



Jean Baptiste Joseph Fourier (1768–1830)

Fourier-Transformation klassisch (z.B. Signalverarbeitung):

- ▶ FOURIERS Idee: Darstellung von (periodischen) Funktionen als Überlagerung von einfachen periodischen Schwingungen verschiedener Frequenz und Intensität
- ▶ Dualität: Schwingungsphänomene (Wellenausbreitung) lassen sich darstellen sowohl im “Ortsbereich” als auch im “Frequenzbereich”
- ▶ Fourier-Transformation: Übergang von einer Funktionsdarstellung im Ortsbereich zu einer Darstellung im Frequenzbereich (und umgekehrt)
- ▶ Wichtige Anwendung: Filterung

▶ Fouriertransformation

$$\mathcal{F} : f(x) \mapsto F(s) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \cdot s \cdot x} dx = \mathcal{F}[f](s)$$

▶ Rücktransformation

$$\mathcal{F}_{\text{inv}} : F(s) \mapsto f(x) = \int_{-\infty}^{\infty} F(s) e^{2\pi i \cdot s \cdot x} ds = \mathcal{F}_{\text{inv}}[F](x)$$

▶ inverse Beziehung — spektrale Zerlegung

$$f(x) = \mathcal{F}_{\text{inv}}[\mathcal{F}[f]](x) = \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} f(y) e^{-2\pi i \cdot s \cdot y} dy \right] e^{2\pi i \cdot s \cdot x} ds$$

Die Funktion  $x \mapsto f(x)$  ist dargestellt als Überlagerung (Linearkombination) der periodischen Funktionen (“harmonischen Schwingungen”)  $x \mapsto e^{2\pi i \cdot s \cdot x}$  mit Frequenz  $2\pi s$ . Das innere Integral ist deren “Intensität”.

▶ Faltung

$$(f(x), g(x)) \mapsto (f \otimes g)(x) = \int_{-\infty}^{\infty} f(y) \cdot g(x - y) dy$$

Faltungsoperationen verwendet man, um Merkmale der einer gegebenen Funktion  $f$  mittels geeigneter “Test”-Funktionen  $g$  zu erkennen und zu bewerten.

▶ Faltungstheorem

$$\mathcal{F}[f \otimes g](s) = \mathcal{F}[f](s) \cdot \mathcal{F}[g](s)$$

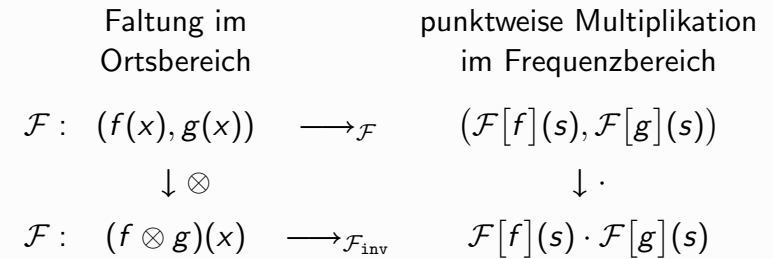
Konsequenz: die Faltung im “Ortsbereich” kann auch im “Frequenzbereich” durchgeführt werden.

### Beweis des Faltungstheorems

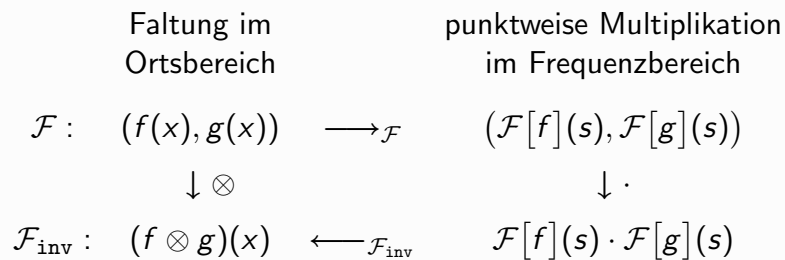
$$\begin{aligned}
 \mathcal{F}[f \otimes g](s) &= \mathcal{F}\left[\int_{-\infty}^{\infty} f(y) \cdot g(x-y) dy\right](s) \\
 &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(y) \cdot g(x-y) dy\right] e^{-2\pi i \cdot s \cdot x} dx \\
 &= \int_{-\infty}^{\infty} f(y) \left[\int_{-\infty}^{\infty} g(x-y) e^{-2\pi i \cdot s \cdot x} dx\right] dy \\
 &= \int_{-\infty}^{\infty} f(y) \left[\int_{-\infty}^{\infty} g(z) e^{-2\pi i \cdot s \cdot z} e^{-2\pi i \cdot s \cdot y} dz\right] dy \\
 &= \int_{-\infty}^{\infty} f(y) \left[\int_{-\infty}^{\infty} g(z) e^{-2\pi i \cdot s \cdot z} dz\right] e^{-2\pi i \cdot s \cdot y} dy \\
 &= \mathcal{F}[f](s) \cdot \mathcal{F}[g](s)
 \end{aligned}$$



### ► Schema des Faltungstheorems



### ► typische Anwendung des Faltungstheorems



### ► Analogie: Darstellung von Polynomen

- $\mathbb{K}[X]$ : Ring der Polynome in einer Variablen  $X$  und mit Koeffizienten im Körper  $\mathbb{K}$
- übliche Koeffizienten-Darstellung

$$a(X) = \sum_{i=0}^n a_i X^i \leftrightarrow \langle a_0, a_1, a_2, \dots, a_{n-1}, a_n \rangle \in \mathbb{K}^{n+1}$$

- $\mathbb{K}[X]_{\leq n}$ : Vektorraum der Polynome vom Grad  $\leq n$  hat als Standardbasis die  $n+1$  Polynome  $X^i$  ( $0 \leq i \leq n$ )
- algebraische Sicht:  $\langle a_0, a_1, a_2, \dots, a_{n-1}, a_n \rangle$  ist die Darstellung von  $a(X)$  bezüglich der Standardbasis
- der Vektorraum  $\mathbb{K}[X]_{\leq n}$  hat noch viele andere (und nützliche!) Basen



- ▶ Auswertung von Polynomen an einer Stelle  $y \in \mathbb{K}$ :

$$L_y : \mathbb{K}[X]_{\leq n} \rightarrow \mathbb{K} : \\ a(X) \mapsto a(y) = a_n y^n + a_{n-1} y^{n-1} + \dots + a_1 y + a_0 \\ = (\dots (a_n y + a_{n-1}) y + \dots + a_1) y + a_0$$

- ▶ praktisches Verfahren: Horner-Schema (optimal!) benötigt  $n$  Multiplikationen und  $n$  Additionen in  $\mathbb{K}$
- ▶ algebraisch: die Auswertungsabbildung  $L_y$  an der Stelle  $y \in \mathbb{K}$  ist ein Ring-Homomorphismus (Verträglichkeit mit Addition und Multiplikation)

- ▶ Rekonstruktion durch Interpolation  
ein Polynom  $n$ -ten Grades  $a(X) = \sum_{i=0}^n a_i X^i$  ist durch seine Werte  $a(y_i)$  ( $0 \leq i \leq n$ ) an  $n + 1$  verschiedenen Stellen  $\mathbf{y} = (y_0, y_1, y_2, \dots, y_n)$  eindeutig bestimmt [Bem.: dies gilt über einem Körper mit mindestens  $n + 1$  Elementen] denn das lineare Gleichungssystem mit Vandermonde-Matrix

$$\underbrace{\begin{pmatrix} 1 & y_0 & y_0^2 & \dots & y_0^n \\ 1 & y_1 & y_1^2 & \dots & y_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & y_n & y_n^2 & \dots & y_n^n \end{pmatrix}}_{V_n(y_0, y_1, \dots, y_n)} \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} a(y_0) \\ a(y_1) \\ \vdots \\ a(y_n) \end{pmatrix}$$

hat eine eindeutig bestimmte Lösung wegen

$$\det V_n(y_0, y_1, \dots, y_n) = \prod_{0 \leq i < j \leq n} (y_j - y_i) \neq 0$$

- ▶ simultane Auswertung an mehreren (verschiedenen) Stellen  $\mathbf{y} = (y_0, y_1, y_2, \dots, y_n) \in \mathbb{K}^{n+1}$

$$L_{\mathbf{y}} : \mathbb{K}[X]_{\leq n} \rightarrow \mathbb{K}^{n+1} : \\ a(X) \mapsto \langle L_{y_0}(a(X)), L_{y_1}(a(X)), L_{y_2}(a(X)), \dots, L_{y_n}(a(X)) \rangle \\ = \langle a(y_0), a(y_1), a(y_2), \dots, a(y_n) \rangle$$

- ▶ praktisches Verfahren:  $(n + 1)$ -mal Horner-Schema benötigt  $n(n + 1)$  Multiplikationen und  $n(n + 1)$  Additionen in  $\mathbb{K}$  (das ist nicht notwendig optimal!)
- ▶ algebraisch: die Auswertungsabbildung  $L_{\mathbf{y}}$  an den Stellen  $\mathbf{y} = (y_0, y_1, y_2, \dots, y_n) \in \mathbb{K}^{n+1}$  ist ein Ring-Homomorphismus (Verträglichkeit mit Addition und Multiplikation)

- ▶ Lösung des Gleichungssystems mittels Gauss-Elimination erfordert  $O(n^3)$  Operationen in  $\mathbb{K}$
- ▶ Effizienter: Interpolationsformel von Lagrange mit  $O(n^2)$  Operationen

$$a(X) = \sum_{i=0}^n a(y_i) \cdot e_i(X)$$

wobei

$$e_i(X) = \frac{\prod_{0 \leq j \leq n, j \neq i} (X - y_j)}{\prod_{0 \leq j \leq n, j \neq i} (y_i - y_j)} \quad \text{d.h.} \quad e_i(y_j) = \begin{cases} 1 & \text{falls } i = j \\ 0 & \text{falls } i \neq j \end{cases}$$

- ▶ Die Polynome  $e_i(X)$  ( $0 \leq i \leq n$ ) sind linear unabhängig und bilden eine Basis des Vektorraums  $\mathbb{K}[X]_{\leq n}$ .

► schematisch

$$\langle a_0, a_1, \dots, a_{n-1}, a_n \rangle \begin{array}{c} \xrightarrow{L_y = \text{Auswert.}} \\ \xleftarrow{L_y^{-1} = \text{Interpol.}} \end{array} \langle a(y_0), a(y_1), \dots, a(y_{n-1}), a(y_n) \rangle$$

► Folgerung:  $L_y$  ist ein Isomorphismus von Vektorräumen

$$\mathbb{K}[X]_{\leq n} \simeq \underbrace{\mathbb{K} \times \mathbb{K} \times \dots \times \mathbb{K}}_{n+1 \text{ Faktoren}}$$



► Für ein Objekt “Polynom vom Grad  $\leq n$ ” sind

► Koeffizientendarstellung

$$\langle a_0, a_1, \dots, a_{n-1}, a_n \rangle$$

► Wertedarstellung

$$\langle a(y_0), a(y_1), \dots, a(y_{n-1}), a(y_n) \rangle$$

Darstellungen in verschiedenen Basen des VR  $\mathbb{K}[X]_{\leq n}$ .

► Die Transformationen zwischen diesen Basen heissen “Auswertung” und “Interpolation”.



Reminiszenz: Chinesischer Restesatz, modulare Arithmetik

► Die Beziehung

Auswertung  $\leftrightarrow$  Interpolation

für Polynome erinnert nicht zufällig an den Chinesischen Restesatz für ganze Zahlen — es ist eine völlig analoge Situation im Bereich der Polynome

► Für Polynome über einem Körper  $\mathbb{K}$  gilt die Divisioneigenschaft

$$a(X) = b(X) \cdot q(X) + r(X)$$

wobei  $r(X) = 0$  oder  $\deg r(X) < \deg b(X)$ .

► Man schreibt:

$$r(X) = a(X) \bmod b(X) \quad \text{und} \quad b(X) \mid a(X), \quad \text{falls} \quad r(X) = 0$$

► Grösste gemeinsame Teiler, Eindeutigkeit der “Prim-Zerlegung” (“irreduzible” Polynome) funktionieren analog zu  $\mathbb{Z}$ .



► Algebraisch formuliert:  $\mathbb{K}[X]$  ist ein “euklidischer Ring”.

► Insbesondere: es gibt im Ring  $\mathbb{K}[X]$  einen euklidischen Algorithmus zur Berechnung grösster gemeinsamer Teiler

► Alle Folgerungen (z.B. Bézout-Beziehung) ergeben sich daraus exakt wie im Falle des Ringes  $\mathbb{Z}$ .

► Der euklidische Algorithmus für Polynome ist der Algorithmus der Polynomarithmetik schlechthin — auch für viele Anwendungen (z.B. effiziente Decodierverfahren für fehlerkorrigierende “zyklische” Codes)

► Wenn Sie mehr wissen wollen: besuchen Sie meine Vorlesungen über “Computeralgebra”!





- ▶ Polynommultiplikation mittels Auswertung und Interpolation
  - ▶ Gegeben Polynome  $a(X), b(X) \in \mathbb{K}[X]_{<n}$  in Koeffizientendarstellung

$$a(X) = \sum_{i=0}^{n-1} a_i X^i \quad b(X) = \sum_{i=0}^{n-1} b_i X^i$$

- ▶ Berechne die Koeffizientendarstellung des Produkts

$$a(X) \cdot b(X) = c(X) = \sum_{i=0}^{2n-1} c_i X^i$$

$$c_i = \sum_{j=0}^i a_j \cdot b_{i-j} \quad (0 \leq i \leq 2n-1)$$

- ▶ als Operation auf Vektoren (Faltung, convolution)

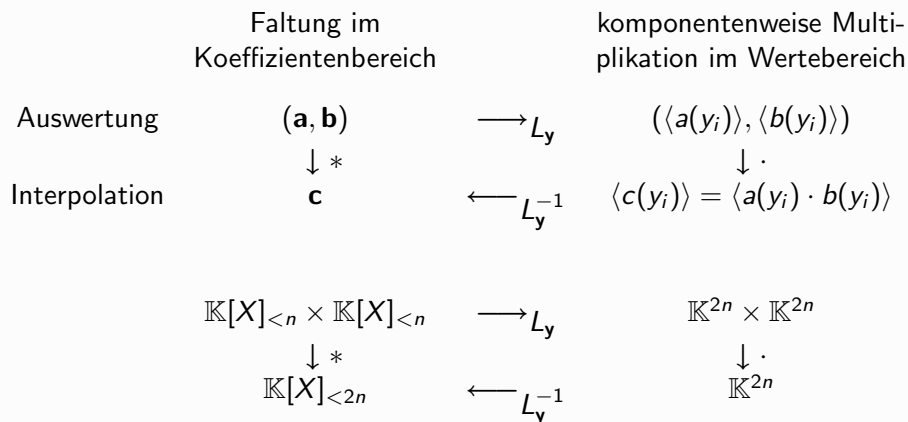
$$\left( \underbrace{\langle a_0, \dots, a_{n-1} \rangle}_{\mathbf{a}}, \underbrace{\langle b_0, \dots, b_{n-1} \rangle}_{\mathbf{b}} \right) \mapsto \underbrace{\langle c_0, \dots, c_{2n-1} \rangle}_{\mathbf{c}} = \mathbf{a} * \mathbf{b}$$



- ▶ traditionelle Lösung:  
Faltungsformel direkt berechnen  $\rightarrow O(n^2)$  Operationen in  $\mathbb{K}$
- ▶ alternative Lösung mittels "modularer Arithmetik"
  - ▶ Polynome  $a(X)$  und  $b(X)$  an  $2n$  Stützstellen  $\mathbf{y} = (y_0, \dots, y_{2n-1})$  auswerten
    - $\mathbf{a} = \langle a_0, \dots, a_{n-1} \rangle \mapsto L_{\mathbf{y}}(\mathbf{a}) = \langle a(y_0), \dots, a(y_{2n-1}) \rangle$
    - $\mathbf{b} = \langle b_0, \dots, b_{n-1} \rangle \mapsto L_{\mathbf{y}}(\mathbf{b}) = \langle b(y_0), \dots, b(y_{2n-1}) \rangle$
  - ▶ Multiplikation der Funktionswerte an den  $2n$  Stützstellen  $y_i$ 
    - $c(y_i) = a(y_i) \cdot b(y_i) \quad (0 \leq i < 2n)$
  - ▶ Rekonstruktion von  $c(X)$  durch die Funktionswerte an den Stützstellen
    - $\mathbf{c} = \langle c_0, \dots, c_{2n-1} \rangle = L_{\mathbf{y}}^{-1} \langle c(y_0), \dots, c(y_{2n-1}) \rangle$
  - ▶ Aufwand dieses Verfahrens:  $O(n^2)$  Operationen in  $\mathbb{K}$



- ▶ Schema der Polynommultiplikation mittels modularer Arithmetik



- ▶ Diskrete Fourier-Transformation

- ▶ Ursprünglich Approximation der kontinuierlichen Fouriertransformation für numerische Berechnungen
- ▶ Durchbruch bei der praktischen Verwendung durch Entdeckung der "Schnellen Fourier-Transformation" (FFT) durch COOLEY und TUKEY<sup>1</sup> — gehört seitdem zu den meistverwendeten Algorithmen überhaupt
- ▶ Das FFT-Prinzip war schon GAUSS bekannt (Notiz in seinem Tagebuch) und wurde auch später mehrmals entdeckt und publiziert<sup>2</sup>
- ▶ Durchbruch aber erst als Computer-Verfahren
- ▶ DFT bedeutet: Auswertung an "Einheitswurzeln", d.h. Lösungen der Gleichungen  $X^n = 1$  in  $\mathbb{C}$

<sup>1</sup>J. W. COOLEY and J. W. TUKEY, *An algorithm for the machine calculation of complex Fourier series*, Mathematics of Computation 19 (1965)

<sup>2</sup>C. RUNGE, R. KÖNIG *Vorlesungen über Numerisches Rechnen*, Springer (1924)





►  $n = 10$

$$\mathcal{R}_{10} = \mathcal{R}_2 \cup \mathcal{R}_5 \cup \left\{ \cos\left(\frac{k\pi}{10}\right) \pm i \sin\left(\frac{k\pi}{10}\right) ; k \in \{1, 3\} \right\}$$

►  $n = 11$

$$\mathcal{R}_{11} = \{1\} \cup \left\{ \cos\left(\frac{k\pi}{11}\right) \pm i \sin\left(\frac{k\pi}{11}\right) ; 1 \leq k \leq 5 \right\}$$

►  $n = 12$

$$\omega_{12} = \frac{\sqrt{3} + i}{2}$$

$$\mathcal{R}_{12} = \mathcal{R}_4 \cup \mathcal{R}_6 \cup \left\{ \pm \frac{\sqrt{3} \pm i}{2} \right\}$$

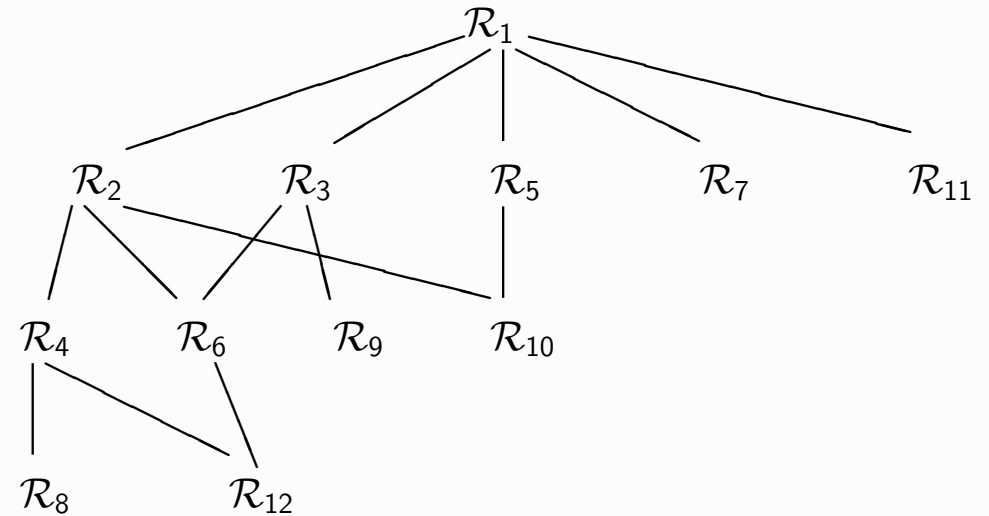
► (Komplexe) Diskrete Fouriertransformation der Ordnung  $n$ :

$$DFT_{n,\omega} : \left\{ \begin{array}{l} \langle a_0, a_1, \dots, a_{n-1} \rangle \rightarrow \langle a(\omega_n^0), a(\omega_n^1), \dots, a(\omega_n^{n-1}) \rangle \\ \mathbb{C}^n \rightarrow \mathbb{C}^n \end{array} \right.$$

wobei  $a(X) = \sum_{i=0}^{n-1} a_i X^i$

►  $DFT_{n,\omega}$  ist simultane Auswertung von Polynomen  $\in \mathbb{C}[X]_{<n}$  an den  $n$ -ten Einheitswurzeln

$$DFT_{n,\omega} = L_{\langle \omega_n^0, \omega_n^1, \dots, \omega_n^{n-1} \rangle}$$



► DFT als lineare Transformation

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_j \\ \vdots \\ a_{n-1} \end{pmatrix} \mapsto \underbrace{\begin{pmatrix} 1 & 1 & 1 & \dots & 1 & \dots & 1 \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^j & \dots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \dots & \omega_n^{2j} & \dots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & \omega_n^j & \omega_n^{2j} & \dots & \omega_n^{jj} & \dots & \omega_n^{j(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{j(n-1)} & \dots & \omega_n^{(n-1)^2} \end{pmatrix}}_{V_n(\omega_n^0, \omega_n^1, \omega_n^2, \dots, \omega_n^{n-1})} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_j \\ \vdots \\ a_{n-1} \end{pmatrix}$$

► Beachten:  $\omega_n$  hat die Ordnung  $n$  in  $\mathcal{R}_n$ , also  $\omega_n^{i \cdot j} = \omega_n^{i \cdot j \bmod n}$

►  $n = 2$

$$DFT_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

►  $n = 3$

$$DFT_3 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \omega_3 & \omega_3^2 \\ 1 & \omega_3^2 & \omega_3^4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \omega_3 & \omega_3^2 \\ 1 & \omega_3^2 & \omega_3 \end{bmatrix}$$

mit  $\omega_3 = \frac{-1+i\sqrt{3}}{2}, \omega_3^2 = \frac{-1-i\sqrt{3}}{2}$

►  $n = 4$

$$DFT_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^0 & i^2 \\ 1 & i^3 & i^2 & i^1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}$$



►  $n = 5$

$$DFT_5 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \omega_5 & \omega_5^2 & \omega_5^3 & \omega_5^4 \\ 1 & \omega_5^2 & \omega_5^4 & \omega_5^6 & \omega_5^8 \\ 1 & \omega_5^3 & \omega_5^6 & \omega_5^9 & \omega_5^{12} \\ 1 & \omega_5^4 & \omega_5^8 & \omega_5^{12} & \omega_5^{16} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \omega_5 & \omega_5^2 & \omega_5^3 & \omega_5^4 \\ 1 & \omega_5^2 & \omega_5^4 & \omega_5 & \omega_5^3 \\ 1 & \omega_5^3 & \omega_5 & \omega_5^4 & \omega_5^2 \\ 1 & \omega_5^4 & \omega_5^3 & \omega_5^2 & \omega_5 \end{bmatrix}$$

mit

$$\omega_5 = \frac{\sqrt{5} - 1 + i\sqrt{2}\sqrt{5 + \sqrt{5}}}{4}$$



►  $n = 6$

$$DFT_6 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega_6 & \omega_6^2 & \omega_6^3 & \omega_6^4 & \omega_6^5 \\ 1 & \omega_6^2 & \omega_6^4 & \omega_6^6 & \omega_6^8 & \omega_6^{10} \\ 1 & \omega_6^3 & \omega_6^6 & \omega_6^9 & \omega_6^{12} & \omega_6^{15} \\ 1 & \omega_6^4 & \omega_6^8 & \omega_6^{12} & \omega_6^{16} & \omega_6^{20} \\ 1 & \omega_6^5 & \omega_6^{10} & \omega_6^{15} & \omega_6^{20} & \omega_6^{25} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega_6 & \omega_6^2 & -1 & \omega_6^4 & \omega_6^5 \\ 1 & \omega_6^2 & \omega_6^4 & 1 & \omega_6^2 & \omega_6^4 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & \omega_6^4 & \omega_6^2 & 1 & \omega_6^4 & \omega_6^2 \\ 1 & \omega_6^5 & \omega_6^4 & 1 & \omega_6^2 & \omega_6 \end{bmatrix}$$

mit

$$\omega_6 = \frac{1+i\sqrt{3}}{2}$$

$$\omega_6^2 = \frac{-1+i\sqrt{3}}{2}$$

$$\omega_6^4 = \frac{-1-i\sqrt{3}}{2} = -\omega_6$$

$$\omega_6^5 = \frac{1-i\sqrt{3}}{2} = -\omega_6^2$$



►  $n = 8$

$$DFT_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega_8 & i & \omega_8^3 & -1 & \omega_8^5 & -i & \omega_8^7 \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & \omega_8^3 & -i & \omega_8 & -1 & \omega_8^7 & i & \omega_8^5 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & \omega_8^5 & i & \omega_8^7 & -1 & \omega_8 & -i & \omega_8^3 \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & \omega_8^7 & -i & \omega_8^5 & -1 & \omega_8^3 & i & \omega_8 \end{bmatrix}$$

mit

$$\omega_8 = \frac{1+i}{\sqrt{2}}$$

$$\omega_8^3 = \frac{-1+i}{\sqrt{2}} = \omega_8 - \sqrt{2}$$

$$\omega_8^5 = \frac{-1-i}{\sqrt{2}} = -\omega_8$$

$$\omega_8^7 = \frac{1-i}{\sqrt{2}} = \omega_8 - i\sqrt{2}$$



► Faktorisierung von  $DFT_4$

- Spalten vertauschen

$$DFT_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \mapsto \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & i & -i \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -i & i \end{bmatrix} = \widetilde{DFT}_4$$

- Blockstruktur erkennen

$$\left[ \begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & -1 & i & -i \\ \hline 1 & 1 & -1 & -1 \\ 1 & -1 & -i & i \end{array} \right] = \left[ \begin{array}{c|c} DFT_2 & \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} DFT_2 \\ \hline DFT_2 & \begin{pmatrix} -1 & 0 \\ 0 & -i \end{pmatrix} DFT_2 \end{array} \right]$$

► Faktorisierung von  $DFT_4$

- Faktorisierung erkennen

$$\widetilde{DFT}_4 = \left[ \begin{array}{c|c} I_2 & \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \\ \hline I_2 & -\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \end{array} \right] \cdot \left[ \begin{array}{c|c} DFT_2 & 0_2 \\ \hline 0_2 & DFT_2 \end{array} \right]$$

wobei  $I_2 =$  Einheitsmatrix,  $0_2 =$  Nullmatrix

► Faktorisierung von  $DFT_8$

- Spalten in  $DFT_8$  vertauschen

$$\widetilde{DFT}_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i & \omega_8 & \omega_8^3 & \omega_8^5 & \omega_8^7 \\ 1 & -1 & 1 & -1 & i & -i & i & -i \\ 1 & -i & -1 & i & \omega_8^3 & \omega_8 & \omega_8^7 & \omega_8^5 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & i & -1 & -i & \omega_8^5 & \omega_8^7 & \omega_8 & \omega_8^3 \\ 1 & -1 & 1 & -1 & -i & i & -i & i \\ 1 & -i & -1 & i & \omega_8^7 & \omega_8^5 & \omega_8^3 & \omega_8 \end{bmatrix}$$

► Faktorisierung von  $DFT_8$

- Blockstruktur erkennen

$$\widetilde{DFT}_8 = \left[ \begin{array}{c|c} DFT_4 & \begin{bmatrix} 1 & & & \\ & \omega_8 & & \\ & & \omega_8^2 & \\ & & & \omega_8^3 \end{bmatrix} \cdot DFT_4 \\ \hline DFT_4 & \begin{bmatrix} \omega_8^4 & & & \\ & \omega_8^5 & & \\ & & \omega_8^6 & \\ & & & \omega_8^7 \end{bmatrix} \cdot DFT_4 \end{array} \right]$$

- Faktorisierung erkennen

$$\widetilde{DFT}_8 = \begin{bmatrix} I_4 & D_4 \\ I_4 & -D_4 \end{bmatrix} \cdot \begin{bmatrix} DFT_4 & 0_4 \\ 0_4 & DFT_4 \end{bmatrix}$$

► Faktorisierung von  $DFT_{2n}$

$$\widetilde{DFT}_{2n} = \begin{bmatrix} DFT_n & \begin{bmatrix} 1 & & & \\ & \omega_{2n} & & \\ & & \ddots & \\ & & & \omega_{2n}^{n-1} \end{bmatrix} \cdot DFT_n \\ DFT_n & \begin{bmatrix} \omega_{2n}^n & & & \\ & \omega_{2n}^{n+1} & & \\ & & \ddots & \\ & & & \omega_{2n}^{2n-1} \end{bmatrix} \cdot DFT_n \end{bmatrix}$$

$$= \begin{bmatrix} I_n & D_n \\ I_n & -D_n \end{bmatrix} \cdot \begin{bmatrix} DFT_n & 0_n \\ 0_n & DFT_n \end{bmatrix}$$



► Dabei ist

$$D_n = \begin{bmatrix} 1 & & & \\ & \omega_{2n} & & \\ & & \ddots & \\ & & & \omega_{2n}^{n-1} \end{bmatrix}$$

$$-D_n = \omega_{2n}^n \cdot D_n = \begin{bmatrix} \omega_{2n}^n & & & \\ & \omega_{2n}^{n+1} & & \\ & & \ddots & \\ & & & \omega_{2n}^{2n-1} \end{bmatrix}$$



► Beachte: in der Gleichung

$$\widetilde{DFT}_{2n} = \begin{bmatrix} I_n & D_n \\ I_n & -D_n \end{bmatrix} \cdot \begin{bmatrix} DFT_n & 0_n \\ 0_n & DFT_n \end{bmatrix}$$

- hat die Matrix  $\widetilde{DFT}_{2n}$   $4n^2$  Koeffizienten  $\neq 0$
- haben die Matrizen auf der rechten Seite  $4n + 2n^2$  Koeffizienten  $\neq 0$
- FFT: diese Zerlegungs idee *rekursiv* durchführen
- Folgerung: Reduktion der Anzahl der Koeffizienten  $\neq 0$  von  $O(n^2)$  auf  $O(n \log n)$



► FFT: der entscheidende "divide-and-conquer"-Trick:

Rückführung von  $DFT_{2n, \omega_{2n}}$  auf  $DFT_{n, \omega_{2n}^2}$

- $\omega = \omega_{2n}$  : primitive  $2n$ -te Einheitswurzel, d.h.

$$\mathcal{R}_{2n} = \{1 = \omega^0, \omega, \omega^2, \dots, \omega^{2n-1}\}$$

- $\eta = (\omega_{2n})^2 = \omega_n$  : primitive  $n$ -te Einheitswurzel, d.h.

$$\mathcal{R}_n = \{1 = \eta^1, \eta, \eta^2, \dots, \eta^{n-1}\}$$

- Inklusion:  $\mathcal{R}_n \subset \mathcal{R}_{2n}$
- Genauer: unter der Abbildung  $x \mapsto x^2$  wird  $\mathcal{R}_{2n}$  2-zu-1 auf  $\mathcal{R}_n$  abgebildet:

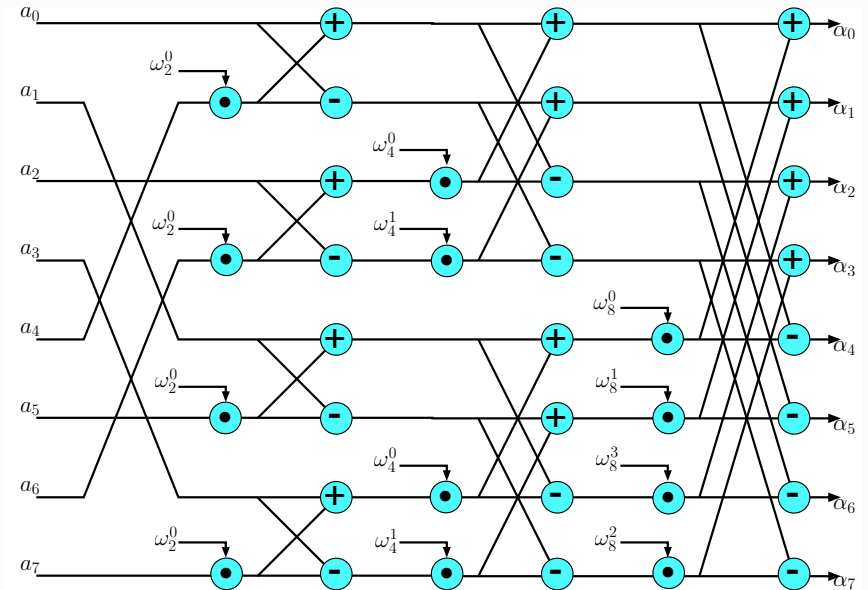
$$\omega^k, \omega^{n+k} \mapsto \eta^k \quad (0 \leq k < n)$$





```

FFT := proc (A,k)
n := 2^k;
if k=0 then RETURN(A) fi;
omega_n := exp(2*Pi*I/n);
omega := 1;
a_0 := [A[0],A[2],...,A[n-2]];
a_1 := [A[1],A[3],...,A[n-1]];
y_0 := FFT(a_0,k-1);
y_1 := FFT(a_1,k-1);
for t from 0 to (n/2)-1 do
y[t] := y_0[t] + omega*y_1[t];
y[t+(n/2)] := y_0[t] - omega*y_1[t];
omega := omega*omega_n
od;
RETURN(y);
end;
    
```



► Komplexität der FFT

- $F(n)$  : Anzahl der komplexen arithmetischen Operationen zur Berechnung von  $DFT(n)$  mittels FFT
- "divide-and-conquer"-Rekursionsgleichung

$$F(2n) = 2 \cdot F(n) + O(n), \quad F(1) = 0$$

- Lösung:

$$F(n) \in \Theta(n \cdot \log n)$$

► Umkehrabbildung

$$DFT(n)^{-1} = \text{"Interpolation" an den Stellen } \omega_n^k \in \mathcal{R}_n$$

► Inverses der Vandermonde-Matrix von  $DFT(n)$

$$V_n = V_n(\omega_n^0, \omega_n^1, \dots, \omega_n^{n-1}) = (\omega_n^{ij})_{0 \leq i,j < n}$$

ist

$$\frac{1}{n} V_n(\bar{\omega}_n^0, \bar{\omega}_n^1, \dots, \bar{\omega}_n^{n-1}) = \left( \frac{1}{n} \bar{\omega}_n^{ij} \right)_{0 \leq i,j < n} = \left( \frac{1}{n} \omega_n^{-ij} \right)_{0 \leq i,j < n}$$

wobei  $\bar{\omega}_n = e^{-2\pi i/n} = \omega_n^{-1}$

- Beweis: für  $0 \leq i, k < n$  gilt

$$\sum_{j=0}^{n-1} \omega_n^{ij} \cdot \bar{\omega}_n^{jk} = \sum_{j=0}^{n-1} \omega_n^{ij} \cdot \omega_n^{-jk} = \sum_{j=0}^{n-1} \omega_n^{(i-k)j} = \begin{cases} n & i = k \\ 0 & i \neq k \end{cases}$$

▶ Folgerung:

- ▶ Die Rücktransformation  $DFT(n)^{-1}$  ist
  - bis auf banale Änderungen:  
 $\omega_n$  durch  $\omega_n^{-1}$  ersetzen und alles durch  $n$  dividieren —
  - die gleiche Operation wie  $DFT(n)$
- ▶  $DFT(n)^{-1}$  kann also ebenfalls mit  $\Theta(n \log n)$  komplexen Operationen berechnet werden
- ▶ Man kann die gleichen Programme/Hardware verwenden

▶ Ergänzende Bemerkungen

- ▶ FFT kombiniert zwei Prinzipien algebraischer Natur:
  - Domain-Transformation (Koeffizientendarstellung vs. Wertedarstellung) mittels Evaluation  $\leftrightarrow$  Interpolation
  - Rekursive Struktur der Einheitswurzeln als Lösungen von  $X^n = 1$  und damit als Elemente einer zyklischen Gruppe
- ▶ Zahlreiche Varianten und Variationen über Ringen, die genügend viele Einheitswurzeln enthalten, darunter
  - Ringe  $\mathbb{Z}_n$  für geeignete  $n$
  - endliche Körper (bestehen vollständig aus Einheitswurzeln!)
  - andere Zerlegungen als nach Zweierpotenzen
- ▶ Beim Rechnen über  $\mathbb{C}$ : Rundungsfehler, numerische Stabilität? Andere Ringe/Körper mit exakter Arithmetik oft vorteilhaft
- ▶ Schnellstes bekanntes Multiplikationsverfahren für ganze Zahlen (SCHÖNHAGE-STRASSEN,  $O(n \log n \log \log n)$ ) beruht auf FFT über Ringen  $\mathbb{Z}_{2^k+1}$

- ▶ Anwendung: Multiplikation (“Faltung”) von zwei Polynomen  $a, b \in \mathbb{C}[X]_{<n}$  nach dem Schema der modularen Arithmetik
  - ▶ jeweils  $2n$  Auswertungen mittels FFT

$$a(\omega_{2n}^k), b(\omega_{2n}^k) \quad (0 \leq k < 2n)$$

- ▶  $2n$  komplexe Multiplikationen

$$c(\omega_{2n}^k) = a(\omega_{2n}^k) \cdot b(\omega_{2n}^k) \quad (0 \leq k < 2n)$$

- ▶ Interpolation mittels FFT-Rücktransformation

$$c(X) = DFT(2n)^{-1}([c(\omega_{2n}^0), \dots, [c(\omega_{2n}^{2n-1})]])$$

erfordert  $3 \cdot \Theta(2n \cdot \log 2n) + O(2n) \in \Theta(n \cdot \log n)$  komplexe Operationen, ist also deutlich schneller als das klassische  $O(n^2)$ -Verfahren