \mathbb{R}$, and if N is even:

$$\hat{f}[N/2] = \overline{\hat{f}[N - \frac{N}{2}]} = \overline{\hat{f}[\frac{N}{2}]} \text{ is real too}$$

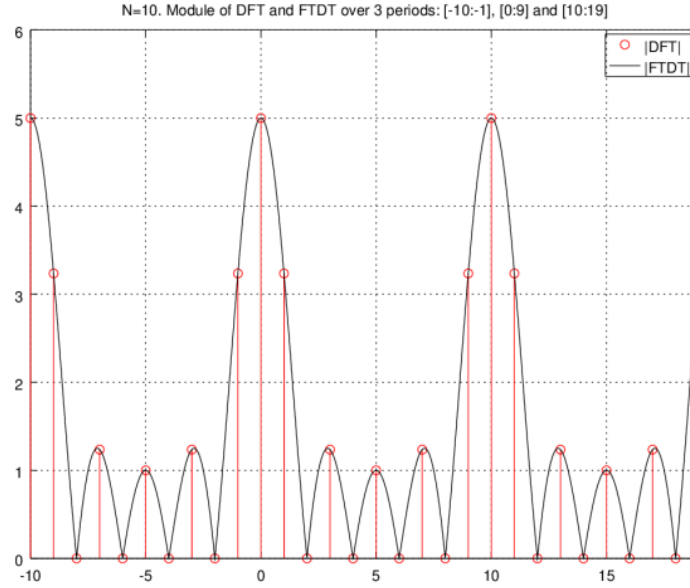
Remark: if $z = \bar{z}$, then we are on the real line.

To represent N DFT coefficients, one uses the range:

$$[-\frac{N}{2} : \frac{N}{2} - 1], \text{ if } N \text{ is even}$$

$$[-\frac{N-1}{2} : \frac{N-1}{2} - 1], \text{ if } N \text{ is odd}$$

The coefficients close to $k = 0$ and $k = N-1$ are the low frequencies and around $k = \lfloor \frac{N}{2} \rfloor$ are the high frequencies of the signal.

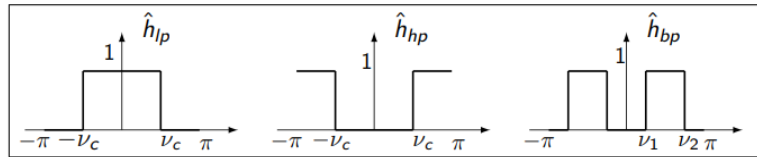


Filter In Discrete Time | Ideal Filter

Let H be a filter defined by the discrete signal $(h[n])_{n \in \mathbb{Z}}$. To a (finite) discrete signal x , we associate the discrete signal $y = H[x]$. These signals are related in temporal and frequency domain by

$$y[n] = (h * x)[n], n \in \mathbb{Z} \quad \text{and} \quad \hat{y}(v) = \hat{h}(v)\hat{x}(v), v \in [-\pi, \pi]$$

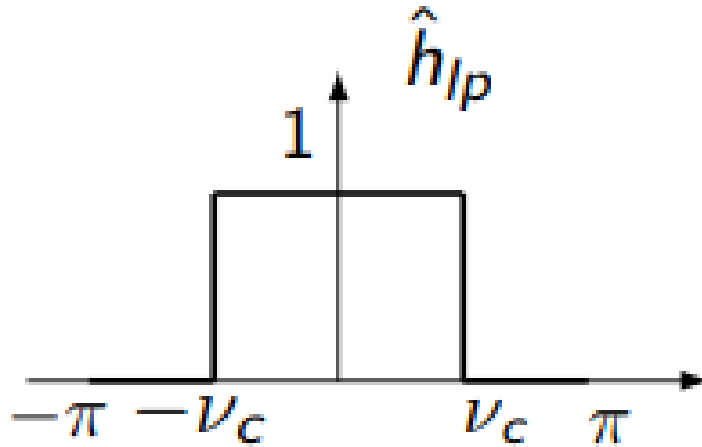
Ideal filters are designed to cut off specified frequencies of the input signal. One obtains ideal **low-pass**, **high-pass** and **band-pass** filters by defining (in the frequency domain) \hat{h} :



In here, v_c is the cutoff frequency (everything is 0 outside of the $[-v_c, v_c]$) and $[v_1, v_2]$ is the allowed frequency band.

Ideal Low-pass Filter

$$\hat{h}_{LP}(v) = \begin{cases} 1, & |v| \leq v_c \\ 0, & v_c < |v| < \pi \end{cases}$$



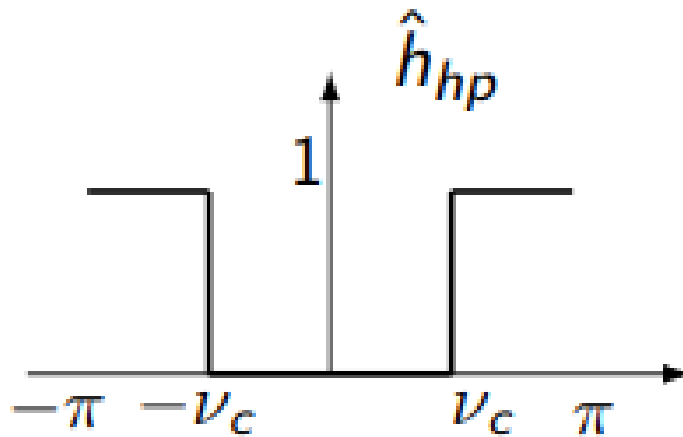
The **impulse response** is:

$$h_{LP}[n] = \frac{\sin(v_c n)}{\pi n}$$

It is not finite and leads to ringing artifacts (Gibbs phenomenon).

Ideal High-pass Filter

$$\hat{h}_{HP}(v) = \begin{cases} 0, & |v| \leq v_c \\ 1, & v_c < |v| < \pi \end{cases}$$



The **impulse response** is:

$$h_{HP}[n] = \delta[n] - \frac{\sin(w_c n)}{\pi n}$$

Ideal Band-pass Filter

$$\hat{h}_{BP}(v) = \begin{cases} 1, & v_1 \leq |v| \leq v_2 \\ 0, & |v| < v_1 \text{ or } v_2 < |v| \leq \pi \end{cases}$$

