

计算方法

Computational Methods

Lecture 5: 离散傅里叶变换与快速傅里叶变换

Discrete Fourier Transform and Fast Fourier Transform

刘景铎

南京大学 · 计算机学院

课程信息

计算方法 Fall 2026

授课教师 刘景铎 liu@nju.edu.cn

上课 周四 5-6 节 · 1-18 周 · 仙 I-319

答疑 周四下午 4 点

助教 周宇恒 · 侯哲

作业 nm_nju_2026@163.com

课程主页 TCS Wiki · Fall 2026

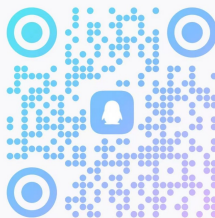
QQ 群 1054896504 加入时注明姓名、专业、
学号

教材: Timothy Sauer, 《数值分析》(原书第 2 版)。 最新公告以
课程主页为准。



计算方法 Fall 2026

群号: 1054896504



扫一扫二维码, 加入群聊

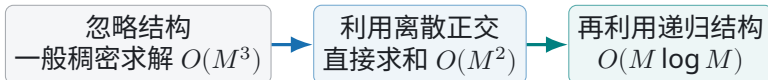


QQ

回顾：三角插值留下了两个问题

上一讲的核心经验是：**换到正交坐标以后，原本耦合的系数可以彼此独立地求出。** 对奇数个等距样本，上一讲的三角插值最终变成一个结构化线性系统。为避免和后面 FFT 的递归长度混淆，这里暂记

$$\boxed{M := 2m + 1}, \quad x_j = \frac{2\pi j}{M}, \quad j = 0, \dots, M - 1.$$



今天只回答两个问题：为什么这个结构化系统**容易求逆**？为什么同一个变换又能**算得更快**？

同一个多项式，两种表示

给定次数小于 n 的多项式

$$p(x) = a_0 + a_1x + \cdots + a_{n-1}x^{n-1}.$$

它可以用两种方式表示：

系数表示

$$(a_0, a_1, \dots, a_{n-1}).$$

适合：加法、求导、直接读出各阶系数。

取值表示

$$(p(x_0), p(x_1), \dots, p(x_{N-1})).$$

若节点互异且 $N \geq n$ ，这些取值也能唯一确定 p 。

第二讲已经见过“系数 $\leftrightarrow$ 取值”；今天问的是：能不能在两种表示之间快速来回？

多项式乘法：哪一种表示更合适？

给定

$$p(x) = \sum_{j=0}^{n-1} a_j x^j, \quad q(x) = \sum_{j=0}^{n-1} b_j x^j,$$

要求 $r(x) = p(x)q(x)$ 的系数。

在系数表示中

$$c_k = \sum_{j=0}^k a_j b_{k-j},$$

其中把超出 $0, \dots, n-1$ 的系数视为 0。
每个 c_k 都要做一段求和，朴素算法总共需要

$$\boxed{O(n^2)}.$$

在取值表示中

$$r(x_\ell) = p(x_\ell)q(x_\ell).$$

若已经知道两边的函数值，只需要逐点相乘：

$$\boxed{O(N)}.$$

乘法本身不难；真正的瓶颈是“系数 $\leftrightarrow$ 取值”。

目标：把“换表示”也做到 $O(n \log n)$

若 $\deg p, \deg q < n$ ，则 $\deg(pq) \leq 2n - 2$ 。因此只要取

$$N \geq 2n - 1$$

个互异采样点，乘积多项式的这些取值就足以唯一确定它的系数。取 $N = \Theta(n)$ 即可。



如果正、逆两次“换表示”都只需 $O(N \log N)$ ，那么

$$O(N \log N) + O(N) + O(N \log N) = O(n \log n).$$

问题变成：能否找到一组特殊采样点，既让变换容易求逆，又让变换本身可以快速计算？

复数：今天只需要三件事

对复数 $z = a + bi$ ，也可以写成极坐标形式

$$z = re^{i\theta} = r(\cos \theta + i \sin \theta).$$

其中模长 $|z| = r$ 。今天只会反复用到：

1. 乘法对应角度相加：

$$e^{i\alpha}e^{i\beta} = e^{i(\alpha+\beta)};$$

2. 共轭对应角度取反：

$$\overline{re^{i\theta}} = re^{-i\theta};$$

3. 单位圆上的点满足 $|z| = 1$ 。

复数不是额外负担：它把正弦和余弦统一成一个指数函数。

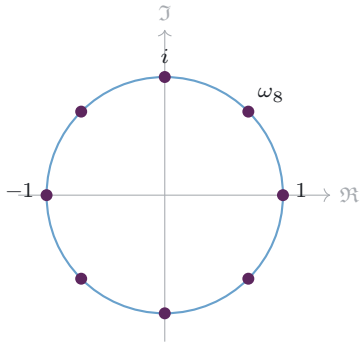
单位根：复平面上的等距采样点

定义 N 次单位根

$$\omega_N := e^{2\pi i/N},$$

$$1, \omega_N, \omega_N^2, \dots, \omega_N^{N-1}.$$

它们恰好是方程 $z^N = 1$ 的全部解。



- 在单位圆上**等角度间隔**；
- 从 2 次单位根到 4, 8, ... 次单位根，
可以看成逐层对前一层根**开平方**；
- 当 N 为偶数时， z 与 $-z$ 都在根集中；
平方后二者合并成一个点，根集规模减半。

为什么偏偏选单位根？两个性质正好够用

性质 1：共轭 / 周期

$$\overline{\omega_N^j} = \omega_N^{-j}, \quad \omega_N^N = 1.$$

因此许多指数和会出现**精确抵消**。

⇒ **离散正交、容易求逆**

性质 2：平方 / 成对

若 N 为偶数，

$$(\omega_N^j)^2 = \omega_{N/2}^j, \quad -\omega_N^j = \omega_N^{j+N/2}.$$

所以 z 与 $-z$ 平方后落到同一个点。

⇒ **规模减半、可以分治**

同一组采样点，同时适合“显式求逆”和“递归计算”。

DFT: 在单位根上做多项式求值

设 $a = (a_0, \dots, a_{N-1})$, 并定义

$$p(z) = \sum_{j=0}^{N-1} a_j z^j.$$

离散傅里叶变换 (DFT) 就是在 N 次单位根上求值:

$$\hat{a}_k := p(\omega_N^k) = \sum_{j=0}^{N-1} a_j \omega_N^{kj}, \quad k = 0, \dots, N-1.$$

因此

$$(a_0, \dots, a_{N-1}) \mapsto (\hat{a}_0, \dots, \hat{a}_{N-1}).$$

DFT 不是凭空出现的一条公式: 它就是在一组特殊节点上做多项式求值。

一个简单例子：(1, 1, 1, 1) 的 DFT

取 $N = 4$,

$$a = (1, 1, 1, 1), \quad p(z) = 1 + z + z^2 + z^3.$$

4 次单位根是

$$1, i, -1, -i.$$

于是

$$p(1) = 4,$$

而对其他三个单位根 $z \neq 1$, 因为 $z^4 = 1$,

$$1 + z + z^2 + z^3 = \frac{1 - z^4}{1 - z} = 0.$$

所以

$$\text{DFT}(1, 1, 1, 1) = (4, 0, 0, 0).$$

常数序列只留下 $k = 0$ 分量（通常称为“零频”）；其余分量恰好完全抵消。

把 DFT 写成一个矩阵：先看 $N = 4$

一般地，

$$\hat{a} = F_N a, \quad (F_N)_{kj} = \omega_N^{kj}.$$

先看 $N = 4$ ，其中 $\omega_4 = i$ ：

$$F_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}.$$

四列依次是单项式

$$1, z, z^2, z^3$$

在 $1, i, -1, -i$ 这四个单位根上的取值向量。现在“DFT 为什么可逆？”变成一个熟悉的问题：这些列向量之间有什么几何关系？

复数空间中，“正交矩阵”对应酉矩阵

实数空间：第四讲

$$\langle u, v \rangle = u^\top v, \quad Q^\top Q = I.$$

列向量标准正交

$$\Rightarrow Q^{-1} = Q^\top.$$

复数空间：今天

$$\langle u, v \rangle = u^* v = \sum_k \overline{u_k} v_k,$$

其中

$$A^* := \overline{A}^\top$$

称为 A 的**共轭转置**。对方阵 U ，若 $U^* U = I$ ，称 U 为**酉矩阵**。

今天的傅里叶矩阵没有做归一化；我们将证明

$$F_N^* F_N = NI$$

$$\frac{1}{\sqrt{N}} F_N \text{ 是酉矩阵}$$

离散正交性的核心：一个几何级数

对整数 r ，考虑

$$S_r := \sum_{k=0}^{N-1} \omega_N^{kr}.$$

若 $N \mid r$ ，每一项都是 1，所以 $S_r = N$ 。若 $N \nmid r$ ，则 $\omega_N^r \neq 1$ ，用几何级数公式：

$$S_r = \frac{1 - (\omega_N^r)^N}{1 - \omega_N^r} = \frac{1 - 1}{1 - \omega_N^r} = 0.$$

因此

$$\sum_{k=0}^{N-1} \omega_N^{kr} = \begin{cases} N, & N \mid r, \\ 0, & N \nmid r. \end{cases}$$

等距分布让不同“频率”在一整圈上精确抵消。

傅里叶矩阵的列向量彼此正交

$F_N^* F_N$ 的第 (i, j) 个元素是

$$(F_N^* F_N)_{ij} = \sum_{k=0}^{N-1} \overline{\omega_N^{ki}} \omega_N^{kj} = \sum_{k=0}^{N-1} \omega_N^{k(j-i)}.$$

由上一页的单位根求和公式,

$$(F_N^* F_N)_{ij} = \begin{cases} N, & i = j, \\ 0, & i \neq j. \end{cases}$$

所以

$$\boxed{F_N^* F_N = NI.}$$

第四讲的主角再次出现：一旦列向量正交，求系数就不需要解一个耦合的线性系统。

逆 DFT: 正交性直接给出逆变换

由

$$F_N^* F_N = NI$$

立刻得到

$$F_N^{-1} = \frac{1}{N} F_N^*.$$

也就是说, 若

$$\hat{a}_k = \sum_{j=0}^{N-1} a_j \omega_N^{kj},$$

则

$$a_\ell = \frac{1}{N} \sum_{k=0}^{N-1} \hat{a}_k \omega_N^{-k\ell}.$$

正变换与逆变换的公式几乎一样：只需把 ω_N 换成 ω_N^{-1} ，最后除以 N 。

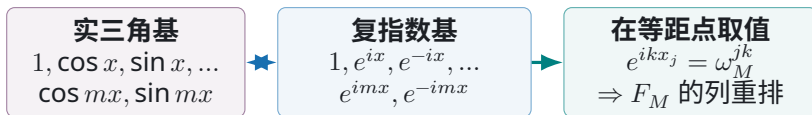
回到上一讲：三角插值为什么唯一可解？

上一讲取奇数 $M = 2m + 1$ ，在

$$x_j = \frac{2\pi j}{M}, \quad j = 0, \dots, M-1,$$

用实三角基做插值。把这件事分成三步看：

Euler 公式：可逆换基



并且

$$\{-m, -m+1, \dots, m\} \pmod{M} = \{0, 1, \dots, M-1\},$$

所以复指数基下的采样矩阵只是 F_M 的列重排。

F_M 可逆 + 换基可逆 $\Rightarrow$ 上一讲的三角插值系统唯一可解。

从复指数正交，看懂实三角正交

仍在 $M = 2m + 1$ 个等距采样点 $x_j = \frac{2\pi j}{M}$ 上。把复指数函数的样本值看成 $\mathbb{C}^M$ 中的向量：

$$v_k := (e^{ikx_0} \quad e^{ikx_1} \quad \dots \quad e^{ikx_{M-1}})^\top, \quad k = -m, \dots, m.$$

单位根求和一次给出全部内积：

$$v_k^* v_\ell = \sum_{j=0}^{M-1} \omega_M^{j(\ell-k)} = \begin{cases} M, & k = \ell, \\ 0, & k \neq \ell. \end{cases}$$

特别地， $v_0 = \mathbf{1}$ ；而对 $k = 1, \dots, m$ ，余弦、正弦的样本向量只是

$$\mathbf{c}_k = \frac{v_k + v_{-k}}{2}, \quad \mathbf{s}_k = \frac{v_k - v_{-k}}{2i}.$$

因此 $v_0, \mathbf{c}_1, \mathbf{s}_1, \dots, \mathbf{c}_m, \mathbf{s}_m$ 两两正交。**第四讲的实三角正交，与今天的傅里叶矩阵正交，是同一个向量几何事实的两套坐标。**

第一个问题解决了；但直接计算仍是 $O(N^2)$

回到一般长度 N 的 DFT，离散正交已经告诉我们：

$$\boxed{\text{系数 } a \xleftrightarrow[\frac{1}{N} F_N^*]{F_N} \text{取值 } \hat{a}}$$

而且正、逆变换都可以写成一次**稠密矩阵-向量乘法**。但是一个稠密的 $N \times N$ 矩阵乘向量仍要

$$\boxed{O(N^2)}$$

次运算。

正交性解决了“怎么求逆”；它还没有解释“为什么能算得更快”。

接下来利用单位根的第二个性质：平方以后，问题规模会减半。

快速傅里叶变换 (FFT): 先把偶数次和奇数次拆开

FFT 用分治来计算 DFT。先设 N 为 2 的幂, 并写

$$p(z) = \sum_{j=0}^{N-1} a_j z^j.$$

把偶数次项和奇数次项分开:

$$E(w) := a_0 + a_2 w + a_4 w^2 + \cdots,$$

$$O(w) := a_1 + a_3 w + a_5 w^2 + \cdots.$$

于是

$$p(z) = E(z^2) + zO(z^2).$$

E 和 O 的次数都小于 $N/2$ 。如果 z^2 只需要遍历 $N/2$ 个点, 原来的 N 点求值问题就被拆成两个一半规模的问题。

为什么拆完以后，问题真的变小了？

当 z 遍历全部 N 次单位根时，

$$z = \omega_N^k, \quad k = 0, \dots, N-1.$$

平方以后

$$z^2 = \omega_N^{2k} = \omega_{N/2}^k.$$

所以 z^2 只遍历 $N/2$ 次单位根。更具体地， z 与 $-z$ 平方后落到同一个点：

$$(-z)^2 = z^2.$$

因此只要已经知道

$$E(z^2), \quad O(z^2),$$

就可以同时得到一对答案：

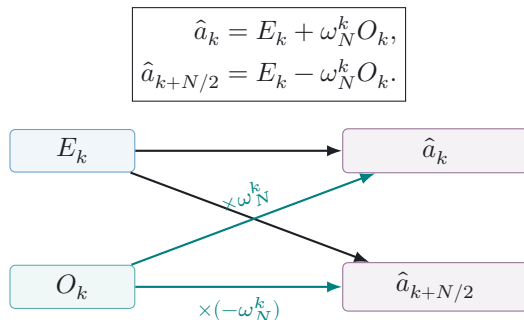
$$\begin{aligned} p(z) &= E(z^2) + zO(z^2), \\ p(-z) &= E(z^2) - zO(z^2). \end{aligned}$$

一次“蝶形”操作，把两个子问题的答案合并

若已经递归算出

$$E_k := E(\omega_{N/2}^k), \quad O_k := O(\omega_{N/2}^k),$$

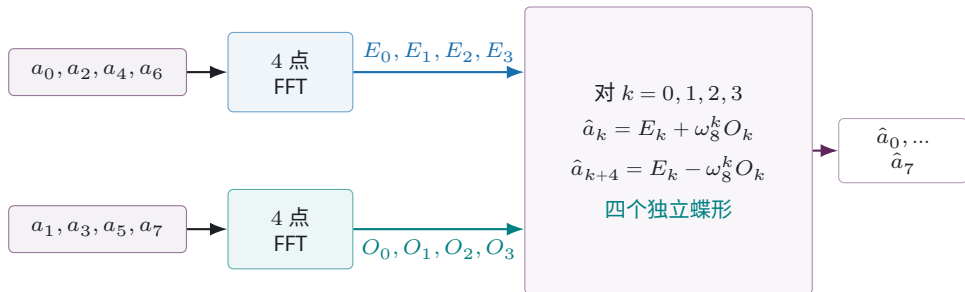
则令 $z = \omega_N^k$:



每个 k 只需要常数次加法和乘法；一层合并总共 $O(N)$ 。

8 点 FFT: 两个 4 点变换, 再做四个蝶形

以 $N = 8$ 为例, 先把偶数下标和奇数下标分开, 各做一个 4 点 FFT:



两个 4 点 FFT 继续按同样方式递归, 直到长度 1。递归负责“算小问题”, 蝶形负责“把两个小问题在线性时间内拼回大问题”。

把推导写成递归算法

设 $N = 2^s$ ，输入系数向量 $a = (a_0, \dots, a_{N-1})$ ，并令 $\omega = \omega_N$ 。

FFT(a, ω)

1. 若 $N = 1$ ，返回 (a_0) ；
2. 拆成偶数、奇数下标： $a^{(e)} = (a_0, a_2, \dots)$ ， $a^{(o)} = (a_1, a_3, \dots)$ ；
3. 递归计算

$$E = \text{FFT}(a^{(e)}, \omega^2), \quad O = \text{FFT}(a^{(o)}, \omega^2);$$

4. 对 $k = 0, \dots, N/2 - 1$ ，做一个蝶形：

$$\hat{a}_k = E_k + \omega^k O_k, \quad \hat{a}_{k+N/2} = E_k - \omega^k O_k.$$

两个规模 $N/2$ 的递归调用 + $N/2$ 个常数成本蝶形。

复杂度：每一层 $O(N)$ ，一共 $\log_2 N$ 层

记 $T(N)$ 为长度 N 的 DFT 用 FFT 计算所需的运算次数。递归式是

$$T(N) = 2T(N/2) + cN.$$

展开：

$$\begin{aligned} T(N) &= 2\left(2T(N/4) + cN/2\right) + cN \\ &= 4T(N/4) + 2cN \\ &= \dots \\ &= N T(1) + cN \log_2 N. \end{aligned}$$

因此

$$T(N) = O(N \log N).$$

不是“矩阵变稀疏了”，而是这个特殊矩阵乘法可以递归分解成很多很小的结构化运算。

逆 FFT：同一套递归，只把单位根方向反过来

逆 DFT 是

$$a_\ell = \frac{1}{N} \sum_{k=0}^{N-1} \hat{a}_k \omega_N^{-k\ell}.$$

它和正 DFT 的形式完全一样，只是

$$\omega_N \longrightarrow \omega_N^{-1}$$

并在最后乘 $1/N$ 。因此同样的偶/奇拆分、平方单位根、蝶形合并都可以照搬：

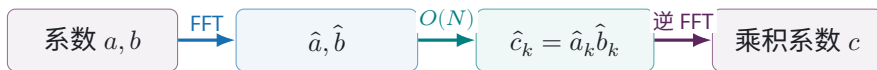
逆 FFT 也只需要 $O(N \log N)$ 。

DFT 与逆 DFT 不仅公式相似，还共享同一套快速递归结构。

回到多项式乘法：现在整条流程都快了

若 $\deg p, \deg q < n$ ，取**最小的 2 的幂** $N \geq 2n - 1$ ，并把系数补零到长度 N 。于是

$$2n - 1 \leq N < 2(2n - 1) < 4n, \quad \text{所以 } N = \Theta(n).$$



总复杂度：

$$O(N \log N) + O(N) + O(N \log N) = \boxed{O(n \log n)}.$$

本讲开头提出的“快速换表示”，到这里完整实现。

卷积 $\leftrightarrow$ 逐点乘法

把序列补零到足够长以后，同一个乘法运算在两组坐标里长得完全不同：

原来的坐标：线性卷积

一个输出坐标要汇总许多输入坐标：

$$c_k = \sum_j a_j b_{k-j}.$$

傅里叶坐标：逐点乘法

每个输出坐标彼此独立：

$$\hat{c}_k = \hat{a}_k \hat{b}_k.$$

DFT 严格对应
循环卷积

补零到
 $N \geq n_a + n_b - 1$

没有“回绕”
 $\Rightarrow$ **线性卷积**

选择好的表示，可以把一个耦合运算变成彼此独立的标量运算。

为什么今天还到处能看到卷积？

卷积自然出现在许多计算问题中：

信号与滤波

平滑、去噪等操作，常可写成信号与滤波核的卷积。

图像处理

模糊、锐化可写成二维卷积；去模糊则是相应的反问题。

深度学习

卷积层、长卷积都包含结构化乘加；大核或长序列时可考虑 FFT 加速。

共同思路：把卷积换到傅里叶坐标，就变成彼此独立的逐点乘法。

本讲主线：同一组单位根，两种结构

1. 离散正交 $\Rightarrow$ 容易求逆

$$\sum_{k=0}^{N-1} \omega_N^{kr} = \begin{cases} N, & N \mid r, \\ 0, & N \nmid r, \end{cases}$$

$$F_N^* F_N = NI, \quad F_N^{-1} = \frac{1}{N} F_N^*.$$

2. 递归结构 $\Rightarrow$ 算得更快

$$p(z) = E(z^2) + zO(z^2),$$

$$(\omega_N^k)^2 = \omega_{N/2}^k,$$

$$T(N) = 2T(N/2) + O(N) = O(N \log N).$$

好的表示不只是让公式更漂亮；它可以直接改变算法的复杂度。

一般线性系统：高斯消元

傅里叶矩阵非常特殊：离散正交让它容易求逆，递归结构让 $F_N a$ 可以快速计算。

如果 A 没有这些特殊结构，怎样求解

$$\boxed{Ax = b}?$$

回顾：线性方程组的解集只有三种可能

给定 $A \in \mathbb{R}^{m \times n}$ 、 $b \in \mathbb{R}^m$ ，寻找 $x \in \mathbb{R}^n$ 使 $Ax = b$ 。第三讲已经知道：

无解	唯一解	无穷多解.
----	-----	-------

超定： $m > n$

Overdetermined

方程多于未知量，精确解不一定存在；
第三讲因此改为求

$$\min_x \|Ax - b\|.$$

欠定： $m < n$

Underdetermined

若 $Ax = b$ 有解，则存在非零 h 满足
 $Ah = 0$ ，从而

$$A(x + th) = b.$$

下面先聚焦方阵 $m = n$ ；矩阵若奇异，消元过程也会把它暴露出来。

解 $Ax = b$ 与显式计算 A^{-1} 是两回事。

高斯消元：沿主元列推进，只做等价行变换

高斯消元逐列消去主元下方的元素，把矩阵变成上三角形；矩形系统也使用同样的行操作。为了保持未知量 x 不变，每一步都用**初等行操作**把方程组变成等价方程组：

1. **交换两行**：交换两组方程；
2. **给一行乘一个非零常数**；
3. **在一行上减去另一行的一个倍数**。

这些操作都可以写成左乘一个可逆矩阵：

$$A \mapsto EA, \quad b \mapsto Eb,$$

因此

$$\boxed{Ax = b \iff EAx = Eb.}$$

高斯消元不是改变问题，而是在一串等价系统中逐步暴露三角结构。

设 σ 是 $\{1, \dots, n\}$ 上的一个置换。对应的置换矩阵 P_σ 由标准基向量重排得到：

$$P_\sigma = \begin{pmatrix} e_{\sigma(1)}^\top \\ e_{\sigma(2)}^\top \\ \vdots \\ e_{\sigma(n)}^\top \end{pmatrix}.$$

于是

$$\boxed{P_\sigma A}$$

就是把 A 的行按 σ 重排。例如

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} c & d \\ a & b \end{pmatrix}.$$

交换列则对应右乘置换矩阵。

例如要把第 j 行乘上非零常数 c ，可以左乘

$$D = \text{diag}(1, \dots, 1, \underset{j\text{th}}{c}, 1, \dots, 1).$$

于是

$$A \mapsto DA, \quad b \mapsto Db.$$

更一般地，

$$D = \text{diag}(c_1, \dots, c_n)$$

会把第 i 行乘上 c_i 。只要所有 $c_i \neq 0$ ，这个操作可逆。

要在第 j 行减去第 i 行的 c 倍，令

$$E = I - ce_j e_i^\top.$$

其中 $e_j e_i^\top$ 只有第 (j, i) 个元素为 1。那么

$$EA$$

正好完成这次行消元。这个操作也是可逆的：

$$(I + ce_j e_i^\top)(I - ce_j e_i^\top) = I \quad (i \neq j).$$

每一次“消去”都可以理解为左乘一个非常简单的可逆矩阵。

先假设不需要换行。高斯消元反复使用

$$R_j \leftarrow R_j - \ell_{ji} R_i$$

把 A 化成上三角矩阵 U 。若把所有消元矩阵合起来，

$$U = E_k \cdots E_1 A \implies \boxed{A = LU}, \quad L := (E_k \cdots E_1)^{-1}.$$

消元时出现的乘子 ℓ_{ji} 正好被记录在单位下三角矩阵 L 中；例如三阶情形的形状是

$$L = \begin{pmatrix} 1 & 0 & 0 \\ \ell_{21} & 1 & 0 \\ \ell_{31} & \ell_{32} & 1 \end{pmatrix}, \quad U \text{ 为上三角矩阵.}$$

求解时不必形成 A^{-1} ：

$$Ax = b \implies Ly = b, \quad Ux = y,$$

分别做一次前代和回代即可。**一次 LU 分解可以复用于多个不同的右端项 b 。**

选主元：为什么要交换行？

若当前主元为 0 或过小，就需要**选主元**。例如

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

的第一步必须先交换两行。

部分选主元（常用）

只交换行，并把所有行交换记录在置换矩阵 P 中：

$$PA = LU.$$

完全选主元

还允许交换列，并用置换矩阵 Q 记录列交换：

$$PAQ = LU.$$

若当前列在尚未消元的行中没有任何非零候选主元，则方阵必为奇异矩阵。

选主元既关系到“能不能继续消”，也关系到有限精度下的数值稳定性。

第 k 步要更新一个约 $(n - k) \times (n - k)$ 的尾部子矩阵, 因此

$$\sum_{r=1}^{n-1} O(r^2) = \boxed{O(n^3)}.$$

一旦已经得到 $PA = LU$, 新的右端项 b 只需要做两次三角求解:

$$Ly = Pb, \quad Ux = y,$$

每次只需

$$\boxed{O(n^2)}.$$

实际软件 高性能实现通常直接调用 **LAPACK** 中的 LU 分解例程; 核心更新会组织成块矩阵运算, 底层大量调用高度优化的 **BLAS**。

下一步: 算法能算出一个答案; 但这个答案对输入扰动有多敏感?

补充材料

DFT 约定 · 离散三角正交 · 三角最小二乘 · FFT 实现与应用

本讲采用

$$\hat{a}_k = \sum_{j=0}^{N-1} a_j \omega_N^{kj}, \quad \omega_N = e^{2\pi i/N},$$

因此

$$a_j = \frac{1}{N} \sum_{k=0}^{N-1} \hat{a}_k \omega_N^{-kj}.$$

不同教材常见的另外两种写法：

- ▶ 把负号放在正变换里、正号放在逆变换里；
- ▶ 在正、逆变换两边各放一个 $1/\sqrt{N}$ ，使傅里叶矩阵直接成为酉矩阵。

这些只是约定不同；只要正、逆变换彼此匹配，数学内容完全相同。

沿用主线的记号，取奇数个样本

$$M = 2m + 1, \quad x_j = \frac{2\pi j}{M}, \quad j = 0, \dots, M - 1,$$

并只使用到频率 $r \leq m$ 的实三角函数

$$1, \quad \cos(kx), \quad \sin(kx), \quad k = 1, \dots, r.$$

由单位根求和公式，不同频率之间的离散内积为 0；例如

$$\sum_{j=0}^{M-1} \cos(kx_j) \cos(\ell x_j) = 0 \quad (k \neq \ell),$$

以及对应的 sin-sin、sin-cos 正交关系。同时

$$\sum_{j=0}^{M-1} \cos^2(kx_j) = \sum_{j=0}^{M-1} \sin^2(kx_j) = \frac{M}{2}.$$

$r = m$ 就是第四讲的完整三角插值空间； $r < m$ 时则得到真正的超定最小二乘问题。

若改用偶数个等距样本，最高的奈奎斯特频率需要单独处理；奇数采样正好避开这个边界特例。

单位根求和：一次证明全部三角正交关系

若整数 q 不是 M 的倍数，则

$$\sum_{j=0}^{M-1} e^{iqx_j} = \sum_{j=0}^{M-1} \omega_M^{qj} = 0.$$

取实部和虚部立刻得到

$$\sum_{j=0}^{M-1} \cos(qx_j) = 0, \quad \sum_{j=0}^{M-1} \sin(qx_j) = 0.$$

再使用积化和差，例如

$$\cos u \cos v = \frac{1}{2}(\cos(u-v) + \cos(u+v)),$$

$$\sin u \sin v = \frac{1}{2}(\cos(u-v) - \cos(u+v)),$$

$$\sin u \cos v = \frac{1}{2}(\sin(u+v) + \sin(u-v)),$$

就得到上一页全部离散正交关系。**多个实三角正交恒等式，在复指数坐标下都来自同一个单位根几何级数。**

离散三角最小二乘：正交让法线方程完全解耦

给定 $M = 2m + 1$ 个等距实值数据 $y_j \in \mathbb{R}$ ，选一个拟合阶数 $r \leq m$ ，考虑

$$p(x) = c_0 + \sum_{k=1}^r (\alpha_k \cos(kx) + \beta_k \sin(kx))$$

并最小化

$$\sum_{j=0}^{M-1} |y_j - p(x_j)|^2.$$

由于离散正交，各个系数互不耦合：

$$c_0 = \frac{1}{M} \sum_{j=0}^{M-1} y_j,$$

$$\alpha_k = \frac{2}{M} \sum_{j=0}^{M-1} y_j \cos(kx_j), \quad \beta_k = \frac{2}{M} \sum_{j=0}^{M-1} y_j \sin(kx_j).$$

$r < m$ ：超定最小二乘； $r = m$ ：恰好退化为第四讲的三角插值。

对 $M = 2m + 1$ 个等距样本，完整的复指数展开写成

$$y_j = \sum_{k=-m}^m d_k e^{ikx_j}, \quad d_k = \frac{1}{M} \sum_{j=0}^{M-1} y_j e^{-ikx_j}.$$

按照本讲“正变换用正指数”的约定，若

$$\hat{y}_s := \sum_{j=0}^{M-1} y_j e^{isx_j},$$

则更精确地说

$$d_k = \frac{1}{M} \hat{y}_{(-k) \bmod M}.$$

对实值数据， $d_{-k} = \overline{d_k}$ ，且

$$\alpha_k = 2 \operatorname{Re} d_k, \quad \beta_k = -2 \operatorname{Im} d_k.$$

若只拟合到频率 $r < m$ ，只保留 $|k| \leq r$ 的这些正交坐标即可。**截断傅里叶坐标，就是把离散数据正交投影到低频三角子空间。**

- ▶ **大整数乘法：**把长整数按块看成多项式系数，用 FFT 做卷积可以把朴素二次复杂度的乘法显著加速。若采用复数 FFT，还必须控制舍入误差；在合适的分块与误差分析下，经典构造所需的工作精度只需随输入规模对数增长。
- ▶ **数字水印：**把信号变换到频率表示后，可以在选定频率分量中嵌入肉眼不易察觉的信息，再通过逆变换回到原表示。

整数乘法参考：Donald E. Knuth, *The Art of Computer Programming*, Vol. 2, §4.3.3.C.

共同的结构是“快速换表示”：多项式里是系数 $\leftrightarrow$ 单位根取值，信号里是时域 $\leftrightarrow$ 频域。

补充 长度不是 2 的幂怎么办？

本讲为了把递归结构看得最清楚，只推导了 **radix-2 FFT**：

$$N = 2^s.$$

但“FFT”不是只有这一种算法。

- ▶ 若 N 可以分解成较小因子，可以继续做 Cooley-Tukey 型的混合基数分解；
- ▶ 一般长度也可以通过 Bluestein 等方法转化成卷积，再调用一个合适长度的 FFT；
- ▶ 在多项式乘法中最简单：直接**补零到下一个 2 的幂**，不会改变所需的线性卷积结果。

所以：上一讲的奇数 $M = 2m + 1$ 并不意味着 DFT 不能快速计算；本讲主线只是选择了最清楚的二分递归情形。