UNIVERSITY OF CAMBRIDGE

3F1, Signals and Systems

PART V.1

Discrete Fourier Transform

Fulvio Forni (f.forni@eng.cam.ac.uk)

November 13, 2018

Goal of the lecture:

The DFT tool in digital filtering

DFT definition and comparison with other transforms

From $\mathcal{Z}$ to DTFT

z-transform:

$$\text{Signal: } x_k \quad \Rightarrow \quad X(z) := \mathcal{Z}(x) = \sum_{k=0}^{\infty} x_k z^{-k}$$

Discrete time Fourier transform:

Fourier transform of a sampled signal, sampling time T

$$\bar{x}_\omega := \mathrm{DTFT}(x) = \sum_{k=-\infty}^{\infty} x_k e^{-j\omega Tk}$$

Relation between $\mathcal{Z}$ and DTFT:

if $x_k = 0$ for $k < 0$ then

$$\bar{x}_\omega = X(z)_{|z=e^{j\omega T}}$$

From DTFT to DFT

Digital signal: $\{x_k\}$ $x_k = 0$ for $k < 0$

Discrete time Fourier transform:
Fourier transform of a sampled signal, sampling time T

$$\bar{x}_\omega := \mathrm{DTFT}(x) = \sum_{k=0}^{\infty} x_k e^{-j\omega T k}$$

- ω is continuous, how to represent it in a computer?
- infinite sum, how to compute it?

$$\Rightarrow$$

- Sampling a finite number of frequencies ω
  (small sampling $\rightarrow$ good approximation).
- Truncation, use a finite number of data points.

From DTFT to DFT

Digital signal: $\{x_k\}$ $\qquad x_k = 0$ for $k < 0$

Discrete time Fourier transform:

$$\bar{x}_\omega := \mathrm{DTFT}(x) = \sum_{k=0}^{\infty} x_k e^{-j\omega Tk}$$

Discrete Fourier transform:

$$\bar{x}_p := \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk}$$

- Finite sampling: $\omega = \frac{2\pi}{NT}$
- Finite computation: finite data points $0 \le k \le N$
- Finite representation: $0 \le p \le N-1$

periodicity $\quad \bar{x}_p = \bar{x}_{p+N} \quad$ since $e^{-j\frac{2\pi}{N}(p+N)} = e^{-j\frac{2\pi}{N}p}$

DFT and $\mathcal{Z}$

Digital signal: $\{x_k\}$ $\qquad x_k = 0$ for $k < 0$

<u>Discrete Fourier transform:</u>

$$\bar{x}_p := \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk} \qquad 0 \leq p \leq N-1$$

Relation with z-transform:
(for truncated signals $x_k = 0$ for $k < 0$ and $k \geq N$)

$$X(z) := \sum_{k=0}^{\infty} x_k z^{-k} = \sum_{k=0}^{N-1} x_k z^{-k}$$

$$\bar{x}_p = X(z)_{|z=e^{-j\frac{2\pi}{N}p}}$$

Let's understand the DFT...

Digital signal: $\{x_k\}$

Discrete Fourier transform:

$$\bar{x}_p := \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk}$$

Take

$$x := \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix}, \quad b(p, N) := \begin{bmatrix} e^{-j\frac{2\pi}{N}p\cdot 0} \\ e^{-j\frac{2\pi}{N}p\cdot 1} \\ \dots \\ e^{-j\frac{2\pi}{N}p\cdot(N-1)} \end{bmatrix}$$

Then

$$\bar{x}_p := b(p, N)'x$$

($b(p, N)'$ is the transpose of $b(p, N)$)

$\bar{x}_p$ is the projection of x on the base $b(p, N)$!

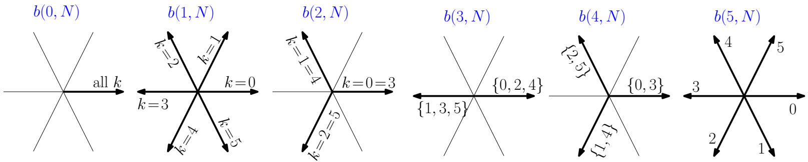
Digital signal: $\{x_k\}$

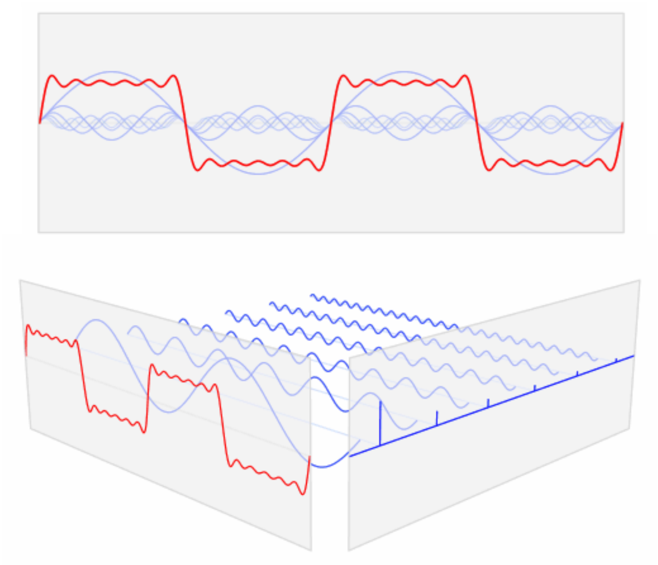
Discrete Fourier transform:

$$\bar{x}_p := \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk} = b(p,N)'x$$

$$x := \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix}, \quad b(p,N) := \begin{bmatrix} e^{-j\frac{2\pi}{N}p\cdot 0} \\ e^{-j\frac{2\pi}{N}p\cdot 1} \\ \cdots \\ e^{-j\frac{2\pi}{N}p\cdot (N-1)} \end{bmatrix}$$

Example N = 6: Animation





From Wikimedia Commons, by Lucas V. Barbosa

Discrete Fourier transform: $\bar{x}_p := b(p,N)'x$

- one DFT sample = product of two vectors (N operations)
- N DFT samples = $N \times N$ operations.

$$\bar{x} := \begin{bmatrix} \bar{x}_0 \\ \bar{x}_1 \\ \vdots \\ \bar{x}_{N-1} \end{bmatrix} = \underbrace{\begin{bmatrix} b(0,N)' \\ b(1,N)' \\ \dots \\ b(N-1,N)' \end{bmatrix}}_{=:B(N)} x$$

- inverse DFT = product by inverse matrix...

$$x := B(N)^{-1}\bar{x}$$

- Linearity: if $z = (x+y)$ then $\bar{z} = \bar{x} + \bar{y}$. For instance,

$$\bar{z} = B(N)(x+y) = B(N)x + B(N)y = \bar{x} + \bar{y}$$

Antitransform

DFT

$$\bar{x}_p := \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk}$$

Inverse DFT

$$x_n := \frac{1}{N}\sum_{p=0}^{N-1} \bar{x}_p e^{j\frac{2\pi}{N}pn}$$

Proof [inverse DFT]

$$\frac{1}{N}\sum_{p=0}^{N-1}\underbrace{\sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk}}_{\bar{x}_p} e^{j\frac{2\pi}{N}pn} = \frac{1}{N}\sum_{p=0}^{N-1}\sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}p(k-n)}$$

$$= \frac{1}{N}\sum_{k=0}^{N-1} x_k \underbrace{\sum_{p=0}^{N-1} e^{-j\frac{2\pi}{N}p(k-n)}}_{\substack{0 \text{ if } k \neq n, \\ N \text{ if } k = n.}}$$

$$= \frac{1}{N} x_n N \ = \ x_n$$

DFT

$$\bar{x}_p := \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk}$$

$$\Rightarrow$$

$$\bar{x}_p := b(p,N)'x$$

$$\Rightarrow$$

$$\bar{x} = \underbrace{\begin{bmatrix} b(0,N)' \\ b(1,N)' \\ \cdots \\ b(N-1,N)' \end{bmatrix}}_{B(N)} x$$

Inverse DFT

$$x_n := \frac{1}{N}\sum_{p=0}^{N-1} \bar{x}_p e^{j\frac{2\pi}{N}pn}$$

$$\Rightarrow$$

$$x_n := \frac{1}{N} b(-n,N)'\bar{x}$$

$$\Rightarrow$$

$$x = \underbrace{\frac{1}{N}\begin{bmatrix} b(0,N)' \\ b(-1,N)' \\ \cdots \\ b(-N+1,N)' \end{bmatrix}}_{B(N)^{-1}=\frac{1}{N}B(N)^*} \bar{x}$$

$$b(q,N)' := \begin{bmatrix} e^{-j\frac{2\pi}{N}q\cdot 0} & e^{-j\frac{2\pi}{N}q\cdot 1} & \ldots & e^{-j\frac{2\pi}{N}q\cdot(N-1)} \end{bmatrix}$$

DFT and inverse DFT use the same algorithm!

DFT and inverse DFT share the same properties: periodicity, linearity,...

Proof [inverse DFT, based on $b(q,N)$]

$$\begin{aligned}
\frac{b(-n,N)'}{N}\bar{x} &= \frac{b(-n,N)'}{N}\begin{bmatrix} b(0,N)' \\ b(1,N)' \\ \dots \\ b(N-1,N)' \end{bmatrix} x \\
&= \frac{b(-n,N)'}{N}\begin{bmatrix} e^{-j\frac{2\pi}{N}0\cdot 0} & e^{-j\frac{2\pi}{N}0\cdot 1} & \dots & e^{-j\frac{2\pi}{N}0\cdot(N-1)} \\ e^{-j\frac{2\pi}{N}1\cdot 0} & e^{-j\frac{2\pi}{N}1\cdot 1} & \dots & e^{-j\frac{2\pi}{N}1\cdot(N-1)} \\ \dots & & & \\ e^{-j\frac{2\pi}{N}(N-1)\cdot 0} & e^{-j\frac{2\pi}{N}(N-1)\cdot 1} & \dots & e^{-j\frac{2\pi}{N}(N-1)\cdot(N-1)} \end{bmatrix} x \\
&= \frac{b(-n,N)'}{N}\begin{bmatrix} b(0,N) & b(1,N) & \dots & b(N-1,N) \end{bmatrix} x \\
&= [\underbrace{0 \;\dots\; 0}_{n-1} \;\; 1 \;\; \underbrace{0 \;\dots\; 0}_{N-n}]\, x = x_n
\end{aligned}$$

since

$$b(-n,N)'b(p,N) = \sum_{k=0}^{N-1} e^{-j\frac{2\pi}{N}(p-n)k} \left\{ \begin{array}{ll} 0 & p \neq n \\ N & p = n \end{array} \right.$$

Properties of the DFT

DFT: $x \xrightarrow{DFT} \bar{x}$

- Periodicity: (slide 7)

$$\bar{x}_p = \bar{x}_{p+N}$$

- Linearity: (slide 12)

$$DFT(x+y) = DFT(x) + DFT(y)$$

- Symmetry: if x is a real sequence then $\bar{x}_p = \bar{x}^*_{-p} = \bar{x}^*_{N-p}$

$$\bar{x}^*_{-p} = \left(\sum_{k=0}^{N-1} x_k e^{j\frac{2\pi}{N}pk} \right)^* = \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk} = \bar{x}_p$$

inverse DFT: $\bar{x} \xrightarrow{iDFT} x$

- the same, DFT and iDFT are similar operations (slide 15)

DFT and DTFT - frequency comparison

Discrete time Fourier transform:

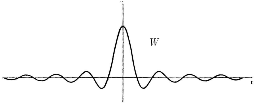
$$\bar{x}_\omega := \mathrm{DTFT}(x) = \sum_{k=0}^{\infty} x_k e^{-j\omega Tk}$$

Discrete Fourier transform:

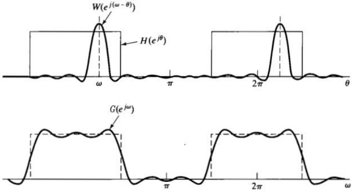
$$\bar{x}_p := \sum_{k=0}^{N-1} x_k e^{-j\frac{2\pi}{N}pk}$$

- Finite sampling $\omega = \frac{2\pi}{NT}$. No loss: we can always sample at the frequency of interest.
- Finite horizon N. Frequency distortion: rectangular window, convolution in frequency (Lecture 8). Improves for $N \to \infty$.

Rectangular window



towards impulse as $N \to \infty$



Circular convolution

Signal: $\{x_k\} \xrightarrow{DFT} \{\bar{x}_p\}$ FIR filter: $\{g_k\} \xrightarrow{DFT} \{\bar{g}_p\}$

The inverse DFT of the **product** of the DFTs

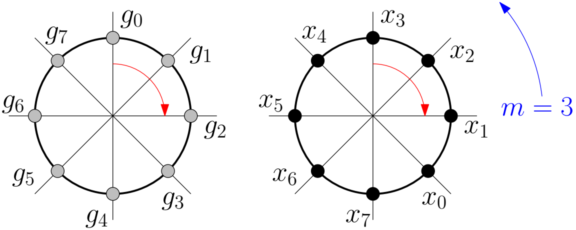
$$\bar{y}_p := \bar{g}_p \bar{x}_p \,, \qquad \{\bar{y}_p\} \xrightarrow{iDFT} \{y_m\}$$

is the **circular convolution** of x and g

$$y_m := \sum_{k=0}^{N-1} g_k x_{mod(m-k,N)}$$

where $mod(k-n, N)$ denotes $k-n$ in modulo N arithmetic

$$\bar{y}_p = \bar{g}_p \bar{x}_p \xrightarrow{iDFT} y_m = \sum_{k=0}^{N-1} g_k x_{mod(m-k,N)}$$



<u>Proof</u>: compute iDFT of $\{\bar{y}_p\} = \{\bar{g}_p \bar{x}_p\}$

$$y_m = \frac{1}{N} \sum_{p=0}^{N-1} \bar{g}_p \bar{x}_p e^{j\frac{2\pi}{N}pm} \quad (iDFT)$$

$$= \frac{1}{N} \sum_{p=0}^{N-1} \left(\sum_{k_1=0}^{N-1} g_{k_1} e^{-j\frac{2\pi}{N}pk_1} \right) \left(\sum_{k_2=0}^{N-1} x_{k_2} e^{-j\frac{2\pi}{N}pk_2} \right) e^{j\frac{2\pi}{N}pm}$$

$$= \frac{1}{N} \sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} g_{k_1} x_{k_2} \underbrace{\sum_{p=0}^{N-1} e^{-j\frac{2\pi}{N}p(k_1+k_2-m)}}_{\substack{N \text{ if } mod(k_1+k_2-m,N)=0 \\ 0 \text{ otherwise}}}$$

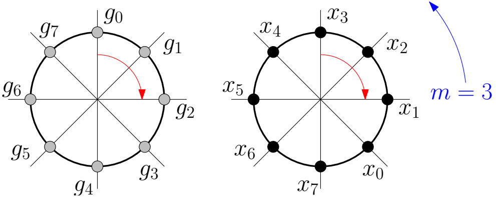
$\rightarrow$ take $k_2 = m - k_1$

$$= \frac{N}{N} \sum_{k_1=0}^{N-1} g_{k_1} x_{mod(m-k_1,N)} = \sum_{k=0}^{N-1} g_k x_{mod(m-k,N)}$$

Filter response via DFT

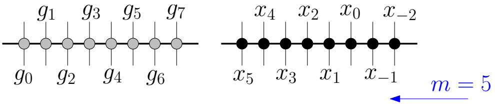
Circular convolution

$$\bar{y}_p = \bar{g}_p \bar{x}_p \xrightarrow{iDFT} y_m = \sum_{k=0}^{N-1} g_k x_{mod(m-k,N)}$$



Filter response (standard/linear convolution)

$$y_m = \sum_{k=0}^{\infty} g_k x_{m-k}$$



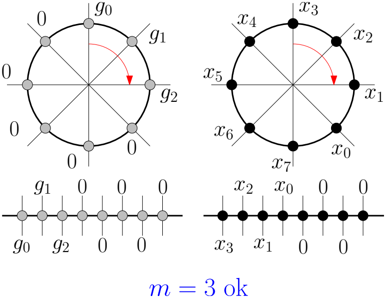
$\{g_k\}$ FIR filter with $M + 1 \ll N$ nonzero coefficients

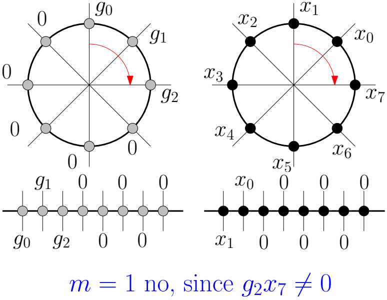
If $M \leq m < N$,

$$\sum_{k=0}^{\infty} g_k x_{m-k} = \sum_{k=0}^{M} g_k x_{m-k} = \sum_{k=0}^{N-1} g_k x_{mod(m-k,N)}$$

standard convolution = circular convolution

Example: $M = 2,\ N = 8$





$\{g_k\}$ FIR filter with $M+1 \ll N$ nonzero coefficients

If $M \le m < N$,

$$\sum_{k=0}^{\infty} g_k x_{m-k} = \sum_{k=0}^{M} g_k x_{m-k} = \sum_{k=0}^{N-1} g_k x_{mod(m-k,N)}$$

standard convolution = circular convolution

We can use DFT for FIR filtering!

$$\begin{tikzcd} x \quad g \arrow[r, bend left, "\text{standard convolution}"] \arrow[r, bend right, "\text{circular convolution (+ batch operation)}"'] \arrow[d, "\text{DFT}"] & y \\ \bar{x} \quad \bar{g} \arrow[r, "\text{product}"] & \bar{y} \arrow[u, "\text{iDFT}"'] \end{tikzcd}$$

$$
\begin{tikzcd}
x \quad g \arrow[rr, bend left, "\text{standard convolution}"] \arrow[rr, bend right, "\text{circular convolution (+ batch operation)}"'] \arrow[d, "\text{DFT}"] & & y \\
\bar{x} \quad \bar{g} \arrow[rr, "\text{product}"] & & \bar{y} \arrow[u, "\text{iDFT}"']
\end{tikzcd}
$$

with fast algorithms DFT-product-iDFT may be more efficient than circular convolution to compute the filter response: FFT!

Example: computation of the filter response via DFT

$M = 2$ (FIR horizon) and $N = 8$ (DFT horizon) [on Moodle]

- filter: $[g_0\ g_1\ g_2\ 0\ 0\ 0\ 0\ 0] \xrightarrow{DFT} \bar{g}$
- frame 1: $[0\ 0\ x_0\ x_1\ x_2\ x_3\ x_4\ x_5] \xrightarrow{DFT} \bar{x}^1$
- frame 2: $[x_4\ x_5\ x_6\ x_7\ x_8\ x_9\ x_{10}\ x_{11}] \xrightarrow{DFT} \bar{x}^2$
- frame q : $[\ \underbrace{*\ *}_{M \text{ from previous frame}}\ |\ \underbrace{*\ *\ *\ *\ *\ *}_{N-M \text{ new points}}\ ]$

$$\xrightarrow{product, iDFT} y^q = [\ \underbrace{*\ *}_{\text{remove}}\ |\ \underbrace{*\ *\ *\ *\ *\ *}_{collect}]$$