Algorithms: COMP3121/3821/9101/9801

Aleks Ignjatović

School of Computer Science and Engineering
University of New South Wales

LECTURE 4: FAST FOURIER TRANSFORM

Fast multiplication of polynomials

- $P_A(x) = A_n x^n + \ldots + A_0, \quad P_B(x) = B_n x^n + \ldots + B_0$ two polynomials.
- We pad them with zeros to degree $2n$, by setting

$$A_{2n} = \ldots = A_{n+1} = B_{2n} = \ldots = B_{n+1} = 0$$

so

$$P_A(x) = 0 \cdot x^{2n} + \ldots + 0 \cdot x^{n+1} + A_n x^n + \ldots + A_0$$

$$P_B(x) = 0 \cdot x^{2n} + \ldots + 0 \cdot x^{n+1} + B_n x^n + \ldots + B_0$$

- We saw that in this case we have

$$P_A(x) \cdot P_B(x) = \sum_{j=0}^{2n} \left(\sum_{i=0}^{j} A_i B_{j-i} \right) x^j$$

Fast multiplication of polynomials

- If we let $a = \langle A_0, \dots, A_n \rangle$ and $b = \langle B_0, \dots, B_n \rangle$, then the sequence

  $$a * b = \left\langle \sum_{i=0}^{j} A_i B_{j-i} \right\rangle_{j=0}^{2n}$$

  is called the *Linear Convolution* of sequences a and b.

  Thus,

  $$\begin{aligned} a * b = \langle & A_n B_n,\ A_{n-1}B_n + A_n B_{n-1},\ A_{n-2}B_n + A_{n-1}B_{n-1} + A_n B_{n-2}, \\ & \dots, A_2B_0 + A_1B_1 + A_0B_2,\ A_1B_0 + A_0B_1,\ A_0B_0 \rangle \end{aligned}$$

- Note that the indices of A_i and B_{j-i} in the j^{th} term all sum up to j.

Coefficient vs value representation of polynomials

- Every polynomial $P_A(x)$ of degree n is uniquely determined by its values at any $n+1$ distinct input values for x:

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_n, P_A(x_n))\}$$

- If $P_A(x) = A_n x^n + A_{n-1} x^{n-1} + \ldots + A_0$, we can write in matrix form:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ \vdots \\ A_n \end{pmatrix} = \begin{pmatrix} P_A(x_0) \\ P_A(x_1) \\ \vdots \\ P_A(x_n) \end{pmatrix}. \tag{1}$$

- It can be shown that if x_i are all distinct, then this matrix is invertible.

Coefficient vs value representation of polynomials - ctd.

- Thus, if all x_i are distinct, given any values $P_A(x_0), P_A(x_1), \ldots, P_A(x_n)$ the coefficients $A_0, A_1, \ldots, A_n$ are uniquely determined:

$$\begin{pmatrix} A_0 \\ A_1 \\ \vdots \\ A_n \end{pmatrix} = \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{pmatrix}^{-1} \begin{pmatrix} P_A(x_0) \\ P_A(x_1) \\ \vdots \\ P_A(x_n) \end{pmatrix} \tag{2}$$

- Equations (1) and (2) show how we can commute between:
  1. a representation of a polynomial $P_A(x)$ via its coefficients $A_n, A_{n-1}, \ldots, A_0$, i.e. $P_A(x) = A_n x^n + \ldots + A_1 x + A_0$
  2. a representation of a polynomial $P_A(x)$ via its values

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_n, P_A(x_n))\}$$

Our strategy to multiply polynomials fast:

- Given two polynomials of degree at most n,

$$P_A(x) = A_n x^n + \ldots + A_0; \qquad P_B(x) = B_n x^n + \ldots + B_0$$

1. convert them into value representation at $2n+1$ distinct points $x_0, x_1, \ldots, x_{2n}$:

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_{2n}, P_A(x_{2n}))\}$$

$$P_B(x) \leftrightarrow \{(x_0, P_B(x_0)), (x_1, P_B(x_1)), \ldots, (x_{2n}, P_B(x_{2n}))\}$$

2. multiply them point by point using $2n+1$ multiplications:

$$P_A(x)P_B(x) \leftrightarrow \{(x_0, \underbrace{P_A(x_0)P_B(x_0)}_{P_C(x_0)}), (x_1, \underbrace{P_A(x_1)P_B(x_1)}_{P_C(x_1)}), \ldots, (x_{2n}, \underbrace{P_A(x_{2n})P_B(x_{2n})}_{P_C(x_{2n})})\}$$

3. Convert such value representation of $P_C(x)$ to its coefficient form

$$P_C(x) = C_{2n}x^{2n} + C_{2n-1}x^{2n-1} + \ldots + C_1 x + C_0;$$

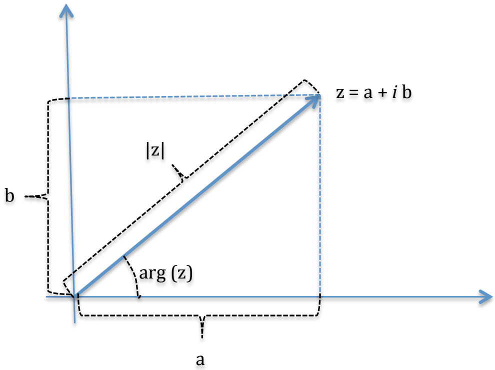
Key Question: What values should we take for $x_0, \ldots, x_{2n}$ to avoid "explosion" of size when we evaluate x_i^n while computing $P_A(x_i) = A_n x_i^n + \ldots + A_0$?

Complex numbers revisited

Complex numbers $z = a + ib$ can be represented using their *modulus* $|z| = \sqrt{a^2 + b^2}$ and their *argument*, $\arg(z)$, which is an angle taking values in $(-\pi, \pi]$ and satisfying:

$$z = |z|e^{i\arg(z)} = |z|(\cos\arg(z) + i\sin\arg(z)),$$

see figure below.

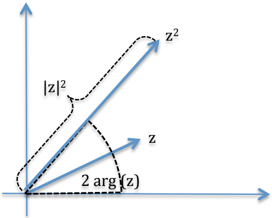


Complex numbers revisited

Recall that

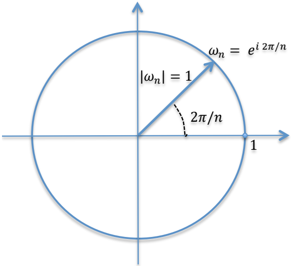
$$z^n = \left(|z|e^{i\ \arg(z)}\right)^n = |z|^n e^{i\,n\,\arg(z)} = |z|^n(\cos(n\arg(z)) + i\sin(n\arg(z))),$$

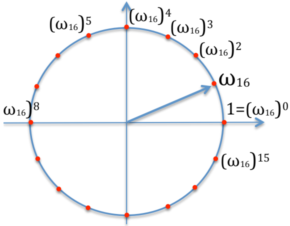
see the figure.



Complex roots of unity

- *Roots of unity of order n* are complex numbers which satisfy $z^n = 1$.
- If $z^n = |z|^n(\cos(n \arg(z)) + i \sin(n \arg(z))) = 1$ then $|z| = 1$ and $n \arg(z)$ is a multiple of 2π;
- Thus, $n \arg(z) = 2\pi\, k$, i.e., $\arg(z) = \dfrac{2\pi\, k}{n}$
- We denote $\omega_n = e^{i\, 2\pi/n}$; such ω_n is a *primitive root of unity of order n*.





Roots of unity of order 16

- A root of unity ω of order n is “primitive” not if it is uncivilised, but
- if all other roots of unity (of the same order) can be obtained as its powers ω^k.

Complex roots of unity

- For $\omega_n = e^{i\,2\pi/n}$
$$((\omega_n)^k)^n = (\omega_n)^{n\,k} = ((\omega_n)^n)^k = 1^k = 1$$
Thus, $\omega_n^k = (\omega_n)^k$ is also a root of unity, and it can be shown that it is primitive just in case k is relatively prime with n.
- Since ω_n^k are roots of unity for $k = 0, 1, \ldots, n-1$ and there are exactly n roots of unity of order n (i.e., solutions to the equation $x^n - 1 = 0$) we get that every root of unity of order n is of the form ω_n^k.
- For any power m of a root of unity ω_n^k we have $(\omega_n^k)^m = \omega_n^{k\,m}$
- if $k\,m > n$ then for some integers $p \geq 1$ and $0 \leq l < n$ we have $k\,m = p\,n + l$ (i.e., $k\,m = l \mod n$) and thus $\omega_n^{k\,m} = \omega_n^{p\,n+l} = \omega_n^{p\,n}\omega_n^l = (\omega_n^n)^p\omega_n^l = \omega_n^l$.
- Thus, any power of any root of unity is just another root of unity of the same order.
- Similarly, a product of any two roots of unity ω_n^k and ω_n^m of the same order we have $\omega_n^k\omega_n^m = \omega_n^{k+m} = \omega_n^l$ where $0 \leq l < n$ and $l = (k+m) \mod n$.
- Thus, product of any two roots of unity of the same order is just another root of unity of the same order.
- So in the set of all roots of unity of order n, i.e., $\{1, \omega_n, \omega_n^2, \ldots, \omega_n^{n-1}\}$ we can multiply any two elements or raise an element to any power without going out of this set.
- Note that this is not true for addition, i.e., the sum of two roots of unity is NOT another root of unity!

Complex roots of unity

- **Cancelation Lemma**: $\omega_{kn}^{km} = \omega_n^m$.

  Proof:

  $$\omega_{kn}^{km} = (\omega_{kn})^{km} = (e^{i\frac{2\pi}{kn}})^{km} = e^{i\frac{2\pi km}{kn}} = e^{i\frac{2\pi m}{n}} = (e^{i\frac{2\pi}{n}})^m = \omega_n^m$$

- Thus, in particular, $(\omega_{2n}^k)^2 = (\omega_{2n}^2)^k = \omega_{2n}^{2k} = \omega_n^k$;

- squares of the roots of unity of order $2n$ are just the roots of unity of order n.

The Discrete Fourier Transform

- Let $A = \langle A_0, A_1, \ldots, A_n \rangle$ be a sequence of $n+1$ real or complex numbers.
- We can form the corresponding polynomial $P_A(x) = \sum_{j=0}^{n} A_j x^j$,
- We evaluate it at all complex roots of unity of order $n+1$, i.e., we can evaluate $P_A(\omega_{n+1}^k)$ for all $0 \leq k \leq n$.
- The sequence of values $\langle P_A(1), P_A(\omega_{n+1}), P_A(\omega_{n+1}^2), \ldots, P_A(\omega_{n+1}^n) \rangle$, is called **the Discrete Fourier Transform (DFT)** of the sequence $A = \langle A_0, A_1, \ldots, A_n \rangle$.

New way for fast multiplication of polynomials

- To multiply two polynomials of degree (at most) n we will evaluate them at the roots of unity of order $2n+1$ (instead of at $-n, \ldots, -1, 0, 1, \ldots, n$ as in Karatsuba's method)

- this produces the DFT of the (0 padded) sequence of their coefficients $(A_0, A_1, \ldots, A_n, \underbrace{0, \ldots, 0}_{n})$;

- we will then multiply the corresponding values $P_A(\omega_{2n+1}^k)$ and $P_B(\omega_{2n+1}^k)$;

- we then use the inverse transformation for DFT, called IDFT, to recover the coefficients of the product polynomial from its values at these roots of unity.

New way for fast multiplication of polynomials

$$P_A(x) = A_0 + A_1 x + \ldots + A_{n-1} x^{n-1} \qquad P_B(x) = B_0 + B_1 x + \ldots + B_{n-1} x^{n-1}$$

$$\Downarrow \text{ DFT} \qquad\qquad \Downarrow \text{ DFT}$$

$$\{P_A(1), P_A(\omega_{2n+1}), P_A(\omega_{2n+1}^2), \ldots, P_A(\omega_{2n+1}^{2n})\}; \quad \{P_B(1), P_B(\omega_{2n+1}), P_B(\omega_{2n+1}^2), \ldots, P_B(\omega_{2n+1}^{2n})\}$$

$$\Downarrow \text{ multiplication}$$

$$\{P_A(1)P_B(1), \quad P_A(\omega_{2n+1})P_B(\omega_{2n+1}), \ldots, P_A(\omega_{2n+1}^{2n})P_B(\omega_{2n+1}^{2n})\}$$

$$\Downarrow \text{IDFT}$$

$$P_C(x) = \underbrace{\left(\sum_{i=0}^{j} A_i B_{j-i}\right)}_{C_j} x^j = \sum_{j=0}^{2n} C_j x^j = P_A(x) \cdot P_B(x)$$

Fast multiplication of polynomials

- Multiplying $2n+1$ values of $P_A(\omega_{2n+1}^k)$ and $P_B(\omega_{2n+1}^k)$ is done in linear time;
- so we have to find an efficient way to compute DFT and IDFT;
- For each fixed k we need to evaluate

$$P_A(x) = A_0 + A_1\omega_{2n+1}^k + A_2\omega_{2n+1}^{2k} + \ldots + A_{n-1}\omega_{2n+1}^{(n-1)k}$$

$$P_B(x) = B_0 + B_1\omega_{2n+1}^k + B_2\omega_{2n+1}^{2k} + \ldots + B_{n-1}\omega_{2n+1}^{(n-1)k}$$

- we could precompute all of the values ω_{2n+1}^k, but, by brute force, for each k we would have to do $n+1$ multiplications of the form $A_m \cdot \omega_{2n+1}^{km}$, for $0 \leq m \leq n$.
- Thus, since k ranges from 0 to $2n$, we would have to do $O(n^2)$ multiplications.
- Can we do it faster??
- This is precisely what the **Fast Fourier Transform (FFT)** does; it computes all of the values $P_A(\omega_{2n+1}^k)$ in $\mathbf{O(n \log n)}$ time.

Fast multiplication of polynomials

- Let $P_A(x) = A_0 + A_1 x + \ldots + A_{n-1}x^{n-1}$;
  - we can assume that n is a power of 2 - otherwise we can pad $P_A(x)$ with zero coefficients until its degree becomes equal to the nearest power of 2.
  - Exercise: show that for every n which is not a power of two the smallest power of 2 larger than n is smaller than $2n$.
  - *Hint:* consider n in binary. How many bits does the nearest power of two have?

Fast multiplication of polynomials

- Idea: divide-and-conquer by splitting the polynomial into even powers and odd powers:

$$\begin{aligned}P_A(x) &= (A_0 + A_2x^2 + A_4x^4 + \ldots + A_{n-2}x^{n-2}) + (A_1x + A_3x^3 + \ldots A_{n-1}x^{n-1}) \\ &= A_0 + A_2x^2 + A_4(x^2)^2 + \ldots + A_{n-2}(x^2)^{\frac{n}{2}-1} \\ &\qquad + x\left(A_1 + A_3x^2 + A_5(x^2)^2 + \ldots + A_{n-1}(x^2)^{\frac{n}{2}-1}\right)\end{aligned}$$

- Let us define

$$A^0(y) = A_0 + A_2y + A_4y^2 + \ldots + A_{n-2}y^{\frac{n}{2}-1}$$

$$A^1(y) = A_1 + A_3y + A_5y^2 + \ldots + A_{n-1}y^{\frac{n}{2}-1}$$

- Then

$$P_A(x) = A^0(x^2) + xA^1(x^2)$$

- Note that the degree of the polynomials $A^0(y)$ and $A^1(y)$ is **half** of the degree of the polynomial $P_A(x)$.

Fast multiplication of polynomials

- **Problem of size n:**
  Evaluate a polynomial of degree $n-1$ at n many roots of unity.

- **Problem of size $n/2$:**
  Evaluate a polynomial of degree $n/2-1$ at $n/2$ many roots of unity.

- We reduced evaluation of our polynomial $P_A(x)$ of degree $n-1$ at inputs $x = \omega_n^0,\ x = \omega_n^1,\ x = \omega_n^2, \ldots, x = \omega_n^{n-1}$ to evaluation of two polynomials $A^0(y)$ and $A^1(y)$ of degree $n/2-1$, at points $y = x^2$ for the same values of inputs x.

- As x ranges through values $\{\omega_n^0, \omega_n^1, \omega_n^2, \ldots, \omega_n^{n-1}\}$, the value of $y = x^2$ ranges through $\{\omega_{\frac{n}{2}}^0, \omega_{\frac{n}{2}}^1, \omega_{\frac{n}{2}}^2, \ldots, \omega_{\frac{n}{2}}^{n-1}\}$, and there are only $n/2$ distinct such values.

- Once we got these $n/2$ values of $A^0(x^2)$ and $A^1(x^2)$ we need n additional multiplications to obtain the values of

$$P_A(\omega_n^k) = A^0((\omega_n^k)^2) + \omega_n^k A^1((\omega_n^k)^2)$$

Fast multiplication of polynomials

- Note that by the Cancelation Lemma $\omega_n^{\frac{n}{2}} = \omega_{2\frac{n}{2}}^{\frac{n}{2}} = \omega_2 = -1$; thus,

$$\omega_n^{k+\frac{n}{2}} = \omega_n^{\frac{n}{2}} \omega_n^k = \omega_2 \omega_n^k = -\omega_n^k;$$

  We can now simplify evaluation of

$$P_A(\omega_n^k) = A^0((\omega_n^k)^2) + \omega_n^k A^1((\omega_n^k)^2)$$

  for $k > n/2$ as follows: let $k = \frac{n}{2} + m$; then

$$\begin{aligned}
P_A(\omega_n^{\frac{n}{2}+m}) &= A^0((\omega_n^{\frac{n}{2}+m})^2) + \omega_n^{\frac{n}{2}+m} A^1((\omega_n^{\frac{n}{2}+m})^2) \\
&= A^0(\omega_n^{n+2m}) + \omega_n^{\frac{n}{2}} \omega_n^m A^1(\omega_n^{n+2m}) \\
&= A^0(\omega_n^n \omega_n^{2m}) + \omega_{2\frac{n}{2}}^{\frac{n}{2}} \omega_n^m A^1(\omega_n^n \omega_n^{2m}) \\
&= A^0(\omega_n^{2m}) + \omega_2\, \omega_n^m A^1(\omega_n^{2m}) \\
&= A^0((\omega_n^m)^2) - \omega_n^m A^1((\omega_n^m)^2)
\end{aligned}$$

- Compare this with $P_A(\omega_n^m) = A^0((\omega_n^m)^2) + \omega_n^m A^1((\omega_n^m)^2)$

Fast multiplication of polynomials

- So we can replace evaluations of

$$P_A(\omega_n^k) = A^0((\omega_n^k)^2) + \omega_n^k A^1((\omega_n^k)^2)$$

  for $k = 0$ to $k = n - 1$

  with such evaluations only for $k = 0$ to $k = n/2 - 1$

  and just let for $m = 0$ to $m = n/2 - 1$

$$P_A(\omega_n^{\frac{n}{2}+m}) = A^0((\omega_n^k)^2) - \omega_n^k A^1((\omega_n^k)^2)$$

- We can now write a pseudo-code for our FFT algorithm:

FFT algorithm

function FFT(A)
    $n \leftarrow \text{length}[A]$
    if $n = 1$ **then return** A
    else
        $A^{[0]} \leftarrow (A_0, A_2, \ldots A_{n-2});$
        $A^{[1]} \leftarrow (A_1, A_3, \ldots A_{n-1});$
        $y^{[0]} \leftarrow FFT(A^{[0]});$
        $y^{[1]} \leftarrow FFT(A^{[1]});$
        $\omega_n \leftarrow e^{i\frac{2\pi}{n}};$
        $\omega \leftarrow 1;$
        for $k = 0$ to $k = \frac{n}{2} - 1$ **do**;
            $y_k \leftarrow y_k^{[0]} + \omega \cdot y_k^{[1]};$
            $y_{\frac{n}{2}+k} \leftarrow y_k^{[0]} - \omega \cdot y_k^{[1]}$
            $\omega \leftarrow \omega \cdot \omega_n;$
        end for
        return y
    end if
end function

Fast multiplication of polynomials

To recapitulate:

- Problem of size n:

  "Evaluate a polynomial of degree $n-1$ at n many roots of unity"

has been reduced to two problems of size $n/2$:

- *"Evaluate a polynomial of degree $n/2-1$ at $n/2$ many roots of unity"*

because we reduced evaluation of our polynomial $P_A(x)$ of degree $n-1$ to evaluation of two polynomials $A^0(y)$ and $A^1(y)$ of degree $n/2-1$, where $y = x^2$, and:

- as x ranges through values $\{\omega_n^0, \omega_n^1, \omega_n^2, \ldots, \omega_n^{n-1}\}$, the value of $y = x^2$ ranges through $\{\omega_{\frac{n}{2}}^0, \omega_{\frac{n}{2}}^1, \omega_{\frac{n}{2}}^2, \ldots, \omega_{\frac{n}{2}}^{n-1}\}$, and there are only $n/2$ distinct such values.

Fast multiplication of polynomials

- Once we get these $n/2$ values of $A^0(x^2)$ and $A^1(x^2)$ we need $n/2$ additional multiplications to obtain the values of

$$P_A(\omega_n^k) = A^0((\omega_n^k)^2) + \omega_n^k A^1((\omega_n^k)^2)$$

and

$$P_A(\omega_n^{\frac{n}{2}+k}) = A^0((\omega_n^k)^2) - \omega_n^k A^1((\omega_n^k)^2)$$

- Thus, we reduced a problem of size n to two such problems of size $n/2$, plus a linear overhead;
- so our algorithm's run time satisfies the recurrence

$$T(n) = 2\,T\left(\frac{n}{2}\right) + c\,n$$

- The Master Theorem gives $T(n) = \Theta(n \log n)$.

Recall our strategy for fast multiplication of polynomials

- Evaluation of a polynomial $P_A(x) = A_0 + A_1 x + \ldots + A_{n-1}x^{n-1}$ at roots of unity ω_n^k of order n can be represented in the matrix form as follows:

$$\begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \ldots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \ldots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \end{pmatrix} = \begin{pmatrix} P_A(1) \\ P_A(\omega_n) \\ P_A(\omega_n^2) \\ \vdots \\ P_A(\omega_n^{n-1}) \end{pmatrix} \tag{3}$$

- The FFT is just a method replacing this matrix-vector multiplication taking n^2 many multiplications with an $n \log n$ procedure;
- From $P_A(1) = P_A(\omega_n^0), \quad P_A(\omega_n), \; P_A(\omega_n^2), \ldots, P_A(\omega_n^{n-1})$, we get the coefficients from

$$\begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \ldots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \ldots & \omega_n^{(n-1)(n-1)} \end{pmatrix}^{-1} \begin{pmatrix} P_A(1) \\ P_A(\omega_n) \\ P_A(\omega_n^2) \\ \vdots \\ P_A(\omega_n^{n-1}) \end{pmatrix} \tag{4}$$

Recall our strategy for fast multiplication of polynomials

- Another remarkable feature of the roots of unity: to obtain the inverse of the above matrix, all we have to do is just change the signs of the exponents and divide everything by n:

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \dots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)} \end{pmatrix}^{-1} =$$

$$\frac{1}{n}\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n^{-1} & \omega_n^{-2} & \dots & \omega_n^{-(n-1)} \\ 1 & \omega_n^{-2} & \omega_n^{-2\cdot 2} & \dots & \omega_n^{-2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \dots & \omega_n^{-(n-1)(n-1)} \end{pmatrix}$$

To see this, note that if we compute the product

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \dots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n^{-1} & \omega_n^{-2} & \dots & \omega_n^{-(n-1)} \\ 1 & \omega_n^{-2} & \omega_n^{-2\cdot 2} & \dots & \omega_n^{-2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \dots & \omega_n^{-(n-1)(n-1)} \end{pmatrix}$$

the (i, j) entry in the product matrix is equal to a product of i^{th} row and j^{th} column:

$$\begin{pmatrix} 1 & \omega_n^i & \omega_n^{2\cdot i} & \dots & \omega_n^{i\cdot(n-1)} \end{pmatrix} \begin{pmatrix} 1 \\ \omega_n^{-j} \\ \omega_n^{-2j} \\ \vdots \\ \omega_n^{-(n-1)j} \end{pmatrix} = \sum_{k=0}^{n-1} \omega_n^{ik} \omega_n^{-jk} = \sum_{k=0}^{n-1} \omega_n^{(i-j)k}$$

Recall our strategy for fast multiplication of polynomials

We now have two possibilities:

1. $i = j$: then
$$\sum_{k=0}^{n-1} \omega_n^{(i-j)k} = \sum_{k=0}^{n-1} \omega_n^0 = \sum_{k=0}^{n-1} 1 = n;$$

2. $i \neq j$: then $\sum_{k=0}^{n-1} \omega_n^{(i-j)k}$ represents a geometric series with the ratio ω_n and thus
$$\sum_{k=0}^{n-1} \omega_n^{(i-j)k} = \frac{1-\omega_n^{(i-j)n}}{1-\omega_n^{i-j}} = \frac{1-(\omega_n^n)^{i-j}}{1-\omega_n^{i-j}} = \frac{1-1}{1-\omega_n^{i-j}} = 0$$

So,

$$\begin{pmatrix} 1 & \omega_n^i & \omega_n^{2\cdot i} & \dots & \omega_n^{i\cdot(n-1)} \end{pmatrix} \begin{pmatrix} 1 \\ \omega_n^{-j} \\ \omega_n^{-2j} \\ \vdots \\ \omega_n^{-(n-1)j} \end{pmatrix} = \sum_{k=0}^{n-1} \omega_n^{(i-j)k} = \begin{cases} n & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{5}$$

$$
\begin{pmatrix}
1 & 1 & 1 & \dots & 1 \\
1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \\
1 & \omega_n^2 & \omega_n^{2\cdot 2} & \dots & \omega_n^{2\cdot(n-1)} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)}
\end{pmatrix}
\begin{pmatrix}
1 & 1 & 1 & \dots & 1 \\
1 & \omega_n^{-1} & \omega_n^{-2} & \dots & \omega_n^{-(n-1)} \\
1 & \omega_n^{-2} & \omega_n^{-2\cdot 2} & \dots & \omega_n^{-2\cdot(n-1)} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \dots & \omega_n^{-(n-1)(n-1)}
\end{pmatrix}
=
\begin{pmatrix}
n & 0 & 0 & \dots & 0 \\
0 & n & 0 & \dots & 0 \\
0 & 0 & n & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \dots & n
\end{pmatrix}
$$

i.e.

$$
\begin{pmatrix}
1 & 1 & 1 & \dots & 1 \\
1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \\
1 & \omega_n^2 & \omega_n^{2\cdot 2} & \dots & \omega_n^{2\cdot(n-1)} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)}
\end{pmatrix}^{-1}
= \frac{1}{n}
\begin{pmatrix}
1 & 1 & 1 & \dots & 1 \\
1 & \omega_n^{-1} & \omega_n^{-2} & \dots & \omega_n^{-(n-1)} \\
1 & \omega_n^{-2} & \omega_n^{-2\cdot 2} & \dots & \omega_n^{-2\cdot(n-1)} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \dots & \omega_n^{-(n-1)(n-1)}
\end{pmatrix}
$$

- We now have

$$\begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \dots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)} \end{pmatrix}^{-1} \begin{pmatrix} P_A(1) \\ P_A(\omega_n) \\ P_A(\omega_n^2) \\ \vdots \\ P_A(\omega_n^{n-1}) \end{pmatrix} =$$

$$= \frac{1}{n} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n^{-1} & \omega_n^{-2} & \dots & \omega_n^{-(n-1)} \\ 1 & \omega_n^{-2} & \omega_n^{-2\cdot 2} & \dots & \omega_n^{-2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \dots & \omega_n^{-(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} P_A(1) \\ P_A(\omega_n) \\ P_A(\omega_n^2) \\ \vdots \\ P_A(\omega_n^{n-1}) \end{pmatrix}$$

- This means that to covert from the values

$$\langle P_A(1), P_A(\omega_n), P_A(\omega_n^2), \dots, P_A(\omega_n^{n-1}) \rangle$$

back to the coefficient form

$$P_A(x) = A_0 + A_1 x + A_2 x^2 + A_{n-1} x^{n-1}$$

we can use **the same** FFT algorithm with the only change that: (1) the root of unity ω_n is replaced by $\omega_n^{-1} = e^{-i\frac{2\pi}{n}}$, and that (2) the resulting values are divided by n.

IFFT algorithm

Inverse Fourier Transform:

function IFFT(A)
$n \leftarrow \text{length}[A]$
if $n = 1$ **then return** A
else
$A^{[0]} \leftarrow (A_0, A_2, \ldots A_{n-2});$
$A^{[1]} \leftarrow (A_1, A_3, \ldots A_{n-1});$
$y^{[0]} \leftarrow FFT(A^{[0]});$
$y^{[1]} \leftarrow FFT(A^{[1]});$
$\omega_n \leftarrow e^{-i\frac{2\pi}{n}};$ $\Leftarrow$ different from FFT
$\omega \leftarrow 1;$
for $k = 0$ to $k = \frac{n}{2} - 1$ **do**;
$y_k \leftarrow y_k^{[0]} + \omega \cdot y_k^{[1]};$
$y_{\frac{n}{2}+k} \leftarrow y_k^{[0]} - \omega \cdot y_k^{[1]}$
$\omega \leftarrow \omega \cdot \omega_n;$
end for
return $\frac{y}{n};$ $\Leftarrow$ different from FFT
end if
end function

Interpretation of DFT

- We have followed the textbook (CLRS);
- however, what CLRS calls DFT, namely, the sequence

$$\langle P_A(\omega_n^0), P_A(\omega_n^1), P_A(\omega_n^2), \ldots, P_A(\omega_n^{n-1})\rangle$$

  is usually considered the Inverse Discrete Fourier Transform (IDFT) of the sequence of the coefficients

$$\langle A_0, A_1, A_2, \ldots, A_{n-1}\rangle$$

  of the polynomial $P_A(x)$;
- $$\langle P_A(\omega_n^0), P_A(\omega_n^{-1}), P_A(\omega_n^{-2}), \ldots, P_A(\omega_n^{-(n-1)})\rangle$$

  is considered the "forward operation" i.e., the DFT.
- taking this as the "forward operation" has an important conceptual advantage and is used more often than the textbook's choice.

Interpretation of DFT

- Another "tweak" of DFT: note that

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \dots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n^{-1} & \omega_n^{-2} & \dots & \omega_n^{-(n-1)} \\ 1 & \omega_n^{-2} & \omega_n^{-2\cdot 2} & \dots & \omega_n^{-2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \dots & \omega_n^{-(n-1)(n-1)} \end{pmatrix}$$

$$= n \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

implies:

Interpretation of DFT

$$\begin{pmatrix} \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{n}} & \vdots & \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} & \frac{\omega_n}{\sqrt{n}} & \frac{\omega_n^2}{\sqrt{n}} & \vdots & \frac{\omega_n^{n-1}}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} & \frac{\omega_n^2}{\sqrt{n}} & \frac{\omega_n^{2\cdot 2}}{\sqrt{n}} & \vdots & \frac{\omega_n^{2\cdot(n-1)}}{\sqrt{n}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{\sqrt{n}} & \frac{\omega_n^{n-1}}{\sqrt{n}} & \frac{\omega_n^{2(n-1)}}{\sqrt{n}} & \vdots & \frac{\omega_n^{(n-1)(n-1)}}{\sqrt{n}} \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{n}} & \vdots & \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} & \frac{\omega_n^{-1}}{\sqrt{n}} & \frac{\omega_n^{-2}}{\sqrt{n}} & \vdots & \frac{\omega_n^{-(n-1)}}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} & \frac{\omega_n^{-2}}{\sqrt{n}} & \frac{\omega_n^{-2\cdot 2}}{\sqrt{n}} & \vdots & \frac{\omega_n^{-2\cdot(n-1)}}{\sqrt{n}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{\sqrt{n}} & \frac{\omega_n^{-(n-1)}}{\sqrt{n}} & \frac{\omega_n^{-2(n-1)}}{\sqrt{n}} & \vdots & \frac{\omega_n^{-(n-1)(n-1)}}{\sqrt{n}} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

Thus, these two matrices are inverses of each other.

Interpretation of DFT

- This motivates us to "tweak" the definition of DFT:
- Given a sequence of numbers $(A_0, A_1, \ldots, A_{n-1})$ the Discrete Fourier Transform of this sequence is the sequence of the values of the polynomial

$$A^*(x) = \frac{1}{\sqrt{n}} \left(A_0 + A_1 x + \ldots + A_{n-1} x^{n-1}\right)$$

for $x = \omega_n^{-k}$ for $k = 0, \ldots, n-1$; i.e., the sequence of values $A^*(\omega_n^{-k})$:

$$A^*(\omega_n^{-k}) = \frac{1}{\sqrt{n}} \left(A_0 (\omega_n^{-k})^0 + A_1 (\omega_n^{-k})^1 + \ldots + A_{n-1} (\omega_n^{-k})^{n-1}\right)$$

- Given a sequence of numbers $(A_0, A_1, \ldots, A_{n-1})$ the **Inverse Discrete Fourier Transform** of this sequence is the sequence of the values of the same polynomial

$$A^*(x) = \frac{1}{\sqrt{n}} \left(A_0 + A_1 x + \ldots + A_{n-1} x^{n-1}\right)$$

but for $x = \omega_n^{k}$ for $k = 0, \ldots, n-1$; i.e., the sequence of values $A^*(\omega_n^{k})$

$$A^*(\omega_n^{k}) = \frac{1}{\sqrt{n}} \left(A_0 (\omega_n^{k})^0 + A_1 (\omega_n^{k})^1 + \ldots + A_{n-1} (\omega_n^{k})^{n-1}\right)$$

Interpretation of DFT

function FFT(A)
    $n \leftarrow \text{length}[A]$
    if $n = 1$ **then return** A
    else
        $A^{[0]} \leftarrow (A_0, A_2, \dots A_{n-2});$
        $A^{[1]} \leftarrow (A_1, A_3, \dots A_{n-1});$
        $y^{[0]} \leftarrow FFT(A^{[0]});$
        $y^{[1]} \leftarrow FFT(A^{[1]});$
        $\omega_n \leftarrow e^{-i\frac{2\pi}{n}};$
        $\omega \leftarrow 1;$
        for $k = 0$ to $k = \frac{n}{2} - 1$ **do**;
            $y_k \leftarrow y_k^{[0]} + \omega \cdot y_k^{[1]};$
            $y_{\frac{n}{2}+k} \leftarrow y_k^{[0]} - \omega \cdot y_k^{[1]}$
            $\omega \leftarrow \omega \cdot \omega_n;$
        end for
        return $\frac{y}{\sqrt{n}};$
    end if
end function

function IFFT(A)
    $n \leftarrow \text{length}[A]$
    if $n = 1$ **then return** A
    else
        $A^{[0]} \leftarrow (A_0, A_2, \dots A_{n-2});$
        $A^{[1]} \leftarrow (A_1, A_3, \dots A_{n-1});$
        $y^{[0]} \leftarrow FFT(A^{[0]});$
        $y^{[1]} \leftarrow FFT(A^{[1]});$
        $\omega_n \leftarrow e^{i\frac{2\pi}{n}};$
        $\omega \leftarrow 1;$
        for $k = 0$ to $k = \frac{n}{2} - 1$ **do**;
            $y_k \leftarrow y_k^{[0]} + \omega \cdot y_k^{[1]};$
            $y_{\frac{n}{2}+k} \leftarrow y_k^{[0]} - \omega \cdot y_k^{[1]}$
            $\omega \leftarrow \omega \cdot \omega_n;$
        end for
        return $\frac{y}{\sqrt{n}};$
    end if
end function

Interpretation of DFT

- *scalar product* (also called *dot* product) of two vectors with real coordinates, $\vec{x} = (x_0, x_1, \ldots, x_{n-1})$ and $\vec{y} = (y_0, y_1, \ldots, y_{n-1})$, denoted by $\langle \vec{x}, \vec{y} \rangle$ is defined as

$$\langle \vec{x}, \vec{y} \rangle = \sum_{i=0}^{n-1} x_i y_i$$

- If the coordinates of our vectors are complex numbers, then the scalar product of such two vectors is defined as

$$\langle \vec{x}, \vec{y} \rangle = \sum_{i=0}^{n-1} x_i \overline{y_i}$$

where $\overline{z}$ denotes the complex conjugate of z, i.e., $\overline{a + i\,b} = a - i\,b$.

- Note that

$$\overline{\omega_n^k} = \overline{e^{i\frac{2\pi k}{n}}} = \overline{\cos\frac{2\pi k}{n} + i\sin\frac{2\pi k}{n}} = \cos\frac{2\pi k}{n} - i\sin\frac{2\pi k}{n}$$
$$= \cos\frac{-2\pi k}{n} + i\sin\frac{-2\pi k}{n} = e^{-i\frac{2\pi k}{n}} = \omega_n^{-k}$$

- Thus, what we had before,

$$\begin{pmatrix} 1 & \omega_n^k & \omega_n^{2\cdot k} & \dots & \omega_n^{k\cdot(n-1)} \end{pmatrix} \begin{pmatrix} 1 \\ \omega_n^{-m} \\ \omega_n^{-2m} \\ \vdots \\ \omega_n^{-(n-1)m} \end{pmatrix} = \sum_{j=0}^{n-1} \omega_n^{(k-m)j} = \begin{cases} n & \text{if } k = m \\ 0 & \text{if } k \neq m \end{cases} \tag{6}$$

simply means that for $k \neq m$ vectors $\left(1,\ \omega_n^k,\ \omega_n^{2\cdot k},\ \dots,\ \omega_n^{k\cdot(n-1)}\right)$ and $\left(1,\ \omega_n^m,\ \omega_n^{2\cdot m},\ \dots,\ \omega_n^{m\cdot(n-1)}\right)$ are orthogonal.

Interpretation of DFT

- If we define $$\vec{e_k} = \frac{1}{\sqrt{n}} \left(\omega_n^{k\cdot 0}, \;\; \omega_n^{k\cdot 1}, \; \omega_n^{k\cdot 2}, \;\; \ldots \;\; , \omega_n^{k\cdot (n-1)} \right)$$

  then $$\|\vec{e_k}\| = \sqrt{\langle \vec{e_k}, \vec{e_k} \rangle} = 1$$

  and $$\langle \vec{e_k}, \vec{e_m} \rangle = 0 \;\; \text{for} \quad k \neq m$$

- thus, the set of vectors $\{\vec{e_0}, \vec{e_1}, \ldots, \vec{e_n}\}$ is an orthonormal base of the vector space of all complex valued sequences of length n.
- Let $\vec{A} = (A_0, A_1, A_2, \ldots, A_{n-1})$; then for the DFT of this sequence we have

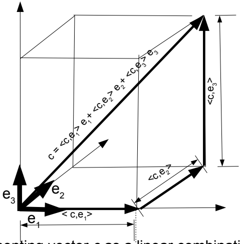
$$\begin{aligned}
\frac{1}{\sqrt{n}} P_A(\omega_n^{-k}) &= \frac{A_0}{\sqrt{n}}(\omega_n^{-k})^0 + \frac{A_1}{\sqrt{n}}(\omega_n^{-k})^1 + \frac{A_2}{\sqrt{n}}(\omega_n^{-k})^2 + \ldots + \frac{A_{n-1}}{\sqrt{n}}(\omega_n^{-k})^{n-1} \\
&= A_0 \frac{\overline{(\omega_n^k)}^0}{\sqrt{n}} + A_1 \frac{\overline{(\omega_n^k)}^1}{\sqrt{n}} + A_2 \frac{\overline{(\omega_n^k)}^2}{\sqrt{n}} + \ldots + A_{n-1} \frac{\overline{(\omega_n^k)}^{n-1}}{\sqrt{n}} \\
&= \left\langle (A_0, A_1, A_2, \ldots, A_{n-1}), \left(\frac{(\omega_n^k)^0}{\sqrt{n}}, \frac{(\omega_n^k)^1}{\sqrt{n}}, \frac{(\omega_n^k)^2}{\sqrt{n}}, \ldots, \frac{(\omega_n^k)^{n-1}}{\sqrt{n}} \right) \right\rangle \\
&= \langle \vec{A}, \vec{e_k} \rangle
\end{aligned}$$

- Thus, the DFT of a vector $\vec{A}$ is simply the sequence of projections of $\vec{A}$ onto the base vectors $\vec{e_k}$, $(k = 0, \ldots, n-1)$.

Interpretation of DFT

- In an n-dimensional vector space V with an orthonormal base $\mathbf{B}$ every vector $\vec{A}$ can be represented as a linear combination of the base vectors with coefficients equal to the projections of $\vec{A}$ onto the base vectors, i.e., the scalar product $\langle \vec{A}, e_k \rangle$:

$$\vec{A} = \langle \vec{A}, \vec{e_0} \rangle \vec{e_0} + \langle \vec{A}, \vec{e_1} \rangle \vec{e_1} + \ldots + \langle \vec{A}, \vec{e}_{n-1} \rangle \vec{e}_{n-1}$$



Representing vector c as a linear combination of the basis vectors e_1, e_2, e_3 with projections as coefficients

Interpretation of DFT

- Thus, in our case

$$\vec{A} = \langle \vec{A}, \vec{e_0} \rangle \vec{e_0} + \langle \vec{A}, \vec{e_1} \rangle \vec{e_1} + \ldots + \langle \vec{A}, \vec{e}_{n-1} \rangle \vec{e}_{n-1}$$
$$= \frac{P_A(\omega_n^0)}{\sqrt{n}} \vec{e_0} + \frac{P_A(\omega_n^{-1})}{\sqrt{n}} \vec{e_1} + \ldots + \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}} \vec{e}_{n-1}$$

- Looking at the k^{th} cordnate of both the left and the right side we get

$$A_k = \frac{P_A(\omega_n^0)}{\sqrt{n}} \frac{(\omega_n^0)^k}{\sqrt{n}} + \frac{P_A(\omega_n^{-1})}{\sqrt{n}} \frac{(\omega_n^1)^k}{\sqrt{n}} + \ldots + \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}} \frac{(\omega_n^{n-1})^k}{\sqrt{n}} \tag{7}$$
$$= \frac{P_A(\omega_n^0)}{\sqrt{n}} \frac{(\omega_n^k)^0}{\sqrt{n}} + \frac{P_A(\omega_n^{-1})}{\sqrt{n}} \frac{(\omega_n^k)^1}{\sqrt{n}} + \ldots + \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}} \frac{(\omega_n^k)^{n-1}}{\sqrt{n}} \tag{8}$$

- A_k is obtained evaluating the polynomial

$$\frac{P_A(\omega_n^0)}{\sqrt{n}} + \frac{P_A(\omega_n^{-1})}{\sqrt{n}} \frac{x}{\sqrt{n}} + \ldots + \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}} \frac{x^{n-1}}{\sqrt{n}}$$

at $x = \omega_n^k$, which is exactly what the Inverse Discrete Fourier Transform is.

Interpretation of DFT

- Let us denote the usual orthonormal base of $\mathbb{C}^n$ by $\mathcal{B}$:

$$\vec{f_0} = (1,0,0,0,\ldots,0),\ \vec{f_1} = (0,1,0,0,\ldots,0),\ \vec{f_2} = (0,0,1,0,\ldots,0),\ \vec{f}_{n-1} = (0,0,0,0,\ldots,1)$$

and by $\mathcal{F}$ the base $\mathcal{F} = \{\vec{e}_0, \vec{e}_1, \ldots, \vec{e}_{n-1}\}$ where $\vec{e_k} = \left(\frac{1}{\sqrt{n}}, \ \frac{\omega_n^{k\cdot 1}}{\sqrt{n}}, \ \frac{\omega_n^{k\cdot 2}}{\sqrt{n}}, \ \ldots \ , \frac{\omega_n^{k\cdot(n-1)}}{\sqrt{n}}\right)$.

- then

$$\vec{A} = (A_0, A_1, A_2, \ldots, A_{n-1})_{\mathcal{B}} = A_0\vec{f_0} + A_1\vec{f_1} + A_2\vec{f_2} + \ldots + A_{n-1}\vec{f}_{n-1}$$

and also

$$\vec{A} = (P_A(\omega_n^0), P_A(\omega_n^1), A_2, \ldots, P_A(\omega_n^{n-1}))_{\mathcal{F}} = \frac{P_A(\omega_n^0)}{\sqrt{n}}\vec{e_0} + \frac{P_A(\omega_n^{-1})}{\sqrt{n}}\vec{e_1} + \ldots + \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}}\vec{e}_{n-1}$$

- DFT is just change of base operation: it transforms the sequence of coordinates

$$(A_0, A_1, A_2, \ldots, A_{n-1})_{\mathcal{B}}$$

in the base $\mathcal{B}$ of vector A into the sequence

$$\left(\frac{P_A(\omega_n^0)}{\sqrt{n}}, \frac{P_A(\omega_n^{-1})}{\sqrt{n}}, \ldots, \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}}\right)_{\mathcal{F}}$$

of the coordinates in the base $\mathcal{F}$;

Interpretation of DFT

- The k^{th} coordinate $\frac{P_A(\omega_n^{-k})}{\sqrt{n}}$ is obtained by projecting vector

$$\vec{A} = (A_0, A_1, A_2, \ldots, A_{n-1})_{\mathcal{B}} = A_0\vec{f_0} + A_1\vec{f_1} + A_2\vec{f_2} + \ldots + A_{n-1}\vec{f}_{n-1}$$

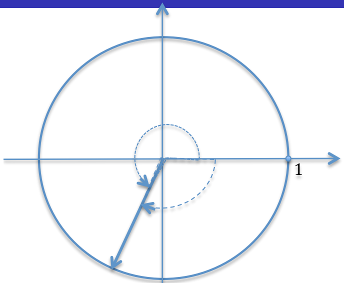
  onto the corresponding base vector $e_k = ((\omega_n^k)^0, (\omega_n^k)^1, \ldots, (\omega_n^k)^{n-1}) \in \mathcal{F}$.

- Recall that the k^{th} coordinate A_k of $\vec{A}$ in the usual base $\mathcal{B}$ was obtained by looking at the k^{th} coordinate of

$$\vec{A} = \frac{P_A(\omega_n^0)}{\sqrt{n}}\vec{e_0} + \frac{P_A(\omega_n^{-1})}{\sqrt{n}}\vec{e_1} + \ldots + \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}}\vec{e}_{n-1}$$

  i.e., by projecting $\frac{P_A(\omega_n^0)}{\sqrt{n}}\vec{e_0} + \frac{P_A(\omega_n^{-1})}{\sqrt{n}}\vec{e_1} + \ldots + \frac{P_A(\omega_n^{-(n-1)})}{\sqrt{n}}\vec{e}_{n-1}$ onto the corresponding base vector $\vec{f_k} = (\underbrace{0, 0, \ldots, 0}_{k-1}, 1, 0, \ldots, 0)$.

- Thus, the Inverse Discrete Fourier Transform (IDFT) simply transforms the sequence of the coordinates of $\vec{A}$ in the base $\mathcal{F}$ back to the sequence of coordinates of $\vec{A}$ in base $\mathcal{B}$, i.e., into $(A_0, A_1, A_2, \ldots, A_{n-1})_{\mathcal{B}}$



$$(\omega_{2n+1})^k = e^{i\frac{2\pi k}{2n+1}} = e^{-i\frac{2\pi(2n+1-k)}{2n+1}}; \quad k > n$$

- Note that by replacing n with $2n+1$ we get

$$A_k = \frac{P_A(\omega_{2n+1}^0)}{\sqrt{2n+1}}\frac{(\omega_{2n+1}^k)^0}{\sqrt{2n+1}} + \frac{P_A(\omega_{2n+1}^1)}{\sqrt{2n+1}}\frac{(\omega_{2n+1}^k)^1}{\sqrt{2n+1}} + \ldots + \frac{P_A(\omega_{2n+1}^{2n})}{\sqrt{2n+1}}\frac{(\omega_{2n+1}^k)^{2n}}{\sqrt{2n+1}}$$

$$= \frac{P_A(\omega_{2n+1}^k)}{2n+1}e^{i\frac{2\pi\, k\cdot 0}{2n+1}} + \frac{P_A(\omega_{2n+1}^1)}{2n+1}e^{i\frac{2\pi\, k\cdot 1}{2n+1}} + \frac{P_A(\omega_{2n+1}^2)}{2n+1}e^{i\frac{2\pi\, k\cdot 2}{2n+1}} + \ldots + \frac{P_A(\omega_{2n+1}^{2n})}{2n+1}e^{i\frac{2\pi\, k\cdot 2n}{2n+1}}$$

$$= \sum_{j=-n}^{n}\frac{P_A(\omega_{2n+1}^j)}{2n+1}e^{i\frac{2\pi\, k\cdot j}{2n+1}} = \sum_{j=-n}^{n}\left|\frac{P_A(\omega_{2n+1}^j)}{2n+1}\right|e^{i\arg(P_A(\omega_{2n+1}^j))}e^{i\frac{2\pi\, k\cdot j}{2n+1}}$$

Interpretation of DFT

- Thus,

$$A_k = \sum_{j=-n}^{n} \left| \frac{P_A(\omega_{2n+1}^{j})}{2n+1} \right| e^{i \arg(P_A(\omega_{2n+1}^{j}))} e^{i \frac{2\pi\ k \cdot j}{2n+1}}$$

$$= \sum_{j=-n}^{n} \left| \frac{P_A(\omega_{2n+1}^{j})}{2n+1} \right| e^{i\left(\frac{2\pi\ k \cdot j}{2n+1} + \arg(P_A(\omega_{2n+1}^{j}))\right)}$$

- If we let

$$a(t) = \sum_{j=-n}^{n} \left| \frac{P_A(\omega_{2n+1}^{j})}{2n+1} \right| e^{i\left(\frac{2\pi\ t \cdot j}{2n+1} + \arg(P_A(\omega_{2n+1}^{j}))\right)}$$

$$= \sum_{j=-n}^{n} \left| \frac{P_A(\omega_{2n+1}^{j})}{2n+1} \right| \left(\cos\left(\frac{2\pi\ \cdot j}{2n+1} t + \arg(P_A(\omega_{2n+1}^{j})) \right) + i \sin\left(\frac{2\pi\ \cdot j}{2n+1} t + \arg(P_A(\omega_{2n+1}^{j})) \right) \right)$$

  then $A_k = a(k)$. Thus the sequence $\langle A_0, A_1, \ldots, A_{2n} \rangle$ has been represented as a linear combination of samples of sinusoids of frequencies $\frac{2\pi k}{2n+1}$ for $k = -n$ to $k = n$.

- If $\vec{A}$ is a real vector, the imaginary part of $a(t)$ cancel out because $P_A(\omega_{2n+1}^{-j}) = \overline{P_A(\omega_{2n+1}^{j})}$ and we get a real valued interpolation signal $a(t)$.

Interpretation of DFT

- Thus, we get

$$a(t) = 2\sum_{j=0}^{n}\left|\frac{P_A(\omega_{2n+1}^{j})}{2n+1}\right|\cos\left(\frac{2\pi\ \cdot j}{2n+1}t + \arg(P_A(\omega_{2n+1}^{j}))\right)$$

  and again $A_k = a(k)$. Thus the sequence $\langle A_0, A_1, \ldots, A_{2n}\rangle$ has been represented as a linear combination of samples of sinusoids of frequencies $\frac{2\pi k}{2n+1}$ for $k = -n$ to $k = n$.

- In essence we have approximated the signal with a linear combination of pure harmonic oscillations of frequencies $\frac{2\pi k}{2n+1}$ with amplitudes $2\left|\frac{P_A(\omega_{2n+1}^{j})}{2n+1}\right|$ and phase shifts $\arg(P_A(\omega_{2n+1}^{j}))$.