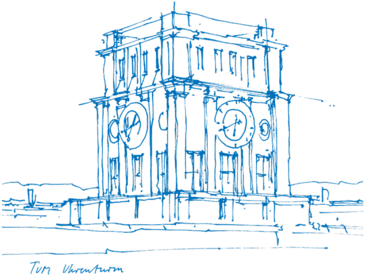
# Algorithms of Scientific Computing

## The Quarter-Wave DFT and the (Quarter-Wave) Discrete Cosine Transform (QW-DCT)

Michael Bader
Technical University of Munich

Summer 2022



# Motivation: Compression of Image Data (JPEG)

## Compression steps of the JPEG method

1. Conversion into a suitable colour model (YCbCr, e.g.), separation of brightness and colour information
2. Downsampling (in particular of the colour components)
3. **blockwise "quarter-wave discrete cosine transform"** (blocks of size $8 \times 8$)
4. Quantisation of the coefficients ($\rightarrow$ reduce information)
5. run-length encoding, Huffman/arithmetic coding (loss-free compression of the quantified coefficients)

## Our next topics therefore:

- What is a "quarter-wave" transform?
- What is a "cosine transform"?

# Revisited: Discrete Fourier Transform (DFT)

**Definition:**
For a vector of $N$ complex numbers $(f_0, \ldots, f_{N-1})^T$, the **discrete Fourier transform** is given by the vector $(F_0, \ldots, F_{N-1})^T$, where

$$F_k = \frac{1}{N} \sum_{n=0}^{N-1} f_n e^{-i2\pi nk/N}.$$

**Interpretation:**

- as trigonometric interpolation/approximation
- **as approximation of the coefficients of the Fourier series**

# Fourier Coefficients and Numerical Quadrature

For a $2\pi$-periodic function $f$, the corresponding **Fourier series** is defined as

$$f(x) \sim \sum_{k=-\infty}^{\infty} c_k e^{ikx}, \qquad \text{where} \quad c_k = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ikx}\, \mathrm{dx}$$

The $c_k$ are called (continuous) **Fourier coefficients**.

If $f$ is piecewise smooth, the Fourier series converges pointwise
(i.e. for each $x$) towards

$$\tfrac{1}{2}\left(f(x^+) + f(x^-)\right),$$

i.e. in particular towards $f(x)$, if $f$ is continuously differentiable at $x$.

# Computation of Fourier Coefficients $c_k$

Assume: $f(x)$ given by Fourier series, then $f(x) = \sum_{k=-\infty}^{\infty} c_k e^{ikx}$

Multiply by $e^{-inx}$ and integrate:

$$\int_0^{2\pi} f(x)e^{-inx}\,\mathrm{dx} = \sum_{k=-\infty}^{\infty} \int_0^{2\pi} c_k e^{ikx} e^{-inx}\,\mathrm{dx} = \sum_{k=-\infty}^{\infty} c_k \underbrace{\int_0^{2\pi} e^{i(k-n)x}\,\mathrm{dx}}_{=0,\text{ if } k \neq n}$$

$\Rightarrow$ only term for $k = n$ remains in the series, and $\int_0^{2\pi} e^{i(n-n)x}\,\mathrm{dx} = 2\pi$

The Fourier coefficients $c_k$ thus need to be

$$c_k = \frac{1}{2\pi} \int_0^{2\pi} f(x)e^{-ikx}\,\mathrm{dx}$$

# Orthogonal Functions

If we consider

- the functions $e^{ikx}$ as vectors
- and the operation $\langle f(x), g(x) \rangle := \frac{1}{2\pi} \int f(x) g(x)\, \mathrm{dx}$ as a scalar product

then the formula

$$\frac{1}{2\pi} \int_0^{2\pi} e^{ikx} e^{-inx}\, \mathrm{dx} = \frac{1}{2\pi} \int_0^{2\pi} e^{i(k-n)x}\, \mathrm{dx} = \begin{cases} 1 & \text{if } k = n \\ 0 & \text{if } k \neq n \end{cases}$$

suggests that $e^{ikx}$ and $e^{-inx}$ are **orthogonal** functions.

But we are dealing with complex numbers!

- vector scalar product is therefore $v^H w = \left(v^T\right)^* w$
- **scalar product on functions** is thus $\langle f(x), g(x) \rangle := \frac{1}{2\pi} \int \left(f(x)\right)^* g(x)\, \mathbf{dx}$
- thus: $e^{ikx}$ are orthonormal functions!

The Fourier coefficients $c_k$ are thus computed via an orthogonal projection:

$$c_k = \left\langle e^{ikx}, f(x) \right\rangle = \frac{1}{2\pi} \int_0^{2\pi} e^{-ikx} f(x)\, \mathrm{dx}$$

# Approximate Computation of $c_k$

The continuous Fourier coefficients are given as

$$c_k = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ikx} \, \mathrm{dx}$$

Steps to compute $c_k$ approximately:

- consider $c_k$ only for $\pm k = 0, \ldots, K$; then: $f(x) \approx \sum_{k=-K}^{K} c_k e^{ikx}$
- compute numerical approximation of integral $\int_0^{2\pi} f(x) e^{-ikx} \, \mathrm{dx}$

# Computation of $c_k$ via Trapezoidal Sum

**Trapezoidal sum:** for equidistant $x_n := \frac{2\pi n}{N}$:

$$\int_0^{2\pi} g(x)\,\mathrm{dx} \approx T_N\{g\} := \frac{2\pi}{N}\left(\frac{1}{2}g(x_0) + \sum_{n=1}^{N-1} g(x_n) + \frac{1}{2}g(x_N)\right)$$

Use $g(x) := f(x)e^{-ikx}$ and $f_n := f(x_n)$, then:

$$c_k \approx \frac{1}{2\pi} T_N\left\{f(x)e^{-ikx}\right\} = \frac{1}{N}\left(\frac{1}{2}f_0 e^0 + \sum_{n=1}^{N-1} f_n e^{-i2\pi nk/N} + \frac{1}{2}f_N e^{-i2\pi Nk/N}\right)$$

$$= \frac{1}{N}\left(\frac{f_0}{2} + \sum_{n=1}^{N-1} f_n e^{-i2\pi nk/N} + \frac{f_N}{2}\right)$$

# Computation of $c_k$ via Trapezoidal Sum (2)

If $f_0 = f_N$ (periodic data), we obtain

$$c_k \approx F_k = \frac{1}{N} \sum_{n=0}^{N-1} f_n e^{-i2\pi nk/N}$$

- $\Rightarrow$ $F_k$ are approximations of $c_k$
- $\Rightarrow$ approximate computation leads to solution of the interpolation problem
- $\Rightarrow$ approximation error is of order $\mathcal{O}(N^{-2})$

For $f_0 \neq f_N$, or for “discontinuities”, we get a recommendation:

**Average Values at Endpoints and Discontinuities (AVED)**

# Computation of $c_k$ via Midpoint Rule

**Midpoint rule:** evaluate $g(x)$ at midpoints $x_n$:

$$\int_0^{2\pi} g(x)\,\mathrm{dx} \approx \frac{2\pi}{N} \sum_{n=0}^{N-1} g(x_n) \qquad \text{with} \quad x_n := \frac{2\pi\left(n+\frac{1}{2}\right)}{N}\,.$$

With $g(x) := f(x)e^{-ikx}$ and $f_n := f(x_n)$, we obtain:

$$c_k \approx \widetilde{F}_k := \frac{1}{N} \sum_{n=0}^{N-1} f_n e^{-i2\pi\left(n+\frac{1}{2}\right)k/N}$$

**“Quarter-Wave Discrete Fourier Transform”**

# DFT and Symmetry

| | INPUT | | TRANSFORM |
|---|---|---|---|
| **real symmetry** | $f_n \in \mathbb{R}$ | $\rightarrow$ | Real DFT (RDFT) |
| **even symmetry** | $f_n = f_{-n}$ | $\rightarrow$ | Discrete Cosine Transform (DCT) |
| **odd symmetry** | $f_n = -f_{-n}$ | $\rightarrow$ | Discrete Sine Transform (DST) |

| "QUARTER-WAVE" | INPUT | | TRANSFORM |
|---|---|---|---|
| **even symmetry** | $f_n = f_{-n-1}$ | $\rightarrow$ | QW-DCT |
| **odd symmetry** | $f_n = -f_{-n-1}$ | $\rightarrow$ | QW-DST |

# Quarter-Wave Discrete Fourier Transform

- new variant of DFT:

$$\widetilde{F}_k := \frac{1}{N} \sum_{n=0}^{N-1} f_n e^{-i2\pi\left(n+\frac{1}{2}\right)k/N} \qquad f_n := \sum_{k=0}^{N-1} \widetilde{F}_k e^{i2\pi\left(n+\frac{1}{2}\right)k/N}$$

- Comparison with coefficients $F_k$ of the “usual” DFT:

$$F_k = \widetilde{F}_k e^{i\pi k/N} = \widetilde{F}_k \omega_N^{k/2}$$

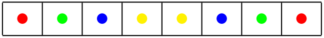
- Supporting points compared to “usual” DFT shifted by a “quarter wave length” (midpoints of intervals).
- Derivation via midpoint rule motivates usage for piecewise constant data

$\Rightarrow$ **Transformation of image data**

# Quarter-Wave DFT on Symmetric Data

Given $2N$ real-valued input data $f_0, \ldots, f_{2N-1}$ with symmetry

$$f_{2N-n-1} = f_n \qquad \text{e.g., for } N = 4:$$




Inserting the symmetric data in Quarter-Wave DFT results in

$$
\begin{aligned}
\widetilde{F}_k &= \frac{1}{2N} \sum_{n=0}^{2N-1} f_n \, \omega_{2N}^{-k\left(n+\frac{1}{2}\right)} = \sum_{n=0}^{N-1} f_n \, \omega_{2N}^{-k\left(n+\frac{1}{2}\right)} + \sum_{n=N}^{2N-1} f_n \, \omega_{2N}^{-k\left(n+\frac{1}{2}\right)} \\
&= \frac{1}{2N} \sum_{n=0}^{N-1} f_n \, \omega_{2N}^{-k\left(n+\frac{1}{2}\right)} + \frac{1}{2N} \sum_{n=0}^{N-1} f_{2N-n-1} \, \omega_{2N}^{-k\left(2N-n-1+\frac{1}{2}\right)} \\
&= \frac{1}{2N} \sum_{n=0}^{N-1} f_n \, \omega_{2N}^{-k\left(n+\frac{1}{2}\right)} + \frac{1}{2N} \sum_{n=0}^{N-1} f_n \, \omega_{2N}^{-k\left(-n-\frac{1}{2}\right)} \omega_{2N}^{-k\cdot 2N} \\
&= \frac{1}{2N} \sum_{n=0}^{N-1} f_n \left( \omega_{2N}^{-k\left(n+\frac{1}{2}\right)} + \omega_{2N}^{k\left(n+\frac{1}{2}\right)} \right) = \frac{1}{N} \sum_{n=0}^{N-1} f_n \cos\left( \frac{\pi k \left(n + \frac{1}{2}\right)}{N} \right) .
\end{aligned}
$$

# Quarter-Wave DFT on Symmetric Data (2)

Quarter-Wave DFT of symmetric data results in **real-valued** coefficients:

$$\widetilde{F}_k = \frac{1}{N} \sum_{n=0}^{N-1} f_n \cos\left( \frac{\pi k \left(n + \frac{1}{2}\right)}{N} \right) \qquad \text{for } k = 0, \ldots, 2N-1$$

Additional symmetry:

$$\begin{aligned} \widetilde{F}_{2N-k} &= \frac{1}{N} \sum_{n=0}^{N-1} f_n \cos\left( \frac{\pi(2N-k)\left(n + \frac{1}{2}\right)}{N} \right) \\ &= \frac{1}{N} \sum_{n=0}^{N-1} f_n \cos\left( 2\pi n + \pi - \frac{\pi k \left(n + \frac{1}{2}\right)}{N} \right) = -\widetilde{F}_k \end{aligned}$$

$\Rightarrow$ again: only $N$ independent coefficients

**Exercise:** verify that $\widetilde{F}_{-k} = \widetilde{F}_k$, but $\widetilde{F}_{k+2N} = -\widetilde{F}_k$

# Quarter-Wave Even Discrete Cosine Transform

**Backward transform:**

$$f_n := \sum_{k=0}^{2N-1} \widetilde{F}_k e^{i2\pi\left(n+\frac{1}{2}\right)k/2N} \quad \xrightarrow{\widetilde{F}_{2N-k}=-\widetilde{F}_k} \quad f_n = \widetilde{F}_0 + 2\sum_{k=1}^{N-1} \widetilde{F}_k \cos\left(\frac{\pi k\left(n+\frac{1}{2}\right)}{N}\right)$$

**Exercise:** insert $\widetilde{F}_{2N-k} = -\widetilde{F}_k$ into $f_n := \sum \widetilde{F}_k e^{i2\pi\left(n+\frac{1}{2}\right)k/2N}$ and verify!

$\rightsquigarrow$ we obtain an inverse quarter-wave even discrete cosine transform

**We define a pair of transforms – (inverse) quarter-wave even DCT:**

$$\widetilde{F}_k = \frac{1}{N}\sum_{n=0}^{N-1} f_n \cos\left(\frac{\pi k\left(n+\frac{1}{2}\right)}{N}\right) \qquad f_n = \widetilde{F}_0 + 2\sum_{k=1}^{N-1} \widetilde{F}_k \cos\left(\frac{\pi k\left(n+\frac{1}{2}\right)}{N}\right)$$

***N* real values** $\longleftrightarrow$ ***N* real-valued coefficients**

(no symmetry any more in data/coefficients!)

# Summary: QW-DCT and inverse QW-DCT

We obtain a new pair of transforms:

$$\widetilde{F}_k = \frac{1}{N} \sum_{n=0}^{N-1} f_n \cos\left( \frac{\pi k \left(n + \frac{1}{2}\right)}{N} \right) \qquad f_n = \widetilde{F}_0 + 2 \sum_{k=1}^{N-1} \widetilde{F}_k \cos\left( \frac{\pi k \left(n + \frac{1}{2}\right)}{N} \right)$$

- both transforms work on **data sets** that **are neither symmetric nor periodic**
- however, if we extend the data sets according to the symmetry rules, then the reflected (and thus symmetric) sets become periodic as well

The two transforms are connected to the QW-DFT and QW-iDFT via a 3-step procedure:

1. extend/duplicate the data set in a symmetric way
2. apply the QW-DFT/QW-iDFT
3. extract the symmetric half of the transformed data set

This equivalence has two important consequences:

1. we may compute the cosine transforms ($N$ numbers that require sums over $N$ terms $\Rightarrow \mathcal{O}(N^2)$ operations) by using an FFT in step 2 $\Rightarrow$ reduces work to $\mathcal{O}(N \log N)$
2. we prove that QW-DCT and QW-iDCT are inverse operations to each other (because the QW-DFT and QW-iDFT are inverse to each other)

# 2D Cosine Transform

**Definition of the 2D-DCT:**

$$\widetilde{F}_{kl} = \frac{1}{N \cdot M} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f_{nm} \cos\left(\frac{\pi k \left(n + \frac{1}{2}\right)}{N}\right) \cos\left(\frac{\pi l \left(m + \frac{1}{2}\right)}{M}\right)$$

$$f_{nm} = 4 \sum_{k=0}^{N-1}{}' \sum_{l=0}^{M-1}{}' \widetilde{F}_{kl} \cos\left(\frac{\pi k \left(n + \frac{1}{2}\right)}{N}\right) \cos\left(\frac{\pi l \left(m + \frac{1}{2}\right)}{M}\right)$$

shortened notation: $\sum_{k=0}^{N-1}{}' x_k := \frac{x_0}{2} + \sum_{k=1}^{N-1} x_k$

**Application:** blockwise 2D-DCT in JPEG/MPEG compression

# Reduction of the 2D-FCT to 1D-FCTs

In the 2D cosine transform, we can rearrange:

$$
\begin{aligned}
\widetilde{F}_{kl} &= \frac{1}{N^2} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} f_{nm} \cos\left(\frac{\pi k \left(n+\frac{1}{2}\right)}{N}\right) \cos\left(\frac{\pi l \left(m+\frac{1}{2}\right)}{N}\right) \\
&= \frac{1}{N} \sum_{n=0}^{N-1} \underbrace{\left(\frac{1}{N} \sum_{m=0}^{N-1} f_{nm} \cos\left(\frac{\pi l \left(m+\frac{1}{2}\right)}{N}\right)\right)}_{:= \widehat{F}_{nl}} \cos\left(\frac{\pi k \left(n+\frac{1}{2}\right)}{N}\right) .
\end{aligned}
$$

- For each $n$, $\widehat{F}_{nl}$ are computed via $N$ 1D transforms
- we may first 1D-transform all rows and then all columns to get the 2D-transform

$\rightarrow$ **see tutorials for more details!**

# Application Example: Compression of Image Data (JPEG)

**Compression steps of the JPEG method**

1. Conversion into a suitable colour model (YCbCr, e.g.), separation of brightness and colour information
2. Downsampling (in particular of the colour components)
3. **blockwise "quarter-wave discrete cosine transform"** (blocks of size $8 \times 8$)
4. Quantisation of the coefficients ($\rightarrow$ reduce information)
5. run-length encoding, Huffman/arithmetic coding (loss-free compression of the quantified coefficients)

Example: jpeg on matlab central (see link on webpage)

# QW-DCT – Algorithm

**Reduce to Real FFT:**

**(1)** for $n = 0, \ldots, N-1$:

$$g_n = f_n \qquad g_{2N-n-1} = f_n$$

**(2)** $2N$-Real-FFT: compute $G_k$ from $g_n$ (for $k = 0, \ldots, N$)

**(3)** for $k = 0, \ldots, N-1$:

$$\widetilde{F}_k = G_k e^{-i\pi k/2N}$$

**Important Note:**

- this results in an algorithm that requires $\mathcal{O}(N \log N)$ operations (FFT!)
- whereas computing all $\widetilde{F}_k = \frac{1}{N} \sum f_n \cos\left(\pi k(n+\frac{1}{2})/N\right)$ would require $\mathcal{O}(N^2)$ operations

**Possible Further Optimisations:**

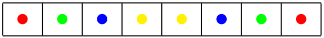
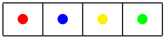
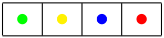
- substitute real $2N$-FFT by complex $N$-FFT
- compact (divide-and-conquer) real FFT
- is there a compact/fast (QW-)DCT? $\longrightarrow$ see paper by *Swarztrauber*

# Compact Fast DCT ($\rightarrow$ Swarztrauber, 1986)

**Consider QW-DCT:** with symmetry $f_{2N-n-1} = f_n$

$$\widetilde{F}_k = \frac{1}{2N} \sum_{n=0}^{2N-1} f_n \omega_{2N}^{-k\left(n+\frac{1}{2}\right)} \quad \longrightarrow \quad \widetilde{F}_k = \frac{1}{N} \sum_{n=0}^{N-1} f_n \cos\left(\frac{\pi k \left(n+\frac{1}{2}\right)}{N}\right) .$$

**Split into even and odd indices:** $g_n := f_{2n}$ and $h_n := f_{2n+1}$ (as in FFT)

 $\rightsquigarrow$  and 

- $g_n := f_{2n}$:

$$g_n = f_{2n} = f_{2N-2n-1} = f_{2(N-n)-1} = f_{2(N-n-1)+1} = h_{N-n-1}$$

- $h_n := f_{2n+1}$:

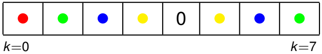
$$h_n = f_{2n+1} = f_{2N-(2n+1)-1} = f_{2(N-n-1)} = g_{N-n-1}$$

- thus: **two real DFTs** with symmetric data sets
  see exercises: reversed-data DFT easily obtained from DFT

# Compact Fast Inverse QW-DCT

**Consider backward transform:** with symmetry $\widetilde{F}_{2N-k} = -\widetilde{F}_k$

e.g., for $N = 4$:




$$f_n := \sum_{k=0}^{2N-1} \widetilde{F}_k e^{i2\pi\left(n+\frac{1}{2}\right)k/2N} \quad \longrightarrow \quad f_n = \widetilde{F}_0 + 2\sum_{k=1}^{N-1} \widetilde{F}_k \cos\left(\frac{\pi k \left(n+\frac{1}{2}\right)}{N}\right)$$

**Split into even and odd indices:** (as in FFT)

- $G_k := \widetilde{F}_{2k}$: **again leads to Inverse QW-DCT**

$$-G_k = -\widetilde{F}_{2k} = \widetilde{F}_{2N-2k} = \widetilde{F}_{2(N-k)} = G_{N-k}$$

- $H_k := \widetilde{F}_{2k+1}$: **leads to new kind of inverse DCT**

$$-H_k = -\widetilde{F}_{2k+1} = \widetilde{F}_{2N-(2k+1)} = \widetilde{F}_{2(N-k)-1} = \widetilde{F}_{2(N-k-1)+1} = H_{N-k-1}$$

(next even/odd split leads to two real DFTs with symm. data sets)