Multimedia

§6 De-correlation

Prof. Dr. Georg Umlauf

Hochschule Konstanz
University of Applied Sciences

Content

Content

Why transformations?

Transformations are used to convert data such that

1. processing of the data becomes less complex/costly or even possible in the first place

   and

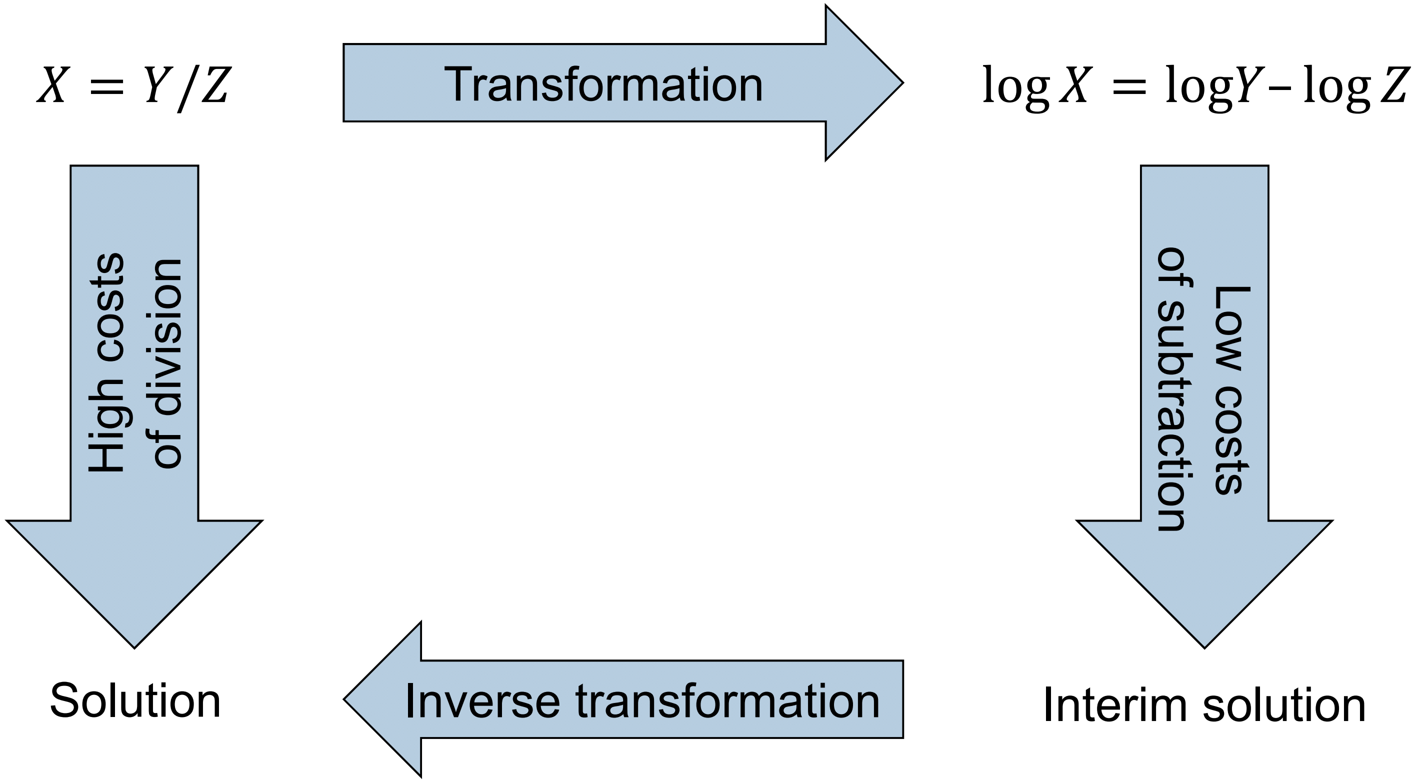
2. a unique reconstruction is possible via a suitable inverse transformation.

Remarks:

- Transformation and inverse transformation are costly.
- Computations in the transformed space are often significantly simpler.

Why transformations?

Example: Compute $X = Y/Z$.



Why Fourier transformations?

- Periodic functions can be represented as sums of sine- and cosine-functions.
- Non-periodic functions can be represented as integrals of sine- and cosine-functions.

Source: wikipedia

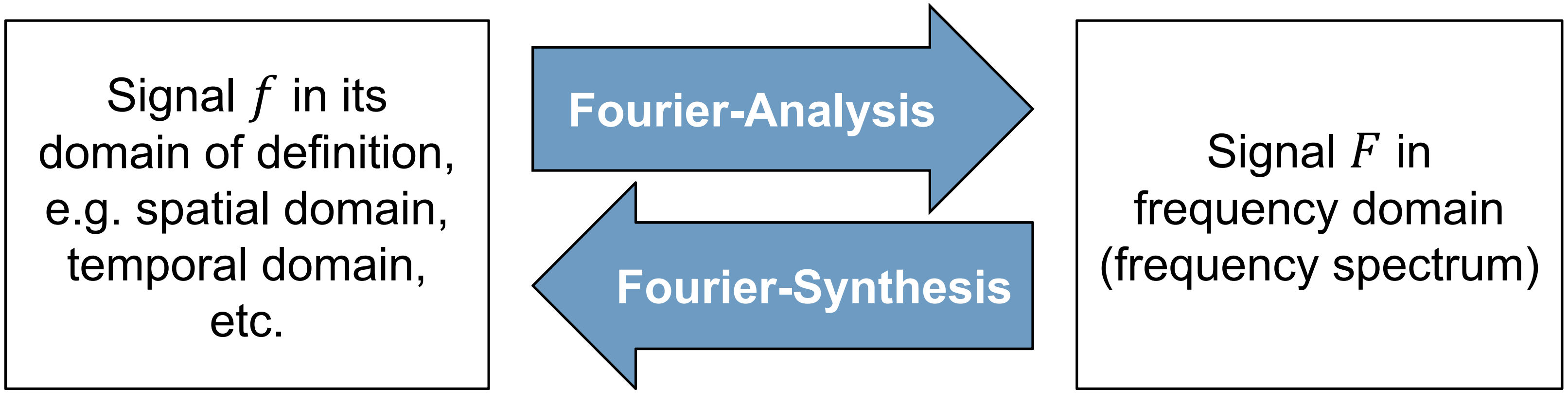
Jean-Babtiste-Joseph Fourier

- These representations allow to directly read off certain properties of the functions:
  - occurring frequencies,
  - energy of individual frequency components,
  - symmetry of function.
- Temporal information gets lost, but can partially be recovered by evaluation in separated time slots (spectrogram).

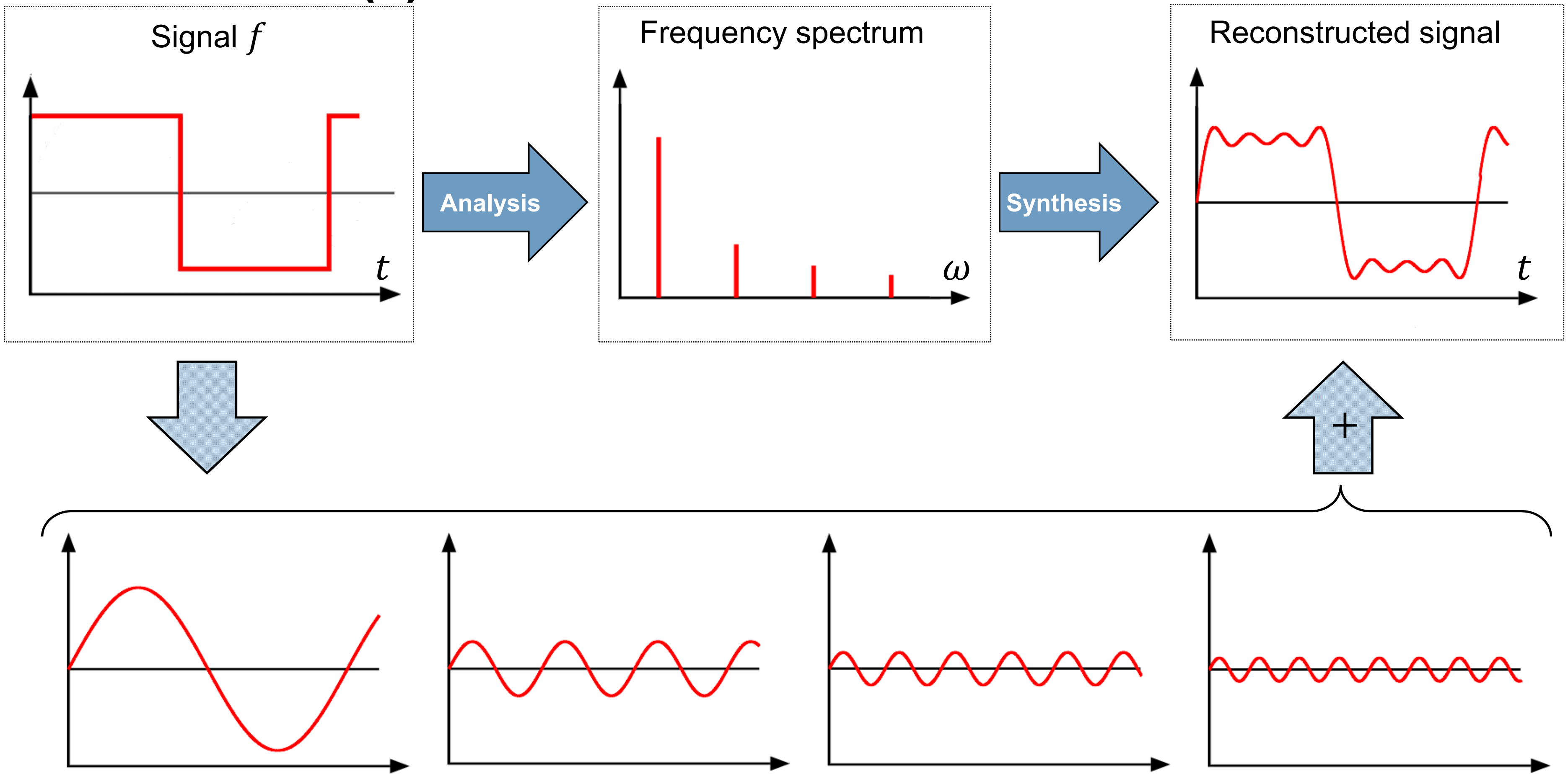
Fourier idea (1)

- The Fourier-Analysis decomposes signals into sums of simple sine-waves.

➡ Mapping of a spatial/temporal signal to its components in the frequency domain.



Fourier idea (2)



Fourier-Analysis

- **Fourier series:** Periodic functions can be represented as (in-)finite sums of sine-/cosine-functions.
  - ➡ Domain of definition: Signal is continuous over interval, periodic.
  - ➡ Frequency spectrum: Discrete.
- **Fourier transformation:** Non-periodic functions can be represented as integrals of sine-/cosine-functions.
  - ➡ Domain of definition: Signal is continuous, aperiodic.
  - ➡ Frequency spectrum: Continuous.
- **Discrete Fourier transformation (DFT, FFT):** Finite sequences can be represented as finite sequences of sine-/cosine-functions.
  - ➡ Domain of definition: Signal is discrete, finite, periodically continued.
  - ➡ Frequency spectrum: Discrete, finite.

Fourier series

- A periodic signal f with period $T > 0$ can be decomposed into simple sine- and cosine-signals:

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_k \cos(k\omega t) + b_k \sin(k\omega t)), \omega = \frac{2\pi}{T}$$

with

$a_k = \frac{2}{T} \int_0^T f(t) \cdot \cos(k\omega t)\, dt$ for $k = 0,1,2,3, \dots$ and

$b_k = \frac{2}{T} \int_0^T f(t) \cdot \sin(k\omega t)\, dt$ for $k = 1,2,3, \dots$

- The coefficients a_k, b_k are called Fourier-coefficients of f.

Properties of Fourier series

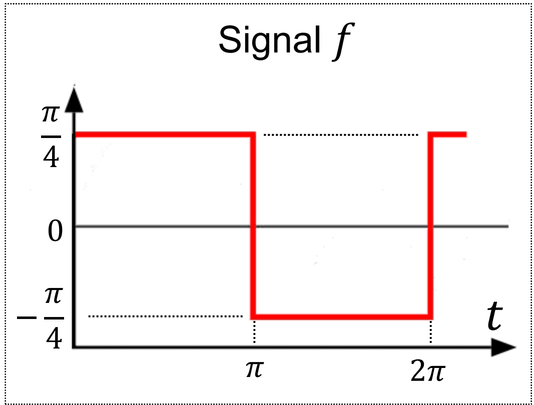
- A function f is called **even**, if $f(t) = f(-t)$.
  - ➡ For an even function f we get $b_k = 0, k \in \mathbb{N}$.
- A function f is called **odd**, if $f(t) = -f(-t)$.
  - ➡ For an odd function f we get $a_k = 0, k \in \mathbb{N}$.

➡ Every signal can be decomposed into its even and odd components.

➡ **Synthesis** (reconstruction): Assemble the **trigonometric polynomials**

$$f_n(t) = \frac{a_0}{2} + \sum_{k=1}^{n} \left(a_k \cos(k\omega t) + b_k \sin(k\omega t)\right),$$

which converge for $n \to \infty$ to f, if $f \in C^1$.



Example Fourier series

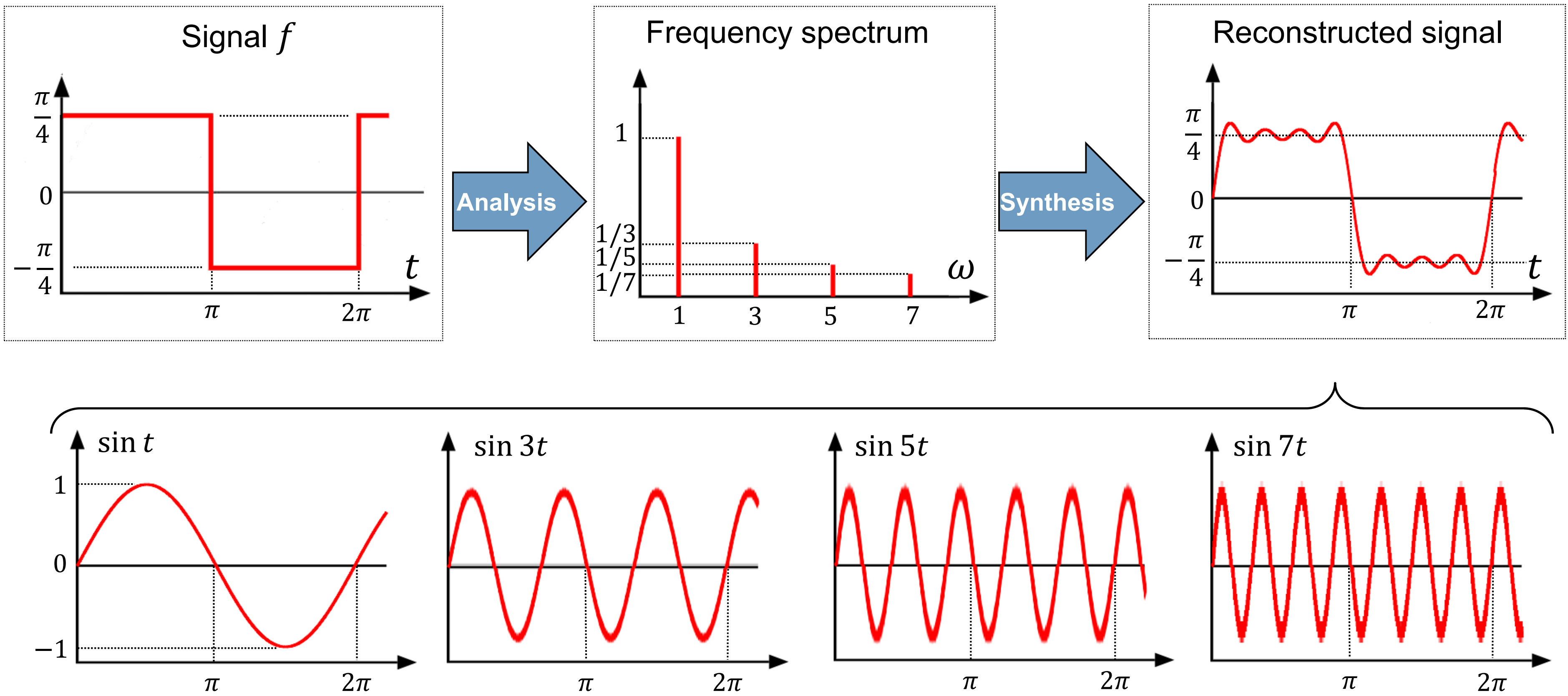
- Rectangle wave with amplitude $h = \pi/4$ and period $T = 2\pi$.
  - ➡ $\omega = 1.$

➡ $a_k = 0$, because the rectangle wave is an odd function.

- $a_0 = \frac{1}{\pi}\left(\int_0^{\pi} h\,dt - \int_{\pi}^{2\pi} h\,dt\right) = 0.$
- $b_1 = \frac{1}{\pi}\left(\int_0^{\pi} h \sin t\,dt - \int_{\pi}^{2\pi} h \sin t\,dt\right) = \frac{1}{\pi} h \cdot (2 - (-2)) = 1.$
- $b_2 = \frac{1}{4}\left(\int_0^{\pi} \sin 2t\,dt - \int_{\pi}^{2\pi} \sin 2t\,dt\right) = 0.$
- $b_3 = \frac{1}{4}\left(\int_0^{\pi} \sin 3t\,dt - \int_{\pi}^{2\pi} \sin 3t\,dt\right) = \frac{1}{4}\left(\frac{2}{3} - \left(-\frac{2}{3}\right)\right) = \frac{1}{3}\ ....$

Example Fourier series

- Rectangle wave: $f(t) = \sum \frac{1}{2k-1} \sin(2k-1)t$



Amplitude- and phase-spectrum

- The discrete Fourier transformation decomposes a signal into its sine- and a cosine-spectrum.
  - However, the sine-function is only a phase-shifted cosine-function, i.e. $\cos(x - \pi/2) = \sin(x)$.

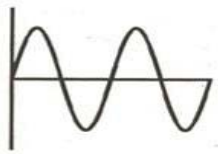
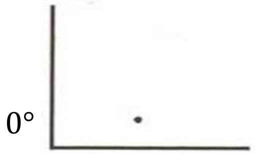
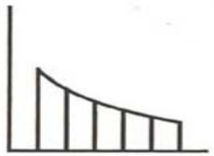
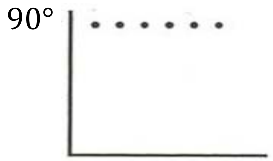
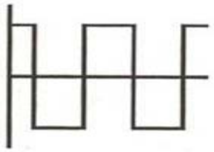
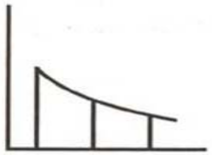
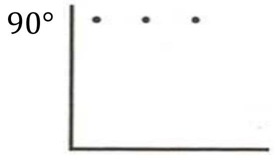
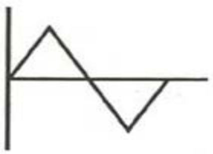
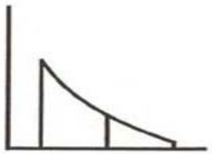
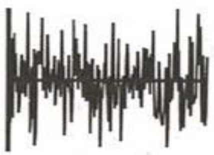
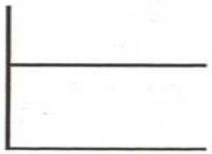
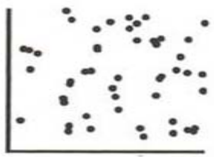
➡ The sine-spectrum can be neglected, if also the phase-shift if considered

$$f(t) = \frac{a_0}{2} + \sum A_k \cos(k\omega t - \varphi_k).$$

➡ **Amplitude-spectrum:** $A_k = \sqrt{a_k^2 + b_k^2}$

➡ **Phase-spectrum:** $\varphi_k = \operatorname{atan2}(b_k, a_k) \in [-\pi, \pi).$

Examples

Signal	Wave form	Amplitude-spectrum	Phase-spectrum
Sine wave			
Saw-tooth wave			
Rectangle wave			
Triangle wave			
White noise			

Complex Fourier series (1)

- The Euler identity $e^{ix} = \cos x + i \sin x,\ i = \sqrt{-1}$, yields a more compact form of the Fourier series

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{ik\omega t}$$

with $c_k = \frac{1}{T}\int_0^T f(t)e^{-ik\omega t}\,dt \in \mathbb{C}$, i.e.

$$c_k = \frac{a_{-k} + i\, b_{-k}}{2}, k < 0,$$

$$c_0 = \frac{a_0}{2},$$

$$c_k = \frac{a_k - i\, b_k}{2}, k > 0.$$

Complex Fourier series (2)

- The complex Fourier series corresponds to the polar coordinate representation of the real Fourier series.

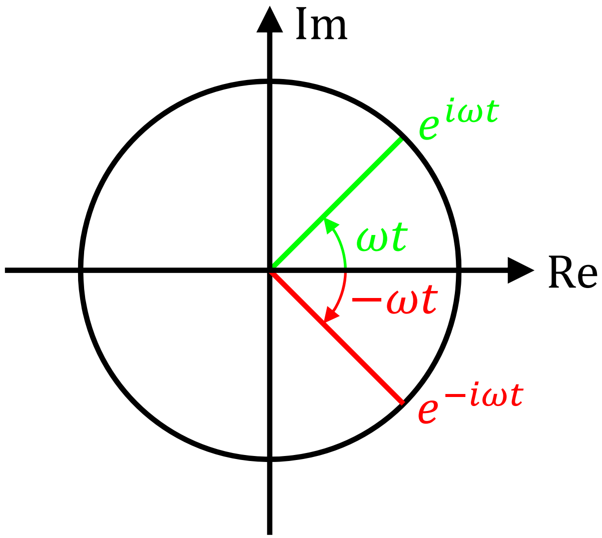
➡ Amplitude spectrum: $A_k = 2|c_k|$.

➡ Phase spectrum: $\varphi_k = \arg(c_k)$.

➡ Positive and negative frequencies occur:

$$e^{i\omega t} = \cos(\omega t) + i\sin(\omega t), \omega > 0,$$

$$e^{-i\omega t} = \cos(\omega t) - i\sin(\omega t), \omega > 0.$$



Fourier transformation (also: Fourier integral)

- To represent non-periodic signals define the amplitude density spectrum

$$F_T(\omega) = \int_{-T/2}^{T/2} f(t) e^{-i\omega t} dt$$

with $c_k = F_T\left(\frac{2k\pi}{T}\right)$.

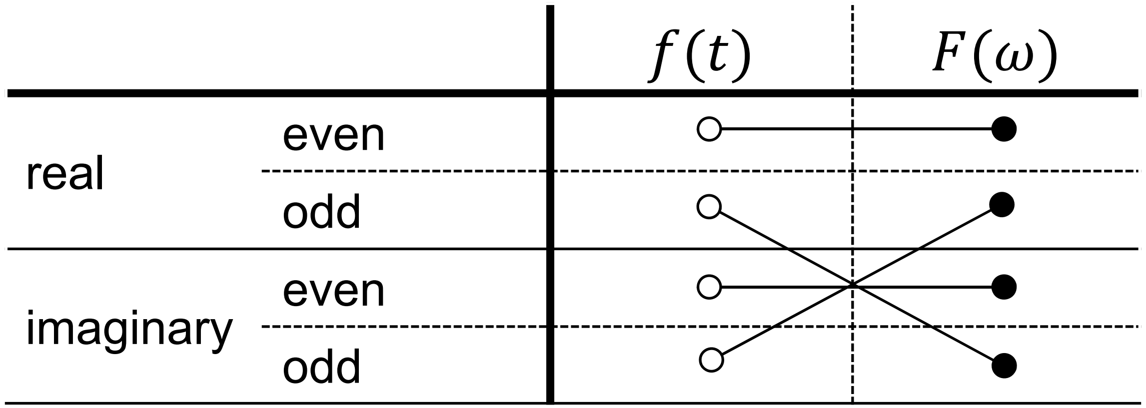
- Limit for $T \to \infty$ (and $k \to \infty$) yields the **Fourier transform** $F_T(\omega)$ of a non-periodic signals f

$$F(\omega) = \lim_{T\to\infty} F_T(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt\,.$$

- **Notation:** $f \circ\!\!-\!\!\bullet F$.

Properties of the Fourier transformation (1)

- Many properties of Fourier series carry over to the Fourier transformation:
  - The Fourier transform of an even function is even.
  - The Fourier transform of an odd function is odd.



Properties of the Fourier transformation (2)

- **Synthesis:** Inverse Fourier transformation

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega\,.$$

- **Translation** in the time domain changes the phase spectrum but not the amplitude spectrum

$$f(t-b) \circ\!\!-\!\!\bullet\; F(\omega)e^{-i\omega b}.$$

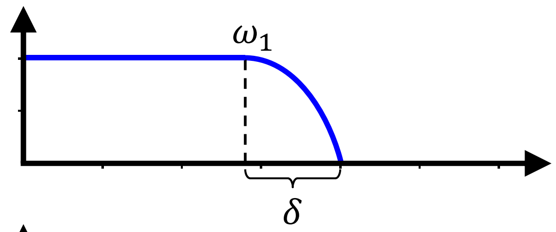
- **Compression** in the time domain corresponds to a dilation in the frequency domain and vice versa

$$f(at) \circ\!\!-\!\!\bullet\; \frac{1}{|a|}F\left(\frac{\omega}{a}\right).$$

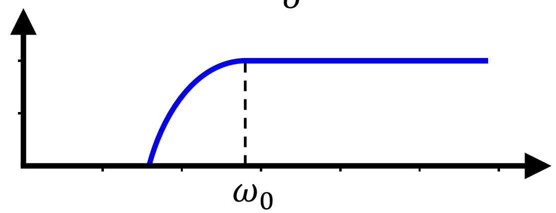
- **Remark:** Analog properties hold also for F series and DFT.

Low-, high- and band-pass filters

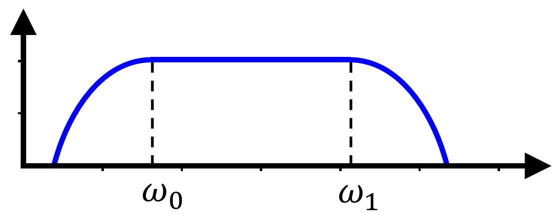
- A high-/low-pass filter corresponds to a multiplication of the Fourier-transform with the transfer function of the filter.
  - Remark: This corresponds in the spatial domain to a **convolution** of the signal with the **impulse response** (IFT of the transfer function) of the filter.
- Low-pass filter: $H(\omega) \neq 0$ for $|\omega| < \omega_1$ and
  $H(\omega) = 0$ for $|\omega| > \omega_1 + \delta$.



- High-pass filter: $H(\omega) = 0$ for $|\omega| < \omega_0 + \delta$ and
  $H(\omega) \neq 0$ for $|\omega| > \omega_0$.



- Band-pass filter: $H(\omega) = 0$ for $|\omega| < \omega_0 + \delta$ and
  $H(\omega) = 0$ for $|\omega| > \omega_1 + \delta$ and
  $H(\omega) \neq 0$ for $\omega_0 < |\omega| < \omega_1$.



Sampling the Fourier series (1)

- What is the Fourier series of a time-discrete, periodic function?

➡ Ideal sampling of n values of a continuous, periodic function f within the period T, i.e.

$$f_s(t) = f(t) \cdot \sum_{j=0}^{n-1} \delta\left(t - \frac{j}{n}T\right) \text{ with } \delta(t) = \begin{cases} 1, \text{for } t = 0 \\ 0, \text{otherwise} \end{cases}.$$

➡ Fourier series of f_s with $\omega = \frac{2\pi}{T}$ and coefficients

$$F_k = \frac{1}{T}\int_0^T f(t) \sum_{j=0}^{n-1} \delta\left(t - \frac{j}{n}T\right) e^{-ik\omega t}\, dt = \frac{1}{T}\sum_{j=0}^{n-1} \underbrace{f\left(\frac{j}{n}T\right)}_{=f_j} \cdot e^{-i\frac{2\pi jk}{n}}.$$

➡ The F_k are the Fourier coefficients of the DFT of the discrete, periodic sequence $f_j = f\left(\frac{j}{n}T\right)$.

Sampling the Fourier series (2)

- **Interpretation:** A finite sequence of values is interpreted as samples of a periodic signal within one period.

- **Synthesis:** Inverse DFT

$$f_j = \frac{1}{n} \sum_{k=0}^{n-1} F_k \cdot e^{i\frac{2\pi jk}{n}}.$$

Discrete Fourier transformation (DFT)

- A finite sequence of values $x_k \in \mathbb{R}$ is mapped by the **discrete Fourier transformation (DFT)** to the finite sequence X_k

$$X_k = \sum_{j=0}^{n-1} x_j \cdot \omega_n^{-jk} \in \mathbb{C}$$

with $\omega_n^{jk} = e^{i\frac{2\pi jk}{n}}$.

- **Synthesis:** Inverse DFT

$$x_k = \frac{1}{n} \sum_{j=0}^{n-1} X_j \cdot \omega_n^{jk}.$$

- **Properties:**
  - The discrete Fourier transform is ***n*-periodic**, i.e. for $l \geq 1$

$$X_k = X_{ln+k}.$$

  - The discrete Fourier transform is **symmetric**, i.e. for $l \geq 1$

$$X_k = X^*_{ln-k} \text{ and } |X_k| = |X_{ln-k}|.$$

➡ **Amplitude spectrum:** $A_k = |X_k|$.

➡ **Phase spectrum:** $\varphi_k = \mathrm{atan2}(\mathrm{Im}(X_k), \mathrm{Re}(X_k)) \in [-\pi, \pi)$.

➡ The signal is the sum of cosine functions with amplitude A_k and phase φ_k, analog to Fourier series.

- Matrix notation for the DFT

$$\begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_{n-1} \end{bmatrix} = \begin{bmatrix} \omega_n^0 & \omega_n^0 & \cdots & \omega_n^0 \\ \omega_n^0 & \omega_n^{-1} & \cdots & \omega_n^{-(n-1)} \\ \omega_n^0 & \omega_n^{-2} & \cdots & \omega_n^{-2(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_n^0 & \omega_n^{-(n-1)} & \cdots & \omega_n^{-(n-1)^2} \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix}$$

Shorthand: $X = F_n \cdot x$.

- ➡ **Synthesis:** inverse DFT $x = \frac{1}{n} F_n^{-1} \cdot X$ with $F_n^{-1} = F_n^*$.
- ➡ Computation via matrix-vector-multiplications takes $O(n^2)$ operations.
- ➡ How can we improve efficiency?

n-th roots of unity

- In $\mathbb{C}$ the equation $x^n - 1 = 0$ has n different solutions

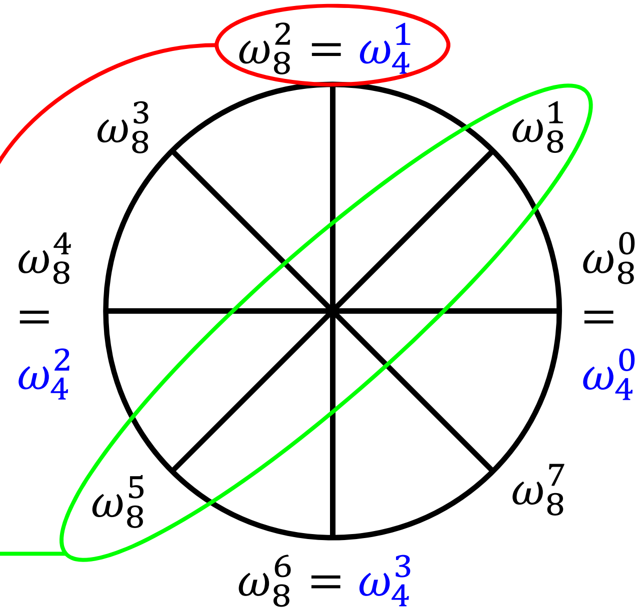
$$\omega_n^k = e^{i\frac{2\pi k}{n}} = \cos\frac{2\pi k}{n} + i\sin\frac{2\pi k}{n},\ k = 0, \dots, n-1.$$

- **Example:** $x^4 - 1 = 0$ has four solutions
  $\omega_4^0 = 1,\ \omega_4^1 = i,\ \omega_4^2 = -1,\ \omega_4^3 = -i.$
- **Example:** $x^8 - 1 = 0$ has eight solutions
  $\omega_8^0, \omega_8^1, \omega_8^2, \omega_8^3, \omega_8^4, \omega_8^5, \omega_8^6, \omega_8^7.$

➡ Two properties for $n, m \in \mathbb{N}, j \in \mathbb{Z}$.

(1) $\omega_{nm}^{jm} = \omega_n^j.$

(2) $\omega_{2n}^{n+j} = -\omega_{2n}^j.$



Example DFT

- The Fourier matrix for $n = 4$:

Rotate by multiples of $\frac{n}{2} = 2$ using (2)

Rotate by multiples of $\frac{n}{2} = 2$ using (2)

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} \omega_4^0 & \omega_4^0 & \omega_4^0 & \omega_4^0 \\ \omega_4^0 & \omega_4^{-1} & \omega_4^{-2} & \omega_4^{-3} \\ \omega_4^0 & \omega_4^{-2} & \omega_4^{-4} & \omega_4^{-6} \\ \omega_4^0 & \omega_4^{-3} & \omega_4^{-6} & \omega_4^{-9} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

➡ Permuting the left hand side...

$$\begin{bmatrix} X_0 \\ X_2 \\ X_1 \\ X_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4^{-2} & \omega_4^{-4} & \omega_4^{-6} \\ 1 & \omega_4^{-1} & \omega_4^{-2} & \omega_4^{-3} \\ 1 & \omega_4^{-3} & \omega_4^{-6} & \omega_4^{-9} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & \omega_4^{-2} & 1 & \omega_4^{-2} \\ \hline 1 & \omega_4^{-1} & \omega_4^{-2} & \omega_4^{-3} \\ 1 & \omega_4^{-3} & \omega_4^{-2} & \omega_4^{-1} \end{array}\right] \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Example FFT

$$\Rightarrow \begin{bmatrix} X_0 \\ X_2 \\ X_1 \\ X_3 \end{bmatrix} = \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & \omega_4^{-2} & 1 & \omega_4^{-2} \\ \hline 1 & \omega_4^{-1} & \omega_4^{-2} & \omega_4^{-3} \\ 1 & \omega_4^{-3} & \omega_4^{-2} & \omega_4^{-1} \end{array}\right] \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$\overset{(1)}{=} \left[\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 1 & \omega_2^{-1} & 0 & 0 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & \omega_2^{-1} \end{array}\right] \left[\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & \omega_4^{-2} & 0 \\ 0 & \omega_4^{-1} & 0 & \omega_4^{-3} \end{array}\right] \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$= \begin{bmatrix} F_2 & 0 \\ 0 & F_2 \end{bmatrix} \cdot \begin{bmatrix} I_2 & I_2 \\ D_2 & -D_2 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \text{ with } D_n = \operatorname{diag}(1, \omega_{2n}^{-1}, \dots, \omega_{2n}^{-(n-1)}).$$

Fast Fourier transformation (FFT)

- Idea:
  a) Compute the steps of the matrix-vector-multiplication in a certain order and
  b) reuse already computed intermediate values, i.e.

$$P_{2n}\begin{bmatrix} X_0 \\ \vdots \\ X_{2n-1} \end{bmatrix} = \begin{bmatrix} F_n & 0 \\ 0 & F_n \end{bmatrix} \cdot \begin{bmatrix} I_n & I_n \\ D_n & -D_n \end{bmatrix} \begin{bmatrix} x_0 \\ \vdots \\ x_{2n-1} \end{bmatrix},$$

  where the permutation matrix P_{2n} yields the bit-reverse representation.
  - n has to be a power of two.
  - ➡ Add appropriate values up to the next power of two, e.g. zeros.

➡ Runtime: $O(n \log n)$.

MATLAB-commands for the FFT

- `fft` discrete Fourier transformation
- `ifft` inverse discrete Fourier transformation
- `fft2` discrete 2d Fourier transformation
- `ifft2` inverse discrete 2d Fourier transformation
- `fftshift` „fold over“ negative frequencies
- `real` real part of a complex number
- `imag` imaginary part of a complex number
- `abs` absolute value of a complex number
- `angle` argument/angle of a complex number

§6.1 Discrete Fourier transformation
§6.1.7 Two-dimensional DFT

- A $m \times n$ matrix of values x_{jk} is mapped by the **two-dimensional Fourier transformation** to a $m \times n$ matrix X_{JK}

$$X_{JK} = \sum_{j=0}^{m-1} \sum_{k=0}^{n-1} x_{jk} \cdot \omega_m^{-jJ} \cdot \omega_n^{-kK} \in \mathbb{C}.$$

➡ $|X_{00}|$ is the average grey value of an image.

- **Synthesis:** Inverse DFT

$$x_{jk} = \frac{1}{mn} \sum_{J=0}^{m-1} \sum_{K=0}^{n-1} X_{jk} \cdot \omega_m^{jJ} \cdot \omega_n^{kK} .$$

Representation

- **Amplitude spectrum:** Represent an image as linear mapping

  $$G: [\min|X_{JK}|, \max|X_{JK}|] \to [0,1]$$

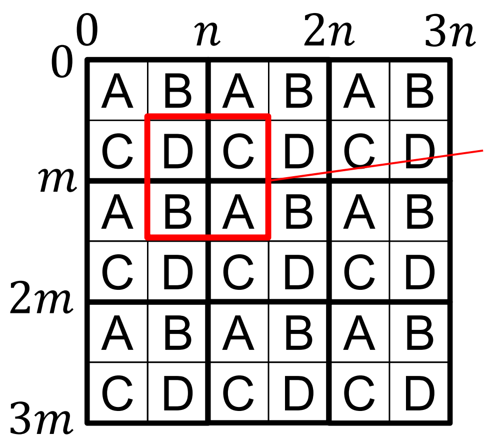
  with $G_{JK} = G(|X_{JK}|)$.
  - Where necessary logarithmize.

G_{00}	G_{01}	$\cdots$	$G_{0,n-1}$
G_{10}	G_{11}		$G_{1,n-1}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$G_{m-1,0}$	$G_{m-1,1}$	$\cdots$	$G_{m-1,n-1}$

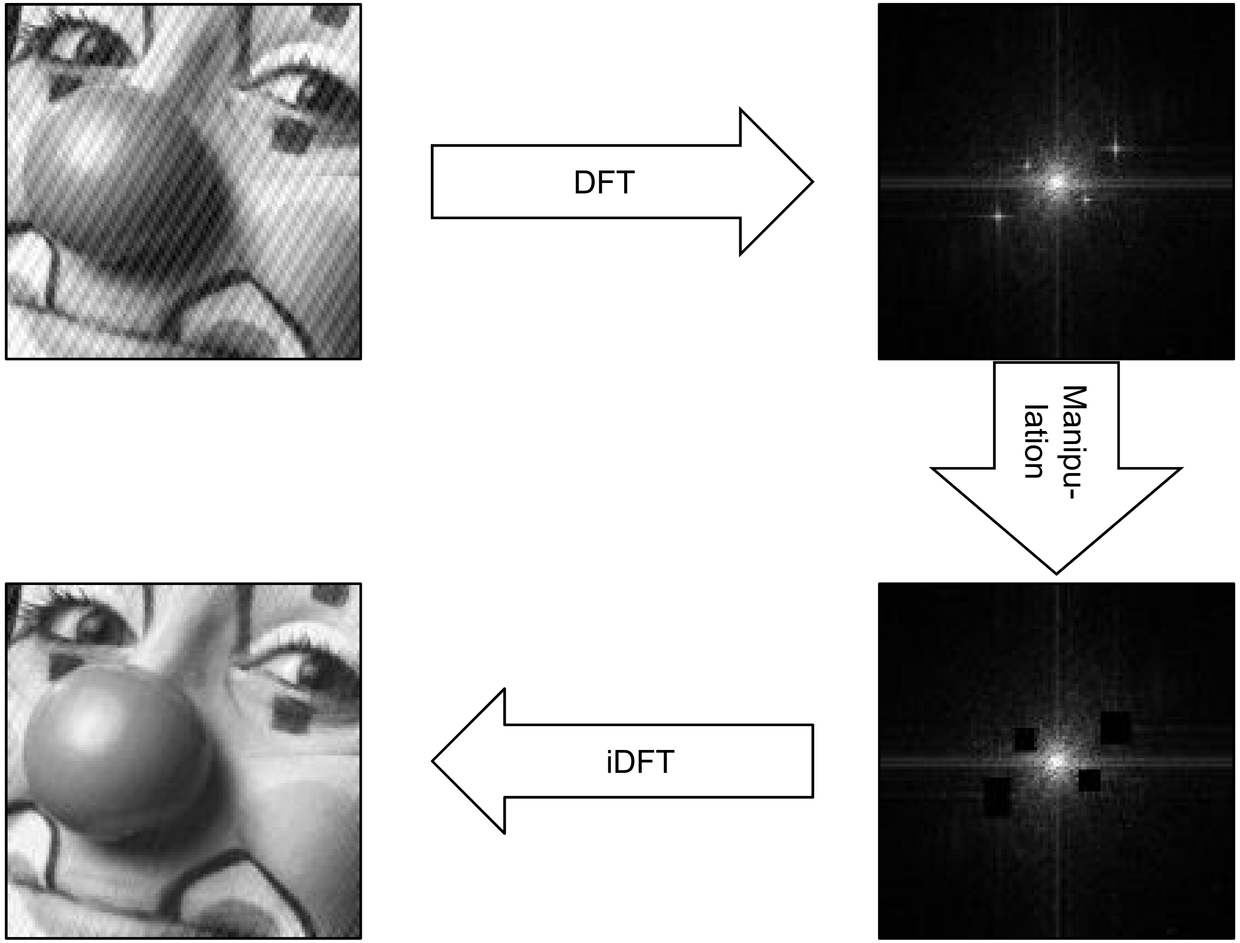
- **Properties:**
  - **Periodicity:**

    $X_{JK} = X_{J+jm,K+kn}$ for $j,k \geq 1$.
  - **Symmetry:**

    $X_{JK} = X^*_{jm-J,kn-K}$ for $j,k \geq 1$.



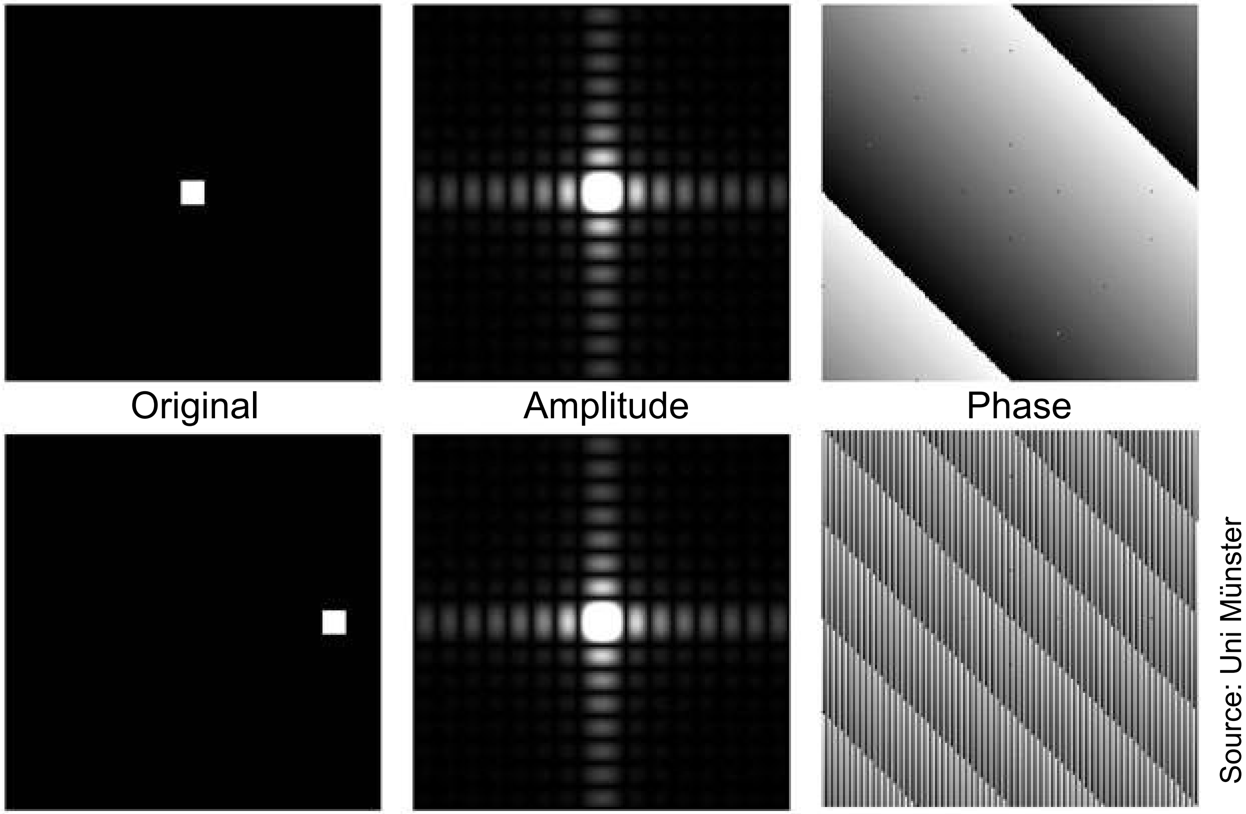
Application example for the 2d DFT.



Is more information in the phase or the amplitude? (1)

- Translation of a function in the spatial domain by (d_x, d_y).

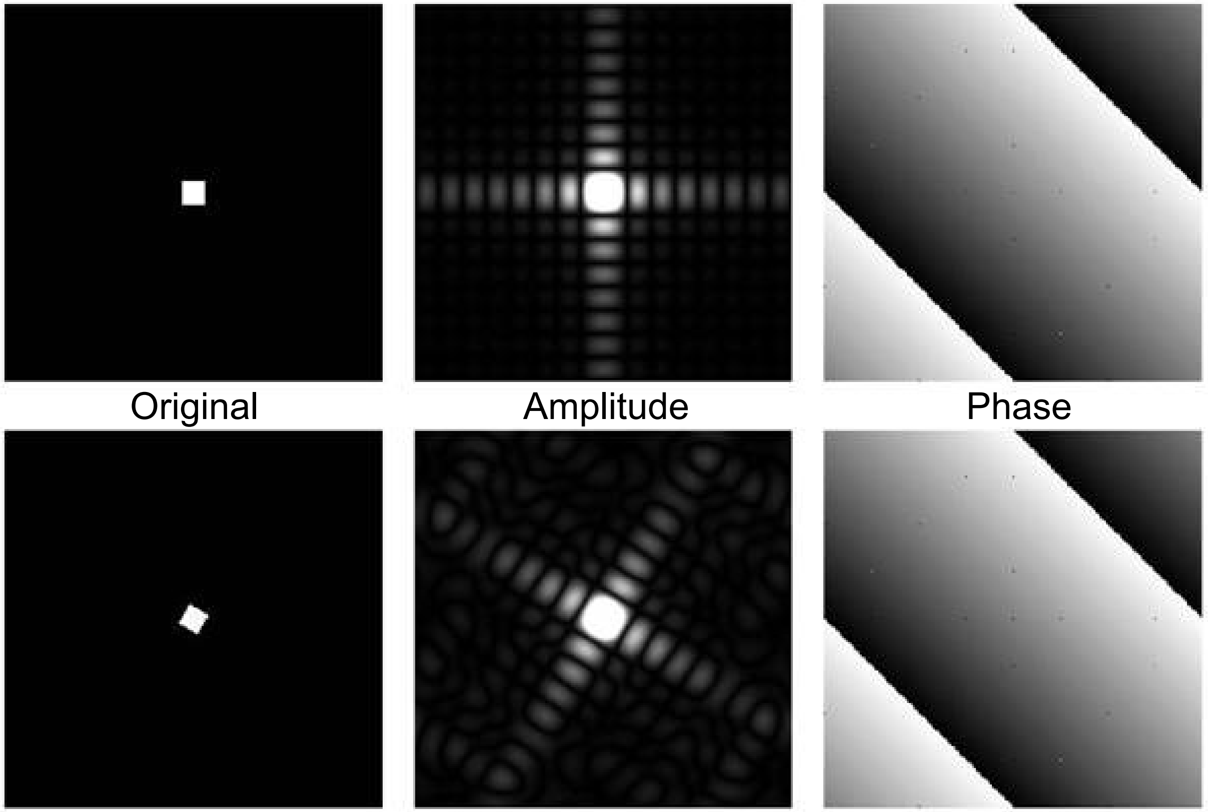
➡ Change of the phase.



Is more information in the phase or the amplitude? (2)

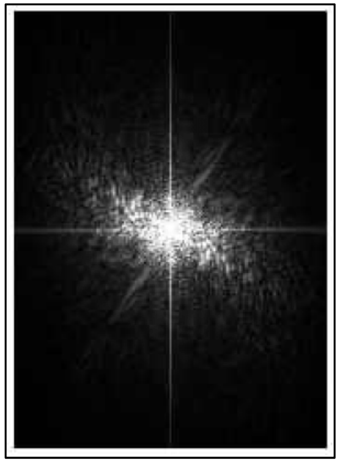
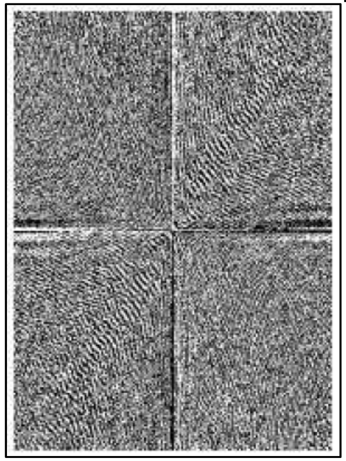
- Rotation of a function in the spatial domain.

➡ The same rotation of the amplitude.

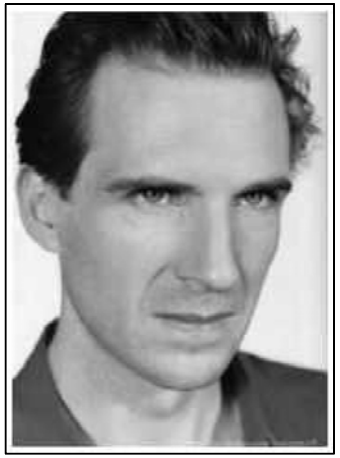
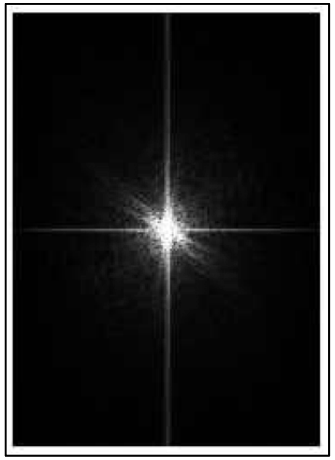
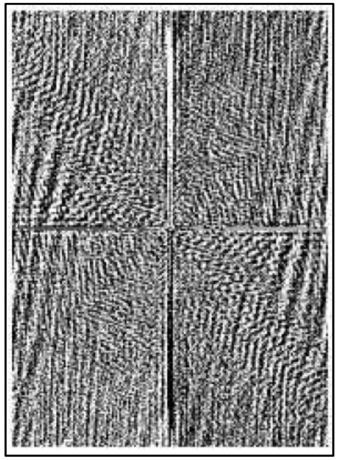


Source: Uni Münster

Is more information in the phase or the amplitude? (3)

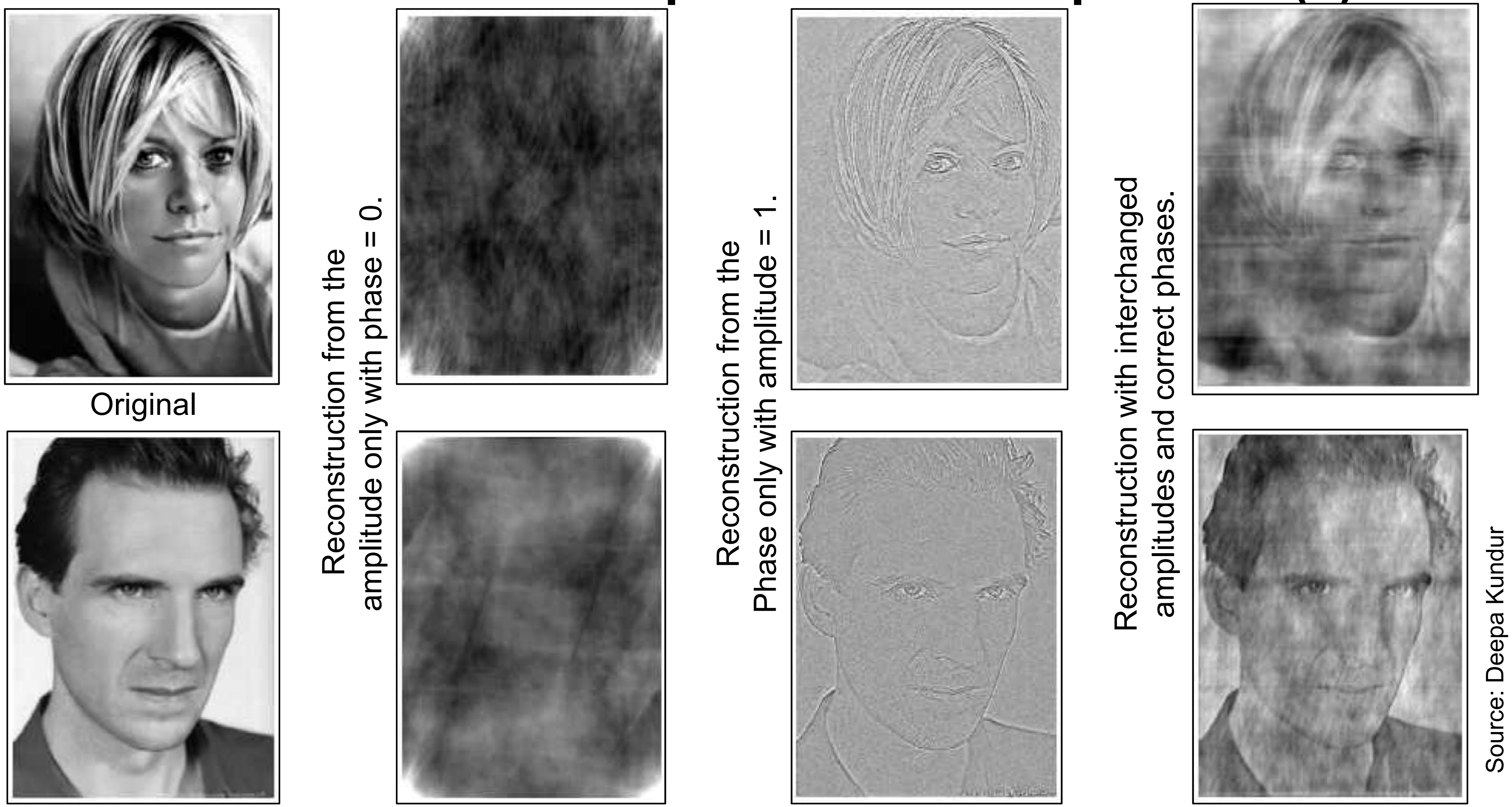




Original | Amplitude | Phase

Source: Deepa Kundur

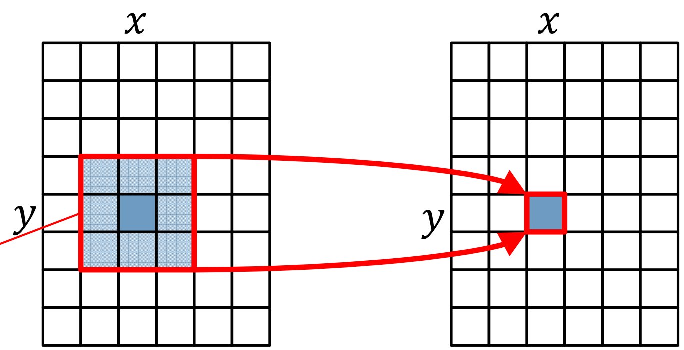
Is more information in the phase or the amplitude? (4)



Filters in 2d (1)

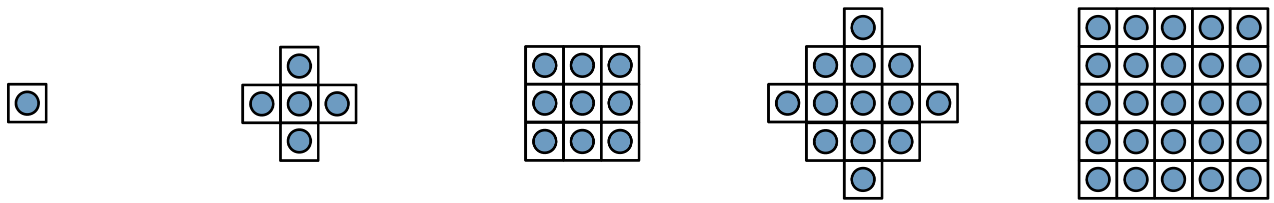
- Compute new pixel-values by a function of the pixel-values of neighboring pixels (kernel)

  Value$(x, y) = F($Neighborhood of $(x, y))$.



- **Neighborhood** (Kernel of the filter)

$$N_c(x, y) = \{(i, j): 0 < (i - x)^2 + (j - y)^2 \leq c\}$$

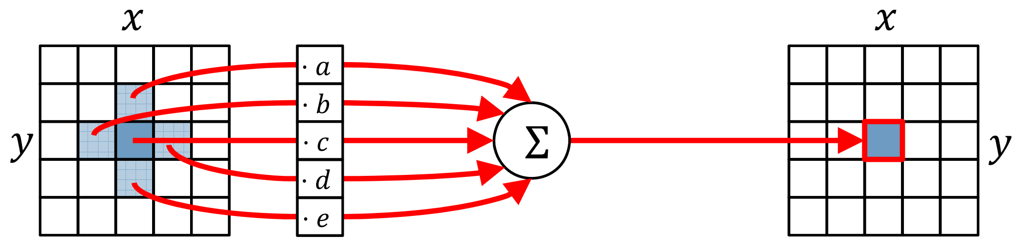


$c = 0$ $c = 1$ $c = 2$ $c = 4$ $c = 8$

4-neighborhood 8-neighborhood

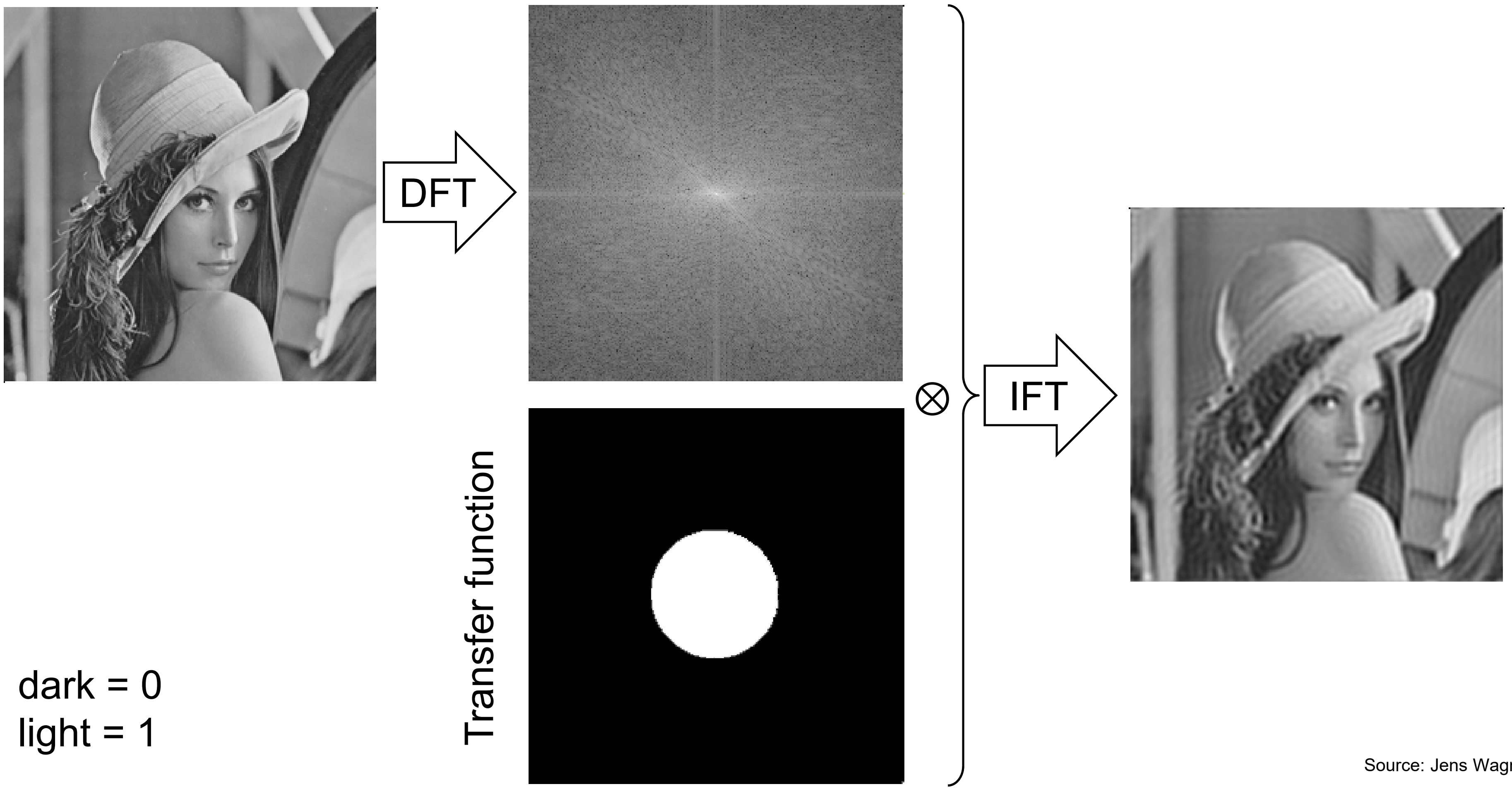
Filters in 2d (2)

- **Linear filters:** Compute new pixel-values by a linear combination of the pixel-values of the filter kernel (corresponds to a convolution).



- **Low-pass filter:** Sum of filter coefficients is one.
  - ➡ Averaging
  - ➡ Noise is removed, image becomes blurred
- **High-pass filter:** Sum of filter coefficients is zero.
  - ➡ Computes differences
  - ➡ Edges are emphasized

Example: Low-pass filter in 2d

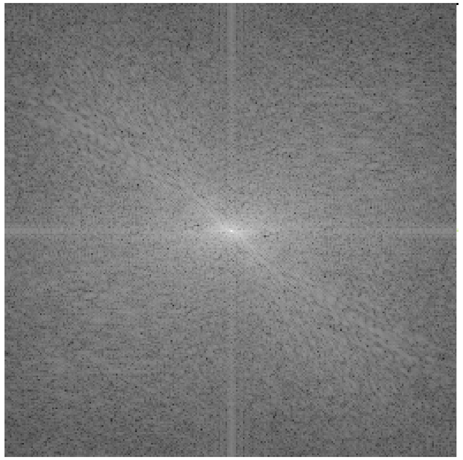


Source: Jens Wagner

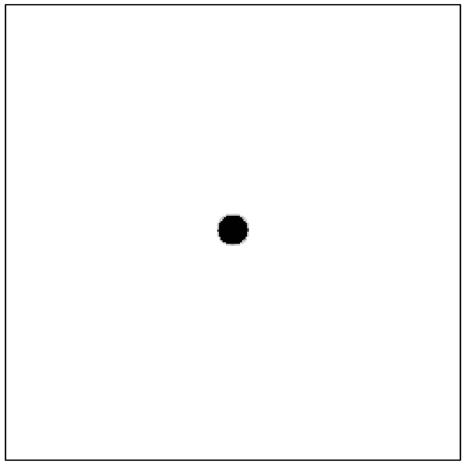
§6.1 Discrete Fourier transformation
§6.1.7 Two-dimensional DFT

Example: High-pass filter in 2d

DFT



Transfer function



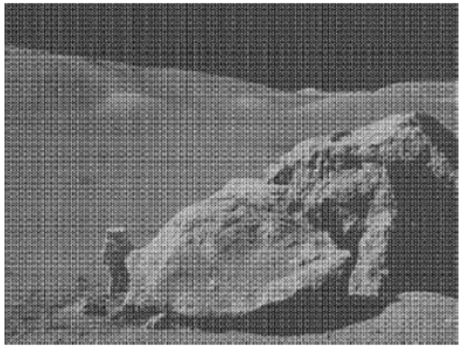
⊗

IFT

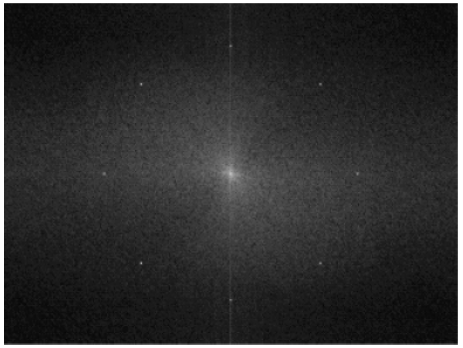
dark = 0
light = 1

Source: Jens Wagner

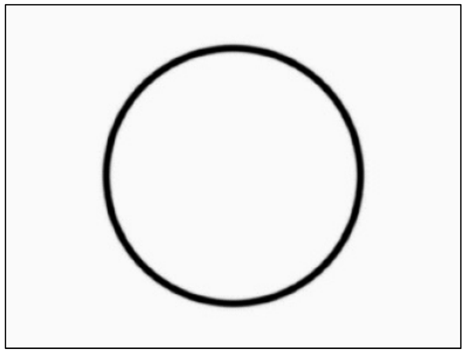
Example: Band-stop filter in 2d



DFT

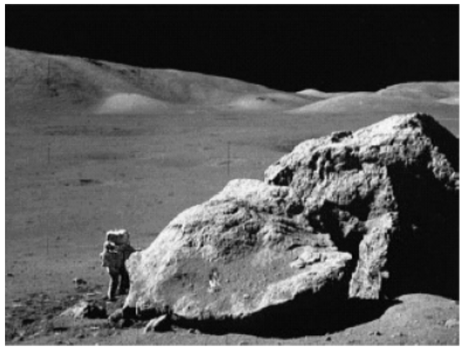


Transfer function



$\otimes$

IFT

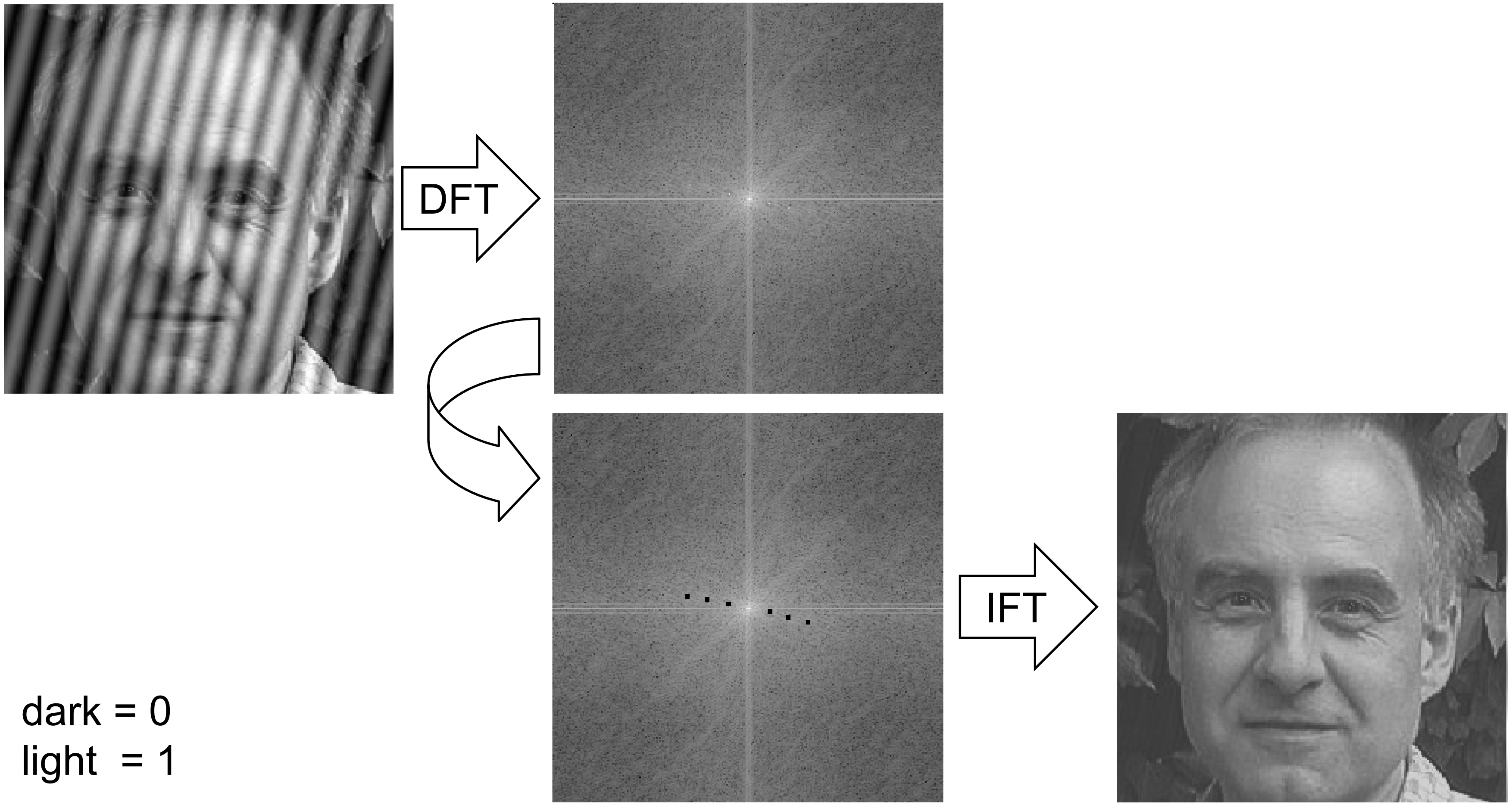


dark = 0
light = 1

Source: Gonzales/Woods

§6.1 Discrete Fourier transformation
§6.1.7 Two-dimensional DFT

Example: Filter in the frequency domain in 2d



dark = 0
light = 1

Source: Jens Wagner

Content

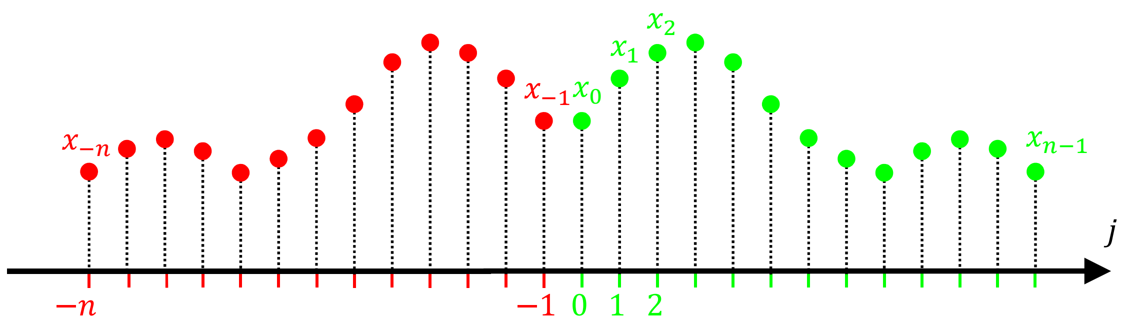
§6.2 Discrete Cosine-Transformation

- **Disadvantage** of the Fourier transform:

  For images the Fourier transform is due to its complex representation rather unhandy and unusual.

- **But:** The Fourier transform of the data $x_j,\ j = 0, \ldots, n-1,$ of a real, even function is real and even.

➡ **Idea:** Imitate data from an even function by doubling the data, i.e. $x_{-(j+1)} := x_j, j = 0, \ldots, n-1$.



§6.2 Discrete Cosine-Transformation

- For the DFT this data yields:

$$X_k = \sum_{j=-n}^{n-1} x_j \omega_{2n}^{-jk}$$

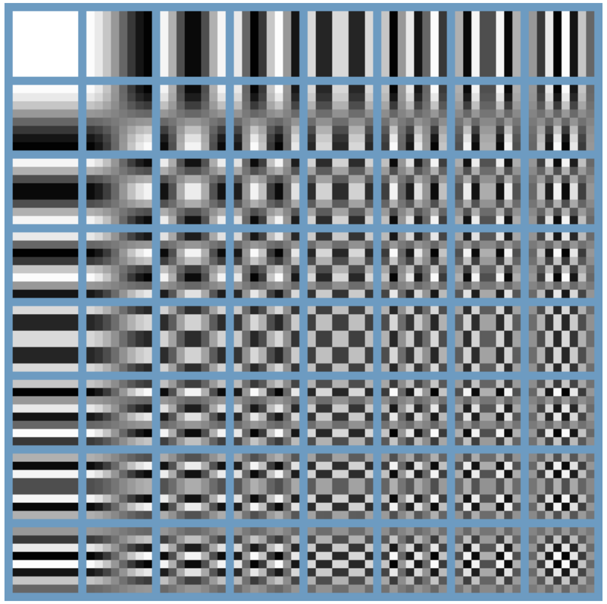
$$= \sum_{j=0}^{n-1} x_j \cos\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)k\right) \in \mathbb{R}, k = 0, \dots, n-1.$$

Short-hand: $X = C_n \cdot x$. (C_n is orthonormal)

- This is the **discrete cosine-transformation** (DCT).
- Remark:
  - There are eight different forms of the DCT, four of them are in regular use.
  - This is the most common form DCT-II.

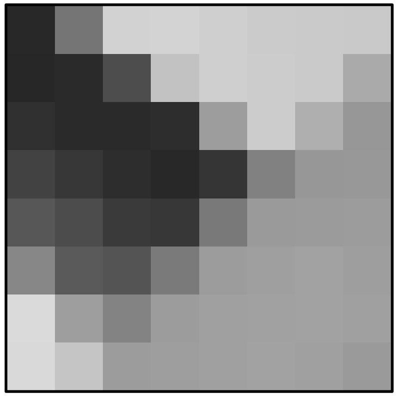
Application

- The DCT is used e.g. in JPG and MPG.
  - In JPG the DCT is applied to (8×8)-blocks of pixels.

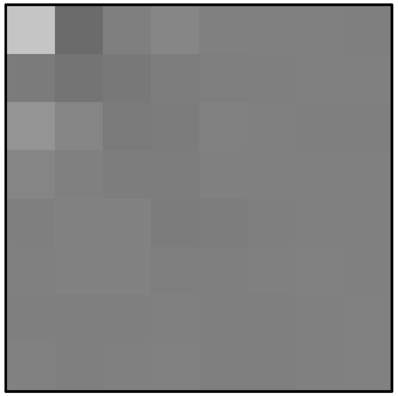


Structure of the DCT-matrix C_8.

Application of the (8×8)-DCT



Original



After DCT

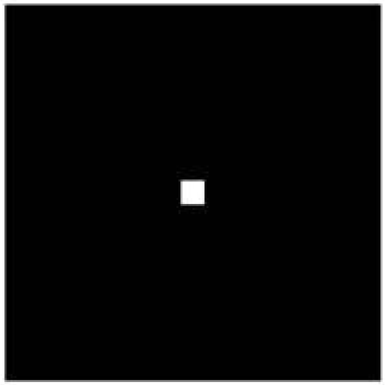
MATLAB-commands for DCT

- `dct` discrete cosine-transform
- `idct` inverse discrete cosine-transform
- `dct2` discrete 2d cosine-transform
- `idct2` inverse discrete 2d cosine-transform

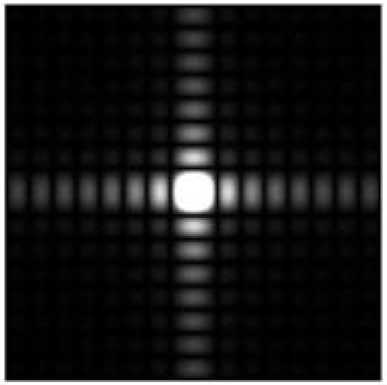
Content

§6.3 Discrete Wavelet-Transformation

- For DFT and DCT the functions are represented as sums of exponential or sins/cosine functions.
- **Disadvantage:** The functions $\cos, \sin, \exp$ have global support.
  - ➡ For every coefficient the complete signal is analyzed.
  - ➡ There is no space/time localization.
  - ➡ These representations are inefficient due to many coefficients that counterbalance local, high frequencies.



Original



Amplitude

§6.3 Discrete Wavelet-Transformation

- Possible **solutions**

  a) Analyze only a time-limited segment of the signal.

  ➡ Spectrogram, windowed Fourier transformation (WFT), short-time FT (STFT), Gabor-Transformation, etc.

  ➡ Leak-effect caused by time-limited segmentation of the signal.

  b) Use local basis functions.

  ➡ Functions can be represented as sums of other functions (basis functions).

➡ **Requirement:** The signal has only finite energy.

§6.3 Discrete Wavelet-Transformation

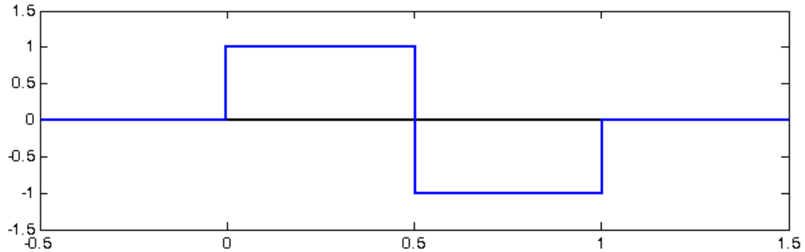
Desirable Properties

- **Local support („window width"):** Localize the function ψ in space or time domain requiring $\int_{-\infty}^{\infty} |\psi(t)|^2 dt < \infty$.
- **Frequency dependent window width:** The larger the frequency the smaller the support to get a detailed frequency resolution.
  - ➡ Stepwise approach to sample every frequency at every location.
- **Orthogonality:** simple inversion, simple and robust computations.
- **„Wave property":** Use functions ψ with $\int_{-\infty}^{\infty} \psi(t) dt = 0$.
  - ➡ This property infers the name **wavelet**.
  - ➡ $\Psi(0) = 0$ for the Fourier-transform Ψ of ψ.
  - ➡ A wavelet ψ acts like a band-pass filter.

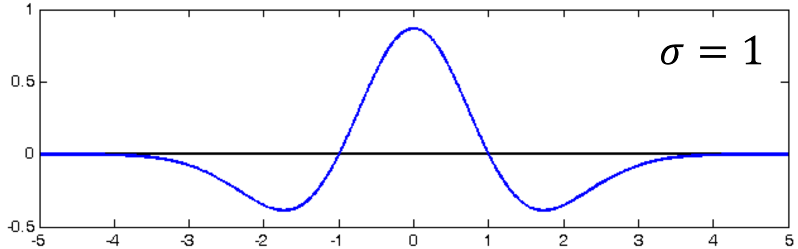
§6.3 Discrete Wavelet-Transformation

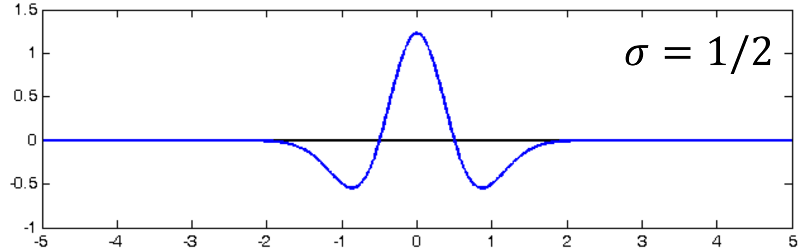
Example (1)

- Haar-Wavelet $\psi(t) = \begin{cases} 1, t \in [0, \frac{1}{2}) \\ -1, t \in [\frac{1}{2}, 1) \\ 0, t \notin [0,1) \end{cases}$
  - Compact support, discontinuous



- Mexican-Hat $\psi(t) = \frac{c}{\sqrt{\sigma}}\left(1 - \frac{t^2}{\sigma^2}\right) e^{-\frac{t^2}{2\sigma^2}}$
  - Unbounded support, C^∞

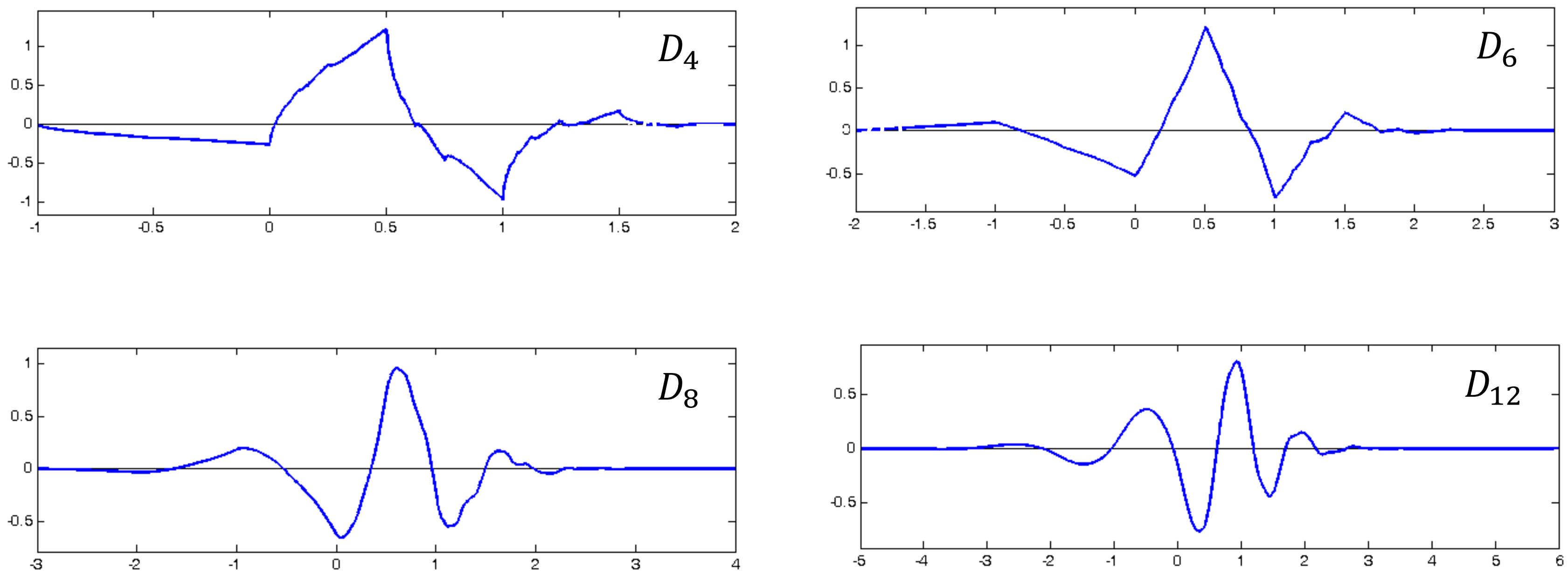




§6.3 Discrete Wavelet-Transformation

Example (2)

- Daubechies-Wavelets D_N, $N \in 2\mathbb{N}$.
  - Compact support, C^k with $k \geq 1$ for $N \geq 8$.



§6.3 Discrete Wavelet-Transformation

Principle (CTW – continuous wavelet transform)

- The function ψ is called **mother-wavelet**.
- ➡ How can a signal f be represented using the wavelet ψ?
- ➡ **Wavelet-Synthesis**

$$f(t) = \frac{1}{C_\psi} \int_{\mathbb{R}} \int_{\mathbb{R}^+} d_{a,b} \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right) \frac{da\, db}{a^2}$$

with $\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right)$ (dilation a, translation b)

and $d_{a,b} = \int_{\mathbb{R}} f(t) \psi_{a,b}(t) dt$ (**Wavelet-Analysis**)

and $C_\psi = \int_0^\infty \frac{|\Psi(\omega)|^2}{\omega} d\omega$ (admissibility condition).

§6.3 Discrete Wavelet-Transformation

Discrete Time-Wavelet-Transformation (DTWT) (1)

- Use only discrete translations and dilations.
  - Example: $a = a_0^{-j},\ b = k \cdot b_0 \cdot a_0^{-j}$

$$\psi_{j,k} = a_0^{j/2}\,\psi\left(a_0^j t - k\,b_0\right), j,k \in \mathbb{Z}.$$

  - Especially: $a_0 = 2, b_0 = 1$ (dyadic grid):

$$\psi_{j,k} = 2^{j/2}\,\psi\left(2^j\,t - k\right), j,k \in \mathbb{Z}.$$

➡ **Discrete time-wavelet transform** (DTWT)

$$f(t) = \sum_j \sum_k d_{j,k}\psi_{j,k}(t) \text{ with}$$

$$d_{j,k} = a_0^{j/2} \int f(t)\psi_{j,k}(t)dt.$$

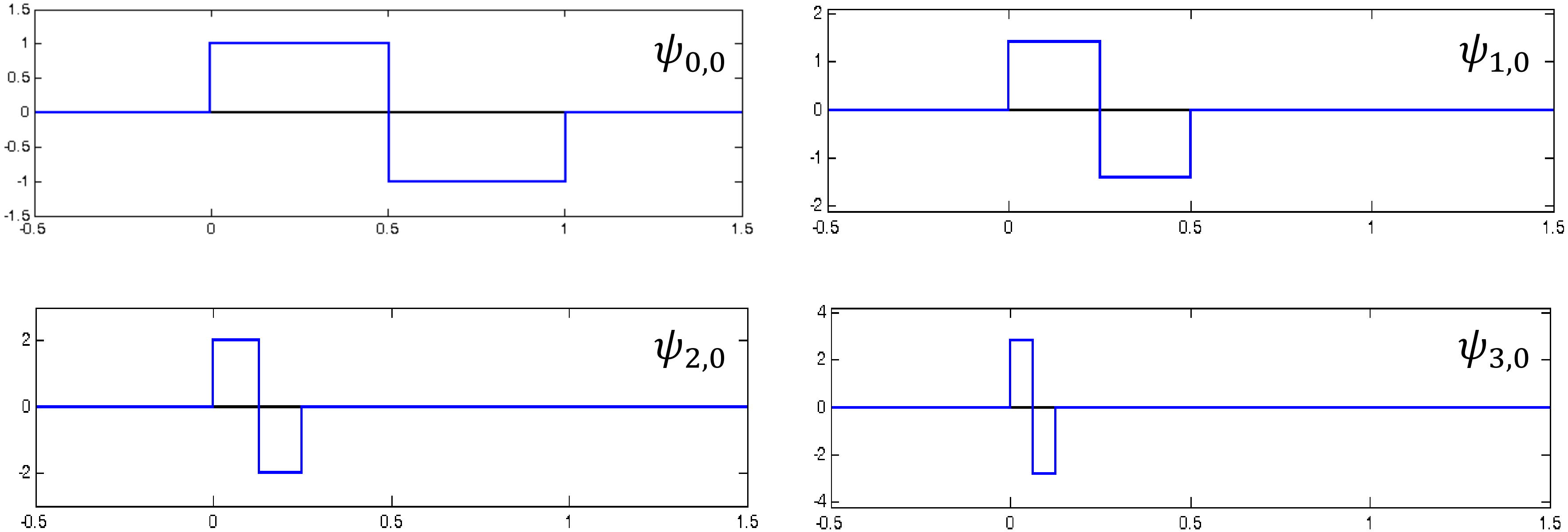
  - Here $t \in \mathbb{R}$ is a continuous variable.

§6.3 Discrete Wavelet-Transformation

Discrete Time-Wavelet-Transformation (DTWT) (2)

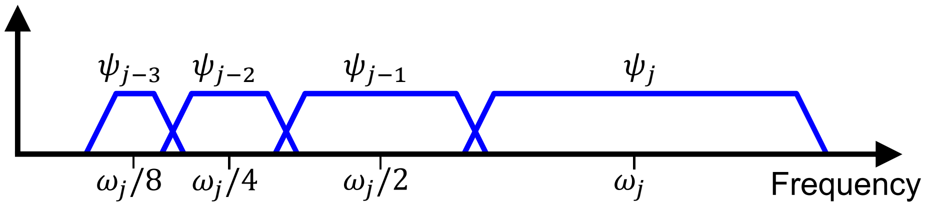
- **Example:**

  Haar-Wavelet $\psi_{0,0}(t) = \begin{cases} 1, t \in [0, \frac{1}{2}) \\ -1, t \in [\frac{1}{2}, 1) \\ 0, t \notin [0,1) \end{cases}$ and $\psi_{j,0} = 2^{j/2}\,\psi\left(2^j\ t\right)$.



Discrete Time-Wavelet-Transformation (DTWT) (3)

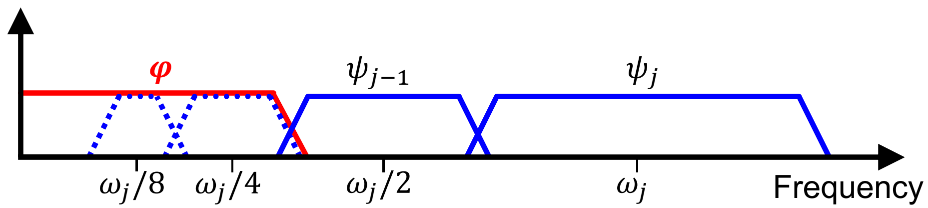
- Practical problem: The sum $f(t) = \sum_j \sum_k d_{j,k}\psi_{j,k}(t)$ runs over infinitely many translations and dilations.
- The signal has finite length in time domain.
  - ➡ Only a finite number of translations need to be considered.
- The signal has finite length in frequency domain (band-limited).
  - ➡ Expansion/dilation in time domain yields dilation/expansion in frequency domain.
  - ➡ Every scaling of a wavelet in time domain by factor 2 halves its band width.



  - ➡ You need infinitely many scalings to yield a band width of zero.

The scaling function (1)

- **Idea:** Cut-off the sum and use a special function to represent the remaining rest of the signal, the so-called **scaling function** φ.



- The scaling function corresponds to a low-pass filter.
- Its Fourier transform must satisfy

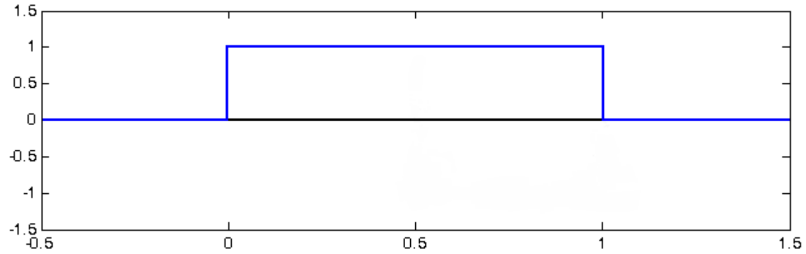
$$\Phi(0) = \int \varphi(t)dt = 1.$$

§6.3 Discrete Wavelet-Transformation

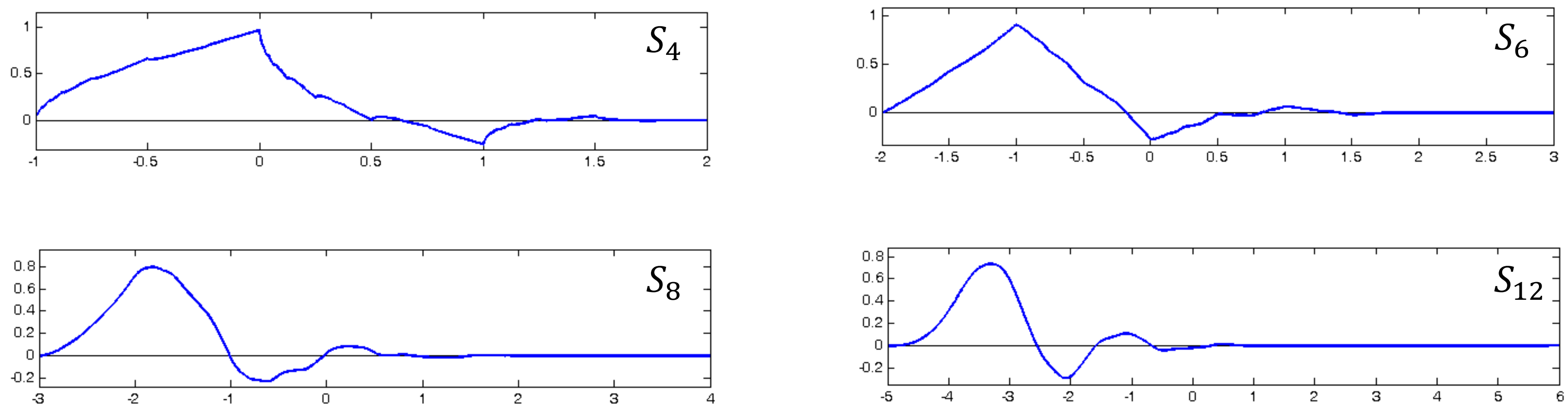
Examples for scaling functions

- Haar-Wavelet:

$$\varphi(t) = \begin{cases} 1, t \in [0,1) \\ 0, t \notin [0,1) \end{cases}$$



- Daubechies-Wavelets: scaling function S_N for the wavelet D_N



The scaling function (2)

- The scaling function is a combination of an infinite number of wavelets up to scale m:

$$\varphi_m(t) = \sum_{j=-\infty}^{m} \sum_{k} \gamma_{j,k} \psi_{j,k}(t).$$

- Translation of the scaling function $\varphi_{m,k}(t) = \varphi_m(t-k)$ yields the components of the signal, that got lost by cutting off the wavelet representation, i.e. $V_m = \mathrm{span}(\varphi_{m,k}(t))$.
- The signal f is decomposed into a scaling function component and a component of a finite number of wavelets:

$$f(t) = \sum_{k} c_{m,k} \varphi_{m,k}(t) + \sum_{j \geq m+1} \sum_{k} d_{j,k} \psi_{j,k}(t).$$

§6.3 Discrete Wavelet-Transformation

The scaling function (3)

- The scaling functions $\varphi_{m,k}$ span spaces V_m which define a **multiresolution analysis**:

$$\cdots \subset V_m \subset V_{m+1} \subset V_{m+2} \subset \cdots.$$

- What is the difference between the spaces V_m and V_{m+1}?

$$V_{m+1} \setminus V_m = \operatorname{span}(\psi_{m+1,k}) =: W_m.$$

$$\Rightarrow \quad V_m \oplus W_m = V_{m+1}.$$

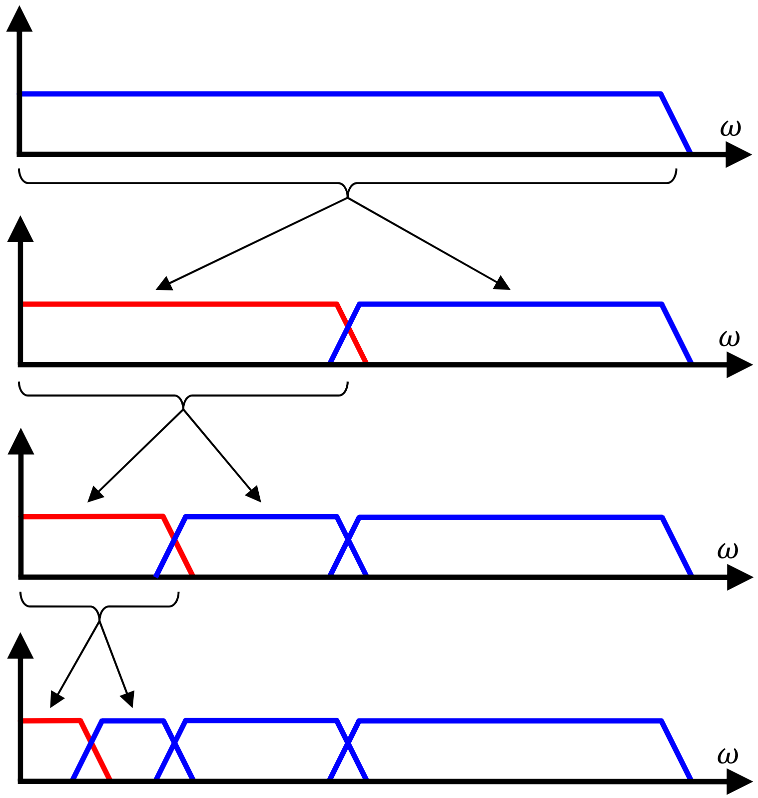
- Iterating this relation yields:

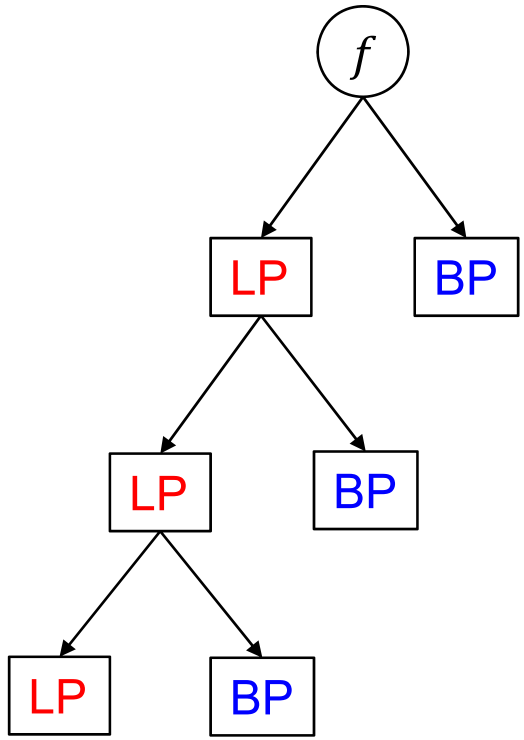
$$V_{m+1} = V_j \oplus W_j \oplus W_{j+1} \oplus \cdots \oplus W_m.$$

Stephane Mallat

Sub-band-Coding

- Analyze the signal using a filter bank of band-pass and low-pass filters.





§6.3 Discrete Wavelet-Transformation

How to compute the coefficients of a signal?

- Because of $V_m \oplus W_m = V_{m+1}$ we get in particular:
  - $V_m \subset V_{m+1}$: $\quad \varphi(t) = \sum_k h_k \sqrt{2}\varphi(2t-k)$
  - $W_m \subset V_{m+1}$: $\quad \psi(t) = \sum_k g_k \sqrt{2}\varphi(2t-k)$

  with $g_k = (-1)^k h_{1-k}$.

➡ The coefficients h_k and g_k are the coefficients of a low-pass and a high-pass filter.

➡ The coefficients $c_{j,k}$ und $d_{j,k}$ can be computed by filter (and subsequent **down-sampling**), i.e. convolution with the filter coefficients

$$c_{j,k} = \sum_l h_{l-2k} c_{j+1,l} \text{ and}$$

$$d_{j,k} = \sum_l g_{l-2k} c_{j+1,l}.$$

How to reconstruct the signal from the coefficients?

➡ The inverse DWT (IDWT, reconstruction) uses the adjoint filters, i.e.

$$c_{j,k} = \sum_l h_{k-2l} c_{j-1,l} + \sum_l g_{k-2l} c_{j-1,l} \,.$$

§6.3 Discrete Wavelet-Transformation

Filter-examples:

- Haar Wavelet: $h_0 = h_1 = \frac{1}{\sqrt{2}}$ and $g_0 = \frac{1}{\sqrt{2}}, g_1 = -\frac{1}{\sqrt{2}}$.
- Daubechies-Wavelet:

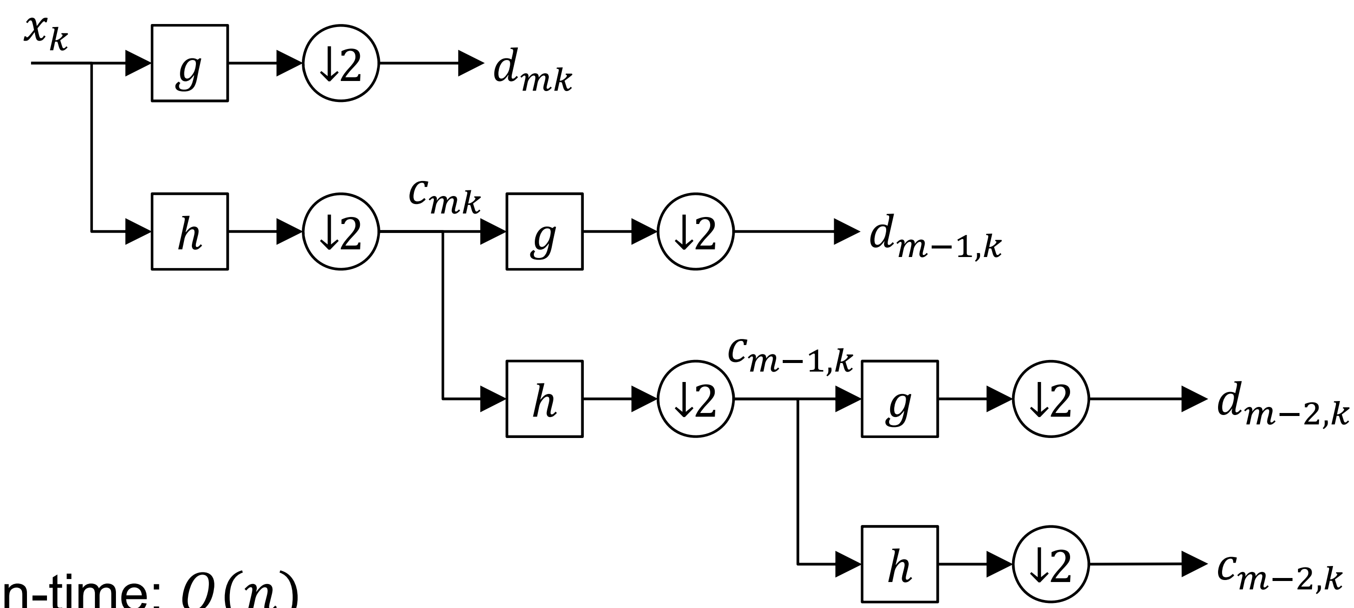
D_2	D_4	D_6	D_8	D_{10}	D_{12}
1	0,6830127	0,47046721	0,32580343	0,22641898	0,15774243
1	1,1830127	1,14111692	1,01094572	0,85394354	0,69950381
	0,3169873	0,650365	0,8922014	1,02432694	1,06226376
	−0,1830127	−0,19093442	−0,03967503	0,19576696	0,44583132
		−0,12083221	−0,26450717	−0,34265671	−0,31998660
		0,0498175	0,0436163	−0,04560113	−0,18351806
			0,0465036	0,10970265	0,13788809
			−0,01498699	−0,00882680	0,03892321
				−0,01779187	−0,04466375
				4,71742793 e−3	7,83251152 e−4
					6,75606236 e−3
					−1,52353381 e−3

Haar-Wavelet

Table of un-normalized h_k

Computation of the DWT (1)

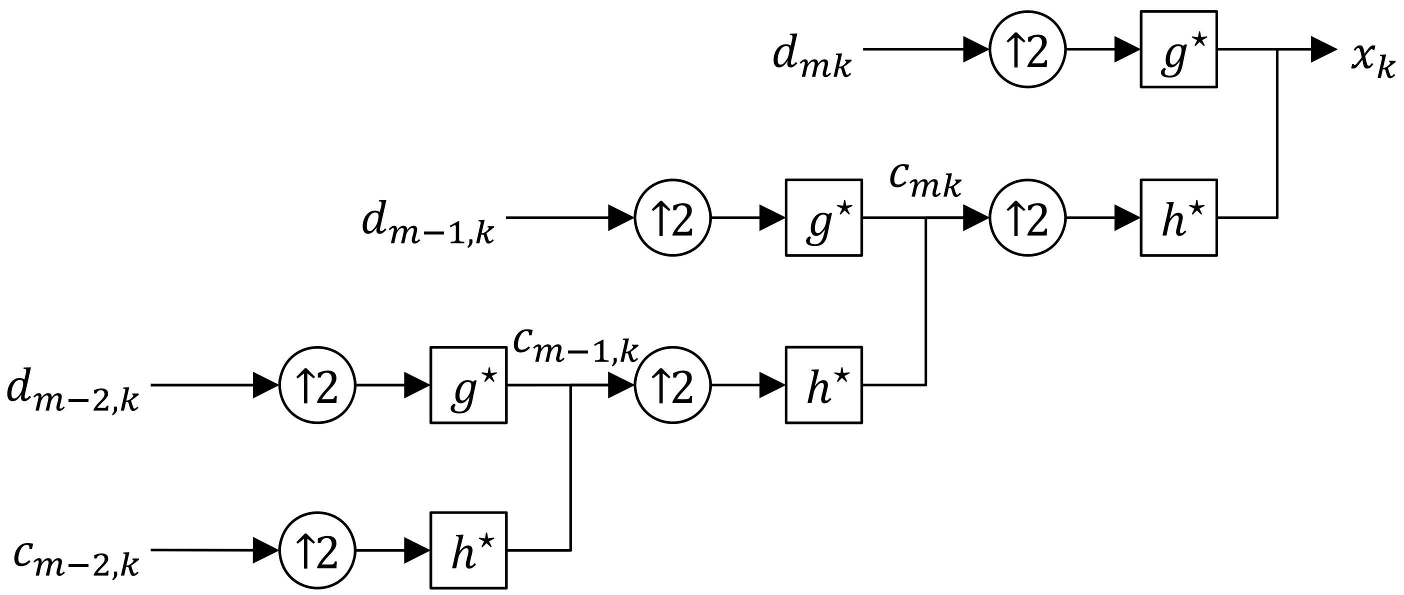
- The computation of the DWT is described by a pyramid-scheme :



- Run-time: $O(n)$
- Termination: When there is only one coefficient left.

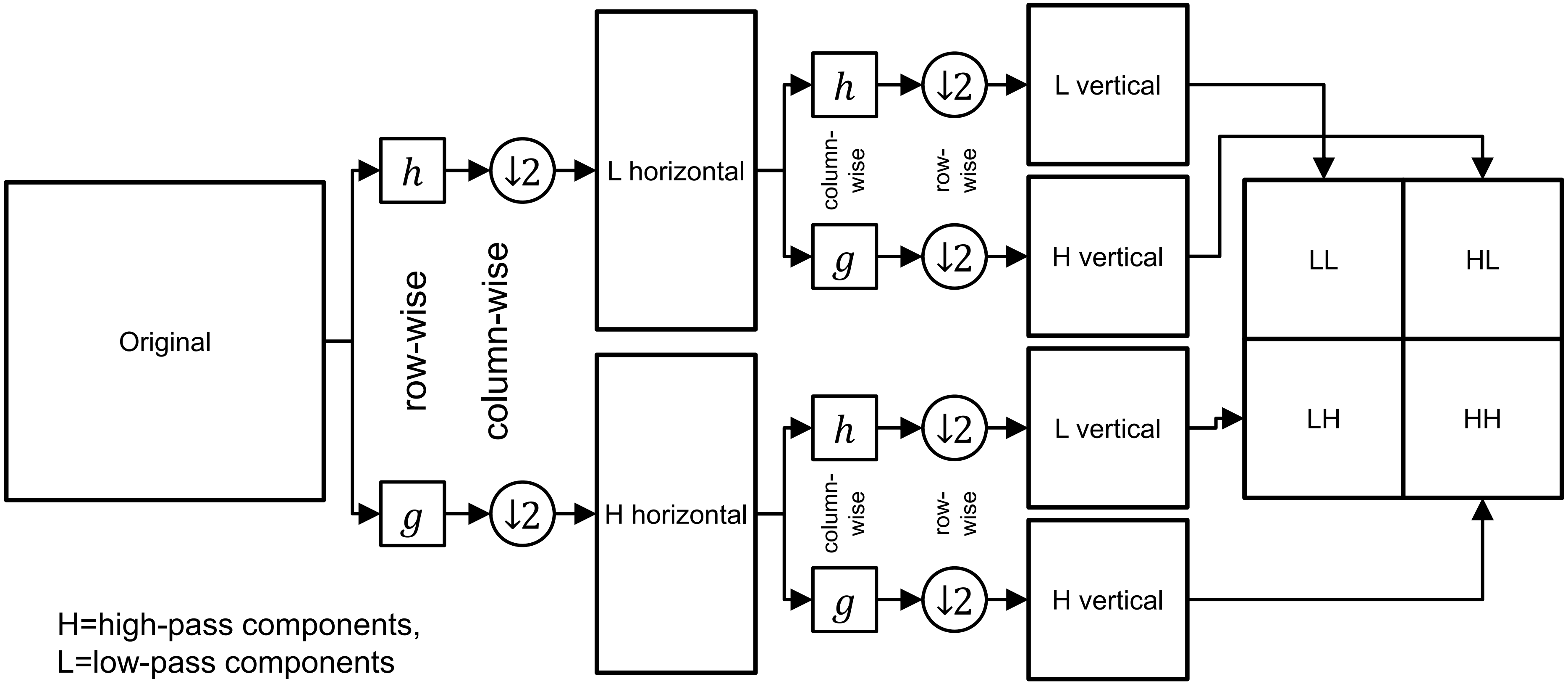
Computation of the DWT (2)

- The computation of the IDWT is described by the inverse pyramid-scheme:



Representation of the 2d-DWT (1)

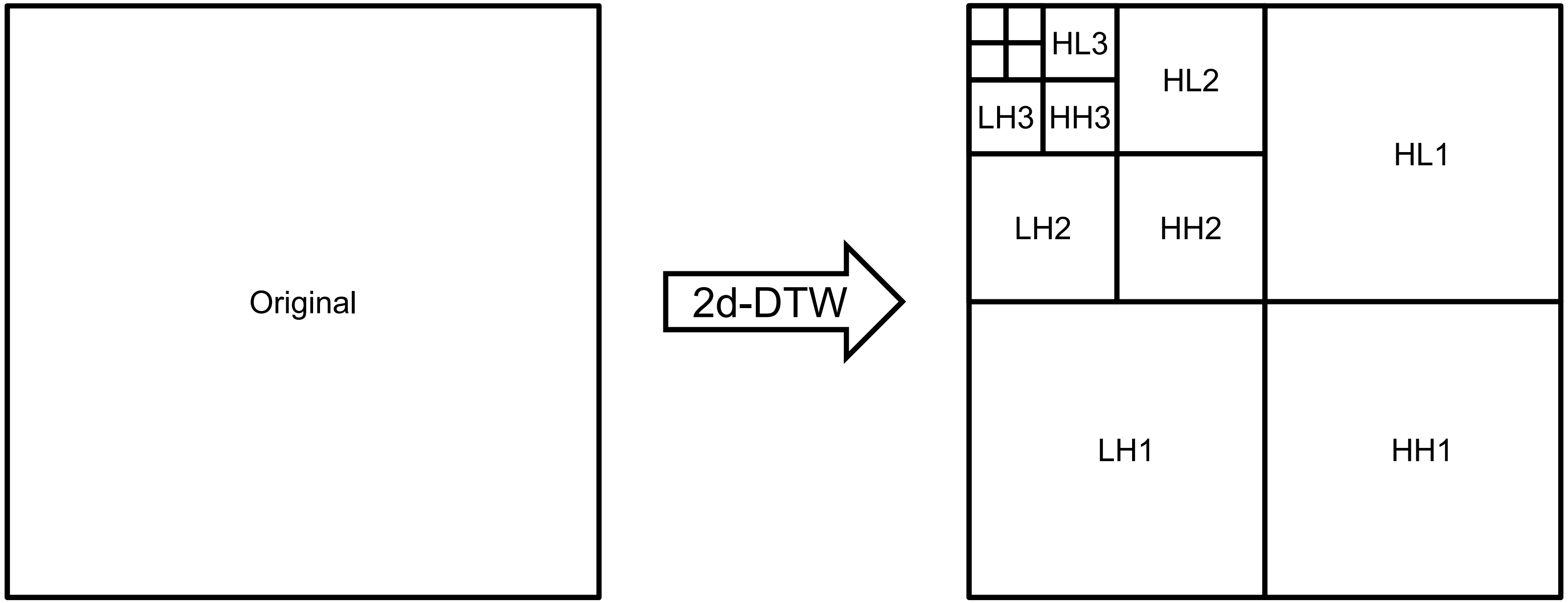
- Apply the DWT to the row- and columns of discrete 2d-data ...



H=high-pass components,
L=low-pass components

Representation of the 2d-DWT (2)

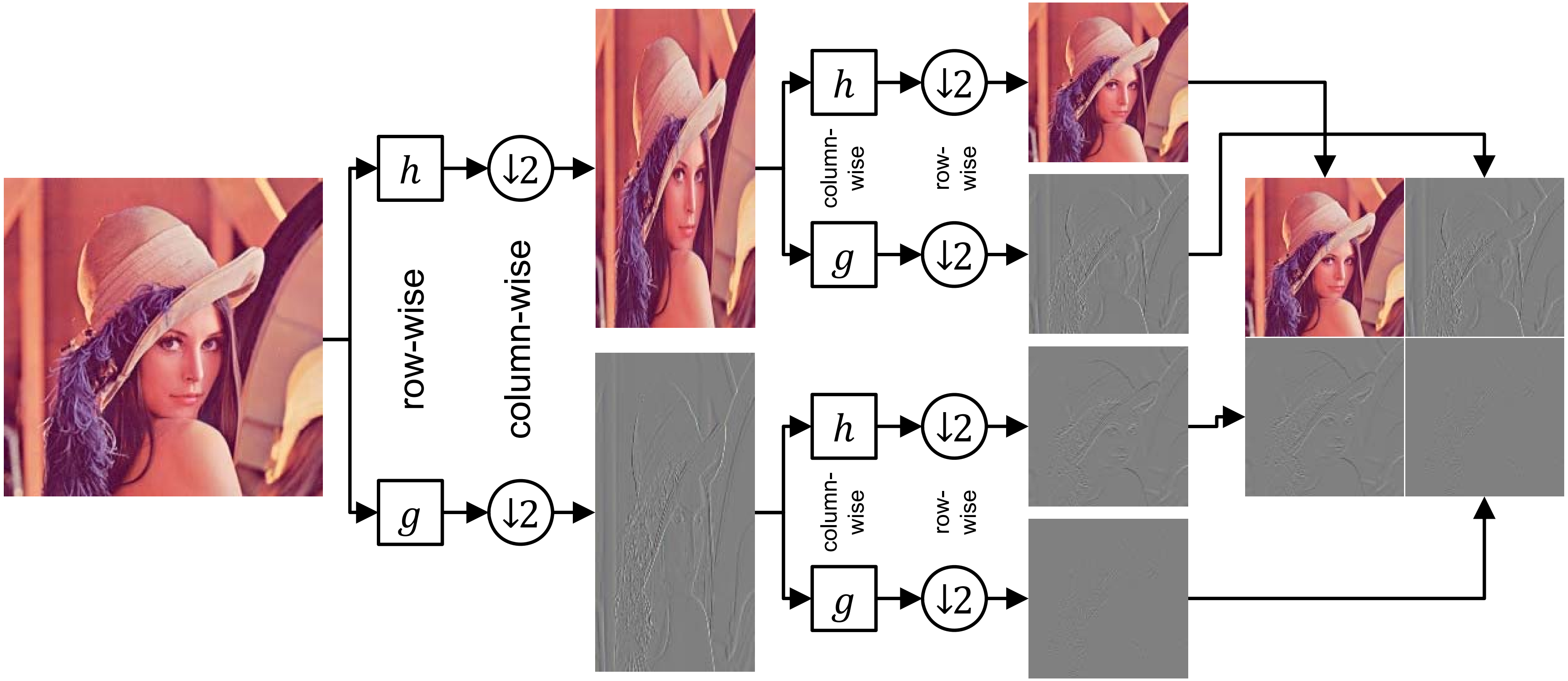
- Apply the DWT to the row- and columns of discrete 2d-data and iterate.



H=high-pass components, L=low-pass components

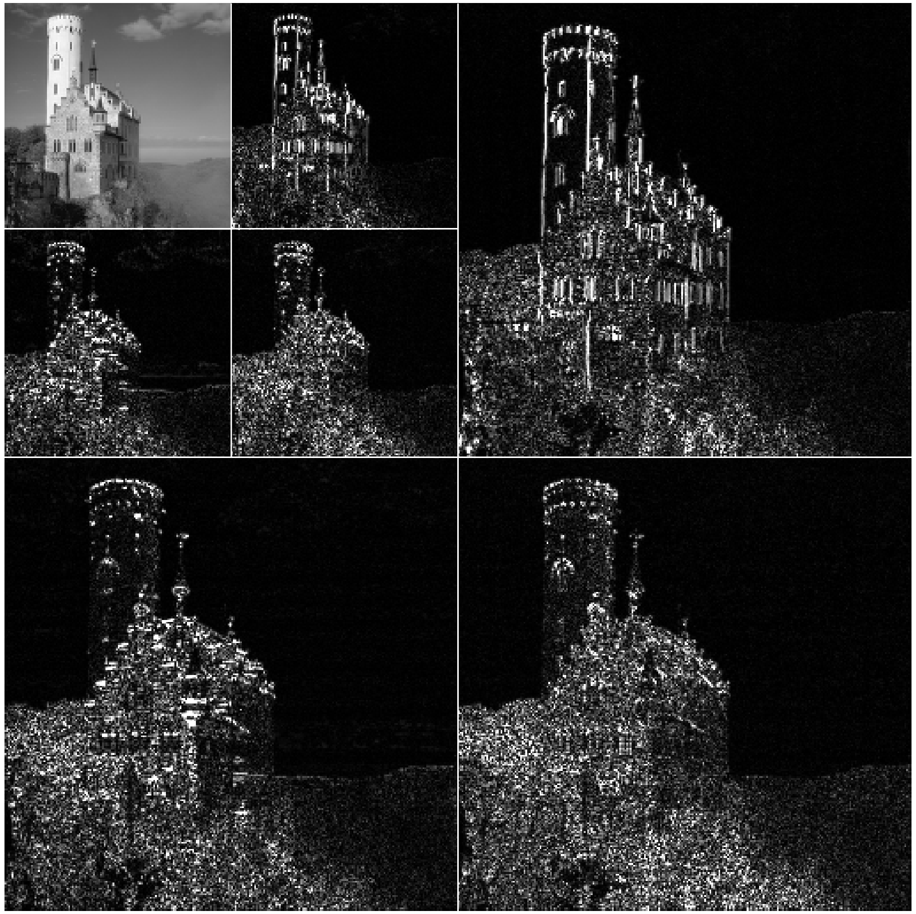
§6.3 Discrete Wavelet-Transformation

Example 2d-DWT (1)



Example 2d-DWT (2)

Example 2d-DWT (3)



Source: wikipedia

§6.3 Discrete Wavelet-Transformation

Application

- Wavelets are used e.g. in JPEG2000 or by the FBI to store finger prints.
- Example: JPEG2000

Original

compression 1:25

compression 1:50

Goals

- What is DFT, FFT, DCT, DWT?
- What are the pro and cons of DFT, FFT, DCT, DWT?
- How do you apply DFT, DCT, DWT to images?
- How can a DWT be realized using filters?