

Beispiel: FFT für $n=3^q$

$$\begin{aligned}
 v_j &= \sum_{k=0}^n c_k e^{2\pi i j k / n} = \sum_{k=0}^{n/3-1} c_{3k} e^{2\pi i j 3k / n} + \\
 &+ \sum_{k=0}^{n/3-1} c_{3k+1} e^{2\pi i j (3k+1) / n} + \sum_{k=0}^{n/3-1} c_{3k+2} e^{2\pi i j (3k+2) / n} = \\
 &= F_0 + e^{2\pi i j / n} F_1 + e^{2\pi i 2 j / n} F_2
 \end{aligned}$$

Dazu benötigt

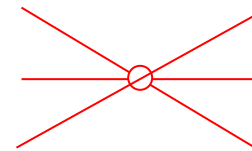
- IDFT(c_0, c_3, c_6, \dots)
- IDFT(c_1, c_4, c_7, \dots)
- IDFT(c_2, c_5, c_8, \dots)

Sortieren modulo 3: Bitreversal in ternärer Darstellung
z.B. dreistellig für $n = 3^3 = 27$:

$$5 = 1 \cdot 3 + 2 \cdot 1 = (012)_3 \rightarrow (210)_3 = 2 \cdot 9 + 1 \cdot 3 = 21$$

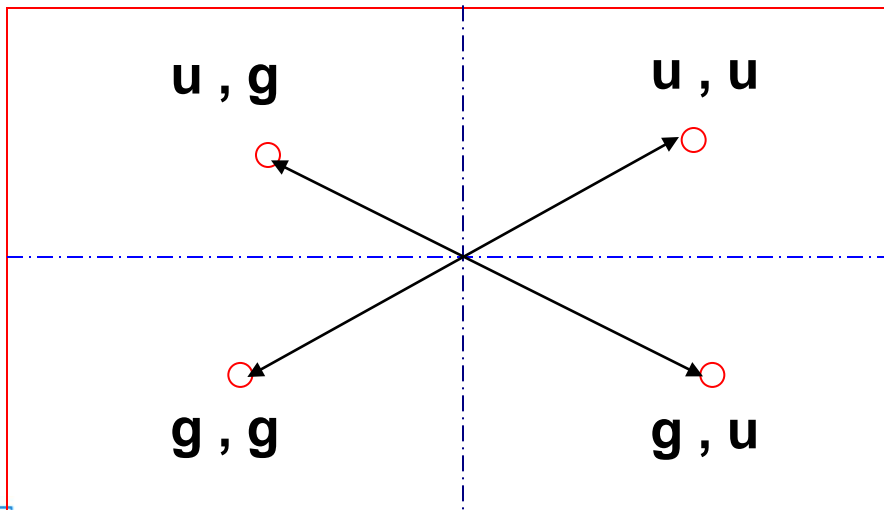
In Sortierphase wandert c_5 an die Position c_{21} .

„Dreier-Butterfly“



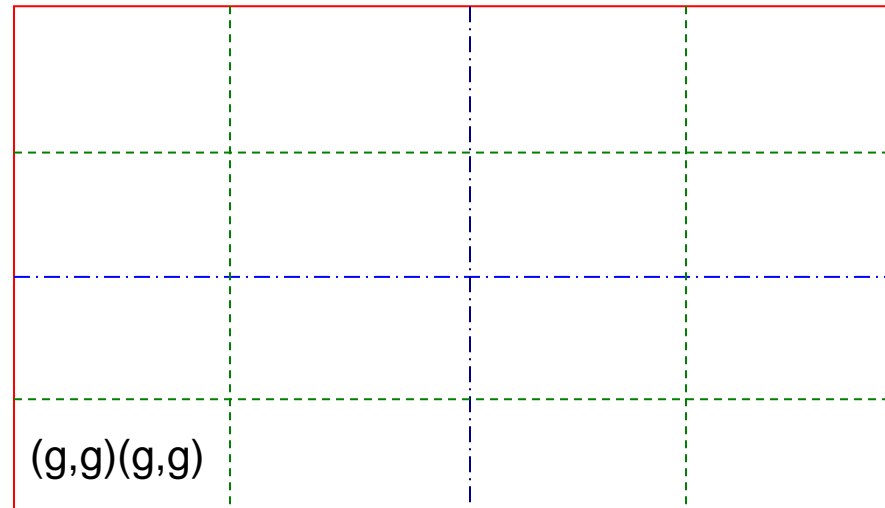
Beispiel: FFT in 2D

$$\begin{aligned}
 v_{j,s} &= \sum_{k=0}^N \sum_{r=0}^M c_{k,r} e^{2\pi ijk/N} e^{2\pi irs/M} = \\
 &= \sum_{k=0}^{N/2-1} \sum_{r=0}^{M/2-1} c_{2k,2r} e^{2\pi ij2k/N} e^{2\pi i2rs/M} + c_{2k,2r+1} e^{2\pi ij2k/N} e^{2\pi i(2r+1)s/M} \\
 &+ c_{2k+1,2r} e^{2\pi ij(2k+1)/N} e^{2\pi i2rs/M} + c_{2k+1,2r+1} e^{2\pi ij(2k+1)/N} e^{2\pi i(2r+1)s/M} = \\
 &= F_{00} + e^{2\pi is/M} F_{10} + e^{2\pi ij/N} F_{01} + e^{2\pi ij/N} e^{2\pi is/M} F_{11}
 \end{aligned}$$



Butterfly kombiniert 4
Komponenten zu 4 neuen
Komponenten

Rekursiv weiter aufspalten:



Doppelsumme erlaubt unterschiedliches Abarbeiten:

Entweder 1-dim. FFT in Zeilen + 1-dim. FFT in Spalten,
 oder
 rekursiv 2-dim. FFT in Blöcken.



1-dim. FFT für allgemeines n :

Allgemein kann man bei einer Primfaktorzerlegung der Form $n = n_1 \dots n_p$ eine Aufspaltung in n_1 Teilsummen im ersten Schritt, \dots , in n_p Teilsummen im letzten Schritt ausführen.

Ist z.B. $n=8$, so lässt sich die IDFT-Summe aufspalten in 2-2-2, 2-4, 4-2, oder 8 in 3, 2, oder 1 Level!

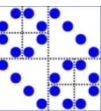
Ist n eine Primzahl \rightarrow FFT geht so nicht!

Durch gruppentheoretische Überlegungen lässt sich dann aber z.B. dieser Fall auf die Berechnung der FFT zu $n-1$ und $n-2$ zurückführen, die dann keine Primzahlen sind.

Andere Idee: Zero Padding oder Einbettung in Zweier-Potenz.



Verallgemeinerung auf allgemeine Stützstellen: NFFT



Teste für CPU, Cache, usw. die verschiedenen Faktorzerlegungen von n .

Wähle diejenige mit schnellster Laufzeit aus und erstelle automatisch FFT-Routine.

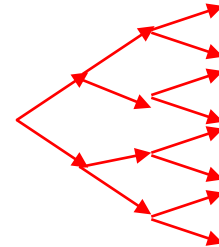
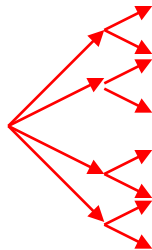
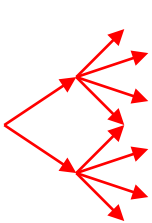
$$n = 8 = 1 * 8 = 2 * 4 = 2 * 2 * 2 = 4 * 2 = 8 * 1$$

IDFT Länge 8 oder

IDFT Länge 2 gefolgt von IDFT 4 oder

IDFT Länge 2, IDFT Länge 2, IDFT Länge 2 oder

IDFT Länge 4 gefolgt von IDFT 2



5.3. Anwendungen

5.3.1. Fourierreentwicklung:

Stückweise stetige, 2π -periodische (in $[-\pi, \pi]$) Funktion $f(x)$ lässt sich als Fourier-Reihe darstellen (vgl. Taylor/Potenz-Reihe):

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_k \cos(kx) + b_k \sin(kx))$$
$$\left\{ \equiv \sum_{k=-\infty}^{k=\infty} c_k e^{ikx} = \sum_{k=-\infty}^{k=\infty} c_k (e^{ix})^k = \sum c_k z^k \right\}$$

a_k und b_k heißen Fourier-Koeffizienten von $f(x)$ und berechnen sich (wegen der Orthogonalität der Funktionen $\cos(kx)$ und $\sin(kx)$) aus

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(kx) dx, \quad b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(kx) dx$$

Sie geben die Größe der Anteile von Vielfachen der Grundfrequenz ($1/2\pi$) an, aus denen sich f zusammensetzt

Wellenzahl: k

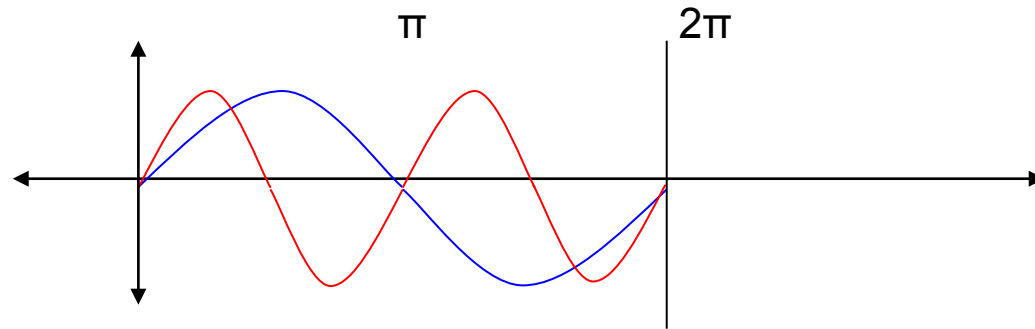
Periode: $2\pi/k$

Frequenz: $k/(2\pi)$

Grundfrequenz für $k=1$, $\cos(x)$:

Periode 2π , Frequenz $1/(2\pi)$

**(a_k, b_k) messen Anteil von f zur Frequenz $k/(2\pi)$,
(bzw. Periode $2\pi/k$, oder Wellenzahl k).**



Schwingung mit Wellenzahl **$k=1$** und **$k=2$**

Wellenzahl **$k=2$** entspricht Periode π , bzw. Frequenz $1/\pi$

$\cos(kx)$, $\sin(kx)$ bilden Orthonormalsystem (Basis)!

Fourierreihe entspricht Taylorreihe

Für periodische/auf Intervall definierte Funktion

Satz: Das trigonometrische Polynom

$$f_n(x) = \frac{a_0}{2} + \sum_{k=1}^n (a_k \cos(kx) + b_k \sin(kx))$$

stellt die optimale Approximation an die Funktion f dar aus dem Vektorraum der trigonometrischen Polynome vom Grad n :

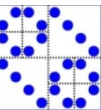
$$\|f_n(x) - f(x)\|_2^2 = \int_{-\pi}^{\pi} (f_n(x) - f(x))^2 dx \quad \text{ist minimal.}$$

f_n ist die Orthogonalprojektion von f bzgl. \int auf $[-\pi, \pi]$ auf den Raum der trigonometrischen Polynome vom Grad n

Näherungsweise Berechnung der Fourierkoeffizienten aus dem Integral mittels Trapezregel und äquidistanten Stützstellen

$$x_0 = -\pi, \quad x_j = -\pi + j \frac{2\pi}{n}, \quad x_n = \pi.$$

$$\begin{aligned} a_k &= \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(kx) dx \approx \\ &\approx \frac{1}{\pi} \cdot \frac{2\pi}{n} \cdot \left(\frac{1}{2} f(-\pi) \cos(-k\pi) + f(x_1) \cos(kx_1) + \dots \right. \\ &\quad \left. \dots + f(x_{n-1}) \cos(kx_{n-1}) + \frac{1}{2} f(\pi) \cos(k\pi) \right) = \\ &= \frac{2}{n} \sum_{j=0}^{n-1} f(x_j) \cos\left(k\left(-\pi + j \frac{2\pi}{n}\right)\right) = \\ &= \frac{2}{n} \cos(k\pi) \sum_{j=0}^{n-1} f(x_j) \cos\left(\frac{2\pi k j}{n}\right) + \frac{2}{n} \sin(k\pi) \sum_{j=0}^{n-1} f(x_j) \sin\left(\frac{2\pi k j}{n}\right) \end{aligned}$$



Daher ergeben sich die a_k näherungsweise aus den Real- und Imaginärteilen von

$$\sum_{j=0}^{n-1} f(x_j) \exp\left(\frac{2i\pi kj}{n}\right) = \sum_{j=0}^{n-1} f(x_j) \cos\left(\frac{2\pi kj}{n}\right) + i \cdot \sum_{j=0}^{n-1} f(x_j) \sin\left(\frac{2\pi kj}{n}\right)$$

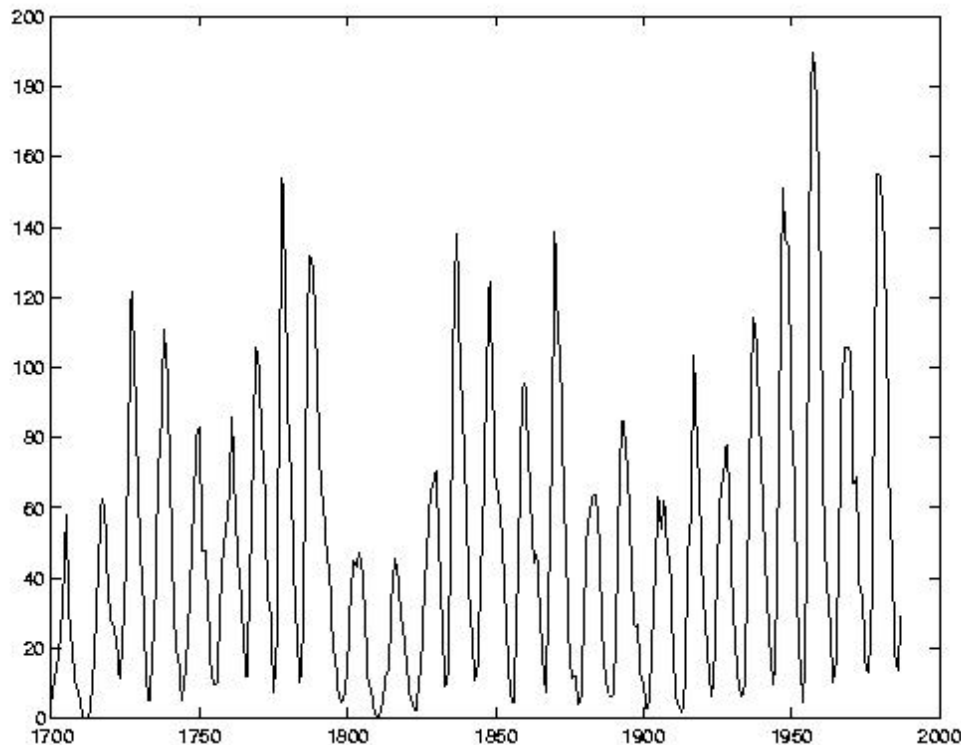
Diese Werte erhält man aus der IDFT angewendet auf die Funktionswerte an den Stützstellen.

Genauso lassen sich die b_k annähern.

Also können mittels DFT auf den Vektor der Funktionswerte die Frequenzanteile von f näherungsweise bestimmen werden.

DFT transformiert Funktionswerte in Koeffizienten

Sonnenflecken treten alle 11 Jahre verstärkt auf.
 Seit 1700 wird die jährliche Sonnenfleckenaktivität beschrieben durch die sog. Wolfer-Zahl (Größe und Anzahl der Flecken) (Rudolf Wolf ca. 1850).



**Wolferzahlen
für die Jahre
1700 bis 2000**

Wolferzahlen in Vektor v als Funktionswerte einer unbekannt periodischen Funktion g mit den Jahreszahlen als Stützstellen.

Gesamtbeobachtungszeitraum T besteht aus 300 Jahren.

Ersetze daher Intervall $[0, 2\pi]$ durch das Intervall $[0, T]$ durch den Übergang von

$$\exp(ikx) \rightarrow \exp(ikx \cdot 2\pi/300)$$

Zeitintervall $\Delta = 1$ Jahr, in dem eine Wolferzahl bestimmt wurde;

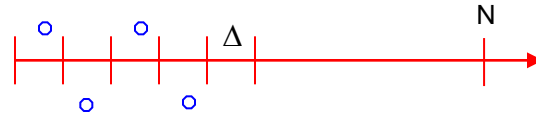
$N = T / \Delta$ ist Anzahl der Beobachtungsintervalle =

= Länge des Vektors v

= Anzahl der Stützstellen;

Wellenzahl wieder k , Periode $T/k = 300 \text{ J.}/k$, Frequenz k/T ;

Grundfrequenz also $1/T$.



Gibt es Schwingungen, die nicht erkannt werden können?

Es können nur periodische Vorgänge erfasst werden mit einer Periode $> 2\Delta = 2$ Jahre, da bei kleineren Perioden die ‚Schwingung‘ feiner als die feinste Unterteilung (1 Jahr) wäre, und daher als solche nicht erkennbar ist.

Daher ist die größte beobachtbare Wellenzahl k gleich $N/2=150$, entspricht der Frequenz $(N/2)(1/T)=1/(2\Delta)$, der sog. Nyquist-Frequenz

Berechne $y = \text{FFT}(v)$

$y(0)$ ist die Gesamtsumme aller Wolferzahlen und wird gleich 0 gesetzt (enthält keine Angabe über Periodizität).

Die anderen Koeffizienten von y enthalten die Größe der verschiedenen Frequenzanteile von g (aber als komplexe Zahlen).

Berechne $p(k) = |y(k)|^2$ für $k = 1, \dots, N/2$

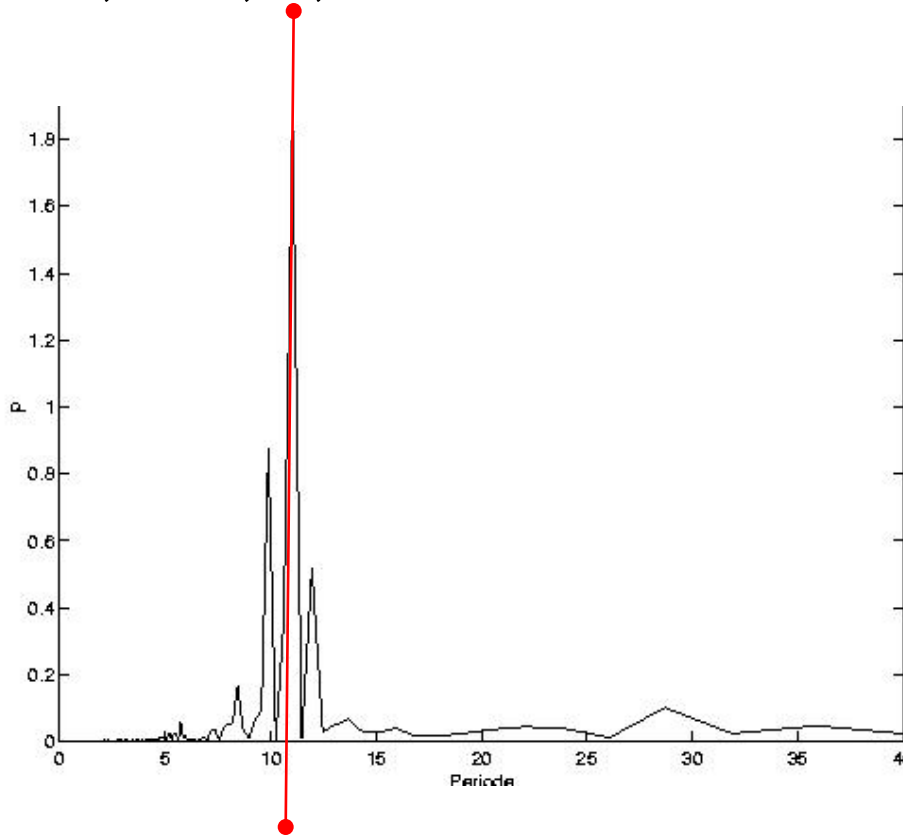
also untersuche nur die Beträge der Frequenzanteile bis zur Nyquist-Wellenzahl $k = (N/2)$.

$p(k)$ misst die Größe der k -ten Vielfachen der Grundfrequenz, also der Frequenz

$$f(k) = k / T = k / (\Delta N) \quad \text{für } k=1,2,\dots,N/2$$

Anders ausgedrückt bestimmt $p(k)$ das Gewicht der Periode $1 / f(k) = T / k$ in der Funktion f .

Wir tragen nun den Vektor p auf gegen die dazugehörigen Perioden $T/k, k=1, \dots, N/2$



Betragsquadrate der Fourierkoeffizienten, aufgetragen über die dazugehörigen Perioden.

Wir können den Zyklus (Periode) von 11 Jahren direkt ablesen.

Allgemeines Problem:

In welcher Form stellt man numerische Daten dar?

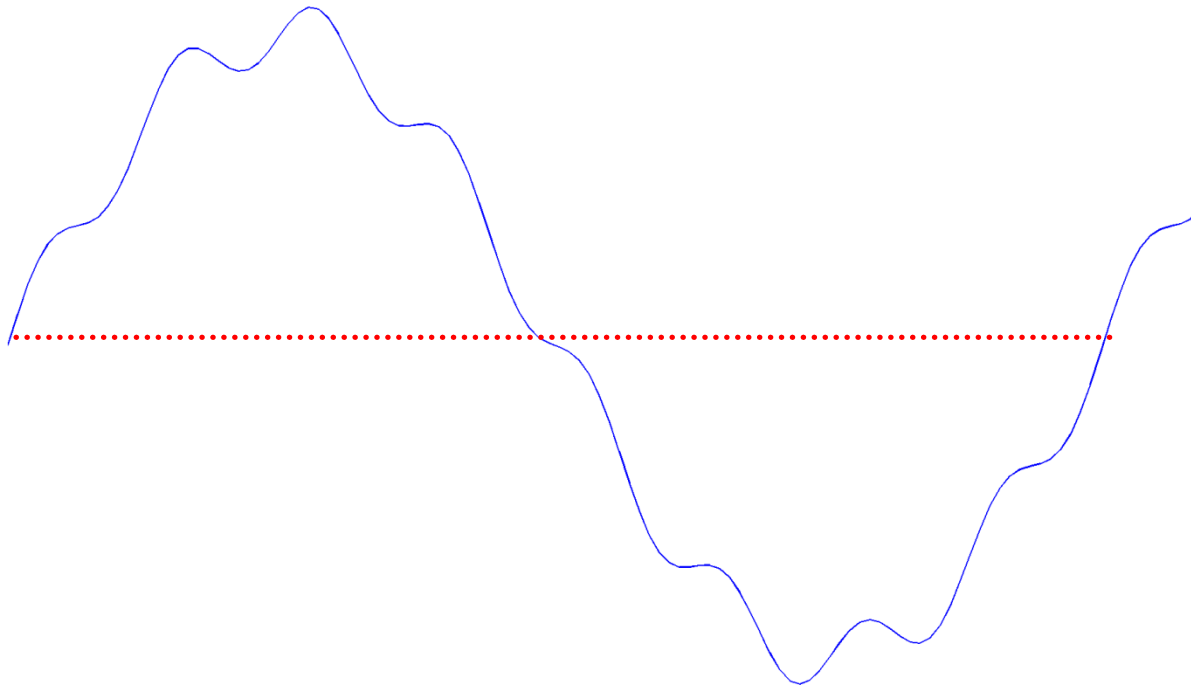
Normalerweise aus diskreten Messwerten: Samples $f(x_j)$
Dies entspricht einer Darstellung im Ortsraum.

Nachteil: Viele Daten, schlecht komprimierbar, keine Analyse!

Besser: Übergang zu einer Darstellung bzgl. geeigneter
Basisfunktionen (z.B. x^j , $\exp(jx)$, $\cos(jx)$,...)

Dann reichen ev. wenige Koeffizienten aus, um das Signal (fast) vollständig zu beschreiben. Außerdem kann man aus den Koeffizienten Schlüsse über das Verhalten der Funktion ziehen (Frequenzanalyse).

Beispiel: $f(x) = 0.5 \cdot \sin(x) + 0.1 \cdot \sin(10x)$



Zwei Koeffizienten reichen aus zur Beschreibung

Die Koeffizienten bezeichnet man auch als Beschreibung der Funktion im Phasenraum (Frequenzraum)

Bilder werden üblicherweise als Matrix gespeichert, deren Einträge pixelweise den Grauwert des entsprechend nummerierten Pixels enthalten, bzw. bei Farbbildern den Anteil einer der drei Farbkomponenten Rot, Grün oder Blau (= RGB).

RGB wird meist in der YUV-Form umgewandelt, bei der die Matrix Y die Helligkeit beschreibt, und U und V den Farbton (U, V lassen sich stärker komprimieren!).

JPEG zerlegt das Bild in 8 x 8 – Pixel große Blöcke, die zunächst getrennt bearbeitet werden. Auf jede dieser Teilmatrizen wird eine zweidimensionale Cosinus-Transformation angewandt: DCT

Warum DCT, warum nicht DFT?

Aus Anwendung der DCT erhält man für die Teilmatrizen wieder die Frequenzanteil-koeffizienten a_k und b_k in Teilmatrizen.

Die Teilmatrix mit den Frequenzanteilen wird nun quantisiert, d.h. die darin stehenden reellen Zahlen werden bestimmten Zahlenintervallen zugeordnet und Intervallweise jeweils durch eine Zahl angenähert, die für ein ganzes Intervall gilt. Maschinenzahlen!

Dadurch werden auch automatisch alle kleinen Komponenten, die in dem ersten Intervall mit den kleinsten Werten liegen, durch Null ersetzt.

Dies erzeugt einen Qualitätsverlust!

*Die entstandene Gesamtmatrix aus den diskreten Zahlenwerten wird codiert – **Huffman-Codierung***

Nachteile:

Fourier-Methoden haben Schwierigkeiten mit Kanten in Bildern
Kanten \leftrightarrow Unstetigkeiten

Durch stetige Funktionen $\cos(kx)$, $\sin(kx)$ sind Kanten schlecht darstellbar! Differenzierbarkeit!

Kosten der DCT: $O(n \log(n))$ bei n Pixel. Zu teuer.

Daher DCT auf 8×8 -Blöcke.

JPEG2000:

Anstatt Cosinus-Transformation auf 8×8 wendet man eine Wavelet-Transformation auf größere Teilmatrizen an.

Wavelet-Ansatzfunktionen haben keine Problem mit Kanten, vgl. B-Splines. Genauere Untersuchung folgt später.

Kosten für Wavelet-Transformation $O(n)$

Beispiel: MATLAB Gibbs-Phänomen (recht.m)

Fourierkoeffizienten für Rechtecksignal;
Überschwingen an Kante unvermeidbar bei Darstellung durch
endliches trigonometrisches Polynom!

5.3.4. Lineare Filter:

Betrachte Vektor v , dessen Komponenten wieder diskrete Werte einer Funktion oder eines Bildes darstellen (Sampling).

Zunächst 1-dimensional.

Wir *filtern* diesen Vektor, indem wir jede Komponente ersetzen durch eine gewichtete Kombination der Nachbarkomponenten.

So entspricht die *Maske*:

$$\frac{1}{4} [1 \quad 2 \quad 1] \quad \text{der Operation} \quad v_j = \frac{v_{j-1} + 2v_j + v_{j+1}}{4},$$

bzw. der Matrixmultiplikation

$$v \rightarrow \frac{1}{4} \begin{pmatrix} \blacksquare & 2 & 1 & & & \\ & 1 & 2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & 2 & 1 \\ & & & & 1 & 2 \\ & & & & & \blacksquare \end{pmatrix} \cdot v$$

Vorsicht am Rand! Fehlende Werte haben Einfluß auf Messwerte!
 Man benötigt Annahmen für v_0 und v_{n+1} . Was ist sinnvoll?

Im zwei-dim. entspricht dies z.B. der Maske

$$\frac{1}{8} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \rightarrow v_{i,j} = \frac{v_{i,j-1} + v_{i-1,j} + 4v_{i,j} + v_{i,j+1} + v_{i+1,j}}{8},$$

Genauso Gaussfilter:

$$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Dies entspricht glättenden Filtern, die zu einem weicherem Bild führen:

Mittelwertfilter, Glätter, Weichzeichner, oder Tiefpassfilter zur Abschwächung von Rauschen, bzw. hochfrequenten Anteilen. *Hochfrequente Störungen, die einzelne Komponente verändern, werden durch die Mittelwertbildung verringert!*

Entsprechend kann man Hochpassfilter definieren, die die Unterschiede hervorheben, das Bild härter machen und niederfrequente (glatte) Anteile abschwächen (Differenzfilter, Scharfzeichner).

z.B. Laplacefilter:
(Sobelfilter)

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Bisher entsprechen die Methoden einer *Filterung im Ortsraum*.
Im Gegensatz dazu kann man auch *Filter im Phasenraum*
(Frequenz~) zum Entfernen von Rauschen (Noise) verwenden:

$$v \rightarrow \text{DFT}(v) \rightarrow \text{Filter} \rightarrow \text{IDFT}(v)$$

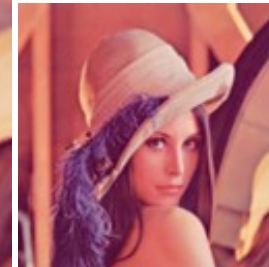
Also Transformation in (Fourier-)Koeffizientenraum,
dann Filtern der Koeffizienten = Gewichten/Löschen,
dann Rücktransformation (vgl. MP3)
Hochpassfilter, Tiefpass, Filterband,...

Filtern von Lena:

Original

Mittelwertfilter

Reduktion



Differenzfilter

Reduktion

Betrachte **Kombination** von Tiefpassfilter und Hochpassfilter im 1-Dimensionalen zu Masken

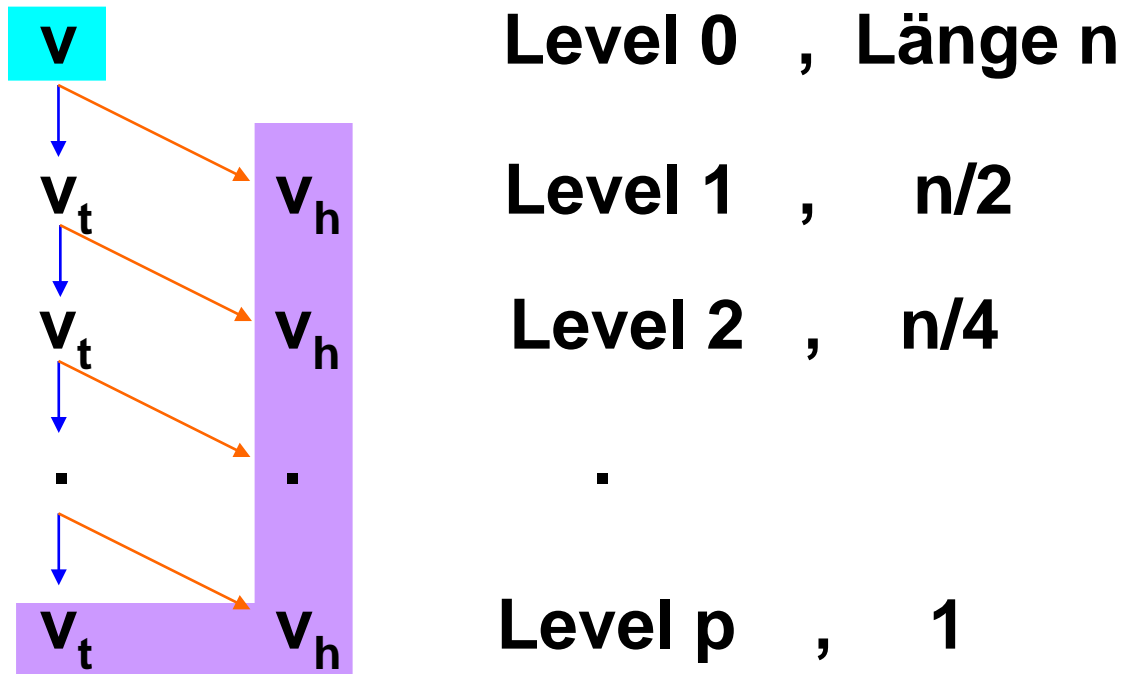
$$\begin{bmatrix} 1 & 1 \end{bmatrix} / 2 \quad \text{und} \quad \begin{bmatrix} 1 & -1 \end{bmatrix} / 2$$

Ersetze den ursprünglichen Vektor verlustfrei durch tief-, bzw. hochpass-gefilterte Vektoren halber Länge (down sampling):

$$\begin{pmatrix} v_t \\ v_h \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 & & & & \\ & & 1 & 1 & & \\ & & & & \ddots & \ddots \\ 1 & -1 & & & & \\ & & 1 & -1 & & \\ & & & & \ddots & \ddots \end{pmatrix} \cdot v$$

oder

Insgesamt:



Ersetze nun den Ausgangsvektor $v \in \mathbf{R}^n$ durch den letzten Mittelwertanteil auf Level p und sämtliche Differenzanteile v_h , das sind auch genau n Zahlen.

(v_h und v_t entsprechen quasi den Fourierkoeffizienten)

Vorteile:

Gesamtkosten der Transformation $O(n)$! Warum?
Differenz-Anteile meist klein \rightarrow gut komprimierbar

Differenzanteil auf Level k enthält Information über das Verhalten von v auf diesem Level, bzw. in dieser Auflösung
 \rightarrow

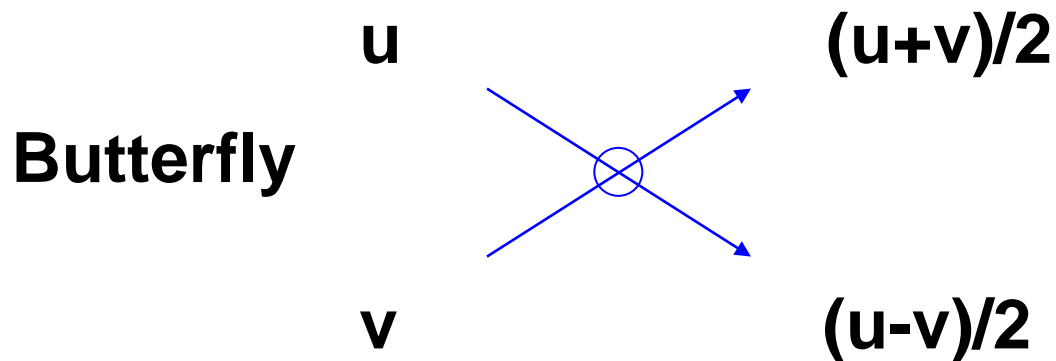
Multiskalenanalyse, Zerlegung des Vektors in verschiedene Frequenzbereiche = Skalen
Entspricht in etwa der Fourier-Analyse.

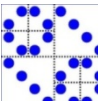
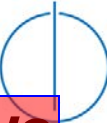
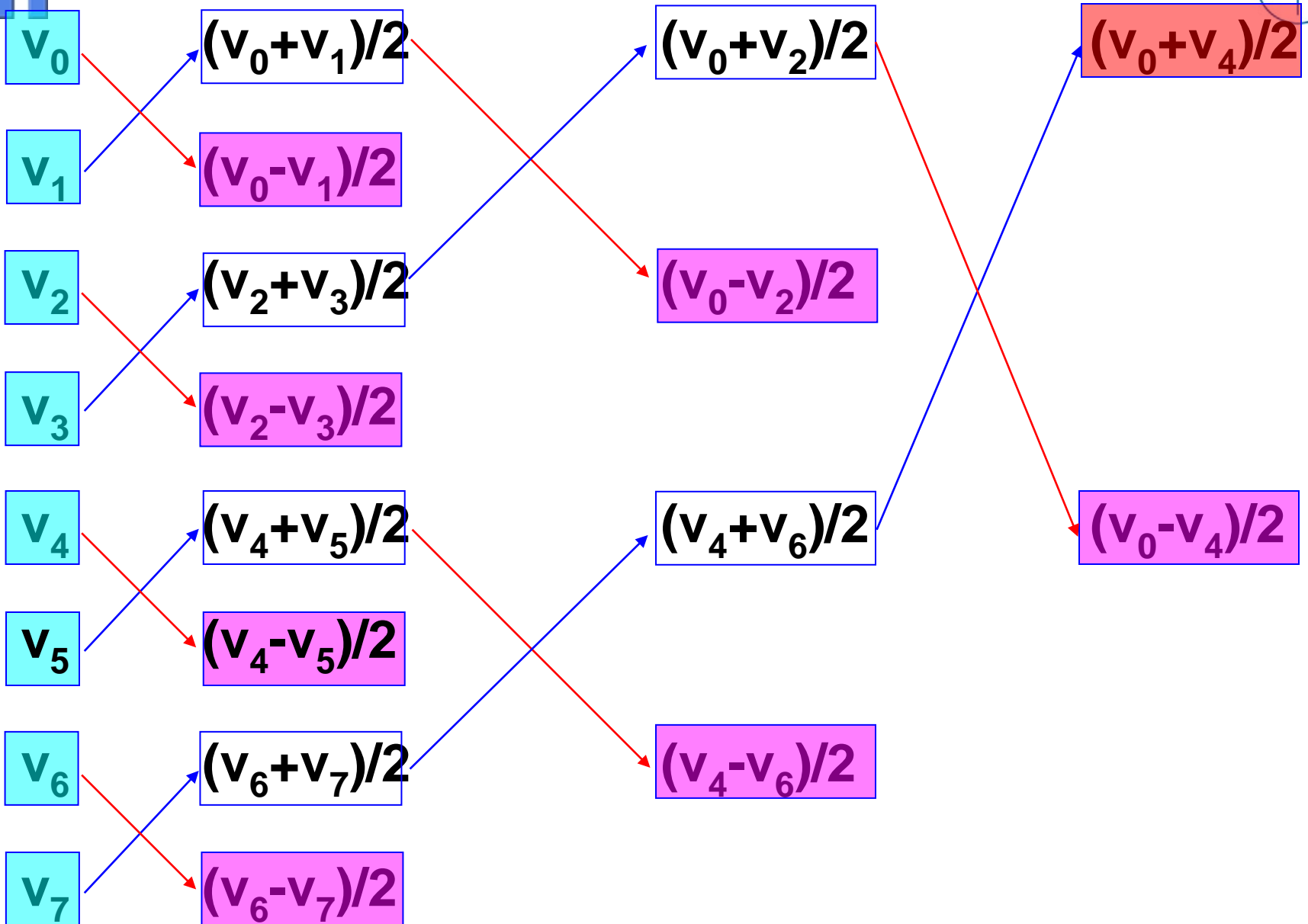
Filter müssen sparse sein damit billig,
leicht umkehrbar (invertierbar) damit Original rekonstruierbar!
(vgl. DFT und IDFT)

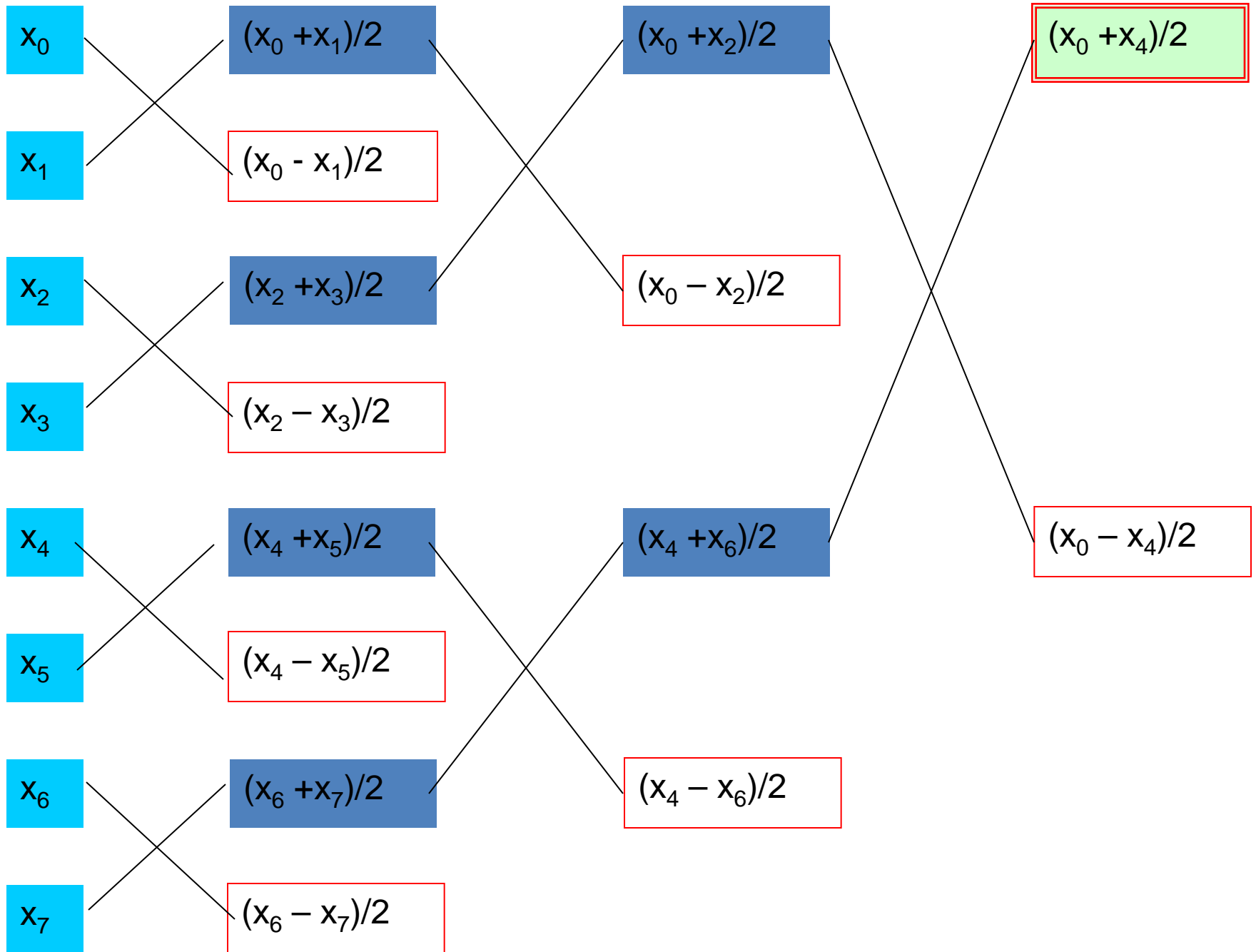
Aufspalten in Frequenzanteil (hoch – tief).

Algorithmische Ähnlichkeit zur Fourier-Transformation am Beispiel des Haar-Filters mit den Masken

$$\frac{1}{2} \begin{bmatrix} 1 & 1 \end{bmatrix} \quad \text{und} \quad \frac{1}{2} \begin{bmatrix} 1 & -1 \end{bmatrix} \quad :$$







v wird dabei transformiert in alle Differenzanteile und den letzten Mittelwert.

Unterschied zur DFT:

keine Sortierphase,

einfacherer Butterfly,

nur die Mittelwertanteile werden weiter bearbeitet

$\rightarrow O(n)$

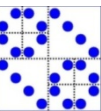
Bessere Filterkombination, z.B. Daubechey-Wavelets

$$\left[1 + \sqrt{3} \quad 3 + \sqrt{3} \quad 3 - \sqrt{3} \quad 1 - \sqrt{3} \right] / 4\sqrt{2}$$

Mittelwert-Filter

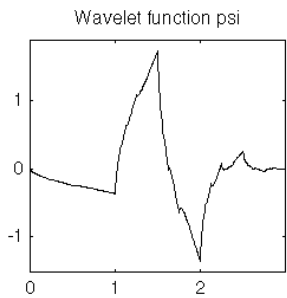
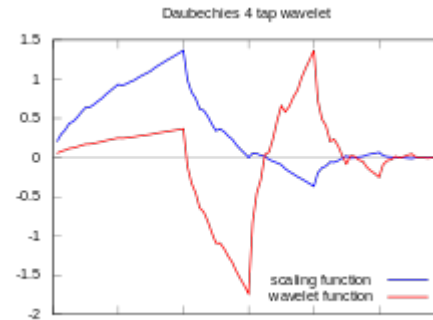
$$\left[1 - \sqrt{3} \quad -3 + \sqrt{3} \quad 3 + \sqrt{3} \quad -1 - \sqrt{3} \right] / 4\sqrt{2}$$

Differenz-filter

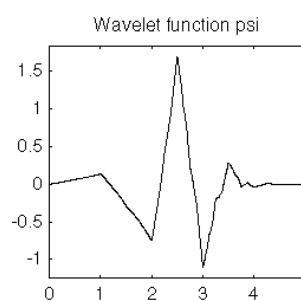




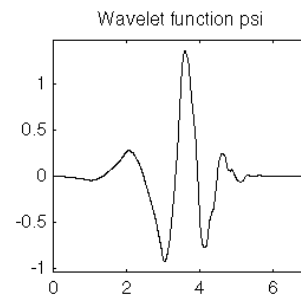
Daubechies Wavelets (orthogonal)



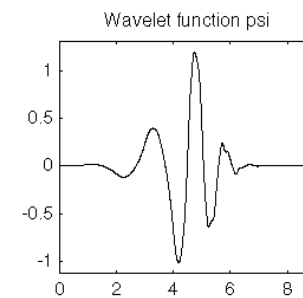
db2



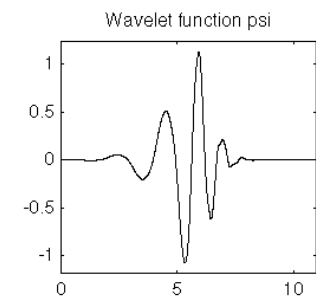
db3



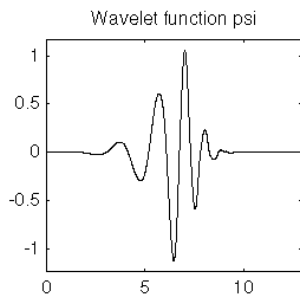
db4



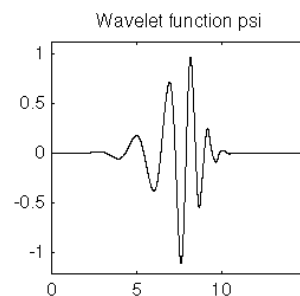
db5



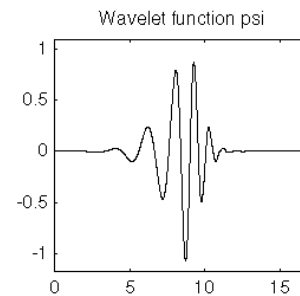
db6



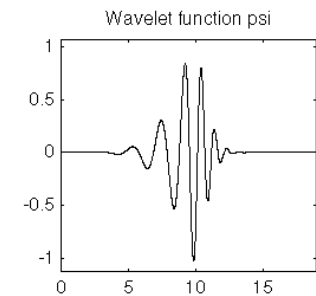
db7



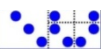
db8



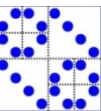
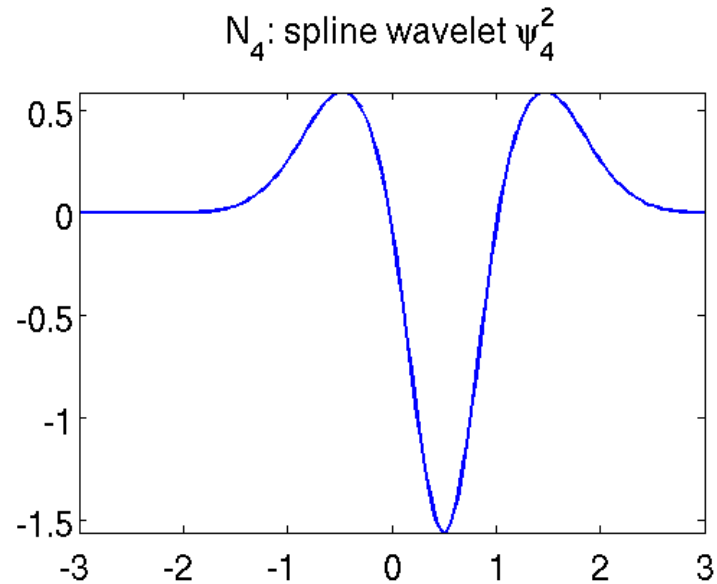
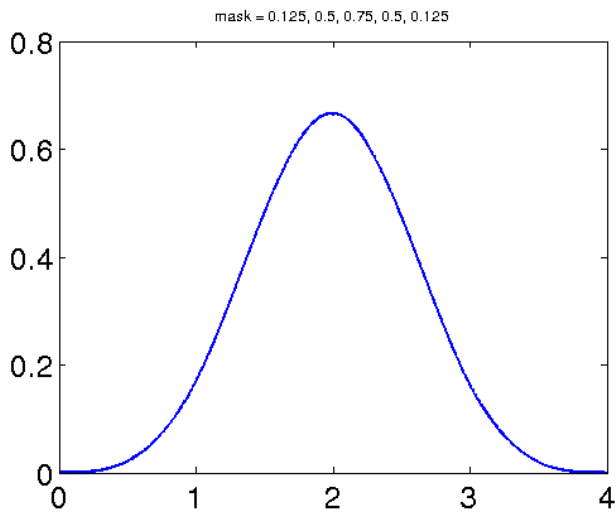
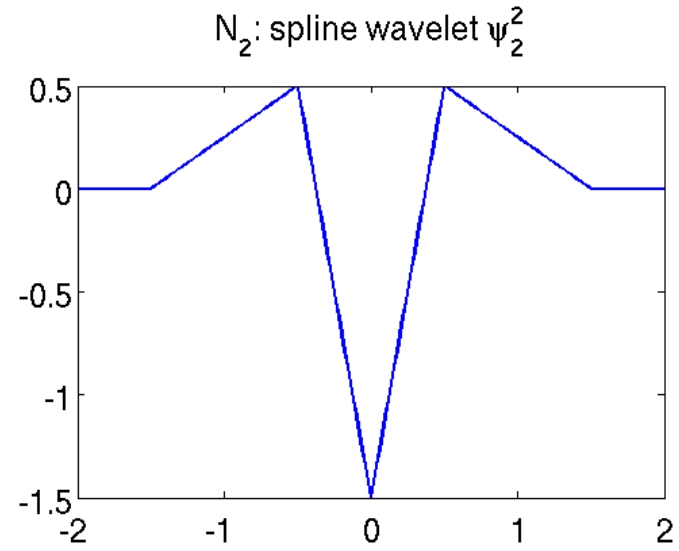
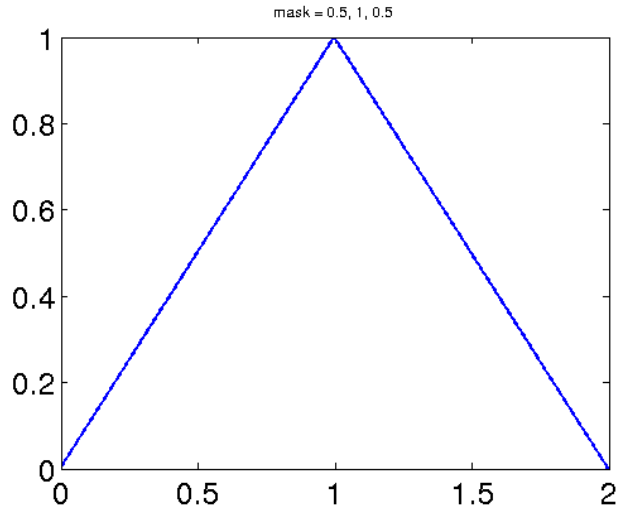
db9



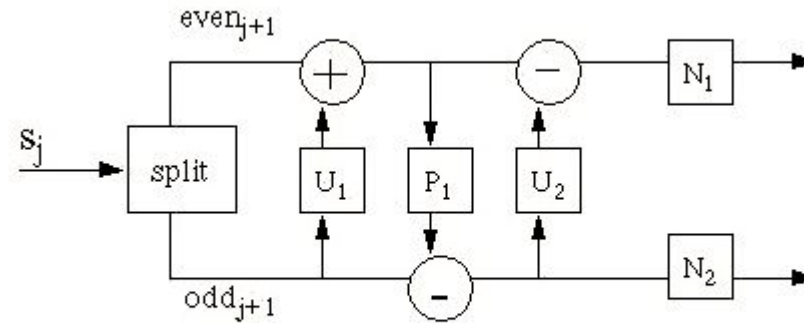
db10



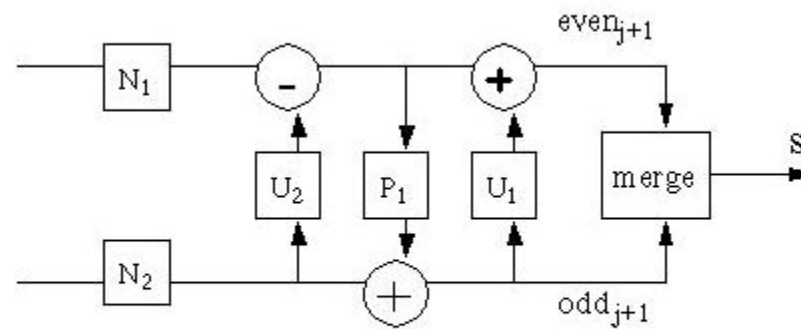
Spline Wavelets



Lifting Scheme



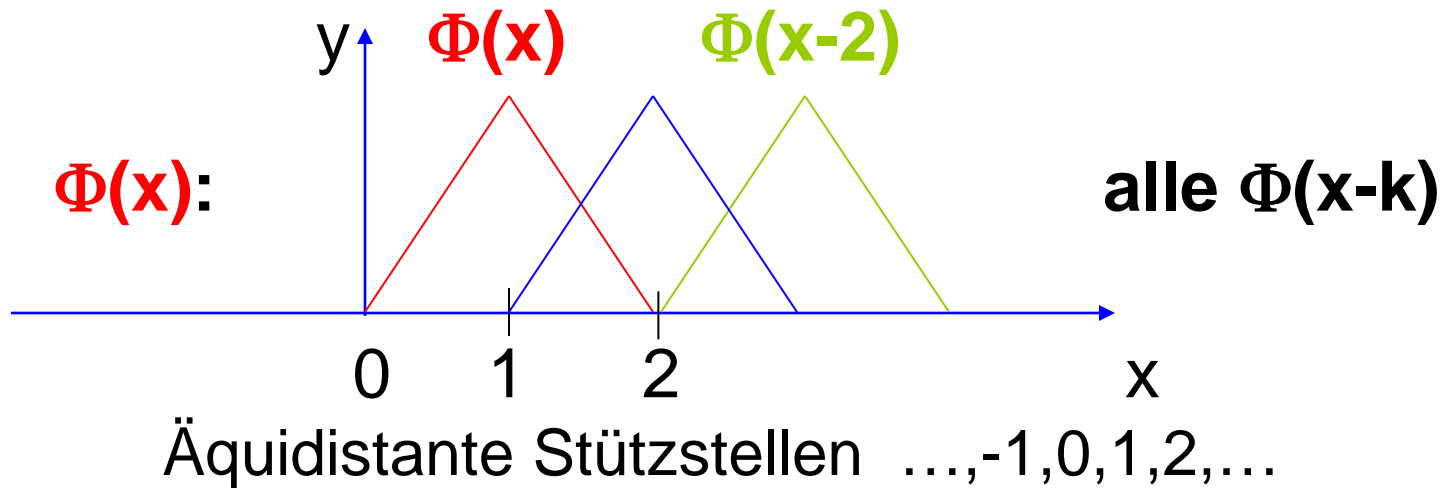
Daubechies D4 forward wavelet transform



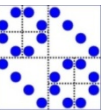
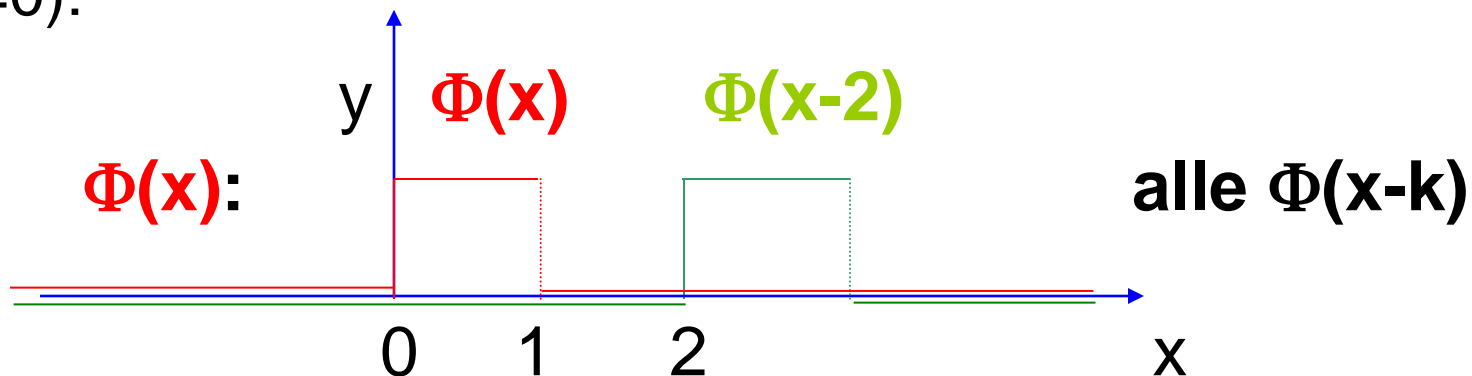
Daubechies D4 inverse wavelet transform



Ergänzung: Funktionsansatz: Erinnerung: lineare B-Splines (Hut-Funktion)



Zur Vereinfachung betrachten wir den konstanten B-Spline ($k=0$):

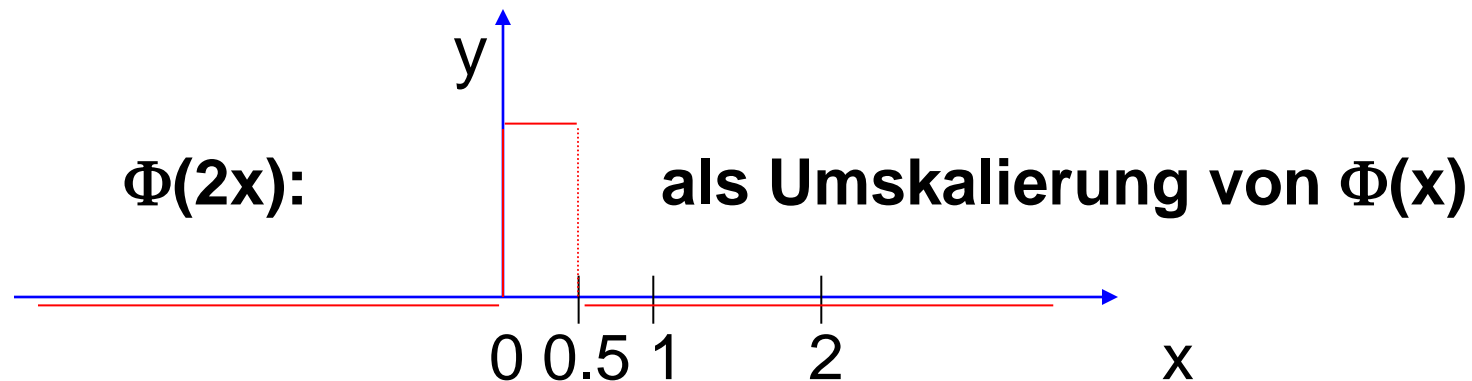




$\Phi(x)$ heißt Skalierungsfunktion und liefert Basis $\Phi(x-k)$, k
 Weiterhin erfüllt $\Phi(x)$ die Skalierungsgleichung

$$\Phi(x) = (1 \cdot \Phi(2x) + 1 \cdot \Phi(2x-1))$$

Stellt die Beziehung her zwischen fein/grob diskretisiert.



Genauso liefern $\Phi(2x-k)$ (feinere) Ansatzfunktionen

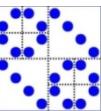
z.B. $1 = \Phi(0.25) =$

$$= (\Phi(2 \cdot 0.25) + \Phi(2 \cdot 0.25 - 1)) = \Phi(0.5) = 1$$

Filter-Maske $[1 \quad 1]$, denn

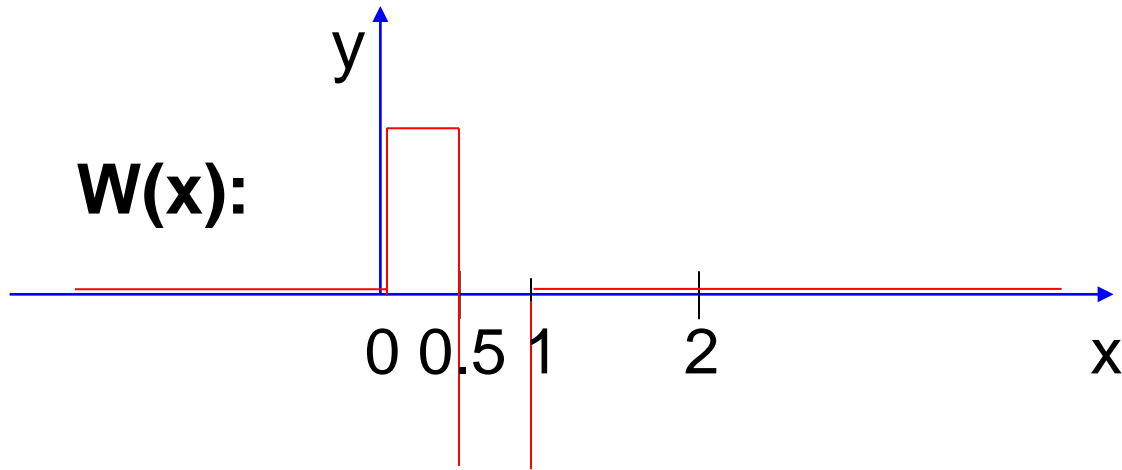
$$1 \cdot \Phi(2x) + 1 \cdot \Phi(2x-1)$$

zu Tiefpass-Filter (Mittelwertfilter)



Notwendig ist zweiter Hochpass-Filter (Differenz-Filter).
 Definiere dazu zur Skalierungsfunktion $\Phi(x)$ die eigentliche Waveletfunktion

$$W(x) = 1 * \Phi(2x) - 1 * \Phi(2x-1) , \quad \text{Maske } [1 \quad -1]$$



Wichtig: Orthogonalität der Ansatzfunktionen!

$$\int \Phi(x - k) \cdot W(x - j) dx = 0$$

Warum sind diese Funktionen orthogonal?

$\Phi(2x-k)$ sind Basis zu feinerer Diskretisierung und höherer Auflösung (Genauigkeit).

Mit den obigen Beziehungen kann $\Phi(2x)$ eindeutig durch $\Phi(x)$ und $W(x)$ dargestellt werden:

$$\Phi(x) = \Phi(2x) + \Phi(2x-1)$$

$$W(x) = \Phi(2x) - \Phi(2x-1)$$

oder umgekehrt

$$\Phi(2x) = (\Phi(x) + W(x)) / 2$$

$$\Phi(2x-1) = (\Phi(x) - W(x)) / 2$$

Gegeben: $f(x) = \sum a_k \cdot \Phi(2x-k)$

Nun können wir den Übergang von Koeffizienten zur Basis $\Phi(2x-k)$ zu Koeffizienten bezgl. $\Phi(x-k)$, $W(x-k)$ beschreiben

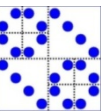
$$a_{2k} \cdot \Phi(2x-2k) \rightarrow a_{2k} \cdot (\Phi(x-k) + W(x-k)) / 2$$

$$a_{2k+1} \cdot \Phi(2x-2k-1) \rightarrow a_{2k+1} \cdot (\Phi(x-k) - W(x-k)) / 2$$



gerade/ungerade

$$\begin{aligned} f(x) &= \sum a_k^{(1)} \Phi(2x - k) = \\ &= \sum a_{2k}^{(1)} \Phi(2x - 2k) + \sum a_{2k+1}^{(1)} \Phi(2x - 2k - 1) = \\ &= \sum \frac{a_{2k}^{(1)}}{2} (\Phi(x - k) + W(x - k)) + \\ &\quad + \sum \frac{a_{2k+1}^{(1)}}{2} (\Phi(x - k) - W(x - k)) = \\ &= \sum \frac{a_{2k}^{(1)} + a_{2k+1}^{(1)}}{2} \cdot \Phi(x - k) + \sum \frac{a_{2k}^{(1)} - a_{2k+1}^{(1)}}{2} \cdot W(x - k) = \\ &= \sum a_k^{(0)} \cdot \Phi(x - k) + \sum b_k^{(0)} \cdot W(x - k) = \\ &= f^{\text{tiefpass}}(x) + f^{\text{hochpass}}(x) \end{aligned}$$

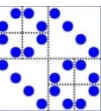




Anwendung der beiden Filtermasken $[1 \ 1]/2$ und $[1 \ -1]/2$ auf die Koeffizienten a_k liefert neue Koeffizienten zu hoch/tiefpass-gefilterten Teil-Funktionen.

Umgekehrt kann aus den beiden gefilterten Teilfunktionen die Gesamtfunktion $f(x)$ wieder rekonstruiert werden mittels

$$\begin{aligned} f^{\text{tiefpass}}(x) + f^{\text{hochpass}}(x) &= \\ \sum a_k^{(0)} \Phi(x-k) + b_k^{(0)} W(x-k) &= \\ = \sum a_k^{(0)} (\Phi(2x-2k) + \Phi(2x-2k-1)) + \\ &\quad + \sum b_k^{(0)} (\Phi(2x-2k) - \Phi(2x-2k-1)) = \\ = \sum (a_k^{(0)} + b_k^{(0)}) \cdot \Phi(2x-2k) + \\ &\quad + \sum (a_k^{(0)} - b_k^{(0)}) \Phi(2x-2k-1) = \\ = \sum a_{2k}^{(1)} \Phi(2x-2k) + \sum a_{2k+1}^{(1)} \Phi(2x-2k-1) = \\ = \sum a_k^{(1)} \Phi(2x-k) = f(x) \end{aligned}$$



Also Anwendung der Filter

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \text{ und } \begin{bmatrix} 1 & -1 \end{bmatrix}$$

auf die Koeffizienten der hoch/tiefpassgefilterten Koeffizienten liefert die Koeffizienten zur feineren Diskretisierung.

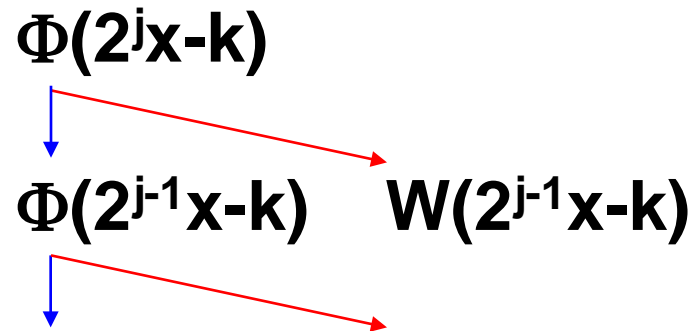
Funktionenfamilien auf verschiedenen Skalen:

$$\Phi(x-k), \quad k = \dots, -1, 0, 1, \dots$$

$$\Phi(2x-k), \quad k = \dots, -1, 0, 1, \dots$$

....

$$\Phi(2^j x - k), \quad k = \dots, -1, 0, 1, \dots, \quad j = \dots, -1, 0, 1, \dots$$



beschrieben durch die Transformation der Koeffizienten:

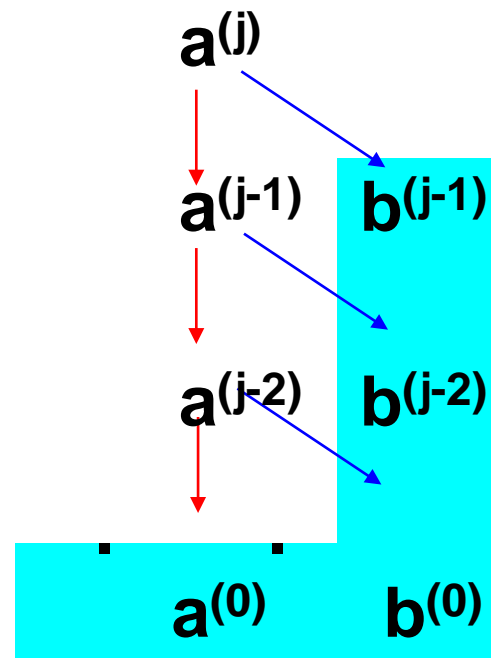
$$\begin{pmatrix} \vdots \\ \vdots \\ a_k^{(j-1)} \\ b_k^{(j-1)} \\ \vdots \\ \vdots \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 & & & & \\ 1 & -1 & & & & \\ & & 1 & 1 & & \\ & & 1 & -1 & & \\ & & & & \ddots & \ddots \\ & & & & \ddots & \ddots \end{pmatrix} \cdot a^{(j)}$$

entsprechend Tief- und Hochpassfilter mit Haar-Filter.

Umkehrung mit

$$a^{(j)} = \begin{pmatrix} 1 & 1 & & & & \\ 1 & -1 & & & & \\ & & 1 & 1 & & \\ & & 1 & -1 & & \\ & & & & \ddots & \ddots \\ & & & & \ddots & \ddots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ \vdots \\ a_k^{(j-1)} \\ b_k^{(j-1)} \\ \vdots \\ \vdots \end{pmatrix}$$

Rekursive Wiederholung:



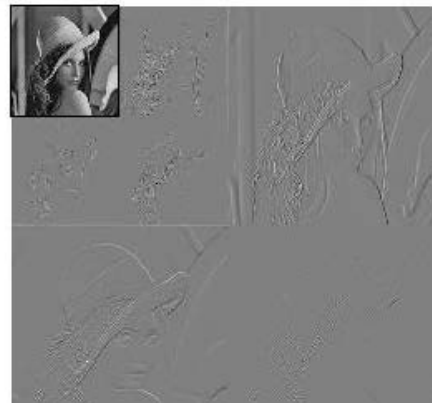
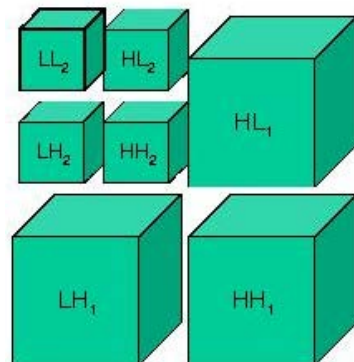
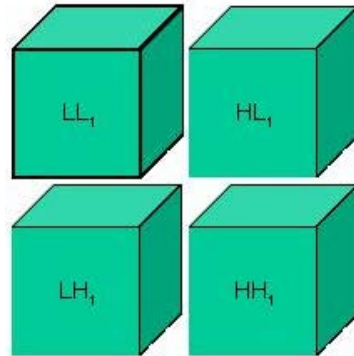
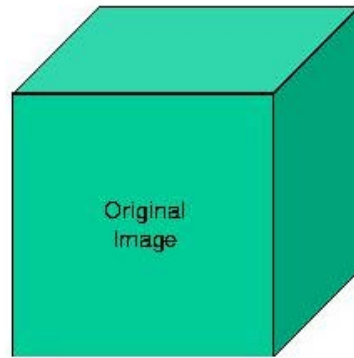
Wavelet als Verschmelzung der
klassischen Filter und
Fourieranalyse

mit der Idee

lokaler Basisfunktionen wie bei den B-Splines.

Haar-Filter \leftrightarrow Haar-Wavelet

Orthogonale Wavelet, biorthogonale Wavelets, Frames,...





Fourier-Analyse

FFT

$\cos(kx), \sin(kx)$

Skalierung k

Alle gleichberechtigt

globaler Träger



keine Variations-
möglichkeit

Kantenproblem

Volle Rekursion

Kosten $O(n \log(n))$

Frequenz aus

Fourierkoeffizienten



Multiskalen-Analyse

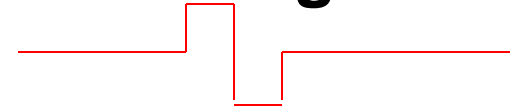
Wavelet

$W(2^jx-m), \Phi(2^jx-m)$

Shift m , Skalierung 2^j

Mittelwert/Differenz

lokaler Träger



Wahl von Approximations-
güte und Diff'barkeit

Lokal, adaptiv

Rekursion nur im Mittelwert

Kosten $O(n)$

Frequenz aus

Waveletkoeffizienten

