The Fast Fourier Transform (FFT)
A top 10 Algorithm*

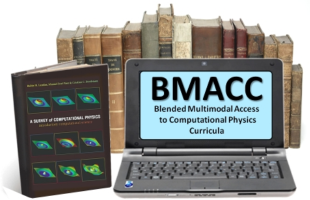
Rubin H Landau

Sally Haerer, Producer-Director

Based on *A Survey of Computational Physics* by Landau, Páez, & Bordeianu

with Support from the National Science Foundation

Course: **Computational Physics II**



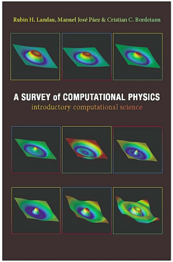
Outline

Last of Three Fourier Units

Unit I: Fourier Series, DFT & Aliases

Unit II: Signal Filtering to Reduce Noise

Unit III: Fast Fourier Transform (FFT) (on request)



Problem: Speed Up the DFT

DFT's One Complex Number Z

$$Y_n = \frac{1}{\sqrt{2\pi}} \sum_k Z^{nk} y_k \qquad \text{(DFT)} \tag{1}$$

$$Z = e^{-2\pi i/N}, \qquad 0 \le n, k \le N-1 \tag{2}$$

- $\sim \; N^2$ complex mults & adds (geometric)
- FFT: 1965, Cooley & Tukey; 1942, Danielson & Lanczos; Gauss
- $e^{-2\pi i/N}$ periodic $\Rightarrow N^2 \;\rightarrow\; N \log_2 N$
- $100 \times$ for $N = 1000$; full day $\rightarrow$ 15 min

FFT: Trig $\Rightarrow$ Economy

- $Z^{nk}[= ((Z)^n)^k]$ = expensive & repetitive (n, k) vary
- *E.g.* $N = 8$, $Z^0(\equiv 1)$ only 4 independent Z's:

$$
\begin{aligned}
Y_0 &= Z^0 y_0 + Z^0 y_1 + Z^0 y_2 + Z^0 y_3 + Z^0 y_4 + Z^0 y_5 + Z^0 y_6 + Z^0 y_7 \\
Y_1 &= Z^0 y_0 + Z^1 y_1 + Z^2 y_2 + Z^3 y_3 + Z^4 y_4 + Z^5 y_5 + Z^6 y_6 + Z^7 y_7 \\
Y_2 &= Z^0 y_0 + Z^2 y_1 + Z^4 y_2 + Z^6 y_3 + Z^8 y_4 + Z^{10} y_5 + Z^{12} y_6 + Z^{14} y_7 \\
Y_3 &= Z^0 y_0 + Z^3 y_1 + Z^6 y_2 + Z^9 y_3 + Z^{12} y_4 + Z^{15} y_5 + Z^{18} y_6 + Z^{21} y_7 \\
Y_4 &= Z^0 y_0 + Z^4 y_1 + Z^8 y_2 + Z^{12} y_3 + Z^{16} y_4 + Z^{20} y_5 + Z^{24} y_6 + Z^{28} y_7 \\
Y_5 &= Z^0 y_0 + Z^5 y_1 + Z^{10} y_2 + Z^{15} y_3 + Z^{20} y_4 + Z^{25} y_5 + Z^{30} y_6 + Z^{35} y_7 \\
Y_6 &= Z^0 y_0 + Z^6 y_1 + Z^{12} y_2 + Z^{18} y_3 + Z^{24} y_4 + Z^{30} y_5 + Z^{36} y_6 + Z^{42} y_7 \\
Y_7 &= Z^0 y_0 + Z^7 y_1 + Z^{14} y_2 + Z^{21} y_3 + Z^{28} y_4 + Z^{35} y_5 + Z^{42} y_6 + Z^{49} y_7
\end{aligned}
$$

The Four Independent Z's

$$
\begin{array}{ll}
Z^0 = \exp(0) = +1, & Z^1 = \exp(-\frac{2\pi}{8} i) = +\frac{\sqrt{2}}{2} - i\,\frac{\sqrt{2}}{2}, \\
Z^2 = \exp(-\frac{2\pi}{8} 2i) = -i, & Z^3 = \exp(-\frac{2\pi}{8} 3i) = -\frac{\sqrt{2}}{2} - i\,\frac{\sqrt{2}}{2}, \\
Z^4 = \exp(-\frac{2\pi}{8} 4i) = -Z^0, & Z^5 = \exp(-\frac{2\pi}{8} 5i) = -Z^1, \\
Z^6 = \exp(-\frac{2\pi}{8} 6i) = -Z^2, & Z^7 = \exp(-\frac{2\pi}{8} 7i) = -Z^3, \\
Z^8 = \exp(-\frac{2\pi}{8} 8i) = +Z^0, & Z^9 = \exp(-\frac{2\pi}{8} 9i) = +Z^1, \\
Z^{10} = \exp(-\frac{2\pi}{8} 10i) = +Z^2, & Z^{11} = \exp(-\frac{2\pi}{8} 11i) = +Z^3, \\
Z^{12} = \exp(-\frac{2\pi}{8} 11i) = -Z^0, & \cdots
\end{array}
$$

Resulting DFT (64 Ops)

$$
\begin{aligned}
Y_0 &= Z^0 y_0 + Z^0 y_1 + Z^0 y_2 + Z^0 y_3 + Z^0 y_4 + Z^0 y_5 + Z^0 y_6 + Z^0 y_7 \\
Y_1 &= Z^0 y_0 + Z^1 y_1 + Z^2 y_2 + Z^3 y_3 - Z^0 y_4 - Z^1 y_5 - Z^2 y_6 - Z^3 y_7 \\
Y_2 &= Z^0 y_0 + Z^2 y_1 - Z^0 y_2 - Z^2 y_3 + Z^0 y_4 + Z^2 y_5 - Z^0 y_6 - Z^2 y_7 \\
Y_3 &= Z^0 y_0 + Z^3 y_1 - Z^2 y_2 + Z^1 y_3 - Z^0 y_4 - Z^3 y_5 + Z^2 y_6 - Z^1 y_7 \\
Y_4 &= Z^0 y_0 - Z^0 y_1 + Z^0 y_2 - Z^0 y_3 + Z^0 y_4 - Z^0 y_5 + Z^0 y_6 - Z^0 y_7 \\
Y_5 &= Z^0 y_0 - Z^1 y_1 + Z^2 y_2 - Z^3 y_3 - Z^0 y_4 + Z^1 y_5 - Z^2 y_6 + Z^3 y_7 \\
Y_6 &= Z^0 y_0 - Z^2 y_1 - Z^0 y_2 + Z^2 y_3 + Z^0 y_4 - Z^2 y_5 - Z^0 y_6 + Z^2 y_7 \\
Y_7 &= Z^0 y_0 - Z^3 y_1 - Z^2 y_2 - Z^1 y_3 - Z^0 y_4 + Z^3 y_5 + Z^2 y_6 + Z^1 y_7 \\
Y_8 &= Y_0
\end{aligned}
$$

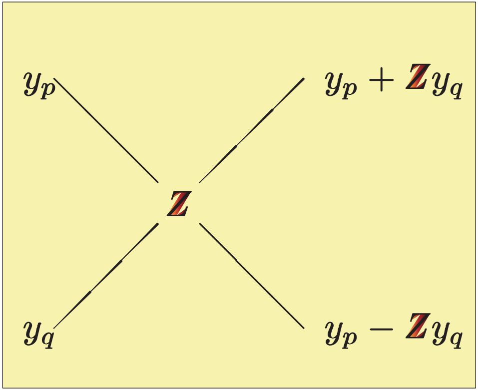
FFT: As Sums & Differences of Measurements

$$
\begin{aligned}
Y_0 &= Z^0(y_0+y_4)+Z^0(y_1+y_5)+Z^0(y_2+y_6)+Z^0(y_3+y_7)\\
Y_1 &= Z^0(y_0-y_4)+Z^1(y_1-y_5)+Z^2(y_2-y_6)+Z^3(y_3-y_7)\\
Y_2 &= Z^0(y_0+y_4)+Z^2(y_1+y_5)-Z^0(y_2+y_6)-Z^2(y_3+y_7)\\
Y_3 &= Z^0(y_0-y_4)+Z^3(y_1-y_5)-Z^2(y_2-y_6)+Z^1(y_3-y_7)\\
Y_4 &= Z^0(y_0+y_4)-Z^0(y_1+y_5)+Z^0(y_2+y_6)-Z^0(y_3+y_7)\\
Y_5 &= Z^0(y_0-y_4)-Z^1(y_1-y_5)+Z^2(y_2-y_6)-Z^3(y_3-y_7)\\
Y_6 &= Z^0(y_0+y_4)-Z^2(y_1+y_5)-Z^0(y_2+y_6)+Z^2(y_3+y_7)\\
Y_7 &= Z^0(y_0-y_4)-Z^3(y_1-y_5)-Z^2(y_2-y_6)-Z^1(y_3-y_7)\\
Y_8 &= Y_0
\end{aligned}
$$

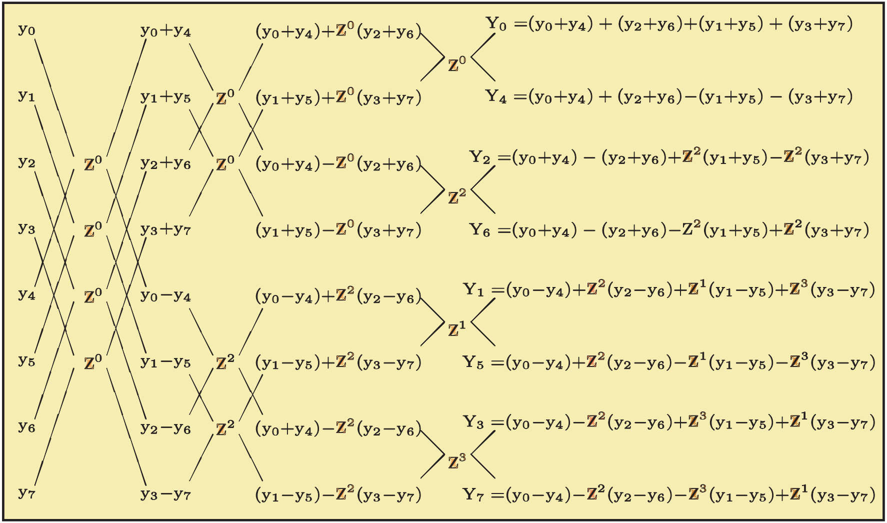
- NB: repeating factors: $(y_p \pm y_q)$ $\Rightarrow$ Butterfly Op

Basic Butterfly Operation

Left Wing → Right Wing



Butterfly Operation on Four Data (64 ops → 24 ops)

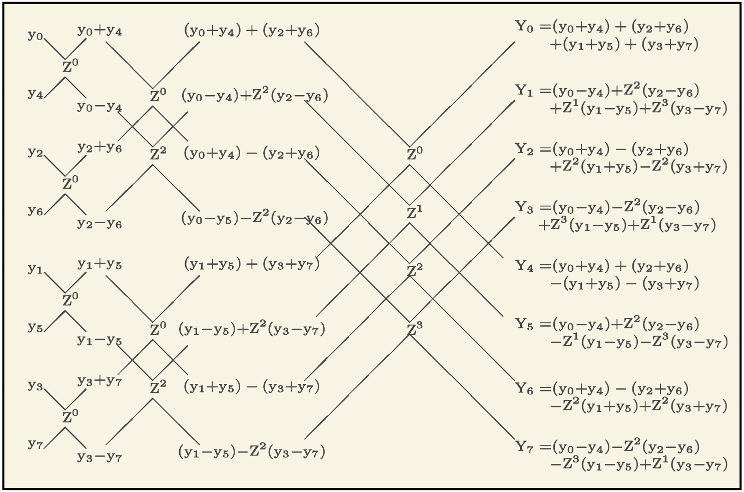


Bit Reversal: Order 0–7 $\rightarrow$ 0, 4, 2, 6, 1, 5, 3, 7

Dec	*Binary*	*Reversed*	*Dec Rev*
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

Binary-Reversed 0–11			
Dec	*Binary*	*Reversed*	*Dec Rev*
0	0000	0000	0
1	0001	1000	8
2	0010	0100	4
3	0011	1100	12
4	0100	0010	2
5	0101	1010	10
6	0110	0110	6
7	0111	1110	14
8	1000	0001	1
9	1001	1001	9
10	1010	0101	5
11	1011	1101	13

Reversed Input → Ordered Output



Implementation

FFT.java, python

- First FFT: Brenner, 1967 Fortran IV, ??
- Program: 16 complex data points; Reorders via bit reversal
- Four butterfly ops; 1-D array for speed
- Look at program!

Exercise

Exercise

1. Compile, execute `FFT.java`; understand output
2. Invert output and compare
3. Compare FFT to DFT for precision and speed