Context For Discrete Fourier Transform

In MAT 219 we used continuous Fourier transform to write functions as Fourier series

Function

$f(t)$

- function with period 2π

-or-

- function defined on $[-\pi, \pi]$

$\xrightarrow{\mathcal{F}}$

Coefficients

$\begin{Bmatrix} a_0, a_1, a_2, a_3, \ldots \\ b_1, b_2, b_3, \ldots \end{Bmatrix}$

$\begin{Bmatrix} a_n \\ b_n \end{Bmatrix}$ "frequency n coefficients"

$\xrightarrow{\mathcal{F}^{-1}}$

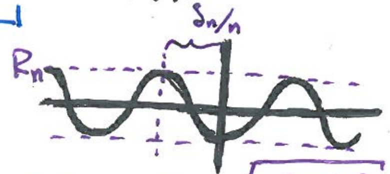
Function (Fourier Series)

Fourier Series Thm

$$a_0/2 + \underbrace{\sum_{n=1}^{\infty} a_n \cos(nt) + b_n \sin(nt)}_{\text{frequency } n} = f(t).$$

$$c_0 + \sum_{n=1}^{\infty} R_n \cos(nt + \delta_n)$$

shifted cosine



$R_n = \sqrt{a_n^2 + b_n^2}$

$\delta_n = \arctan(b_n/a_n)$

$\mathcal{F}$ = Fourier Transform (converts function to coefficients)

$\mathcal{F}^{-1}$ = Inverse Fourier Transform (converts coefficients to function)

It is more efficient to combine $\begin{Bmatrix} a_n \\ b_n \end{Bmatrix}$ into a single coefficient c_n — this gives $\mathbb{C}$ Fourier

$\mathbb{C}$ Fourier Series Thm

$$f(t) \xrightarrow{\mathcal{F}} \{ c_0, c_1, c_2, c_3, \ldots \} \xrightarrow{\mathcal{F}^{-1}} \underbrace{c_0 + \sum_{n=1}^{\infty} c_n e^{int} + \overline{c_n e^{int}}}_{=\ \sum_{n=-\infty}^{\infty} c_n e^{int}} = f(t)$$

conjugate $= \overline{c_n}\, e^{-int}$

where $c_{-n} = \overline{c_n}$

- $c_n = \dfrac{1}{2\pi} \displaystyle\int_{-\pi}^{\pi} f(t) e^{-int}\, dt$

$= \dfrac{a_n - b_n i}{2}$

$= \frac{1}{2} \cdot R_n e^{\delta_n i}$

Note: frequency n is $\left(c_n e^{int} + c_{-n} e^{-int}\right)$

Amplitude: $R_n = 2\,|c_n|$

Now we discretize

Function		Coefficients		Function
$\underline{f} = [f_0, f_1, f_2, \ldots, f_{N-1}]$	$\xrightarrow{\mathcal{F}}$	$\underline{c} = [c_0, c_1, c_2, \ldots, c_{N-1}]$	$\xrightarrow{\mathcal{F}^{-1}}$	$[f_0, f_1, f_2, \ldots, f_{N-1}]$

↑ vector of data, length N

"periodic" function:

$$f_{N-2}, f_{N-1}, f_0, f_1, f_2, \ldots, f_{N-1}, f_0, f_1$$

$$f_{-2} \quad f_{-1} \qquad\qquad f_N \quad f_{N+1}$$

- $c_k = \frac{1}{N} \sum_{n=0}^{N-1} f_n \, \overline{\omega}_N^{\,k \cdot n}$

  $\hookrightarrow c_{N-k} = c_{-k} = \overline{c_k}$

- $\underbrace{f_n = \sum_{k=0}^{N-1} c_k \, \omega_N^{\,n \cdot k}}$

combine f_n formulas to get $\underline{f}$ formula

"Discrete Fourier Series" would look like

$$\begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{bmatrix} = c_0 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + c_1 \begin{bmatrix} 1 \\ \omega_N \\ \vdots \\ \omega_N^{N-1} \end{bmatrix} + c_2 \begin{bmatrix} 1 \\ \omega_N^2 \\ \vdots \\ \omega_N^{2(N-1)} \end{bmatrix} + c_3 \begin{bmatrix} 1 \\ \omega_N^3 \\ \vdots \\ \omega_N^{3(N-1)} \end{bmatrix} + \cdots$$

$\mathbb{C}$ Fourier Series for f

$$= \sum_{k=0}^{N-1} c_k \begin{bmatrix} 1 \\ \omega_N^k \\ \omega_N^{2k} \\ \vdots \\ \omega_N^{(N-1)k} \end{bmatrix} \leftarrow \text{discrete } e^{kti}$$

$\mathbb{R}$ Fourier Series for f

$$= a_0/2 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + \sum_{k=1}^{N-1} a_k \begin{bmatrix} 1 \\ \cos(k \cdot 2\pi/N) \\ \cos(2k \cdot 2\pi/N) \\ \vdots \\ \cos((N-1)k \cdot 2\pi/N) \end{bmatrix} + b_k \begin{bmatrix} 0 \\ \sin(k \cdot 2\pi/N) \\ \sin(2k \cdot 2\pi/N) \\ \vdots \\ \sin((N-1)k \cdot 2\pi/N) \end{bmatrix}$$

discrete $\cos(kt)$ discrete $\sin(kt)$

$\left(\text{where } c_k = \frac{a_k - b_k i}{2}\right)$

<u>BUT</u> we don't really care about the Fourier series <u>formulas</u> in the discrete case...

The usefulness of discrete Fourier is <u>not</u> the Fourier series, but instead the

<u>Discrete Fourier Transform</u> <u>DFT</u> $\mathcal{F}\{\cdot\}$

and the inverse

<u>Inverse Discrete Fourier</u> <u>IFT</u> $\mathcal{F}^{-1}\{\cdot\}$

→ Use this like <u>Laplace transform</u> in Diff. Eq.

- Many problems which are difficult to solve become simpler if you take the DFT changing $\underline{f}$ to $\underline{c}$.

- $\underline{f}$ can be <u>any</u> vector of data
  - often when working through the theory in class, we imagine that $\underline{f}$ is a recording of sound, but DFT is useful in many other settings
  - one application involves taking large numbers like 2,342,176 and considering them as vectors of digits like $f = [2\ 3\ 4\ 2\ 1\ 7\ 6]$

The main tools that make Fourier transforms useful in discrete case are

FFT - Fast Fourier Transform algorithm

$*$ - Convolutions

→ Why switch $\underline{f} \rightsquigarrow \mathcal{F}\{f\} = c$?

- Writing a function as $f = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{bmatrix}$ uses "impulse basis"

$$\begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{bmatrix} = f_0 \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + f_1 \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \cdots + f_{N-1} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

- Writing a function as $\mathcal{F}\{f\} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix}$ uses "ω_N-basis"

$$\begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{bmatrix} = c_0 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + c_1 \begin{bmatrix} 1 \\ \omega_N \\ \omega_N^2 \\ \vdots \end{bmatrix} + c_2 \begin{bmatrix} 1 \\ \omega_N^2 \\ \omega_N^4 \\ \vdots \end{bmatrix} + \cdots + c_{N-1} \begin{bmatrix} 1 \\ \omega_N^{N-1} \\ \omega_N^{2(N-1)} \\ \vdots \end{bmatrix}$$

- Because of the nice cyclic symmetry of the ω_N basis, it ends up being a very good presentation for solving many problems.