NTNU
Norwegian University of Science and Technology

TMA4130 Matematikk 4N

Week 39, second lecture: The discrete Fourier transform

Douglas R. Q. Pacheco

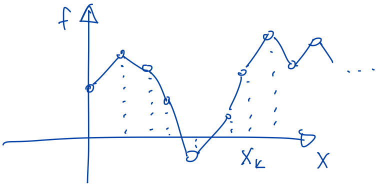
Department of Mathematical Sciences, NTNU.

Autumn semester, 2022

Fourier series

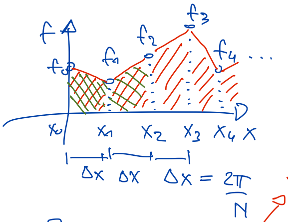
- Decompose a function into "simpler" ones (spectrum)

- In practice, we usually have discrete signals (finite number of sampling points)



The discrete Fourier transform (DFT)

– We have $f(x)$, 2π-periodic, sampled at N equally spaced points x_k, $k = 0, 1, 2, \dots, N-1$: $\boxed{x_k = k\Delta x = k\frac{2\pi}{N}}$



– Fourier coefficients: $c_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)\, e^{-inx}\, dx =$

$$= \frac{1}{2\pi}\int_{0}^{2\pi} f(x)\, e^{-inx}\, dx$$

integrand

– Trapezoidal rule: $c_n \approx \frac{1}{2\pi}\sum_{k=0}^{N-1} \Delta x \left[\frac{f_k e^{-inx_k} + f_{k+1} e^{-inx_{k+1}}}{2}\right] = \frac{1}{\cancel{2\pi}}\,\frac{\cancel{2\pi}}{N}\sum_{k=0}^{N-1}\frac{\cancel{2} f_k e^{-inx_k}}{\cancel{2}}$

$2\pi/N$

repetition of terms

$$\Rightarrow c_n \approx \frac{1}{N}\sum_{k=0}^{N-1} f_k\, e^{-inx_k} = \frac{1}{N}\sum_{k=0}^{N-1} f_k\, e^{-\frac{2\pi k n}{N} i}$$

$f_k = f(x_k)$

The discrete Fourier transform (DFT)

- For computational reasons, we consider the scaled version of the coeffs.:

$$\hat{f}_n = N c_n = \sum_{k=0}^{N-1} f_k e^{-\frac{2\pi k n}{N} i}, \quad n = 0, 1, \dots, N-1$$

↳ the $N \times 1$ vector $\underline{\hat{f}}$ is called the discrete Fourier transform (DFT) of $\underline{f} = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{pmatrix}$

$$\underline{\hat{f}} = \begin{pmatrix} \hat{f}_0 \\ \hat{f}_1 \\ \vdots \\ \hat{f}_{N-1} \end{pmatrix}$$

The Fourier matrix

NTNU

$$\rightarrow \hat{f}_n = \sum_{k=0}^{N-1} f_k e^{\frac{-2\pi k n}{N} i} \; ; \quad \text{let } w = e^{\frac{-2\pi}{N} i} \Rightarrow \boxed{\hat{f}_n = \sum_{k=0}^{N-1} f_k w^{kn}}$$

Let's see what happens for different values of n

$$* \; \hat{f}_0 = \sum_{k=0}^{N-1} f_k w^{0 \cdot k} = f_0 + f_1 + f_2 + \dots + f_{N-1}$$

$$* \; \hat{f}_1 = \sum_{k=0}^{N-1} f_k w^{1 \cdot k} = f_0 + f_1 w + f_2 w^2 + f_3 w^3 + \dots + f_{N-1} w^{N-1}$$

$$* \; \hat{f}_2 = \sum_{k=0}^{N-1} f_k w^{2 \cdot k} = f_0 + f_1 w^2 + f_2 w^4 + f_3 w^6 + \dots + f_{N-1} w^{2(N-1)}$$

$$\vdots$$

$$* \; \hat{f}_{N-1} = \sum_{k=0}^{N-1} f_k w^{k(N-1)} = \dots$$

Fourier matrix (square, $N \times N$)

$$\rightarrow \underline{\hat{f}} = \underline{\underline{F}}_N \, \underline{f}, \quad \begin{pmatrix} \hat{f}_0 \\ \hat{f}_1 \\ \hat{f}_2 \\ \vdots \\ \hat{f}_{N-1} \end{pmatrix} = \underbrace{\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & w & w^2 & \dots & w^{N-1} \\ 1 & w^2 & w^4 & \dots & w^{2(N-1)} \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & w^{N-1} & w^{2(N-1)} & \dots & w^{(N-1)^2} \end{bmatrix}}_{\text{Fourier matrix (square, } N\times N)} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{pmatrix}$$

$$\Rightarrow (F_N)_{nk} = w^{nk}, \quad \text{where } w = e^{\frac{-2\pi}{N} i}$$

n-th row

k-th column

The Fourier matrix

– The Fourier matrix is symmetric and orthogonal

$$\underline{\underline{F_N}}^{-1} = \frac{1}{N}\, \overline{\underline{\underline{F_N}}}$$

→ complex conjugate

The inverse transform

DFT : $\underline{\hat{f}}_N = \underline{\underline{\tilde{F}}}_N \, \underline{f}_N$

- If we want to transform back from $\underline{\hat{f}}$ to $\underline{f}$, we need to do:

$$\underline{f} = \underline{\underline{\tilde{F}}}_N^{-1} \, \underline{\hat{f}} = \frac{1}{N} \, \overline{\underline{\underline{\tilde{F}}}_N} \, \underline{\hat{f}}$$

inverse transform

NTNU

Example : $\underline{f} = \begin{pmatrix} 2 \\ 0 \\ -1 \\ 4 \end{pmatrix}$ (signal) $\Rightarrow N = 4 \Rightarrow W = e^{-\frac{2\pi}{N}i} = e^{-\frac{\pi}{2}i} = \cos\left(\frac{\pi}{2}\right) - i\sin\left(\frac{\pi}{2}\right)$

$= -i$

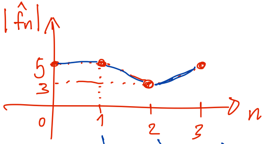
$\boxed{f \text{ is } 2\pi\text{-periodic} : L = \pi}$

Fourier matrix : $(\mathcal{F}_N)_{nk} = W^{nk} = (-i)^{nk}$

$$\Rightarrow \underline{\underline{\mathcal{F}_N}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W & W^2 & W^3 \\ 1 & W^2 & W^4 & W^6 \\ 1 & W^3 & W^6 & W^9 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix}$$

$\boxed{\omega_n = \frac{n\pi}{L} = n}$

$$\Rightarrow \underline{\hat{f}} = \underline{\underline{\mathcal{F}_N}}\,\underline{f} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix} \begin{pmatrix} 2 \\ 0 \\ -1 \\ 4 \end{pmatrix} = \begin{pmatrix} 5 \\ 3+4i \\ -3 \\ 3-4i \end{pmatrix} \begin{array}{l} \to n=0 \\ \to n=1 \\ \to n=2 \\ \to n=3 \end{array}$$



$\omega_1 = 1$ $\omega_2 = 2$ $\omega_3 = 3$

$|3-4i| = \sqrt{3^2 + (-4)^2} = 5$

Example

→ The inverse matrix $\underline{\underline{\tilde{F}_N}}^{-1}$ is simply $\frac{1}{N}\overline{\underline{\underline{\tilde{F}_N}}}$

$$= \frac{1}{4}\overline{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & +1 & -1 \\ 1 & i & -1 & -i \end{bmatrix}} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & +i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & +i \end{bmatrix} \quad //$$

Remember:

if $z = a+bi$, then

$\bar{z} = a - bi$

(complex conjugate)

The fast Fourier transform (FFT)

$\hat{\underline{f}} = \underline{\underline{F}}_N \underline{f}$ (N-dimensional signal)

↳ matrix-vector product "costs" around $\Theta(N^2)$ operations

* Cooley and Tukey (1965): algorithm that reduces the cost to $\Theta(N \log N)$

* Gauss (1805, unpublished)

Ex.: $N = 10^3$ ⇒ * standard DFT: $\Theta((10^3)^2) = \Theta(10^6)$ operations

* Fast one (FFT): $\Theta(10^3 \log 10^3) = \Theta(3000)$ operations

↳ real-time computations (e.g., filtering)

Application (credits: Ronny Bergmann)

172 kB (100 %)

Application (credits: Ronny Bergmann)

29.9 kB (17.3 %)

Application (credits: Ronny Bergmann)

18.9 kB (11 %)

Application (credits: Ronny Bergmann)

10 kB (5.8 %)

Application (credits: Ronny Bergmann)

5.59 kB (3.25 %)

Application (credits: Ronny Bergmann)

3 kB (1.74 %)