The Fast Fourier Transform (FFT)

Known for centuries, first published in 1965 by Cooley/Tukey

A direct computation of DFT for vector of length N takes N^2 multiplications. FFT takes $O(N \log N)$ (if $N = 2^p$)

$$w = \sqrt[N]{1} = e^{2\pi i / N}, \text{ so } \quad w^N = 1, \quad w^{\frac{N}{2}} = -1$$

DFT:
$$\hat{f}_k = \sum_{m=0}^{N-1} f_m w^{mk} \qquad k = 0..N-1, \quad m = 0..N-1$$

Assume N is even

$$k = k_1 \cdot \frac{N}{2} + k_2 \qquad k_1 = 0, 1, \quad k_2 = 0..N/2-1$$

$$m = m_1 + 2m_2 \qquad m_1 = 0, 1, \quad m_2 = 0, \ldots, \frac{N}{2}-1$$

example: $N = 8$

m, k	0	1	2	3	4	5	6	7
k_1	0	0	0	0	1	1	1	1
k_2	0	1	2	3	0	1	2	3
m_1	0	1	0	1	0	1	0	1
m_2	0	0	1	1	2	2	3	3

$$\hat{f}_{k_1 \cdot \frac{N}{2} + k_2} = \sum_{m_1=0}^{1} \sum_{m_2=0}^{\frac{N}{2}-1} f_{m_1 + 2m_2} \, w^{(m_1 + 2m_2)(k_1 \cdot \frac{N}{2} + k_2)}$$

$$= \underbrace{w^{m_1 k_1 \frac{N}{2}}}_{= (-1)^{m_1 k_1}} \cdot \underbrace{w^{m_2 k_1 N}}_{= 1} \cdot w^{m_1 k_2} \cdot \underbrace{w^{2 m_2 k_2}}_{= (w^2)^{m_2 k_2}}$$

$$\hat{f}_{k_1, \frac{N}{2}+k_2} = \sum_{m_1=0}^{1} (-1)^{m_1 k_1} \omega^{m_1 k_2} \sum_{m_2=0}^{\frac{N}{2}-1} f_{m_1+2m_2} \cdot (\omega^2)^{m_2 k_2}$$

$$= \underbrace{\sum_{m_2=0}^{\frac{N}{2}-1} f_{2m_2} (\omega^2)^{m_2 k_2}}_{m_1=0} + \underbrace{(-1)^{k_1} \omega^{k_2} \sum_{m_2=0}^{\frac{N}{2}-1} f_{2m_2+1} (\omega^2)^{m_2 k_2}}_{m_1=1} \quad (*)$$

$m_1=0$: = DFT of terms with even subscript

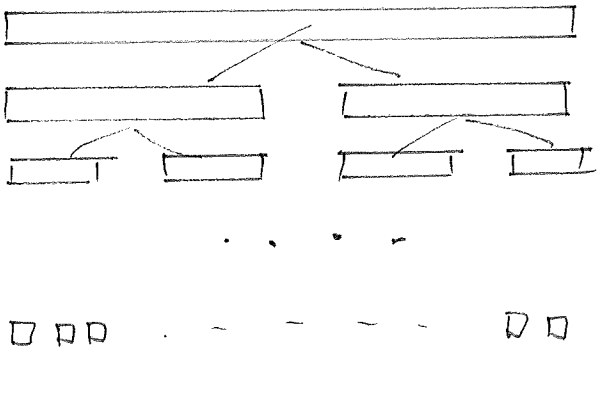
$m_1=1$: = DFT of terms with odd subscript

"twiddle factor"

What we have done is to write the original DFT of length N as a combination of 2 DFTs of length $\frac{N}{2}$, plus $O(N)$ operations to put them together.

Assume $N = 2^p$

repeat this



$p = \log_2 N$ layers

This is where the $O(N \log N)$ operation count comes from.

Combine the two sums in (*) into one, but split the left side into $k_1=0$ and $k_1=1$:

$k_1=0$

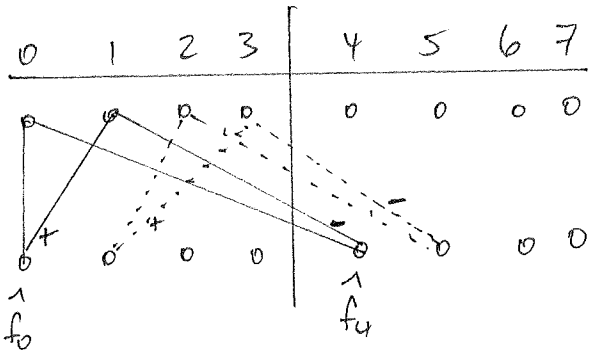
$$\hat{f}_{k_2} = \sum_{m_2=0}^{\frac{N}{2}-1} f_{2m_2} (w^2)^{m_2 k_2} + w^{k_2} \sum_{m_2=0}^{\frac{N}{2}-1} f_{2m_2+1} (w^2)^{m_2 k_2}$$

$$= \sum_{m_2=0}^{\frac{N}{2}-1} \underbrace{[f_{2m_2} + w^{k_2} f_{2m_2+1}]}_{\text{"butterfly operation"}} (w^2)^{m_2 k_2}$$

$k_1=1$

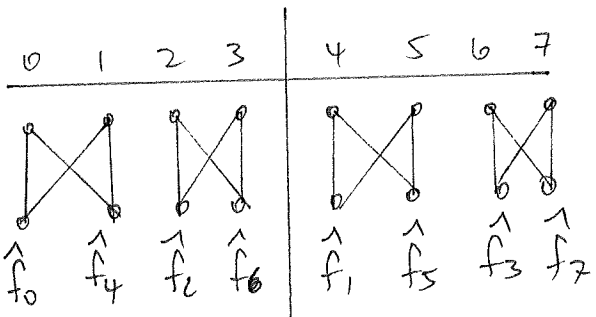
$$\hat{f}_{k_2+\frac{N}{2}} = \sum_{m_2=0}^{\frac{N}{2}-1} [f_{2m_2} - w^{k_2} f_{2m_2+1}] (w^2)^{m_2 k_2}$$

Example first layer for $N=8$



problem f_0, f_1 combine into $\hat{f}_0, \hat{f}_4$ but we can't store $\hat{f}_4$ where f_4 used to be

solution store $\hat{f}_4$ where f_0 used to be instead



What is the order?

answer: bit-reversed subscripts

compare

$N = 8 = 2^3$

	3-bit binary	reverse	
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Generalizations

(1) Instead of dividing by 2, you could use 3, 5, ...
For good results, N should factor into powers of small prime numbers.
If N = prime, there is no speedup.
You may have to pad the data with a few zeros to get to a good N

(2) The DFT originally goes complex → complex.
If your data is real, you can save some effort.
The FT of real signal has real part even, imaginary part odd. You could put half the signal in real part of a shorter vector, the other half in imaginary part, and sort things out by splitting $\vec{f}$ into even/odd parts.
Even better, use real version of DFT, such as the DCT (discrete cosine transform, used in JPEG)

Applications

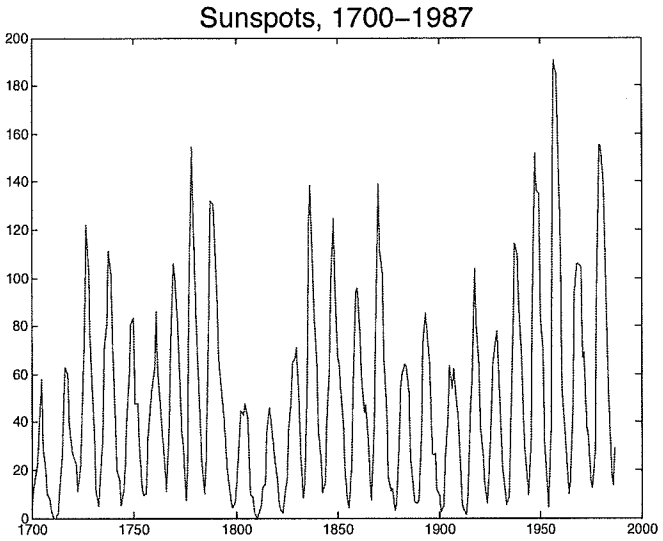
(1) $\vec{f} * \vec{g} \xrightarrow{DFT} \hat{f} \cdot \hat{g}$

convolution → multiplication

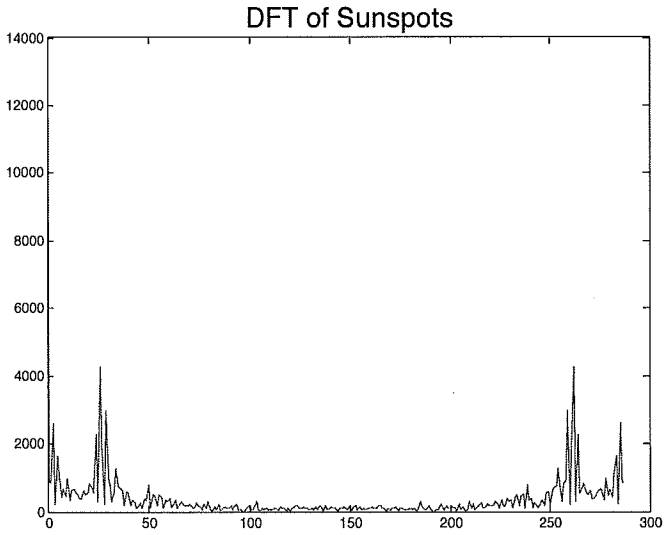
(2)
$$\begin{pmatrix} a_1 & a_2 & \cdots & a_n \\ a_n & a_1 & a_2 & a_{n-1} \\ a_{n-1} & a_n & a_1 & \\ \vdots & & & a_2 \\ a_2 & a_3 & \cdots & a_1 \end{pmatrix}$$
circulant matrix $\xrightarrow{DFT}$
$$\begin{pmatrix} \hat{a}_1 & & & \\ & \hat{a}_2 & & 0 \\ & & \ddots & \\ & 0 & & \hat{a}_n \end{pmatrix}$$
diagonal matrix

Example for applying FFT

- Assume signal is real, length N
- Take FFT. The transform will be complex.
  Real part is even around middle
  Imaginary part is odd
- We usually look at absolute value.
  This is even about middle.
  Only values in first half are relevant
- peak at 0 is the average. You can ignore that
- if there is a big peak at index k
  (starting with index 0, not 1), that indicates
  a component with k periods in the N data points
  $\Rightarrow$ period $\frac{N}{k}$
- Sometimes a strong peak "bleeds over" into
  nearby values. Maybe the actual period does
  not fit neatly into the signal length, or the
  period fluctuates a little.

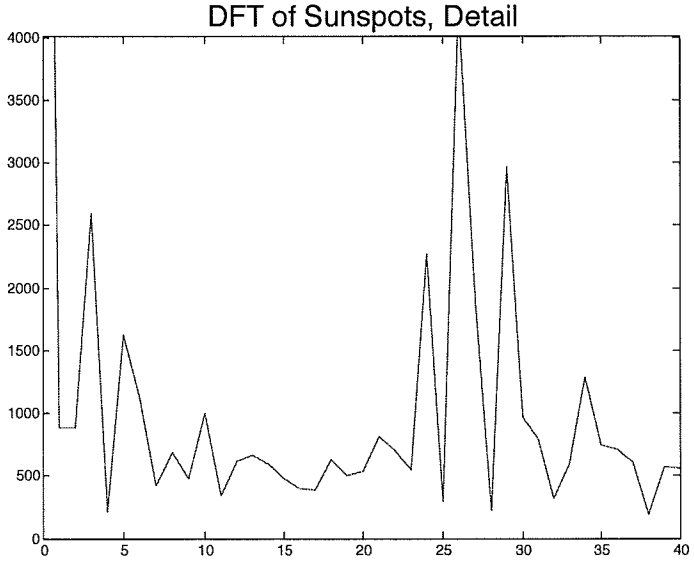


count of sunspots per year, 1700 - 1987



Note: - big peak at 0 (hard to see) does not mean anything

- big peak around 30
- plot is symmetric about middle; only left half is relevant



- peak is at index 26
  signal length is 288
  $\Rightarrow$ peak corresponds to 26 periods in 288 years
  period is $\approx \frac{288}{26} \approx 11$ years
- other peaks at 24, 29 may be caused by same period
- more minor peaks at 3, 5 indicate periodicity around $\frac{288}{4} = 72$ years