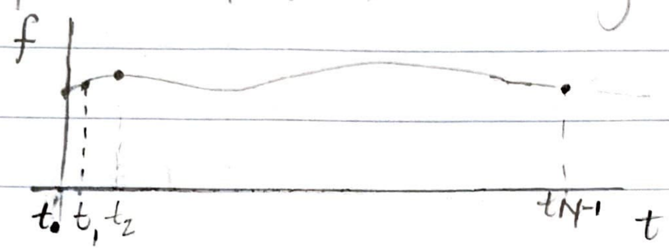
DISCRETE FOURIER TRANSFORM

Typically, we have our function (in-time) defined by a set of data points, not analytically



The smallest freq. we can detect as a basis for this set of points that constitute our discrete function (fundamental frequency) is $1/N$, or angular freq. $2\pi/N$

So we can now write the discrete fourier transform (DFT) as

$$\hat{f}_k = \sum_{j=0}^{N-1} f_j e^{-ik\frac{2\pi}{N}j},$$

and the inverse discrete fourier transform (iDFT)

$$f_j = \frac{1}{N}\sum_{k=0}^{N-1} \hat{f}_k e^{ij\frac{2\pi}{N}k}.$$

NOTE: We could have written $\frac{1}{\sqrt{N}}$ before the DFT & iDFT

Now, we can express $\{\hat{f}_k\}$ for $k=0,\dots,n-1$ as a linear operator (matrix) acting on $\{f_j\}$ for $j=0,\dots,n-1$ by denoting

$e^{-i\frac{2\pi}{N}}$ by ω_N

$$\begin{array}{cc} k \backslash j & \\ 0 & \\ 1 & \\ 2 & \\ \vdots & \\ N-1 & \end{array} \begin{bmatrix} \hat{f}_0 \\ \hat{f}_1 \\ \hat{f}_2 \\ \vdots \\ \hat{f}_{N-1} \end{bmatrix} = \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & & N-1 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_N^1 & \omega_N^2 & & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \cdots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \cdots & \omega_N^{(N-1)^2} \end{bmatrix} \end{array} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{bmatrix}$$

It should be obvious that this DFT matrix F is symmetric; in fact, it is unitary (see below). In addition, it is a Vandermonde matrix, i.e

$$F_{i,j} = (\omega_N^i)^j \qquad i,j = 0, \dots N-1$$

NOTE: Actually $F^{-1} = \frac{1}{N} F^H$.

However if we had defined the DFT with the normalization $1/\sqrt{N}$, i.e

$$\hat{f}_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} f_j e^{-ik\frac{2\pi}{N}j} \quad , \quad f_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} f_k e^{ij\frac{2\pi}{N}k}$$

we would have had a unitary matrix $F_{\sqrt{N}}$ for which $F_{\sqrt{N}}^{-1} = F_{\sqrt{N}}^H$

where $F_{\sqrt{N}} = \frac{1}{\sqrt{N}} F$.

Fast Fourier Transform

The DFT requires $\mathcal{O}(N^2)$ operations. But given the symmetry of the DFT Matrix, it is interesting to note that

$$\hat{f}_{k+N} = \sum_{j=0}^{N-1} f_j e^{-i(k+N)\frac{2\pi}{N}j}$$

$$= \sum f_j e^{-ik\frac{2\pi}{N}j} = \hat{f}_k .$$

In fact,

$$\hat{f}_{k+nN} = \hat{f}_k \text{ for any integer } n$$

so

$$\hat{f}_k = \sum_{j=0}^{N/2-1} f_{2j} e^{-ik\frac{2\pi}{N}2j} + \sum_{j=0}^{N/2-1} f_{2j+1} e^{-ik\frac{2\pi}{N}(2j+1)}$$

$$= \sum_{j=0}^{N/2-1} \underbrace{f_{2j}}_{\text{even}} e^{-ik\frac{2\pi}{N/2}j} + e^{-ik2\pi/N} \sum_{j=0}^{N/2-1} \underbrace{f_{2j+1}}_{\text{odd}} e^{-ik\frac{2\pi}{N}2j}$$

$$\begin{bmatrix} f_k \\ f_{\frac{N}{2}+k} \end{bmatrix} = \begin{bmatrix} \sum_{j=0}^{N/2-1} f_{2j} e^{-ik\frac{2\pi}{N/2}j} + e^{-ik2\pi/N} \sum_{j=0}^{N/2-1} f_{2j+1} e^{-ik2\pi/(N/2)j} \\ \sum_{j=0}^{N/2-1} f_{2j} e^{-ik\frac{2\pi}{N/2}j} - e^{-ik\frac{2\pi}{N}} \sum f_{2j+1} e^{-ik2\pi/(N/2)j} \end{bmatrix}$$

this is just another FT for which the routine can be repeated

Only need to evaluate this

In terms of matrices, we can express one step of the 'divide-and-conquer' notion as:

$$\begin{bmatrix} \hat{f}_k\text{'s} \end{bmatrix} = \begin{bmatrix} I_{N/2} & D_{N/2} \\ I_{N/2} & -D_{N/2} \end{bmatrix} \begin{bmatrix} F_{N/2} & 0 \\ 0 & F_{N/2} \end{bmatrix} \underbrace{\begin{bmatrix} P_1 \\ P_2 \end{bmatrix}}_{\text{Permutation matrices}} \begin{bmatrix} f_j\text{'s} \end{bmatrix}$$

where:

$$D = \begin{bmatrix} \left(e^{-i2\pi/N}\right)^0 & & \\ & \ddots & \\ & & \left(e^{-i\frac{2\pi}{N}}\right)^{N/2-1} \end{bmatrix}$$

Permutation matrices to seperate $\begin{bmatrix} f_{even} \\ f_{odd} \end{bmatrix}$

& $I_{N/2}$ is the identity matrix.

The partitioning of the Fourier transform can be repeated recursively.

By finding the Fourier transform in this manner, we can cut the number of arithmetic operations from $O(N^2)$ to $O(N \log N)$, which makes the Fourier transformation for very large N feasible computationally.