数字信号处理

快速傅里叶变换之一：Radix-2 DIT FFT

李光平

2026

广东工业大学信息工程学院

目录

引言

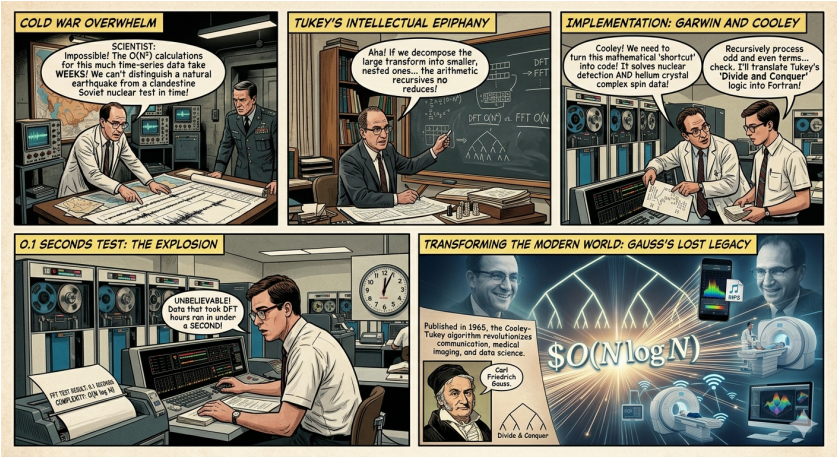
引言

- 快速傅里叶变换（FFT）是计算离散傅里叶变换（DFT）的高效算法
- 直接计算 DFT 的复杂度为 $O(N^2)$
- FFT 将计算复杂度降低到 $O(N \log N)$

备注

FFT 不是一种新的变换，而是 DFT 的一种快速计算方法。

FFT 的历史


COLD WAR OVERWHELM
SCIENTIST: Impossible! The O(N²) calculations for this much time-series data take WEEKS! We can't distinguish a natural earthquake from a clandestine Soviet nuclear test in time!
TUKEY'S INTELLECTUAL EPIPHANY
Aha! If we decompose the large transform into smaller, nested ones... the arithmetic recursives no reduces!
DFT
FFT
DFT O(N²) vs. FFT O(N
IMPLEMENTATION: GARWIN AND COOLEY
Cooley! We need to turn this mathematical 'shortcut' into code! It solves nuclear detection AND helium crystal complex spin data!
Recursively process odd and even terms... check. I'll translate Tukey's 'Divide and Conquer' logic into Fortran!
0.1 SECONDS TEST: THE EXPLOSION
UNBELIEVABLE! Data that took DFT hours ran in under a SECOND!
FFT TEST RESULT: 0.1 SECONDS COMPLEXITY: O(N log N)
TRANSFORMING THE MODERN WORLD: GAUSS'S LOST LEGACY
Published in 1965, the Cooley-Tukey algorithm revolutionizes communication, medical imaging, and data science.
Carl Friedrich Gauss.
Divide & Conquer
$O(N \log N)$

DFT 的矩阵表示

正变换（DFT）：$X[k] = \sum_{n=0}^{N-1} x[n]\, W_N^{nk}$，矩阵形式：

$$\begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & W_N^{1} & \cdots & W_N^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & \cdots & W_N^{(N-1)^2} \end{bmatrix}}_{\mathbf{W}_N} \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}$$

即 $\mathbf{X} = \mathbf{W}_N\, \mathbf{x}$，其中 $[\mathbf{W}_N]_{k,n} = W_N^{nk}$。

反变换（IDFT）：$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]\, W_N^{-nk}$，矩阵形式：$\mathbf{x} = \frac{1}{N}\, \mathbf{W}_N^{*}\, \mathbf{X}$

其中 $\mathbf{W}_N^{*}$ 是 $\mathbf{W}_N$ 的共轭矩阵，满足 $\mathbf{W}_N^{*} \mathbf{W}_N = N\, \mathbf{I}$。

预备数学等式

旋转因子的基本性质：

$$W_N^{nk} = e^{-j\frac{2\pi nk}{N}}$$

- 抽取性质：$W_{Nm}^{nkm} = W_{N/m}^{nk/m} = W_N^{nk}$
- 半周期性质：$W_N^{N/2} = -1$
- 周期性质：$W_N^{k+N} = W_N^k$

备注

这些性质是推导 FFT 算法的数学基础。

基本原理

第一级分解：奇偶抽取

序列长度 $N = 2^M$，将 $x[n]$ 按 n 的奇偶分为两组：

- n 为偶数：$x[0], x[2], \dots, x[N-2]$　令 $x_1[r] = x[2r]$
- n 为奇数：$x[1], x[3], \dots, x[N-1]$　令 $x_2[r] = x[2r+1]$

DFT 按奇偶拆分：

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk} = \sum_{n\text{ 偶}} x[n] W_N^{nk} + \sum_{n\text{ 奇}} x[n] W_N^{nk}$$

$$= \sum_{r=0}^{N/2-1} x[2r] W_N^{2rk} + \sum_{r=0}^{N/2-1} x[2r+1] W_N^{(2r+1)k}$$

$$= \sum_{r=0}^{N/2-1} x_1[r] W_{N/2}^{rk} + W_N^k \sum_{r=0}^{N/2-1} x_2[r] W_{N/2}^{rk}$$

$$\boxed{X[k] = X_1[k] + W_N^k X_2[k]}, \quad k = 0, 1, \dots, N/2 - 1$$

第一级分解：对称性质

利用 $W_N^{N/2} = -1$，可得后半部分频点：

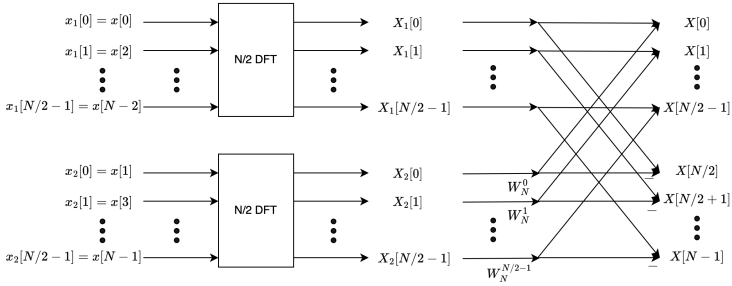
$$\begin{aligned} X[k + N/2] &= X_1[k + N/2] + W_N^{(k+N/2)} X_2[k + N/2] \\ &= X_1[k] - W_N^k X_2[k] \end{aligned}$$

$$\boxed{X[k + N/2] = X_1[k] - W_N^k X_2[k]}, \quad k = 0, 1, \ldots, N/2 - 1$$

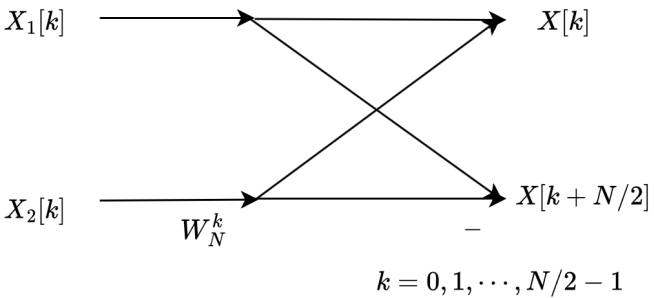
小结

一个 N 点 DFT 分解为两个 $N/2$ 点 DFT（X_1、X_2），再通过蝶形运算组合。

蝶形运算


$x_1[0] = x[0]$
$x_1[1] = x[2]$
$x_1[N/2-1] = x[N-2]$
N/2 DFT
$X_1[0]$
$X_1[1]$
$X_1[N/2-1]$
$x_2[0] = x[1]$
$x_2[1] = x[3]$
$x_2[N/2-1] = x[N-1]$
N/2 DFT
$X_2[0]$
$X_2[1]$
$X_2[N/2-1]$
W_N^0
W_N^1
$W_N^{N/2-1}$
$X[0]$
$X[1]$
$X[N/2-1]$
$X[N/2]$
$X[N/2+1]$
$X[N-1]$

蝶形运算结构



计算量分析

每个蝶形运算需要：1 次复数乘法，2 次复数加法

第二级分解

对 $X_1[k]$ 继续按奇偶拆分——令 $x_3[r] = x_1[2r]$，$x_4[r] = x_1[2r+1]$：

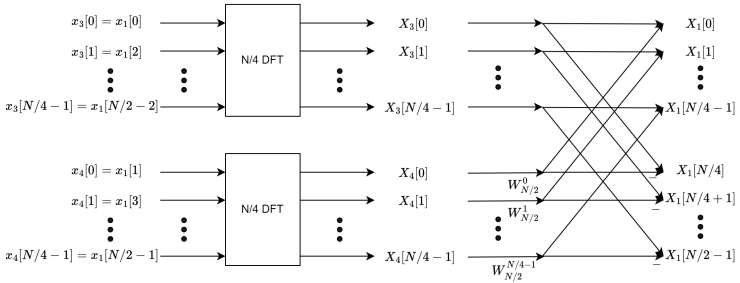
$$
\begin{aligned}
X_1[k] &= \sum_{n=0}^{N/2-1} x_1[n]\, W_{N/2}^{nk} \\
&= \sum_{r=0}^{N/4-1} x_1[2r]\, W_{N/2}^{2rk} + \sum_{r=0}^{N/4-1} x_1[2r+1]\, W_{N/2}^{(2r+1)k} \\
&= \sum_{r=0}^{N/4-1} x_3[r]\, W_{N/4}^{rk} + W_{N/2}^{k} \sum_{r=0}^{N/4-1} x_4[r]\, W_{N/4}^{rk}
\end{aligned}
$$

$$
\boxed{X_1[k] = X_3[k] + W_{N/2}^{k}\, X_4[k]}, \quad k = 0, \dots, N/4 - 1
$$

$$
\boxed{X_1[k + N/4] = X_3[k] - W_{N/2}^{k}\, X_4[k]}, \quad k = 0, \dots, N/4 - 1
$$

对 $X_2[k]$ 同理进行分解

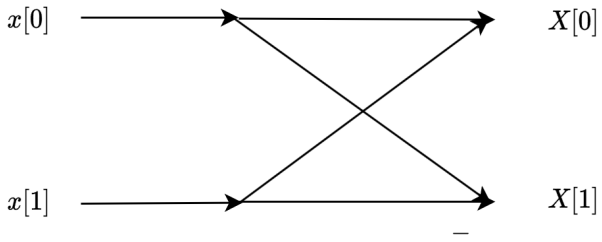
逐级分解



- 继续对 $X_1[k]$ 和 $X_2[k]$ 进行分解
- 直到分解为 2 点 DFT 为止

2 点 DFT

2 点 DFT 矩阵及蝶形运算：



- 2 点 DFT 不需要乘法运算
- $X[0] = x[0] + x[1]$，$X[1] = x[0] - x[1]$

计算示例

8 点 FFT 推导：第一级分解

设 $x[n] = [1,\ 2,\ 0,\ 1,\ 3,\ 2,\ 1,\ 0]$，$N = 8$。

按奇偶抽取：

$$x_1 = [x[0], x[2], x[4], x[6]] = [1,\ 0,\ 3,\ 1]$$

$$x_2 = [x[1], x[3], x[5], x[7]] = [2,\ 1,\ 2,\ 0]$$

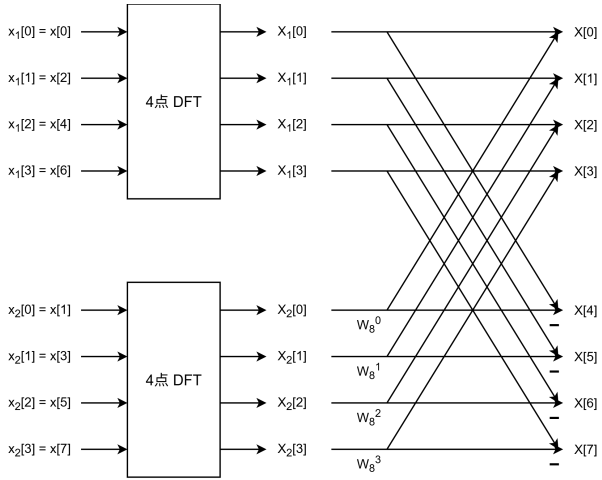
由第一级分解公式：

$$X[k] = X_1[k] + W_8^k X_2[k]$$

$$X[k+4] = X_1[k] - W_8^k X_2[k]$$

$k = 0, 1, 2, 3$

8 点 FFT 推导：第一级分解图示



8 点 FFT 推导：第二级分解

对 $x_1 = [1,\ 0,\ 3,\ 1]$ 继续奇偶抽取：

$$x_3 = [1,\ 3], \quad x_4 = [0,\ 1]$$

对 $x_2 = [2,\ 1,\ 2,\ 0]$ 继续奇偶抽取：

$$x_5 = [2,\ 2], \quad x_6 = [1,\ 0]$$

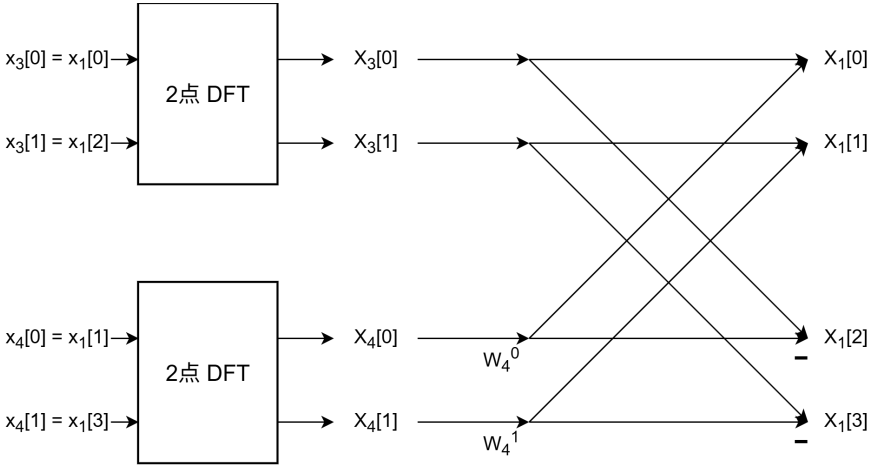
以 x_1 为例：

$$X_1[k] = X_3[k] + W_4^k X_4[k]$$

$$X_1[k+2] = X_3[k] - W_4^k X_4[k]$$

$k = 0, 1$

8 点 FFT 推导：第二级分解图示



8 点 FFT 推导：第三级——2 点 DFT

第三级：对每个 2 点子序列直接做 DFT：

$x_3 = [1,\ 3]$：

$$X_3[0] = 1 + 3 = 4, \quad X_3[1] = 1 - 3 = -2$$

$x_4 = [0,\ 1]$：

$$X_4[0] = 0 + 1 = 1, \quad X_4[1] = 0 - 1 = -1$$

$x_5 = [2,\ 2]$：

$$X_5[0] = 2 + 2 = 4, \quad X_5[1] = 2 - 2 = 0$$

$x_6 = [1,\ 0]$：

$$X_6[0] = 1 + 0 = 1, \quad X_6[1] = 1 - 0 = 1$$

8 点 FFT 推导：第二级蝶形组合

利用 $W_4^0 = 1$，$W_4^1 = -j$，组合得 4 点 DFT：

$X_1[k]$（由 X_3、X_4 组合）：

$$X_1[0] = 4 + 1 \cdot 1 = 5$$
$$X_1[1] = -2 + (-j)(-1) = -2 + j$$
$$X_1[2] = 4 - 1 \cdot 1 = 3$$
$$X_1[3] = -2 - (-j)(-1) = -2 - j$$

$X_2[k]$（由 X_5、X_6 组合）：

$$X_2[0] = 4 + 1 \cdot 1 = 5$$
$$X_2[1] = 0 + (-j)(1) = -j$$
$$X_2[2] = 4 - 1 \cdot 1 = 3$$
$$X_2[3] = 0 - (-j)(1) = j$$

8 点 FFT 推导：第一级蝶形组合

旋转因子：$W_8^0 = 1$，$W_8^1 = \frac{\sqrt{2}}{2}(1-j)$，$W_8^2 = -j$，$W_8^3 = \frac{\sqrt{2}}{2}(-1-j)$

$$X[0] = X_1[0] + W_8^0 X_2[0] = 5 + 1 \cdot 5 = 10$$

$$X[1] = X_1[1] + W_8^1 X_2[1] = (-2 + j) + \frac{\sqrt{2}}{2}(1-j)(-j) = -2 + j + \frac{\sqrt{2}}{2}(-j - 1)$$

$$= -2 - \frac{\sqrt{2}}{2} + (1 - \frac{\sqrt{2}}{2})j$$

$$X[2] = X_1[2] + W_8^2 X_2[2] = 3 + (-j)(3) = 3 - 3j$$

$$X[3] = X_1[3] + W_8^3 X_2[3] = (-2 - j) + \frac{\sqrt{2}}{2}(-1-j)(j) = -2 + \frac{\sqrt{2}}{2} + (-1 + \frac{\sqrt{2}}{2})j$$

后 4 点由对称性：$X[k+4] = X_1[k] - W_8^k X_2[k]$

$$X[4] = 5 - 5 = 0, \quad X[5] = -2 + \frac{\sqrt{2}}{2} + (1 + \frac{\sqrt{2}}{2})j$$

$$X[6] = 3 + 3j, \quad X[7] = -2 - \frac{\sqrt{2}}{2} + (-1 - \frac{\sqrt{2}}{2})j$$

8 点 FFT 推导：结果验证

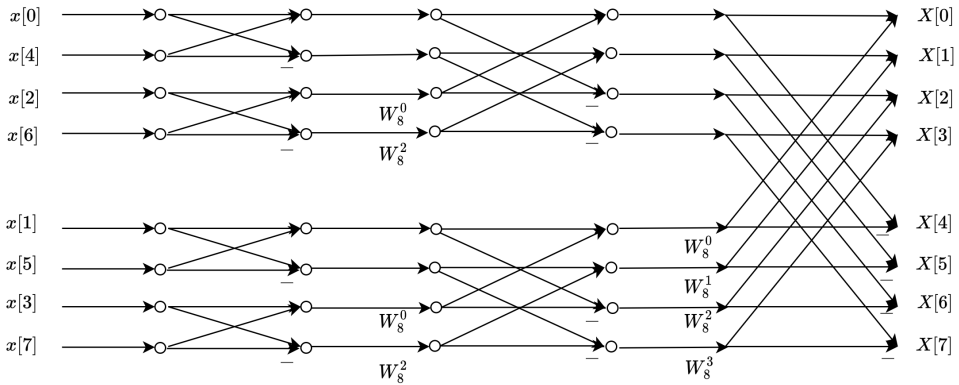
$$X = [10,\ -2-\frac{\sqrt{2}}{2}+(1-\frac{\sqrt{2}}{2})j,\ 3-3j,\ -2+\frac{\sqrt{2}}{2}+(-1+\frac{\sqrt{2}}{2})j,\ 0,\ \cdots]$$

回顾整个过程

1. **第一级**：$N = 8$ 点序列按奇偶拆为两个 4 点子序列
2. **第二级**：每个 4 点子序列再拆为两个 2 点子序列
3. **第三级**：2 点 DFT 直接计算（无需乘法）
4. **逐级回代**：用蝶形公式 $X[k] = A + W \cdot B$，$X[k+N/2] = A - W \cdot B$ 逐级组合

共 $\log_2 8 = 3$ 级，每级 4 个蝶形，总计 12 次复数乘法（直接 DFT 需 64 次）。

8 点 FFT 流图


$x[0]$
$x[4]$
$x[2]$
$x[6]$
$x[1]$
$x[5]$
$x[3]$
$x[7]$
W_8^0
W_8^2
W_8^0
W_8^2
W_8^0
W_8^1
W_8^2
W_8^3
$X[0]$
$X[1]$
$X[2]$
$X[3]$
$X[4]$
$X[5]$
$X[6]$
$X[7]$

计算复杂度

计算复杂度

- 复数乘法次数：$\frac{N}{2}\log_2 N$
- 复数加法次数：$N\log_2 N$

备注

与直接 DFT 的 N^2 次复数乘法和 $N(N-1)$ 次复数加法相比，FFT 具有显著的计算优势。当 $N = 1024$ 时，FFT 的乘法次数仅为直接 DFT 的约 0.5%。

DFT 与 FFT 计算量对比

N	DFT 乘法 N^2	FFT 乘法 $\frac{N}{2}\log_2 N$	比值	加速倍数
8	64	12	18.8%	×5
16	256	32	12.5%	×8
64	4096	192	4.7%	×21
256	65536	1024	1.6%	×64
1024	1048576	5120	0.49%	×205
4096	16777216	24576	0.15%	×683

结论

N 越大，FFT 的优势越明显。实际工程中 N 常取 1024 ~ 4096，FFT 可获得数百倍加速。

实现分析

倒位序（Bit-Reversal）

- DIT FFT 的输入需要按倒位序排列
- 将自然顺序的二进制表示进行位倒置

自然顺序	自然顺序二进制	倒位序二进制	倒位序
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

旋转因子与原位计算

- 每一级的旋转因子 W_N^k 可按规则选取
- 第 m 级（$m = 1, 2, \ldots, M$）使用的旋转因子为：

$$W_N^p, \quad p = 0,\ \frac{N}{2^m},\ 2\cdot\frac{N}{2^m},\ \ldots,\ (2^{m-1}-1)\cdot\frac{N}{2^m}$$

  即步长为 $N/2^m$，共 2^{m-1} 个不同的旋转因子
- 原位计算（In-place computation）：蝶形运算的输出可以直接存放在输入所占用的存储单元中
- 整个 FFT 运算只需要 N 个复数存储单元

备注

原位计算大大节省了存储空间，是 FFT 硬件实现的重要特性。

本节小结

1. **FFT 本质**：不是新变换，而是 DFT 的高效计算算法
2. **核心思想**：利用旋转因子的对称性和周期性，将 N 点 DFT 逐级分解为 2 点 DFT（分治策略）
3. **蝶形运算**：每级 $N/2$ 个蝶形，共 $\log_2 N$ 级
4. **计算复杂度**：复数乘法 $\frac{N}{2}\log_2 N$，相比 DFT 的 N^2 大幅降低
5. **实现要点**：输入需倒位序排列；支持原位计算，仅需 N 个存储单元

谢谢！ 欢迎提问与讨论