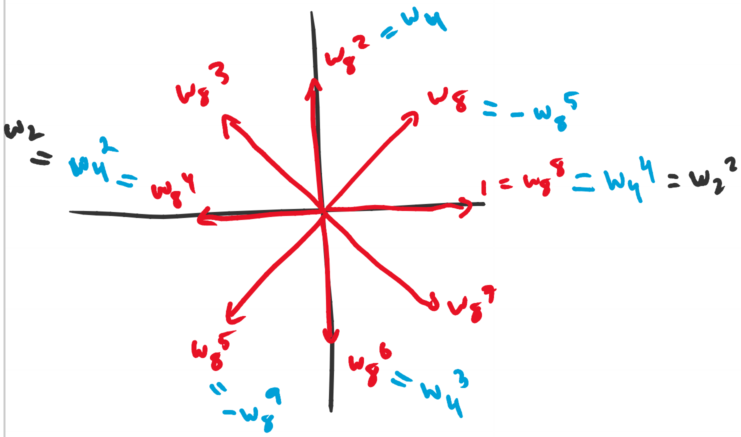
Lecture 35: Fast Fourier transform (9.3) (FFT)

Fast (Discrete) Fourier Transform (DFT): An algorithm for quickly computing change of basis to (& from) Fourier basis.

Let $w_n = e^{2\pi i/n}$



Key facts: If n even

(i) $w_n^k = -w_n^{k+n/2}$

$(w_n^{n/2} = -1)$

(ii) $w_n^2 = w_{n/2}$

$$F_n = \underbrace{\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & w & w^2 & & w^{n-1} \\ 1 & w^2 & w^4 & & w^{2(n-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & w^{n-1} & w^{2(n-1)} & \cdots & w^{(n-1)^2} \end{pmatrix}}_{\text{orthogonal basis for } \mathbb{C}^n}, \quad F_n^{-1} = \frac{1}{n} F_n^H$$

naive matrix multiplication: n^2 multiplications

$$\vec{x} \xrightarrow{\text{change of basis to Fourier basis}} \underbrace{\frac{1}{n} F_n^H \vec{x}}_{\vec{y}} \xrightarrow{\text{change back}} F_n \vec{y} = \hat{x}$$

How to compute $F_4\vec{x}$ quickly:

$$F_4\vec{x} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & w_4 & w_4^2 & w_4^3 \\ 1 & w_4^2 & w_4^4 & w_4^6 \\ 1 & w_4^3 & w_4^6 & w_4^9 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

$$= \begin{pmatrix} x_0 + x_1 + x_2 + x_3 \\ x_0 + w_4 x_1 + w_4^2 x_2 + w_4^3 x_3 \\ x_0 + w_4^2 x_1 + w_4^4 x_2 + w_4^6 x_3 \\ x_0 + w_4^3 x_1 + w_4^6 x_2 + w_4^9 x_3 \end{pmatrix}$$

$$\begin{array}{ccc} \text{even} & & \text{odd} \end{array}$$

$$= \begin{pmatrix} x_0 + x_2 & + & x_1 + x_3 \\ x_0 + \underbrace{w_4^2}_{=w_2} x_2 & + & w_4 (x_1 + \underbrace{w_4^2}_{=w_2} x_3) \\ x_0 + \underbrace{w_4^4}_{=w_2^2} x_2 & + & \underbrace{w_4^2}_{=-1} (x_1 + \underbrace{w_4^4}_{=w_2^2} x_3) \\ x_0 + \underbrace{w_4^6}_{=w_2^3} x_2 & + & \underbrace{w_4^3}_{=-w_4} (x_1 + \underbrace{w_4^6}_{=w_2^3} x_3) \end{pmatrix}$$

(i) & (ii)

$$= \begin{pmatrix} x_0 + x_2 & + & x_1 + x_3 \\ x_0 + w_2 x_2 & + & w_4 (x_1 + w_2 x_3) \\ x_0 + \underbrace{w_2^2}_{=1} x_2 & - & (x_1 + \underbrace{w_2^2}_{=1} x_3) \\ x_0 + \underbrace{w_2^3}_{=w_2} x_2 & - & w_4 (x_1 + \underbrace{w_2^3}_{=w_2} x_3) \end{pmatrix}$$

$$= \begin{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & w_2 \end{pmatrix} \begin{pmatrix} x_0 \\ x_2 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & w_4 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & w_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_3 \end{pmatrix} \\ \begin{pmatrix} 1 & 1 \\ 1 & w_2 \end{pmatrix} \begin{pmatrix} x_0 \\ x_2 \end{pmatrix} + \begin{pmatrix} -1 & 0 \\ 0 & -w_4 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & w_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_3 \end{pmatrix} \end{pmatrix}$$

$$= \left(\begin{array}{cc|cc} 1 & & 1 & \\ & 1 & & w_4 \\ \hline 1 & & -1 & \\ & 1 & & -w_4 \end{array}\right) \left(\begin{array}{cc|cc} 1 & 1 & & \\ 1 & w_2 & & \\ \hline & & 1 & 1 \\ & & 1 & w_2 \end{array}\right) \begin{pmatrix} x_0 \\ x_2 \\ x_1 \\ x_3 \end{pmatrix}$$

$$= \left(\begin{array}{c|c} I_2 & A_2 \\ \hline I_2 & -A_2 \end{array}\right) \left(\begin{array}{c|c} F_2 & \\ \hline & F_2 \end{array}\right) \begin{pmatrix} 1 & & & \\ & & 1 & \\ & 1 & & \\ & & & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

In general,

$$F_n = \left(\begin{array}{c|c} I_{n/2} & A_{n/2} \\ \hline I_{n/2} & -A_{n/2} \end{array}\right) \left(\begin{array}{c|c} F_{n/2} & \\ \hline & F_{n/2} \end{array}\right) P$$

permutation putting evens before odds.

$$\text{where } A_{n/2} = \begin{pmatrix} 1 & & & \\ & w_n & & \\ & & \ddots & \\ & & & w_n^{n/2-1} \end{pmatrix}$$

If $T(n)$ = time to compute $F_n \vec{x}$, then

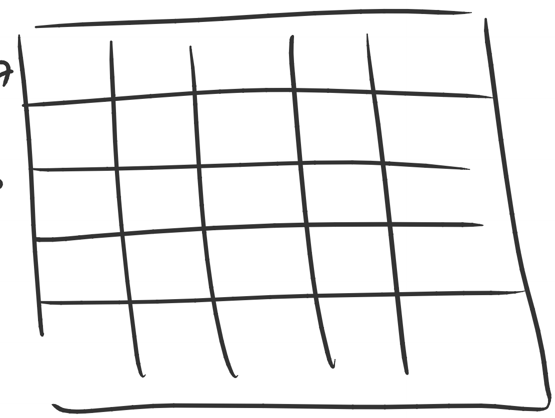
$$T(n) = \underset{\substack{\uparrow \\ F_{n/2}\text{'s}}}{2T(n/2)} + \underset{\substack{\uparrow \\ \text{everything else}}}{\Theta(n)}$$

$$= O(n \log n)$$

$\left(< n^2 \right.$:) $\left. \right)$

JPEG:

8x8 blocks



compress by

DCT

(same idea as DFT)

generally 10:1 compression w/ no noticable effect.