Algorithms: COMP3121/3821/9101/9801

Aleks Ignjatović

School of Computer Science and Engineering
University of New South Wales

TOPIC 3: THE FAST FOURIER TRANSFORM

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 $x_0, x_1, \ldots, x_n$:

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

- 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:

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

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

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

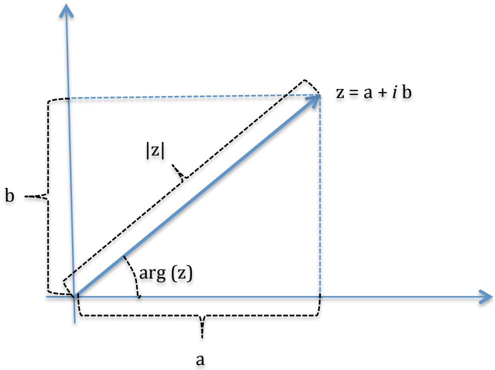
- **Key Question:** What values should we take for $x_0, \ldots, x_{2n}$ to avoid "explosion" of size when we evaluate x_i^n while computing $P_A(x_i) = A_n x_i^n + \ldots + A_0$?

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.

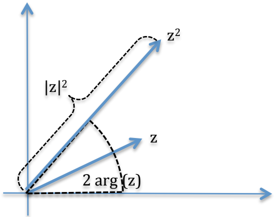


Complex numbers revisited

Recall that

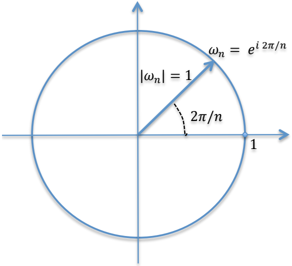
$$z^n = \left(|z|e^{i\ \arg(z)}\right)^n = |z|^n e^{i\,n\arg(z)} = |z|^n(\cos(n\arg(z)) + i\sin(n\arg(z))),$$

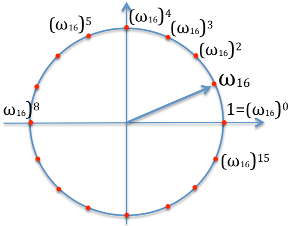
see the figure.



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 called a *primitive root of unity of order* n.





Roots of unity of order 16

- A root of unity ω of order n is *primitive* 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 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, $(\omega_{2n}^k)^2 = \omega_{2n}^{2k} = (\omega_{2n}^2)^k = \omega_n^k$;

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

The Discrete Fourier Transform

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

New way for fast multiplication of polynomials

- If we multiply two polynomials $P_A(x)$ and $P_B(x)$ of degree (at most) $n-1$ we get a polynomial of degree $2n-2$ with $2n-1$ coefficients.

- To uniquely determine a polynomial of degree $2n-2$ we need $2n-1$ many values;

- Thus, we will evaluate both $P_A(x)$ and $P_B(x)$ at all the roots of unity of order $2n-1$ (instead of at $-(n-1), \ldots, -1, 0, 1, \ldots, n-1$ as in Karatsuba's method!)

New way for fast multiplication of polynomials

- Note that we defined the DFT of a sequence of length n as the values of the corresponding polynomial of degree $n-1$ at the n roots of unity of order n, ω_n^k $(0 \leq k < n)$.

- Since we need $2n-1$ values of $P_A(x)$ and $P_B(x)$, we pad both sequences with $n-1$ zeros at the end, $(A_0, A_1, \ldots, A_{n-1}, \underbrace{0, \ldots, 0}_{n-1})$.

- We can now compute the DFTs of the two (0 padded) sequences $(A_0, A_1, \ldots, A_{n-1}, \underbrace{0, \ldots, 0}_{n-1})$ and $(B_0, B_1, \ldots, B_{n-1}, \underbrace{0, \ldots, 0}_{n-1})$.

- 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

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

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

$\Downarrow$ DFT $\qquad\qquad$ $\Downarrow$ DFT

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

$\Downarrow$ multiplication

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

$\Downarrow$ IDFT

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

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

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

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

- We could precompute all of the values ω_{2n-1}^k, but, by brute force, for each k we would have to do n multiplications of the form $A_m \cdot \omega_{2n-1}^{km}$, for $0 \le m < 2n-1$.
- Thus, since both k, m range from 0 to $2n-2$, 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 the values $P_A(\omega_n^k)$ for all k such that $0 \le k < n$ in $\mathbf{O}(\mathbf{n}\log\mathbf{n})$ time.

The Fast Fourier Transform (FFT)

- 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 number of coefficients 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 or equal to n is smaller than $2n$.
    - *Hint:* consider n in binary. How many bits does the nearest power of two have?

The Fast Fourier Transform (FFT)

- **The main idea of the FFT algorithm:** divide-and-conquer by splitting the polynomial into the even powers and the 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

$$P_A(x) = A^{[0]}(x^2) + x\,A^{[1]}(x^2)$$

- Note that the number of coefficients of the polynomials $A^{[0]}(y)$ and $A^{[1]}(y)$ is $n/2$ each, while the number of coefficients of the polynomial $P_A(x)$ is n. Thus, the number of coefficients of each of the two polynomials $A^{[0]}(y)$ and $A^{[1]}(y)$ is only one half of the number of coefficients of the polynomial $P_A(x)$.

The Fast Fourier Transform (FFT)

- **Problem of size n:**
  Evaluate a polynomial with n coefficients at n many roots of unity.

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

- We reduced evaluation of our polynomial $P_A(x)$ with n coefficients 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 $A^{[1]}(y)$ each with $n/2$ coefficients, at points $y = x^2$ for the same values of inputs x.

- However, 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 get these $n/2$ values of $A^{[0]}(x^2)$ and $A^{[1]}(x^2)$ we need n additional multiplications with numbers ω_n^k to obtain the values of

$$\begin{aligned}P_A(\omega_n^k) &= A^{[0]}((\omega_n^k)^2) + \omega_n^k \cdot A^{[1]}((\omega_n^k)^2)\\ &= A^{[0]}(\omega_{n/2}^k) + \omega_n^k \cdot A^{[1]}(\omega_{n/2}^k).\end{aligned}$$

- Thus, we have reduced a problem of size n to two such problems of size $n/2$, plus a linear overhead.

The Fast Fourier Transform (FFT) - a simplification

- 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 $n/2 \leq k < n$ as follows: let $k = \frac{n}{2} + m$ where $0 \leq m < n/2$; then

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

- Compare this with
$P_A(\omega_n^m) = A^{[0]}((\omega_n^m)^2) + \omega_n^m A^{[1]}((\omega_n^m)^2) = A^{[0]}(\omega_{n/2}^m) + \omega_n^m A^{[1]}(\omega_{n/2}^m)$
for $0 \leq m < n/2$.

The Fast Fourier Transform (FFT) - a simplification

- So we can replace evaluations of

$$\begin{aligned} P_A(\omega_n^k) &= A^{[0]}((\omega_n^k)^2) + \omega_n^k A^{[1]}((\omega_n^k)^2) \\ &= A^{[0]}(\omega_{n/2}^k) + \omega_n^k A^{[1]}(\omega_{n/2}^k) \end{aligned}$$

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

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

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

$$\begin{aligned} P_A(\omega_n^{\frac{n}{2}+k}) &= A^{[0]}((\omega_n^k)^2) - \omega_n^k A^{[1]}((\omega_n^k)^2) \\ &= A^{[0]}(\omega_{n/2}^k) - \omega_n^k A^{[1]}(\omega_{n/2}^k) \end{aligned}$$

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

FFT algorithm

```
function FFT(A)
    n ← length[A]
    if n = 1 then return A
    else
        A^[0] ← (A_0, A_2, ... A_{n-2});
        A^[1] ← (A_1, A_3, ... A_{n-1});
        y^[0] ← FFT(A^[0]);
        y^[1] ← FFT(A^[1]);
        ω_n ← e^{i 2π/n};
        ω ← 1;
        for k = 0 to k = n/2 - 1 do;
            y_k ← y_k^[0] + ω · y_k^[1];
            y_{n/2+k} ← y_k^[0] - ω · y_k^[1]
            ω ← ω · ω_n;
        end for
        return y
    end if
end function
```

How fast is the Fast Fourier Transform?

- We have recursively reduced evaluation of a polynomial $P_A(x)$ with n coefficients at n roots of unity of order n to evaluations of two polynomials $A^{[0]}(y)$ and $A^{[1]}(y)$, each with $n/2$ coefficients, at $n/2$ many roots of unity of order $n/2$.
- Once we get these $n/2$ values of $A^{[0]}(y)$ and $A^{[1]}(y)$ we need $n/2$ additional multiplications to obtain the values of

$$\underbrace{P_A(\omega_n^k)}_{y_k} = \underbrace{A^{[0]}(\omega_{n/2}^k)}_{y_k^{[0]}} + \omega_n^k \underbrace{A^{[1]}(\omega_{n/2}^k)}_{y_k^{[1]}} \tag{3}$$

and

$$\underbrace{P_A(\omega_n^{\frac{n}{2}+k})}_{y_{\frac{n}{2}+k}} = \underbrace{A^{[0]}(\omega_{n/2}^k)}_{y_k^{[0]}} - \omega_n^k \underbrace{A^{[1]}(\omega_{n/2}^k)}_{y_k^{[1]}} \tag{4}$$

for all $0 \leq k < n/2$

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

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

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

Matrix representation of polynomial evaluation

- 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:

$$\begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \ldots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \ldots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \end{pmatrix} = \begin{pmatrix} P_A(1) \\ P_A(\omega_n) \\ P_A(\omega_n^2) \\ \vdots \\ P_A(\omega_n^{n-1}) \end{pmatrix} = \begin{pmatrix} \widehat{A}_0 \\ \widehat{A}_1 \\ \widehat{A}_2 \\ \vdots \\ \widehat{A}_{n-1} \end{pmatrix}$$

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

$$\begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^{2\cdot 2} & \ldots & \omega_n^{2\cdot(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \ldots & \omega_n^{(n-1)(n-1)} \end{pmatrix}^{-1} \begin{pmatrix} P_A(1) \\ P_A(\omega_n) \\ P_A(\omega_n^2) \\ \vdots \\ P_A(\omega_n^{n-1}) \end{pmatrix} \tag{5}$$

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^{i-j} 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,

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

So we get:

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

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

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

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

which we denoted by $\langle \widehat{A}_0, \widehat{A}_1, \widehat{A}_2, \dots, \widehat{A}_{n-1} \rangle$ 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 $\overline{\omega_n} = \omega_n^{-1} = e^{-i\frac{2\pi}{n}}$,
2. the resulting output values are divided by n.

Inverse Fast Fourier Transform (IFFT):

function IFFT$^*(\widehat{A})$
  $n \leftarrow \mathrm{length}(\widehat{A})$
  if $n = 1$ **then return** $\widehat{A}$
  else
    $\widehat{A}^{[0]} \leftarrow (\widehat{A}_0, \widehat{A}_2, \ldots \widehat{A}_{n-2});$
    $\widehat{A}^{[1]} \leftarrow (\widehat{A}_1, \widehat{A}_3, \ldots \widehat{A}_{n-1});$
    $y^{[0]} \leftarrow IFFT^*(\widehat{A}^{[0]});$
    $y^{[1]} \leftarrow IFFT^*(\widehat{A}^{[1]});$
    $\omega_n \leftarrow e^{-i\frac{2\pi}{n}};$ $\Leftarrow$ 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 y;
  end if
end function

function IFFT$(\widehat{A})$ $\Leftarrow$ different from FFT
  return $IFFT^*(\widehat{A})/\mathrm{length}(\widehat{A})$
end function

More on 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 $P_A(x)$;
- Instead,

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

  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, especially in electrical engineering literature.

More on 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:

More on 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.

More on DFT

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

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

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

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

- 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

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

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

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

More on DFT

function $\mathrm{FFT}^*(A)$
    $n \leftarrow \mathrm{length}[A]$
    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 y;
    end if
end function

function $\mathrm{IFFT}^*(A)$
    $n \leftarrow \mathrm{length}[A]$
    if $n = 1$ **then return** A
    else
        $A^{[0]} \leftarrow (A_0, A_2, \dots A_{n-2})$;
        $A^{[1]} \leftarrow (A_1, A_3, \dots A_{n-1})$;
        $y^{[0]} \leftarrow IFFT^*(A^{[0]})$;
        $y^{[1]} \leftarrow IFFT^*(A^{[1]})$;
        $\omega_n \leftarrow e^{i\frac{2\pi}{n}}$;
        $\omega \leftarrow 1$;
        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

function $\mathrm{FFT}(A)$
    return $\mathrm{FFT}^*(A)/\sqrt{\mathrm{length}(A)}$;
end function

function $\mathrm{IFFT}(A)$
    return $\mathrm{IFFT}^*(A)/\sqrt{\mathrm{length}(A)}$;
end function

Back to fast multiplication of polynomials (convolution)

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

$$\Downarrow \text{ DFT } \quad O(n \log n) \qquad\qquad \Downarrow \text{ DFT } \quad O(n \log n)$$

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

$$\Downarrow \text{ multiplication } \quad O(n)$$

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

$$\Downarrow \text{ IDFT } \quad O(n \log n)$$

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

Thus, convolution can be computed in time $O(n \log n)$.

Optional Material: Interpretation of DFT

- The **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

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

- Recall that $\overline{z}$ denotes the complex conjugate of z, i.e., $\overline{a + i\,b} = a - i\,b$.
- The **norm** of a vector $\vec{x} = (x_0, x_1, \ldots, x_{n-1})$ is defined as

$$\|\vec{x}\| = \sqrt{\langle \vec{x}, \vec{x} \rangle} = \sqrt{\sum_{i=0}^{n-1} x_i \overline{x_i}} = \sqrt{\sum_{i=0}^{n-1} |x_i|^2}$$

Optional Material: Interpretation of DFT

- 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,

$$\begin{pmatrix} 1 & \omega_n^k & \omega_n^{2\cdot k} & \dots & \omega_n^{k\cdot(n-1)} \end{pmatrix} \begin{pmatrix} 1 \\ \omega_n^{-m} \\ \omega_n^{-2m} \\ \vdots \\ \omega_n^{-(n-1)m} \end{pmatrix} = \sum_{j=0}^{n-1} \omega_n^{(k-m)j} = \begin{cases} n & \text{if } k = m \\ 0 & \text{if } k \neq m \end{cases} \tag{7}$$

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 mutually orthogonal and that their norm is equal to $\sqrt{n}$.

Optional Material: Interpretation of DFT

- If we define $\vec{e_k} = \frac{1}{\sqrt{n}}\left(1, \ \omega_n^{k\cdot 1}, \ \omega_n^{k\cdot 2}, \ \ldots \ , \omega_n^{k\cdot(n-1)}\right)$ then $\|\vec{e_k}\| = \sqrt{\langle \vec{e_k}, \vec{e_k}\rangle} = 1$ and $\langle \vec{e_k}, \vec{e_m}\rangle = 0$ for $k \neq m$
- Thus, the set of vectors $\{\vec{e_0}, \vec{e_1}, \ldots, \vec{e_n}\}$ is an orthonormal basis 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 $P_A^*(x) = \frac{1}{\sqrt{n}}\left(A_0 + A_1 x + \ldots + A_{n-1}x^{n-1}\right)$ we have

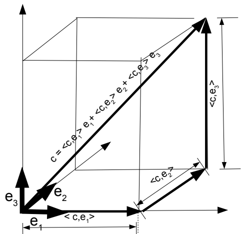
$$\begin{aligned}
\widehat{A}_k &= P_A^*(\omega_n^{-k}) \\
&= \frac{1}{\sqrt{n}} P_A(\omega_n^{-k}) \\
&= \frac{A_0}{\sqrt{n}}(\omega_n^{-k})^0 + \frac{A_1}{\sqrt{n}}(\omega_n^{-k})^1 + \frac{A_2}{\sqrt{n}}(\omega_n^{-k})^2 + \ldots + \frac{A_{n-1}}{\sqrt{n}}(\omega_n^{-k})^{n-1} \\
&= A_0 \frac{\overline{(\omega_n^k)}^0}{\sqrt{n}}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

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

Optional Material: Interpretation of DFT

- In an n-dimensional vector space V with an orthonormal basis $\mathbf{B}$ every vector $\vec{A}$ can be represented as a linear combination of the basis vectors with coefficients equal to the projections of $\vec{A}$ onto the basis 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

Optional Material: Interpretation of DFT

- Thus, in our case

$$\begin{aligned}\vec{A} &= \langle \vec{A}, \vec{e_0}\rangle \vec{e_0} + \langle \vec{A}, \vec{e_1}\rangle \vec{e_1} + \ldots + \langle \vec{A}, \vec{e}_{n-1}\rangle \vec{e}_{n-1} \\ &= P_A^*(\omega_n^0)\vec{e_0} + P_A^*(\omega_n^{-1})\vec{e_1} + \ldots + P_A^*(\omega_n^{-(n-1)})\vec{e}_{n-1} \\ &= \widehat{A}_0\vec{e_0} + \widehat{A}_1\vec{e_1} + \ldots + \widehat{A}_{n-1}\vec{e}_{n-1}\end{aligned}$$

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

$$\begin{aligned}A_k &= \widehat{A}_0\frac{(\omega_n^k)^0}{\sqrt{n}} + \widehat{A}_1\frac{(\omega_n^k)^1}{\sqrt{n}} + \ldots + \widehat{A}_{n-1}\frac{(\omega_n^k)^{n-1}}{\sqrt{n}} \\ &= \frac{1}{\sqrt{n}}\left(\widehat{A}_0 + \widehat{A}_1(\omega_n^k)^1 + \ldots + \widehat{A}_{n-1}(\omega_n^k)^{n-1}\right)\end{aligned}$$

- Thus, A_k is obtained by evaluating the polynomial

$$Q(x) = \frac{1}{\sqrt{n}}\left(\widehat{A}_0 + \widehat{A}_1 x + \ldots + \widehat{A}_{n-1}x^{n-1}\right)$$

  at $x = \omega_n^k$.
- But this is precisely the Inverse Discrete Fourier Transform of the sequence

$$\left(\widehat{A}_0, \widehat{A}_1, \ldots, \widehat{A}_{n-1}\right)$$

Optional Material: Interpretation of DFT

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

$$\vec{b}_0 = (1,0,0,0,\ldots,0),\ \vec{b}_1 = (0,1,0,0,\ldots,0),\ \ldots,\ \vec{b}_{n-1} = (0,0,0,0,\ldots,1)$$

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

- then

$$\vec{A} = (A_0, A_1, A_2, \ldots, A_{n-1})_{\mathcal{B}} = A_0\vec{b}_0 + A_1\vec{b}_1 + A_2\vec{b}_2 + \ldots + A_{n-1}\vec{b}_{n-1}$$

and also

$$\vec{A} = (P_A^*(\omega_n^0), P_A^*(\omega_n^{-1}), \ldots, P_A^*(\omega_n^{-(n-1)}))_{\mathcal{F}} = (\widehat{A}_0, \widehat{A}_1, \ldots, \widehat{A}_{n-1})_{\mathcal{F}}$$
$$= \widehat{A}_0\vec{e_0} + \widehat{A}_1\vec{e_1} + \ldots + \widehat{A}_{n-1}\vec{e}_{n-1}$$

- Thus, DFT is just a change of basis operation: it transforms the sequence of coordinates

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

of a vector $\vec{A}$ in the basis $\mathcal{B}$ into the sequence

$$\left(P_A^*(\omega_n^0), P_A(\omega_n^{-1}), \ldots, P_A(\omega_n^{-(n-1)})\right)_{\mathcal{F}} = (\widehat{A}_0, \widehat{A}_1, \ldots, \widehat{A}_{n-1})_{\mathcal{F}}$$

of the coordinates of the same vector in the basis $\mathcal{F}$.

Optional Material: Interpretation of DFT

- Basis $\mathcal{F}$ is very important!
- Let us look again at the equation

$$\begin{aligned} A_k &= \widehat{A}_0 \frac{(\omega_n^k)^0}{\sqrt{n}} + \widehat{A}_1 \frac{(\omega_n^k)^1}{\sqrt{n}} + \ldots + \widehat{A}_{n-1} \frac{(\omega_n^k)^{n-1}}{\sqrt{n}} \\ &= \frac{1}{\sqrt{n}} \left(\widehat{A}_0 + \widehat{A}_1 (\omega_n^k)^1 + \ldots + \widehat{A}_{n-1} (\omega_n^k)^{n-1} \right) \end{aligned}$$

- Assume that $A_k = A(k)$ is actually a sample of a signal $A(t)$ at a time instant k;
- Let us also consider $(\omega_n^k)^m = e^{i \frac{2\pi m k}{n}}$ to be a sample of the function $f_m(t) = e^{i \frac{2\pi m}{n} t}$ also at an instant k;
- Note that $f_m(t) = e^{i \frac{2\pi m}{n} t} = \cos\left(\frac{2\pi m}{n} t\right) + i \sin\left(\frac{2\pi m}{n} t\right)$;
- This means that the samples $A(k)$ of the signal $A(t)$ at instants $t = 0, 1, \ldots, n-1$ are represented as a linear combination of samples at these instants of sinusoids (also called pure harmonic oscillations) $\cos\left(\frac{2\pi m}{n} t\right)$ and $\sin\left(\frac{2\pi m}{n} t\right)$ of increasing frequencies $\frac{2\pi \cdot 1}{n}, \frac{2\pi \cdot 2}{n}, \ldots \frac{2\pi \cdot (n-1)}{n}$:

$$A(k) = \widehat{A}_0 f_0(k) + \widehat{A}_1 f_1(k) + \widehat{A}_2 f_2(k) + \ldots + \widehat{A}_{n-1} f_{n-1}(k)$$

Optional Material: Interpretation of DFT

- Thus, in a sense the absolute value $|\widehat{A}_m| = |P_A^*(\omega_n^{-m})|$ of the m^{th} coordinate $\widehat{A}_m$ in the basis $\mathcal{F}$ tells us "how much" of the sinusoids $\cos\left(\frac{2\pi m}{n}t\right)$ and $\sin\left(\frac{2\pi m}{n}t\right)$ are present in the signal $A(t)$.
- This is very important in very many fields, ranging from astronomy (detecting planets orbiting distant stars), electrical engineering (signal and image processing), to economics (looking for cycles in economic indicators).
- We call this *spectral analysis*.
- Dolby engineers will soon give you a guest lecture with a demo of cool applications of the FFT in audio processing.