# How to Write Fast Numerical Code

Spring 2012
Lecture 19

**Instructor:** Markus Püschel

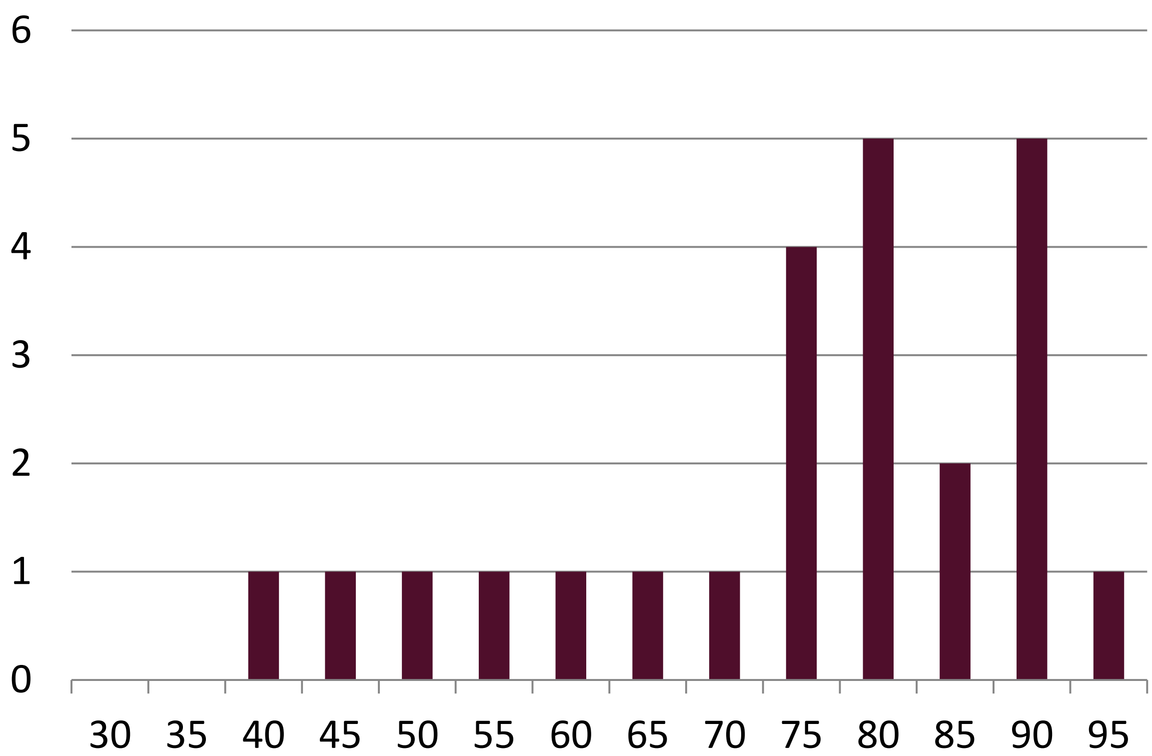
**TAs:** Georg Ofenbeck & Daniele Spampinato

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

# Miscellaneous

- Roofline tool
- Project report etc. online
- Midterm (solutions online)



# Planning

**May 2012**

| Sunday | Monday | Tuesday | Wednesday | Thursday | Friday | Saturday |
|---|---|---|---|---|---|---|
| 29 | 30 | 1 | 2 | 3 | 4 | 5 |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 27 | 28 | 29 | 30 | 31 | 1 | 2 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |

**Today**

**Lecture**

**Project meetings**

**Project presentations**

- 10 minutes each
- random order
- random speaker

*Reports due ~7-10 days after semester end*

# Linear Transforms

$$\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{pmatrix} = y = Tx \quad \longleftarrow \quad \boxed{T\cdot} \quad \longleftarrow \quad x = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix}$$

***Output*** ***Input***

***Example:*** $T = \mathbf{DFT}_n$

$$\begin{aligned} T = \mathbf{DFT}_n &= [e^{-2k\ell\pi i/n}]_{0\leq k,\ell<n} \\ &= [\omega_n^{k\ell}]_{0\leq k,\ell<n}, \quad \omega_n = e^{-2\pi i/n} \end{aligned}$$

# Algorithms: Example FFT, n = 4

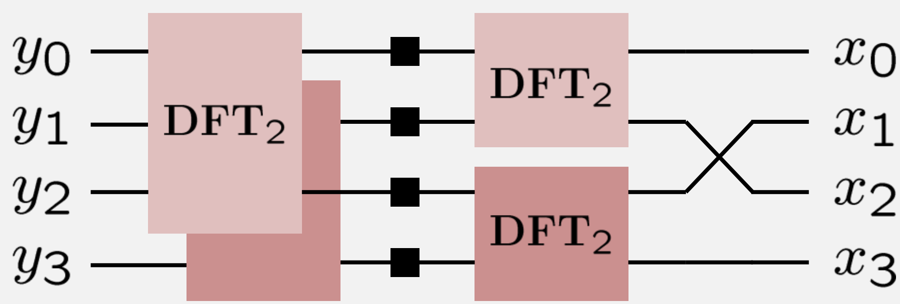
***Fast Fourier transform (FFT)***

$$\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & i & -1 & -i\\ 1 & -1 & 1 & -1\\ 1 & -i & -1 & i\end{bmatrix} x = \begin{bmatrix}1 & \cdot & 1 & \cdot\\ \cdot & 1 & \cdot & 1\\ 1 & \cdot & -1 & \cdot\\ \cdot & 1 & \cdot & -1\end{bmatrix} \begin{bmatrix}1 & \cdot & \cdot & \cdot\\ \cdot & 1 & \cdot & \cdot\\ \cdot & \cdot & 1 & \cdot\\ \cdot & \cdot & \cdot & i\end{bmatrix} \begin{bmatrix}1 & 1 & \cdot & \cdot\\ 1 & -1 & \cdot & \cdot\\ \cdot & \cdot & 1 & 1\\ \cdot & \cdot & 1 & -1\end{bmatrix} \begin{bmatrix}1 & \cdot & \cdot & \cdot\\ \cdot & \cdot & 1 & \cdot\\ \cdot & 1 & \cdot & \cdot\\ \cdot & \cdot & \cdot & 1\end{bmatrix} x$$

***Representation using matrix algebra***

$$\mathrm{DFT}_4 = (\mathrm{DFT}_2 \otimes \mathrm{I}_2)\,\mathrm{diag}(1,1,1,i)(\mathrm{I}_2 \otimes \mathrm{DFT}_2)\,\mathrm{L}^4_2$$

***Data flow graph***

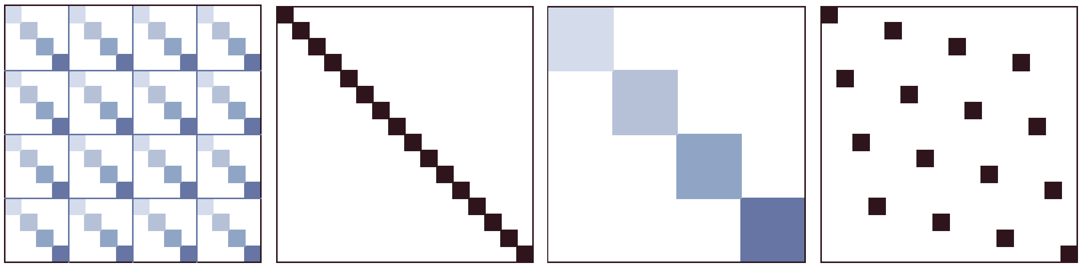


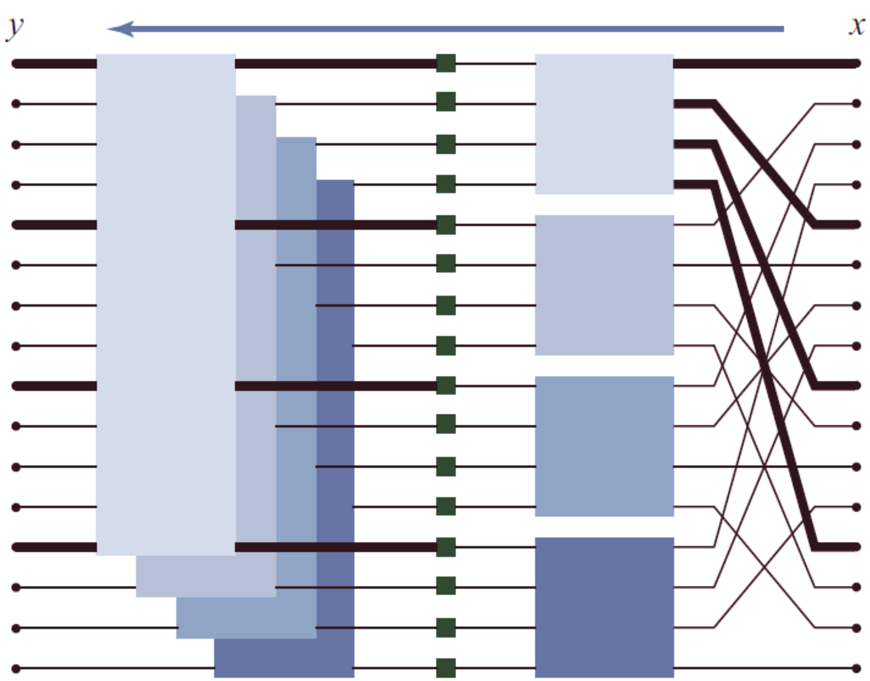
# Cooley-Tukey FFT (Recursive, General-Radix)

- **Blackboard**

# Example FFT, n = 16 *(Recursive, Radix 4)*

$$\mathrm{DFT}_{16} = (\mathrm{DFT}_4 \otimes I_4)\, T_4^{16}\, (I_4 \otimes \mathrm{DFT}_4)\, L_4^{16}$$





# Recursive Cooley-Tukey FFT

*radix*

$$\mathbf{DFT}_{km} = (\mathbf{DFT}_k \quad \mathrm{I}_m) T_m^{km} (\mathrm{I}_k \quad \mathbf{DFT}_m) L_k^{km}$$ *decimation-in-time*

$$\mathbf{DFT}_{km} = L_m^{km} (\mathrm{I}_k \quad \mathbf{DFT}_m) T_m^{km} (\mathbf{DFT}_k \quad \mathrm{I}_m)$$ *decimation-in-frequency*

- **For powers of two n = $2^t$ sufficient together with base case**

$$\mathbf{DFT}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

# Recursive vs. Iterative FFT

- **Recursive, radix-k Cooley-Tukey FFT**

$$\mathbf{DFT}_{km} = (\mathbf{DFT}_k \quad \mathrm{I}_m) T_m^{km} (\mathrm{I}_k \quad \mathbf{DFT}_m) L_k^{km}$$

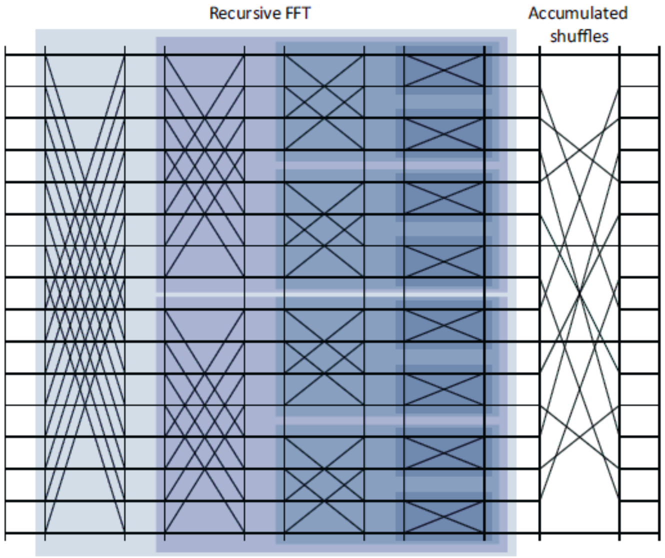
$$\mathbf{DFT}_{km} = L_m^{km} (\mathrm{I}_k \quad \mathbf{DFT}_m) T_m^{km} (\mathbf{DFT}_k \quad \mathrm{I}_m)$$

- **Iterative, radix 2, decimation-in-time/decimation-in-frequency**

$$\mathbf{DFT}_{2^t} = \left( \prod_{j=1}^{t} (\mathrm{I}_{2^{j-1}} \quad \mathbf{DFT}_2 \quad \mathrm{I}_{2^{t-j}}) \cdot (\mathrm{I}_{2^{j-1}} \quad T_{2^{t-j}}^{2^{t-j+1}}) \right) \cdot R_{2^t}$$

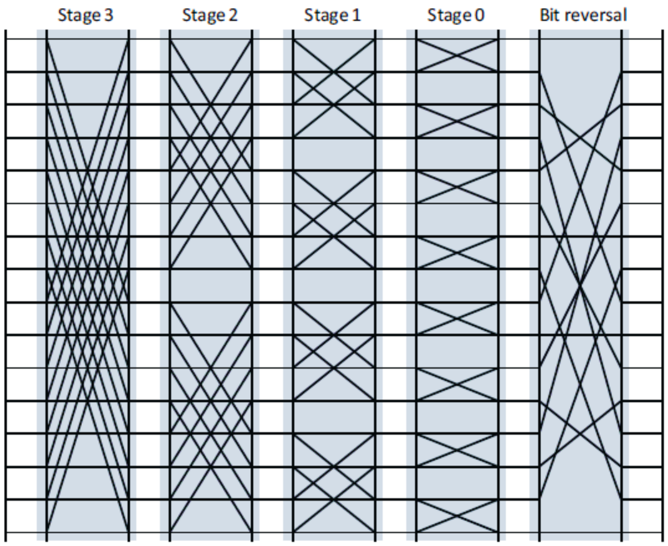
$$\mathbf{DFT}_{2^t} = R_{2^t} \cdot \left( \prod_{j=1}^{t} (\mathrm{I}_{2^{t-j}} \quad T_{2^{j-1}}^{2^j}) \cdot (\mathrm{I}_{2^{t-j}} \quad \mathbf{DFT}_2 \quad \mathrm{I}_{2^{j-1}}) \right)$$

## *Radix 2, recursive*




$$(\mathrm{DFT}_2 \otimes I_8)T_8^{16}\Big(I_2 \otimes \Big((\mathrm{DFT}_2 \otimes I_4)T_4^8(I_2 \otimes ((\mathrm{DFT}_2 \otimes I_2)T_2^4(I_2 \otimes \mathrm{DFT}_2)L_2^4))L_2^8\Big)\Big)L_2^{16}$$

## *Radix 2, iterative*




$$\Big((I_1 \otimes \mathrm{DFT}_2 \otimes I_8)D_0^{16}\Big)\Big((I_2 \otimes \mathrm{DFT}_2 \otimes I_4)D_1^{16}\Big)\Big((I_4 \otimes \mathrm{DFT}_2 \otimes I_2)D_2^{16}\Big)\Big((I_8 \otimes \mathrm{DFT}_2 \otimes I_1)D_3^{16}\Big)R_2^{16}$$

# Recursive vs. Iterative

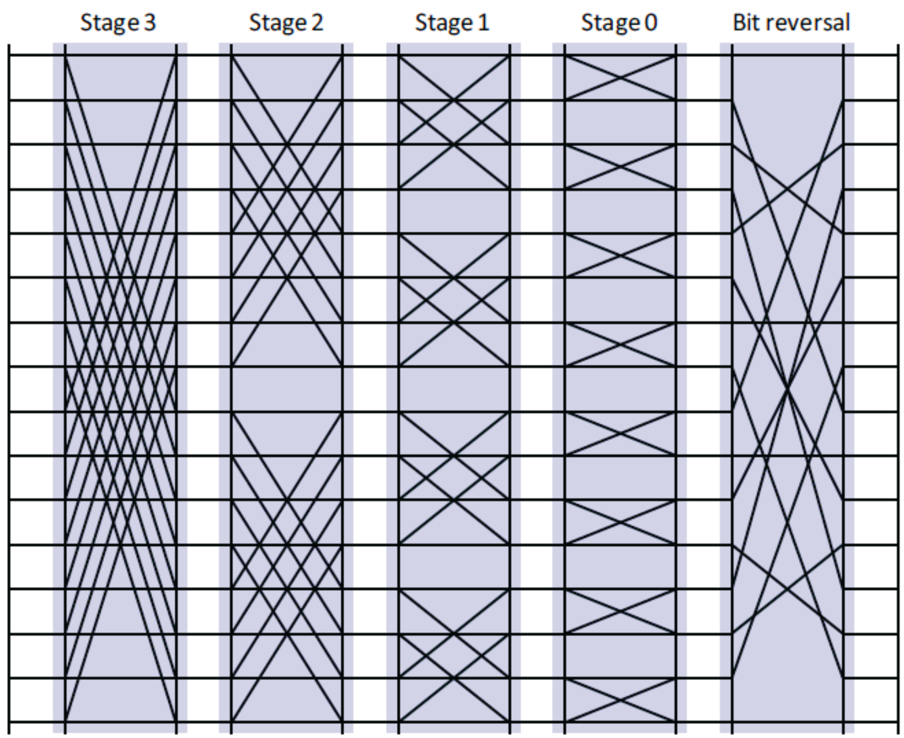
- **Iterative FFT computes in stages of butterflies = $\log_2(n)$ passes through the data**
- **Recursive FFT reduces passes through data = better locality**
- **Same computation graph but different topological sorting**

- **Rough analogy:**

| MMM | DFT |
|---|---|
| Triple loop | Iterative FFT |
| Blocked | Recursive FFT |

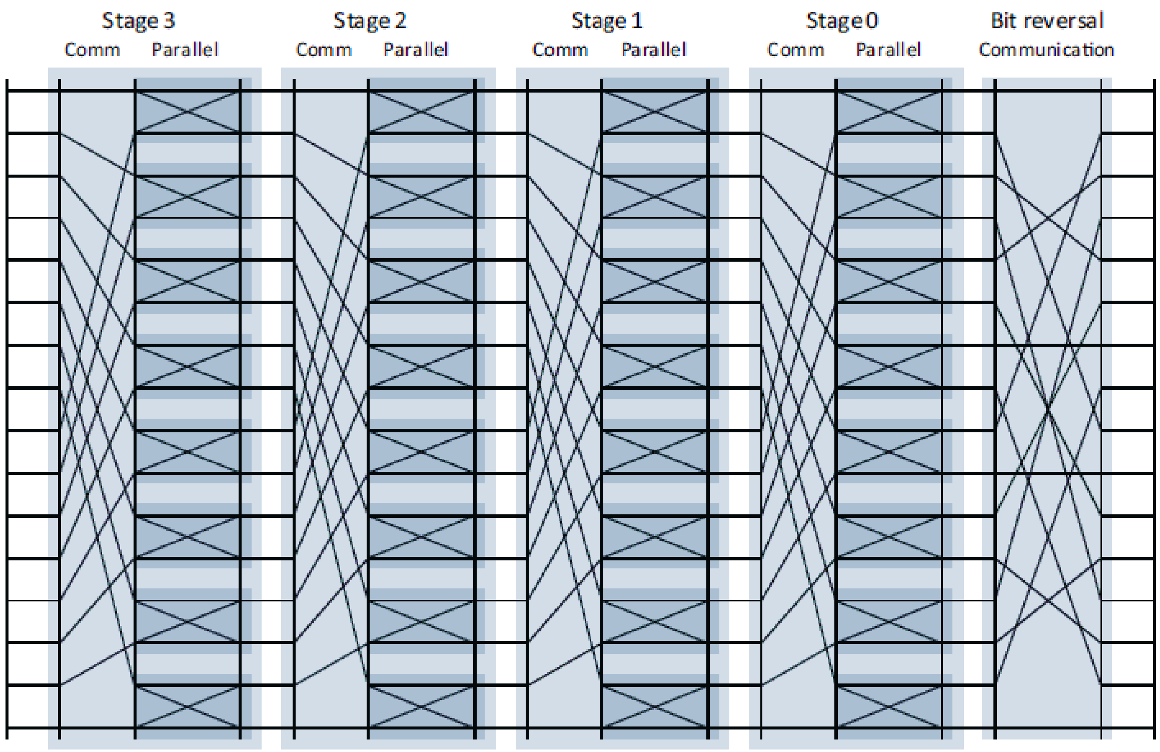
# The FFT Is Very Malleable

# Iterative FFT, Radix 2



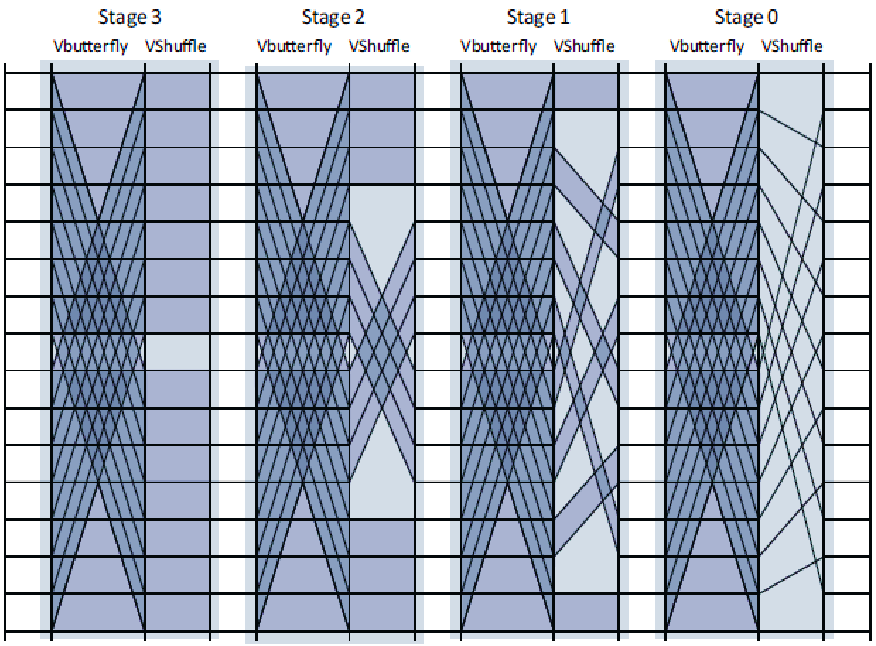

$$\left(\left(I_1 \otimes \mathrm{DFT}_2 \otimes I_8\right) D_0^{16}\right)\left(\left(I_2 \otimes \mathrm{DFT}_2 \otimes I_4\right) D_1^{16}\right)\left(\left(I_4 \otimes \mathrm{DFT}_2 \otimes I_2\right) D_2^{16}\right)\left(\left(I_8 \otimes \mathrm{DFT}_2 \otimes I_1\right) D_3^{16}\right) R_2^{16}$$

# Pease FFT, Radix 2



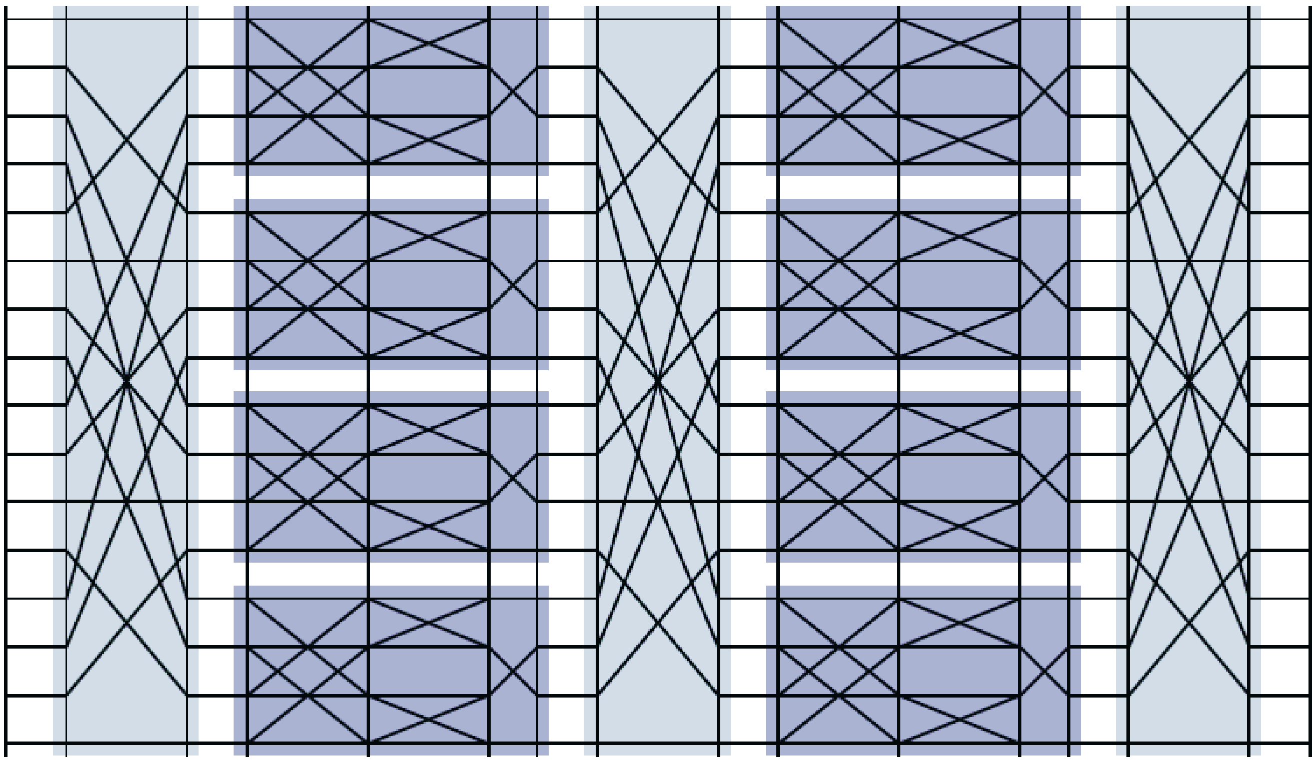

$$\left(L_2^{16}\left(I_8 \otimes \mathrm{DFT}_2\right) D_0^{16}\right)\left(L_2^{16}\left(I_8 \otimes \mathrm{DFT}_2\right) D_1^{16}\right)\left(L_2^{16}\left(I_8 \otimes \mathrm{DFT}_2\right) D_2^{16}\right)\left(L_2^{16}\left(I_8 \otimes \mathrm{DFT}_2\right) D_3^{16}\right) R_2^{16}$$

# Stockham FFT, Radix 2



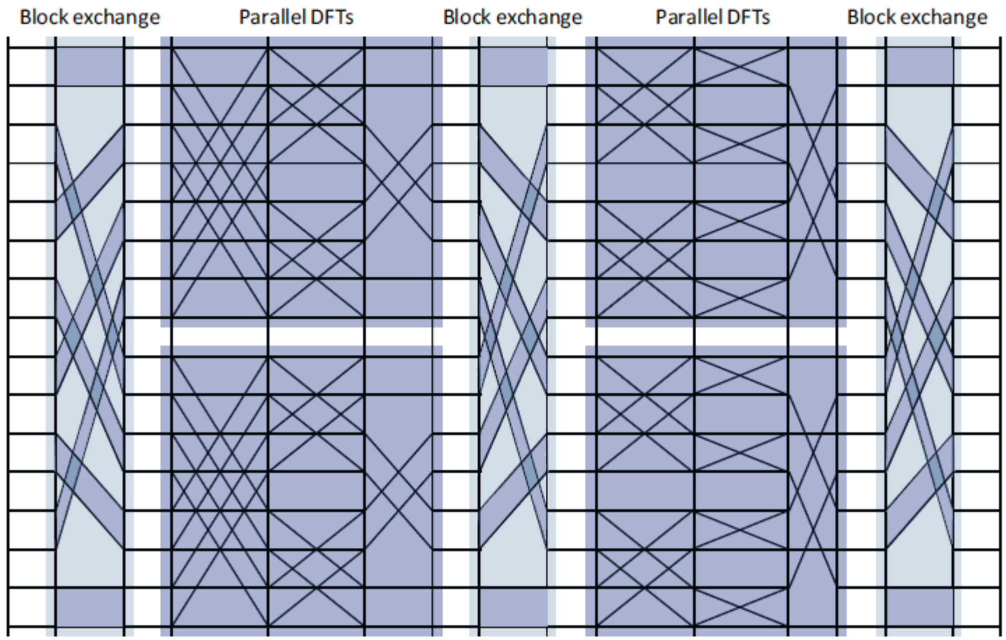

$$\Big(\big(\mathrm{DFT}_2 \otimes I_8\big) D_0^{16} \big(L_2^2 \otimes I_8\big)\Big)\Big(\big(\mathrm{DFT}_2 \otimes I_8\big) D_1^{16} \big(L_2^4 \otimes I_4\big)\Big)\Big(\big(\mathrm{DFT}_2 \otimes I_8\big) D_2^{16} \big(L_2^8 \otimes I_2\big)\Big)\Big(\big(\mathrm{DFT}_2 \otimes I_8\big) D_3^{16} \big(L_2^{16} \otimes I_1\big)\Big)$$

# Six-Step FFT




$$L_4^{16}\Big(I_4 \otimes \big((\mathrm{DFT}_2 \otimes I_2)T_2^4(I_2 \otimes \mathrm{DFT}_2)L_2^4\big)\Big)L_4^{16}T_4^{16}\Big(I_4 \otimes \big((\mathrm{DFT}_2 \otimes I_2)T_2^4(I_2 \otimes \mathrm{DFT}_2)L_2^4\big)\Big)L_4^{16}$$

# Multi-Core FFT




$$\left(L^8_4 \otimes I_2\right) \left(I_2 \otimes \left((\mathrm{DFT}_2 \otimes I_2) T^4_2 (I_2 \otimes \mathrm{DFT}_2) L^4_2\right) \otimes I_2\right) \left(L^8_2 \otimes I_2\right) T^{16}_4 \left(I_2 \otimes \left(I_2 \otimes (\mathrm{DFT}_2 \otimes I_2) T^4_2 (I_2 \otimes \mathrm{DFT}_2)\right) R^8_2\right) \left(L^8_2 \otimes I_2\right)$$

# Transform Algorithms

$$\mathbf{DFT}_n \to P_{k/2,2m}^\top \left(\mathbf{DFT}_{2m} \oplus \left(I_{k/2-1} \quad {}_i\, C_{2m}\, \mathbf{rDFT}_{2m}(i/k)\right)\right)\left(\mathbf{RDFT}'_k \quad I_m\right), \quad k \text{ even},$$

$$\begin{vmatrix}\mathbf{RDFT}_n\\ \mathbf{RDFT}'_n\\ \mathbf{DHT}_n\\ \mathbf{DHT}'_n\end{vmatrix} \to (P_{k/2,m}^\top \quad I_2)\left(\begin{vmatrix}\mathbf{RDFT}_{2m}\\ \mathbf{RDFT}'_{2m}\\ \mathbf{DHT}_{2m}\\ \mathbf{DHT}'_{2m}\end{vmatrix} \oplus \left(I_{k/2-1} \quad {}_i\, D_{2m} \begin{vmatrix}\mathbf{rDFT}_{2m}(i/k)\\ \mathbf{rDFT}_{2m}(i/k)\\ \mathbf{rDHT}_{2m}(i/k)\\ \mathbf{rDHT}_{2m}(i/k)\end{vmatrix}\right)\right)\left(\begin{vmatrix}\mathbf{RDFT}'_k\\ \mathbf{RDFT}'_k\\ \mathbf{DHT}'_k\\ \mathbf{DHT}'_k\end{vmatrix} \quad I_m\right), \quad k \text{ even},$$

$$\begin{vmatrix}\mathbf{rDFT}_{2n}(u)\\ \mathbf{rDHT}_{2n}(u)\end{vmatrix} \to L_m^{2n}\left(I_k \quad {}_i \begin{vmatrix}\mathbf{rDFT}_{2m}((i+u)/k)\\ \mathbf{rDHT}_{2m}((i+u)/k)\end{vmatrix}\right)\left(\begin{vmatrix}\mathbf{rDFT}_{2k}(u)\\ \mathbf{rDHT}_{2k}(u)\end{vmatrix} \quad I_m\right),$$

$$\mathbf{RDFT\text{-}3}_n \to (Q_{k/2,m}^\top \quad I_2)\,(I_k \quad {}_i\, \mathbf{rDFT}_{2m})(i+1/2)/k))\,(\mathbf{RDFT\text{-}3}_k \quad I_m), \quad k \text{ even},$$

$$\mathbf{DCT\text{-}2}_n \to P_{k/2,2m}^\top \left(\mathbf{DCT\text{-}2}_{2m} K_2^{2m} \oplus \left(I_{k/2-1} \quad N_{2m}\, \mathbf{RDFT\text{-}3}_{2m}^\top\right)\right) B_n (L_{k/2}^{n/2} \quad I_2)(I_m \quad \mathbf{RDFT}'_k) Q_{m/2,k},$$

$$\mathbf{DCT\text{-}3}_n \to \mathbf{DCT\text{-}2}_n^\top,$$

$$\mathbf{DCT\text{-}4}_n \to Q_{k/2,2m}^\top \left(I_{k/2} \quad N_{2m}\, \mathbf{RDFT\text{-}3}_{2m}^\top\right) B'_n (L_{k/2}^{n/2} \quad I_2)(I_m \quad \mathbf{RDFT\text{-}3}_k) Q_{m/2,k}.$$

$$\mathbf{DFT}_n \to (\mathbf{DFT}_k \quad \mathrm{I}_m)\, \mathrm{T}_m^n (\mathrm{I}_k \quad \mathbf{DFT}_m)\, \mathrm{L}_k^n, \quad n = km$$ — ***Cooley-Tukey FFT***

$$\mathbf{DFT}_n \to P_n (\mathbf{DFT}_k \quad \mathbf{DFT}_m) Q_n, \quad n = km, \ \gcd(k,m) = 1$$ — ***Prime-factor FFT***

$$\mathbf{DFT}_p \to R_p^T (\mathrm{I}_1 \oplus \mathbf{DFT}_{p-1}) D_p (\mathrm{I}_1 \oplus \mathbf{DFT}_{p-1}) R_p, \quad p \text{ prime}$$ — ***Rader FFT***

$$\mathbf{DCT\text{-}3}_n \to (\mathrm{I}_m \oplus \mathrm{J}_m)\, \mathrm{L}_m^n (\mathbf{DCT\text{-}3}_m(1/4) \oplus \mathbf{DCT\text{-}3}_m(3/4)) \cdot (\mathrm{F}_2 \quad \mathrm{I}_m) \begin{bmatrix} \mathrm{I}_m & 0 \oplus -\mathrm{J}_{m-1} \\ & \frac{1}{\sqrt{2}}(\mathrm{I}_1 \oplus 2\, \mathrm{I}_m) \end{bmatrix}, \quad n = 2m$$

$$\mathbf{DCT\text{-}4}_n \to S_n \mathbf{DCT\text{-}2}_n \operatorname{diag}_{0 \le k < n}(1/(2\cos((2k+1)\pi/4n)))$$

$$\mathbf{IMDCT}_{2m} \to (\mathrm{J}_m \oplus \mathrm{I}_m \oplus \mathrm{I}_m \oplus \mathrm{J}_m)\left(\left(\begin{bmatrix}1\\-1\end{bmatrix} \quad \mathrm{I}_m\right) \oplus \left(\begin{bmatrix}-1\\-1\end{bmatrix} \quad \mathrm{I}_m\right)\right) \mathrm{J}_{2m}\, \mathbf{DCT\text{-}4}_{2m}$$

$$\mathbf{WHT}_{2^k} \to \prod_{i=1}^{t} (\mathrm{I}_{2^{k_1+\cdots+k_{i-1}}} \quad \mathbf{WHT}_{2^{k_i}} \quad \mathrm{I}_{2^{k_{i+1}+\cdots+k_t}}), \quad k = k_1 + \cdots + k_t$$

$$\mathbf{DFT}_2 \to \mathrm{F}_2$$

$$\mathbf{DCT\text{-}2}_2 \to \operatorname{diag}(1, 1/\sqrt{2})\, \mathrm{F}_2$$

$$\mathbf{DCT\text{-}4}_2 \to \mathrm{J}_2\, \mathrm{R}_{13\pi/8}$$

# Complexity of the DFT

- **Measure: $L_c$, 2 ≤ c**
  - Complex adds count 1
  - Complex mult by a constant a with |a| < c counts 1
  - $L_2$ is strictest, $L_\infty$ the loosest (and most natural)
- **Upper bounds:**
  - $n = 2^k$: $L_2(DFT_n) \leq 3/2\ n \log_2(n)$ *(using Cooley-Tukey FFT)*
  - General n: $L_2(DFT_n) \leq 8\ n \log_2(n)$ *(needs Bluestein FFT)*
- **Lower bound:**
  - Theorem by Morgenstern: If $c < \infty$, then $L_c(DFT_n) \geq ½\ n \log_c(n)$
  - Implies: in the measure $L_c$, the DFT is $\Theta(n \log(n))$

# History of FFTs

- **The advent of digital signal processing is often attributed to the FFT** *(Cooley-Tukey 1965)*
- **History:**
  - Around 1805: FFT discovered by Gauss [1]
    (Fourier publishes the concept of Fourier analysis in 1807!)
  - 1965: Rediscovered by Cooley-Tukey

*[1]: Heideman, Johnson, Burrus: "Gauss and the History of the Fast Fourier Transform" Arch. Hist. Sc. 34(3) 1985*

# Carl-Friedrich Gauss


**1777 - 1855**

- **Contender for the greatest mathematician of all times**
- **Some contributions:** Modular arithmetic, least square analysis, normal distribution, fundamental theorem of algebra, Gauss elimination, Gauss quadrature, Gauss-Seidel, non-Euclidean geometry, …

# Fast Implementation (≈ FFTW 2.x)

- Choice of algorithm
- Locality optimization
- Constants
- Fast basic blocks
- Adaptivity

- Blackboard