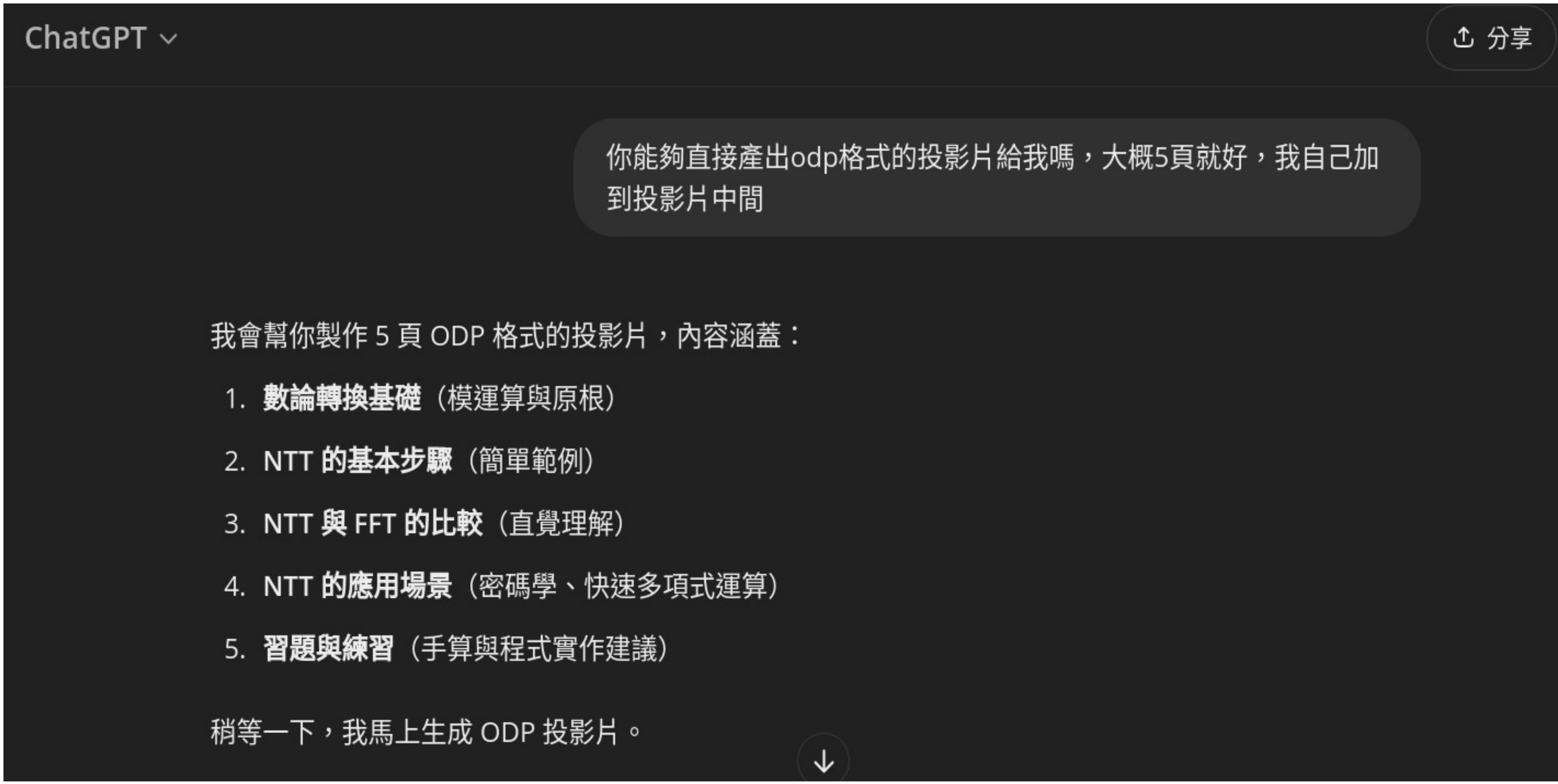
數論轉換介紹

Number Theoretic Transform (NTT)

2025/04/09

請 gpt 幫我生投影片


ChatGPT
分享
你能夠直接產出odp格式的投影片給我嗎，大概5頁就好，我自己加到投影片中間
我會幫你製作 5 頁 ODP 格式的投影片，內容涵蓋：
1. **數論轉換基礎**（模運算與原根）
2. NTT **的基本步驟**（簡單範例）
3. NTT **與** FFT **的比較**（直覺理解）
4. NTT **的應用場景**（密碼學、快速多項式運算）
5. **習題與練習**（手算與程式實作建議）
稍等一下，我馬上生成 ODP 投影片。

數論轉換補充投影片

補充基礎概念、範例與應用

數論轉換基礎

- - 模運算： $a \equiv b \pmod{m}$
- - 原根的概念：找到一個數 g ，使得 g^k mod m 遍歷所有非零元素
- - 選取適當的質數 p ，使得 p-1 有較小的質因數

NTT 的基本步驟

- 1. 選擇適當的模數 p 和原根 g
- 2. 建立 NTT 變換矩陣
- 3. 計算 NTT ，類似 FFT 的蝶形運算
- 4. 進行逆變換來還原原始數據

NTT 與 FFT 的比較

- - FFT 使用浮點數， NTT 使用整數模運算
- - NTT 不受浮點誤差影響，適用於精確計算
- - NTT 適用於密碼學與計算機代數

NTT 的應用場景

- - 密碼學（如同態加密）
- - 多項式乘法加速
- - 競程（如大數計算與快速卷積）

習題與練習

- 1. 計算 3^4 mod 7 是多少？
- 2. 找到 7 的一個原根
- 3. 手算一個簡單的 NTT 變換
- 4. 嘗試用 Python/Octave 實作簡單的 NTT

XIV. Number Theoretic Transform (NTT)

• 14-A Definition

Number Theoretic Transform and Its Inverse

$$F(k) = \sum_{n=0}^{N-1} f(n)\alpha^{nk} \pmod{M},\ k = 0, 1, 2\cdots, N-1$$

$$f(n) = N^{-1}\sum_{k=0}^{N-1} F(k)\alpha^{-nk} \pmod{M} \quad ,n = 0, 1, 2\cdots, N-1 \qquad f(n) \underset{INTT}{\overset{NTT}{\Longleftrightarrow}} F(k)$$

Note ：

因為希望有 inverse

(1) M is a prime number , (mod M): 是指除以 M 的餘數

(2) N is a factor of $M-1$

(Note: when $N \neq 1$, N must be relatively prime to M)

(3) N^{-1} is an integer that satisfies $(N^{-1})N \bmod M = 1$

(When $N = M-1$, $N^{-1} = M-1$), it is also called the inverse of N (mod M)

(4) α is a root of unity of order N

$$\alpha^N = 1 \pmod{M}$$

$$\alpha^k \neq 1 \pmod{M},\ k = 1, 2, \cdots, N-1$$

When α satisfies the above equations and $N = M - 1$, we call α the "primitive root".

$$\alpha^k \neq 1 \pmod{M} \quad \text{for } k = 1, 2, \cdots, M-2$$

$$\alpha^{M-1} = 1 \pmod{M}$$

沒有可以快速球出 primitive root 的方法
>> 只能慢慢算

所以找 primitive root 來建立 transform
才會只用一些比較小的數字

α^{-1} 的求法與 N^{-1} 相似

α^{-1} is an integer that satisfies $\boxed{(\alpha^{-1})\alpha \bmod M = 1}$

Example 1:

$M = 5 \quad \alpha = 2 \quad \alpha^1 = 2 \pmod 5 \quad \alpha^2 = 4 \pmod 5 \quad \alpha^3 = 3 \pmod 5 \quad \alpha^4 = 1 \pmod 5$

和 DFT 相同，都是 1

(1) When $N = 4$

可以用 N 的因數做

在 5 的世界，最多可以做到四點的 length

$2^2 \pmod 5$ $\quad 2^3 \pmod 5$ $\quad 2^1 \pmod 5$

$$\begin{bmatrix} F[0] \\ F[1] \\ F[2] \\ F[3] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 3 \\ 1 & 4 & 1 & 4 \\ 1 & 3 & 4 & 2 \end{bmatrix} \begin{bmatrix} f[0] \\ f[1] \\ f[2] \\ f[3] \end{bmatrix}$$

和 DFT 是相同的

(2) When $N = 2$

可以用比 N 小的數字做，primitive root 只是給一個上界（最多可以做到多少）

$$\begin{bmatrix} F[0] \\ F[1] \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} f[0] \\ f[1] \end{bmatrix}$$

- N：取幾次方以後 mod 完會 = 1

Example 2:

$M = 7$, $N = 6$: α cannot be 2 but can be 3.

$\alpha = 2$: $\alpha^1 = 2 \pmod 7$ $\alpha^2 = 4 \pmod 7$ $\alpha^3 = 1 \pmod 7$

$\alpha = 3$: $\alpha^1 = 3 \pmod 7$ $\alpha^2 = 2 \pmod 7$ $\alpha^3 = 6 \pmod 7$

$\alpha^4 = 4 \pmod 7$ $\alpha^5 = 5 \pmod 7$ $\alpha^6 = 1 \pmod 7$

Advantages of the NTT:

1. 使用整數，比較容易運算
2. 使用整數無誤差
3. 很多性質都跟 DFT 相同

(幾乎完美的複製體)

Disadvantages of the NTT:

1. 只能對整數做轉換
2. 不容易找到 root of unity
3. 較缺乏物理意義

（1） DFT 的 basis 是三角函數的波

（2） DFT 的特徵向量仍像是 Hermite-Gaussian 函數，這個性質讓 DFT 可以做 fractional ，但是 NTT 特徵向量沒有這樣的性質

• 14-B 關於餘數我們應該要知道的性質

(1) $x \pmod{M}$ 的值，必定為 $0 \sim M-1$ 之間

(2) $a + b \pmod{M} = \{a \pmod{M} + b \pmod{M}\} \pmod{M}$

例： $78 + 123 \pmod{5} = 3 + 3 \pmod{5} = 1$

(Proof): If $a = a_1M + a_2$ and $b = b_1M + b_2$, then

$$a + b = (a_1 + b_1)M + a_2 + b_2$$

(3) $a \times b \pmod{M} = \{a \pmod{M} \times b \pmod{M}\} \pmod{M}$

例： $78 \times 123 \pmod{5} = 3 \times 3 \pmod{5} = 4$

(Proof): If $a = a_1M + a_2$ and $b = b_1M + b_2$, then

$$a \times b = (a_1 b_1M + a_1b_2 + a_2b_1)M + a_2b_2$$

(4) 給定 a ，如何計算出 a^{-1} mod M?

答案：

1. 首先前面說 M 是質數，表示 gcd(a,M)=1

2. Bezout’s Lemma 說 if gcd(a,b)=d, 則存在 s,t 使得 as+bt=d

3. 所以存在 as+Mt=1 ，兩邊 mod M 後，就可以得到 s 就是 a 的 inverse

至於怎麼求出 s,t ，就是要使用 the Extended Euclidean Algorithm or Blankinship’s Method ，中文叫做輾轉相除法。

原理：

1. Euclidean Algorithm 的原理是若 a=bq+r 則 gcd(a,b)=gcd(b,r)

2. Extended Euclidean Algorithm 就是把求 gcd 的過程紀錄下來
然後一路反推

Example 1.10.1

For our example, we will continue with the numbers used in Example 1.8.1 from Section 1.8. There we found that $\gcd(803, 154) = 11$. Bezout's Lemma guarantees that $11 = 803s + 154t$ for some integers s, t. Let us determine a possible choice of s and t.

In Example 1.8.1 we found the following:

$$\begin{aligned} 803 &= 154 \cdot 5 + 33 \\ 154 &= 33 \cdot 4 + 22 \\ 33 &= 22 \cdot 1 + 11 \\ 22 &= 11 \cdot 2 + 0. \end{aligned}$$

Rearranging and reordering all but the last of these equations, we get

$$\begin{aligned} 11 &= 33 - 22 \cdot 1 \\ 22 &= 154 - 33 \cdot 4 \\ 33 &= 803 - 154 \cdot 5 \end{aligned}$$

We now set about to find an expression of 11 as a linear combination of 803 and 154. Starting with the equation $11 = 33 - 22 \cdot 1$, we substitute $22 = 154 - 33 \cdot 4$ (the next equation) to obtain

$$\begin{aligned} 11 &= 33 - (154 - 33 \cdot 4) \cdot 1 \\ &= 5 \cdot 33 - 1 \cdot 154. \end{aligned}$$

Now we will substitute for 33, using the equation $33 = 803 - 154 \cdot 5$ above:

$$1 = 5 \cdot (803 - 154 \cdot 5) - 1 \cdot 154$$

and finally obtain

$$\boxed{11 = 5 \cdot 803 - 26 \cdot 154.}$$

請求出 47 在 101 的 inverse

1. $101 = 47 \times 2 + 7$
2. $47 = 7 \times 6 + 5$
3. $7 = 5 + 2$
4. $5 = 2 \times 2 + 1$

1. $5 - 2 \times 2 = 1, \because 5 = 2 \times 2 + 1$
2. $5 - (7 - 5) \times 2 = 1, \because 7 = 5 + 2$
3. $5 \times 3 - 7 \times 2 = 1$
4. $(47 - 7 \times 6) \times 3 - 7 \times 2 = 1, \because 47 = 7 \times 6 + 5$
5. $47 \times 3 - 7 \times 20 = 1$
6. $47 \times 3 - (101 - 47 \times 2) \times 20 = 1, \because 101 = 47 \times 2 + 7$
7. $47 \times 43 - 101 \times 20 = 1$ 答案 43 即爲所求

(5) 給定 a ，如何計算出 $a^n \bmod M$?

動機：回憶一下 NTT 的定義

$$F(k) = \sum_{n=0}^{N-1} f(n)\alpha^{nk} \pmod{M},\ k = 0, 1, 2\cdots, N-1$$

$$f(n) = N^{-1}\sum_{k=0}^{N-1} F(k)\alpha^{-nk} \pmod{M} \quad ,n = 0, 1, 2\cdots, N-1$$

$$f(n) \underset{INTT}{\overset{NTT}{\Longleftrightarrow}} F(k)$$

答案：

1. 乍聽之下這是很蠢的問題，不就是先把 a^n 算出來，然後 mod M
2. 但是我們真的有辦法把 a^n 算出來嗎？當 a 和 n 都不小的時候，電腦會爆
3. 舉例： 37^{89} mod 101

ex: a^{16} = (a^8)^2 = ((a^4)^2)^2 = (((a^2)^2)^2)^2
(16)_{10} = (10000)_2

所以實際上，我們先把指數分解成 binary

然後看到 0 就平方

看到 1 就平方後再乘上 37

```
z=37;n=89;N=101;
x=dec2bin(n);  x = 1011001
   t=1;
   for k=1:length(x)   for k = 1:7
     if x(k)=='0'
       t=mod(t.^2,N);
     else
       t=mod((t.^2).*z,N);
     endif
   endfor
 y=t;
```

答案的後續：

1. 看起來已經有個 happy ending 了，趕快來試試看 37^{140548} mod 101

BUT!

還有一個更快的高招，費馬小定理

Theorem 1.24.2: Fermat's Little Theorem

If p is prime and a is relatively prime to p then

$$a^{p-1} \equiv 1 \pmod{p}.$$

範例：

Ex1: p=5,a=2, $2^4=16=1$ (mod 5)

Ex2: p=13, a=3, $3^2=9$, $3^3=27=1$ (mod 13) 所以 $3^{12}=(3^3)^4=1^4=1$ mod 13

費馬小定理說 3^{12} 是 1，不代表 3 的 1~11 次方之間不可能有 mod 13 同餘 1 的
>> 代表 3 並非 primitive root

primitive root: 只有在 12 次方的時候才會同餘 1，在 12 次方之前都不會同餘 1

如何使用費馬小定理更快算出 37^{140548} mod 101 呢

首先我們知道 37^{100}=1 mod 101

所以 37^{140500}=1 mod 101

也就是說

別人還在想辦法把 140548 變成 2 進位

你卻只要算 48 的 2 進位就好

(6) 既然講到費馬小定理，就補充兩個數論的定理吧

1. 中國剩餘定理

2. Wilson 定理

Theorem 1.23.1: Chinese Remainder Theorem

Let $m_1, m_2, \ldots, m_k$ be natural numbers such that each is greater than 1, and every pair of them is relatively prime. Let $M = m_1 m_2 \cdots m_k$, and let $b_1, b_2, \ldots, b_k$ be integers. The system of congruences

$$\begin{aligned} x &\equiv b_1 \pmod{m_1}; \\ x &\equiv b_2 \pmod{m_2}; \\ &\vdots \\ x &\equiv b_k \pmod{m_k}; \end{aligned}$$

has a unique solution in the set $\{0, 1, 2, \ldots, M-1\}$.

Theorem 1.24.1: Wilson's Theorem

If p is a prime, then

$$(p-1)! \equiv -1 \pmod{p}.$$

中國剩餘定理實戰

Q: 一個數介於 0~34 之間，被 5 除餘 2 ，被 7 除則餘 3

請問某數為多少

A: 分開處理，

甲：先找到一個數被 5 除餘 0 ，被 7 除餘 3

乙：再找一個數，被 5 除餘 2 ，被 7 除餘 0

兩者相加即為所求

甲如何達成？既然被 5 整除，叫它 5K 吧。又知道 5 在 mod 7 的 inverse 是 3 (3x5=15=1 mod 7) ，所以如果要被 7 除餘 3 的話，就再乘以 3 即可，換句話說， 5x3x3=45 就是 5 的倍數，且 mod 7 為 3

正規化，變成餘 1

但我們要餘 3 ，所以再乘 3

同理可以得到乙應該是 3x7x2=42 (3^{-1} =7 mod 5)

兩者相加為 87 =17 mod 35 ，所以答案是 17

中國剩餘定理實戰

另外一種解法

	0	1	2	3	4
0	0		7		
1		1		8	
2			2		9
3	10			3	
4		11			4
5	5		12		
6		6		13	

中國剩餘定理實戰

Q: 超過兩個數怎麼辦？一個數被 3 除餘 2 ，被 5 除餘 1 ，被 7 除餘 4 ，答案在 0~104

A: 一樣分開處理

35a+21b+15c=2 mod 3

35a+21b+15c=1 mod 5

35a+21b+15c=4 mod 7

=>

2a=2

b=1

c=4

35x1+21x1+15x4=116=11 mod 105 ，所以答案是 11

Wilson 定理，與它的用途

（中國剩餘定理的用途太廣，大家自己搜尋）

以 p=11 為例 10!=10x9x8x...x1=10= -1 mod 11

如何計算 2^{-1}

可以這樣想：(p-2)!=1, (p-2)x(p-3)!=1, 所以 (p-3)! 就是 -2 的 inverse

所以 2 的 inverse 就是 -(p-3)!

	1	2	3	4	5	6	7	8	9	10
!	1	2	6	2	10	5	2	5	1	10
inv	1	-5							5	10

如何計算 3^{-1}

可以這樣想：(p-2)!=1, (p-2)x(p-3)!=1, (p-2)x(p-3)x(p-4)!=1

所以 (p-3) 的 inverse 就是 (p-2)x(p-4)!

所以 3 的 inverse 就是 -(p-2)x(p-4)!

如何計算 4^{-1}

(p-2)(p-3)(p-4)x(p-5)!=1，所以 3!x(p-5)! 就是 (p-4) 的 inverse

….

簡言之，可以透過計算階層，利用 O(N) 的乘法，算出所有的 inverse

7	1	2	3	4	5	6	7	8	9	10
!	1	2	6	2	10	5	2	5	1	10
inv	1	-5	-7				8	7	5	10

小結：

(1) 如果只求某個特定數的 inverse ，請使用輾轉相除法

(2) 如果想求所有的 inverse ，可使用 Wilson 定理

思考一下，輾轉相除法是 O(?)

另外，如果做 NTT ，只需要乘法就好

那麼由於 mod M 中，只有 M^2 個可能的加法， M^2 個可能的乘法

可事先將加法和乘法的結果存在記憶體當中

需要時再 "LUT"

LUT : lookup table

• 14-C Properties of Number Theoretic Transforms

P.1) Orthogonality Principle

$$S_N = \sum_{n=0}^{N-1} \alpha^{nk} \alpha^{-n\ell} = \sum_{n=0}^{N-1} \alpha^{n(k-\ell)} = N \cdot \delta_{k.\ell}$$

proof ： for $k = \ell$, $S_N = \sum_{n=0}^{N-1} \alpha^{0} = N$

for $k \neq 0$, $(\alpha^{k-\ell l} - 1)\, S_N = (\alpha^{k-\ell} - 1) \sum_{n=0}^{N-1} \alpha^{n(k-\ell)} = \alpha^{N(k-\ell)} - 1 = 1 - 1 = 0$

$\because \alpha^{k-\ell} \neq 1 \qquad \therefore S_N = 0$

P.2) The NTT and INTT are exact inverse

proof ： $g(n) = \frac{1}{N} \sum_{k=0}^{N-1} F(k)\, \alpha^{-nk} = \frac{1}{N} \sum_{k=0}^{N-1} \left(\sum_{\ell=0}^{N-1} f(\ell)\, \alpha^{\ell k} \right) \alpha^{-nk}$

$$= \frac{1}{N} \sum_{\ell=0}^{N-1} f(\ell) \sum_{k=0}^{N-1} \alpha^{(\ell-n)k} = \frac{1}{N} \sum_{\ell=0}^{N-1} f(\ell) \cdot N \delta_{\ell,n} = f(n)$$

P.3) Symmetry

$$f(n) = f(N-n) \overset{\text{NTT}}{\Leftrightarrow} F(k) = F(N-k)$$

$$f(n) = -f(N-n) \overset{\text{NTT}}{\Leftrightarrow} F(k) = -F(N-k)$$

P.4) INNT from NTT

$$f(n) = \frac{1}{N}\sum_{k=0}^{N-1} F(k)\,\alpha^{-nk} = \frac{1}{N}\sum_{(-k)=0}^{N-1} F(-k)\,\alpha^{nk} = NTT\ of\ \frac{1}{N}F(-k)$$

Algorithm for calculating the INNT from the NTT

(1) $F(-k)$: time reverse

$$F_0, F_1, F_2, \ldots, F_{N\text{-}1} \xrightarrow[\text{reverse}]{\text{time}} F_0, F_{N\text{-}1}, \ldots, F_2, F_1$$

(2) NTT[$F(-k)$]

(3) 乘上 N^{-1}

P.5) Shift Theorem

$$f(n+\ell) \leftrightarrow F(k)\,\alpha^{-\ell k}$$

$$f(n)\,\alpha^{n\ell} \leftrightarrow F(k+\ell)$$

P.6) Parseval's Theorem

$$N\sum_{n=0}^{N-1} f(n)\, f(-n) = \sum_{k=0}^{N-1} F^2(k)$$

$$N\sum_{n=0}^{N-1} f(n)^2 = \sum_{k=0}^{N-1} F(k)F(-k)$$

P.7) Linearity

$$a\, f(n) + b\, g(n) \leftrightarrow a\, F(k) + b\, G(k)$$

P.8) Reflection

If $f(n) \leftrightarrow F(k)$ then $f(-n) \leftrightarrow F(-k)$

P.9) Circular Convolution (the same as that of the DFT)

If $f(n) \leftrightarrow F(k)$

$g(n) \leftrightarrow G(k)$

then $f(n) \otimes g(n) \leftrightarrow F(k)G(k)$

i.e., $f(n) \otimes g(n) = INTT\left\{ NTT\left[f(n) \right] NTT\left[g(n) \right] \right\}$

$$f(n) \cdot g(n) \leftrightarrow \frac{1}{N} F(k) \otimes G(k)$$

(Proof): $INNT\big(NNT(f[n])NNT(g[n])\big) = N^{-1}\sum_{k=0}^{N-1} \alpha^{-nk} F(k)G(k)$

$$= N^{-1}\sum_{k=0}^{N-1} \alpha^{-nk} \sum_{m=0}^{N-1} f[m]\alpha^{mk} \sum_{q=0}^{N-1} g[q]\alpha^{qk}$$

$$= \sum_{m=0}^{N-1}\sum_{q=0}^{N-1} f[m]g[q] N^{-1}\sum_{k=0}^{N-1} \alpha^{-nk}\alpha^{mk}\alpha^{qk}$$

$$= \sum_{m=0}^{N-1}\sum_{q=0}^{N-1} f[m]g[q]\delta\left[((m+q-n))_N\right]$$

$$= \sum_{m=0}^{N-1}\sum_{q=0}^{N-1} f[m]g[((n-m))_N] = f[n] \otimes g[n]$$

We apply the fact that

$$\sum_{k=0}^{N-1} \alpha^{nk} = \begin{cases} 1 & \text{if } n \text{ is a multiple of } N \\ 0 & \text{otherwise} \end{cases}$$

When $q = ((n-m))_N$

$m+q-n$ is a multiple of N

• 14-D Complex Number Theoretic Transform (CNT)

可以找稍微小一點的數字，就能有比較大的 length

The integer field Z_M can be extended to complex integer field

If the following equation does not have a sol. in Z_M

$$x^2 = -1 \pmod{M}$$ 無解

This means (-1) does not have a square root

質數分兩種，4k+1 和 4k+3 的

When $\boxed{M = 4k+1}$, there is a solution for $x^2 = -1 \pmod{M}$.

When $\boxed{M = 4k+3}$, there is no solution for $x^2 = -1 \pmod{M}$.

For example, when $M = 13$, $8^2 = -1 \pmod{13}$.

$2^1 = 2$, $2^2 = 4$, $2^3 = 8$, $2^4 = 3$, $2^5 = 6$, $2^6 = 12 = -1$,

$2^7 = 11$, $2^8 = 9$, $2^9 = 5$, $2^{10} = 10$, $2^{11} = 7$, $2^{12} = 1$

把 1~12 的平方都寫出來，
直接找 x^2 = –1 (mod M) 的解

When $M = 11$, there is no solution for $x^2 = -1 \pmod{M}$.

M = 4k + 3

If there is no solution for $x^2 = -1 \pmod{M}$, we can define an imaginary number i such that

$$\boxed{i^2 = -1 \pmod{M}}$$

ex: M = 7 = 4 * 1 + 3
>> 會有 48 個數字可用（7 * 7 扣掉 0）
>> 在小小的 M (M = 7) 就可以做長度為 48 的 NTT

Then, "i" will play a similar role over finite field Z_M such that plays over the complex field.

$$(a+i\,b) \pm (c+i\,d) = (a \pm c) + i\,(b \pm d)$$

$$(a+i\,b)\cdot(c+i\,d) = ac + i^2 bd + i\,bc + i\,ad$$

$$= (ac - bd) + i\,(bc + ad)$$

• 14-E Legendre sequence and Gauss sum

Legendre sequence 只有 0, 1, –1 三種數字，長度為 p ，開頭一定是 0，
接下來如果是平方數則為 1，不是平方數就是 –1

ChatGPT

MU

Legendre序列（Legendre sequence）和高斯和（Gauss sum）是數論和數字信號處理中的重要概念，兩者之間有一定的關聯。

Legendre序列

Legendre序列是基於Legendre符號定義的一個二進制序列。給定一個奇素數p，Legendre符號$\left(\frac{a}{p}\right)$定義如下：

$$\left(\frac{a}{p}\right)=\begin{cases}0 & \text{如果 } a\equiv 0 \pmod p\\ 1 & \text{如果 } a \text{ 是模 } p\text{的二次剩餘}\\ -1 & \text{如果 } a \text{ 不是模 } p\text{的二次剩餘}\end{cases}$$

Legendre序列$\{a_n\}$是由Legendre符號生成的，定義如下：

$$a_n=\left(\frac{n}{p}\right) \quad \text{對於 } n=0,1,2,\ldots,p-1$$

這個序列長度為p，且其中的值是-1和1。

https://en.wikipedia.org/wiki/Quadratic_Gauss_sum

高斯和

高斯和是一類複雜數值和，定義為：

$$G(\chi)=\sum_{n=0}^{p-1}\chi(n)e^{2\pi in/p}$$

其中χ是一個模p的狄利克雷特徵（Dirichlet character），特別是當χ是Legendre符號時，高斯和稱為Legendre高斯和。

對於Legendre符號，對應的高斯和記為$G(\left(\frac{\cdot}{p}\right))$，即：

$$G=\sum_{n=0}^{p-1}\left(\frac{n}{p}\right)e^{2\pi in/p}$$

關聯

Legendre序列和高斯和之間的關聯在於，Legendre序列的結構和性質對應於高斯和的性質。具體而言，Legendre高斯和的模長度為：

$$|G|=\sqrt{p}$$

這是一個重要的結果，因為它揭示了Legendre符號生成的二進制序列的自相關性質。Legendre序列在很多應用中具有良好的自相關性質，這使得它們在數字通訊和密碼學中具有重要應用。

Let $M = 11$, $N = 5$, primitive root $g = 2$, $\alpha = g^2 = 4$

$$F = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ \boxed{1} & \boxed{4} & \boxed{5} & \boxed{9} & \boxed{3} \\ 1 & 5 & 3 & 4 & 9 \\ 1 & 9 & 4 & 3 & 5 \\ 1 & 3 & 9 & 5 & 4 \end{bmatrix}, x = \begin{array}{cl} & \text{是平方數放 1} \\ \left[\begin{array}{c} 0 \\ 1 \\ -1 \\ -1 \\ 1 \end{array}\right] & \begin{array}{l} \\ 1 = 1\text{^}2 \ (\text{mod } 5) \\ 2 \\ 3 \\ 4 = 2\text{^}2 \ (\text{mod } 5) \end{array} \end{array}$$

比較一下，Fx 和 fft(x) 有何相似

$1\times 0 + 4\times 1 + 5\times(-1) + 9\times(-1) + 3\times 1 = 0 + 4 - 5 - 9 + 3$
$= -7$
$\equiv 4 \pmod{11}$

Fx=[0 $\boxed{4}$ 7 7 4]

注意在 mod 11 中　4 的平方就是 5

換句話說 4 就是　根號 5

$4^2 = 16 \equiv 5 \pmod{11}$

```
>> fft([0 1 -1 -1 1]')
ans =

         0
    2.2361
   -2.2361
   -2.2361
    2.2361

>> 
```

• 14-F Impulse train and Ramanujan's sum

假設長度 N 有一個因數 d ，則 Impulse train 為

An 1D impulse train $\Pi_{N,d}$ is

$$\Pi_{N,d}(n) = \begin{cases} 1 & n = md, \\ 0 & \text{otherwise,} \end{cases}$$

with $d|N$. Alternatively, we can write

$$\Pi_{N,d}(n) = \sum_{m=0}^{\frac{N}{d}-1} \delta(n - md).$$

0×3 1×3

$$\begin{array}{cccccc} \| & & & \| & & \\ n=0 & 1 & 2 & 3 & 4 & 5 \\ [\boxed{1} & 0 & 0 & \boxed{1} & 0 & 0] \end{array}$$

time domain 是 impulse train，frequency domain 也會是 impulse train

範例： $\boxed{[1\ 0\ 0\ 1\ 0\ 0]}$ 為 N=6,d=3 的 impulse train

[1 0 1 0 1 0] 為 N=6,d=2 的 impulse train

注意： [1 0 0 0 0 0] 為 N=6,d=6 的 impulse train

[1 1 1 1 1 1] 為 N=6,d=1 的 impulse train

Impulse train 的性質：做完轉換後變成 weighted impulse train

範例：

[1 0 0 1 0 0] 做 DFT 之後，變成 [2 0 2 0 2 0]

[1 1 1 1 1 1] 做 DFT 之後，變成 [6 0 0 0 0 0]

注意從 N,d 的 impulse train ，變成 N,N/d 的

請大家想想看

這個性質在 NTT 中會不會成立呢

M=13 為例做一個 6 點的 NTT

$$F=\begin{bmatrix}1&1&1&1&1&1\\1&4&3&12&9&10\\1&3&9&1&3&9\\1&12&1&12&1&12\\1&9&3&1&9&3\\1&10&9&12&3&4\end{bmatrix},x=\begin{bmatrix}1\\0\\0\\1\\0\\0\end{bmatrix},y=\begin{bmatrix}1\\0\\1\\0\\1\\0\end{bmatrix}$$

比較一下，Fx 和 fft(x) 有何相似，Fy 和 fft(y) 呢？

Ramanujan's sum

Ramanujan's sum

12 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

Not to be confused with Ramanujan summation.

In number theory, **Ramanujan's sum**, usually denoted $c_q(n)$, is a function of two positive integer variables q and n defined by the formula

$$c_q(n) = \sum_{\substack{1 \le a \le q \\ (a,q)=1}} e^{2\pi i \frac{a}{q} n},$$

where $(a, q) = 1$ means that a only takes on values coprime to q.

Ramanujan's sum

What is Ramanujan's Sum

Let signal length $N = 12$

Time Domain	After DFT(Ramanujan's Sum)
$n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11$	$k = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11$
$x_{12}(n) = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$	$\hat{x}_{12}(k) = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$
$x_6(n) = [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]$	$\hat{x}_6(k) = [1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1]$
$x_4(n) = [0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0]$	$\hat{x}_4(k) = [2, -1, -1, 2, -1, -1, 2, -1, -1, 2, -1, -1]$
$x_3(n) = [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0]$	$\hat{x}_3(k) = [2, 0, -2, 0, 2, 0, -2, 0, 2, 0, -2, 0]$
$x_2(n) = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0]$	$\hat{x}_2(k) = [2, 1, -1, -2, -1, 1, 2, 1, -1, -2, -1, 1]$
$x_1(n) = [0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1]$	$\hat{x}_1(k) = [4, 0, 2, 0, -2, 0, -4, 0, -2, 0, 2, 0]$

簡單說： Ramanujan's sum 可以看作把特定的 0-1 sequence 去做 DFT

這個特定的 0-1 sequence 我叫 gcd-delta function

gcd-delta

Impulse train

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

gcd-delta

Impulse train

做 fourier transform

Ramanujan’s sum

independent

$$\boxed{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}} = \begin{array}{c} \begin{array}{cccc} 6 & 3 & 2 & 1 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{bmatrix} 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

一定有一個 linear combination 可以讓 impulse train 轉換成互相獨立

6 點時因為 6 有四個因數：1, 2, 3, 6
所以有四個 impulse train

impulse train 做完 DFT 就會變成 weighted impulse train

兩邊去做 DFT

Weighted Impulse train

Ramanujan's sum

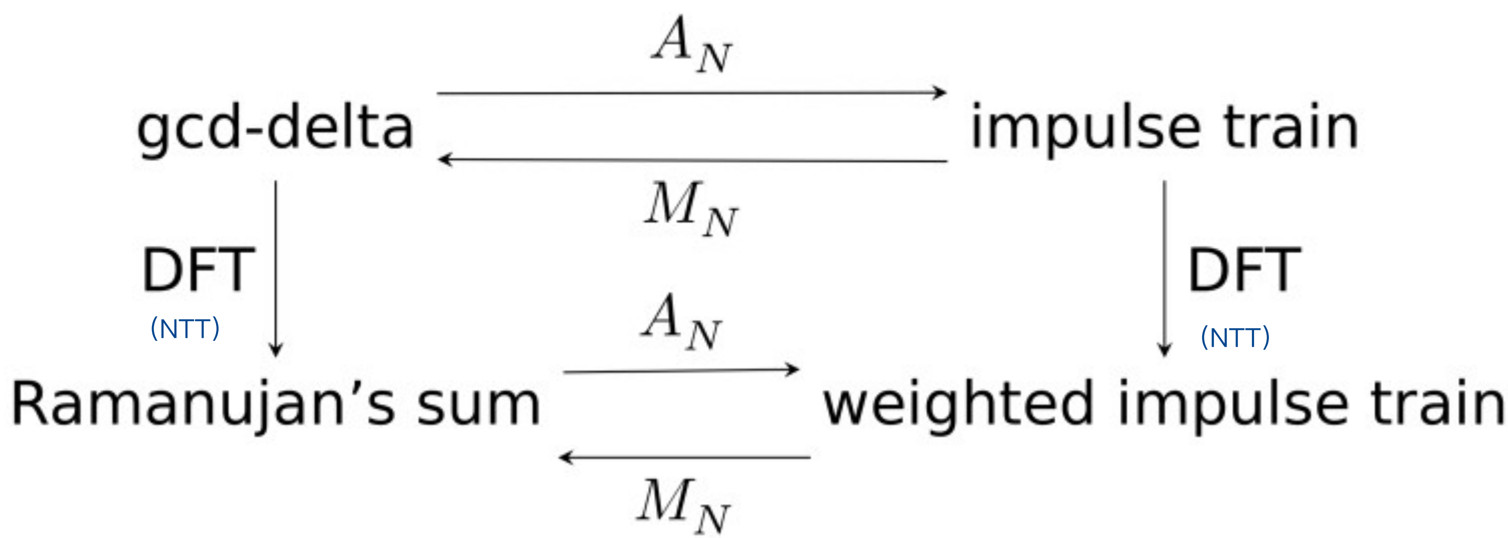
Take DFT at both side in eq. (15),

$$\begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 2 & -2 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 6 \\ 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 1 & 0 & 3 & 0 \\ 1 & 2 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

好處：因為這兩個矩陣都是整數，所以 ramanujan’s sum 這個矩陣也會都是整數

It shows Ramanujan’s sum can be easily calculate by impulse train, without DFT.

最後結論，大家可以把 DFT 的地方換成 NTT ，結論還是一樣



• 14-G Applications of the NTT

NTT 適合作 convolution

但是有不少的限制

新的應用： encryption (密碼學)

CDMA

References:

(1) R. C. Agavard and C. S. Burrus, “Number theoretic transforms to implement fast digital convolution,” *Proc. IEEE*, vol. 63, no. 4, pp. 550-560, Apr. 1975.

(2) T. S. Reed & T. K. Truoay, ”The use of finite field to compute convolution,” *IEEE Trans. Info. Theory*, vol. IT-21, pp.208-213, March 1975

(3) E. Vegh and L. M. Leibowitz, “Fast complex convolution in finite rings,” *IEEE Trans ASSP*, vol. 24, no. 4, pp. 343-344, Aug. 1976.

(4) J. H. McClellan and C. M. Rader, *Number Theory in Digital Signal Processing*, Prentice-Hall, New Jersey, 1979.

(5) 華羅庚, “數論導引,” 凡異出版社, 1997。

(6) S. -C. Pei and K. -W. Chang, "Two Dimensional Efficient Multiplier-Less Structures of Möbius Function for Ramanujan Filter Banks," in IEEE Transactions on Signal Processing, vol. 68, pp. 5079-5091, 2020, doi: 10.1109/TSP.2020.3021245.

(7) https://math.libretexts.org/Bookshelves/Combinatorics_and_Discrete_Mathematics/Elementary_Number_Theory_(Barrus_and_Clark)/01%3A_Chapters

1.8、1.9、1.10、1.13、1.14、1.23、1.24、1.26