# Numerical Analysis

Zhiping Li

LMAM and School of Mathematical Sciences
Peking University

# Fourier 变换— 一种重要的分析与计算工具

Fourier 变换是一种重要的分析与计算工具, 在科学工程计算中有广泛应用。

在数值逼近中, 不论在分析还是计算方面，利用三角多项式逼近周期函数与用普通多项式逼近一般函数有许多相通之处。

在微分方程中，利用 Fourier 变换常可将微分方程变换为代数方程，从而给分析带来方便。

许多高科技中应用到 Fourier 变换，例如, CT 扫描层析成像、数据压缩、图像处理、频谱分析，等等。

# Fourier 变换

**定义：** 设 $f(x) \in L^1(-\infty, +\infty)$, 则 $f(x)$ 的 *Fourier* 变换为

$$\hat{f}(k) = \int_{-\infty}^{+\infty} f(x) e^{-\mathrm{i}kx}\, dx.$$

这里 i 为虚数单位. 如果 $f(x) \in L^1(-\infty, +\infty) \cap L^2(-\infty, +\infty)$, 则 $\hat{f}(k) \in L^1(-\infty, +\infty) \cap L^2(-\infty, +\infty)$ 有 *Fourier* 逆变换

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \hat{f}(k) e^{\mathrm{i}kx}\, dk.$$

**注 1:** 也可引入更具对称性的定义 $\hat{f}(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} f(x) e^{-\mathrm{i}kx}\, dx$, $f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \hat{f}(k) e^{\mathrm{i}kx}\, dk$.

**注 2:** 将 Fourier 变换推广至以 $2\pi$ 为周期的周期函数 $f(x)$, 就是常用的 Fourier 级数 $f(x) = \sum_{-\infty}^{\infty} c_n \exp(-\mathrm{i}nx)$,

$c_n = \frac{1}{2\pi} \int_{-\pi}^{+\pi} f(x) e^{\mathrm{i}nx}\, dx$ (复数形式), 或 $f(x) = c_0 + \sum_{n=1}^{\infty} (a_n \cos(nx) + b_n \sin(nx))$ (实数形式).

# Fourier 变换的基本性质

1. 求导变系数： $\widehat{\left(\frac{df}{dx}\right)}(k) = \mathrm{i}k\hat{f}(k)$;
2. 平移性质： $\widehat{(f \circ T_a)}(k) = e^{\mathrm{i}ka}\hat{f}(k)$, 这里 $T_a(x) = x - a$ 是平移函数, $f \circ T_a(x) = f(x-a)$;
3. 卷积变乘积： $\widehat{(f * g)}(k) = \hat{f}(k)\hat{g}(k)$;
4. Parseval 等式： 设 $f(x) \in L^1(-\infty, +\infty) \cap L^2(-\infty, +\infty)$, 则

$$\int_{-\infty}^{+\infty} |f(x)|^2\, dx = \frac{1}{2\pi}\int_{-\infty}^{+\infty} |\hat{f}(k)|^2\, dk.$$

**注:** 在更具对称性的定义下, 相应的 Parseval 等式为 $\int_{-\infty}^{+\infty} |f(x)|^2\, dx = \int_{-\infty}^{+\infty} |\hat{f}(k)|^2\, dk$.

# 离散 Fourier 变换 ($2\pi$ 周期函数 Fourier 变换(级数)的离散化)

**定义：** 设向量 $\mathbf{a} = (a_0, a_1, \cdots, a_{N-1})^T$, 定义其离散 *Fourier* 变换为 $\mathbf{c} = (c_0, c_1, \cdots, c_{N-1})^T \triangleq \hat{\mathbf{a}}$, 其中

$$c_k = \sum_{j=0}^{N-1} a_j e^{-\mathrm{i}jk\frac{2\pi}{N}} = \sum_{j=0}^{N-1} a_j \omega^{jk}, \quad k = 0, 1, \cdots, N-1.$$

这里 $e^{-\mathrm{i}\frac{2\pi}{N}} \triangleq \omega$ 是 $N$ 次基本单位根.

**定理：** 向量 $\mathbf{a}$ 一定可由其离散 *Fourier* 变换 $\mathbf{c}$ 做离散 *Fourier* 逆变换得到, 记作 $\mathbf{a} = \check{\mathbf{c}}$, 即

$$a_j = \frac{1}{N} \sum_{k=0}^{N-1} c_k e^{\mathrm{i}jk\frac{2\pi}{N}} = \frac{1}{N} \sum_{k=0}^{N-1} c_k \omega^{-jk}, \quad j = 0, 1, \cdots, N-1.$$

# $\mathbf{a} = \check{\mathbf{c}}$ 的证明

定义 Fourier 矩阵

$$\mathbf{F} = (f_{kj})_{k,j=0}^{N-1} \triangleq (\omega^{kj})_{k,j=0}^{N-1} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega & \cdots & \omega^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \cdots & \omega^{(N-1)^2} \end{pmatrix}.$$

由定义知 $\mathbf{c} = \mathbf{F}\mathbf{a}$. $\mathbf{F}$ 是一个复对称 Vandermonde 行列式, 只需证明 $\mathbf{F}^{-1} = \mathbf{G}$ 为以下矩阵:

$$\mathbf{G} \triangleq \frac{1}{N}(\omega^{-jk})_{j,k=0}^{N-1} = \frac{1}{N} \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega^{-1} & \cdots & \omega^{-(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(N-1)} & \cdots & \omega^{-(N-1)^2} \end{pmatrix}.$$

## $\mathbf{a} = \check{\mathbf{c}}$ 的证明（续）

记 $\mathbf{X}_j = (1, \omega^j, \cdots, \omega^{(N-1)j})^T$, $\mathbf{Y}_k = (1, \omega^{-k}, \cdots, \omega^{-(N-1)k})^T$, 则有

$$\mathbf{X}_j^T \mathbf{Y}_k = \sum_{l=0}^{N-1} \omega^{(j-k)l} = \begin{cases} 0, & j \neq k, \\ N, & j = k. \end{cases}$$

事实上，当 $j = k$ 时上式显然, 而当 $j \neq k$ 时, 由 $\omega^N = 1$, 有

$$\sum_{l=0}^{N-1} \omega^{(j-k)l} = \frac{1 - \omega^{(j-k)N}}{1 - \omega^{j-k}} = \frac{1 - 1}{1 - \omega^{j-k}} = 0.$$

定理得证. $\blacksquare$

**注：** 以上分析表明 $\mathbf{G} = \mathbf{F}^{-1} = \frac{1}{N}\bar{\mathbf{F}}$, $\therefore$ $N\mathbf{F}^{-1} = \bar{\mathbf{F}} = \overline{\mathbf{F}^T} = \mathbf{F}^*$.

# 离散 Fourier 变换与多项式插值之间的关系

令 $P(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{N-1} x^{N-1}$, 则由 $\mathbf{c} = \mathbf{F}\mathbf{a}$, 记 $\mathbf{F}_k = (1, \omega^k, \cdots, \omega^{(N-1)k})$, 有

$$c_k = \mathbf{F}_k \mathbf{a} = P(\omega^k), \quad k = 0, 1, 2, \cdots, N-1.$$

即

- 求向量 $\mathbf{a} = (a_0, a_1, \cdots, a_{N-1})^T$ 的离散 Fourier 变换 $\hat{\mathbf{a}} = (c_0, c_1, \cdots, c_{N-1})^T$ 相当于求多项式 $P(x)$ 在 $\omega^0, \omega^1, \cdots, \omega^{N-1}$ 这 $N$ 个点上的值.
- 求 $(c_0, c_1, \cdots, c_{N-1})^T$ 的离散 Fourier 逆变换 $\mathbf{a} = \check{\mathbf{c}}$, 则相当于已知插值条件求多项式 $P(x)$ 的系数.

# 离散 Fourier 逆变换与多项式插值之间的关系

另一方面，令 $Q(x) = \frac{1}{N}(c_0 + c_1 x + \cdots + c_{N-1}x^{N-1})$，则由 $\mathbf{a} = \mathbf{F}^{-1}\mathbf{c}$，记 $\mathbf{F}_j^{-1} = \frac{1}{N}(1, \omega^{-j}, \cdots, \omega^{-(N-1)j})$，有

$$a_j = Q(\omega^{-j}), \quad j = 0, 1, 2, \cdots, N-1.$$

即

- 求向量 $(c_0, c_1, \cdots, c_{N-1})^T$ 的离散 Fourier 逆变换 $\mathbf{a} = \check{\mathbf{c}}$，相当于求多项式 $Q(x)$ 在 $\omega^0, \omega^{-1}, \cdots, \omega^{-(N-1)}$ 这 $N$ 个点上的值.
- 求 $\mathbf{a} = (a_0, a_1, \cdots, a_{N-1})^T$ 的离散 Fourier 变换 $\hat{\mathbf{a}} = (c_0, c_1, \cdots, c_{N-1})^T$ 相当于已知插值条件求多项式 $Q(x)$ 的系数.

# 离散 Fourier 变换与三角多项式插值之间的关系

对 $[0, 2\pi]$ 上的周期函数 $f(x)$, 给定其在点 $x_j = \frac{2\pi j}{N}$ 上的值 $a_j$, $j = 0, 1, 2, \cdots, N-1$, 做三角插值多项式

$$f_N(x) = \begin{cases} \sum_{k=-\frac{N}{2}+1}^{k=\frac{N}{2}} \tilde{c}_k e^{ikx}, & N \text{ 为偶数;} \\ \sum_{k=-\frac{N-1}{2}}^{k=\frac{N-1}{2}} \tilde{c}_k e^{ikx}, & N \text{ 为奇数,} \end{cases}$$

使得 $f_N(x_j) = a_j$, $j = 0, 1, 2, \cdots, N-1$. 令 $c_k = N\tilde{c}_k$, $0 \le k \le \left[\frac{N}{2}\right]$,

$$\begin{cases} c_{\frac{N}{2}+k} = N\tilde{c}_{-\frac{N}{2}+k}, & k = 1, 2, \cdots, \frac{N}{2}-1, \quad N \text{ 为偶数;} \\ c_{\frac{N-1}{2}+k} = N\tilde{c}_{-\frac{N-1}{2}+k}, & k = 1, 2, \cdots, \frac{N-1}{2}, \quad N \text{ 为奇数,} \end{cases}$$

则由 $\omega^N = 1$, 有 $e^{ikx_j} = \omega^{kj}$ 及 $a_j = f_N(x_j) = \frac{1}{N}\sum_{k=0}^{N-1} c_k\omega^{kj}$, 即 $\mathbf{a} = \check{\mathbf{c}}$. 因此, $\mathbf{c} = \hat{\mathbf{a}}$, 即三角插值多项式的系数 $\tilde{\mathbf{c}}$ 可由插值条件的离散 Fourier 变换方便地得到.

# 离散 Fourier 变换的卷积变乘积

**卷积变乘积**: 设 $f_l$ 关于指标是以 $N$ 为周期的, 即 $f_{-j} = f_{N-j}$, $\forall j$. 记 $(f * g)_l = \sum_{j=0}^{N-1} f_{l-j} g_j$, $l = 0, 1, 2, \cdots, N-1$. 则有

$$\widehat{(f * g)}_k = \hat{f}_k \hat{g}_k, \quad k = 0, 1, 2, \cdots, N-1.$$

**证明:** 由定义

$$\widehat{(f * g)}_k = \sum_{j=0}^{N-1} \sum_{l=0}^{N-1} f_{j-l} g_l e^{-\mathrm{i}jk\frac{2\pi}{N}},$$

$$\hat{f}_k \hat{g}_k = \sum_{j_2=0}^{N-1} f_{j_2} e^{-\mathrm{i}j_2 k\frac{2\pi}{N}} \sum_{l=0}^{N-1} g_l e^{-\mathrm{i}lk\frac{2\pi}{N}} = \sum_{j_2=0}^{N-1} \sum_{l=0}^{N-1} f_{j_2} g_l e^{-\mathrm{i}(j_2+l)k\frac{2\pi}{N}}.$$

# 离散 Fourier 变换的卷积变乘积证明（续）

交换第二式的求和顺序, 并对每个 $l$, 做内层求和指标变换

$$j = \begin{cases} j_2 + l, & 0 \le j_2 \le N-1-l; \\ j_2 + l - N, & N-l \le j_2 \le N-1, \end{cases}$$

则由 $f_l$ 的周期性得

$$\hat{f}_k \hat{g}_k = \sum_{l=0}^{N-1} \sum_{j_2=0}^{N-1} f_{j_2} g_l e^{-\mathrm{i}(j_2+l)k\frac{2\pi}{N}} = \sum_{l=0}^{N-1} \sum_{j=0}^{N-1} f_{j-l} g_l e^{-\mathrm{i}jk\frac{2\pi}{N}}.$$

此即 $\hat{f}_k \hat{g}_k = (\widehat{f * g})_k$. ■

# 离散 Fourier 变换的 Parseval 等式

Parseval 等式:
$$N\sum_{j=0}^{N-1}|a_j|^2=\sum_{k=0}^{N-1}|c_k|^2.$$

**证明：** 设 $\mathbf{F}$ 为 Fourier 矩阵. 由定义 $\mathbf{c}=\mathbf{F}\mathbf{a}$. 由于 Fourier 矩阵满足 $N\mathbf{F}^{-1}=\bar{\mathbf{F}}=\overline{\mathbf{F}^T}=\mathbf{F}^*$, 所以有

$$\sum_{k=0}^{N-1}|c_k|^2=\overline{\mathbf{c}^T}\mathbf{c}=\overline{\mathbf{a}^T}\,\overline{\mathbf{F}^T}\mathbf{F}\mathbf{a}=N\overline{\mathbf{a}^T}\mathbf{a}=N\sum_{j=0}^{N-1}|a_j|^2.$$

等式得证. ■

# 快速 Fourier 变换的基本思想

先看一个 $N = 4$ 的例子.

$$\mathbf{F} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -\mathrm{i} & -1 & \mathrm{i} \\ 1 & -1 & 1 & -1 \\ 1 & \mathrm{i} & -1 & -\mathrm{i} \end{pmatrix}, \quad \mathbf{Fa} = \begin{pmatrix} (a_0 + a_2) + (a_1 + a_3) \\ (a_0 - a_2) - \mathrm{i}(a_1 - a_3) \\ (a_0 + a_2) - (a_1 + a_3) \\ (a_0 - a_2) + \mathrm{i}(a_1 - a_3) \end{pmatrix}$$

- 直接计算需要 $N^2 = 16$ 次乘法和 $N(N-1) = 12$ 次加法;
- 而注意到 $\omega^{\frac{N}{4}} = -\mathrm{i}$, $\omega^{\frac{N}{2}} = -1$, $\omega^N = 1$, 并充分利用四则运算的结合律和分配律, 则只需 1 次乘法和 8 次加法.

(可以证明对 $N = 2^m$ 时, 总共需要不超过 $\frac{1}{2} N \log_2 N$ 次乘法和 $N \log_2 N$ 次加法.)

# 快速 Fourier 变换的基本思想

这种做法可以直接推广至 $N = 2^m$ 的情形，从而将计算量由 $O(N^2)$ 量级降低至 $O(N\log_2 N)$ 量级.

快速 Fourier 变换的基本思想: 记 $\omega_N = e^{-\mathrm{i}\frac{2\pi}{N}}$, 利用 $\omega_N^{2j} = \omega_{\frac{N}{2}}^{j}$, $\omega_N^{\frac{N}{2}+j} = -\omega_N^{j}$, $\omega_N^{N+2j} = \omega_{\frac{N}{2}}^{j}$, 和四则运算的结合律和分配律,

1. 尽量合并同类项以减少乘法运算;
2. 将变换各分量一并考虑, 尽量减少重复计算.

这种思想当然也可以推广至更一般的情形.

# 快速 Fourier 变换的基本算法

我们以 $N = 2^m$ 的情形为例来分析快速 Fourier 变换的基本算法.

记 $P(x) = a_0 + a_1 x^1 + \cdots + a_{N-1} x^{N-1}$, 注意到

$$\begin{aligned} P(x) &= (a_0 + a_2 x^2 + \cdots + a_{N-2} x^{N-2}) + x(a_1 + a_3 x^2 + \cdots + a_{N-1} x^{N-2}) \\ &= P_e(x^2) + x P_o(x^2). \end{aligned}$$

于是对 $j = 0, 1, \cdots, \frac{N}{2} - 1$, 有

$$\begin{cases} c_j = P(\omega_N^j) = P_e(\omega_N^{2j}) + \omega_N^j P_o(\omega_N^{2j}), \\ c_{\frac{N}{2}+j} = P(\omega_N^{\frac{N}{2}+j}) = P_e(\omega_N^{2(\frac{N}{2}+j)}) + \omega_N^{\frac{N}{2}+j} P_o(\omega_N^{2(\frac{N}{2}+j)}). \end{cases}$$

# 快速 Fourier 变换的基本算法（续 1）

由于 $P_e(\omega_N^{2j}) = P_e(\omega_{\frac{N}{2}}^j)$, $P_o(\omega_N^{2j}) = P_o(\omega_{\frac{N}{2}}^j)$, $j = 0, 1, \cdots, \frac{N}{2} - 1$,
则有

$$\begin{cases} c_j = P(\omega_N^j) = P_e(\omega_{\frac{N}{2}}^j) + \omega_N^j P_o(\omega_{\frac{N}{2}}^j), \\ c_{\frac{N}{2}+j} = P(\omega_N^{\frac{N}{2}+j}) = P_e(\omega_{\frac{N}{2}}^j) - \omega_N^j P_o(\omega_{\frac{N}{2}}^j), \end{cases} \qquad j = 0, 1, \cdots, \frac{N}{2} - 1.$$

同理 $P_e(x) = P_{ee}(x^2) + xP_{eo}(x^2)$, $P_o(x) = P_{oe}(x^2) + xP_{oo}(x^2)$,
(注意这里新指标排在老指标右侧, 即逆序). 于是有

$$\begin{cases} P_e(\omega_{\frac{N}{2}}^j) = P_{ee}(\omega_{\frac{N}{4}}^j) + \omega_{\frac{N}{2}}^j P_{eo}(\omega_{\frac{N}{4}}^j), \\ P_o(\omega_{\frac{N}{2}}^j) = P_{oe}(\omega_{\frac{N}{4}}^j) - \omega_{\frac{N}{2}}^j P_{oo}(\omega_{\frac{N}{4}}^j), \end{cases} \qquad j = 0, 1, \cdots, \frac{N}{4} - 1.$$

# 快速 Fourier 变换的基本算法（续 2）

如此递推至 $N=2^m$ 个 0 次多项式在 1 点的取值. 显然，最底层的每个 0 次多项式在 1 点的取值即相应的系数值，无需做任何运算, 即需 0 次乘法和 0 次加法.

现设计算 $\frac{N}{2}=2^{m-1}$ 的 Fourier 变换总共需要 $M_{\frac{N}{2}}$ 次乘法和 $A_{\frac{N}{2}}$ 次加法, 则由 $N=2^m$ 的 Fourier 变换与 $\frac{N}{2}=2^{m-1}$ 的 Fourier 变换之间的关系式知（需要算一奇一偶两个 $\frac{N}{2}=2^{m-1}$ 的 Fourier 变换，外加 $\frac{N}{2}$ 次乘法和 $N$ 次加法）, 因此有

$$M_N=2M_{\frac{N}{2}}+\frac{N}{2},\quad A_N=2A_{\frac{N}{2}}+N.$$

归纳得 $M_N=2^mM_{\frac{N}{2^m}}+m\frac{N}{2}=m\frac{N}{2}$, $A_N=2^mA_{\frac{N}{2^m}}+mN=mN$.
即 $N=2^m$ 时, 总共需要 $\frac{1}{2}N\log_2 N$ 次乘法和 $N\log_2 N$ 次加法.

# 快速 Fourier 变换基本算法的实现

设 $N=2^m$, 由以上分析知，FFT 的实现主要分两个步骤:

1. **分割**: 将第 $m$ 层的 $2^m$ 维向量 $\mathbf{a}$ 分割成 $\mathbf{a}_e$, $\mathbf{a}_o$, 奇偶两个第 $m-1$ 层的 $2^{m-2}$ 维向量, 再将它们每一个分割成奇偶两个第 $m-2$ 层的 $2^{m-2}$ 维向量 $\mathbf{a}_{ee}$, $\mathbf{a}_{eo}$, $\mathbf{a}_{oe}$, $\mathbf{a}_{oo}$, (共 $2^2$ 个, 指标按逆序重排), $\cdots$, 直至第 0 层分割成逆序的 $2^m$ 个数;
2. **组装**: 设第 $k$ 层的一个向量 $\mathbf{a}_\delta^k$ 分解成 $\mathbf{a}_{\delta e}^{k-1}$ 和 $\mathbf{a}_{\delta o}^{k-1}$ 两个 $k-1$ 层的向量, 记它们的 Fourier 变换为 $\mathbf{c}_{\delta e}^{k-1}$ 和 $\mathbf{c}_{\delta o}^{k-1}$, 记 $K=2^k$, $\omega_K=e^{-\mathrm{i}\frac{2\pi}{K}}$, $\boldsymbol{\omega}_K=(\omega_K^0,\cdots,\omega_K^{\frac{K}{2}-1})^T$, 则有

$$\mathbf{c}_\delta^k=\begin{pmatrix}\mathbf{c}_{\delta e}^{k-1}+\boldsymbol{\omega}_{2^k}\circ\mathbf{c}_{\delta o}^{k-1}\\ \mathbf{c}_{\delta e}^{k-1}-\boldsymbol{\omega}_{2^k}\circ\mathbf{c}_{\delta o}^{k-1}\end{pmatrix},\quad \forall\delta, k=1,\cdots,m,$$

其中 $\circ:\mathbb{R}^{\frac{K}{2}}\times\mathbb{R}^{\frac{K}{2}}\to\mathbb{R}^{\frac{K}{2}}$, $\mathbf{x}\circ\mathbf{y}=(x_0y_0,x_1y_1,\cdots,x_{\frac{K}{2}-1}y_{\frac{K}{2}-1})^T$.

# 快速 Fourier 变换基本算法的实现

由组装的方式可以看出, FFT 算法实现的关键在于将 $N = 2^m$ 个数据 **a** 给出一个第 0 层的排序方法, 使得:

- 将所排数据( $N = 2^m$ 个 $2^0$ 维向量)顺序两两成对组装得到第 1 层的 $N/2 = 2^{m-1}$ 个顺序排列的二维（$2^1$）向量;
- 将顺序排列的 $2^{m-1}$ 个二维（$2^1$）向量顺序两两成对组装得到第 2 层顺序排列的 $2^{m-2}$ 个四维（$2^2$）向量;
- $\cdots\cdots$ $\cdots\cdots$ $\cdots\cdots$;
- 将顺序排列的两个 $2^{m-1}$ 维向量顺序两两成对组装得到第 $m$ 层顺序排列的 $2^m$ 维向量, 而最终得到的向量正是我们要求的所给数据的 Fourier 变换。

# 快速 Fourier 变换基本算法的实现

我们以 $N = 2^3$ 为例, 来观察排序的规律. 设 $\mathbf{a} = (a_0, a_1, \cdots, a_7)^T$,

1. **第一步分割：** $\mathbf{a}_e = (a_0, a_2, a_4, a_6)^T$, $\mathbf{a}_o = (a_1, a_3, a_5, a_7)^T$;
2. **第二步分割：** $\mathbf{a}_{ee} = (a_0, a_4)^T$, $\mathbf{a}_{eo} = (a_2, a_6)^T$, $\mathbf{a}_{oe} = (a_1, a_5)^T$, $\mathbf{a}_{oo} = (a_3, a_7)^T$;
3. **第三步分割：** $\mathbf{a}_{eee} = a_0$, $\mathbf{a}_{eeo} = a_4$, $\mathbf{a}_{eoe} = a_2$, $\mathbf{a}_{eoo} = a_6$, $\mathbf{a}_{oee} = a_1$, $\mathbf{a}_{oeo} = a_5$, $\mathbf{a}_{ooe} = a_3$, $\mathbf{a}_{ooo} = a_7$.

这种逆序 $(a_0, a_4, a_2, a_6, a_1, a_5, a_3, a_7)^T$, 即为满足要求的重新排序. 一般地，对任意的 $N = 2^m$, 逆序就是我们需要的重新排序.

# 快速 Fourier 变换基本算法的实现

事实上, 把每次分割的指标做重新表示，将 $e$ 换为 0, 将 $o$ 换为 1, 则分割结果的顺序为 $\mathbf{a}_{000} = a_0$，$\mathbf{a}_{001} = a_4$，$\mathbf{a}_{010} = a_2$，$\mathbf{a}_{011} = a_6$，$\mathbf{a}_{100} = a_1$，$\mathbf{a}_{101} = a_5$，$\mathbf{a}_{110} = a_3$，$\mathbf{a}_{111} = a_7$，而这正是按二进制中的逆序排的序.

$$
\begin{array}{lcl}
0 = 000_2; & & 000_2 = 0; \\
1 = 001_2; & & 100_2 = 4; \\
2 = 010_2; & & 010_2 = 2; \\
3 = 011_2; & \text{逆序} & 110_2 = 6; \\
4 = 100_2; & \longrightarrow & 001_2 = 1; \\
5 = 101_2; & & 101_2 = 5; \\
6 = 110_2; & & 011_2 = 3; \\
7 = 111_2; & & 111_2 = 7.
\end{array}
$$

# 快速 Fourier 变换基本算法的实现

组装过程是从 $\mathbf{a}$ 的逆序按最新指标的偶 0 奇 1(前面的指标相同)配对计算产生的，每组装一次, 子向量个数减少一半, 维数增一倍, 组装 $m$ 次后得到 Fourier 变换 $\mathbf{c}$ (一个 $N = 2^m$ 维的向量).

$$
\begin{array}{llcccccc}
a_0, & \mathbf{a}_{000}; & & & & & & \\
a_4, & \mathbf{a}_{001}; & & \mathbf{c}_{00}; & & & & \\
a_2, & \mathbf{a}_{010}; & & & & \mathbf{c}_{0}; & & \\
a_6, & \mathbf{a}_{011}; & \pm & \mathbf{c}_{01}; & \pm & & \pm & \\
 & & \boldsymbol{\omega}_2\circ & & \boldsymbol{\omega}_4\circ & & \boldsymbol{\omega}_8\circ & \mathbf{c}. \\
a_1, & \mathbf{a}_{100}; & \longrightarrow & \mathbf{c}_{10}; & \longrightarrow & & \longrightarrow & \\
a_5, & \mathbf{a}_{101}; & & & & \mathbf{c}_{1}; & & \\
a_3, & \mathbf{a}_{110}; & & \mathbf{c}_{11}; & & & & \\
a_7, & \mathbf{a}_{111}; & & & & & &
\end{array}
$$

其中 $\omega_K = e^{-\mathrm{i}\frac{2\pi}{K}}$, $\boldsymbol{\omega}_K = (\omega_K^0, \cdots, \omega_K^{\frac{K}{2}-1})^T$, $\mathbf{x} \circ \mathbf{y} = (x_0 y_0, x_1 y_1, \cdots, x_{\frac{K}{2}-1} y_{\frac{K}{2}-1})^T$.

习题五：1，2，3；  上机习题五：2，3.

# Thank You!