Universität Duisburg-Essen, Standort Duisburg

BLaC-Wavelets und nicht ineinander geschachtelte Wavelets

Diplomarbeit

Oliver Nemitz

Betreuer: Prof. Dr. Heinz H. Gonska

Duisburg, im Oktober 2003

Zusammenfassung

Eine Multiskalenanalyse erlaubt die Repräsentation einer Funktion, bzw. gegebenen Daten, in verschiedenen Detailstufen (Leveln). Klassischerweise basieren diese unterschiedlichen Level auf einer Folge ineinander geschachtelter Funktionenräume. Mit zunehmendem Level konvergieren Approximationen in diesen Funktionenräumen gegen die Ausgangsdaten.
Im Fall der Haar-Wavelets sind dies z.B. Treppenfunkionen mit verschieden breiten Trägern. Je höher der Level ist, desto feiner werden die „Stufen“, und desto genauer wird die Approximation.
G.P. Bonneau entwirft nun in seinem Artikel *BLaC-wavelets and non-nested wavelets* den theoretischen Hintergrund für eine Multiskalenanalyse mit nicht ineinander geschachtelten Räumen. Er beschreibt zwei Anwendungen:

BLaC-Wavelets sind eine von einem Parameter abhängige Familie von Wavelets, mit denen es möglich ist eine stetige Vermischung des Haar-Wavelets und des linearen Wavelets zu definieren (BLaC = Blending of Linear and Constant). Die Skalierungsfunktion der Haar-Wavelets bietet als Treppenfunktion eine perfekte Lokalität, ist aber nur sehr schlecht geeignet, stetige Daten zu approximieren. Eine bessere Regularität hierfür bietet die Hütchenfunktion, die Skalierungsfunktion des linearen Wavelets. Diese wiederum besitzt eine nicht so gute Lokalität und ist zum Darstellen unstetiger Daten weniger geeignet. Mit den BLaC-Wavelets ist es dem Anwender nun möglich, einen für ihn optimalen Kompromiss zwischen der perfekten Lokalität der Haar-Wavelets und der besseren Regularität der linearen Wavelets zu wählen.

Als zweite Anwendung der Multiskalenanalyse mit nicht ineinander geschachtelten Räumen sind hierarchische Triangulierungen aufgeführt. Hierbei wird eine gegebene Triangulierung in mehreren Schritten vereinfacht, d.h. die Anzahl der verwendeten Dreiecke wird reduziert. Hierfür existieren bereits eine Reihe Wavelet-basierter Verfahren, diese sind jedoch aufgrund der ineinander geschachtelten Räume immer mit Subdivision verbunden. Bonneau's Entwicklung erlaubt, sich nun davon zu lösen. Dies wird im letzten Teil der Arbeit geschildert.

Im ersten grösseren Teil der Arbeit wird das Konzept dieser neuen Multiskalenanalyse theoretisch eingeführt und erläutert. Anschliessend werden die BLaC-Wavelets definiert, wobei Bonneau's Ausführungen, z.B. durch eine Fehleranalyse, erheblich ergänzt werden. Verschiedene Eigenschaften der Skalierungsfunktionen werden aufgestellt und bewiesen. Anschliessend wird ein hauptsächlich der Anschauung dienender Implementierungsansatz ausführlich beschrieben und verwirklicht.

Die Theorie wird daraufhin noch auf zwei Dimensionen erweitert. Als Anwendung wird hier ein einfaches Bildkompressionsverfahren vorgestellt und implementiert. In Zusammenhang damit wird auch eine Methode zur Wahl des Parameters für ein gegebenes Bild entwickelt.

Inhaltsverzeichnis

1 Multiskalenanalyse

Mit einer Multiskalenanalyse kann eine Funktion in verschiedenen Detailstufen (Leveln) repräsentiert werden. Die Informationen, die von einem Level zum nächsten fehlen, werden in den sogenannten Wavelet- bzw. Detailkoeffizienten gespeicbert. Die Funktion kann somit in einen groben Anteil und die fehlenden Detailstufen zerlegt werden.

Zu Beginn meiner Arbeit möchte ich vorerst einige einleitende Definitionen und Sätze zitieren und dann erst genauer auf das Konzept der herkömmlichen Multiskalenanalyse eingehen (im Folgenden mit MSA abgekürzt, englisch: Multiresolution-Analysis, kurz MRA).

1.1 Hilberträume

Definition 1 *Sei H ein Vektorraum. Eine Funktion $< \cdot, \cdot >: H \times H \longrightarrow \mathbb{C}$ heisst **Skalarprodukt**, wenn gilt:*

$$
\begin{aligned}
& f \longmapsto < f, g > \text{ linear}, \\
& g \longmapsto < f, g > \text{ antilinear} \quad (< f, \lambda g >= \overline{\lambda} < f, g >), \\
& < f, g >= \overline{< g, f >}, \\
& < f, f >> 0 \quad \forall f \neq 0.
\end{aligned}
$$

*Existiert eine solche Funktion, so ist H ein **Prähilbertraum**.*

Lemma 1 $||f|| = \sqrt{< f, f >}$ *ist eine **Norm**, d.h. es gilt*

$$
\begin{aligned}
& ||f + g|| \leq ||f|| + ||g|| \quad \forall f, g \in H, \\
& ||\lambda f|| = |\lambda| ||f|| \quad \forall f \in H, \quad \forall \lambda \in \mathbb{C} \\
& ||f|| \geq 0, \ (||f|| = 0 \Longleftrightarrow f = 0) \qquad \forall f \in H.
\end{aligned}
$$

Der Beweis ist in der gängigen Literatur (z.B. in [6]) zu finden.

Definition 2 *Ein **Hilbertraum** ist ein Prähilbertraum derart, dass $(H, || \cdot ||)$ vollständig ist, d.h. jede Cauchy-Folge bzgl. $|| \cdot ||$ konvergiert.*

Beispiele:

1) $L^2[a,b] = \left\{ f : [a,b] \mapsto \mathbb{R} \,\middle|\, \int_a^b |f(t)|^2 dt < \infty \right\}$

mit Skalarprodukt $< f, g >= \int_a^b f(t)\overline{g(t)}dt$ und

2) $L^2(\mathbb{R}) = \left\{ f : \mathbb{R} \mapsto \mathbb{R} \,\middle|\, \int_{-\infty}^{\infty} |f(t)|^2 dt < \infty \right\}$

mit Skalarprodukt $< f, g >= \int_{\mathbb{R}} f(t)\overline{g(t)}dt$ sind jeweils Hilberträume.

Definition 3 *f und g sind **orthogonal**, kurz $f \perp g$, falls $< f, g >= 0$.*

Definition 4 *Eine Basis $\{e_i, i \in I\}$, I eine Indexmenge, eines Hilbertraums H, heisst **Orthonormalbasis**, kurz **ONB**, falls für alle $i, j \in I$ gilt:*

1) $||e_i|| = 1$

2) $< e_i, e_j >= \delta_{i,j} = \begin{cases} 1 & : \quad i = j \\ 0 & : \quad sonst \end{cases}$.

Für eine Multiskalenanalyse werden noch folgende zwei Abbildungen eingeführt:

1. *Dilatation*: $x \mapsto \psi(ax), \quad a \in \mathbb{R}$. Beispiel:

$$\psi(x) = \begin{cases} 1 & : \quad x \in (0, \frac{1}{2}] \\ -1 & : \quad x \in (\frac{1}{2}, 1] \\ 0 & : \quad sonst \end{cases}$$

$$\text{Dann ist } \psi(2x) = \begin{cases} 1 & : \quad x \in (0, \frac{1}{4}] \\ -1 & : \quad x \in (\frac{1}{4}, 1] \\ 0 & : \quad sonst \end{cases}$$

   Der Träger der Funktion wird verkleinert.

2. *Translation*: $x \mapsto \psi(x - b), \quad b \in \mathbb{R}$.
   Die Translation bewirkt eine Verschiebung des Trägers.

1.2 Herkömmliche Multiskalenanalyse

Definition 5 *Eine **(schwache) MSA** besteht aus einer Folge von Teilräumen $V_n \subset L^2(\mathbb{R})$, $n \in \mathbb{Z}$, mit folgenden Eigenschaften:*

$$V_n \subset V_{n+1} \tag{1}$$

$$\bigcup_n V_n \text{ ist dicht in } L^2(\mathbb{R}) \tag{2}$$

$$\bigcap_n V_n = 0 \tag{3}$$

Gelten zusätzlich noch die Bedingungen

$$\exists \varphi \in L^2(\mathbb{R}), \text{ so dass } \varphi(x - k), k \in \mathbb{Z}, \text{ ONB von } V_0 \text{ ist, und} \tag{4}$$

$$V_n = \{f(2^n x) | f \in V_0\} \tag{5}$$

*so spricht man von einer **MSA**.*

Bedingung (1) besagt, dass die Räume V_n mit steigendem n wachsen. Haben alle Räume endliche Dimensionen, so lässt sich diese Bedingung auch als $\dim V_n \leq \dim V_{n+1}$ schreiben. Die zweite Bedingung sichert, dass jede Funktion aus $L^2(\mathbb{R})$ mit beliebiger Genauigkeit approximiert werden kann.

φ ist die sogenannte *Skalierungsfunktion.* Aus dieser lassen sich durch Dilatationen und Translationen Basen der Räume V_n bilden. Es gilt:

$$\{\varphi_n^k := 2^{\frac{n}{2}}\varphi(2^n x - k), \quad k \in \mathbb{Z}\}, \quad \text{ist ONB von } V_n. \tag{6}$$

Beweis siehe z.B. [12].

Die Forderung, dass die ganzzahligen Translate von φ eine ONB bilden sollen, ist eine sehr strenge Bedingung, die nicht immer erfüllt ist. Es ist allerdings möglich diese Bedingung abzuschwächen und durch

$\exists \varphi \in L^2(\mathbb{R})$, so dass $\varphi(x-k), k \in \mathbb{Z},$ eine ***Riesz-Basis*** von V_0 ist,
d.h. es existieren positive Konstanten A, B so, dass für alle Folgen $(c_k)_{k\in\mathbb{Z}} \in l^2$ gilt:
$A\sum_{k\in\mathbb{Z}} |c_k|^2 \leq ||\sum_{k\in\mathbb{Z}} c_k\varphi(\cdot - k)||_{L^2}^2 \leq B\sum_{k\in\mathbb{Z}} |c_k|^2,$

zu ersetzen. Auch mit einer solchen Basis kann eine MSA konstruiert werden.

Desweiteren existieren wegen $V_n \subset V_{n+1}$ Koeffizienten $p_{i,j} \in \mathbb{R}$, so dass sich die φ_j^n als Linearkombination der φ_j^{n+1} in der sog. *Verfeinerungsgleichung* darstellen lassen:

$$\varphi_j^n = \sum_i p_{i,j}^n \varphi_i^{n+1} \qquad , p_{i,j} \in \mathbb{R} \qquad \text{(Refinement Equation)} \tag{7}$$

Damit lässt sich nun der ***Waveletraum*** W_n als orthogonales Produkt von V_n in V_{n+1} definieren:

$$V_n \oplus W_n = V_{n+1} \Leftrightarrow W_n = V_{n+1} \ominus V_n$$

(Bemerkung: Das intuitive Zeichen „ $\ominus$ “ ist durch genau diesen Zusammenhang definiert, d.h.: $A \oplus B = C \Leftrightarrow B =: C \ominus A$).

Es gilt der folgende

Satz 1 *Sei H ein Hilbertraum und $(V_j)_{j\in\mathbb{Z}}$ eine schwache Multiresolution mit $V_j \subset H, j \in \mathbb{Z}$. Dann gilt:* $H = \bigotimes_{j=-\infty}^{\infty} W_j$

Beweis siehe z.B. [12].

Basen dieser Waveleträume sind die sog. *Wavelets* ψ_j^n . Auch diese lassen sich als Linearkombination der φ_j^n darstellen:

$$\psi_j^n = \sum_i q_{i,j}^n \varphi_i^{n+1} \qquad , q_{i,j} \in \mathbb{R}$$

Ferner existiert auch für die Wavelets eine Funktion ψ, so dass sich die ψ_j^n durch Dilatation und Translation von ψ erzeugen lassen:

$$\psi_j^n := 2^{\frac{n}{2}} \varphi(2^n x - j), \quad j, n \in \mathbb{Z} \tag{8}$$

Wegen obigem Satz bilden also die $\psi_j^n, \quad j, n \in \mathbb{Z}$, eine ONB von H.

1.3 Anwendung der Multiskalenanalyse

Wie schon zu Anfang gesagt, ist es mit einer Multiskalenanalyse möglich, gegebene Daten in verschiedenen Detailstufen zu repräsentieren. Die zwischen zwei aufeinanderfolgenden Leveln fehlenden Details werden in den sogenannten Waveletkoeffizienten gespeichert.

In der Praxis, z.B. bei Computeranwendungen, ist eine MSA ein mächtiges Werkzeug zur Behandlung grosser Datenmengen (z.B. Bilder). Da die Daten in diesem Fall jedoch immer begrenzt sind, genügt es, sich auf endlich viele Räume V_n zu beschränken. Üblicherweise werden diese mit $V_0, ..., V_N$ bezeichnet.
Liegt nun eine Funktion $f \in L^2(\mathbb{R})$ vor, so hat man oft eine feinste Approximation f_N von f in V_N gegeben (z.B. ein Bild, Messwerte etc.). Der unten stehende Index n bzw. N wird als Level bezeichnet. Je höher der Level ist, desto feiner ist die Approximation. f_N wird nun sukzessive in einen „groben“, Überblick bietenden Anteil und die fehlenden Details zerlegt:

$$\begin{array}{ccccccccc} f_N & \longrightarrow & f_{N-1} & \longrightarrow & f_{N-2} & \longrightarrow & \cdots & \longrightarrow & f_0 \\ & \searrow & & \searrow & & \searrow & & \searrow & \\ & & g_{N-1} & & g_{N-2} & & \cdots & & g_0 \end{array} \qquad (Analyse)$$

Hierbei ist $f_n \in V_n$ und $g_n \in W_n$. Man erhält als Transformation der Funktion f einen „gröbsten“ Anteil f_0 und verschiedene Detailstufen $g_0, g_1, ..., g_{N-1}$.

Eine mögliche Anwendung wäre nun eine Kompression, die dadurch erreicht wird, dass zu kleine Waveletkoeffizienten, d.h. diejenigen, die eine gegebene Schranke unterschreiten, entfernt werden. Natürlich ist dieses Verfahren verlustbehaftet, die resultierende Darstellung dem Benutzer aber womöglich immer noch detailliert genug.
Beim Aufbau einer Funktion, z.B. eines Bildes, ist es möglich, dieses schrittweise von einer groben Darstellung aus immer detaillierter anzuzeigen. Hierzu werden von einem gegebenen Level aus nur die Details dazu addiert.
Ebenso ist es möglich, an den Koeffizienten eines niedrigen Levels Manipulationen vorzunehmen, dann die Synthese wieder durchzuführen, um somit Veränderungen an „grösseren“ Bereichen der Funktion zu erreichen.

Im Folgenden möchte ich zwei Skalierungsfunktionen und die dazu gehörigen Wavelets vorstellen. Zum einen das wohl einfachste und bekannteste Wavelet, das Haarwavelet, und zum anderen das auf den B-Splines basierende sog. lineare Wavelet.

1.4 Das Haar-Wavelet

Die Skalierungsfunktion des Haar-Wavelets ist gegeben durch die 1-Funktion auf dem Intervall $[0, 1)$:

$$\varphi(x) = \chi_{[0,1)}(x) = \begin{cases} 1 & : \quad x \in [0,1) \\ 0 & : \quad sonst \end{cases}$$

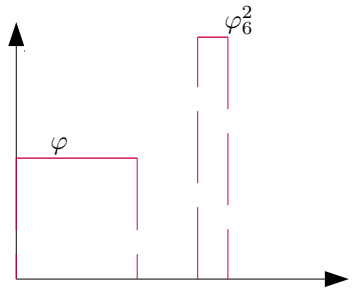


Abbildung 1: Skalierungsfunktion φ und φ_6^2

Nun können durch Dilatation und Translation Basen der Räume V_n definiert werden. Es ist

$$\varphi_k^n(x) := 2^{\frac{n}{2}}\varphi(2^n - k) = \begin{cases} 2^{\frac{n}{2}} & : \quad x \in [2^{-\frac{n}{2}}k, 2^{-\frac{n}{2}}(k+1)) \\ 0 & : \quad sonst \end{cases} \quad , k \in \mathbb{Z} \text{ eine ONB von } V_n$$

Linearkombinationen dieser Basisfunktionen sind Treppenfunktionen. Der Raum V_n enthält somit alle Treppenfunktionen mit „Stufenbreite“ $2^{-\frac{n}{2}}$. Je höher der Level n ist, desto genauer kann eine gegebene Funktion durch Funktionen aus V_n approximiert werden.

φ erfüllt zudem folgende Verfeinerungsgleichung:

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

Das zugehörige Haar-Wavelet ist folgendermassen definiert:

$$\psi(x) = \begin{cases} -1 & : \quad x \in [0, \frac{1}{2}) \\ 1 & : \quad x \in [\frac{1}{2}, 1) \\ 0 & : \quad sonst \end{cases}$$

Damit ist

$$\psi_k^n(x) := 2^{\frac{n}{2}}\psi(2^n - k) \begin{cases} -2^{\frac{n}{2}} & : \quad x \in [2^{\frac{n}{2}}k, 2^{\frac{n}{2}}(k+1)) \\ 2^{\frac{n}{2}} & : \quad x \in [2^{\frac{n}{2}}(k+1), 2^{\frac{n}{2}}(k+2)) \\ 0 & : \quad sonst \end{cases}$$

Diese Wavelets bilden ein vollständiges ONS in $L^2(\mathbb{R})$ (Beweis s. z.B. [12]).

Die Schnittmenge der Träger zweier verschiedener Treppenfunktionen eines Levels ist eine Nullmenge. Das heisst, ändert man den Koeffizienten einer solchen Funktion, so hat dies nur Auswirkungen auf diese eine „Treppenstufe“. Möchte man umgekehrt die Approximation manipulieren, so ist direkt bekannt, welche Koeffizienten dafür geändert werden müssen und welche nicht. Man spricht hierbei von *perfekter Lokalität*.
Der grosse Nachteil dieser Funktionen ist, dass Linearkombinationen bedingt durch ihre vielen Unstetigkeiten, nur sehr schlecht zum Approximieren von stetigen Funktionen geeignet sind. Hierfür eignen sich die nachfolgend aufgeführten linearen Wavelets besser.

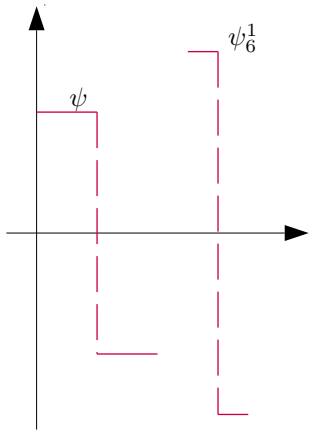


Abbildung 2: Das Haar-Wavelet ψ und ψ_6^1

1.5 Lineare Wavelets

Es ist möglich B-Splines als Skalierungsfunktion zu verwenden. Sei $N_1 = \chi_{[0,1)}$ der kardinale B-Spline der Ordnung 1 (Grad 0). Dann ist durch

$$N_k(x) := (N_{k-1} * N_1)(x) = \int_{-\infty}^{\infty} N_{k-1}(x-y)N_1(y)dy$$

der kardinale B-Spline der Ordnung $k \in \mathbb{N}$ gegeben. Für uns ist nur der lineare B-Spline (d.h. Grad=1) von Interesse. Dieser ergibt sich als:

$$\varphi(x) = \begin{cases} x & : \quad x \in [0,1) \\ 2-x & : \quad x \in [1,2) \\ 0 & : \quad sonst \end{cases}$$

Linearkombinationen dieser Hütchenfunktion ergeben einen Polygonzug und sind somit besser zur Approximation stetiger Daten geeignet als Treppenfunktionen. Der lineare B-Spline erfüllt folgende Verfeinerungsgleichung:

$$\varphi(x) = \frac{1}{2}\varphi(2x) + \varphi(2x-1) + \frac{1}{2}\varphi(2x-2)$$

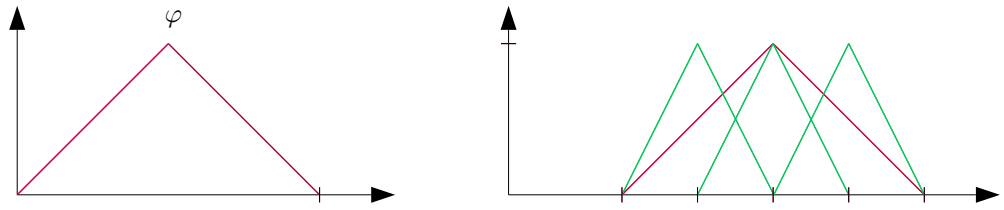


Abbildung 3: Der kardinale B-Spline und seine Verfeinerung

Wiederum lassen sich die Basen der Räume V_n durch Dilatation und Translation erzeugen:

$\{\varphi_k^n(x) := 2^{\frac{n}{2}}\varphi(2^n x - k), k \in \mathbb{Z}\}$ ist eine Basis von V_n.

Wie man sieht, überlappen sich die Träger zweier benachbarter Funktionen. Daher ist die Lokalität nicht so gut wie bei den Skalierungsfunktionen im Haar-Fall. Aufgrund der Nicht-Negativität der Funktionen bewirkt diese Überlappung auch, dass ihr Skalarprodukt nicht verschwindet. Es liegt also keine Orthonormalbasis vor, wohl aber eine Riesz-Basis (s. [12]). Die Gestalt des zugehörigen linearen Wavelets ist in Abbildung 4 zu sehen.
Auf die genaue Berechnung wird in Kapitel 3 eingegangen.

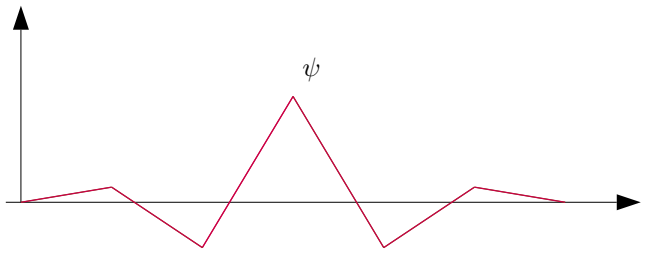


Abbildung 4: Das lineare Wavelet

Es existieren jedoch auch Skalierungsfunktionen, die keine Verfeinerungsgleichung erfüllen können. Folglich sind die von ihnen aufgespannten Räume auch nicht ineinander enthalten. Für diesen Fall entwirft Bonneau in [1] das Konstrukt der *MRA with Non-Nested Spaces,* , d.h. *der MSA mit nicht ineinander geschachtelten Räumen*, und gibt zwei verschiedene Anwendungen:

1. Die sog. BlaC-Wavelets - eine Mischung aus linearem und Haar-Wavelet, die den grössten Teil dieser Arbeit einnehmen.

2. Verfeinerung von Triangulierungen, die nicht auf Subdivision-Schemata basieren.

Im nächsten Kapitel wird jedoch zunächst das Konstrukt der *MSA mit nicht ineinander geschachtelten Räumen* eingeführt.

2 MSA mit nicht ineinander geschachtelten Räumen

2.1 Das allgemeine Konzept

In diesem Kapitel wird das Konzept der *MSA mit nicht ineinander geschachtelten Räumen,* eingeführt. Da dieses Konzept zur Implementierung gedacht ist, gehen wir im Folgenden davon aus, dass alle Räume endlich-dimensional sind. Ferner müssen die gegebenen Daten nicht aus $L^2(\mathbb{R})$ sein, sondern nur noch aus $L^2(\Omega)$, wobei $\Omega \subset \mathbb{R}$ ein Gebiet sein soll.

Wir betrachten also jetzt Teilräume $V_n \subset L_2(\Omega)$. Diese Räume sollen jedoch nicht ineinander enthalten sein, stattdessen sollen sie folgende Bedingungen erfüllen:

1) V_n ist isomorph zu einem Unterraum $\tilde{V}_n$ von V_{n+1}, d.h. $V_n \simeq \tilde{V}_n \subset V_{n+1}$
2) $\bigcup V_n$ ist dicht in $L_2(\Omega)$

Die zweite Bedingung ist intuitiv, da gegebene Approximationen f_n natürlich immer noch gegen f konvergieren sollen. Da die Räume nicht mehr ineinander enthalten sein müssen, ist die Forderung, dass der Schnitt nur die Null enthalten soll, nicht mehr sinnvoll.
Wir wählen folgende Bezeichnungen für die Dimensionen und die Basen:

$$\dim V_n = \dim \tilde{V}_n =: \nu(n) \leq \dim V_{n+1} =: \nu(n+1).$$

Es sei $\left\{\varphi_1^n, ..., \varphi_{\nu(n)}^n\right\}$ eine Basis von V_n, $\left\{\tilde{\varphi}_1^n, ..., \tilde{\varphi}_{\nu(n)}^n\right\}$ eine Basis von $\tilde{V}_n$ und $\left\{\varphi_1^{n+1}, ..., \varphi_{\nu(n+1)}^{n+1}\right\}$ eine Basis von V_{n+1}.

Eine Verfeinerungsgleichung, in der eine Funktion φ_i^n als Linearkombination der φ_j^{n+1} ausgedrückt wird, ist nun nicht mehr möglich. Stattdessen wird versucht, die Funktionen $\tilde{\varphi}_i^n$ als Linearkombination der φ_j^{n+1} möglichst nah an den Funktionen φ_i^n zu wählen. Die Verbindung zwischen einem Level n und Level $n+1$ wird nun daher in zwei Schritten erreicht:

1. Wende den Isomorphismus an, d.h. ersetze φ_j^n durch $\tilde{\varphi}_j^n$
2. Wende die Verfeinerungsgleichung für $\tilde{\varphi}_j^n$ an, d.h. drücke $\tilde{\varphi}_j^n$ als Linearkombination der φ_i^{n+1} aus:

$$\varphi_j^n \longleftrightarrow \tilde{\varphi}_j^n = \sum_{i=1}^{\nu(n+1)} p_{i,j}^n \varphi_i^{n+1}$$

In Matrixschreibweise:

$$\varphi^n \longleftrightarrow \tilde{\varphi}^n = \varphi^{n+1} P^n, \quad (p_{i,j}) = P^n \in \mathbb{R}^{\nu(n+1)\times\nu(n)} \tag{9}$$

Dies ist die sog. *approximierende Verfeinerungsgleichung,* da $\tilde{\varphi}_j^n$ die Funktion φ_j^n möglichst gut approximieren soll.

Analog zur herkömmlichen MSA wird nun der Waveletraum W_n definiert, diesmal jedoch als orthogonales Komplement von $\tilde{V}_n$ in V_{n+1}:

$$\tilde{V}_n \oplus W_n = V_{n+1} \Leftrightarrow W_n := V_{n+1} \ominus \tilde{V}_n.$$

Hierbei sei $\dim W_n = \nu(n+1) - \nu(n) =: \omega(n)$ und $\left\{\psi_1^n, ..., \psi_{\omega(n)}^n\right\}$ eine Basis von W_n bestehend aus den *Waveletfunktionen.* Da die Funktionen ψ_j^n in V_{n+1} enthalten sind, existiert eine Matrix $Q^n \in \mathbb{R}^{\nu(n+1)\times\omega(n)}$ mit

$$\psi^n = \varphi^{n+1}Q^n, \qquad \text{d.h.} \qquad \psi_j^n = \sum_{i=1}^{\nu(n+1)} q_{i,j}^n \varphi_i^{n+1}. \tag{10}$$

Wegen $\tilde{V}_n \oplus W_n = V_{n+1}$ existieren ausserdem Koeffizienten $(a_{i,j}^n)$, $(b_{i,j}^n)$, so dass

$$\varphi_j^{n+1} = \sum_{i=1}^{\nu(n)} a_{i,j}^n \tilde{\varphi}_i^n + \sum_{i=1}^{\omega(n)} b_{i,j}^n \psi_i^n$$

In Matrixschreibweise:

$$\tilde{\varphi}^n A^n + \psi^n B^n = \varphi^{n+1}, \text{ mit } A \in \mathbb{R}^{\nu(n)\times\nu(n+1)}, B \in \mathbb{R}^{\omega(n)\times\nu(n+1)} \tag{11}$$

P,Q heissen *Synthese-Matrizen,* A,B *Analyse-Matrizen.*
Aus (11) ergibt sich:

$$\begin{aligned} \varphi^{n+1} &= \tilde{\varphi}^n A^n + \psi^n B^n \\ &= \varphi^{n+1}P^n A^n + \varphi^{n+1}Q^n B^n \\ &= \varphi^{n+1}(P^n A^n + Q^n B^n) \end{aligned}$$

$$\Rightarrow P^n A^n + Q^n B^n = I \qquad \text{(Rekonstruktionsbedingung)} \tag{12}$$

In Blockschreibweise geschrieben bedeutet (11):

$$(\tilde{\varphi}^n|\psi^n)\begin{pmatrix} A^n \\ \hline B^n \end{pmatrix} = (\varphi^{n+1})$$

Andererseits gilt: $(\tilde{\varphi}^n|\psi^n) = (\varphi^{n+1})(P^n|Q^n)$

Daraus ergibt sich: $\left(\begin{array}{c} A^n \\ \hline B^n \end{array}\right) = (P^n|Q^n)^{-1}$

Beachte: $\left(\begin{array}{c} A^n \\ \hline B^n \end{array}\right)$ und $(P^n|Q^n)$ sind beide quadratisch ($\mathbb{R}^{\nu(n+1)\times\nu(n+1)}$),

und es gilt: $A^n P^n = Id_{\nu(n)}$ und $B^n Q^n = Id_{\omega(n)}$, d.h.
$A^n = (P^n)^{-1}, B^n = (Q^n)^{-1}$. (Pseudo-Inverse, da nicht quadratisch)

Insbesondere liefert dies zu wenig Gleichungen, um aus A,B die Matrizen P,Q oder umgekehrt zu berechnen. Man beachte weiterhin, dass allerdings A,B und P,Q voneinander abhängen, d.h. hat man z.B. ein Analyseverfahren gewählt, d.h. die Matrizen A,B festgelegt, so sind, wenn man noch gewisse Orthogonalitätsbedingungen zugrunde legt, auch P und Q bestimmt.
Umgekehrt sind nach Wahl von Skalierungsfunktion und Wavelet unter Orthogonalitätsbedingungen auch die Analyse-Matrizen eindeutig bestimmbar.

2.2 Orthogonalitätsbedingungen

Im Folgenden werden gewisse Orthogonalitätsbedingungen an die Wavelets und die Skalierungsfunktionen untersucht. Bonneau fordert hier, dass die Wavelets ψ_i^n zu allen Skalierungsfunktionen $\tilde{\varphi}_j^n$ orthogonal sein müssen, d.h. $< \psi_i^n, \tilde{\varphi}_j^n >= 0 \quad \forall i,j$. Diese Bedingung ist wegen $\tilde{V}_n \oplus W_n = V_{n+1}$ auf jeden Fall erfüllt. Man beachte, dass weder die Wavelets noch die Skalierungsfunktionen untereinander orthogonal sein müssen. Eine solche MSA heisst dann *semi-orthogonal.*
Bezeichnet $G_{n+1} =< \varphi_i^{n+1}, \varphi_j^{n+1} >$ die *Gram-Schmidt-Matrix,* so ist die Bedingung

$$< \psi_i^n, \tilde{\varphi}_j^n >= 0, \quad i = 1, ..., \omega(n), j = 1, ..., \nu(n)$$

äquivalent zu $P^T G_{n+1} Q = 0$, denn:

$$P^T G_{n+1} Q =$$

$$= \begin{pmatrix} p_{11} & \cdots & p_{\nu(n+1),1} \\ \vdots & & \vdots \\ p_{1,\nu(n)} & \cdots & p_{\nu(n+1),\nu(n)} \end{pmatrix} \begin{pmatrix} < \varphi_1^{n+1}, \varphi_1^{n+1} > & \cdots & < \varphi_1^{n+1}, \varphi_{N_{n+1}}^{n+1} > \\ \vdots & & \vdots \\ < \varphi_{N_{n+1}}^{n+1}, \varphi_1^{n+1} > & \cdots & < \varphi_{\nu(n+1)}^{n+1}, \varphi_{11}^{n+1} > \end{pmatrix} Q$$

$$= \begin{pmatrix} < \sum_{i=1}^{\nu(n+1)} p_{i1}\varphi_i^{n+1}, \varphi_1^{n+1} > & \cdots & < \sum_{i=1}^{\nu(n+1)} p_{i1}\varphi_i^{n+1}, \varphi_{\nu(n+1)}^{n+1} > \\ \vdots & & \vdots \\ < \sum_{i=1}^{\nu(n+1)} p_{i,\nu(n)}\varphi_i^{n+1}, \varphi_1^{n+1} > & \cdots & < \sum_{i=1}^{\nu(n+1)} p_{i,\nu(n)}\varphi_i^{n+1}, \varphi_{\nu(n+1)}^{n+1} > \end{pmatrix} \begin{pmatrix} q_{11} & \cdots & q_{1,\omega(n)} \\ \vdots & & \vdots \\ q_{N_{n+1},1} & \cdots & q_{\nu(n+1),\omega(n)} \end{pmatrix}$$

$$= \begin{pmatrix} < \sum_{i=1}^{\nu(n+1)} p_{i1}\varphi_i^{n+1}, \sum_{i=1}^{\nu(n+1)} q_{i1}\varphi_1^{n+1} > & \cdots & < \sum_{i=1}^{\nu(n+1)} p_{i1}\varphi_i^{n+1}, \sum_{i=1}^{\nu(n+1)} q_{i,\omega(n)}\varphi_i^{n+1} > \\ \vdots & & \vdots \\ < \sum_{i=1}^{\nu(n+1)} p_{i,\nu(n)}\varphi_i^{n+1}, \sum_{i=1}^{\nu(n+1)} q_{i1}\varphi_1^{n+1} > & \cdots & < \sum_{i=1}^{\nu(n+1)} p_{i,\nu(n)}\varphi_i^{n+1}, \sum_{i=1}^{\nu(n+1)} q_{i,\omega(n)}\varphi_i^{n+1} > \end{pmatrix}$$

$$= \begin{pmatrix} < \tilde{\varphi}_1^n, \psi_1^n > & \cdots & < \tilde{\varphi}_1^n, \psi_{\omega(n)}^n > \\ \vdots & & \vdots \\ < \varphi_{\nu(n)}^{\tilde{n}}, \psi_1^n > & \cdots & < \varphi_{\nu(n)}^{\tilde{n}}, \psi_{\omega(n)}^n > \end{pmatrix} = 0$$

Ferner ist $P^T G_{n+1} Q = 0$ äquivalent zu $A G_{n+1}^{-1} B^T = 0$, denn:

$$\begin{array}{lll} & P^T G_{n+1} Q = 0 & | \quad A^{-1} = P,\ B^{-1} = Q \\ \Leftrightarrow & (A^{-1})^T G_{n+1} B^{-1} = 0 & | \quad ()^T \\ \Leftrightarrow & [(A^{-1})^T (G_{n+1} B^{-1})]^T = 0 & \\ \Leftrightarrow & (B^{-1})^T G_{n+1}^T (A^{-1}) = 0 & | \quad (B^{-1})^T = (B^T)^{-1},\ G_{n+1}^T = G_{n+1} \\ \Leftrightarrow & (B^T)^{n+1} G_{n+1} A^{-1} = 0 & | \quad ()^{-1} \\ \Leftrightarrow & [(B^T)^{-1} G_{n+1} A^{-1}]^{-1} = 0 & \\ \Leftrightarrow & A G_{n+1}^{-1} B^T = 0. & \end{array}$$

Welche Bedingung man verwendet hängt davon ab, ob man die Synthese-Matrizen P und Q oder die Analyse-Matrizen A und B konstruiert. In letzterem Fall ergeben sich dann P und Q durch:

$$P = G_{n+1}^{-1} A^T (A G_{n+1}^{-1} A^T)^{-1}; \qquad Q = G_{n+1}^{-1} B^T (B G_{n+1}^{-1} B^T)^{-1} \tag{13}$$

Beweis: 1)

$$\begin{aligned}
&P = G_{n+1}^{-1} A^T (A G_{n+1}^{-1} A^T)^{-1} \\
&\Leftrightarrow [P(A G_{n+1}^{-1} A^T)]^T = [G_{n+1}^{-1} A^T]^T \\
&\Leftrightarrow (A G_{n+1}^{-1} A^T)^T P^T = A (G_{n+1}^{-1})^T \\
&\Leftrightarrow A (G_{n+1}^{-1})^T A^T P^T = A (B_{n+1}^T)^{-1} \\
&\Leftrightarrow A G_{n+1}^{-1} (\underbrace{PA}_{=I-QB})^T = A G_{n+1}^{-1} \\
&\Leftrightarrow A G_{n+1}^{-1} \underbrace{[I - QB]^T}_{=I-B^T Q^T} G_{n+1} = A \\
&\Leftrightarrow A \underbrace{G_{n+1}^{-1} G_{n+1}}_{=I} - \underbrace{A G_{n+1}^{-1} B^T}_{=0} Q^T G_{n+1} = A \\
&\Leftrightarrow A = A.
\end{aligned}$$

2)

$$\begin{aligned}
&Q = G_{n+1}^{-1} B^T (B G_{n+1}^{-1} B^T)^{-1} \\
&\Leftrightarrow Q(B G_{n+1}^{-1} B^T) = G_{n+1}^{-1} B^T \\
&\Leftrightarrow G_{n+1} \underbrace{QB}_{=I-PA} G_{n+1}^{-1} B^T = B^T \\
&\Leftrightarrow G_{n+1}[I - PA] G_{n+1}^{-1} B^T = B^T \\
&\Leftrightarrow G_{n+1} G_{n+1}^{-1} B^T - G_{n+1} P \underbrace{A G_{n+1}^{-1} B^T}_{=0} = B^T \\
&\Leftrightarrow B = B.
\end{aligned}$$

Hat man dagegen P und Q konstruiert, so ergeben sich A und B durch:

$$A = (P^T G_{n+1} P)^{-1} P^T G_{n+1}; \qquad B = (Q^T G_{n+1}^{-1} Q)^{-1} Q^T G_{n+1}^{-1} \tag{14}$$

Beweis: 1)

$$\begin{aligned}
&A = (P^T G_{n+1} P)^{-1} P^T G_{n+1} \\
&\Leftrightarrow (P^T G_{n+1} P) A = P^T G_{n+1} \\
&\Leftrightarrow (P^T G_{n+1} \underbrace{P)A}_{I-QB} G_{n+1}^{-1} = P^T \\
&\Leftrightarrow P^T G_{n+1} [I - QB] G_{n+1}^{-1} \\
&\Leftrightarrow P^T \underbrace{G_{n+1} G_{n+1}^{-1}}_{=I} - \underbrace{P^T G_{n+1} Q}_{=0} B G_{n+1}^{-1} = P^T \\
&\Leftrightarrow P^T = P^T.
\end{aligned}$$

2)

$$
\begin{aligned}
& B = (Q^T G_{n+1} Q)^{-1} Q^T G_{n+1} \\
\Leftrightarrow\ & B^T = G_{n+1} Q (Q^T G_{n+1} Q)^{-1} \\
\Leftrightarrow\ & G_{n+1}^{-1} B^T = Q (Q^T G_{n+1} Q)^{-1} \\
\Leftrightarrow\ & G_{n+1}^{-1} \underbrace{B^T Q^T}_{[I-PA]^T} G_{n+1} Q = Q \\
\Leftrightarrow\ & G_{n+1}^{-1} [I - (PA)^T] G_{n+1} Q = Q \\
\Leftrightarrow\ & \underbrace{G_{n+1}^{-1} G_{n+1}}_{=I} Q - G_{n+1}^{-1} A^T \underbrace{P^T G_{n+1} Q}_{=0} = Q \\
\Leftrightarrow\ & Q = Q. \qquad \square
\end{aligned}
$$

2.3 Analyse und Rekonstruktion

In diesem Abschnitt soll noch einmal näher erläutert werden, wie sich die Koeffizienten der nächst gröberen Darstellung und die Waveletkoeffizienten berechnen lassen. Sei eine Approximation $f^{n+1} = \sum_{i=1}^{\nu(n+1)} x_i^{n+1} \varphi_i^{n+1}$ in V_{n+1} einer Funktion f vorgegeben. Um jetzt die Koeffizienten x_i^n für die nächst gröbere Approximation $f^n = \sum_{i=1}^{\nu(n)} x_i^n \varphi_i^n$ zu erhalten, berechnet man zunächst die Koeffizienten von $\tilde{f}^n = \sum_{i=1}^{\nu(n)} x_i^n \tilde{\varphi}_i^n$ in $\tilde{V}_n$.

Jetzt wird der Isomorphismus angewendet. Dies äussert sich bei Bonneau durch das einfache Übernehmen der für $\tilde{f}^n$ berechneten Koeffizienten für die Funktion f^n. Hierbei entsteht ein Fehler, auf den später noch genauer eingegangen wird.

Weiterhin berechnet man die Detailkoeffizienten von $g^n = \sum_{i=1}^{\omega(n)} y_i^n \psi_i^n$.

Die Berechnung dieser und der Detail-Koeffizienten geschieht mit denselben Analyse-Matrizen:

$$x^n = A x^{n+1}, \quad y^n = B x^{n+1} \tag{15}$$

Denn es gilt: $\tilde{f}_n + g_n = f_{n+1}$

$$
\begin{aligned}
\Leftrightarrow \sum_{i=1}^{\nu(n)} x_i^n \tilde{\varphi}_i^n + \sum_{i=1}^{\omega(n)} y_i^n \psi_i^n &= \sum_{i=1}^{\nu(n+1)} x_i^{n+1} \varphi_i^{n+1} \\
&= \sum_{i=1}^{\nu(n+1)} x_i^{n+1} \Big(\sum_{k=1}^{\nu(n)} a_{ki} \tilde{\varphi}_k^n + \sum_{k=1}^{\omega(n)} b_{ki} \psi_k^n \Big) \\
&= \sum_{i=1}^{\nu(n+1)} \Big(\sum_{k=1}^{\nu(n)} x_i^{n+1} a_{ki} \tilde{\varphi}_k^n + \sum_{k=1}^{\omega(n)} x_i^{n+1} b_{ki} \psi_k^n \Big) \\
&= \sum_{k=1}^{\nu(n)} \tilde{\varphi}_k^n \Big(\sum_{i=1}^{\nu(n+1)} x_i^{n+1} a_{ki} \Big) + \sum_{k=1}^{\omega(n)} \psi_k^n \Big(\sum_{i=1}^{\nu(n+1)} x_i^{n+1} b_{ki} \Big) \\
&= \tilde{f}^n + g^n
\end{aligned}
$$

$$= \sum_{k=1}^{\nu(n)} x_k^n \tilde{\varphi}_k^n + \sum_{k=1}^{\omega(n)} y_k^n \psi_k^n$$

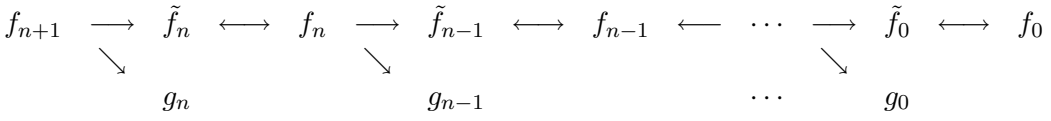
Koeffizientenvergleich ergibt: $x_k^n = \sum_{i=1}^{\nu(n+1)} x_i^{n+1} a_{ki}, \quad y_k^n = \sum_{i=1}^{\nu(n+1)} x_i^{n+1} b_{ki},$

d.h. $x^n = Ax^{n+1}, \quad y^n = Bx^{n+1}$.

Völlig analog lässt sich die inverse Transformation zeigen:

$$x^{n+1} = Px^n + Qy^n \tag{16}$$

Die Analyse- und Rekonstruktionsformeln sind somit dieselben wie im Fall der MSA mit geschachtelten Räumen, jedoch erfolgt die Approximation nun in zwei Schritten:



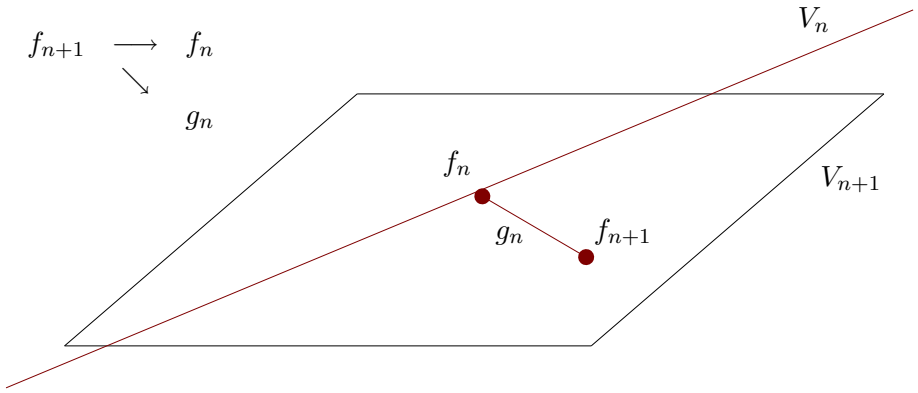


Abbildung 5: Rahmenbedingung der klassischen MSA

Demgegenüber:

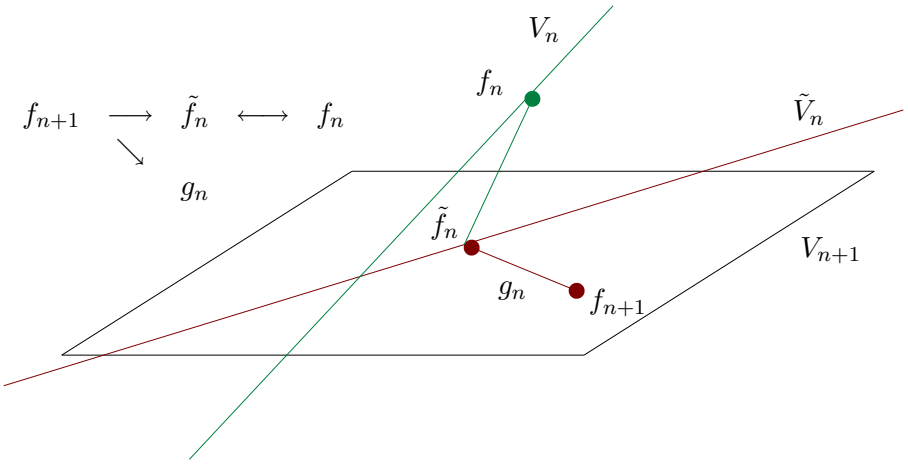


Abbildung 6: Rahmenbedingung der neuen MSA

Ein wichtiger Punkt ist die Wahl der Funktionen $\tilde{\varphi}_j^n$, bzw. die Wahl der Matrix P. Bei den BLaC-Wavelets ergeben sich die Synthese-Matrizen P und Q aus der Konstruktion der Skalierungsfunktionen, bzw. der Teilräume V_n und der Wavelets. Hier hängt die Wahl der $\tilde{\varphi}_j^n$ von der Anwendung und damit von der Wahl der Skalierungsfunktion φ_j^n ab, da $\tilde{\varphi}_j^n$ möglichst dicht an φ_j^n gewählt wird. Die Analyse-Matrizen A und B können dann aus 14 gewonnen werden.

Da die Approximationsräume nicht ineinander geschachtelt sind, kann es theoretisch vorkommen, dass sehr grobe Approximationen von den originalen Daten weit entfernt sind und dadurch Stabilitätsprobleme auftreten. Dies war jedoch in der folgenden Anwendung nicht der Fall. Dafür trat jedoch ein teilweise nicht unerheblicher Fehler dann auf, wenn die ursprünglichen Daten durch Addition von approximierter Skalierungsfunktion und den dazugehörigen Wavelets wiedergewonnen werden sollten. Eine genauere Untersuchung dieses Fehlers wird in Kapitel 5 speziell für die BLaC-Wavelets vorgenommen.
Diese werden nun im nächsten Kapitel eingeführt.

3 BLaC-Wavelets

In diesem Kapitel werden die letzten Ergebnisse verwendet, um eine von einem Parameter abhängige Familie von Wavelets zu definieren, die sog. BLaC-Wavelets. BLaC steht hierbei für *Blending of Linear and Constant.* Diese Familie stellt einen Übergang zwischen den Haar-Wavelets und den linearen Wavelets dar. Ihr Zweck ist es, einen Kompromiss zwischen der perfekten Lokalität der Haarwavelets und der besseren Regularität der linearen Wavelets zu finden. Der Anwender kann selbst entscheiden, welche Eigenschaft für seine Zwecke am dienlichsten ist.

3.1 Skalierungsfunktion des Haar- und des linearen Wavelets

Wie bereits im ersten Kapitel beschrieben ist das wohl einfachste Wavelet das Haar-Wavelet. Die zugehörige Skalierungsfunktion wurde dort wie folgt definiert:

$$\varphi(x) = \chi_{[0,1)}(x) = \begin{cases} 1 & : \quad x \in [0,1) \\ 0 & : \quad sonst \end{cases}$$

Daten die in dieser Basis dargestellt werden, entsprechen einer Treppenfunktion. Das bedeutet zwar eine perfekte Lokalität, gleichzeitig aber auch eine sehr schlechte Regularität. Etwas besser ist die Regularität der linearen Wavelets. Die zugehörige Skalierungsfunktion ist die sog. *Hütchenfunktion.*

$$\varphi(x) = \begin{cases} x & : \quad x \in [0,1) \\ 2-x & : \quad x \in [1,2] \\ 0 & : \quad sonst \end{cases}$$

In dieser Basis dargestellte Daten entsprechen einem Polygonzug.

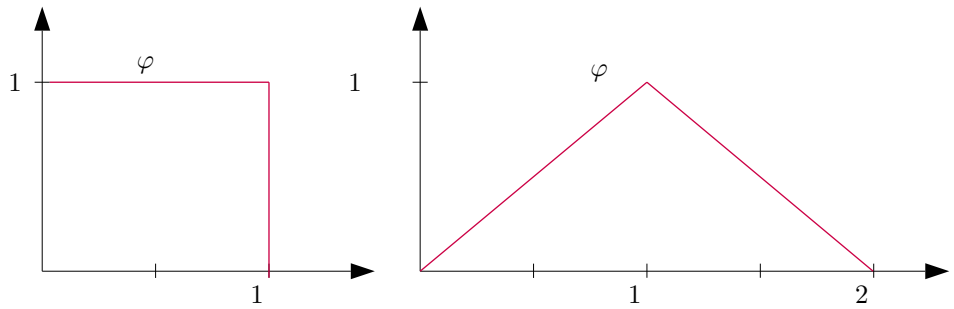


Abbildung 7: Skalierungsfunktion im Haar- und im linearen Fall

3.2 BLaC Skalierungsfunktion und approximierte Verfeinerungsgleichung

Die BLaC Skalierungs- und Waveletfunktionen, sowie die approximierende Verfeinerungsgleichung hängen von einem Übergangsparameter $\Delta \in [0,1]$ ab, und werden so berechnet, dass für $\Delta = 0$ die MSA der des Haar-Wavelets gleicht, und für $\Delta = 1$ der der linearen Wavelets.

Die BLaC Skalierungsfunktionen φ_i^n berechnen für $\Delta \neq 0$ sich aus Dilatationen mit dem Faktor 2^n und Translationen mit $(i2^{-n})$ der folgenden Funktion φ:

$$\varphi = \begin{cases} \frac{x}{\Delta} & : \quad 0 \leq x < \Delta \\ 1 & : \quad \Delta \leq x < 1 \\ -\frac{1}{\Delta}(x - 1 - \Delta) & : \quad 1 \leq x < 1 + \Delta \\ 0 & : \quad sonst \end{cases} \tag{17}$$

D.h. $\varphi_i^n(x) = \varphi(2^n(x - i2^{-n}))$.

Anmerkung: Bonneau verzichtet hier auf die allgemein übliche Normierung mit $2^{-\frac{n}{2}}$.

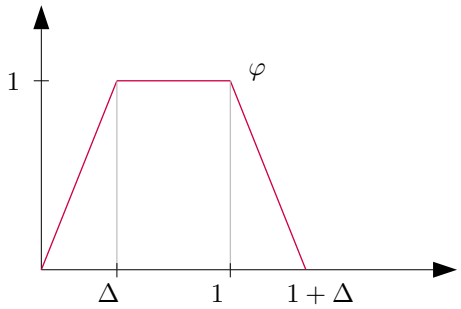


Abbildung 8: Grundfunktion der φ_i^n

Für $\Delta \longrightarrow 0$ konvergiert die Skalierungsfunktion gegen die Haar-Skalierungsfunktion, für $\Delta = 1$ ergibt sich die Hutfunktion.
Die Anwendung der BLaC-Wavelets soll auf beschränkten Gebieten erfolgen, daher wird die Restriktion auf $\Omega = [0, 1]$ betrachtet. Somit werden nur die Skalierungsfunktionen mit den Indizes $i = -1, ..., 2^n - 1$ verwendet.

Beispiel: Für den Level $n = 2$ ergeben sich somit $\varphi_i^2(x) = \varphi(4(x - \frac{i}{4}))$ für $i = -1, 0, 1, 2, 3$ als Skalierungsfunktionen.

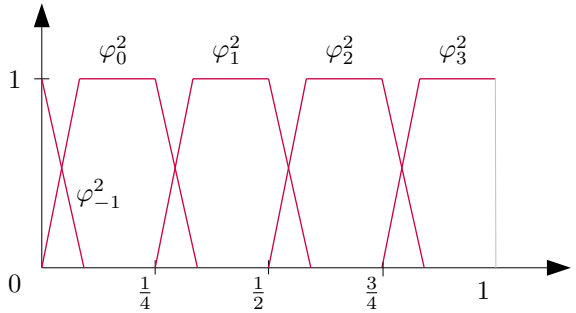


Abbildung 9: Skalierungsfunktionen φ_i^2

Damit wird nun der Raum V_n definiert durch:

$$V_n = span\,\{\varphi_i^n(x); i = -1, ..., 2^n - 1\} \;\text{ mit }\; \dim V_n = |\,\{-1, ..., 2^n - 1\}\,| = 2^n + 1 = \nu(n)$$

3.3 Eigenschaften der Skalierungsfunktion

Satz 2 *Sei $C([0,1])$ der Raum der auf dem Intervall [0,1] stetigen Funktionen, und $f \in C([0,1])$ gegeben. Definiere die Stützstellen η_i^n durch*

$$\eta_i^n = \frac{i}{2^n} + \frac{1}{2}\frac{1+\Delta}{2^n} = \frac{2i+1+\Delta}{2^{n+1}} \quad , i = 0, ..., 2^n - 2 \text{ und } \eta_{-1}^n = 0,\ \eta_{2^n-1}^n = 1.$$

Dann interpoliert die Funktion

$$BL_n f(x) = BLf(x,n) := \sum_{i=-1}^{2^n-1} f(\eta_i^n)\varphi_i^n(x) \tag{18}$$

die Punkte $(\eta_i^n, f(\eta_i^n))$.

Beweis: Die Stützstelle $\eta_i^n, i \in \{0, ..2^n - 2\}$ liegt exakt in der Mitte des Trägers der Funktion φ_i^n, nur η_{-1}^n und $\eta_{2^n-1}^n$ sind fest gewählt.

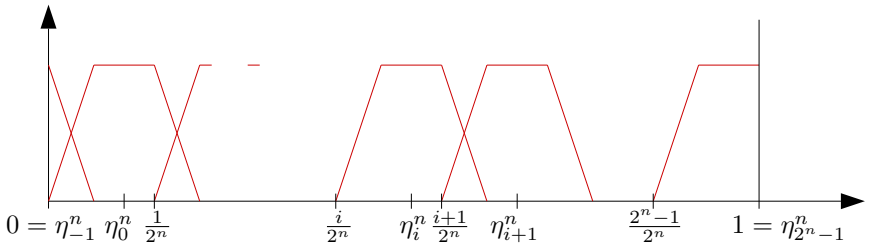


Abbildung 10: Die Stützstellenverteilung

Damit gilt: $\varphi_i^n(\eta_j^n) = \begin{cases} 1 & , \quad i = j \\ 0 & , \quad sonst \end{cases}$ für alle $i, j = -1, ..., 2^n - 1$,
und daraus folgt dann $BL_n f(\eta_j^n) = f(\eta_j^n)\varphi_j^n(\eta_j^n) = f(\eta_j^n)$ für alle $i, j = -1, ..., 2^n - 1$.

□

Bei der späteren Implementierung der Visualisierung einer MSA mit BLaC-Wavelets werden die gegebenen Daten mit Hilfe des BL_n-Operators angezeigt. Hierfür wird das Intervall [0,1] mit einer der Bildschirmauflösung entsprechenden Genauigkeit durchlaufen, und an den jeweiligen Stellen die Funktion $BLf(x,n)$ ausgewertet. Die Werte $f(\eta_i^n), i = -1, ..., 2^n - 1$ entsprechen dann den Glättekoeffizienten zum Level n.

Satz 3 *Die Funktion BL_n ist ein positiver, linearer Operator, d.h. es gilt für alle $f, g \in C([0,1]), \lambda, \mu \in \mathbb{R}$:*

i) $f \geq 0 \Longrightarrow BL_n(f, \cdot) \geq 0$

ii) $BL_n(\lambda f + \mu g) = \lambda BL_n(f) + \mu BL_n(g)$

Beweis:

i) folgt, da die φ_i^n nicht-negativ sind.

ii) Seien $f, g \in C([0,1]), \lambda, \mu \in \mathbb{R}$ dann gilt für alle $x \in [0,1]$:

$$BL_n(\lambda f + \mu g; x) = \sum_{i=-1}^{2^n-1} (\lambda f(\eta_i^n) + \mu g(\eta_i^n))\varphi_i^n(x)$$

$$= \lambda \sum_{i=-1}^{2^n-1} f(\eta_i^n)\varphi_i^n(x) + \mu \sum_{i=-1}^{2^n-1} g(\eta_i^n)\varphi_i^n(x) = \lambda BL_n(f;x) + \mu BL_n(g;x)$$

□

Satz 4 *Die Skalierungsfunktionen eines Levels partitionieren die Eins, d.h. es gilt*

$$\sum_{i=-1}^{2^n-1} \varphi_i^n(x) = \varphi_{j-1}^n(x) + \varphi_j^n(x) = 1$$

Der Operator BL_n reproduziert also die 1-Funktion.

Beweis: Für $x = 1$ ist $\varphi_i^n(x) = 0$ für alle $i = -1, ..., 2^n - 2$ und $\varphi_{2^n-1}^n(x) = 1$, hier gilt die Behauptung also. Für $x \in [\frac{i_0}{2^n}, \frac{i_0+1}{2^n})$ mit $i_0 \in \{0, ..., 2^n - 2\}$ wird folgende Fallunterscheidung betrachtet:

1. Fall: $x \in [\frac{i_0}{2^n}, \frac{i_0+\Delta}{2^n})$
   $\Longrightarrow \sum_{i=-1}^{2^n-1} \varphi_i^n(x) = \varphi_{i_0-1}^n(x) + \varphi_{i_0}^n(x) = \varphi(2^n x - (i_0 - 1)) + \varphi(2^n x - i_0)$
   $= -\frac{1}{\Delta}(2^n x - i_0 + 1 - \Delta - 1) + \frac{2^n x - i_0}{\Delta} = \frac{2^n x - i_0 - 2^n x + i_0 + \Delta}{\Delta} = \frac{\Delta}{\Delta} = 1.$
2. Fall: $x \in [\frac{i_0+\Delta}{2^n}, \frac{i_0+1}{2^n})$
   $\Longrightarrow \sum_{i=-1}^{2^n-1} \varphi_i^n(x) = \varphi_{i_0}^n(x) = 1$ nach Definition von φ.

□

Die Summe $\sum_i x_i \varphi_i^n$ ist endlich und ergibt aufgrund der Nicht-Negativität der Skalierungsfunktion eine Konvexkombination der Daten x_i. Dies sichert numerische Stabilität, denn seien x_i exakte Koeffizienten der Daten und x_i^* mit Fehlern behaftet, dann gilt:

$$|\sum_i x_i \varphi_i^n(x) - \sum_i x_i^* \varphi_i^n(x)| \leq \sum_i |x_i - x_i^*||\varphi_i^n(x)| = \sum_i |x_i - x_i^*|\varphi_i^n(x)$$

$$\leq \max_i |x_i - x_i^*| \underbrace{\sum_i \varphi_i^n(x)}_{1} = \max_i |x_i - x_i^*|$$

Der Ausgabefehler ist bei Konvexkombinationen also höchstens so gross wie der Eingabefehler.

Satz 5 *Für $f \in C([0,1])$ gilt: $\lim_{n\to\infty} BL_n f(x) = f(x)$ für alle $x \in [0,1]$.*

Beweis: Der Beweis wird mit Hilfe des ersten Stetigkeitsmoduls geführt. Dieser ist definiert durch

$$\omega(f,\varepsilon) = sup\{|f(x) - f(y)| : d(x,y) \leq \varepsilon\},$$

wobei f aus einem kompakten Raum $C(X)$ mit der Metrik d stammt. In unserem Fall ist $C(X) = C([0,1])$ und $d(x,y) = |x-y|$ die Betragsnorm. Wir wenden nun aus [7], S. 414 Theorem 3.1 an und setzen dort $A(f,y) = f(y)$, $Y = X$, $\psi_A(y) = 1$ und $g_A(y) = y$. Theorem 3.1 ergibt sich dann wie folgt:
Ist $BL_n \neq 0$ ein positiver linearer Operator von $C([0,1])$ nach $C([0,1])$, dann gilt für alle $f \in C([0,1])$, $y \in X$ und $\varepsilon > 0$ die folgende Ungleichung:

$$|BL_n(f;y) - f(y)| \leq max\{||BL_n||, \frac{1}{\varepsilon} BL_n(d(\cdot,y);y)\}\tilde{\omega}(f;\varepsilon) + |BL_n(1,y) - 1| \cdot |f(y)| \quad (19)$$

Da der BL_n-Operator die 1-Funktion reproduziert, verschwindet der zweite Summand, ferner gilt $||BL_n|| = 1$. Ist $\tilde{\omega}$ wie in [7], S.414 definiert, dann gilt $\tilde{\omega}(f;\varepsilon) \leq 2\omega(f;\varepsilon)$. Damit wird (19) zu:

$$|BL_n(f;y) - f(y)| \leq 2max\{1, \frac{1}{\varepsilon} BL_n(d(\cdot,y);y)\}\omega(f;\varepsilon) \quad (20)$$

Nun ist

$$BL_n(d(\cdot,y);y) = BL_n(|\cdot - y|;y) = \sum_{i=-1}^{2^n-1} |\eta_i^n - y|\varphi_i^n(y) =: (*).$$

Sei nun $x \in [\frac{k}{2^n}, \frac{k+1}{2^n}), k \in \{0,...,2^n-1\}$, ist $x = 1$, so wähle $k = 2^{n-1}$. Die obige Summe besteht dann aus höchstens zwei Summanden:

$$(*) \quad = \quad |\eta_{k-1}^n - y| \underbrace{\varphi_{k-1}^n(y)}_{\leq 1} + |\eta_k^n - y| \underbrace{\varphi_k^n(y)}_{\leq 1} \leq \underbrace{|\eta_{k-1}^n - y|}_{\leq \frac{1}{2^{n-1}}} + \underbrace{|\eta_k^n - y|}_{\leq \frac{1}{2^{n-1}}} \leq \frac{2}{2^{n-1}} = \frac{1}{2^{n-2}}$$

Denn ein Punkt $x \in [\frac{k}{2^n}, \frac{k+1}{2^n})$ hat zur Stützstelle η_{k-1}^n höchstens den Abstand

$$\frac{k+1}{2^n} - \eta_{k-1}^n = \frac{2k+2}{2^{n+1}} - \frac{2k-2+1+\Delta}{2^{n+1}} = \frac{3+\Delta}{2^{n+1}} \leq \frac{4}{2^{n+1}} = \frac{1}{2^{n-1}}.$$

Wähle nun $\varepsilon = \frac{1}{2^{n-2}}$, dann ergibt sich aus (20):

$$\begin{aligned} |BL_n(f;y) - f(y)| &\leq 2max\{1, 2^{n-2}\frac{1}{2^{n-2}}\}\omega(f;\frac{1}{2^{n-2}}) \\ &= 2\omega(f;\frac{1}{2^{n-2}}) \longrightarrow 0 \text{ für } n \longrightarrow \infty. \end{aligned}$$

Damit gilt: $BL_n(f;x) \longrightarrow f(x)$ für $n \longrightarrow \infty \quad \forall x \in [0.1]$.

□

Bemerkungen:

1. Nach dem letzten Satz kann also eine gegebene Funktion $f \in C([0,1])$ mit wachsendem Level beliebig nah approximiert werden. Dies entspricht der zweiten Forderung der Definition einer MSA: $\bigcup_n V_n$ ist dicht in $L^2([0,1])$.

2. Das Intervall [0,1] wird hier nur der Allgemeinheit wegen betrachtet. Durch lineare Transformation lassen sich die Ergebnisse auf beliebige Intervalle [a,b] mit $a < b$ übertragen.

Nun soll noch gezeigt werden, dass die ganzzahligen Translate der BLaC-Skalierungsfunktion eine Riesz-Basis bilden. Hierfür benötigen wir noch einige Resultate aus der Fourier- und Wavelet-Analysis. Die meisten dürften bekannt sein und werden daher nur zitiert. Die nicht aufgeführten Beweise finden sich z.B. in [5] oder in [14].

3.4 Elementare Resultate aus der Fourier- und Wavelet-Analysis

Definition 6 *Für $f \in L^1(\mathbb{R})$ ist die **Fourier-Transformation** $F : L^1 \longmapsto C$ definiert durch:*

$$F(f) = \hat{f}(u) = \int_{-\infty}^{\infty} f(x)e^{-iux}dx \quad (u \in \mathbb{R})$$

*Die **inverse Fourier-Transformation** ist gegeben durch:*

$$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{f}(u)e^{iux}du \quad (\hat{f} \in L^1, x \in \mathbb{R})$$

Insbesondere ist $\hat{\hat{f}} = f$ für $f \in L^1 \cap C, \hat{f} \in L^1 \cap C$.

Satz 6 *(Plancherel) Die Fourier-Transformation lässt sich zu einer beschränkten linearen bijektiven Abbildung von L^2 auf sich ausdehnen, $F : L^2 \longmapsto L^2$.*
Insbesondere gilt:

i) $f \in L^2 \Longrightarrow \hat{f} \in L^2$
ii) $||\hat{f}||_2^2 = 2\pi||f||_2^2$
iii) $< f, g >= \frac{1}{2\pi} < \hat{f}, \hat{g} >$ *(ii) und iii) heissen auch **Parsevalsche Gleichung**)*
iv) *Für $f \in L^2$ ist $F^{-1}(Ff) = f$.*

Lemma 2 *Für $f \in L^2 \cap L^1$ gilt: $\widehat{f(\cdot - h)}(u) = e^{-iuh}\hat{f}(u)$.*

Beweis: Es gilt:

$$\widehat{f(\cdot - h)}(u) = \int_{-\infty}^{\infty} f(x-h)e^{-iux}dx = \int_{-\infty}^{\infty} f(y)e^{-iu(y+h)}dy \qquad \text{(Subst. } y = x - h\text{)}$$

$$= e^{-iuh}\int_{-\infty}^{\infty} f(y)e^{-iuy}dy = e^{-iuh}\hat{f}(u)$$

□

Desweiteren soll noch kurz auf Fourier-Reihen eingegangen werden. Die Idee der Fourier-Reihen beruht auf:

Satz 7 *Das System* $\{e^{ik\cdot}, k \in \mathbb{Z}\}$ *bildet ein vollständiges Orthonormalsystem in* $L^2(0, 2\pi)$. *Hierbei ist* $L^2(0, 2\pi)$ *der Raum der* 2π*-periodischen auf* $(0, 2\pi)$ *quadratisch integrierbaren Funktionen mit dem Skalarprodukt*

$$< f, g >_{L^2(0,2\pi)} = \frac{1}{2\pi} \int_0^{2\pi} f(x)\overline{g(x)}dx \qquad , f, g \in L^2(0, 2\pi).$$

Definition 7 *Sei* $f \in L^2(0, 2\pi)$ *gegeben. Dann ist die* ***Fourier-Reihe*** *von f gegeben durch*

$$Sf(x) = \sum_{k\in\mathbb{Z}} c_k(f)e^{ikx}.$$

Hierbei lassen sich die ***Fourier-Koeffizienten*** $c_k(f)$ *berechnen durch:*

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

Satz 8 *Die Funktion* $\Phi_f(x) := \sum_{k\in\mathbb{Z}} f(x + 2\pi k), \quad f \in L^2(\mathbb{R})$, *heisst die* $\mathbf{2\pi}$***-Periodisierung*** *von f, und es gilt* $\Phi_f \in L^2(0, 2\pi)$.

Satz 9 *Sei* $\varphi \in L^2(\mathbb{R})$. *Dann bildet* $\{\varphi_k = \varphi(\cdot - k); k \in \mathbb{Z}\}$ *genau dann eine Riesz-Basis, wenn für das sog.* ***Autokorrelationssymbol*** $\Phi(u) := \sum_{l\in\mathbb{Z}} |\hat{\varphi}(u+2\pi l)|^2$ *gilt* $A \le \Phi(u) \le B$ *f.ü. mit den Riesz-Basis-Konstanten A,B.*

Mit diesen Resultaten ist es nun möglich, die Riesz-Eigenschaft der BLaC-Skalierungsfunktionen zu zeigen.

3.5 Die Riesz-Eigenschaft der BLaC-Skalierungsfunktionen

Satz 10 *Sei* φ *die Skalierungsfunktion der BLaC-Wavelets und* $\varphi_i^n(x) = \varphi(2^n x - i)$, *dann gilt: Die Funktionen* $\varphi_i^n, i \in \mathbb{Z}$ *bilden eine Riesz-Basis, d.h. es existieren positive Konstanten A,B, so dass für alle Folgen* $(a_k)_{k\in\mathbb{Z}} \in l^2(\mathbb{Z})$ *gilt:*

$$A \sum_k |a_k|^2 \le || \sum_k a_k \varphi_k^n(\cdot)|| \le B \sum_k |a_k|^2$$

Beweis: Wir betrachten zunächst nur die Funktion φ vom Level 0 und ihre ganzzahligen Translate. Nach Satz 9 genügt es zu zeigen:

$$\exists A, B > 0 : \quad A \le \Phi(u) \le B \quad \text{,wobei } \Phi(u) \text{ wie in Satz 9 definiert sei.}$$

Wir entwickeln Φ in eine Fourier-Reihe:

$$\Phi(u) := \sum_{l\in\mathbb{Z}} |\hat{\varphi}(u + 2\pi l)|^2 = \sum_{k\in\mathbb{Z}} c_{-k}e^{-iku}$$

mit den Fourier-Koeffizienten

$$\begin{aligned} c_{-k} &= \frac{1}{2\pi}\int_0^{2\pi}\Phi(u)e^{iku}du = \frac{1}{2\pi}\int_0^{2\pi}\sum_{l\in\mathbb{Z}}|\hat{\varphi}(u+2\pi l)|^2 e^{iku}du = \frac{1}{2\pi}\int_{-\infty}^{\infty}\hat{\varphi}(u)\underbrace{\hat{\varphi}(u)e^{iku}}_{=\hat{\varphi}(\cdot-k)\ \text{(L. 2)}}du \\ &= \frac{1}{2\pi} <\hat{\varphi},\hat{\varphi}(\cdot-k)> \overset{\text{Parseval}}{=} <\varphi,\varphi(\cdot-k)> = <\varphi_0,\varphi_k> \end{aligned}$$

Damit ergibt sich die Fourier-Reihe von Φ als:

$$\Phi(u) = \sum_{k\in\mathbb{Z}} <\varphi_0,\varphi_k> e^{-iku} \tag{21}$$

Es müssen also die Skalarprodukte $<\varphi_0,\varphi_k>, k\in\mathbb{Z}$ bestimmt werden. Es genügt nur $<\varphi_{k+1},\varphi_k> = <\varphi_{k-1},\varphi_k> = <\varphi_{-1},\varphi_0>$ und $<\varphi_k,\varphi_k> = <\varphi_0,\varphi_0>$ zu berechnen, da sich nur die Träger benachbarter Funktionen überschneiden:

$$\begin{aligned} <\varphi_{-1},\varphi_0> &= <\varphi(\cdot+1),\varphi> = \int_0^{\Delta}\frac{x}{\Delta}(-\frac{1}{\Delta}(x+1-1-\Delta))dx = -\frac{1}{\Delta^2}\int_0^{\Delta}x^2-\Delta x dx \\ &= -\frac{1}{\Delta^2}(\frac{1}{3}x^3-\frac{\Delta}{2}x^2)\Big|_0^{\Delta} = -\frac{1}{\Delta^2}(\frac{\Delta^3}{3}-\frac{\Delta^2}{2}) = -\frac{1}{\Delta^2}(-\frac{\Delta^3}{6}) \\ &= \frac{\Delta}{6} \end{aligned}$$

$$\begin{aligned} <\varphi_0,\varphi_0> &= \int_0^{1+\Delta}\varphi(x)^2dx = \underbrace{\int_0^{\Delta}\frac{x^2}{\Delta^2}dx}_{I} + \underbrace{\int_{\Delta}^{1}1dx}_{II} + \underbrace{\int_1^{1+\Delta}(-\frac{1}{\Delta}(x-1-\Delta))^2dx}_{III} \\ I &= \frac{1}{3\Delta^2}x^3\Big|_0^{\Delta} = \frac{\Delta^3}{3\Delta^2}-0 = \frac{\Delta}{3} \\ II &= 1-\Delta \\ III &= \frac{1}{\Delta^2}\int_1^{1+\Delta}(x-1-\Delta)^2dx \overset{y=x-1-\Delta}{=} \frac{1}{\Delta^2}\int_{1-1-\Delta}^{1+\Delta-1-\Delta}y^2dy \\ &= \frac{1}{\Delta^2}\int_{-\Delta}^{0}y^2dy = \frac{1}{3\Delta^2}y^3\Big|_{-\Delta}^{0} = 0-\frac{\Delta^3}{3\Delta^2} = \frac{\Delta}{3} \end{aligned}$$

Also ist: $<\varphi_0,\varphi_0> = \frac{\Delta}{3}+1-\Delta+\frac{\Delta}{3} = 1-\frac{\Delta}{3}$.

Damit ergibt sich für die Fourier-Reihe von Φ:

$$\begin{aligned}\Phi(u) &= \sum_{k\in\mathbb{Z}} < \varphi_0, \varphi_k > e^{-iku} =< \varphi_0, \varphi_{-1} > e^{iu} + < \varphi_0, \varphi_0 > + < \varphi_0, \varphi_1 > e^{-iu} \\ &= \frac{\Delta}{6}e^{iu} + 1 - \frac{\Delta}{3} + \frac{\Delta}{6}e^{-iu} = \frac{\Delta}{6}\underbrace{(e^{iu} + e^{-iu})}_{=2cos(u)} + 1 - \frac{\Delta}{3} \\ &= \frac{\Delta}{3}cos(u) + 1 - \frac{\Delta}{3}\end{aligned}$$

$$\begin{aligned}\text{Nun gilt:} \quad & \frac{\Delta}{3}\underbrace{cos(u)}_{\geq -1} + 1 - \frac{\Delta}{3} \geq 1 - 2\frac{\Delta}{3} =: A \\ & \frac{\Delta}{3}\underbrace{cos(u)}_{\leq 1} + 1 - \frac{\Delta}{3} \leq 1 =: B\end{aligned}$$

A und B sind somit die Riesz-Konstanten für die Skalierungsfunktionen vom Level 0. Für $\Delta = 1$ ergeben sich die für die linearen B-Splines bekannten Riesz-Schranken $A = \frac{1}{3}$ und $B = 1$. Für die Haar-Skalierungsfunktionen gilt $A = B = 1$, hier liegt eine Orthonormalbasis vor. Je näher der Parameter Δ der Null ist, desto geringer ist der Abstand zwischen A und B, d.h. desto grösser ist die numerische Stabilität der Berechnungen. Wegen

$$\begin{aligned}< \varphi_i^n, \varphi_j^n > &= \int_{-\infty}^{\infty} \varphi(2^n x - i)\varphi(2^n x - i)dx \overset{y=2^n x}{=} \int_{-\infty}^{\infty} \varphi(y - i)\varphi(y - j)\frac{1}{2^n}dy \\ &= \frac{1}{2^n} < \varphi_i^0, \varphi_j^0 >\end{aligned}$$

unterscheiden sich die Riesz-Konstanten von φ_i^n von den Funktionen vom Level 0 nur um den Faktor $\frac{1}{2^n}$. Damit bilden die Funktionen $\varphi_i^n, i \in \mathbb{Z}$ eine Riesz-Basis mit den oben berechneten Riesz-Schranken $\frac{1}{2^n}A$ und $\frac{1}{2^n}B$.

□

3.6 Notwendigkeit der MSA mit nicht ineinander geschachtelten Räumen

Abbildung 11 zeigt, dass man allerdings eine Skalierungsfunktion φ_i^n nicht durch Skalierungsfunktionen eines höheren Levels ausdrücken kann, d.h. $V_n \not\subseteq V_{n+1}$. Denn in dem gestrichelten Bereich ist φ_0^n linear steigend, während dort alle φ_i^{n+1} konstant sind. Man benötigt also das Konstrukt der *MSA mit nicht ineinander geschachtelten Räumen.*

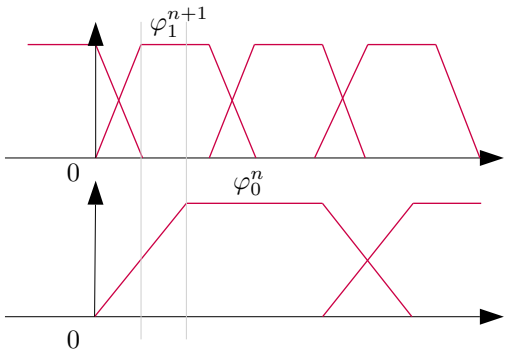


Abbildung 11: Die Räume sind nicht ineinander geschachtelt!

Ziel ist es nun, die isomorphen Funktionen $\tilde{\varphi}_i^n$ so zu wählen, dass die approximierende Verfeinerungsgleichung $\varphi_j^n \longleftrightarrow \tilde{\varphi}_j^n = \sum_i q_{i,j}^n \varphi_i^{n+1}$ eine Vermischung (Blending) der Verfeinerungsgleichung der Haar- und der linearen Verfeinerungsgleichung ist.

Die Verfeinerungsgleichung für das Haar-Wavelet lautete:

$$\varphi_i^n = \varphi_{2i}^{n+1} + \varphi_{2i+1}^{n+1}$$

Und für den linearen Fall:

$$\varphi_i^n = \frac{1}{2}\varphi_{2i}^{n+1} + \varphi_{2i+1}^{n+1} + \frac{1}{2}\varphi_{2i+2}^{n+1}$$

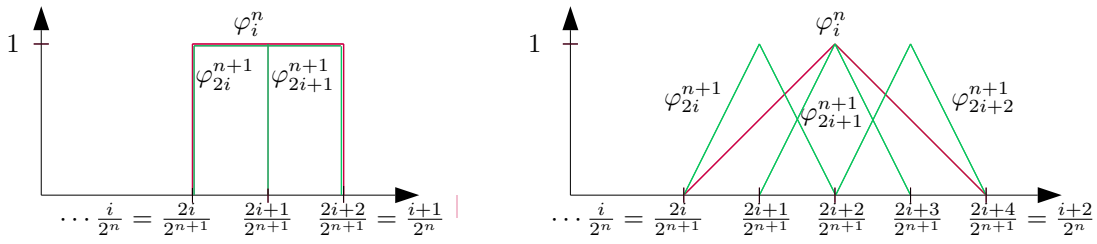


Abbildung 12: Die klassischen Verfeinerungsgleichungen

Dies führt zu folgender Approximation von φ_i^n:

$$\tilde{\varphi}_i^n = \alpha\varphi_{2i}^{n+1} + \varphi_{2i+1}^{n+1} + (1-\alpha)\varphi_{2i+2}^{n+1} \qquad ,\alpha \in \mathbb{R} \tag{22}$$

Hierbei hängt α nur vom Parameter Δ ab, für $\Delta = 0$ (Haar-Fall) ist $\alpha = 1$, und für $\Delta = 1$ (Linearer Fall) ist $\alpha = \frac{1}{2}$. Daraus ergeben sich sofort die vorausgesetzten Verfeinerungsgleichungen.

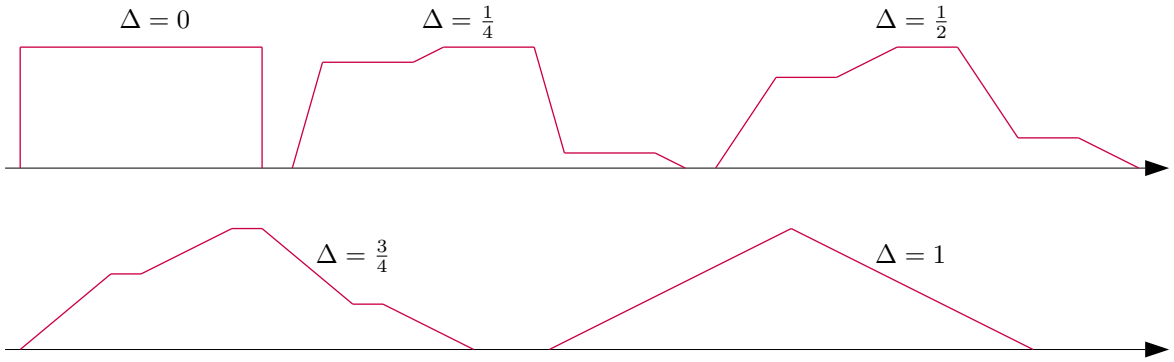


Abbildung 13: Die approximierte Skalierungsfunktion für verschiedene Parameter Δ

Linear interpoliert ergibt sich für α: $\alpha = -\frac{1}{2}\Delta + 1$

An den Randpunkten des Intervalls [0,1] hat die approximierende Verfeinerungsgleichung folgende Gestalt:

$$\varphi_{-1}^{n} \longleftrightarrow \tilde{\varphi}_{-1}^{n} = \varphi_{-1}^{n+1} + (1-\alpha)\varphi_{0}^{n+1}$$

$$\varphi_{2^n-1}^{n} \longleftrightarrow \tilde{\varphi}_{2^n-1}^{n} = \alpha\varphi_{2^{n+1}-2}^{n+1} + \varphi_{0}^{2^{n+1}-1}$$

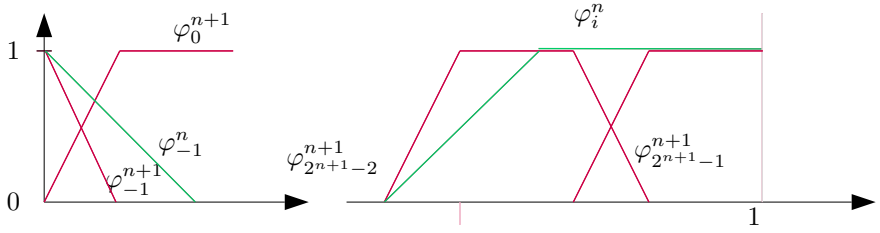


Abbildung 14: Die Situation an den Rändern

Damit ergibt sich nun die Matrix P^n durch: $\tilde{\varphi}^n = \varphi^{n+1} P^n \Leftrightarrow$

$$(\tilde{\varphi}^n_{-1}, ..., \tilde{\varphi}^n_{2^n-1}) = (\varphi^{n+1}_{-1}, ..., \varphi^{n+1}_{2^{n+1}-1}) \underbrace{\left.\begin{pmatrix} 1 & & & & \\ 1-\alpha & \alpha & & 0 & \\ & 1 & \ddots & & \\ & 1-\alpha & \ddots & & \\ & & \ddots & \alpha & \\ & & & 1 & \\ & 0 & & 1-\alpha & \alpha \\ & & & & 1 \end{pmatrix}\right\} 2^{n+1}+1 = \nu(n+1)}_{2^n+1=\nu(n)}$$

3.7 BLaC-Wavelet-Funktionen

Dieser Abschnitt beschäftigt sich mit der Herleitung der BLaC-Wavelets $\psi^n_i = \sum_j q^n_{j,i} \varphi^{n+1}_j$, d.h. mit den Basisfunktionen des Raums $W_n = V_{n+1} \ominus V_n$. Für die Berechnung der Koeffizienten $q^n_{j,i}$ werden zwei Bedingungen gestellt:

1. Semi-Orthogonalität, d.h. das Wavelet ψ^n_i muss zu allen Skalierungsfunktionen vom Level n orthogonal sein:

   $$\psi^n_i \perp \tilde{\varphi}^n_j \Leftrightarrow < \psi^n_i, \tilde{\varphi}^n_j >= 0 \quad \forall i, j.$$

2. Der Träger des Wavelets soll minimal sein.

$\tilde{\varphi}^n_i = \alpha\varphi^{n+1}_{2i} + \varphi^{n+1}_{2i+1} + (1-\alpha)\varphi^{n+1}_{2i+2}$ lebt auf den Trägern von fünf verschiedenen φ^{n+1}_j, nämlich $\varphi^{n+1}_{2j-1}, ..., \varphi^{n+1}_{2j+3}$.

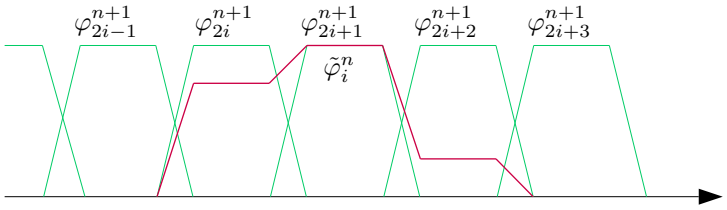


Abbildung 15: $\tilde{\varphi}^n_i$ überschneidet sich mit fünf φ^{n+1}_j

D.h. um die Semi-Orthogonalität zu erfüllen, muss ψ^n_i als Linearkombination von mindestens fünf Skalierungsfunktionen angesetzt werden. Wegen der zweiten Bedingung lautet also der Ansatz:

$$\psi^n_i = a\varphi^{n+1}_{2i} + b\varphi^{n+1}_{2i+1} + c\varphi^{n+1}_{2i+2} + d\varphi^{n+1}_{2i+3} + e\varphi^{n+1}_{2i+4}$$

Da sich auch die ψ_i^n durch Dilatation und Translation aus einer Funktion ψ herleiten lassen, sind die Koeffizienten a, b, c, d, e für alle Level n identisch. Wie bei den Skalierungsfunktionen wird auch bei den Wavelets auf den Vorfaktor $2^{\frac{n}{2}}$ verzichtet:

$$\psi_i^n = \psi(2^n x - i)$$

Der Träger eines solchen Wavelets, also der von fünf aufeinanderfolgenden Skalierungsfunktionen vom Grad n+1, überschneidet sich mit dem Träger von vier aufeinanderfolgenden Basisfunktionen aus $\tilde{V}_n$ (s. Abbildung 16). Damit reduziert sich die Semi-Orthogonalität zu:

$$< \psi_i^n, \tilde{\varphi}_j^n >= 0 \qquad , j = i-1, i, i+1, i+2.$$

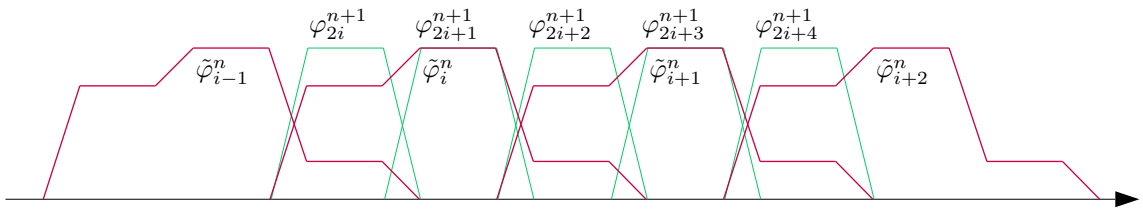


Abbildung 16: Das Wavelet überschneidet sich mit vier Skalierungsfunktionen

Es ergibt sich somit ein lineares Gleichungssystem mit vier Gleichungen und den fünf Unbekannten a, b, c, d, e. Um dieses eindeutig lösen zu können, wird noch die folgende Normierung verlangt:

$$||\psi||_{L_2}^2 = \int |\psi|^2 = 1$$

Die genaue Beschaffenheit dieses Gleichungssystems wird im nächsten Kapitel untersucht. Da die Funktionen φ_{-2}^{n+1} und $\varphi_{2^{n+1}}^{n+1}$ keinen Anteil am Intervall [0,1] haben, ergeben sich für die Wavelets ψ_{-1}^n und $\psi_{2^n-2}^n$ folgende Ansätze:

$$\psi_{-1}^n = b_0\varphi_{-1}^{n+1} + c_0\varphi_0^{n+1} + d_0\varphi_1^{n+1} + e_0\varphi_2^{n+1}$$
$$\psi_{2^n-2}^n = a_1\varphi_{2^{n+1}-4}^{n+1} + b_1\varphi_{2^{n+1}-3}^{n+1} + c_1\varphi_{2^{n+1}-2}^{n+1} + d_1\varphi_{2^{n+1}-1}^{n+1}$$

Diese Koeffizienten berechnet man durch Lösen der beiden linearen Gleichungssysteme

$$< \psi_{-1}^n, \tilde{\varphi}_j^n >= 0 \qquad , j = -1, 0, 1$$
$$< \psi_{2^n-2}^n, \tilde{\varphi}_j^n >= 0 \qquad , j = -2^n - 3, 2^n - 2, 2^n - 1$$

unter Zuhilfenahme der Bedingung

$$\int |\psi_{-1}|^2 = \int |\psi_{2^n-2}|^2 = 1.$$

Damit ergibt sich $Q \in \mathbb{R}^{\nu(n+1)\times\omega(n)}$ durch: $\psi^n = \varphi^{n+1}Q^n \Leftrightarrow$

$$(\psi^n_{-1},...,\psi^n_{2^n-2}) = (\varphi^{n+1}_{-1},...,\varphi^{n+1}_{2^{n+1}-1}) \underbrace{\left.\begin{pmatrix} b_0 & & & & & & \\ c_0 & a & & & & & \\ d_0 & b & & & & & \\ e_0 & c & a & & & & \\ & d & b & \ddots & & & \\ & e & c & \ddots & & & \\ & & d & \ddots & & & \\ & & e & \ddots & a & & \\ & & & & b & & \\ & & & & c & a_1 & \\ & & & & d & b_1 & \\ & & & & e & c_1 & \\ & & & & & d_1 & \end{pmatrix}\right\} 2^{n+1}+1 = \nu(n+1)}_{2^n=\omega(n)}$$

Hierdurch erhält man den Waveletraum $W_n = span\{\psi_{-1},...,\psi_{2^n-2}\}$. Das Wavelet ψ ist in Abbildung 17 für verschiedene Parameter Δ zu sehen.

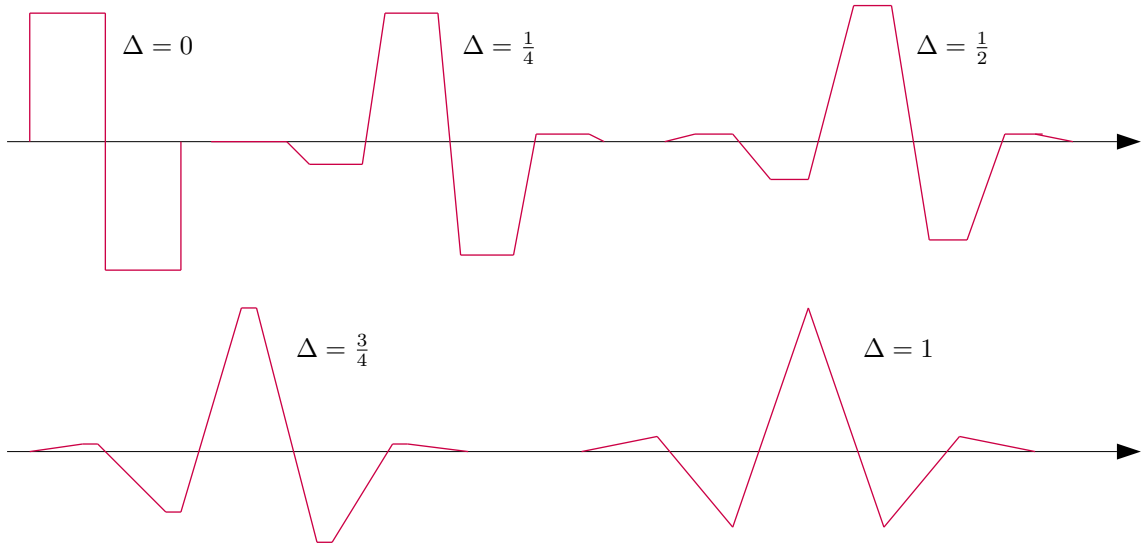


Abbildung 17: Das Wavelet für verschiedene Parameter Δ

4 Die Berechnung der Matrix Q und Implementierung

In diesem Kapitel soll genauer untersucht werden, wie das lineare Gleichungssystem zur Berechnung von Q aufgebaut ist, und wie es implementiert werden kann. Da wir für fünf Variablen nur vier Gleichungen zur Verfügung stehen haben, ist dies nicht auf üblichem Wege möglich. Vier Variablen müssen in Abhängigkeit einer Fünften gespeichert werden, um dies wiederum mit der zusätzlichen Bedingung in Verbindung zu bringen.
Anschliessend wird noch beschrieben wie die Glätte- und Detailkoeffizienten des nächst gröberen Levels effizient und stabil berechnet werden können.

4.1 Die Koeffizientenmatrix des LGS

Wie im vorigen Kapitel hergeleitet, muss das LGS $< \psi_i^n, \tilde{\varphi}_j^n >= 0 \quad , j = i-1, i, i+1, i+2$ gelöst werden. Da sowohl ψ_i^n, als auch $\tilde{\varphi}_j^n$ als Linearkombination der φ_k^{n+1} ausgedrückt werden können, muss hierfür nur das L^2-Skalarprodukt $< \varphi_i^{n+1}, \varphi_j^{n+1} >$ untersucht werden. Dies wurde in Kapitel 3.5 schon durchgeführt, es war

$$< \varphi_i^{n+1}, \varphi_j^{n+1} >= \begin{cases} 1 - \frac{\Delta}{3} & : \quad i = j \\ \frac{\Delta}{6} & : \quad i = j+1, i = j-1 \\ 0 & : \quad sonst \end{cases} .$$

Damit lässt sich jetzt das LGS $< \psi_i^n, \tilde{\varphi}_j^n >= 0 \quad , j = i-1, i, i+1, i+2$ berechnen. Man erinnere sich an die Definitionen:

$$\psi_i^n = a\varphi_{2i}^{n+1} + b\varphi_{2i+1}^{n+1} + c\varphi_{2i+2}^{n+1} + d\varphi_{2i+3}^{n+1} + e\varphi_{2i+4}^{n+1}$$
$$\tilde{\varphi}_j^n = \alpha\varphi_{2j}^{n+1} + \varphi_{2j+1}^{n+1} + (1-\alpha)\varphi_{2j+2}^{n+1}$$

Hiermit lässt sich nun das LGS genauer bestimmen. Im Folgenden wird der Übersicht wegen der obere Index *n+1* weggelassen, und $1 - \frac{\Delta}{3} =: c_\Delta$ abgekürzt.

$$\begin{aligned} j = i-1 : & < \psi_i^n, \tilde{\varphi}_{i-1}^n > \\ = \; & a[\alpha \underbrace{< \varphi_{2i}, \varphi_{2i-2} >}_{0} + \underbrace{< \varphi_{2i}, \varphi_{2i-1} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}} + (1-\alpha) \underbrace{< \varphi_{2i}, \varphi_{2i} >}_{\frac{1}{2^{n+1}}c_\Delta}] \\ & + b[\alpha \underbrace{< \varphi_{2i+1}, \varphi_{2i-2} >}_{0} + \underbrace{< \varphi_{2i+1}, \varphi_{2i-1} >}_{0} + (1-\alpha) \underbrace{< \varphi_{2i+1}, \varphi_{2i} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\ = \; & \frac{1}{2^{n+1}}[a(\frac{\Delta}{6} + c_\Delta(1-\alpha)) + b(\frac{\Delta}{6}(1-\alpha))] \\ = \; & 0 \end{aligned}$$

$$\begin{aligned} j = i : & < \psi_i^n, \tilde{\varphi}_i^n > \\ = \; & a[\alpha \underbrace{< \varphi_{2i}, \varphi_{2i} >}_{\frac{1}{2^{n+1}}c_\Delta} + \underbrace{< \varphi_{2i}, \varphi_{2i+1} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\ & + b[\alpha \underbrace{< \varphi_{2i+1}, \varphi_{2i} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}} + \underbrace{< \varphi_{2i+1}, \varphi_{2i+1} >}_{\frac{1}{2^{n+1}}c_\Delta} + (1-\alpha) \underbrace{< \varphi_{2i+1}, \varphi_{2i+2} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \end{aligned}$$

$$
\begin{aligned}
&+c[\underbrace{<\varphi_{2i+2},\varphi_{2i+1}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+(1-\alpha)\underbrace{<\varphi_{2i+2},\varphi_{2i+2}>}_{\frac{1}{2^{n+1}}c_\Delta}]+d[(1-\alpha)\underbrace{<\varphi_{2i+2},\varphi_{2i+3}>}_{\frac{1}{2^{n+1}}}]\\
&= \frac{1}{2^{n+1}}[a(\alpha c_\Delta+\frac{\Delta}{6})+b(\frac{\Delta}{6}+c_\Delta+(1-\alpha)\frac{\Delta}{6})+c(\frac{\Delta}{6}+c_\Delta(1-\alpha))+d((1-\alpha)\frac{\Delta}{6})]\\
&= \frac{1}{2^{n+1}}[a(\alpha c_\Delta+\frac{\Delta}{6})+b(c_\Delta+\frac{\Delta}{6})+c(\frac{\Delta}{6}+c_\Delta(1-\alpha))+d((1-\alpha)\frac{\Delta}{6})]\\
&= 0
\end{aligned}
$$

$j=i+1: <\psi_i^n,\tilde{\varphi}_{i+1}^n>$

$$
\begin{aligned}
&= b[\alpha\underbrace{<\varphi_{2i+1},\varphi_{2i+2}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]+c[\alpha\underbrace{<\varphi_{2i+2},\varphi_{2i+2}>}_{\frac{1}{2^{n+1}}c_\Delta}+\underbrace{<\varphi_{2i+2},\varphi_{2i+3}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]\\
&\quad +d[\alpha\underbrace{<\varphi_{2i+3},\varphi_{2i+2}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+\underbrace{<\varphi_{2i+3},\varphi_{2i+3}>}_{\frac{1}{2^{n+1}}c_\Delta}+(1-\alpha)\underbrace{<\varphi_{2i+3},\varphi_{2i+4}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]\\
&\quad +e[\underbrace{<\varphi_{2i+4},\varphi_{2i+3}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+(1-\alpha)\underbrace{<\varphi_{2i+4},\varphi_{2i+4}>}_{\frac{1}{2^{n+1}}c_\Delta}]\\
&= \frac{1}{2^{n+1}}[b(\alpha\frac{\Delta}{6})+c(\alpha c_\Delta+\frac{\Delta}{6})+d(c_\Delta+\frac{\Delta}{6})+e(\frac{\Delta}{6}+c_\Delta(1-\alpha))]\\
&= 0
\end{aligned}
$$

$j=i+2: <\psi_i^n,\tilde{\varphi}_{i+2}^n>$

$$
\begin{aligned}
&= d[\alpha\underbrace{<\varphi_{2i+3},\varphi_{2i+4}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]+e[\alpha\underbrace{<\varphi_{2i+4},\varphi_{2i+4}>}_{\frac{1}{2^{n+1}}c_\Delta}+\underbrace{<\varphi_{2i+4},\varphi_{2i+5}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]\\
&= \frac{1}{2^{n+1}}[d(\alpha\frac{\Delta}{6})+e(\alpha c_\Delta+\frac{\Delta}{6})]\\
&= 0
\end{aligned}
$$

Es ergibt sich also:

$$
\frac{1}{2^{n+1}}\begin{pmatrix}
\frac{\Delta}{6}+(1-\alpha) & \frac{\Delta}{6}(1-\alpha) & 0 & 0 & 0\\
\alpha c_\Delta+\frac{\Delta}{6} & c_\Delta+\frac{\Delta}{6} & \frac{\Delta}{6}+c_\Delta(1-\alpha) & \frac{\Delta}{6}(1-\alpha) & 0\\
0 & \alpha\frac{\Delta}{6} & \alpha c_\Delta+\frac{\Delta}{6} & c_\Delta+\frac{\Delta}{6} & \frac{\Delta}{6}+c_\Delta(1-\alpha)\\
0 & 0 & 0 & \alpha\frac{\Delta}{6} & \alpha c_\Delta+\frac{\Delta}{6}
\end{pmatrix}\begin{pmatrix}a\\b\\c\\d\\e\end{pmatrix}=0
$$

Die Ränder müssen noch getrennt betrachtet werden. Für den linken Rand, d.h. $i=-1$, galt

$$\psi_{-1}^n=b_0\varphi_{-1}^{n+1}+c_0\varphi_0^{n+1}+d_0\varphi_1^{n+1}+e_0\varphi_2^{n+1}.$$

Damit ergibt sich aus der Bedingung $<\psi_{-1}^n,\tilde{\varphi}_j^n>=0 \quad ,j=-1,0,1:$

$j=-1: <b_0\varphi_{-1}^{n+1}+c_0\varphi_0^{n+1}+d_0\varphi_1^{n+1}+e_0\varphi_2^{n+1},\alpha\varphi_{-2}^{n+1}+\varphi_{-1}^{n+1}+(1-\alpha)\varphi_0^{n+1}>$

$$
= b_0[\alpha\underbrace{<\varphi_{-1},\varphi_{-2}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+\underbrace{<\varphi_{-1},\varphi_{-1}>}_{\frac{1}{2^{n+1}}c_\Delta}+(1-\alpha)\underbrace{<\varphi_{-1},\varphi_0>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]
$$

$$
\begin{aligned}
& c_0[\underbrace{<\varphi_0,\varphi_{-1}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+(1-\alpha)\underbrace{<\varphi_0,\varphi_0>}_{\frac{1}{2^{n+1}}c_\Delta}]+d_0[(1-\alpha)\underbrace{<\varphi_1,\varphi_0>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\
= \;& \frac{1}{2^{n+1}}[b_0(c_\Delta+\frac{\Delta}{6})+c_0(\frac{\Delta}{6}+c_\Delta(1-\alpha))+d_0(\frac{\Delta}{6}(1-\alpha))] \\
= \;& 0
\end{aligned}
$$

$$
\begin{aligned}
j=0: & <b_0\varphi_{-1}^{n+1}+c_0\varphi_0^{n+1}+d_0\varphi_1^{n+1}+e_0\varphi_2^{n+1},\alpha\varphi_0^{n+1}+\varphi_1^{n+1}+(1-\alpha)\varphi_2^{n+1}> \\
= \;& b_0[\alpha\underbrace{<\varphi_{-1},\varphi_0>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]+c_0[\alpha\underbrace{<\varphi_0,\varphi_0>}_{\frac{1}{2^{n+1}}c_\Delta}+\underbrace{<\varphi_0,\varphi_1>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]+ \\
& d_0[\alpha\underbrace{<\varphi_1,\varphi_0>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+\underbrace{<\varphi_1,\varphi_1>}_{\frac{1}{2^{n+1}}c_\Delta}+(1-\alpha)\underbrace{<\varphi_1,\varphi_2>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\
& +e_0[\underbrace{<\varphi_2,\varphi_1>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+(1-\alpha)\underbrace{<\varphi_2,\varphi_2>}_{\frac{1}{2^{n+1}}c_\Delta}] \\
= \;& \frac{1}{2^{n+1}}[b_0(\alpha\frac{\Delta}{6})+c_0(\alpha c_\Delta+\frac{\Delta}{6})+d_0(c_\Delta+\frac{\Delta}{6})+e_0(\frac{\Delta}{6}+c_\Delta(1-\alpha))] \\
= \;& 0
\end{aligned}
$$

$$
\begin{aligned}
j=1: & <b_0\varphi_{-1}^{n+1}+c_0\varphi_0^{n+1}+d_0\varphi_1^{n+1}+e_0\varphi_2^{n+1},\alpha\varphi_2^{n+1}+\varphi_3^{n+1}+(1-\alpha)\varphi_4^{n+1}> \\
= \;& d_0[\alpha\underbrace{<\varphi_1,\varphi_2>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]+e_0[\alpha\underbrace{<\varphi_2,\varphi_2>}_{\frac{1}{2^{n+1}}c_\Delta}+\underbrace{<\varphi_2,\varphi_3>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]+ \\
= \;& \frac{1}{2^{n+1}}[d_0(\alpha\frac{\Delta}{6})+e_0(\alpha c_\Delta+\frac{\Delta}{6})] \\
= \;& 0
\end{aligned}
$$

Damit ergibt sich:

$$
\frac{1}{2^{n+1}}\begin{pmatrix} c_\Delta+\frac{\Delta}{6} & \frac{\Delta}{6}+c_\Delta(1-\alpha) & \frac{\Delta}{6}(1-\alpha) & 0 \\ \alpha\frac{\Delta}{6} & \alpha c_\Delta+\frac{\Delta}{6} & c_\Delta+\frac{\Delta}{6} & \frac{\Delta}{6}+c_\Delta(1-\alpha) \\ 0 & 0 & \alpha\frac{\Delta}{6} & \alpha c_\Delta+\frac{\Delta}{6} \end{pmatrix}\begin{pmatrix} b_0 \\ c_0 \\ d_0 \\ e_0 \end{pmatrix}=0
$$

Für den rechten Rand, d.h. $i=2^n-2$, galt

$$
\psi_{2^n-2}^n=a_1\varphi_{2^{n+1}-4}^{n+1}+b_1\varphi_{2^{n+1}-3}^{n+1}+c_1\varphi_{2^{n+1}-2}^{n+1}+d_1\varphi_{2^{n+1}-1}^{n+1}.
$$

Damit ergibt sich aus der Bedingung $<\psi_{2^n-2}^n,\tilde{\varphi}_j^n>=0 \quad ,j=2^n-3,2^n-2,2^n-1$:

$$
\begin{aligned}
j=2^n-3: & <a_1\varphi_{2^{n+1}-4}^{n+1}+b_1\varphi_{2^{n+1}-3}^{n+1}+c_1\varphi_{2^{n+1}-2}^{n+1}+d_1\varphi_{2^{n+1}-1}^{n+1}, \\
& \alpha\varphi_{2^{n+1}-6}^{n+1}+\varphi_{2^{n+1}-5}^{n+1}+(1-\alpha)\varphi_{2^{n+1}-4}^{n+1}> \\
= \;& a_1[\alpha\underbrace{<\varphi_{2^{n+1}-6}^{n+1},\varphi_{2^{n+1}-4}^{n+1}>}_{0}+\underbrace{<\varphi_{2^{n+1}-4}^{n+1},\varphi_{2^{n+1}-5}^{n+1}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}+(1-\alpha)\underbrace{<\varphi_{2^{n+1}-4}^{n+1},\varphi_{2^{n+1}-4}^{n+1}>}_{\frac{1}{2^{n+1}}c_\Delta}] \\
& +b_1[(1-\alpha)\underbrace{<\varphi_{2^{n+1}-3}^{n+1},\varphi_{2^{n+1}-4}^{n+1}>}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}]
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{2^{n+1}}[a_1(\frac{\Delta}{6} + c_\Delta(1-\alpha)) + b_1(1-\alpha)\frac{\Delta}{6}] \\
&= 0
\end{aligned}
$$

$j = 2^n - 2$: $< a_1\varphi^{n+1}_{2^{n+1}-4} + b_1\varphi^{n+1}_{2^{n+1}-3} + c_1\varphi^{n+1}_{2^{n+1}-2} + d_1\varphi^{n+1}_{2^{n+1}-1},$

$$
\begin{aligned}
&\alpha\varphi^{n+1}_{2^{n+1}-4} + \varphi^{n+1}_{2^{n+1}-3} + (1-\alpha)\varphi^{n+1}_{2^{n+1}-2} > \\
= \;& a_1[\alpha\underbrace{< \varphi^{n+1}_{2^{n+1}-4}, \varphi^{n+1}_{2^{n+1}-4} >}_{\frac{1}{2^{n+1}}c_\Delta} + \underbrace{< \varphi^{n+1}_{2^{n+1}-4}, \varphi^{n+1}_{2^{n+1}-3} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\
& + b_1[\alpha\underbrace{< \varphi^{n+1}_{2^{n+1}-4}, \varphi^{n+1}_{2^{n+1}-3} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}} + \underbrace{< \varphi^{n+1}_{2^{n+1}-3}, \varphi^{n+1}_{2^{n+1}-3} >}_{\frac{1}{2^{n+1}}c_\Delta} + (1-\alpha)\underbrace{< \varphi^{n+1}_{2^{n+1}-3}, \varphi^{n+1}_{2^{n+1}-2} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\
& + c_1[\underbrace{< \varphi^{n+1}_{2^{n+1}-3}, \varphi^{n+1}_{2^{n+1}-2} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}} + (1-\alpha)\underbrace{< \varphi^{n+1}_{2^{n+1}-2}, \varphi^{n+1}_{2^{n+1}-2} >}_{\frac{1}{2^{n+1}}c_\Delta}] \\
& + d_1[(1-\alpha)\underbrace{< \varphi^{n+1}_{2^{n+1}-1}, \varphi^{n+1}_{2^{n+1}-2} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\
= \;& \frac{1}{2^{n+1}}[a_1(\alpha c_\Delta + \frac{\Delta}{6}) + b_1(c_\Delta + \frac{\Delta}{6}) + c_1(\frac{\Delta}{6} + c_\Delta(1-\alpha)) + d_1((1-\alpha)\frac{\Delta}{6})] \\
= \;& 0
\end{aligned}
$$

$j = 2^n - 1$: $< a_1\varphi^{n+1}_{2^{n+1}-4} + b_1\varphi^{n+1}_{2^{n+1}-3} + c_1\varphi^{n+1}_{2^{n+1}-2} + d_1\varphi^{n+1}_{2^{n+1}-1},$

$$
\begin{aligned}
&\alpha\varphi^{n+1}_{2^{n+1}-2} + \varphi^{n+1}_{2^{n+1}-1} + (1-\alpha)\varphi^{n+1}_{2^{n+1}} > \\
= \;& b_1[\alpha\underbrace{< \varphi^{n+1}_{2^{n+1}-3}, \varphi^{n+1}_{2^{n+1}-2} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}} + c_1[\alpha\underbrace{< \varphi^{n+1}_{2^{n+1}-2}, \varphi^{n+1}_{2^{n+1}-2} >}_{\frac{1}{2^{n+1}}c_\Delta} + \underbrace{< \varphi^{n+1}_{2^{n+1}-2}, \varphi^{n+1}_{2^{n+1}-1} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\
& d_1[\alpha\underbrace{< \varphi^{n+1}_{2^{n+1}-1}, \varphi^{n+1}_{2^{n+1}-2} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}} + \underbrace{< \varphi^{n+1}_{2^{n+1}-1}, \varphi^{n+1}_{2^{n+1}-1} >}_{\frac{1}{2^{n+1}}c_\Delta} + (1-\alpha)\underbrace{< \varphi^{n+1}_{2^{n+1}-1}, \varphi^{n+1}_{2^{n+1}} >}_{\frac{1}{2^{n+1}}\frac{\Delta}{6}}] \\
= \;& \frac{1}{2^{n+1}}[b_1(\alpha\frac{\Delta}{6}) + c_1(\alpha c_\Delta + \frac{\Delta}{6}) + d_1(c_\Delta + \frac{\Delta}{6})] \\
= \;& 0
\end{aligned}
$$

Für den rechten Rand ergibt sich somit:

$$
\frac{1}{2^{n+1}}\begin{pmatrix} \frac{\Delta}{6} + c_\Delta(1-\alpha) & (1-\alpha)\frac{\Delta}{6} & 0 & 0 \\ \alpha c_\Delta + \frac{\Delta}{6} & c_\Delta + \frac{\Delta}{6} & c_\Delta(1-\alpha) + \frac{\Delta}{6} & (1-\alpha)\frac{\Delta}{6} \\ 0 & \alpha\frac{\Delta}{6} & \alpha c_\Delta + \frac{\Delta}{6} & c_\Delta + \frac{\Delta}{6} \end{pmatrix}\begin{pmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \end{pmatrix} = 0
$$

4.2 Berücksichtigung der fehlenden Gleichung

Durch oben angeführte Rechnungen erhält man somit drei lineare Gleichungssysteme $Ax = 0$ mit jeweils einer Gleichung weniger als gesuchten Variablen. Diese sollen mit der zusätzlichen Bedingung

$$||\psi|| = 1$$

gelöst werden. Wie lässt sich das nun implementieren? Ich habe im Folgenden genau den Ansatz umgesetzt, den man auch „per Hand“ wählen würde: Ist ein $n-1 \times n$ Gleichungssystem gegeben, so drückt man $n-1$ Variablen in Abhängigkeit der n-ten aus:

$$x_i = k_i x_n \quad i = 1, .., n-1, k_i \in \mathbb{R}$$

Dies setzt man in die zusätzliche Bedingung ein, erhält einen Wert für x_n und damit auch direkt für x_i , $i = 1, ..., n-1$. Ist die Matrix A' gegeben, so wird sie zunächst wie folgt umgeformt:

$$A' = \begin{pmatrix} a'_{1,1} & \cdots & a'_{1,n} \\ \vdots & \ddots & \vdots \\ a'_{n-1,1} & \cdots & a'_{n-1,n} \end{pmatrix} \longrightarrow \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} \\ & a_{2,2} & \cdots & a_{2,n-1} & a_{2,n} \\ & & \ddots & \vdots & \vdots \\ 0 & & & a_{n-1,n-1} & a_{n-1,n} \end{pmatrix} = A$$

Damit ergibt sich:

$$\begin{aligned} & a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_{n,n} = 0 \\ \Rightarrow \quad & x_{n-1} = -\frac{a_{n-1,n}}{a_{n-1,n-1}} x_{n,n} \\ \Rightarrow \quad & k_{n-1} = -\frac{a_{n-1,n}}{a_{n-1,n-1}} \\ & a_{n-2,n-2}x_{n-2} + a_{n-2,n-1} \underbrace{x_{n-1}}_{k_{n-1}x_n} + a_{n-2,n}x_{n,n} = 0 \\ \Rightarrow \quad & x_{n-2} = -\frac{a_{n-2,n-1}k_{n-1} + a_{n-2,n}}{a_{n,n-1}} x_{n,n} \\ \Rightarrow \quad & k_{n-2} = -\frac{a_{n-2,n-1}k_{n-1} + a_{n-2,n}}{a_{n,n-1}} \end{aligned}$$

Allgemein gilt:

$$k_i = -\frac{\sum_{j=i+2}^{n-1} a_{i+1,j}k_{j-1} + a_{i+1,n}}{a_{i+1,i+1}}$$

Damit lassen sich über $x_i = k_i x_n$ $i = 1, .., n-1$ die restlichen x_i einfach berechnen. Die „Dreiecksgestalt“ einer Matrix A lässt sich mit folgendem Algorithmus in Pseudocode einfach erreichen:

```
{n-2 Zeilen durchgehen}
for r=1 to n-1
    {unterhalb der i-ten Zeile in den ersten i-1 Spalten Nullen erzeugen}
    for i=r+1 to n-2
        {die ganze Zeile durchgehen}
        for j=r to n
            {a_{r,r} ist bei uns immer ungleich 0}
            a_{i,j} = a_{i,j} - a_{i-1,j} * a_{i,r} / a_{i-1,r}
```

```
            end
        end
    end
```

4.3 Berechnung von $||\psi||^2$

Im Folgenden wird nun $||\psi||^2$ berechnet, um aus der Bedingung $||\psi||^2 = 1$ den fehlenden Koeffizient e zu berechnen. $\psi = \psi_0 = a\varphi_0^1 + b\varphi_1^1 + c\varphi_2^1 + d\varphi_3^1 + e\varphi_4^1$ ist eine Linearkombination der φ_j^1.

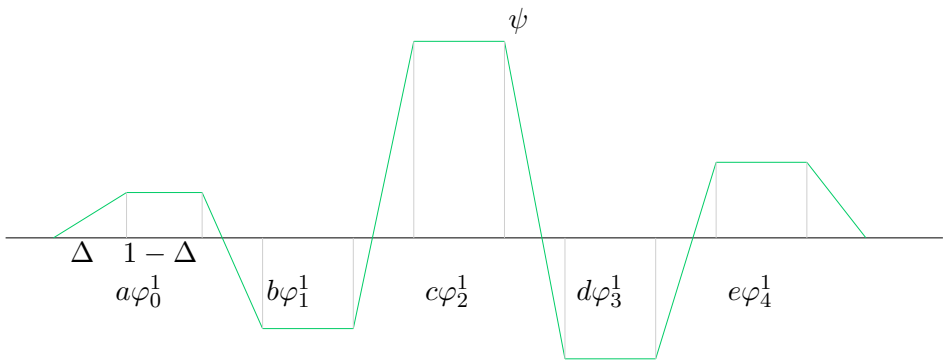


Abbildung 18: ψ als Linearkombination der φ_j^1

Es gilt:

$$||\psi||^2 = \int_{-\infty}^{\infty} |\psi(x)|^2 dx = \int_{0}^{5+\Delta} |\psi(x)|^2 dx$$

Um diesen Ausdruck rechnerisch bestimmen zu können, wird der Integrationsweg nun so unterteilt, dass auf den einzelnen Integrationsstücken die Funktion ψ entweder konstant oder linear ist. ψ^2 ist dort entweder auch konstant oder ein Parabelstück. Hierfür muss der Weg nicht nur an den Punkten unterteilt werden, an dem die Funktion von konstantem zum linearen Verlauf oder umgekehrt wechselt, sondern auch an den Nullstellen. Denn dort beginnt jeweils ein neues Parabelstück.

Die Nullstellen lassen sich wie folgt berechnen: Die Länge des Trägers des j-ten Parabelstückes sei mit g_j bezeichnet (s. Abbildung 19). Im ursprünglichen Wavelet ψ berühren sich in einer Nullstelle zwei ähnliche Dreiecke, eines mit einer Seite der Länge g_j, das andere mit einer Seite der Länge g_{j+1}. Die Höhe dieser Dreiecke entspricht dem Koeffizienten der jeweiligen Skalierungsfunktion (s. Abbildung 20 , dort mit k und l bezeichnet). Es gilt:

$$g_j + g_{j+1} = \Delta$$

und nach dem Strahlensatz:

$$\frac{k}{l} = \frac{g_j}{g_{j+1}}$$

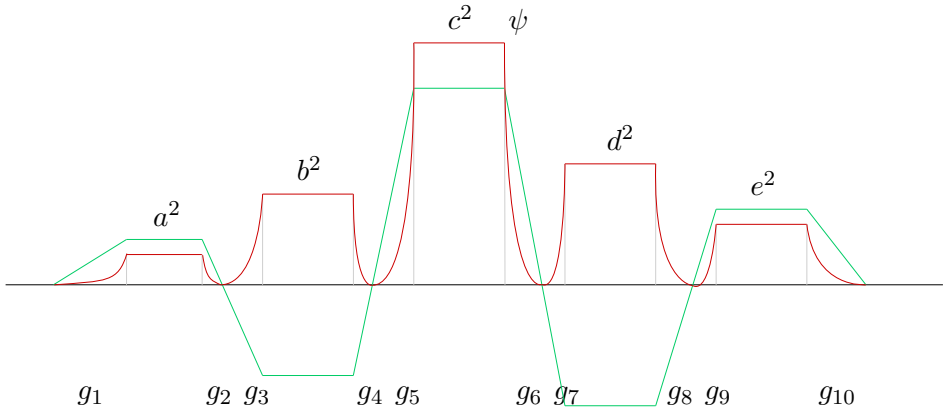


Abbildung 19: $|\psi|^2$ wird auch an den Nullstellen unterteilt

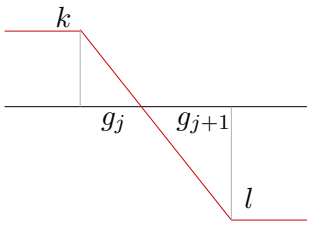


Abbildung 20: Berechnung der Länge von g_j

Aus diesen Bedingungen ergibt sich:

$$\begin{aligned} & g_j = \frac{kg_{j+1}}{l} = \frac{k(\Delta - g_j)}{l} = \frac{k\Delta}{l} - \frac{kg_j}{l} \\ \Rightarrow \quad & g_j + \frac{k}{l} g_j = \frac{k\Delta}{l} \\ \Rightarrow \quad & g_j = \frac{\frac{k\Delta}{l}}{1 + \frac{k}{l}} = \frac{\frac{k\Delta}{l}}{\frac{k+l}{l}} = \frac{k\Delta l}{l(k+l)} = \frac{k\Delta}{k+l} \end{aligned}$$

Und damit folgt für das Integral:

$$\begin{aligned} \int_0^{5+\Delta} |\psi|^2 \quad = \quad & \int_0^{\Delta} |\psi|^2 + \int_{\Delta}^{1} |\psi|^2 + \int_{1}^{1+g_2} |\psi|^2 + \int_{1+g_2}^{1+\Delta} |\psi|^2 + \dots \\ & \dots + \int_{4+g_8}^{4+\Delta} |\psi|^2 + \int_{4+\Delta}^{5} |\psi|^2 + \int_{5}^{5+\Delta} |\psi|^2 \end{aligned}$$

Die Integrale über konstante Funktionen sind einfache Rechtecke und lassen sich direkt bestimmen:

$$
\begin{aligned}
A_{\square_1} &= (1-\Delta)a^2 \\
A_{\square_2} &= (1-\Delta)b^2 \\
A_{\square_3} &= (1-\Delta)c^2 \\
A_{\square_4} &= (1-\Delta)d^2 \\
A_{\square_5} &= (1-\Delta)e^2
\end{aligned}
$$

Für die Berechnung der Integrale über die Parabelstücke werden diese nur monoton wachsend und bei Null beginnend betrachtet, d.h. das Parabelstück über dem Träger g_j beginnt bei Null und endet bei g_j. Dort hat die Parabel die Höhe k^2 erreicht. Das Integral hat also folgende Gestalt:

$$
\int_0^{g_j} (x\frac{k}{g_j})^2 dx = \int_0^{g_j} x^2 \frac{k^2}{{g_j}^2} dx = \frac{1}{3}\, x^3 \frac{k^2}{{g_j}^2}\Big|_0^{g_j} = \frac{1}{3} g_j k^2 = \frac{1}{3}\frac{k\Delta}{2(k+l)}k^2 = \frac{1}{2}\frac{1}{3}\frac{k^3\Delta}{k+l}
$$

Damit ergeben sich die zehn Parabelintegrale:

$$
\begin{aligned}
A_{P1} &= \frac{1}{2}\frac{\Delta}{3}a^2 \\
A_{P2} &= \frac{1}{2}\frac{1}{3}\frac{a^3\Delta}{a+b} \\
A_{P3} &= \frac{1}{2}\frac{1}{3}\frac{b^3\Delta}{a+b} \\
A_{P4} &= \frac{1}{2}\frac{1}{3}\frac{b^3\Delta}{b+c} \\
A_{P5} &= \frac{1}{2}\frac{1}{3}\frac{c^3\Delta}{b+c} \\
A_{P6} &= \frac{1}{2}\frac{1}{3}\frac{c^3\Delta}{c+d} \\
A_{P7} &= \frac{1}{2}\frac{1}{3}\frac{d^3\Delta}{c+d} \\
A_{P8} &= \frac{1}{2}\frac{1}{3}\frac{d^3\Delta}{d+e} \\
A_{P9} &= \frac{1}{2}\frac{1}{3}\frac{e^3\Delta}{d+e} \\
A_{P10} &= \frac{1}{2}\frac{\Delta}{3}e^2
\end{aligned}
$$

Damit lässt sich jetzt endlich das Integral in Abhängigkeit der Koeffizienten der Skalierungsfunktionen ausdrücken. Darüber kann dann e bestimmt werden.

$$
1 \quad = \quad \int |\psi|^2
$$

$$
\begin{aligned}
&= \frac{1}{2}\{(1-\Delta)[a^2+b^2+c^2+d^2+e^2] \\
&\quad +\frac{\Delta}{3}[a^2+\frac{a^3}{a+b}+\frac{b^3}{a+b}+\frac{b^3}{b+c}+\frac{c^3}{b+c}+\frac{c^3}{c+d} \\
&\quad +\frac{d^3}{c+d}+\frac{d^3}{c+d}+\frac{d^3}{d+e}+\frac{e^3}{d+e}+e^2]\} \\
&= \frac{1}{2}\{(1-\Delta)[((k_0e)^2+(k_1e)^2+(k_2e)^2+(k_3e)^2+e^2] \\
&\quad +\frac{\Delta}{3}[(k_0e)^2+\frac{(k_0e)^3}{(k_0e)+(k_1e)}+\frac{(k_1e)^3}{(k_0e)+(k_1e)}+\frac{(k_1e)^3}{(k_1e)+(k_2e)}+\frac{(k_2e)^3}{(k_1e)+(k_2e)} \\
&\quad +\frac{(k_2e)^3}{(k_2e)+(k_3e)}+\frac{(k_3e)^3}{(k_2e)+(k_3e)}+\frac{(k_3e)^3}{(k_3e)+e}+\frac{e^3}{(k_3e)+e}+e^2]\} \\
&= \frac{e^2}{2}\{(1-\Delta)[{k_0}^2+{k_1}^2+{k_2}^2+{k_3}^2+1] \\
&\quad +\frac{\Delta}{3}[{k_0}^2+\frac{{k_0}^3}{k_0+k_1}+\frac{{k_1}^3}{k_0+k_1}+\frac{{k_1}^3}{k_1+k_2}+\frac{{k_2}^3}{k_1+k_2} \\
&\quad +\frac{{k_2}^3}{k_2+k_3}+\frac{{k_3}^3}{k_2+k_3}+\frac{{k_3}^3}{k_3+1}+\frac{e^3}{k_3+1}+1]\} \\
&= \frac{e^2}{2}\{(1-\Delta)[{k_0}^2+{k_1}^2+{k_2}^2+{k_3}^2+1] \\
&\quad +\frac{\Delta}{3}[{k_0}^2+\frac{{k_0}^3+{k_1}^3}{k_0+k_1}+\frac{{k_1}^3+{k_2}^3}{k_1+k_2}+\frac{{k_2}^3+{k_3}^3}{k_2+k_3}+\frac{{k_3}^3+1}{k_3+1}+1]\} \\
&=: \frac{e^2}{2}S
\end{aligned}
$$

$$\Longrightarrow e=\sqrt{\frac{2}{S}}$$

Wir wählen hier nur die positive Lösung, um die in der Literatur gängigen Haar- und linearen Wavelets zu erhalten.

Spezialfälle ergeben sich für

1) $\psi_{-1}^n = b_0\varphi_{-1}^{n+1}+c_0\varphi_0^{n+1}+d_0\varphi_1^{n+1}+e_0\varphi_2^{n+1}$
2) $\psi_{2^n-2}^n = a_1\varphi_{2^{n+1}-4}^{n+1}+b_1\varphi_{2^{n+1}-3}^{n+1}+c_1\varphi_{2^{n+1}-2}^{n+1}+d_1\varphi_{2^{n+1}-1}^{n+1}$.

Die Berechnung erfolgt jedoch völlig analog, es ergibt sich im ersten Fall:

$$
\begin{aligned}
1 &= \int|\psi_{-1}|^2 \\
&= \frac{1}{2}\{(1-\Delta)[{b_0}^2+{c_0}^2+{d_0}^2+{e_0}^2] \\
&\quad +\frac{\Delta}{3}[{b_0}^2+\frac{{b_0}^3}{b_0+c_0}+\frac{{c_0}^3}{b_0+c_0}+\frac{{c_0}^3}{c_0+d_0}+\frac{{d_0}^3}{c_0+d_0}+\frac{{d_0}^3}{d_0+e_0}+\frac{{e_0}^3}{d_0+e_0}+{e_0}^2]\} \\
&= \frac{{e_0}^2}{2}\{(1-\Delta)[{k_0}^2+{k_1}^2+{k_2}^2+1]
\end{aligned}
$$

$$
\begin{aligned}
&+\frac{\Delta}{3}[{k_0}^2+\frac{{k_0}^3+{k_1}^3}{k_0+k_1}+\frac{{k_1}^3+{k_2}^3}{k_1+k_2}+\frac{{k_2}^3+1}{k_2+1}+1]\}\\
=:\quad &\frac{e_0^2}{2}S_1
\end{aligned}
$$

$$\Longrightarrow e_0=\sqrt{\frac{2}{S_1}}$$

Der zweite Fall sieht folgendermassen aus:

$$
\begin{aligned}
1 \quad = \quad & \int |\psi_{2^n-2}|^2\\
= \quad & \frac{1}{2}\{(1-\Delta)[{a_1}^2+{b_1}^2+{c_1}^2+{d_1}^2]\\
&+\frac{\Delta}{3}[{a_1}^2+\frac{{a_1}^3}{a_1+b_1}+\frac{{b_1}^3}{a_1+b_1}+\frac{{b_1}^3}{b_1+c_1}+\frac{{c_1}^3}{b_1+c_1}+\frac{{c_1}^3}{c_1+d_1}+\frac{{d_1}^3}{c_1+d_1}+{d_1}^2]\}\\
= \quad & \frac{{d_1}^2}{2}\{(1-\Delta)[{k_0}^2+{k_1}^2+{k_2}^2+1]\\
&+\frac{\Delta}{3}[{k_0}^2+\frac{{k_0}^3+{k_1}^3}{k_0+k_1}+\frac{{k_1}^3+{k_2}^3}{k_1+k_2}+\frac{{k_2}^3+1}{k_2+1}+1]\}\\
=: \quad & \frac{{d_1}^2}{2}S_2
\end{aligned}
$$

$$\Longrightarrow d_1=\sqrt{\frac{2}{S_2}}$$

Damit sind nun alle für die Wavelets benötigten Koeffizienten berechnet.

4.4 Effiziente Berechnung der Glätte- und Detailkoeffizienten

Die Glättekoeffizienten x^n und die Detailkoeffizienten y^n lassen sich aus den Koeffizienten x^{n+1} zum Beispiel mit folgender Formel berechnen:

$$x^n = A^n x^{n+1}; \qquad y^n = B^n x^{n+1}$$

Wie in Kapitel 2.2 gezeigt, lassen sich die hierfür benötigten Matrizen A und B durch P und Q folgendermassen bestimmen:

$$A = (P^T G_{n+1} P)^{-1} P^T G_{n+1}; \qquad B = (Q^T G_{n+1}^{-1} Q)^{-1} Q^T G_{n+1}^{-1}$$

Ich habe zuerst versucht diese Formeln zu implementieren. Dies stellte sich jedoch zum einen als sehr uneffizient heraus, die Berechnung der Matrizen vom Level 7 dauerte schon mehrere Minuten. Zum anderen war sie sehr instabil. Eine erheblich effizientere Methode stellt Bonneau in [2] vor:

Sei $f_{n+1} \in V_{n+1}$ eine gegebene Approximation einer Funktion f zum Level $n+1$:

$$f_{n+1} = \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} \varphi_i^{n+1}$$

Die Aufgabe ist es nun die Koeffizienten x_i^n und y_j^n der Darstellung

$$f_{n+1} = \sum_{i=-1}^{2^n-1} x_i^n \tilde{\varphi}_i^n + \sum_{j=-1}^{2^n-2} y_j^n \psi_j^n \tag{23}$$

zu finden. Um die Glättekoeffizienten x_i^n zu erhalten, wenden wir auf beide Darstellungen das Skalarprodukt mit der Funktion $\tilde{\varphi}_k^n$ für $k = -1, ..., 2^n - 1$ an. Da aufgrund der Semiorthogonalität die Wavelets ψ_j^n orthogonal auf die Skalierungsfunktionen $\tilde{\varphi}_k^n$ stehen, ergibt sich:

$$< \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} \varphi_i^{n+1}, \tilde{\varphi}_k^n > = < \sum_{i=-1}^{2^n-2} x_i^n \tilde{\varphi}_i^n, \tilde{\varphi}_k^n > + \underbrace{< \sum_{j=-1}^{2^n-2} y_j^n \psi_j^n, \tilde{\varphi}_k^n >}_{=0}$$

$$\iff \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \tilde{\varphi}_k^n > = \sum_{i=-1}^{2^n-2} x_i^n < \tilde{\varphi}_i^n, \tilde{\varphi}_k^n > \qquad , k = -1, ..., 2^n - 1$$

Man erhält also das Gleichungssystem

$$\begin{pmatrix} < \tilde{\varphi}_{-1}^n, \tilde{\varphi}_{-1}^n > & \cdots & < \tilde{\varphi}_{2^n-1}^n, \tilde{\varphi}_{-1}^n > \\ \vdots & & \vdots \\ < \tilde{\varphi}_{-1}^n, \tilde{\varphi}_{2^n-1}^n > & \cdots & < \tilde{\varphi}_{2^n-1}^n, \tilde{\varphi}_{2^n-1}^n > \end{pmatrix} \begin{pmatrix} x_{-1}^n \\ \vdots \\ x_{2^n-1}^n \end{pmatrix} = \begin{pmatrix} \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \tilde{\varphi}_{-1}^n > \\ \vdots \\ \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \tilde{\varphi}_{2^n-1}^n > \end{pmatrix}.$$

Um die Detailkoeffizienten zu berechnen, nimmt man auf beiden Seiten der Gleichung (23) das Skalarprodukt mit der Funktion ψ_k^n und erhält:

$$< \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1}\varphi_i^{n+1}, \psi_k^n >= \underbrace{< \sum_{i=-1}^{2^n-2} x_i^n \tilde{\varphi}_i^n, \psi_k^n >}_{=0} + < \sum_{j=-1}^{2^n-2} y_j^n \psi_j^n, \psi_k^n >$$

$$\iff \sum_{j=-1}^{2^n-2} y_j^n < \psi_j^n, \psi_k^n >= \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \psi_k^n > \qquad , k = -1, ..., 2^n - 2$$

Dies ergibt das Gleichungssystem

$$\begin{pmatrix} < \psi_{-1}^n, \psi_{-1}^n > & \cdots & < \psi_{2^n-2}^n, \psi_{-1}^n > \\ \vdots & & \vdots \\ < \psi_{-1}^n, \psi_{2^n-1}^n > & \cdots & < \psi_{2^n-2}^n, \psi_{2^n-2}^n > \end{pmatrix} \begin{pmatrix} y_{-1}^n \\ \vdots \\ y_{2^n-2}^n \end{pmatrix} = \begin{pmatrix} \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \psi_{-1}^n > \\ \vdots \\ \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \psi_{2^n-2}^n > \end{pmatrix}.$$

Es müssen also noch die folgenden Ausdrücke berechnet werden:

$a)$ $< \tilde{\varphi}_i^n, \tilde{\varphi}_j^n >$
$b)$ $< \varphi_i^{n+1}, \tilde{\varphi}_j^n >$
$c)$ $< \psi_i^n, \psi_j^n >$
$d)$ $< \varphi_i^{n+1}, \psi_j^n >$

Da sowohl die $\tilde{\varphi}_j^n$ als auch die ψ_i^n Linearkombinationen der φ_i^{n+1} sind, werden diese auf die Berechnung des Skalarproduktes $< \varphi_i^{n+1}, \varphi_j^{n+1} >$ zurückgeführt. Dieses Skalarprodukt wurde schon in 3.5 berechnet. Der Übersicht halber wird bei den folgenden Berechnungen der oben stehenden Index „n+1“ weggelassen.

$$\begin{aligned} a) \quad & < \tilde{\varphi}_i^n, \tilde{\varphi}_j^n > \\ = \quad & < \alpha\varphi_{2i} + \varphi_{2i+1} + (1-\alpha)\varphi_{2i+2}, \alpha\varphi_{2j} + \varphi_{2j+1} + (1-\alpha)\varphi_{2j+2} > \\ = \quad & \alpha^2 < \varphi_{2i}, \varphi_{2j} > +\alpha < \varphi_{2i}, \varphi_{2j+1} > +\alpha(1-\alpha) < \varphi_{2i}, \varphi_{2j+2} > +\alpha < \varphi_{2i+1}, \varphi_{2j} > \\ & + < \varphi_{2i+1}, \varphi_{2j+1} > +(1-\alpha) < \varphi_{2i+1}, \varphi_{2j+2} > +\alpha(1-\alpha) < \varphi_{2i+2}, \varphi_{2j} > \\ & +(1-\alpha) < \varphi_{2i+2}, \varphi_{2j+1} > +(1-\alpha)^2 < \varphi_{2i+2}, \varphi_{2j+2} > \end{aligned}$$

Diese Summe enthält jedoch nur Summanden ungleich Null für folgende Indizes:

$$\begin{aligned} j = i-1: \quad & Gm_{i,i-1} :=< \tilde{\varphi}_i^n, \tilde{\varphi}_{i-1}^n > \\ = \quad & \alpha < \varphi_{2i}, \varphi_{2i-1} > +\alpha(1-\alpha) < \varphi_{2i}, \varphi_{2i} > +(1-\alpha) < \varphi_{2i+1}, \varphi_{2i} > \\ = \quad & \frac{1}{2^{n+1}}\{\alpha\frac{\Delta}{6} + \alpha(1-\alpha)c_\Delta + (1-\alpha)\frac{\Delta}{6}\} \end{aligned}$$

$$
\begin{aligned}
= \quad & < \tilde{\varphi}_i^n, \tilde{\varphi}_{i+1}^n >=: Gm_{i,i+1} \\
\\
j = i: \quad & < \tilde{\varphi}_i^n, \tilde{\varphi}_i^n > \\
= \quad & \alpha^2 < \varphi_{2i}, \varphi_{2i} > +\alpha < \varphi_{2i}, \varphi_{2i+1} > +\alpha < \varphi_{2i+1}, \varphi_{2i} > + < \varphi_{2i+1}, \varphi_{2i+1} > \\
& +(1-\alpha) < \varphi_{2i+1}, \varphi_{2i+2} > +(1-\alpha) < \varphi_{2i+2}, \varphi_{2i+1} > \\
& +(1-\alpha)^2 < \varphi_{2i+2}, \varphi_{2i+2} > \\
= \quad & \frac{1}{2^{n+1}}\{\alpha^2 c_\Delta + \alpha\frac{\Delta}{6} + \alpha\frac{\Delta}{6} + c_\Delta + (1-\alpha)\frac{\Delta}{6} + (1-\alpha)\frac{\Delta}{6} + (1-\alpha)^2 c_\Delta\} \\
= \quad & \frac{1}{2^{n+1}}\{\alpha^2 + c_\Delta + (1-\alpha)^2 c_\Delta + \frac{\Delta}{3}(\alpha + (1-\alpha))\} \\
= \quad & \frac{1}{2^{n+1}}\{\alpha^2 + c_\Delta + (1-\alpha)^2 c_\Delta + \frac{\Delta}{3}\} =: Gm_{i,i}
\end{aligned}
$$

Der Rand muss wieder gesondert betrachtet werden, es war $\tilde{\varphi}_{-1}^n = \varphi_{-1}^{n+1} + (1-\alpha)\varphi_0^{n+1}$. Damit ergibt sich für den linken Rand:

$$
\begin{aligned}
Gm_{-1,-1} \quad := \quad & < \tilde{\varphi}_{-1}^n, \tilde{\varphi}_{-1}^n > \\
= \quad & < \varphi_{-1} + (1-\alpha)\varphi_0, \varphi_{-1} + (1-\alpha)\varphi_0 > \\
= \quad & < \varphi_{-1}, \varphi_{-1} > +2(1-\alpha) < \varphi_0, \varphi_{-1} > +(1-\alpha)^2 < \varphi_0, \varphi_0 > \\
= \quad & \frac{1}{2^{n+1}}\{c_\Delta + (1-\alpha)^2 c_\Delta + \frac{\Delta}{6} 2(1-\alpha)\} \\
= \quad & \frac{1}{2^{n+1}}\{c_\Delta + (1-\alpha)^2 c_\Delta + \frac{\Delta}{3}(1-\alpha)\}
\end{aligned}
$$

$$
\begin{aligned}
Gm_{-1,0} \quad := \quad & < \tilde{\varphi}_{-1}^n, \tilde{\varphi}_0^n > \\
= \quad & < \alpha\varphi_0 + \varphi_1 + (1-\alpha)\varphi_2, \varphi_{-1} + (1-\alpha)\varphi_0 > \\
= \quad & \alpha < \varphi_0, \varphi_{-1} > +\alpha(1-\alpha) < \varphi_0, \varphi_0 > +(1-\alpha) < \varphi_1, \varphi_0 > \\
= \quad & \frac{1}{2^{n+1}}\{\alpha\frac{\Delta}{6} + \alpha(1-\alpha)c_\Delta + (1-\alpha)\frac{\Delta}{6}\} \\
= \quad & \frac{1}{2^{n+1}}\{\alpha(1-\alpha)c_\Delta + \frac{\Delta}{6}\} =< \tilde{\varphi}_0^n, \tilde{\varphi}_{-1}^n >=: Gm_{0,-1}
\end{aligned}
$$

Ferner war $\tilde{\varphi}_{2^n-1}^n = \alpha\varphi_{2^{n+1}-2}^{n+1} + \varphi_{2^{n+1}-1}^{n+1}$, damit ergibt sich am rechten Rand folgende Situation:

$$
\begin{aligned}
Gm_{2^n-1,2^n-1} \quad := \quad & < \tilde{\varphi}_{2^n-1}^n, \tilde{\varphi}_{2^n-1}^n > \\
= \quad & < \alpha\varphi_{2^{n+1}-2} + \varphi_{2^{n+1}-1}, \alpha\varphi_{2^{n+1}-2} + \varphi_{2^{n+1}-1} > \\
= \quad & \alpha^2 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-2} > +2\alpha < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-1} > \\
& + < \varphi_{2^{n+1}-1}, \varphi_{2^{n+1}-1} > \\
= \quad & \frac{1}{2^{n+1}}\{\alpha^2 c_\Delta + \frac{\Delta}{3} + c_\Delta\}
\end{aligned}
$$

$$
Gm_{2^n-1,2^n-2} \quad := \quad < \tilde{\varphi}_{2^n-1}^n, \tilde{\varphi}_{2^n-2}^n >
$$

$$
\begin{aligned}
&= < \alpha\varphi_{2^{n+1}-2} + \varphi_{2^{n+1}-1}, \alpha\varphi_{2^{n+1}-4} + \varphi_{2^{n+1}-3} + (1-\alpha)\varphi_{2^{n+1}-2} > \\
&= \alpha < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-3} > + \alpha(1-\alpha) < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-2} > \\
&\quad + (1-\alpha) < \varphi_{2^{n+1}-1}, \varphi_{2^{n+1}-2} > \\
&= \frac{1}{2^{n+1}}\{\alpha\frac{\Delta}{6} + \alpha(1-\alpha)c_\Delta + (1-\alpha)\frac{\Delta}{6}\} \\
&= \frac{1}{2^{n+1}}\{\alpha(1-\alpha)c_\Delta + \frac{\Delta}{6}\} =< \tilde{\varphi}^n_{2^n-2}, \tilde{\varphi}^n_{2^n-1} >=: Gm_{2^n-2,2^n-1}
\end{aligned}
$$

$$
\begin{aligned}
&b) \quad < \varphi_i^{n+1}, \tilde{\varphi}_j^n > \\
&= < \varphi_i, \alpha\varphi_{2j} + \varphi_{2j+1} + (1-\alpha)\varphi_{2j+2} > \\
&= \alpha < \varphi_i, \varphi_{2j} > + < \varphi_i, \varphi_{2j+1} > + (1-\alpha) < \varphi_i, \varphi_{2j+2} >
\end{aligned}
$$

Damit ergibt sich für die Summe im Ergebnisvektor:

$$
\begin{aligned}
\Longrightarrow \quad & Ge_j := \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \tilde{\varphi}_j^n > \\
= \quad & \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1}(\alpha < \varphi_i, \varphi_{2j} > + < \varphi_i, \varphi_{2j+1} > + (1-\alpha) < \varphi_i, \varphi_{2j+2} >) \\
= \quad & x_{2j-1}^{n+1}(\alpha < \varphi_{2j-1}, \varphi_{2j} >) + x_{2j}^{n+1}(\alpha < \varphi_{2j}, \varphi_{2j} > + < \varphi_{2j}, \varphi_{2j+1} >) \\
& + x_{2j+1}^{n+1}(\alpha < \varphi_{2j+1}, \varphi_{2j} > + < \varphi_{2j+1}, \varphi_{2j+1} > + (1-\alpha) < \varphi_{2j+1}, \varphi_{2j+2} >) \\
& + x_{2j+2}^{n+1}(< \varphi_{2j+2}, \varphi_{2j+1} > + (1-\alpha) < \varphi_{2j+2}, \varphi_{2j+2} >) \\
& + x_{2j+3}^{n+1}((1-\alpha) < \varphi_{2j+3}, \varphi_{2j+2} >) \\
= \quad & \frac{1}{2^{n+1}}\{x_{2j-1}^{n+1}(\alpha\frac{\Delta}{6}) + x_{2j}^{n+1}(\alpha c_\Delta + \frac{\Delta}{6}) \\
& + x_{2j+1}^{n+1}(c_\Delta + \frac{\Delta}{6}) + x_{2j+2}^{n+1}(\frac{\Delta}{6} + (1-\alpha)c_\Delta) + x_{2j+3}^{n+1}((1-\alpha)\frac{\Delta}{6})\} \\
& \text{für } j = 0, ..., 2^n - 2
\end{aligned}
$$

Wiederum sind die Ränder Sonderfälle:

$$
\begin{aligned}
j = -1: \quad & Ge_{-1} := \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \tilde{\varphi}_{-1}^n > \\
= \quad & \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \varphi_{-1}^{n+1} + (1-\alpha)\varphi_0^{n+1} > \\
= \quad & \frac{1}{2^{n+1}}\{x_{-1}^{n+1}(c_\Delta + \frac{\Delta}{6}) + x_0^{n+1}(\frac{\Delta}{6} + (1-\alpha)c_\Delta) + x_1^{n+1}((1-\alpha)\frac{\Delta}{6})\}
\end{aligned}
$$

$$
\begin{aligned}
j = 2^n - 1: \quad Ge_{2^n-1} &:= \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \tilde{\varphi}_{2^n-1}^n > \\
&= \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \alpha\varphi_{2^{n+1}-2}^{n+1} + \varphi_{2^{n+1}-1}^{n+1} > \\
&= \frac{1}{2^{n+1}}\{x_{2^{n+1}-3}^{n+1}(\alpha\frac{\Delta}{6}) + x_{2^{n+1}-2}^{n+1}(\alpha c_\Delta + \frac{\Delta}{6}) + x_{2^{n+1}-1}^{n+1}(c_\Delta + \frac{\Delta}{6})\}
\end{aligned}
$$

Damit ist der Ergebnisvektor vollständig und das Gleichungssystem zur Bestimmung der Glättekoeffizienten hat folgende Gestalt:

$$
\begin{pmatrix}
Gm_{-1,-1} & Gm_{0,-1} & & & \\
Gm_{-1,0} & Gm_{0,0} & Gm_{1,0} & & 0 \\
 & Gm_{0,1} & Gm_{1,1} & & \\
 & & & \ddots & \\
 & 0 & & Gm_{2^n-2,2^n-2} & Gm_{2^n-1,2^n-2} \\
 & & & Gm_{2^n-2,2^n-1} & Gm_{2^n-1,2^n-1}
\end{pmatrix}
\begin{pmatrix} x_{-1}^n \\ \vdots \\ x_{2^n-1}^n \end{pmatrix} =
\begin{pmatrix} Ge_{-1} \\ \vdots \\ Ge_{2^n-1} \end{pmatrix}
$$

Schliesslich werden noch die Fälle *c)* und *d)* für das Gleichungssystem zur Berechnung der Detailkoeffizienten betrachtet:

$$
\begin{aligned}
&c) \quad < \psi_i^n, \psi_j^n > \\
&= \quad < a\varphi_{2i} + b\varphi_{2i+1} + c\varphi_{2i+2} + d\varphi_{2i+3} + e\varphi_{2i+4}, \\
&\quad\; a\varphi_{2j} + b\varphi_{2j+1} + c\varphi_{2j+2} + d\varphi_{2j+3} + e\varphi_{2j+4} > \\
&= \quad a < \varphi_{2i}, a\varphi_{2j} + b\varphi_{2j+1} + c\varphi_{2j+2} + d\varphi_{2j+3} + e\varphi_{2j+4} > \\
&\quad\; +b < \varphi_{2i+1}, a\varphi_{2j} + b\varphi_{2j+1} + c\varphi_{2j+2} + d\varphi_{2j+3} + e\varphi_{2j+4} > \\
&\quad\; +c < \varphi_{2i+2}, a\varphi_{2j} + b\varphi_{2j+1} + c\varphi_{2j+2} + d\varphi_{2j+3} + e\varphi_{2j+4} > \\
&\quad\; +d < \varphi_{2i+3}, a\varphi_{2j} + b\varphi_{2j+1} + c\varphi_{2j+2} + d\varphi_{2j+3} + e\varphi_{2j+4} > \\
&\quad\; +e < \varphi_{2i+4}, a\varphi_{2j} + b\varphi_{2j+1} + c\varphi_{2j+2} + d\varphi_{2j+3} + e\varphi_{2j+4} >
\end{aligned}
$$

Ungleich Null sind hierbei jedoch nur folgende Summanden:

$$
\begin{aligned}
j = i - 1: \quad Dm_{i,i-1} &:= < \psi_i^n, \psi_{i-1}^n > \\
&= ab < \varphi_{2i}, \varphi_{2i-1} > + ac < \varphi_{2i}, \varphi_{2i} > + ad < \varphi_{2i}, \varphi_{2i+1} > \\
&\quad\; +bc < \varphi_{2i+1}, \varphi_{2i} > + bd < \varphi_{2i+1}, \varphi_{2i+1} > + be < \varphi_{2i+1}, \varphi_{2i+2} > \\
&\quad\; +cd < \varphi_{2i+2}, \varphi_{2i+1} > + ce < \varphi_{2i+2}, \varphi_{2i+2} > \\
&\quad\; +de < \varphi_{2i+3}, \varphi_{2i+2} > \\
&= \frac{1}{2^{n+1}}\{c_\Delta(ac + bd + ce) + \frac{\Delta}{6}(ab + ad + bc + be + cd + de)\} \\
&= < \psi_i^n, \psi_{i+1}^n > =: Dm_{i,i+1}
\end{aligned}
$$

$$
j = i: \quad Dm_{i,i} := < \psi_i^n, \psi_i^n >
$$

$$
\begin{aligned}
= \quad & a^2 < \varphi_{2i}, \varphi_{2i} > + ab < \varphi_{2i}, \varphi_{2i+1} > \\
& + ba < \varphi_{2i+1}, \varphi_{2i} > + b^2 < \varphi_{2i+1}, \varphi_{2i+1} > + bc < \varphi_{2i+1}, \varphi_{2i+2} > \\
& + cb < \varphi_{2i+2}, \varphi_{2i+1} > + c^2 < \varphi_{2i+2}, \varphi_{2i+2} > + cd < \varphi_{2i+2}, \varphi_{2i+3} > \\
& + dc < \varphi_{2i+3}, \varphi_{2i+2} > + d^2 < \varphi_{2i+3}, \varphi_{2i+3} > + de < \varphi_{2i+3}, \varphi_{2i+4} > \\
& + ed < \varphi_{2i+4}, \varphi_{2i+3} > + e^2 < \varphi_{2i+4}, \varphi_{2i+4} > \\
= \quad & \frac{1}{2^{n+1}} \{ c_\Delta (a^2 + b^2 + c^2 + d^2 + e^2) + \frac{\Delta}{3} (ab + bc + cd + de) \}
\end{aligned}
$$

An den Rändern erhält man:

$$
\begin{aligned}
Dm_{-1,-1} &:= < \psi^n_{-1}, \psi^n_{-1} > \\
&= < b_0 \varphi_{-1} + c_0 \varphi_0 + d_0 \varphi_1 + e_0 \varphi_2, b_0 \varphi_{-1} + c_0 \varphi_0 + d_0 \varphi_1 + e_0 \varphi_2 > \\
&= b_0^2 < \varphi_{-1}, \varphi_{-1} > + b_0 c_0 < \varphi_{-1}, \varphi_0 > \\
&\quad + c_0 b_0 < \varphi_0, \varphi_{-1} > + c_0^2 < \varphi_0, \varphi_0 > + c_0 d_0 < \varphi_0, \varphi_1 > \\
&\quad + d_0 c_0 < \varphi_1, \varphi_0 > + d_0^2 < \varphi_1, \varphi_1 > + d_0 e_0 < \varphi_1, \varphi_2 > \\
&\quad + e_0 d_0 < \varphi_2, \varphi_1 > + e_0^2 < \varphi_2, \varphi_2 > \\
&= \frac{1}{2^{n+1}} \{ c_\Delta (b_0^2 + c_0^2 + d_0^2 + e_0^2) + \frac{\Delta}{3} (b_0 c_0 + c_0 d_0 + d_0 e_0) \}
\end{aligned}
$$

$$
\begin{aligned}
Dm_{0,-1} &:= < \psi^n_0, \psi^n_{-1} > \\
&= < a \varphi_0 + b \varphi_1 + c \varphi_2 + d \varphi_3 + e \varphi_4, b_0 \varphi_{-1} + c_0 \varphi_0 + d_0 \varphi_1 + e_0 \varphi_2 > \\
&= a b_0 < \varphi_0, \varphi_{-1} > + a c_0 < \varphi_0, \varphi_0 > + a d_0 < \varphi_0, \varphi_1 > \\
&\quad + b c_0 < \varphi_1, \varphi_0 > + b d_0 < \varphi_1, \varphi_1 > + b e_0 < \varphi_1, \varphi_2 > \\
&\quad + c d_0 < \varphi_2, \varphi_1 > + c e_0 < \varphi_2, \varphi_2 > + d e_0 < \varphi_3, \varphi_2 > \\
&= \frac{1}{2^{n+1}} \{ c_\Delta (a c_0 + b d_0 + c e_0) + \frac{\Delta}{6} (a b_0 + a d_0 + b c_0 + b e_0 + c d_0 + d e_0) \} \\
&= < \psi^n_{-1}, \psi^n_0 > =: Dm_{-1,0}
\end{aligned}
$$

$$
\begin{aligned}
Dm_{2^n-2,2^n-2} &:= < \psi^n_{2^n-2}, \psi^n_{2^n-2} > \\
&= < a_1 \varphi_{2^{n+1}-4} + b_1 \varphi_{2^{n+1}-3} + c_1 \varphi_{2^{n+1}-2} + d_1 \varphi_{2^{n+1}-1}, \\
&\quad a_1 \varphi_{2^{n+1}-4} + b_1 \varphi_{2^{n+1}-3} + c_1 \varphi_{2^{n+1}-2} + d_1 \varphi_{2^{n+1}-1} > \\
&= a_1^2 < \varphi_{2^{n+1}-4}, \varphi_{2^{n+1}-4} > + a_1 b_1 < \varphi_{2^{n+1}-4}, \varphi_{2^{n+1}-3} > \\
&\quad + b_1 a_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-4} > + b_1^2 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-3} > \\
&\quad + b_1 c_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-2} > + c_1 b_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-3} > \\
&\quad + c_1^2 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-2} > + c_1 d_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-1} > \\
&\quad + d_1 c_1 < \varphi_{2^{n+1}-1}, \varphi_{2^{n+1}-2} > + d_1^2 < \varphi_{2^{n+1}-1}, \varphi_{2^{n+1}-1} > \\
&= \frac{1}{2^{n+1}} \{ c_\Delta (a_1^2 + b_1^2 + c_1^2 + d_1^2) + \frac{\Delta}{3} (a_1 b_1 + b_1 c_1 + c_1 d_1) \}
\end{aligned}
$$

$$
Dm_{2^n-3,2^n-2} := < \psi^n_{2^n-3}, \psi^n_{2^n-2} >
$$

$$
\begin{aligned}
&= < a\varphi_{2^{n+1}-6} + b\varphi_{2^{n+1}-5} + c\varphi_{2^{n+1}-4} + d\varphi_{2^{n+1}-3} + e\varphi_{2^{n+1}-2},\\
&\quad a_1\varphi_{2^{n+1}-4} + b_1\varphi_{2^{n+1}-3} + c_1\varphi_{2^{n+1}-2} + d_1\varphi_{2^{n+1}-1} >\\
&= ba_1 < \varphi_{2^{n+1}-5}, \varphi_{2^{n+1}-4} > + ca_1 < \varphi_{2^{n+1}-4}, \varphi_{2^{n+1}-4} >\\
&\quad + da_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-4} > + cb_1 < \varphi_{2^{n+1}-4}, \varphi_{2^{n+1}-3} >\\
&\quad + db_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-3} > + eb_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-3} >\\
&\quad + dc_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-2} > + ec_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-2} >\\
&\quad + ed_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-1} >\\
&= \frac{1}{2^{n+1}}\{c_\Delta(ca_1 + db_1 + ec_1) + \frac{\Delta}{6}(ba_1 + da_1 + cb_1 + eb_1 + dc_1 + ed_1)\}\\
&= < \psi^n_{2^n-2}, \psi^n_{2^n-3} >=: Dm_{2^n-2,2^n-3}
\end{aligned}
$$

Der Ergebnisvektor im Gleichungssystem zur Berechnung der Details sieht wie folgt aus:

$$
\begin{aligned}
d)\quad & De_j := \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \psi_j^n >\\
= & \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i, a\varphi_{2j} + b\varphi_{2j+1} + c\varphi_{2j+2} + d\varphi_{2j+3} + e\varphi_{2j+4} >\\
= & x_{2j-1}^{n+1}(a < \varphi_{2j-1}, \varphi_{2j} >) + x_{2j}^{n+1}(a < \varphi_{2j}, \varphi_{2j} > + b < \varphi_{2j}, \varphi_{2j+1} >)\\
& + x_{2j+1}^{n+1}(a < \varphi_{2j+1}, \varphi_{2j} > + b < \varphi_{2j+1}, \varphi_{2j+1} > + c < \varphi_{2j+1}, \varphi_{2j+2} >)\\
& + x_{2j+2}^{n+1}(b < \varphi_{2j+2}, \varphi_{2j+1} > + c < \varphi_{2j+2}, \varphi_{2j+2} > + d < \varphi_{2j+2}, \varphi_{2j+3} >)\\
& + x_{2j+3}^{n+1}(c < \varphi_{2j+3}, \varphi_{2j+2} > + d < \varphi_{2j+3}, \varphi_{2j+3} > + e < \varphi_{2j+3}, \varphi_{2j+4} >)\\
& + x_{2j+4}^{n+1}(d < \varphi_{2j+4}, \varphi_{2j+3} > + e < \varphi_{2j+4}, \varphi_{2j+4} >)\\
& + x_{2j+5}^{n+1}(e < \varphi_{2j+5}, \varphi_{2j+4} >)\\
= & \frac{1}{2^{n+1}}\{x_{2j-1}^{n+1}(a\frac{\Delta}{6}) + x_{2j}^{n+1}(ac_\Delta + b\frac{\Delta}{6}) + x_{2j+1}^{n+1}(a\frac{\Delta}{6} + bc_\Delta + c\frac{\Delta}{6})\\
& + x_{2j+2}^{n+1}(b\frac{\Delta}{6} + cc_\Delta + d\frac{\Delta}{6}) + x_{2j+3}^{n+1}(c\frac{\Delta}{6} + dc_\Delta + e\frac{\Delta}{6})\\
& + x_{2j+4}^{n+1}(d\frac{\Delta}{6} + ec_\Delta) + x_{2j+5}^{n+1}(e\frac{\Delta}{6})\}\\
& \text{für } j = 0, ..., 2^n - 3
\end{aligned}
$$

Die Randsituation stellt sich wie folgt dar:

$$
\begin{aligned}
j = -1: \quad & De_{-1} := \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \psi_{-1}^n >\\
= & \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i, b_0\varphi_{-1} + c_0\varphi_0 + d_0\varphi_1 + e_0\varphi_2 >\\
= & x_{-1}^{n+1}(b_0 < \varphi_{-1}, \varphi_{-1} > + c_0 < \varphi_{-1}, \varphi_0 >)\\
& + x_0^{n+1}(b_0 < \varphi_0, \varphi_{-1} > + c_0 < \varphi_0, \varphi_0 > + d_0 < \varphi_0, \varphi_1 >)
\end{aligned}
$$

$$
\begin{aligned}
& +x_1(c_0 < \varphi_1, \varphi_0 > + d_0 < \varphi_1, \varphi_1 > + e_0 < \varphi_1, \varphi_2 >) \\
& +x_2^{n+1}(d_0 < \varphi_2, \varphi_1 > + e_0 < \varphi_2, \varphi_2 >) \\
& +x_3^{n+1}(e_0 < \varphi_3, \varphi_2 >) \\
= \quad & \frac{1}{2^{n+1}}\{x_{-1}^{n+1}(b_0 c_\Delta + c_0 \frac{\Delta}{6}) + x_0^{n+1}(b_0 \frac{\Delta}{6} + c_0 c_\Delta + d_0 \frac{\Delta}{6}) \\
& +x_1^{n+1}(c_0 \frac{\Delta}{6} + d_0 c_\Delta + e_0 \frac{\Delta}{6}) + x_2^{n+1}(d_0 \frac{\Delta}{6} + e_0 c_\Delta) + x_3^{n+1}(e_0 \frac{\Delta}{6})\}
\end{aligned}
$$

$$
\begin{aligned}
j = 2^n - 2: \quad & De_{2^n-2} := \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i^{n+1}, \psi_{2^n-2}^n > \\
= \quad & \sum_{i=-1}^{2^{n+1}-1} x_i^{n+1} < \varphi_i, a_1\varphi_{2^{n+1}-4} + b_1\varphi_{2^{n+1}-3} + c_1\varphi_{2^{n+1}-2} + d_1\varphi_{2^{n+1}-1} > \\
= \quad & x_{2^{n+1}-5}^{n+1}(a_1 < \varphi_{2^{n+1}-5}, \varphi_{2^{n+1}-4} >) \\
& +x_{2^{n+1}-4}^{n+1}(a_1 < \varphi_{2^{n+1}-4}, \varphi_{2^{n+1}-4} > + b_1 < \varphi_{2^{n+1}-4}, \varphi_{2^{n+1}-3} >) \\
& +x_{2^{n+1}-3}(a_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-4} > + b_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-3} > \\
& \qquad +c_1 < \varphi_{2^{n+1}-3}, \varphi_{2^{n+1}-2} >) \\
& +x_{2^{n+1}-2}(b_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-3} > + c_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-2} > \\
& \qquad +d_1 < \varphi_{2^{n+1}-2}, \varphi_{2^{n+1}-1} >) \\
& +x_{2^{n+1}-1}^{n+1}(c_1 < \varphi_{2^{n+1}-1}, \varphi_{2^{n+1}-2} > \\
& \qquad +d_1 < \varphi_{2^{n+1}-1}, \varphi_{2^{n+1}-1} >) \\
= \quad & \frac{1}{2^{n+1}}\{x_{2^{n+1}-5}^{n+1}(a_1 \frac{\Delta}{6}) + x_{2^{n+1}-4}^{n+1}(a_1 c_\Delta + b_1 \frac{\Delta}{6}) \\
& +x_{2^{n+1}-3}^{n+1}(a_1 \frac{\Delta}{6} + b_1 c_\Delta + c_1 \frac{\Delta}{6}) + x_{2^{n+1}-2}^{n+1}(b_1 \frac{\Delta}{6} + c_1 c_\Delta + d_1 \frac{\Delta}{6}) \\
& +x_{2^{n+1}-1}^{n+1}(c_1 \frac{\Delta}{6} + d_1 c_\Delta)\}
\end{aligned}
$$

Damit ist auch das Gleichungssystem zur Bestimmung der Detailkoeffizienten vollständig und hat folgende Gestalt:

$$
\begin{pmatrix}
Dm_{-1,-1} & Dm_{0,-1} & & & \\
Dm_{-1,0} & Dm_{0,0} & Dm_{1,0} & & 0 \\
 & Dm_{0,1} & Dm_{1,1} & & \\
 & & & \ddots & \\
 & 0 & & Dm_{2^n-3,2^n-3} & Dm_{2^n-2,2^n-3} \\
 & & & Dm_{2^n-3,2^n-2} & Dm_{2^n-2,2^n-2}
\end{pmatrix}
\begin{pmatrix} y_{-1}^n \\ \vdots \\ y_{2^n-2}^n \end{pmatrix} =
\begin{pmatrix} De_{-1} \\ \vdots \\ \vdots \\ De_{2^n-2} \end{pmatrix}
$$

Aufgrund der Skalarprodukteigenschaften sind die Koeffizientenmatrizen beider Gleichungssysteme positiv definit und symmetrisch. Zu einer numerisch effizienten und stabilen Berechnung bietet sich damit das Cholesky-Verfahren an (siehe z.B. [15])

4.5 Die Implementierung

Zur Visualisierung einer Multiskalenanalyse mit gewähltem Parameter Δ wurde das Programm *MultiResol* geschrieben. Wird das Programm gestartet, so erscheinen ein Fenster mit Bedienelementen sowie sieben weitere Fenster mit Funktionsgraphen.

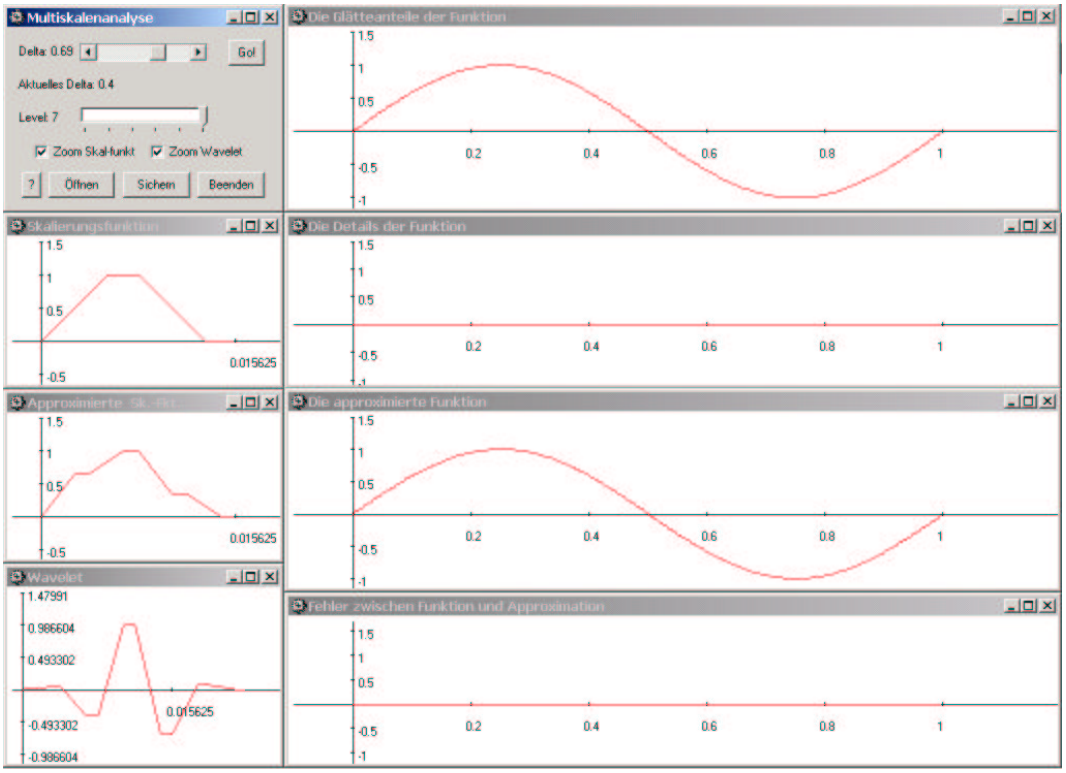


Abbildung 21: Das komplette Erscheinungsbild des Programms

Die drei Graphen auf der linken Seite enthalten von oben nach unten die Skalierungsfunktion φ, die approximierte Skalierungsfunktion $\tilde{\varphi}$, und das dazugehörige Wavelet ψ. Auf der rechten Seite sind ganz oben die groben Anteile der Funktion zu sehen, gefolgt von den fehlenden Details. Hier handelt es sich um die Summe aller Details, d.h. nicht nur um die Details des jeweiligen Levels, sondern um die aller Level.
In dem darunterliegenden Graphen werden die Glätteanteile und die fehlenden Details addiert. Hier lässt sich für ein $\Delta \in (0, 1)$ ein Fehler beobachten (s. Kapitel 5), da anstelle der Funktion $\tilde{\varphi}$ die Funktion φ verwendet wird. Dieser Fehler ist als Differenz der Ausgangsdaten und dieser Summe in dem letzten Funktionsgraphen dargestellt.

4.5.1 Die Bedienoberfläche

Ganz oben in dem Fenster befindet sich ein Schieberegler mit dem sich der Parameter Δ einstellen lässt. Dieser wird direkt links von dem Regler angezeigt. Mit dem Betätigen dieses Reglers,

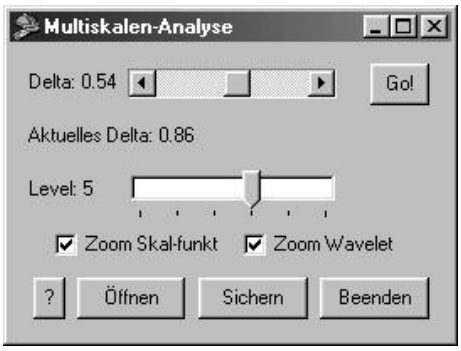


Abbildung 22: Das Bedienfeld

werden die beiden Skalierungsfunktionen und das Wavelet verändert. Hier hat der Anwender die Möglichkeit stetig zwischen der Haar- und der Hutfunktion zu wechseln. Dieses so gewählte Δ hat jedoch noch keinen Einfluss auf die rechts angezeigten Daten.

Betätigt man den Knopf *Go!*, so wird das angewählte Δ als aktuelles Δ deklariert. Die Synthese-Matrizen P und Q werden neu berechnet, und die rechts stehenden Daten werden nun als Linearkombination der Basisfunktionen des aktuellen Parameters dargestellt.

Der Schieberegler *Level* ist für einen Wechsel des aktuellen Levels zuständig. Wird ein niedrigerer Level gewählt, so werden über das Cholesky-Verfahren (s. voriges Kapitel) aus den glatten Anteilen der Daten die nächst gröberen Koeffizienten berechnet und rechts oben im Graphen angezeigt. Ebenfalls über das Cholesky-Verfahren werden die fehlenden Details ermittelt und zu den bisherigen addiert, was im zweiten Graphen rechts zu sehen ist.
Wird ein höherer Level gewählt, so werden aus den aktuellen Glättekoeffizienten und den Details vom aktuellen Level über die Synthese-Matrizen P und Q die nächst höheren Glättekoeffizienten berechnet und angezeigt. Die hierfür verwendeten Details werden von der Detaildarstellung subtrahiert.

Die beiden *Zoom*-Schalter dienen der näheren Betrachtung der Funktionsgraphen auf der linken Seite. Wird *Zoom Skal-funkt.* angewählt, so werden die beiden Skalierungsfunktionen φ und $\tilde{\varphi}$ in voller Fenstergrösse angezeigt. Analoges gilt für das Wavelet bei Wahl von *Zoom Wavelet*. Sind diese Schalter nicht angewählt, so werden die Funktionen in ihrer originalen Grösse dargestellt.

Beim Aufruf des Fragezeichens erscheint ein kleines Info-Fenster, und durch ein souveränes Anklicken des Buttons *Beenden* wird das Programm überraschenderweise beendet.

4.5.2 Das Datenformat

Mit den Buttons *Öffnen* und *Sichern* lassen sich die gerade verwendeten Daten speichern und auch wieder laden.
Beim Sichern werden die Daten in der Aufschlüsselung gespeichert, wie sie auch gerade im

Programm vorliegen, d.h. es werden die Glättekoeffizienten des aktuellen Levels und danach levelweise die fehlenden Details geschrieben. Eine solche *.mdt*-Datei hat folgenden Aufbau:

;Musterdatei.mdt
;MaxLevel:
7
;Level:
4
*;********Daten*********
;
;Glättekoeffizienten:
-0.110837
0.370455
0.716512
...
;
;Waveletkoeffizienten:
;
;Level:4
0.0227733
0.0067778
...
;
;Level:5
0.0187917
-0.000901674
...
;
;Level:6
0.00693689
0.000304413
...

Durch ein Semikolon wird ein Kommentar eingeleitet. Zu Anfang der Datei werden der Dateiname, der maximale Level (also der Ausgangslevel) und der Level zum Zeitpunkt des Speicherns (der aktuelle Level) gesichert. Darauf folgen die Glättekoeffizienten des aktuellen Levels, wiederum gefolgt von den Waveletkoeffizienten. Diese werden levelweise gespeichert, ausgehend vom aktuellen Level bis MaxLevel-1.
Wurde eine solche Datei gespeichert kann sie mit einem beliebigen Texteditor manipuliert werden, z.B. können durch Verändern der Glättekoeffizienten eines geringen Levels ganze Teile der Funktion verändert werden. Hierfür werden die geänderten Daten in *MultiResol* wieder geladen und durch eine Synthese wieder der ursprüngliche Level hergestellt.

5 Fehler-Analyse

5.1 Vorbereitungen

Wird bei der Analyse $\tilde{\varphi}$ durch φ ersetzt, so entsteht für $0 < \Delta < 1$ ein Fehler. In diesem Kapitel soll dieser Fehler genauer untersucht werden.
Ist f_{n+1} eine Approximation einer Funktion f in einem Level $n+1$, so hat f die Darstellung:

$$f_{n+1} = \sum_i x_i^{n+1} \varphi_i^{n+1}$$

Nach der Berechnung der Koeffizienten x_i^n und y_j^n und Anwendung des Isomorphismus, sprich Ersetzung von $\tilde{\varphi}$ durch φ, ergibt sich somit:

$$\begin{aligned} \sum_i x_i^{n+1} \varphi_i^{n+1} &= \sum_i x_i^n \tilde{\varphi_i^n} + \sum_j y_j^n \psi_j^n \\ &\approx \sum_i x_i^n \varphi_i^n + \sum_j y_j^n \psi_j^n \end{aligned}$$

Es entsteht also folgender Fehler:

$$\begin{aligned} e &= |\sum_i x_i^n \tilde{\varphi_i^n} + \sum_j y_j^n \psi_j^n - \sum_i x_i^n \varphi_i^n - \sum_j y_j^n \psi_j^n| \\ &= |\sum_i x_i^n \tilde{\varphi_i^n} - \sum_i x_i^n \varphi_i^n| \\ &= |\sum_i x_i^n (\tilde{\varphi_i^n} - \varphi_i^n)| \\ &\leq \sum_i |x_i^n| |\tilde{\varphi_i^n} - \varphi_i^n| \end{aligned}$$

Damit hängt der Fehler also insbesondere von der Ähnlichkeit der Funktionen $\tilde{\varphi}$ und φ ab. Ziel ist es nun, eine Fehlerfunktion in Abhängigkeit des Parameters Δ herzuleiten. Die Extremstellen dieser Funktion können hierbei ein weiteres Kriterium für die Wahl des Parameters sein.
Da sowohl $\tilde{\varphi}$ als auch φ stückweise linear sind, wird die Fehlerfunktion als $L_1 - Norm$ der Differenz dieser beiden Funktionen definiert:

$$e(\Delta) = |\tilde{\varphi_0^n}(x) - \varphi_0^n(x)|_{L_1} = \int_{\mathbb{R}} |\tilde{\varphi_0^n}(x) - \varphi_0^n(x)| dx$$

Hierbei war

$$\begin{aligned} & \tilde{\varphi_i^n} = \alpha \varphi_{2i}^{n+1} + \varphi_{i+1}^{n+1} + (1-\alpha) \varphi_{2i+1}^{n+1} \quad \text{mit} \quad \alpha = 1 - \frac{\Delta}{2} \\ \Longrightarrow \quad & \tilde{\varphi_i^n} = (1 - \frac{\Delta}{2}) \varphi_{2i}^{n+1} + \varphi_{i+1}^{n+1} + \frac{\Delta}{2} \varphi_{2i+1}^{n+1} \end{aligned}$$

Ferner galt noch $\varphi_i^n(x) = \varphi(2^n(x - i2^{-n})) = \varphi(2^n x - i)$. Damit ist also dann

$$|\tilde{\varphi}_0^n(x) - \varphi_0^n(x)| = |(1 - \frac{\Delta}{2})\varphi(2^{n+1}x) + \varphi(2^{n+1}x - 1) + \frac{\Delta}{2}\varphi(2^{n+1} - 2) - \varphi(2^n x)|.$$

Die Differenz dieser beiden Funktionen soll jetzt abschnittsweise untersucht werden. Die Abschnitte werden, wie in Figur 23 dargestellt, jeweils von einem „Knick“ zum nächsten gewählt. Die Differenz wird dann elementargeometrisch berechnet. Diese Berechnung ist für $\Delta < \frac{1}{2}$ und $\Delta \geq \frac{1}{2}$ etwas unterschiedlich, führt jedoch zum selben Ergebnis. Hier soll nun die Berechnung für $\Delta < \frac{1}{2}$ vorgeführt werden.

5.2 Die Berechnung des Fehlerintegrals

Für $\Delta < \frac{1}{2}$ ergibt sich folgende Einteilung:

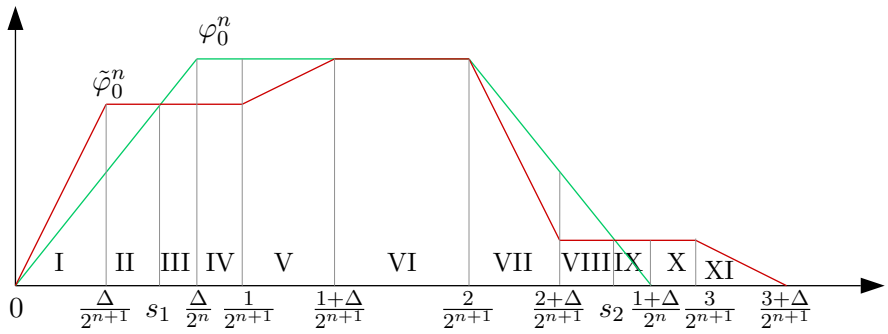


Abbildung 23: Die Einteilung der Berechnungsabschnitte

I) Die Differenz beider Funktionen ist hier die Differenz zweier Dreiecke. Das obere Dreieck hat die Höhe α, das untere die Höhe $\varphi_0^n(\frac{\Delta}{2^{n+1}}) = \varphi(\frac{2^n}{2^{n+1}}\Delta) = \frac{1}{2}$. Die x-achsenparallele Grundseite ist bei beiden Dreiecken identisch und hat die Länge $\frac{\Delta}{2^{n+1}}$, damit ist also

$$\int_0^{\frac{\Delta}{2^{n+1}}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{1}{2}\frac{\Delta}{2^{n+1}}\alpha - \frac{1}{2}\frac{1}{2}\frac{\Delta}{2^{n+1}} = \frac{\Delta}{2^{n+2}}(1 - \frac{\Delta}{2} - \frac{1}{2}) = \frac{\Delta(1-\Delta)}{2^{n+3}}.$$

II) Um die Fläche dieses und des nächsten Dreiecks zu berechnen, muss zuerst s_1 berechnet werden. s_1 ist die Stelle, an der die Funktion φ_0^n den Wert α annimmt:

$$\varphi_0^n(s_1) = \frac{2^n}{\Delta}s_1 = \alpha = 1 - \frac{\Delta}{2} \Longrightarrow s_1 = \frac{\alpha\Delta}{2^n} = \frac{(2-\Delta)\Delta}{2^{n+1}}$$

Damit hat die x-achsenparallele Seite des gesuchten Dreiecks die Länge $s_1 - \frac{\Delta}{2^{n+1}} = \frac{\Delta(1-\Delta)}{2^{n+1}}$. Wie in I) gezeigt, gilt $\varphi_0^n(\frac{\Delta}{2^{n+1}}) = \frac{1}{2}$ und trivialerweise $\tilde{\varphi}_0^n(\frac{\Delta}{2^{n+1}}) = \alpha = 1 - \frac{\Delta}{2}$. Die Höhe des Dreiecks beträgt also $\alpha - \frac{1}{2} = \frac{1}{2} - \frac{\Delta}{2}$, damit ist dann

$$\int_{\frac{\Delta}{2^{n+1}}}^{s1} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{1}{2}\frac{\Delta(1-\Delta)}{2^{n+1}}(\frac{1}{2} - \frac{\Delta}{2}) = \frac{\Delta(1-\Delta)^2}{2^{n+3}}.$$

III) Die Länge der x-achsenparallelen Grundseite beträgt $\frac{\Delta}{2^n} - s_1 = \frac{\Delta^2}{2^{n+1}}$, die Höhe hat die Länge $1 - \alpha = \frac{\Delta}{2}$. Es ergibt sich also

$$\int_{s_1}^{\frac{\Delta}{2^n}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{1}{2}\frac{\Delta^2}{2^{n+1}}\frac{\Delta}{2} = \frac{\Delta^3}{2^{n+3}}.$$

IV) Hier liegt ein Rechteck der Höhe $1 - \alpha = \frac{\Delta}{2}$ und Breite $\frac{1}{2^{n+1}} - \frac{\Delta}{2^n} = \frac{1-2\Delta}{2^{n+2}}$ vor:

$$\int_{\frac{\Delta}{2^n}}^{\frac{1}{2^{n+1}}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{\Delta}{2}\frac{1}{2^{n+1}}(1-2\Delta) = \frac{2\Delta(1-2\Delta)}{2^{n+3}}$$

V) Die x-achsenparallele Seite des zugrundeliegenden Dreiecks hat die Länge $\frac{1+\Delta}{2^{n+1}} - \frac{1}{2^{n+1}} = \frac{\Delta}{2^{n+1}}$ mit der zugehörigen Höhe der Länge $\frac{\Delta}{2}$, also:

$$\int_{\frac{1}{2^{n+1}}}^{\frac{1+\Delta}{2^{n+1}}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{1}{2}\frac{\Delta}{2^{n+1}}\frac{\Delta}{2} = \frac{\Delta^2}{2^{n+3}}$$

VI) Der Fehler ist offensichtlich gleich Null.

VII) Die Fläche des gesuchten Dreiecks berechnet sich hier aus der Fläche des Rechtecks plus Fläche des Dreiecks, welches von φ_0^n begrenzt wird, minus der Fläche des Dreiecks, welches von $\tilde{\varphi}_0^n$ begrenzt wird (s. Abbildung 24).
Es gilt (wie bei I)): $\varphi_0^n(\frac{2+\Delta}{2^{n+1}}) = \frac{1}{2}$ und $\tilde{\varphi}_0^n(\frac{2+\Delta}{2^{n+1}}) = \frac{\Delta}{2}$.

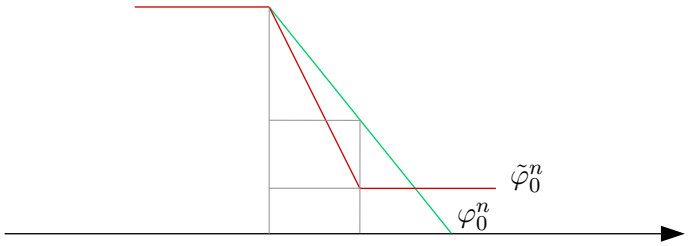


Abbildung 24: Die Berechnung bei VII)

Die Breite dieses Abschnitts beträgt $\frac{2+\Delta}{2^{n+1}} - \frac{2}{2^{n+1}} = \frac{\Delta}{2^{n+1}}$. Es ergibt sich also:

$$\int_{\frac{2}{2^{n+1}}}^{\frac{2+\Delta}{2^{n+1}}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{\Delta}{2^{n+1}}(\frac{1}{2} - \frac{\Delta}{2}) + \frac{1}{2}\frac{1}{2}\frac{\Delta}{2^{n+1}} - \frac{1}{2}(1 - \frac{\Delta}{2})\frac{\Delta}{2^{n+1}} = \frac{\Delta(1-\Delta)}{2^{n+3}}$$

VIII) Hierfür, und für die Berechnung des nächsten Dreiecks, wird der Punkt s_2 benötigt. s_2 ist die Stelle (> 1), an der die Funktion φ_0^n den Wert $1 - \alpha = \frac{\Delta}{2}$ annimmt:

$$\varphi_0^n(s_2) = \varphi(2^n s_2) = -\frac{1}{\Delta}(2^n s_2 - 1 - \Delta) = \frac{\Delta}{2} \Longrightarrow s_2 = \frac{1}{2^{n+1}}(2 + 2\Delta - \Delta^2)$$

Die Breite des Dreiecks beträgt $s_2 - \frac{\Delta}{2^{n+1}} = \frac{\Delta(1-\Delta)}{2^{n+1}}$, und die Höhe $\frac{1}{2} - \frac{\Delta}{2}$, damit ist dann

$$\int\limits_{\frac{2+\Delta}{2^{n+1}}}^{s_2} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{1}{2}\frac{\Delta(1-\Delta)}{2^{n+1}}(\frac{1}{2} - \frac{\Delta}{2}) = \frac{\Delta(1-\Delta)^2}{2^{n+3}}.$$

IX) Die Länge der x-achsenparallelen Seite beträgt $\frac{1+\Delta}{2^n} - s_2 = \frac{\Delta^2}{2^{n+1}}$, die zugehörige Höhe hat die Länge $\frac{\Delta}{2}$:

$$\int\limits_{s_2}^{\frac{1+\Delta}{2^n}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{1}{2}\frac{\Delta^2}{2^{n+1}}\frac{\Delta}{2} = \frac{\Delta^3}{2^{n+3}}$$

X) Hier liegt wieder ein Rechteck vor, diesmal mit Breite $\frac{3}{2^{n+1}} - \frac{1+\Delta}{2^{n+1}} = \frac{1-2\Delta}{2^{n+1}}$ und Höhe $\frac{\Delta}{2}$:

$$\int\limits_{\frac{1+\Delta}{2^n}}^{\frac{3}{2^{n+1}}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{2\Delta(1-2\Delta)}{2^{n+3}}$$

XI) Und zu guter Letzt noch ein Dreieck mit x-achsenparalleler Seite der Länge $\frac{3+\Delta}{2^{n+1}} - \frac{3}{2^{n+1}} = \frac{\Delta}{2^{n+1}}$ und mit Höhe $\frac{\Delta}{2}$, also:

$$\int\limits_{\frac{3}{2^{n+1}}}^{\frac{3+\Delta}{2^{n+1}}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx = \frac{1}{2}\frac{\Delta}{2^{n+1}}\frac{\Delta}{2} = \frac{\Delta^2}{2^{n+3}}$$

Damit haben wir nun alle Zutaten zusammen und können für $\Delta < \frac{1}{2}$ die Fehlerfunktion aufstellen:

$$\begin{aligned}
e(\Delta) &= \int\limits_{\mathbb{R}} |\tilde{\varphi}_0^n(x) - \varphi_0^n(x)|dx \\
&= \frac{1}{2^{n+3}}\{\Delta(1-\Delta) + \Delta^3 + (1-\Delta)^2\Delta + 2\Delta(1-2\Delta) + \Delta^2 + \Delta(1-\Delta) + \\
&\quad \Delta(1-\Delta)^2 + \Delta^3 + 2\Delta(1-2\Delta)\Delta^2\} \\
&= \frac{\Delta}{2^{n+3}}\{1 - \Delta + \Delta^2 + 1 - 2\Delta + \Delta^2 + 2 - 4\Delta + \Delta + 1 - \Delta + 1 - 2\Delta \\
&\quad + \Delta^2 + \Delta^2 + 2 - 4\Delta + \Delta\} \\
&= \frac{1}{2^{n+3}}(4\Delta^3 - 12\Delta^2 + 8\Delta)
\end{aligned}$$

Die ersten beiden Ableitungen lauten:

$$\begin{aligned}
e'(\Delta) &= \frac{1}{2^{n+3}}(12\Delta^2 - 24\Delta + 8) \\
e''(\Delta) &= \frac{1}{2^{n+3}}(24\Delta - 24)
\end{aligned}$$

Als notwendige Bedingung für Extremstellen wird die Ableitung Null gesetzt:

$$\begin{aligned}
& e'(\Delta) = 0 \\
\iff\ & 12\Delta^2 - 24\Delta + 8 = 0 \\
\iff\ & \Delta^2 - 2\Delta + 1 - \frac{1}{3} = 0 \\
\iff\ & (\Delta - 1)^2 - \frac{1}{3} = 0 \\
\iff\ & \Delta = 1 + \sqrt{\frac{1}{3}} \vee \Delta = 1 - \sqrt{\frac{1}{3}} \\
\iff\ & \Delta_1 \approx 1,5773503 \wedge \Delta_2 \approx 0,4226497
\end{aligned}$$

Δ_1 liegt nicht im Intervall $[0, 1]$ und ist deshalb für uns uninteressant. Für Δ_2 gilt

$$e''(\Delta_2) = 24(1 - \frac{1}{\sqrt{3}}) - 24 < 0.$$

Hier hat die Fehlerfunktion also ein Maximum mit $e(1 - \frac{1}{\sqrt{3}}) \approx \frac{1}{2^{n+3}}(1.396007)$. Desweiteren hat die Funktion wie erwartet Nullstellen bei $\Delta = 0$ und bei $\Delta = 1$, da hier die exakten Multiskalenanalysen des Haar-Wavelets und des linearen Wavelets vorliegen. Der Verlauf der Funktion ist in Abbildung 25 dargestellt.

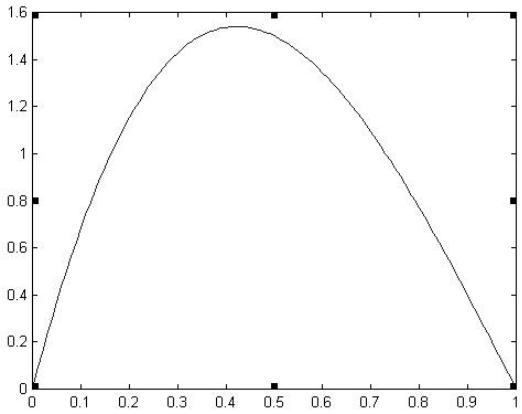


Abbildung 25: Die Fehlerfunktion

5.3 Schlussfolgerungen

Abbildung 26 zeigt für $\Delta = 0.42$ die approxoimierte Rekonstruktion einer Sinus-Kurve vom Level 7 durch Glättekoeffizienten vom Level 2 und den zugehörigen Waveletkoeffizienten. Darunter ist der Betrag der Differenz dieser Approximation und der Orgininaldaten zu sehen. Wie man erkennt, ist diese Differenz sehr gross, und die Approximation entsprechend schlecht. Die approximierte Rekonstruktion hat nur noch entfernte Ähnlichkeit mit einer Sinuskurve.
Für Δ nahe bei 0 wird der Gesamtfehler zwar kleiner, dafür treten jedoch vereinzelte Peaks auf, die das Bild stören. Für die Differenz am Punkt $x = \frac{\Delta}{2^{n+1}}$ gilt beispielsweise:

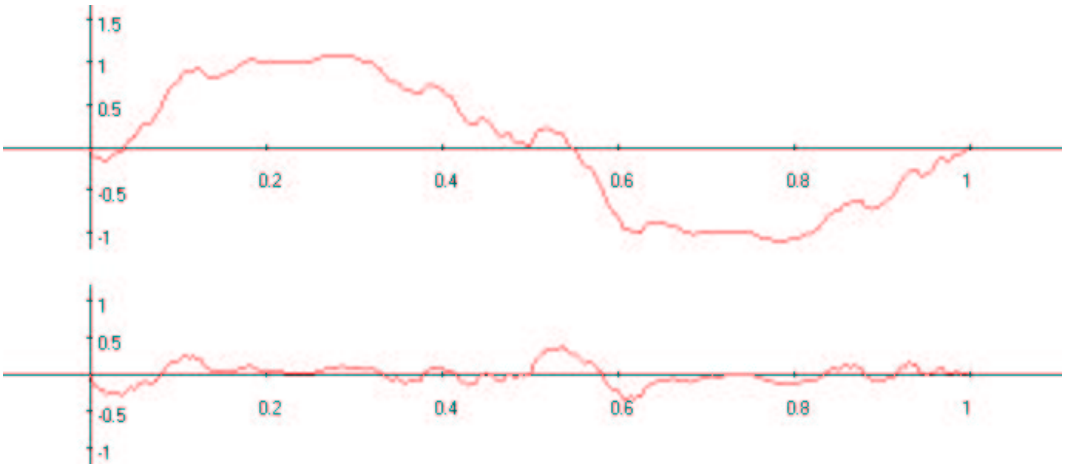


Abbildung 26: Approximierte Rekonstruktion und Fehler für $\Delta = 0.42$

$$|\varphi_0^n(\frac{\Delta}{2^{n+1}}) - \tilde{\varphi}_0^n(\frac{\Delta}{2^{n+1}})| = |\varphi(\frac{2^n}{2^{n+1}}\Delta) - \alpha\varphi(\frac{2^{n+1}}{2^{n+1}}\Delta)|$$
$$= |\varphi(\frac{\Delta}{2}) - (1 - \frac{\Delta}{2})\varphi(\Delta)| = |\frac{\Delta}{2} - \frac{1}{2}| \longrightarrow \frac{1}{2} \text{ für } \Delta \to 0$$

Für $\Delta \to 0$ geht diese Differenz gegen $\frac{1}{2}$, für $\Delta = 0$ ist sie dagegen wieder 0, da φ in die unstetige Haar-Skalierungsfunktion übergeht. Der bei $x = \frac{\Delta}{2^{n+1}}$ enstehende Peak wird somit für $\Delta \to 0$ immer schmaler, aber auch immer höher, begrenzt durch $\frac{1}{2}$. Dies wirkt sich negativ in der Approximation aus. Siehe dazu Abbildung 27, in der die Rekonstruktion der Sinuskurve für $\Delta = 0.08$ dargestellt ist.

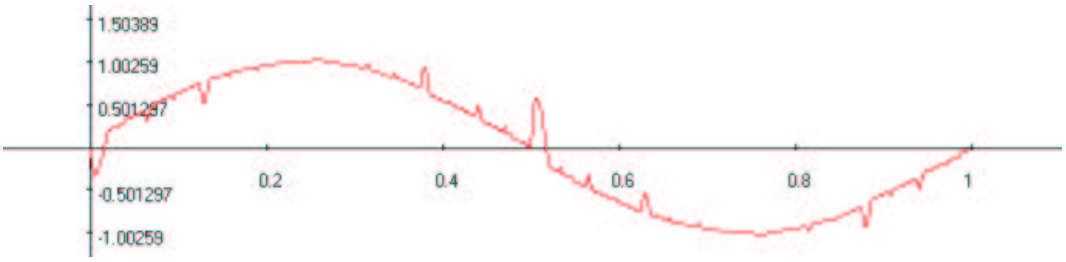


Abbildung 27: Für Δ nahe 0 entstehen Peaks in der Approximation

Allein für $\Delta \to 1$ nähert sich die approximierte Rekonstruktion den Ausgangsdaten an. Dies war auch zu erwarten, da die Hutfunktion ebenfalls stetig ist.

Résumee: Benötigt man die Darstellung der Daten durch Addition der Glätteanteile und der Details, so sind die BLaC-Wavelets für einen Parameter, der nicht in unmittelbarer Nähe der 1 liegt, eigentlich nicht zu gebrauchen. Denn auch für Δ nahe 0, wo sich der Gesamtfehler

betragsmässig der 0 nähert, treten störende Peaks auf, die das Bild der Approximation beeinträchtigen. Der Anwender sollte sich überlegen, ob sich in diesem Fall eine Mehrarbeit an Implementierung lohnt, oder ob es nicht doch ausreicht, nur auf die linearen Wavelets zurückzugreifen.

6 Mehrdimensionale Multiskalenanalyse

6.1 Herkömmliche mehrdimensionale MSA

Ein Ziel dieser Arbeit ist die Anwendung der vorgestellten BLaC-Wavelets in einem Bildkompressionsalgorithmus. Dafür, und auch für viele andere Anwendungen, ist die eindimensionale Theorie nicht mehr ausreichend. In diesem Kapitel wird sie somit für mehrdimensionale MSA erläutert, erst für n Dimensionen - dann speziell für zwei, wie wir sie für Bildkompression benötigen. Die hier vorgestellten Rahmenbedingung sind für MSA mit ineinander geschachtelten Räumen entwickelt worden, sie lassen sich aber auf den nicht ineinander geschachtelten Fall übertragen.

6.1.1 Definition einer multivariaten MRA

Analog zum 1D-Fall besteht eine Multiskalenanalyse des $L^2(\mathbb{R}^n)$ aus einer Folge von abgeschlossenen Teilräumen $V_j \subset L^2(\mathbb{R}^n)$, so dass:

$$V_j \subset V_{j+1} \tag{24}$$

$$\bigcup_j V_j \text{ ist dicht in } L^2(\mathbb{R}^n) \tag{25}$$

$$\bigcap_j V_j = 0 \tag{26}$$

Dies war die Definition einer schwachen Multiresolution.

Grundlage für eine MSA ist die Skalierungsfunktion φ. Die übrigen Basisfunktionen, die einen Raum V_j aufspannen, wurden im 1D-Fall durch Dilatation und Translation aus ihr gebildet:

$$\varphi_i^j(x) = \varphi(2^j(x - i2^{-j})) \qquad , x \in \mathbb{R}, \quad i, j \in \mathbb{Z}$$

Im Mehrdimensionalen lassen wir nun Translationen mit $k = (k_1, ..., k_n) \in \mathbb{Z}^n$ zu. Der Grundraum V_0 wird also folgendermassen aus der Skalierungsfunktion φ erzeugt:

$\{\varphi(\cdot - k) | k \in \mathbb{Z}^n\}$ ist ONB bzw. Riesz-Basis von V_0.

Der Übergang von einem Raum V_j zu V_0 wird nun mit Hilfe einer regulären Matrix A, der sog. ***Dilatationsmatrix*** beschrieben:

$$V_j = \{f(A^j x) | f \in V_0\}$$

Entsprechend der Philosophie einer Multiskalen-Zerlegung soll die Dilatationsmatrix in jede Richtung strecken, d.h. die Eigenwerte von A sollen betragsmässig grösser als 1 sein:

$$\lambda \in \sigma(A) \Longrightarrow |\lambda| > 1$$

Ferner soll A nur ganzzahlige Einträge besitzen, d.h.

$A = (a_{i,j})_{i,j=1}^n$ mit $a_{i,j} \in \mathbb{Z}$.

Damit ergibt sich nun folgende Definition einer mehrdimensionalen MSA:

Definition 8 *Eine* ***MSA*** *in $L^2(\mathbb{R}^n)$ ist eine Folge von abgeschlossenen Teilräumen V_j von $L^2(\mathbb{R}^n)$ derart, dass gilt:*

i) $(V_j)_{j\in\mathbb{Z}}$ *ist schwache Multiresolution*

ii) $\exists A \in \mathbb{Z}^{n\times n}, \quad \lambda \in \sigma(A) \Longrightarrow |\lambda| > 1$, *(Dilatationsmatrix), so dass* $V_j = \{f(A^j x) | f \in V_0\}$

iii) $\exists \varphi \in L^2(\mathbb{R}^n)$, *so dass* $\{\varphi(x-k), k \in \mathbb{Z}^n\}$ *eine Riesz-Basis von* V_0 *ist.*

Man beachte, dass trotz der n Dimensionen nur eine Skalierungsfunktion gefordert wird und nicht n Stück. Die Basis-Funktionen eines Raumes V_j lassen sich aus dieser wie folgt konstruieren:

$$\varphi_k^j = |\det A|^{-\frac{j}{2}} \varphi(A^{-j} \cdot -k) \qquad , j \in \mathbb{Z}, \quad k \in \mathbb{Z}^n$$

6.1.2 Definition des Waveletraumes

Mehrdimensionale Wavelets sind diejenigen Funktionen, die das orthogonale Komplement von V_0 in V_1 aufspannen. Im Allgemeinen benötigt man hierfür im multivariaten Fall mehr als eine Funktion.

Ein heuristisches Beispiel soll den nachfolgenden Satz von Meyer motivieren:

Sei $\varphi(x) = 1_\Omega(x)$ die charakteristische Funktion einer Menge Ω. Basisfunktion von V_1 ist dann die charakteristische Funktion von $A^{-1}\Omega$ mit dem Mass $\lambda(A^{-1}\Omega) = \frac{\lambda(\Omega)}{|\det A|}$.

Eine Basis von V_1 benötigt also „ $|\det A|$-mal “ soviele Elemente wie die Basis von V_0.
In diesem Sinne ist das orthogonale Komplement von V_0 in V_1 also „$(|\det A| - 1)$-mal“ grösser als V_0.

Satz 11 *(Meyer): Sei $\{V_m\}_{m\in\mathbb{Z}}$ eine MSA mit Dilatationsmatrix A. Dann existieren $|\det A| - 1$ Wavelets*

$$\psi_1, \psi_2, ..., \psi_{|\det A|-1} \in V_1, \tag{27}$$

die eine Orthonormalbasis des orthogonalen Komplements von V_0 in V_1 erzeugen, d.h.

$$\{\psi_{j,m,k}(\cdot) = |\det A|^{-\frac{m}{2}} \psi_j(A^{-m}\dot{-}k) | j = 1, ..., \det A - 1, m \in \mathbb{Z}, k \in \mathbb{Z}^n\} \tag{28}$$

ist eine Orthonormalbasis des $L^2(\mathbb{R}^n)$.

Beweis: Siehe [13]

6.2 Der zweidimensionale Fall

Im Folgenden wird nun der Spezialfall der zwei Dimensionen betrachtet. Dieser wird für Bildkompression und Bildanalysen benötigt. Wir möchten von einer gegebenen eindimensionalen Multiskalenanalyse ausgehend eine zweidimensionale MSA konstruieren. Hierfür wird ein Tensorproduktansatz gewählt. Es muss beachtet werden, dass hierbei die charakteristischen Wavelet-Eigenschaften erhalten bleiben. Folgender Weg wäre so z.B. nicht möglich:

$\psi \in L^2(\mathbb{R})$ sei ein Wavelet zu einer Skalierungsfunktion φ. Dann ist

$$\psi_{j,k} := \psi(2^j x - k) \quad , j,k \in \mathbb{Z}$$

ONB des $L^2(\mathbb{R})$. Es folgt, dass

$$(\psi_{j,k} \otimes \psi_{l,m})(x,y) := \psi_{j,k}(x)\psi_{l,m}(y) \quad , j,k,l,m \in \mathbb{Z}$$

eine ONB des $L^2(\mathbb{R}^2)$ bilden. Dies ist jedoch keine Wavelet-Basis, da verschiedene Level j und l gemischt werden!

Wir gehen vielmehr folgendermassen vor: Sei also φ Skalierungsfunktion einer univariaten MSA. Wir definieren die Dilatationsmatrix so, dass sowohl in x-, als auch in y-Richtung um den Faktor 2 gestreckt wird:

$$A = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

Die Einträge der Matrix sind ganzzahlig und die Eigenwerte offensichtlich grösser als 1. Für die neue zweidimensionale Skalierungsfunktion verwenden wir das Tensorprodukt

$$\varphi : \mathbb{R}^2 \longmapsto \mathbb{R}, \qquad \varphi(x,y) := (\varphi \otimes \varphi)(x,y) = \varphi(x)\varphi(y). \tag{29}$$

Damit ist der Grundraum V_0^2 definiert durch:

$$V_0^2 = V_0^1 \otimes V_0^1 \tag{30}$$

Der oben stehende Index gibt hier die Dimension an. Basisfunktionen eines Raumes V_n^2 sind:

$$\varphi_{i,j}^n(x,y) = \varphi_i^n(x)\varphi_j^n(y) = \varphi(2^n(x - i2^{-n}))\varphi(2^n(y - j2^{-n})) \qquad , x,y \in \mathbb{R}$$

Gemäss dem Satz von Meyer gibt es $\det A - 1 = 4 - 1 = 3$ verschiedene Wavelets in W_0. Diese sind gegeben durch:

$$\begin{aligned} \psi_1(x,y) &= \varphi(x)\psi(y) \\ \psi_2(x,y) &= \psi(x)\psi(y) \\ \psi_3(x,y) &= \psi(x)\varphi(y) \end{aligned}$$

Begründung: Es gilt $V_0^1 \oplus W_0^1 = V_1^1$, damit ergibt sich:

$$\begin{aligned} V_1^2 &= V_1^1 \otimes V_1^1 \\ &= (V_0^1 \oplus W_0^1) \otimes (V_0^1 \oplus W_0^1) \\ &= (V_0^1 \otimes V_0^1) \oplus (V_0^1 \otimes W_0^1) \oplus (W_0^1 \otimes V_0^1) \oplus (W_0^1 \otimes W_0^1) \end{aligned}$$

Der Raum wird also von den Basisfunktionen von $V_0^1 \otimes V_0^1$, $V_0^1 \otimes W_0^1$, $W_0^1 \otimes V_0^1$ und $W_0^1 \otimes W_0^1$ aufgespannt. Dies sind aber genau die Dilatationen und Translationen von $\varphi\varphi$, $\varphi\psi$, $\psi\varphi$ und $\psi\psi$.

6.3 Zweidimensionale Wavelet-Transformation

In diesem Abschnitt soll erläutert werden, wie nun ein zweidimensionales Signal, bzw. eine zweidimensionale Funktion, mit Wavelets transformiert werden kann. Als Anwendung soll später ein Bildkompressionsalgorithmus beschrieben werden. Hierfür fassen wir ein Bild als eine Funktion eines geeigneten quadratischen Intervalls in die reellen Zahlen auf:

$$f : [a,b]^2 \longmapsto \mathbb{R}$$

Von dieser Funktion seien Funktionswerte auf einem diskreten Gitter bekannt und in einem zweidimensionalem Array *Data* gesichert. Es gibt nun zwei übliche Wege mit Wavelets eine Transformation dieser Daten durchzuführen, die Standard- und die Nicht-Standard-Dekomposition.

6.3.1 Die Standard-Dekomposition

Um die Standard-Dekomposition einer Funktion zu erhalten, wenden wir zunächst die eindimensionale Wavelet-Transformation auf jede Spalte an. Beispielsweise werden die Glättekoeffizienten in der ersten Hälfte und die Detailkoeffizienten in der zweiten Hälfte jeder Spalte gespeichert. Sodann werden die Glättekoeffizienten einer jeden Spalte als Daten eines um eins niedrigeren Levels betrachtet und erneut in der gleichen Weise transformiert. Dies wird so fortgeführt, bis man die Spalten bis zum kleinsten gewünschten Level transformiert hat.
Nun werden wieder alle Koeffizienten betrachtet und man transformiert alle Zeilen bei höchstem Level. Wieder werden die Glättekoeffizienten in der ersten Hälfte und die Details in der zweiten gespeichert. Anschliessend wird jede erste Hälfte einer Zeile bei einem um eins niedrigeren Level transformiert. Ist der niedrigste Level auch hier erreicht, so ist die Standard-Dekomposition abgeschlossen.
Um wieder die Ausgangsdaten zu erhalten, werden diese Schritte in umgekehrter Reihenfolge rückgängig gemacht. D.h. man führt zuerst für alle Zeilen eine Synthese durch, beginnend mit kleinstem Level und endend mit dem höchsten. Dann erst werden alle Spalten zurücktransformiert, auch hier vom kleinsten bis zum höchsten Level.
Der zugehörige Algorithmus sieht folgendermassen aus:

procedure *StandardDecomposition*(**data: array**[1..$2^{MaxLevel}$,1..$2^{MaxLevel}$])
    {*Zuerst vollständige Zerlegung der Spalten*}
    for j=MaxLevel **downto** MinLevel
        {*Dekomposition der einzelnen Spalten*}
        for Spalte=1 **to** 2^j
            $Decomposition(Data[1..2^j, Spalte])$
        end for
    end for
    {*Jetzt vollständige Zerlegung der Zeilen*}
    for j=MaxLevel **downto** MinLevel
        {*Dekomposition der einzelnen Zeilen*}
        for Zeile=1 **to** 2^j
            $Decomposition(Data[Zeile, 1..2^j])$
        end for
    end for
end procedure

procedure *StandardReconstruction*(**data: array**[1..$2^{MaxLevel}$,1..$2^{MaxLevel}$])
    {*Zuerst vollständige Rekonstruktion der Zeilen*}
    for j=MinLevel **to** MaxLevel
        {*Rekonstruktion der einzelnen Zeilen*}
        for Zeile=1 **to** 2^j
            $Reconstruction(Data[Zeile, 1..2^j])$
        end for
    end for
    {*Jetzt vollständige Rekonstruktion der Spalten*}
    for j=MinLevel **to** MaxLevel
        {*Rekonstruktion der einzelnen Spalten*}
        for Spalte=1 **to** 2^j

$Reconstruction(Data[1..2^j, Spalte])$
end for
end for
end procedure

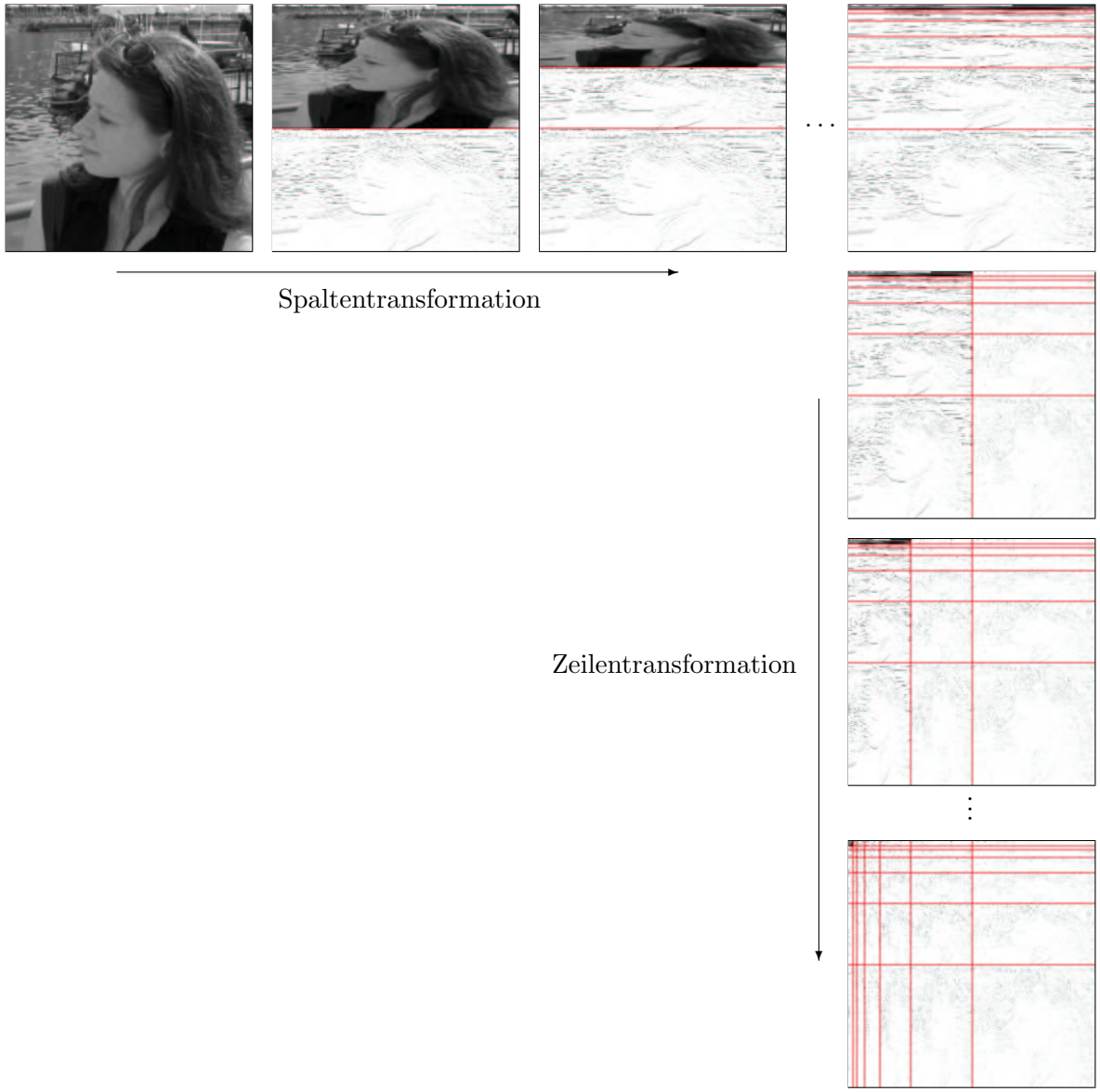


Abbildung 28: Die Standard-Dekomposition

6.3.2 Die Nicht-Standard-Dekomposition

Bei der Nicht-Standard-Dekomposition alterniert man zwischen Spalten- und Zeilentransformationen. Zunächst werden wieder alle Spalten transformiert, und die Glätte- und Detailkoeffizien-

ten gespeichert. Bei gleichem Level werden dann alle Zeilen transformiert. Man erhält somit ein Viertel, in dem alle Glättekoeffiizienten und drei Viertel in denen alle Details enthalten sind. In diesem ersten Viertel werden nun wieder bei einem um eins erniedrigten Level erst die Spalten, dann die Zeilen transformiert. Dies wird bis zum kleinsten Level so fortgeführt.
Die Rekonstruktion erfolgt wieder mit eindimensionalen Rücktransformationen in der umgekehrten Reihenfolge. Mit dem kleinsten Level beginnend werden zunächst die Zeilen, dann die Spalten rekonstruiert, dann erst der Level erhöht, bis der Ausgangslevel wieder erreicht ist.
Der Algorithmus ist folgendermassen aufgebaut:

procedure *NichtStandardDecomposition*(**data: array**[1..$2^{MaxLevel}$,1..$2^{MaxLevel}$])
    {*gewünschte Levelanzahl durchlaufen*}
    for j=MaxLevel **downto** MinLevel
        {*In jedem Level werden sowohl Spalten als auch Zeilen zerlegt*}
        {*Zuerst Dekomposition auf Spalten anwenden*}
        for Spalte=1 **to** 2^j
            $Decomposition(Data[1..2^j, Spalte])$
        end for
        {*Dann Dekomposition auf Zeilen anwenden*}
        for Zeile=1 **to** 2^j
            $Decomposition(Data[Zeile, 1..2^j])$
        end for
    end for
end procedure

procedure *NichtStandardRekonstruktion*(**data: array**[1..$2^{MaxLevel}$,1..$2^{MaxLevel}$])
    {*gewünschte Levelanzahl durchlaufen*}
    for j=MinLevel **to** MaxLevel
        {*Zuerst Zeilen rekonstruieren*}
        for Zeile=1 **to** 2^j
            $Reconstruction(Data[Zeile, 1..2^j])$
        end for
        {*Dann Spalten rekonstruieren*}
        for Spalte=1 **to** 2^j
            $Reconstruction(Data[1..2^j, Spalte])$
        end for
    end for
end procedure

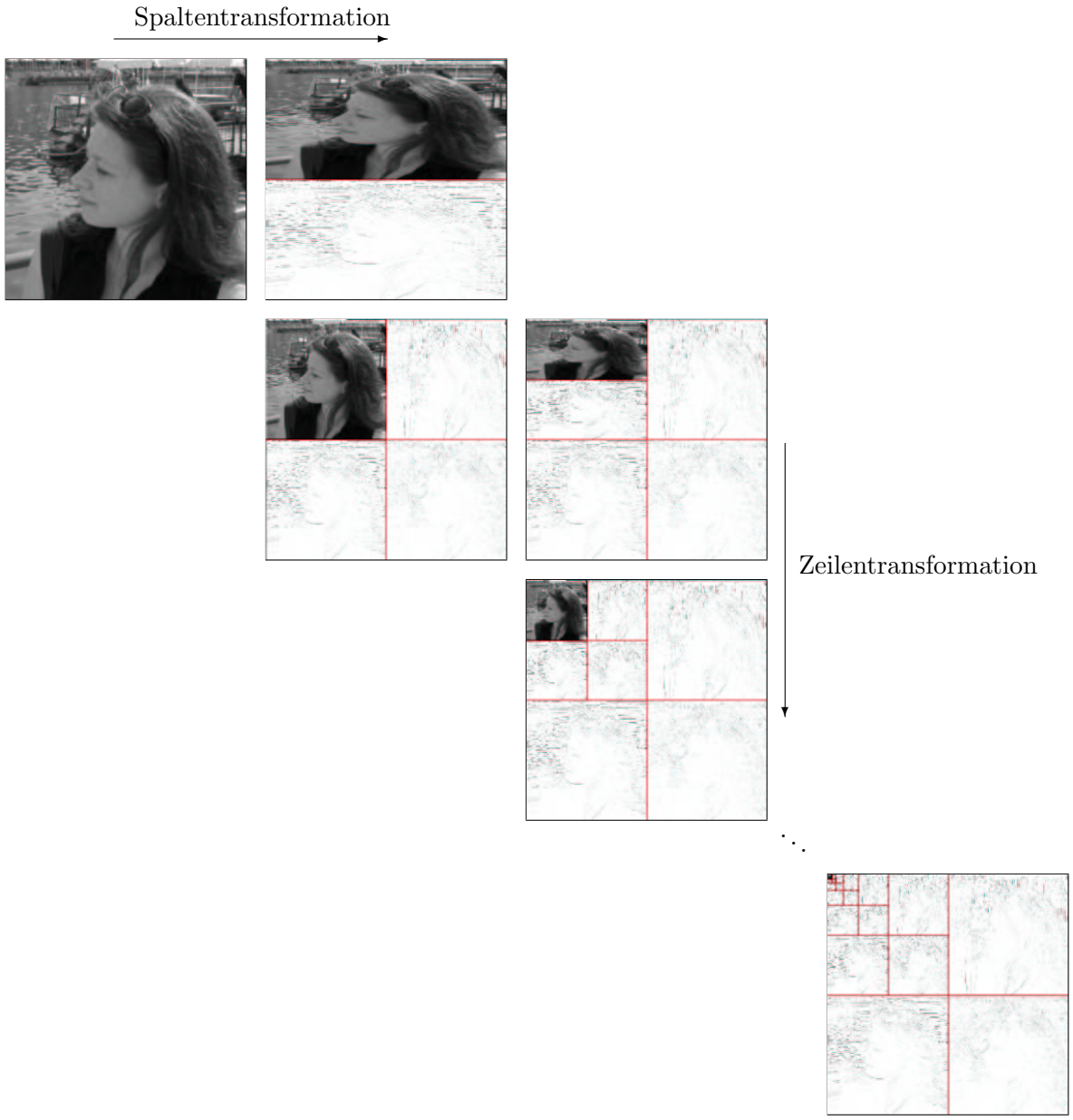


Abbildung 29: Die Nicht-Standard-Dekomposition

6.4 Die zweidimensionale BLaC-Skalierungsfunktion

Wie in Abschnitt 6.2 gezeigt, lässt sich die zweidimensionale Multiskalenanalyse auf die eindimensionale MSA zurückführen.

Wir bilden wie schon beschrieben die zweidimensionale Skalierungsfunktion durch

$$\varphi(x,y) = \varphi(x)\varphi(y) = \begin{cases} \varphi(x) & : \ \Delta \leq y < 1 \\ \varphi(y) & : \ \Delta \leq x < 1 \\ \frac{xy}{\Delta^2} & : \ 0 \leq x, y < \Delta \\ -\frac{x^2}{\Delta}(y-1-\Delta) & : \ 0 \leq x < \Delta, 1 \leq y < 1+\Delta \\ -\frac{y^2}{\Delta}(x-1-\Delta) & : \ 0 \leq y < \Delta, 1 \leq x < 1+\Delta \\ \frac{1}{\Delta^2}(x-1-\Delta)(y-1-\Delta) & : \ 1 \leq x < 1+\Delta, 1 \leq y < 1+\Delta \\ 0 & : \ sonst \end{cases} \tag{31}$$

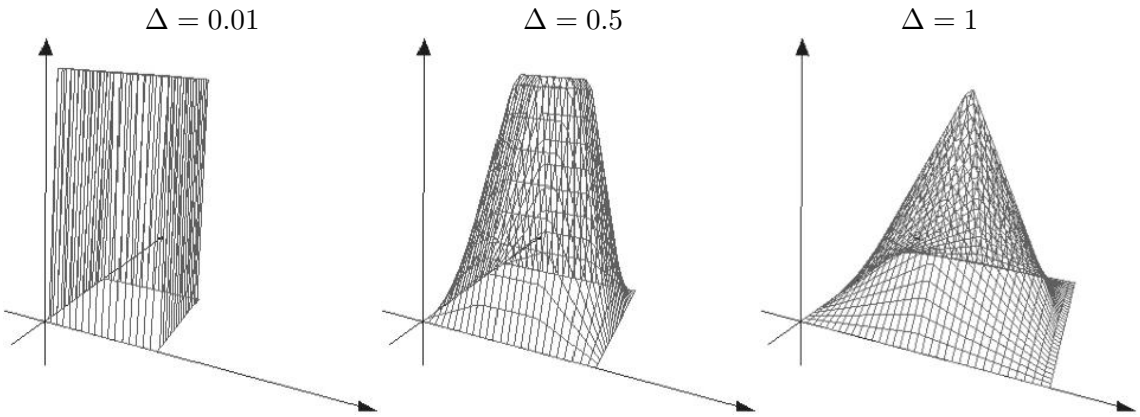


Abbildung 30: Die zweidimensionale BLaC-Skalierungsfunktion für $\Delta = 0.01, 0.5, 1$

Anmerkung: Auf die Dimensions-angebenden Indizes wird der Übersicht halber verzichtet.

Um die zweidimensionalen Räume V_n zu erzeugen, werden folgende Funktionen durch Dilatation mit $A = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$ und Translationen mit $i2^{-n}$ in x-Richtung und $j2^{-n}$ in y-Richtung gebildet:

$$\varphi^n_{i,j}(x,y) = (\varphi \otimes \varphi)(A^n(x - i2^{-n}, y - j2^{-n})) = \varphi(2^n(x - i2^{-n}))\varphi(2^n(y - j2^{-n}))$$

Der Raum V_n wird nun folgendermassen definiert:

$$V_n = \{f : \mathbb{R}^2 \longmapsto \mathbb{R} : f(x,y) = \sum_{i,j} x_{i,j}\varphi^n_{i,j}(x,y), \quad x_{i,j} \in \mathbb{R}\}$$

Da der Bildkompressionsalgorithmus auf beschränkten Gebieten ablaufen soll, wird hier wiederum die Restriktion auf das Intervall $[0,1]^2$ betrachtet. Für einen gegebenen Level n sind daher nur die Funktionen $\varphi^n_{i,j}$ mit $i,j \in \{-1, ..., 2^n - 1\}$ von Interesse. Damit ergibt sich für die Dimension von V_n:

$$\dim V_n = \nu(n)^2 = (2^n + 1)^2$$

Auch im 2D-Fall sind die Skalierungsfunktionen $\varphi^n_{i,j}$ nicht durch Funktionen $\varphi^{n+1}_{i,j}$ darstellbar. Für z.B. ein festes $\Delta \leq y < 1$ ist die 2D-Skalierungsfunktion mit der eindimensionalen Funktion $\varphi(x)$ identisch. Da diese sich schon nicht durch Funktionen höheren Levels darstellen lässt, gilt dies für zweidimensionale Skalierungsfunktionen ebenfalls.

6.5 Eigenschaften der zweidimensionalen Skalierungsfunktion

Die folgenden Eigenschaften werden analog zum 1D-Fall notiert.

Satz 12 *Sei $f \in C([0,1]^2)$ gegeben. Definiere die Stützstellen $\eta^n_{i,j} \in \mathbb{R}^2$ analog zum 1D-Fall durch*

$$\begin{aligned}
&\eta^n_{i,j} = (\frac{2i+1+\Delta}{2^{n+1}}, \frac{2j+1+\Delta}{2^{n+1}}) \quad , i,j = 0,...,2^n-2,\\
&\eta^n_{-1,-1} = (0,0), \quad \eta^n_{2^n-1,2^n-1} = (1,1),\\
&\eta^n_{-1,j} = (0, \frac{2j+1+\Delta}{2^{n+1}}), \quad \eta^n_{i,-1} = (\frac{2i+1+\Delta}{2^{n+1}}, 0),\\
&\eta^n_{2^n-1,j} = (1, \frac{2j+1+\Delta}{2^{n+1}}), \quad \eta^n_{i,2^n-1} = (\frac{2i+1+\Delta}{2^{n+1}}, 1).
\end{aligned}$$

Dann interpoliert die Funktion

$$BL_n f(x,y) = BLf(x,y,n) := \sum_{i=-1}^{2^n-1} \sum_{j=-1}^{2^n-1} f(\eta^n_{i,j}) \varphi^n_{i,j}(x,y) \tag{32}$$

die Punkte $(\eta^n_{i,j}, f(\eta^n_{i,j})) \in \mathbb{R}^3$.

Beweis: Nach Definition der Stützstellen gilt: $\varphi^n_{i,j}(\eta^n_{k,l}) = \begin{cases} 1 & , \quad i=k, j=l \\ 0 & , \quad sonst \end{cases}$ für alle $i,j,k,l = -1,...,2^n-1$, damit ist dann

$$BL_n f(\eta^n_{i,j}) = f(\eta^n_{i,j}) \varphi^n_{i,j}(\eta^n_{i,j}) = f(\eta^n_{i,j}) \text{ für alle } i,j = -1,...,2^n-1.$$

□

Auch hier werden bei der späteren Implementierung der Visualisierung einer zweidimensionalen MSA mit BLaC-Wavelets die gegebenen Daten mit Hilfe des BL_n-Operators angezeigt. Hierfür wird das Intervall $[0,1]^2$ mit einer der Bildschirmauflösung entsprechenden Genauigkeit in x- und in y-Richtung durchlaufen, und an den jeweiligen Stellen die Funktion $BLf(x,y,n)$ ausgewertet. Die Werte $f(\eta^n_{i,j}), i,j = -1,...,2^n-1$ entsprechen den Glättekoeffizienten zum Level n.

Satz 13 *Die Skalierungsfunktionen eines Levels partitionieren die Eins, d.h. es gilt*

$$\sum_{i,j=-1}^{2^n-1} \varphi^n_{i,j}(x,y) = 1$$

Beweis: Es gilt:

$$\sum_{i,j=-1}^{2^n-1} \varphi^n_{i,j}(x,y) = \sum_{i=-1}^{2^n-1} \sum_{j=-1}^{2^n-1} \varphi^n_i(x) \varphi^n_j(y) = \sum_{i=-1}^{2^n-1} \varphi^n_i(x) \underbrace{\left(\sum_{j=-1}^{2^n-1} \varphi^n_j(y) \right)}_{1} = \underbrace{\sum_{i=-1}^{2^n-1} \varphi^n_i(x)}_{1} = 1$$

□

Dies sichert, wie auch im eindimensionalen Fall, numerische Stabilität.

Satz 14 *Für $f \in C([0,1]^2)$ gilt:* $\lim_{n\to\infty} BL_n f(x,y) = f(x,y)$ *für alle* $(x,y) \in [0,1]^2$.

Beweis: Für diesen Beweis verwenden wir Theorem 3.11 aus [7], S.423. Hierbei ist $X = [0,1]^2$ und $d(x,y) = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2}$ die euklidische Metrik. Theorem 3.11 besagt dann:

Ist $BL_n : C([0,1]^2) \longmapsto C([0,1]^2)$ ein positiver linearer Operator, dann gilt für alle $\varepsilon > 0$ $f \in C([0,1]^2)$ und $(x,y) \in [0,1]^2$ die Ungleichung:

$$\begin{aligned} |BL_n(f;x,y) - f(x,y)| &\leq |BL_n(1;x,y)-1| \cdot |f(x,y)| \\ &\quad + max\{||BL_n||, \frac{1}{\varepsilon} BL_n(d(\cdot,(x,y))^2;(x,y))^{\frac{1}{2}}\} \cdot \tilde{\omega}(f;\varepsilon) \end{aligned}$$

Da der BL_n-Operator die 1-Funktion reproduziert, und ferner $||BL_n|| = 1$ und $\tilde{\omega}(f;\varepsilon) \leq 2\omega(f;\varepsilon)$ gilt, folgt aus obiger Ungleichung:

$$|BL_n(f;x,y) - f(x,y)| \leq 2max\{1, \frac{1}{\varepsilon} BL_n(d(\cdot,(x,y))^2;(x,y))^{\frac{1}{2}}\} \cdot \omega(f;\varepsilon) \tag{33}$$

Nun ist

$$BL_n(d(\cdot,(x,y))^2;(x,y))^{\frac{1}{2}} = \sqrt{\sum_{i=-1}^{2^n-1}\sum_{j=-1}^{2^n-1} d(\eta^n_{i,j},(x,y))^2 \varphi^n_{i,j}(x,y)} =: (*).$$

Sei nun $x \in [\frac{k}{2^n}, \frac{k+1}{2^n})$ und $y \in [\frac{l}{2^n}, \frac{l+1}{2^n}), k,l \in \{0,...,2^n-1\}$. Ist $x = 1$ oder $y = 1$, dann wähle $k = 2^{n-1}$, bzw. $l = 2^{n-1}$. Die Summe unter der Wurzel besteht dann aus höchstens vier Summanden:

$$\begin{aligned} (*) &= \Bigg(\underbrace{d(\eta^n_{k-1,l-1},(x,y))^2}_{\leq \frac{1}{2^{2n-3}}} \underbrace{\varphi^n_{k-1,l-1}(x,y)}_{\leq 1} + \underbrace{d(\eta^n_{k,l-1},(x,y))^2}_{\leq \frac{1}{2^{2n-3}}} \underbrace{\varphi^n_{k,l-1}(x,y)}_{\leq 1} \\ &\quad + \underbrace{d(\eta^n_{k-1,l},(x,y))^2}_{\leq \frac{1}{2^{2n-3}}} \underbrace{\varphi^n_{k-1,l}(x,y)}_{\leq 1} + \underbrace{d(\eta^n_{k,l},(x,y))^2}_{\leq \frac{1}{2^{2n-3}}} \underbrace{\varphi^n_{k,l}(x,y)}_{\leq 1} \Bigg)^{\frac{1}{2}} \\ &\leq \left(4\frac{1}{2^{2n-3}}\right)^{\frac{1}{2}} \end{aligned}$$

Denn wie schon im Beweis des eindimensionalen Falls gezeigt, ist der maximale Abstand in jeder Komponente kleiner gleich $\frac{1}{2^{n-1}}$. Damit gilt dann:

$$d(\eta^n_{k-1,l-1},(x,y))^2 \leq \left(\frac{1}{2^{n-1}}\right)^2 + \left(\frac{1}{2^{n-1}}\right)^2 = \frac{2}{2^{2n-2}} = \frac{1}{2^{2n-3}}$$

Wähle nun $\varepsilon = \left(\frac{4}{2^{2n-3}}\right)^{\frac{1}{2}}$, dann ist
$max\{1, \frac{1}{\varepsilon} BL_n(d(\cdot,(x,y))^2;x,y)^{\frac{1}{2}}\} \leq max\{1, (\frac{2^{2n-3}}{4})^{\frac{1}{2}} (\frac{4}{2^{2n-3}})^{\frac{1}{2}}\} = 1$, und aus (33) folgt:

$$|BL_n(f;x,y) - f(x,y)| \leq 2\omega(f;\varepsilon) = 2\omega(f;(\frac{4}{2^{2n-3}})^{\frac{1}{2}}) \longrightarrow 0 \text{ für } n \longrightarrow \infty.$$

Damit gilt: $BL_n(f;x,y) \longrightarrow f(x,y)$ für $n \longrightarrow \infty \quad \forall (x,y) \in [0,1]^2$.

□

6.6 Die zweidimensionalen BLaC-Wavelets

Die zweidimensionalen BLaC-Wavelets ergeben sich, wie in Abschnitt 6.2 gezeigt, durch:

$$\psi_1(x,y) = \varphi(x)\psi(y)$$
$$\psi_2(x,y) = \psi(x)\psi(y)$$
$$\psi_3(x,y) = \psi(x)\varphi(y)$$

Das eindimensionale Wavelet ψ ist hierbei durch die Berechnungen in Kapitel 3.7 gegeben.
In der Abbildung 31 sind von oben nach unten ψ_1 bis ψ_3 zu sehen, jeweils von links nach rechts für den Parameter $\Delta = 0.01, 0.5, 1$.

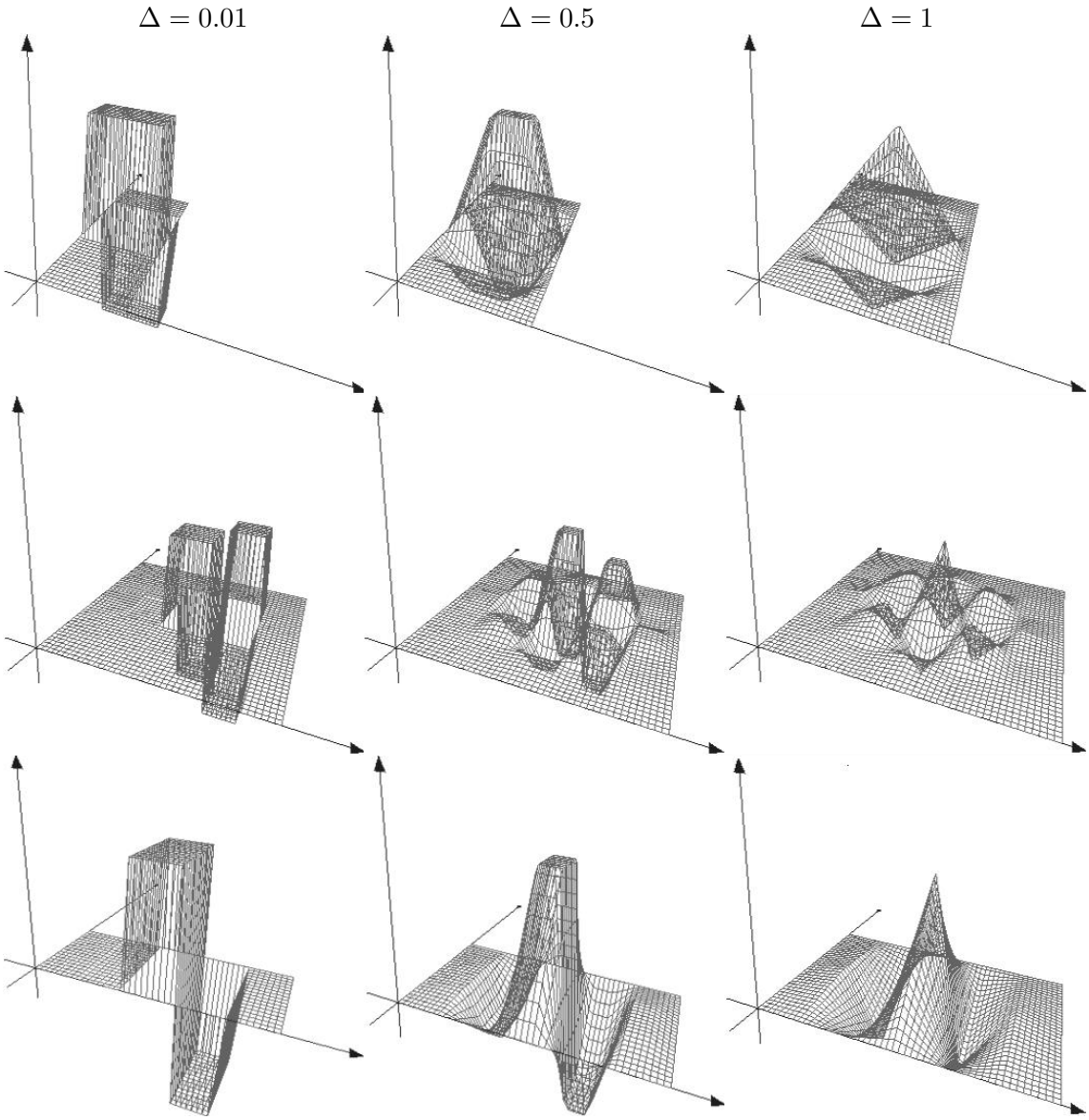


Abbildung 31: Die zweidimensionalen BLaC-Waveletfunktionen für $\Delta = 0.01, 0.5, 1$

6.7 Das Programm MRA2D

Das Programm MRA2D wurde zur Visualisierung der zweidimensionalen BLaC-Skalierungsfunktion, den zugehörigen Wavelets und der Nicht-Standard-Dekomposition geschrieben.
Hinweis: Das Programm verwendet *OpenGL*, d.h. zur Lauffähigkeit des Programms müssen die beiden Dateien *vcl60.bpl* und *rtl60.bpl* in das Verzeichnis *c:\windows\system* kopiert werden.
Startet man das Programm, so erscheinen drei Fenster auf dem Schirm.

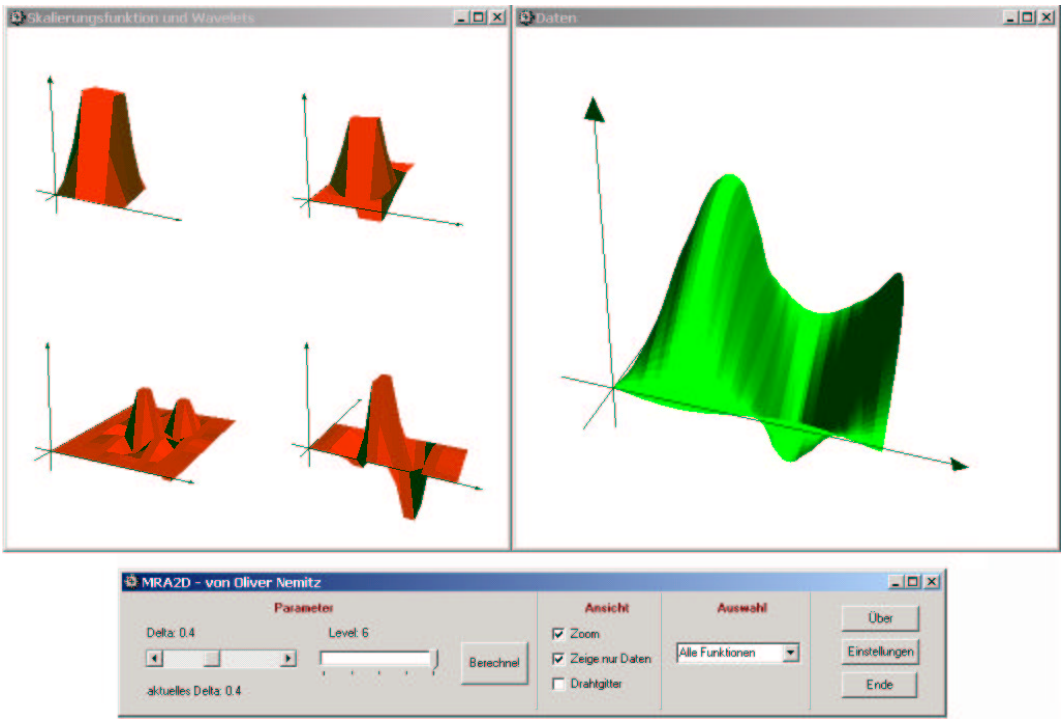


Abbildung 32: Das komplette Erscheinungsbild des Programms

In dem linken oberen Fenster sind vier in rot gezeichnete Funktionsgraphen zu sehen:

1. die Skalierungsfunktion $\varphi\varphi$ (links oben),
2. das erste Wavelet $\varphi\psi$ (rechts oben),
3. das zweite Wavelet $\psi\psi$ (links unten)
4. das dritte Wavelet $\psi\varphi$ (rechts unten).

In dem rechten oberen Fenster sind in grün Beispieldaten gegeben. Alle Koordinatensysteme lassen sich drehen und verschieben. Wird die Taste „r“ gedrückt, so befindet man sich im Rotationsmodus. Nun kann bei Selektion eines Grafikfensters das darin dargestelle Koordinatensystem mit den Pfeiltasten gedreht werden. Die Pfeiltasten nach links und rechts dienen zum Rotieren um die z-Achse, nach oben und unten um die x-Achse und mit Shift-Oben und Shift-Unten wird um die y-Achse gedreht.
Mit Betätigen der Taste „t“ wird in den Translationsmodus gewechselt. Auch hier lassen sich

mit denselben Tastenkombinationen die Koordinatensysteme in die entsprechenden Achsenrichtungen verschieben.

6.7.1 Die Steuerung

In dem unteren Fenster sind die Bedienelemente aufgeführt.

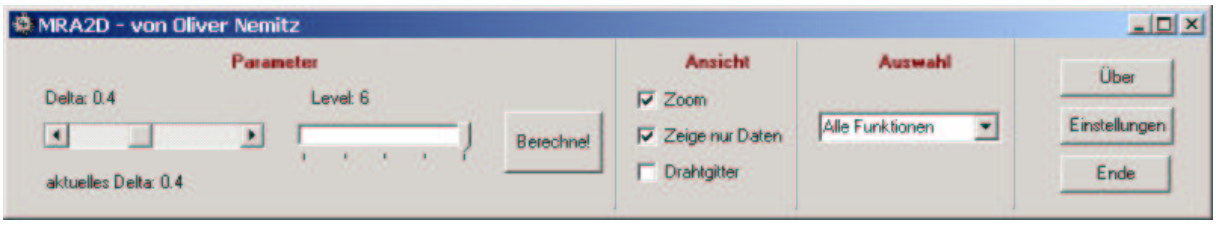


Abbildung 33: Die Steuerungsoberfläche

Diese sind im Einzelnen:

Der Schieberegler *Delta*, mit dem sich wie im Programm *MultiResol* der Parameter Delta einstellen lässt. Wird dieser Regler bewegt, so werden die Skalierungsfunktion und die Wavelets in Echtzeit aktualisiert. Diese Änderung hat jedoch noch keine Auswirkung auf die Daten, diese sind nur von dem darunter angezeigten *aktuellen Delta* abhängig.

Mit dem Schieberegler *Level* lässt sich, analog zum Programm *MultiResol* der aktuelle Level verändern. Wird ein niedrigerer Level gewählt, so werden die Daten durch eine Nicht-Standard-Dekomposition in einen gröberen Anteil und drei Detailanteile zerlegt.
Wird ein grösserer Level gewählt, so wird die Nicht-Standard-Dekomposition wieder rückgängig gemacht.

Durch Betätigen des Buttons *Berechne* werden die benötigten Matrizen mit dem gewählten Delta neu berechnet, und die Daten neu dargestellt. Das gewählte Delta wird zum aktuellen Delta.

In dem mittleren Bereich mit der Überschrift *Ansicht* sind drei Kontrollkästchen zu sehen:

1. Über das erste Kontrollkästchen *Zoom* lassen sich die Skalierungsfunktion und die Wavelets wie in *MultiResol* entweder in Originalgrösse oder in Nahansicht betrachten.
2. Das zweite Kontrollfeld *Zeige nur Daten* bezieht sich auf die Dekomposition der Daten. Ist dieses Feld mit einem Haken versehen, so werden bei der Wahl eines niedrigeren Levels nur die gröberen Daten angezeigt. Ansonsten werden in dem Koordinatensystem sowohl die gröberen Daten in grün angezeigt, als auch die berechneten Details in rot, blau und wieder rot.
3. Mit dem dritten Kontrollfeld lässt sich zwischen der Drahtgitter- und der Flächendarstellung wechseln.

In dem dritten Feld *Auswahl* ist ein Auswahlfeld zu sehen. Hier kann festgelegt werden, welche Funktion in dem linken Grafikfenster zu sehen sein soll.
Werden eine oder mehrere Funktionen zur Ansicht ausgewählt, so werden diese immer wieder in derselben Anfangsposition gezeigt, d.h. eventuelle Verschiebungen oder Rotationen werden wieder rückgängig gemacht.

Ganz rechts befinden sich noch die beiden Buttons *Einstellungen* und *Ende*. Tatsächlich bewahrheitet sich die spontane Vermutung des geneigten Lesers, dass ein Klick auf den zweiten Button zum Beenden des Programms dient. Der andere öffnet ein Fenster mit mehreren Konfigurationsmöglichkeiten:

6.7.2 Die Einstellungen

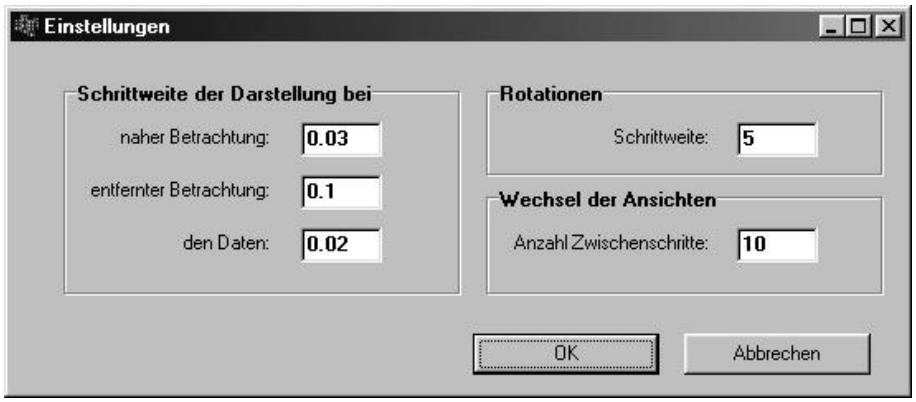


Abbildung 34: Konfiguration des Programms

Zur Darstellung sowohl der Daten, als auch der Skalierungsfunktion und der Wavelets wird das Intervall $[0,1]^2$ mit einer vorgegebenen Schrittweite durchlaufen, und an diesen Punkten die jeweilige Funktion ausgewertet. Diese Schrittweite kann nun links eingestellt werden. Je niedriger die Schrittweite, desto genauer die Darstellung der Funktion. Allerdings leidet bei zu niedriger Schrittweite die Geschwindigkeit des Programms.

Die Schrittweite der Darstellung bei *naher Betrachtung* bezieht sich auf die Situation, in der entweder die Skalierungsfunktion oder nur ein Wavelet in dem linken Fenster dargestellt wird. Da diese Funktion dann entsprechend grösser gezeichnet wird, sollte die Schrittweite hier niedriger eingestellt sein als bei *entfernter Betrachtung*, wenn alle vier Funktionen in einem Fenster zu sehen sind.
Das unterste Feld dient zur Einstellung der Schrittweite bei der Darstellung der Daten.

Rechts oben lässt sich die Schrittweite der Rotationen in Grad einstellen.

Wird im linken Grafikfenster die Ansicht zwischen den Funktionen gewechselt, so wird nur die Kameraperspektive bzgl. der vier Koordinatensysteme verändert. So wandert bei Wahl von nur einer Funktion die Kamera so nah an das entsprechende Koordinatensystem, dass die an-

deren nicht mehr zu sehen sind. Wird dann eine andere Funktion ausgewählt, so wechselt die Kamera nur ihre Position.
Solche Ansichtenwechsel geschehen nicht abrupt, sondern animiert über mehrere Zwischenschritte. Diese Anzahl der Zwischenschritte lässt sich hier rechts unten einstellen.

Schliesslich werden mit einem Klick auf *Ok* die geänderten Einstellungen übernommen, mit einem Klick auf *Abbrechen* verworfen.

7 Bildkompression

In diesem Kapitel werden nun die Ergebnisse der vorangegangenen Abschnitte auf einen Bildkompressionsalgorithmus angewandt. Hierbei wird jedoch der Schwerpunkt nicht darauf gelegt, einen möglichst effektiven, schnellen Algorithmus zu entwerfen, sondern die Unterschiede von mit verschiedenen Parameter Δ komprimierten Bildern zu zeigen. Es soll verdeutlicht werden, dass für gewisse Bilder ein Parameter echt grösser Null und echt kleiner Eins subjektiv bessere Ergebnisse liefert als $\Delta = 0$ oder $\Delta = 1$.

7.1 Digitale Bilder

Ein digitales Grauwertbild besteht aus einer ganzzahligen $(n \times m)$-Matrix $B = (b_{i,j})_{i,j=1}^{n,m}$. Ein solches Bild hat n Zeilen und m Spalten. Die Einträge $b_{i,j}$ geben den zu einem Bildpunkt gehörenden Grauwert an. Üblicherweise liegt dieser im Intervall [0,255], wobei $b_{i,j} = 0$ als schwarz und $b_{i,j} = 255$ als weiss interpretiert wird.
Ein Bild dieser Grösse benötigt $n \cdot m$ Bytes Speicherplatz. Ziel eines Kompressionsalgorithmus ist es nun, den Speicherbedarf eines solchen Bildes zu reduzieren, ohne jedoch dabei das Bild zu sehr zu verfremden.

Für ein digitales Farbbild liegen in der Regel drei Matrizen vor. Bei der additiven Farbmischung sind dies z.B. die Matrizen $R, G, B \in (\mathbb{Z} \cap [0, 255])^{n \times m}$, die die Rot-, Grün- und Blauanteile des Farbwertes enthalten. Haben alle drei Farbkomponenten den Wert 0, so wird der Bildpunkt auch hier schwarz gezeichnet. Haben alle den Wert 255, so wird er entsprechend weiss eingefärbt.
Anstelle der RGB-Werte, können z.B. auch die Luminanz- (Y) und Chrominanz-Werte (U,V) (Helligkeit und Farbigkeit) verwendet werden, die durch eine lineare Abbildung mit den RGB-Werten verknüpft sind. Da das menschliche Auge Helligkeitsunterschiede viel eher registriert als Farbabweichungen, können die Matrizen U und V ohne sichtbare Qualitätseinbussen erheblich stärker reduziert werden als die Matrix Y. Daher ist dieses Farbmodell zur Bildkompression besser geeignet als das RGB-Modell. In der Regel wird von den Matrizen U und V nur jeder zweite Koeffizient übertragen, so dass die Datenmenge für diese beiden Komponenten auf 1/4 reduziert wird.

7.2 Allgemeiner Ablauf eines Kompressions-Algorithmus

Ein Algorithmus zur Bildkompression muss also Matrizen

$$B = (b_{i,j})_{i,j=1}^{n,m} \in (\mathbb{Z} \cap [0, 255])^{n \times m}$$

verarbeiten können, z.B. Grauwert-, Farb- oder Luminanzmatrizen. Aus diesen sollen neue Daten erzeugt werden, die weniger Speicherplatz benötigen. Umgekehrt muss dieser Algorithmus aus den von ihm produzierten Daten auch wieder eine Matrix

$$\tilde{B} = (\tilde{b}_{i,j})_{i,j=1}^{n,m} \in (\mathbb{Z} \cap [0, 255])^{n \times m}$$

rekonstruieren, so dass das daraus entstehende Bild möglichst gut mit dem zu B gehörende Bild übereinstimmt.
Man beachte, dass die Güte des reproduzierten Bildes eher subjektiv zu beurteilen ist, wogegen

die reine Datenreduktion mathematisch leicht zu erfassen ist.

Die dem Algorithmus zugrunde liegende Idee ist, einen Basiswechsel durchzuführen. So liegen die Informationen nach einer Transformation nicht mehr als Pixelwerte vor, sondern als Koeffizienten einer anderen Basis des $\mathbb{R}^2$. Diese Basis soll nun so gewählt werden, dass möglichst viele Koeffizienten sehr klein und damit vernachlässigbar sind. Ein typischer Algorithmus (wie z.B. das JPG-Verfahren) durchläuft drei Schritte:

1. *Transformation:* Berechnung der Koeffizienten der Daten in einer anderen Basis. Standardmethoden verwenden hier Fourier-, Kosinus- bzw. die Wavelet-Transformation. Dieser Übergang der Daten ist bis auf Rundungsfehler praktisch verlustfrei.

2. *Quantisierung:* Hier werden reellen Intervallen ganze Zahlen zugeordnet, d.h. die reellen Koeffizienten werden auf ganzzahlige abgebildet. Dieser Schritt ist notwendig, da reelle Zahlen, im Gegensatz zu ganzzahligen, relativ viel Speicherplatz benötigen. Alle betragsmässig kleinen Koeffizienten werden hier gleich 0 gesetzt.
   Bei der Quantisierung kann wieder Rücksicht auf die Wahrnehmungsfähigkeiten des Menschen genommen werden. Schnelle Helligkeitsänderungen werden schlechter wahrgenommen als ein langsamer Wechsel. Ebenso werden in horizontaler und vertikaler Richtung Helligkeitsschwankungen besser wahrgenommen, als in diagonaler Richtung. In Hinblick auf solche Erkenntnisse können dann die Quantisierungsintervalle gewählt werden. Hier optimale Ergebnisse zu erhalten, ist i.A. mit viel Aufwand verbunden.
   Die Quantisierung arbeitet in der Regel nicht mehr verlustfrei, d.h. insbesondere ist die Rekonstruktion des Bildes nach diesem Schritt nicht mehr exakt.

3. *Kodierung:* Die Koeffizienten können nun in einer optimierten Reihenfolge notiert werden. Optimiert bedeutet, dass möglichst viele Nullen blockweise auftreten und nicht vereinzelt, denn dann genügt es, nur die Länge der jeweiligen Nullsequenz anzugegeben (Lauflängenkodierung), z.B.

   statt 0,0,0,0,0 $\quad 5 \times 0.$

   Die sich ergebende Folge von ganzen Zahlen wird nun im letzten Schritt binär abgespeichert. Dabei sollen häufig auftretende Werte mit möglichst wenig Bits kodiert werden. Hierfür kann z.B. eine Huffman-Kodierung verwendet werden. Ist nach der Quantisierung z.B. die Koeffizientenfolge

   5 214 31 58 146 31 214 214 63

   entstanden, so kann diese mit der Abbildung

   $$\begin{aligned} 214 &\longmapsto 0 \\ 31 &\longmapsto 10 \\ 58 &\longmapsto 1100 \\ 5 &\longmapsto 1101 \\ 146 &\longmapsto 1110 \\ 63 &\longmapsto 1111 \end{aligned}$$

als folgende Bitsequenz gespeichert und daraus auch wieder rekonstruiert werden:

110101011001110100011111

Der Speicheraufwand wurde hierdurch von 9 Bytes (= 72 Bits) auf 23 Bits reduziert. Die Kodierung verläuft verlustfrei.

7.3 Kompression mit BLaC-Wavelets

Das Hauptziel des für diese Arbeit implementierten Algorithmus ist ein Vergleich von mit BLaC-Wavelets mit verschiedenen Parametern Δ komprimierten Bildern. Es wurde deshalb auf aufwändige Quantisierungs- und Kodierungsmethoden verzichtet. Ebenso werden nur Grauwertbilder betrachtet, und diese auch nur in der für unsere BLaC-Wavelets MSA passenden quadratischen Auflösung von $2^n + 1$ Zeilen und Spalten.
Sei nun ein solches Bild in einer $(2^n+1 \times 2^n+1)$-Matrix mit ganzzahligen Einträgen von 0 bis 255 gegeben. Wir wenden dann das Konstrukt der in Kapitel 6.2 vorgestellten zweidimensionalen Multiskalenanalyse an. Mit einer Nicht-Standard-Dekomposition wird das Bild soweit zerlegt, dass nur noch eine kleine quadratische Matrix mit Glätteanteilen und verschiedene Detailanteile vorhanden sind. Diese Dekomposition geschieht mit den zu einem vorher gegebenen Parameter Δ gehörenden Gleichungssystemen (s. Kapitel 4.4).
Eine solche Dekomposition ist relativ zeitintensiv. Weniger Aufwand benötigen hierfür sog. Schnelle-Wavelet-Transformations-Algorithmen. Hierauf jedoch genauer einzugehen würde den Rahmen dieser Arbeit sprengen. Für unsere Zwecke genügt die simple Nicht-Standard-Dekomposition.

Auch die Quantisierung fällt recht bescheiden aus: Lediglich alle Detailkoeffizienten, die betragsmässig kleiner als ein gegebenes $\varepsilon > 0$ sind, werden auf Null abgebildet. Aber auch eine solche Folge könnte mit einer effizienten Kodierung schon sehr reduziert gespeichert werden.

Da eine anschliessende Kodierung verlustfrei arbeiten würde, sind diese wenigen Schritte des Algorithmus schon ausreichend, um mit verschiedenen Parametern Δ reduzierte und wieder rekonstruierte Bilder vergleichen zu können.

7.4 Die Implementierung des Algorithmus

Nach dem Start des Programms *ImgMRA* erscheint das Hauptmenü-Fenster. Auf der linken Seite befinden sich diverse Einstellmöglichkeiten, die das Bild und die MSA betreffen. Rechts unten können Bilder geladen und gespeichert, sowie das Programm verlassen werden.

Mit einem Klick auf *Öffnen* lässt sich ein Bild laden. Dieses muss im BMP-Format vorliegen und die quadratische Grösse $2^n + 1 \times 2^n + 1$ besitzen. Das geladene Bild wird dann über diesen drei Schaltelementen angezeigt. Neben dem Hauptfenster erscheint ein zweites Fenster, in dem das Bild mit der Skalierungsfunktion des aktuellen Levels dargestellt wird. Die Fenstergrösse und der maximale Level werden dem geladenen Bild angepasst. Am unteren Bildrand wird der Name der Bilddatei angezeigt.

Mit Hilfe des Buttons *Speichern* wird das aktuelle Bild gespeichert. Dies geschieht jedoch nicht komprimiert, sondern ebenfalls nur im BMP-Format. Denn wie schon erwähnt, dient dieses

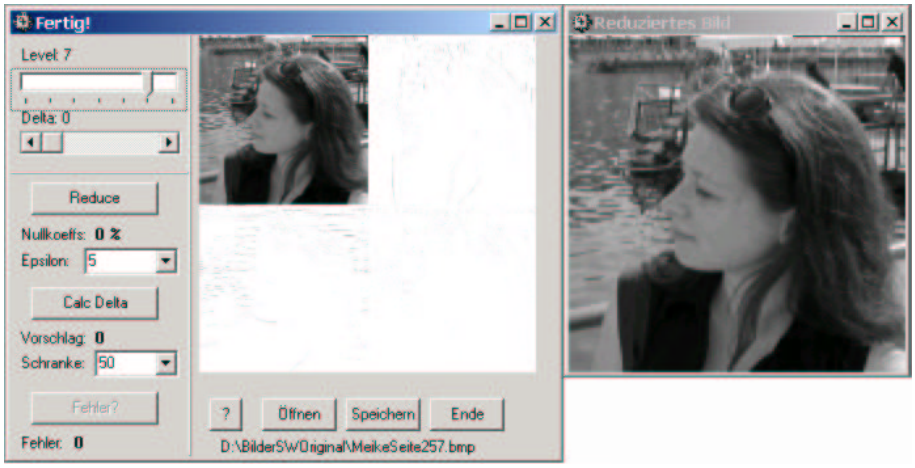


Abbildung 35: Das Programm ImgMRA

Programm nur dem Vergleich von mit unterschiedlichen Parametern reduzierten Bildern.

Auf der linken Seite befindet sich ganz oben ein Schieberegler für den Level. Dieser ist jedoch nur aktiviert, wenn ein Bild geladen wurde. Wie bei den anderen Programmen auch, setzt ein Bewegen dieses Reglers die MSA in Gang. Wird ein niedrigerer Level gewählt, so wird mit einer Nicht-Standard-Dekomposition der Glätteanteil des Bildes in vier Teile zerlegt - einen kleineren Glätteanteil und drei Detailanteile. Bei der Darstellung dieser Detailanteile wurde weiss als Farbe für Null gewählt und schwarz als Farbe für 255. Diese Inversion der Farbzuordnung wurde vorgenommen, da sich graue Farben besser auf weissem, als auf schwarzem Grund erkennen lassen.
Hinweis: Die Graustufen in den angezeigten Bildern sind nur ganzzahlige Werte zwischen 0 und 255. Intern rechnet das Programm jedoch mit Fliesskommazahlen doppelter Genauigkeit.

Der darunter liegende Schieberegler dient der Regulierung des Parameters Δ. Durch Bewegen des Reglers werden die Waveletkoeffizienten und Matrizen hier direkt berechnet.

Mit dem Button *Reduce* werden alle Detailkoeffizienten, die einen Wert kleiner als Epsilon haben, auf Null gesetzt. Zur Übersicht werden diese in dem Bild rot eingefärbt. Ist dies geschehen, so wird unter dem Button angezeigt, wieviel Prozent der Koeffizienten auf Null gesetzt wurden. Diese Schaltfläche ist nur aktiviert, wenn bereits ein Bild geladen wurde.
In dem darunter liegenden Feld lässt sich das Epsilon wählen. Es kann dort direkt eingetragen werden, oder aber mit einem Klick auf den Pfeil aus einer Liste übernommen werden.

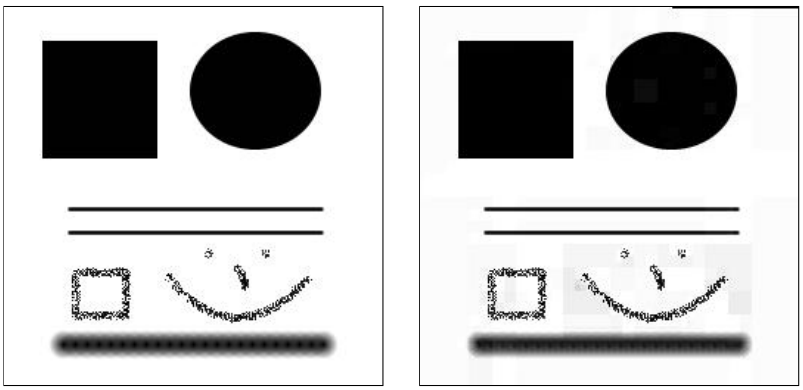
Die darunter liegenden Bedienelemente steuern das automatische Auswählen eines geeigneten Parameters Δ. Hierauf wird im nächsten Abschnitt näher eingegangen.

Der Button *Fehler?* dient einem objektiven Vergleich des reduzierten und des originalen Bildes. Nach Anwählen dieses Elementes wird in dem rechten Fenster das originale Bild angezeigt. Daraufhin werden die Beträge der Differenzen aller Bildpunkte des originalen und des reduzierten Bildes aufaddiert. Diese Fehlergesamtsumme wird anschliessend noch durch die Anzahl der Bildpunkte dividiert und ergibt somit die durchschnittliche Abweichung eines Bildpunktes des

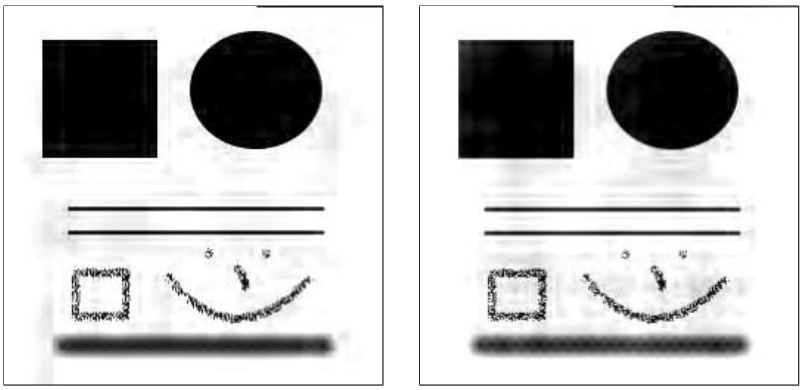
reduzierten Bildes zum Originalbildpunkt.

7.5 Beispiele

In Abbildung 36 ist ein Bild gegeben, das ideal mit den Haar-Wavelets komprimiert werden kann. Es sind wenig fliessende Farbübergänge vorhanden, dafür grosse einfarbige Flächen, die durch scharfe Kanten begrenzt sind. Selbst für $\varepsilon = 6$, wodurch 93 % der Detailkoeffizienten auf Null gesetzt wurden, sieht das rekonstruierte Bild fast wie das Originalbild aus. Lediglich an dem breiten weichen unteren Strich sind leichte Artefakte zu erkennen.
Für $\Delta = 0.5$ und $\Delta = 1$ ist die Rekonstruktion bei gleichem Prozentsatz an Nullkoeffizienten erheblich schlechter. Die Kanten sind stark verschmiert und auch der breite weiche Strich weist Artefakte auf. Für dieses Bild sind solche Parameter unbrauchbar. Das bestätigt auch der berechnete Fehler - er wächst mit steigendem Δ.



Originalbild | $\Delta = 0$, Fehler=0.893

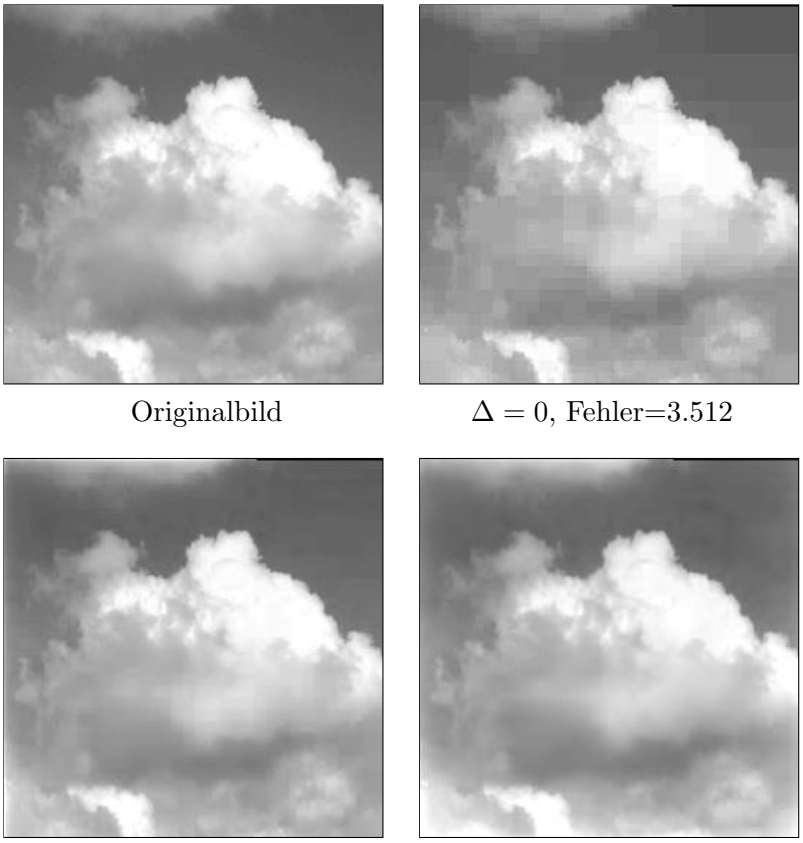


$\Delta = 0.5$, Fehler=2.259 | $\Delta = 1$, Fehler=4.759

Abbildung 36: Testbild mit 93 % Nullkoeffizienten

Die Wolken in Abbildung 37 haben dagegen sehr wenig scharfe Kanten. Das Bild besteht aus vielen weichen Farbverläufen - ideal für die linearen Wavelets. Trotz der vielen Nullkoeffizienten (93 %, $\varepsilon = 4$) ist das mit $\Delta = 1$ rekonstruierte Bild dem Original sehr ähnlich. Hier liefern die Haar-Wavelets ($\Delta = 0$) erheblich schlechtere Resultate. Über das ganze Bild hinweg sind störende Blockbildungen zu beobachten. Diese sind bei $\Delta = 0.5$ zwar schon erheblich reduziert, aber immer noch vorhanden.

Interessanterweise zeichnet der berechnete Fehler ein ganz anderes Bild. Für $\Delta = 1$ ist er eindeutig am grössten und sinkt mit fallendem Δ. Dies ist ein gutes Beispiel dafür, dass sich die Güte eines komprimierten Bildes nur schwer mathematisch ausdrücken lässt. Hier stehen subjektive Empfindungen im Vordergrund.



Originalbild | $\Delta = 0$, Fehler=3.512

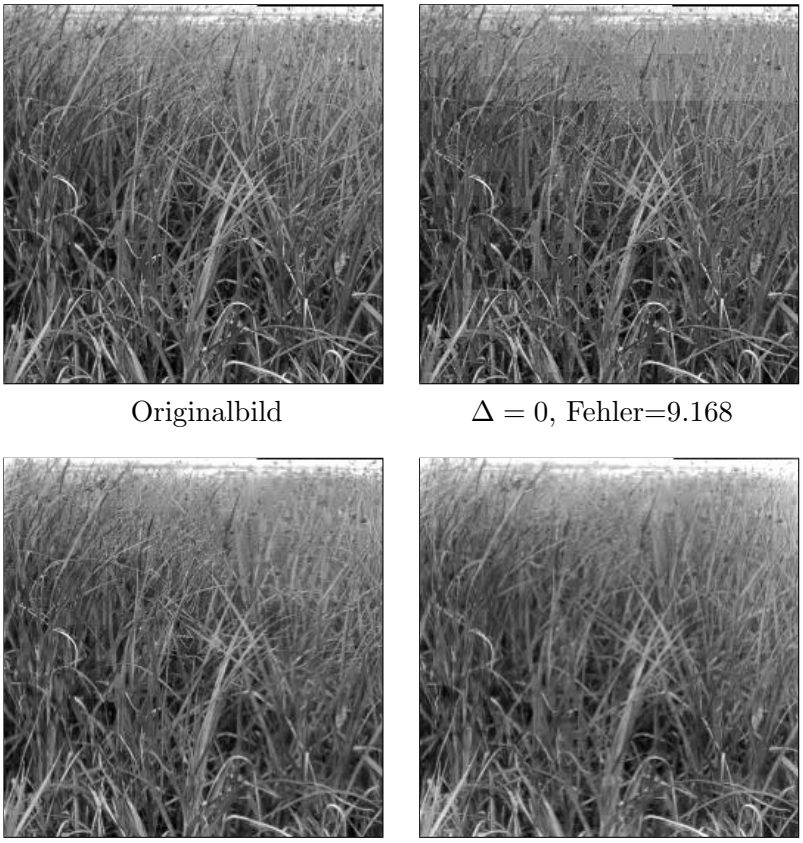
$\Delta = 0.5$, Fehler=4.584 | $\Delta = 1$, Fehler=10.459

Abbildung 37: Wolken mit 93 % Nullkoeffizienten

Abbildung 38 zeigt ein Schilfmotiv. Hier sind sowohl weichere Farbübergänge, z.B. am Himmel, als auch scharfe Kanten, z.B. bei den Pflanzenstengeln, vorhanden. Eine Kompression mit $\Delta = 0$ liefert wieder die bekannten Blockeffekte, dafür treten die Kanten bei den Stengeln scharf hervor. Die Blockeffekte verschwinden bei $\Delta = 1$, dafür wirkt das Bild verschwommen und unscharf. Hier liefert $\Delta = 0.5$ bessere Ergebnisse: Die Blockeffekte sind stark reduziert, und die Kantenschärfe ist trotzdem vorhanden.
Hier bestätigt der berechnete Fehler die subjektiven Eindrücke. Für Δ um 0.5 ist er am geringsten und steigt für kleinere und grössere Parameter.
Aufgrund der vielen Kanten lässt sich das Bild nicht so stark reduzieren wie die bisherigen beiden. Schon 60 % Nullkoeffzienten liefern hier eindeutige Vergleiche.

7.6 Die Wahl des Parameters Δ

Damit ein solches vorgestelltes Kompressionsverfahren für ein gegebenes Bild automatisch ablaufen kann, wird in diesem Abschnitt noch ein Algorithmus entwickelt, der einen optimalen



Originalbild | $\Delta = 0$, Fehler=9.168

$\Delta = 0.5$, Fehler=8.196 | $\Delta = 1$, Fehler=11.752

Abbildung 38: Schilfmotiv mit 60 % Nullkoeffizienten

Parameter Δ vorschlägt. Der Begriff „optimal“ ist hier mit Vorsicht zu geniessen, da die Ergebnisse hauptsächlich subjektiv zu bewerten sind. Es ist auch offensichtlich, dass ein solcher Parameter meistens nur lokal gut geeigent ist. Ideal wäre ein Kompressionsalgorithmus, der ein gegebenes Bild in verschiedene Bereiche aufteilt, und diese je nach Beschaffenheit mit einem eigenen Parameter komprimiert. Solch einen Algorithmus zu entwerfen übertritt jedoch den Rahmen dieser Arbeit. Ansätze, die in diese Richtung gehen, existieren jedoch schon beim JPEG2000-Algorithmus (s. [19]). Hier werden verschiedene Bildabschnitte nach gewissen Kriterien verschieden stark komprimiert. Der Algorithmus, der in dieser Arbeit entworfen wird, könnte dann z.B. für solche lokalen Bildteile eingesetzt werden, um den für sie idealen Parameter zu ermitteln.

7.6.1 Der Algorithmus

Anforderungen: Grosse einfarbige Flächen und scharfe Kanten sind besser mit der Haar-Skalierungsfunktion zu komprimieren. Weiche Farbverläufe und Kanten, unscharfe Bildteile dagegen besser mit der Hütchenfunktion. Der Algorithmus soll ein gegebenes Bild daraufhin untersuchen, und je nach Beschaffenheit ein $\Delta \in [0, 1]$ wählen.

Umsetzung: Um obige Anforderungen zu erfüllen, wird das Bild als Funktion

$$B : [0, 2^N + 1]^2 \longmapsto [0, 255]$$

aufgefasst. Einfarbige Flächen sind durch einen verschwindenden Gradienten gekennzeichnet, scharfe Kanten durch einen betragsmässig grossen Gradienten. Hierfür wird eine Schranke K vorgegeben: Alle Gradienten, die betragsmässig echt zwischen 0 und K liegen, werden als weiche Kante bzw. als weicher Farbverlauf interpretiert.
Um den Gradienten zu berechnen wird ein sogenanntes Differenzenbild erzeugt. Hierfür wird eine 3×3-Matrix $(M_{i,j})_{i,j=1}^{3}$ (genannt ***Sobel-Maske***) verwendet, die den diskreten Differenzenquotienten sowohl in x- als auch in y-Richtung simuliert. Diese wird wie folgt angewendet:
Ist $B(x, y)$ das Originalbild, so ergibt sich das Differenzenbild $B'(x, y)$ durch:

$$B'(x, y) = \sum_{i,j=-1}^{1} M_{i+2,j+2} B(x + i, y + j)$$

Hier ergeben sich jedoch Probleme an den Rändern, da auf Bildpunkte zugegriffen werden soll, die nicht mehr im Bild vorhanden sind. Wir berechnen daher das Differenzenbild nur im Innern, d.h. für $0 < x, y < 2^N + 1$. Dies ist für unsere Zwecke auch völlig ausreichend, da wir nur einen Überblick über die Kanten und Flächen gewinnen wollen, und auch bei den Punkten in der zweiten bzw. vorletzten Spalte und Zeile die Daten aus der ersten bzw. letzten Spalte und Zeile mit einbezogen werden.
Wir verwenden folgende Sobel-Maske:

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

In Abbildung 39 ist das Differenzenbild für das Bild „Meike“ zu sehen.

Abbildung 39: Das Differenzenbild zu „Meike“

Der Algorithmus geht nun folgendermassen vor: Nachdem das Differenzenbild berechnet wurde, wird dieses Punkt für Punkt durchlaufen. Falls der diskretisierte Gradient echt zwischen 0 und der Schranke K liegt, wird ein Zähler *Lin* um den Betragswert des Gradienten erhöht. Dieser

Zähler ermittelt sozusagen den Anteil der weichen Kanten und Farbübergänge. Gleichzeitig wird unabhängig davon ein zweiter Zähler *Ges* um den gleichen Wert erhöht, dieser ermittelt die Betragssumme der Gradienten des ganzen Bildes.
Der Parameter Δ wird anschliessend als prozentualer Anteil von *Lin* in *Ges* gewählt.

```
double function DeltaWahl(Bild B)
    Lin := 0; Ges := 0;
    {inneren Punkte des Bildes zeilen- und spaltenweise von 1 bis 2^N − 1 durchlaufen }
    for y=1 to 2^N − 1
        for x=1 to 2^N − 1
            {Variable zur Berechnung des neuen Bildpunktes}
            SobelSum := 0;
            {Jetzt wird die Doppelsumme zur Anwendung der Sobel-Maske berechnet }
            for i=-1 to 1
                for j=-1 to 1
                    {Sobel-Maske auf Punkt x,y anwenden }
                    SobelSum := SobelSum + SobelMaske(i+2,j+2)*Bild(x+i,y+j);
                end for {j}
            end for {i}
            {Die entsprechenden Zähler erhöhen }
            if abs(SobelSum) > 0 and abs(SobelSum) < K then
                Lin := Lin + abs(SobelSum);
            end if
            Ges := Ges + abs(SobelSum);
        end for {x}
    end for {y}
    {Schliesslich den Parameter delta als Prozentsatz des linearen Zählers ermitteln }
    delta := Lin/Ges;
    return(delta);
end procedure
```

Die Wahl der Schranke K beeinflusst auch den Parameter Δ. Hier muss auf Erfahrungswerte und sicherlich auch auf subjektives Empfinden zurückgegriffen werden. In der folgenden Tabelle sind für verschiedene Bilder diverse Schranken getestet, und die ermittelten Parameter notiert worden.

Bild\K	K=25	K=50	K=75	K=100
Meike	0.39	0.66	0.81	0.89
Schilf	0.18	0.50	0.72	0.85
Wolken	0.76	0.78	0.84	0.90
Testbild	0.02	0.03	0.04	0.14

Bei Bildern, wo die Tendenz des Parameters Δ von vorne herein recht eindeutig ist, so z.B. bei den *Wolken* oder bei dem *Testbild*, hat eine Änderung der Schranke K nicht viel Auswirkungen auf das von dem Algorithmus vorgeschlagene Δ. Bei dem Bild *Meike* und *Schilf* dagegen schon, hier treten Spannen bis zu 0.67 auf. In einem solchen Fall muss subjektiv entschieden

werden. Meiner Meinung nach liefert K=50 für alle Bilder durchaus akzeptable Ergebnisse.

Résumee: In der Bildkompression liefern BLaC-Skalierungsfunktionen mit einem Parameter Δ ungleich Null und ungleich Eins oft bessere Ergebnisse als die reinen Haar- und linearen Skalierungsfunktionen. Hier ist man auch nicht auf Darstellung der Daten durch Addition von Skalierungsfunktion und Wavelets angewiesen, d.h. der dort auftretende Fehler ist hier nicht von Belang. In Verbindung mit z.B. dem JPEG2000-Algorithmus könnte der Parameter mit dem hier vorgestellten Algorithmus lokal ideal gewählt werden. So könnte die Kompressionsrate bei gleich bleibendem oder evtl. sogar besserem Ergebnis erhöht werden, um weniger Daten speichern zu müssen. Somit ist die Bildkompression eine gute Anwendung der BLaC-Wavelets.

8 MSA auf hierarchischen Triangulierungen

Die zweite Anwendung der Multiskalenanalyse mit nicht ineinander geschachtelten Räumen sind hierarchische Triangulierungen. Bonneau nennt hier als Beispiel die Behandlung grosser unstrukturierter Datenmengen, wie sie z.B. bei meteorologischen oder geologischen Messungen auftreten (s. [3, 4]). Mit solchen Daten zu arbeiten benötigt sowohl viel Zeit, als auch viel Speicherplatz. Es gibt viele Wavelet-basierte Methoden, die effizient bei der Visualisierung von Daten in verschiedenen Detailstufen oder bei der Kompression von grossen Datenmengen arbeiten. Alle diese Methoden benötigen jedoch eine Hierarchie von Dreiecksnetzen, die durch Subdivision miteinander verbunden sind. Ein solches Dreiecksnetz, auf dem gegebene Daten definiert sind, kann durch rekursive Subdivision-Verfahren von einem Basisnetz aus erreicht werden. Dies können z.B. Haar-Wavelets über einer 4-to-1-split-Triangulierung sein. Eine so entstehende Hierarchie kann als einfacher Baum gespeichert werden, im Fall der 4-to-1-split Triangulierung durch eine Quadtree-Struktur.
Aber eben diese Notwendigkeit der Subdivision-Konnektivität verhindert eine Anwendung der Wavelet-basierten Methoden auf irregulären Dreiecksnetzen, wie sie z.B. durch Kantenentfernung nach Delauny oder Verschmelzung von Ecken entstehen (s. Abbildung 40). Mit einer MSA mit nicht ineinander geschachtelten Räumen kann nun eine Wavelet-ähnliche Dekomposition entwickelt werden, die auf stückweise konstanten Daten auf irregulären Dreiecksnetzen operiert. Eine solche Dekomposition erweitert bestehende Resultate der Haar-Wavelets über 4-to-1-split-Triangulierungen und erlaubt aber auch eine exakte Rekonstruktion der Daten.
Diese Wavelet-ähnliche Transformation erwartet als Eingabe eine stückweise konstante Funktion auf einem Dreicksnetz und liefert eine Approximation dieser Funktion auf einer einfacheren Triangulierung desselben Gebiets, sowie eine Reihe von Detail-Koeffizienten, die den Fehler zur Eingabefunktion wiedergeben. Die Hierarchie der Triangulierung die hierbei entsteht, kann i.A. nicht mehr als einfacher Baum gespeichert werden. Die hierfür notwendig kompliziertere Graphenstruktur wird in Kapitel 8.3 erklärt.

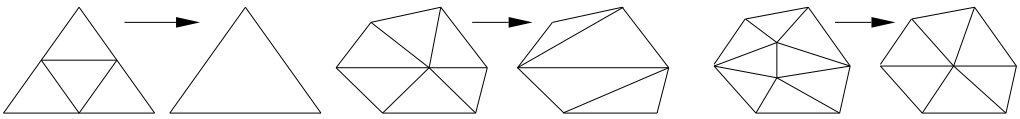


Abbildung 40: Lokale Dreiecksdezimierung

8.1 Lokale Dekomposition und die zugehörigen Approximationsräume

Ausgangspunkt der MSA die hier entwickelt werden soll, ist eine Triangulierung eines Gebietes Ω und eine auf ihr definierte stückweise konstante Funktion f_N, wobei N der Level des feinsten Dreiecksnetzes sei. Diese Funktion kann z.B. selbst eine Approximation einer Funktion $f \in L^2(\Omega)$ sein. Ferner sei $T_{n,k}$ das k-te Dreieck der Triangulierung im n-ten Level, und $K(n)$ eine Indexmenge, so dass Ω die disjunkte Vereinigung der in $K(n)$ aufgeführten Dreiecke vom Level

n ist, d.h.:

$$\Omega = \bigcup_{k \in K(n)} T_{n,k}$$

Die gegebene Funktion ist dabei wie folgt definiert:

$$f_N(x) = \begin{cases} x_k^N & : \quad x \in T_{N,k} \\ 0 & : \quad sonst \end{cases} \qquad \text{mit } x_k^N \in \mathbb{R}, k \in K(N)$$

Man erhält nun eine Triangulierung vom Level n, indem man Kanten der Triangulierung vom Level $n+1$ weglässt, und die Regionen der Dreiecke, die adjazent zu den entfernten Kanten sind, neu trianguliert. Welche Kanten entfernt werden, und wie neu trianguliert wird, soll hier nicht diskutiert werden. Dies wird als bekannt vorausgesetzt. In diesem Abschnitt soll das Problem behandelt werden, wie Skalierungs- und Waveletfunktionen definiert, und wie die entsprechenden Koeffizienten berechnet werden können.
Oben genannte Konstruktion definiert disjunkte Polygonregionen $R_{n,j}$ im Gebiet Ω, eine für jedes Set von adjazenten entfernten Kanten. Gibt $\nu_j(n+1)$ die Anzahl der Dreiecke in der Region $R_{n,j}$ an, so werden $\nu_j(n+1)$ Dreiecke durch $\nu_j(n)$ Dreiecke ersetzt, die das gleiche Gebiet bedecken. Hierbei soll $\nu_j(n+1) > \nu_j(n)$ gelten.

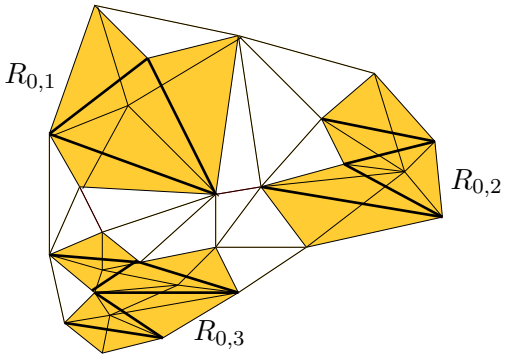


Abbildung 41: Entfernung von Kanten und Neutriangulierung definiert Polygonregionen

In Abbildung 41 sind drei solche Polygonregionen zu sehen. Die Haarlinien innerhalb der eingefärbten Flächen wurden entfernt, die fetten Linien zeigen die neue Triangulierung. Eine auf diesem Wege definierte Polygonregion besteht sowohl aus Dreiecken vom Level n, als auch aus Dreiecken vom Level $n+1$. $L(n,j)$ und $L(n+1,j)$ seien Indexmengen, die die Indizes dieser Dreiecke vom Level n und $n+1$ enthalten, es gilt:

$$R_{n,j} = \bigcup_{k \in L(n+1,j)} T_{n+1,k} = \bigcup_{k \in L(n,j)} T_{n,k}$$

Die Dreiecke $T_{n,k}$ heissen ***Elterndreiecke*** von $R_{n,j}$, die $T_{n+1,j}$ heissen ***Kinderdreiecke*** von $R_{n,j}$.

Für den Auflösungslevel n ist der Approximationsraum V_n die Menge der auf Ω definierten Funktionen, die auf den Dreiecken $T_{n,k}$ konstant sind:

$$V_n := \{f : \Omega \longmapsto \mathbb{R}, f(x) = x_k^n, \quad x \in T_{k,n}, x_k^n \in \mathbb{R}, k \in K(n)\} \tag{34}$$

Da die Verfeinerung der Dreicksnetze nicht auf Subdivisions-Schemata basiert, sind die Räume V_n nicht ineinander enthalten, $V_n \not\subset V_{n+1}$.

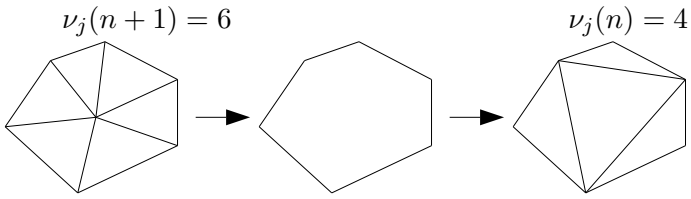


Abbildung 42: Neutriangulierung einer Polygonregion $R_{n,j}$

8.2 Die Analyse- und Synthese-Matrizen

Wie in (34) schon beschrieben, ist der Approximationsraum V_n die Menge aller auf den einzelnen Dreiecken vom Level n konstanten Funktionen. Die Skalierungsfunktionen $\varphi_k^n, k \in K(n)$ sollen eine Basis dieses Raumes bilden. Es liegt also folgende Definition nahe:

$$\varphi_k^n(x) = 1_{T_{n,k}}(x) = \begin{cases} 1 & : \quad x \in T_{n,k} \\ 0 & : \quad sonst \end{cases} \tag{35}$$

Ausserhalb der Regionen $R_{n,j}$ ändern sich die Skalierungsfunktionen zwischen Level n und $n+1$ nicht. Bei einem Analyse-Schritt müssen hier also weder Glätte- noch Detailkoeffizienten ermittelt werden. Es ist daher sinnvoll die Analyse- und Synthese-Matrizen nur lokal für jede Region $R_{n,j}$ zu berechnen.
Anders als bei den BLaC-Wavelets werden hier nicht die Synthese- sondern die Analyse-Matrizen konstruiert. D.h. die Gestalt der Wavelets ψ_i^n und der approximierten Skalierungsfunktionen $\tilde{\varphi}_i^n$ steht hier noch nicht fest. Die Synthese-Matrizen können z.B. durch (13) berechnet werden.

8.2.1 Berechnung von A

Sei nun f eine Funktion aus $L^2(\Omega)$ und $f_n \in V_n$ die Approximation dieser Funktion zum Level n. Im Folgenden soll der Einfachheit halber auf die Betrachtung der unterschiedlichen Regionen verzichtet werden. Die Anzahl der Dreiecke in einem Level sei dann durch $\nu(n)$ gegeben.
Wir wollen nun einen Glätte-Operator definieren, der die Approximation der Eingabefunktion vom Level $n+1$ auf die Approximation vom Level n abbildet. Dieser ist durch eine rechteckige $\nu(n) \times \nu(n+1)$-Matrix gegeben. Diese Matrix A^n soll so gewählt werden, dass f_n immer die beste L^2-Approximation von f_{n+1} in V_n ist. Dies ist der Fall, falls gilt:

$$< \varphi_i^n, f_n >_{L^2} = < \varphi_i^n, f_{n+1} >_{L^2} \quad , i = 1, ..., \nu(n)$$

Um A^n zu erhalten wird dieser Ansatz weiter entwickelt:

$$\begin{pmatrix} < \varphi_1^n, f_n > \\ \vdots \\ < \varphi_{\nu(n)}^n, f_n > \end{pmatrix} = \begin{pmatrix} < \varphi_1^n, f_{n+1} > \\ \vdots \\ < \varphi_{\nu(n)}^n, f_{n+1} > \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} < \varphi_1^n, \sum_{i=1}^{\nu(n)} x_i^n \varphi_i^n > \\ \vdots \\ < \varphi_{\nu(n)}^n, \sum_{i=1}^{\nu(n)} x_i^n \varphi_i^n > \end{pmatrix} = \begin{pmatrix} < \varphi_1^n, \sum_{i=1}^{\nu(n+1)} x_i^{n+1} \varphi_i^{n+1} > \\ \vdots \\ < \varphi_{\nu(n)}^n, \sum_{i=1}^{\nu(n+1)} x_i^{n+1} \varphi_i^{n+1} > \end{pmatrix}$$

$$
\begin{aligned}
&\Leftrightarrow \begin{pmatrix} \sum_{i=1}^{\nu(n)} x_i^n < \varphi_1^n, \varphi_i^n > \\ \vdots \\ \sum_{i=1}^{\nu(n)} x_i^n < \varphi_{\nu(n)}^n, \varphi_i^n > \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{\nu(n+1)} x_i^{n+1} < \varphi_1^n, \varphi_i^{n+1} > \\ \vdots \\ \sum_{i=1}^{\nu(n+1)} x_i^{n+1} < \varphi_{\nu(n)}^n, \varphi_i^{n+1} > \end{pmatrix} \\
&\Leftrightarrow \begin{pmatrix} < \varphi_1^n, \varphi_1^n > & \cdots & < \varphi_1^n, \varphi_{\nu(n)}^n > \\ \vdots & & \vdots \\ < \varphi_{\nu(n)}^n, \varphi_1^n > & \cdots & < \varphi_{\nu(n)}^n, \varphi_{\nu(n)}^n > \end{pmatrix} \begin{pmatrix} x_1^n \\ \vdots \\ x_{\nu(n)}^n \end{pmatrix} \\
&\quad = \begin{pmatrix} < \varphi_1^n, \varphi_1^{n+1} > & \cdots & < \varphi_1^n, \varphi_{\nu(n+1)}^{n+1} > \\ \vdots & & \vdots \\ < \varphi_{\nu(n)}^n, \varphi_1^{n+1} > & \cdots & < \varphi_{\nu(n)}^n, \varphi_{\nu(n+1)}^{n+1} > \end{pmatrix} \begin{pmatrix} x_1^{n+1} \\ \vdots \\ x_{\nu(n+1)}^{n+1} \end{pmatrix} \\
&\Leftrightarrow \quad G_n(x^n) = (< \varphi_i^n, \varphi_j^{n+1} >)(x^{n+1}) \\
&\Leftrightarrow \quad (x^n) = G_n^{-1}(< \varphi_i^n, \varphi_j^{n+1} >)(x^{n+1}) \\
&\Leftrightarrow \quad A^n = G_n^{-1}(< \varphi_i^n, \varphi_j^{n+1} >)
\end{aligned}
$$

G_n ist die Gram-Schmidt-Matrix, und das Skalarprodukt berechnet sich wie folgt:

$$
\begin{aligned}
< \varphi_i^n, \varphi_j^{n+1} > &= \int_{\mathbb{R}^2} (\varphi_i^n \varphi_j^{n+1})(x,y)dxdy = \int_{\mathbb{R}^2} (1_{T_{n,i}} 1_{T_{n+1,j}})(x,y)dxdy \\
&= \int_{T_{n,i} \cap T_{n+1,j}} 1 dxdy = area(T_{n,i} \cap T_{n+1,j})
\end{aligned}
$$

Entsprechend gilt:

$$
< \varphi_i^n, \varphi_j^n > = area(T_{n,i} \cap T_{n,j}) = \begin{cases} area(T_{n,i}) & : \quad i = j \\ 0 & : \quad sonst \end{cases}
$$

Damit ist $G_n = \begin{pmatrix} area(T_{n,1}) & & 0 \\ & \ddots & \\ 0 & & area(T_{n,\nu(n)}) \end{pmatrix}$ und $G_n^{-1} = \begin{pmatrix} \frac{1}{area(T_{n,1})} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{area(T_{n,\nu(n)})} \end{pmatrix}$

Es ergibt sich somit:

$$
A^n = (\frac{area(T_{n,i} \cap T_{n,j})}{area(T_{n,i})})_{i,j} \quad , i = 1, ..., \nu(n), j = 1, ..., \nu(n+1)
$$

Damit lassen sich nun die neuen Glättekoeffizienten berechnen:

$$
\begin{pmatrix} x_1^n \\ \vdots \\ x_{\nu(n)}^n \end{pmatrix} = A^n \begin{pmatrix} x_1^{n+1} \\ \vdots \\ x_{\nu(n+1)}^{n+1} \end{pmatrix}
$$

Beispiel: Für die Verfeinerung in Abbildung 43 ergibt sich:

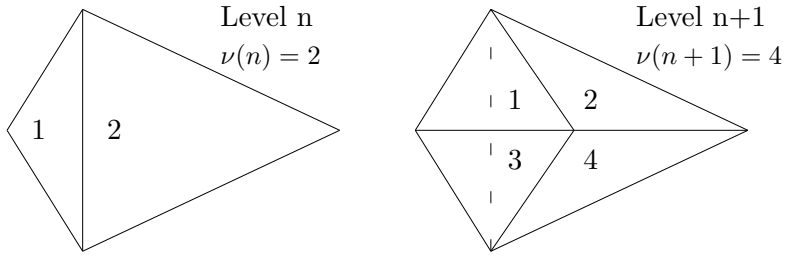


Abbildung 43: Beispiel

$$\begin{pmatrix} x_1^n \\ x_2^n \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{6} & \frac{1}{3} & \frac{1}{6} & \frac{1}{3} \end{pmatrix}}_{A^n} \begin{pmatrix} x_1^{n+1} \\ x_2^{n+1} \\ x_3^{n+1} \\ x_4^{n+1} \end{pmatrix},$$

also

$$\begin{aligned} x_1^n &= \frac{1}{2}x_1^{n+1} + \frac{1}{2}x_3^{n+1} \\ x_2^n &= \frac{1}{6}x_1^{n+1} + \frac{1}{3}x_2^{n+1} + \frac{1}{6}x_3^{n+1} + \frac{1}{3}x_4^{n+1}. \end{aligned}$$

Die Gewichtung eines Koeffizienten x_i^{n+1} einer Funktion φ_i^{n+1} in der Linearkombination für φ_j^n entspricht dem Flächenanteil des Dreiecks $T_{n+1,i}$ an dem des Dreiecks $T_{n,j}$.

8.2.2 Berechnung von B

Nun muss noch der Operator bestimmt werden, der die Koeffizienten (x^{n+1}) auf Fehlerkoeffizienten (y^n) abbildet, die den Fehler zwischen f_{n+1} und f_n wiederspiegeln. Dieser ist durch eine $\omega(n) \times \nu(n+1)$-Matrix B^n gegeben. Hierbei war $\omega(n) = \nu(n+1) - \nu(n)$, d.h. die Anzahl der Waveletbasisfunktionen entspricht der Anzahl der eingesparten Dreiecke. Um B^n zu berechnen, werden folgende Bedingungen gestellt:

1. Semi-Orthogonalität der MSA: $A^n G_{n+1}^{-1} (B^n)^T = 0$
2. Orthonormalität der Wavelets: $B^n G_{n+1}^{-1} (B^n)^T = I$

Die zweite Bedingung erhöht die numerische Stabilität.

Sehen wir uns die erste Bedingung genauer an, es gilt:

$$\underbrace{A^n}_{\in \mathbb{R}^{\nu(n) \times \nu(n+1)}} \underbrace{G_{n+1}^{-1}}_{\in \mathbb{R}^{\nu(n+1) \times \nu(n+1)}} \underbrace{(B^n)^T}_{\in \mathbb{R}^{\nu(n+1) \times \omega(n)}} \in \mathbb{R}^{\nu(n) \times \omega(n)}$$

Allerdings ist $B^n \in \mathbb{R}^{\omega(n)\times\nu(n+1)}$, d.h. wir haben $\nu(n)\omega(n)$ Gleichungen für $\nu(n+1)\omega(n)$ Unbekannte, also $\nu(n+1)\omega(n) - \nu(n)\omega(n) = \underbrace{(\nu(n+1)-\nu(n))}_{=\omega(n)}\omega(n) = \omega^2(n)$ zu wenig Gleichungen, um die Matrix B^n vollständig daraus zu bestimmen. Um dieses Gleichungssystem trotzdem zu lösen, fixieren wir eine $\omega(n)\times\omega(n)$-Untermatrix von B^n als Einheitsmatrix. Dadurch haben wir genau $\omega^2(n)$ Unbekannte weniger und sind so in der Lage die restlichen Einträge von B^n zu bestimmen:

$$B^n = \begin{pmatrix} b_{1,1} & \cdots & b_{1,\nu(n+1)} \\ \vdots & & \vdots \\ b_{\omega(n),1} & \cdots & b_{\omega(n),\nu(n+1)} \end{pmatrix} := \begin{pmatrix} 1 & & 0 & b_{1,\omega(n)+1} & \cdots & b_{1,\nu(n+1)} \\ & \ddots & & \vdots & & \vdots \\ 0 & & 1 & b_{\omega(n),\omega(n)+1} & \cdots & b_{\omega(n),\nu(n+1)} \end{pmatrix}$$

Das Gleichungssystem hat dann folgende Gestalt:

$$\begin{aligned}
& A^n G_{n+1}^{-1} (B^n)^T \\
= & \begin{pmatrix} \frac{area(T_{n,1}\cap T_{n+1,1})}{area(T_{n,1})} & \cdots & \frac{area(T_{n,1}\cap T_{n+1,\nu(n+1)})}{area(T_{n,1})} \\ \vdots & & \vdots \\ \frac{area(T_{n,\nu(n)}\cap T_{n+1,1})}{area(T_{n,\nu(n)})} & \cdots & \frac{area(T_{n,\nu(n)}\cap T_{n+1,\nu(n+1)})}{area(T_{n,\nu(n)})} \end{pmatrix} \begin{pmatrix} \frac{1}{area(T_{n,1})} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{area(T_{n,\nu(n)})} \end{pmatrix} (B^n)^T \\
= & \begin{pmatrix} \frac{area(T_{n,1}\cap T_{n+1,1})}{area(T_{n,1})area(T_{n+1,1})} & \cdots & \frac{area(T_{n,1}\cap T_{n+1,\nu(n+1)})}{area(T_{n,1})area(T_{n+1,\nu(n+1)})} \\ \vdots & & \vdots \\ \frac{area(T_{n,\nu(n)}\cap T_{n+1,1})}{area(T_{n,\nu(n)})area(T_{n+1,1})} & \cdots & \frac{area(T_{n,\nu(n)}\cap T_{n+1,\nu(n+1)})}{area(T_{n,\nu(n)})area(T_{n+1,\nu(n+1)})} \end{pmatrix} \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \\ b_{1,\omega(n)+1} & \cdots & b_{\omega(n),\omega(n)+1} \\ \vdots & & \vdots \\ b_{1,\nu(n+1)} & \cdots & b_{\omega(n),\nu(n+1)} \end{pmatrix} \\
= & \begin{pmatrix} \alpha_{1,1} + \sum_{k=\omega(n)+1}^{\nu(n+1)} b_{1,k}\alpha_{1,k} & \cdots & \alpha_{1,\omega(n)} + \sum_{k=\omega(n)+1}^{\nu(n+1)} b_{\omega(n),k}\alpha_{1,k} \\ \vdots & & \vdots \\ \alpha_{\nu(n),1} + \sum_{k=\omega(n)+1}^{\nu(n+1)} b_{1,k}\alpha_{\nu(n),k} & \cdots & \alpha_{\nu(n),\omega(n)} + \sum_{k=\omega(n)+1}^{\nu(n+1)} b_{\omega(n),k}\alpha_{\nu(n),k} \end{pmatrix} = 0 \\
\iff & \sum_{k=\omega(n)+1}^{\nu(n+1)} b_{i,k} \frac{area(T_{n,j}\cap T_{n+1,k})}{area(T_{n,j})area(T_{n+1,k})} = -\frac{area(T_{n,j}\cap T_{n+1,i})}{area(T_{n,j})area(T_{n+1,i})}
\end{aligned}$$

für $i = 1, ..., \omega(n)$, $j = 1, ..., \nu(n)$

Hierbei ist: $\alpha_{i,j} = \frac{area(T_{n,i}\cap T_{n+1,j})}{area(T_{n,i})area(T_{n+1,j})}$

Um jetzt noch die zweite Bedingung $B^n G_{n+1}^{-1}(B^n)^T = I$ zu erfüllen, kann ein normales Gram-Schmidt-Orthonormalisierungsverfahren auf die Zeilen von B^n angewendet werden (s. z.B. [17]). Solche Verfahren sind zwar für grosse Matrizen instabil, die hier auftretenden Matrizen sind jedoch nur von kleiner Dimension, begrenzt durch den Grad der entfernten Kanten. Darüber hinaus sichert die Orthonormalität von B^n die numerische Stabilität der Algorithmen. Dies ist gerade bei so grossen Datenmengen enorm wichtig.

Durch die oben beschriebene Wahl der Analyse-Matrizen A^n und B^n ist sichergestellt, dass

1. die Approximation f_n zum Level n die beste L_2-Approximation von f_{n+1} in V_n ist,
2. die Approximation $\tilde{f}_n$ die beste L_2-Approximation von f_{n+1} zum Level n in $\tilde{V}_n$ ist.

Die Synthese-Matrizen P^n und Q^n können folgendermassen aus A^n und B^n berechnet werden (siehe Gleichung (13)):

$$P^n = G_{n+1}^{-1}(A^n)^T(A^n G_{n+1}^{-1}(A^n)^T)^{-1}$$
$$Q^n = G_{n+1}^{-1}(B^n)^T(\underbrace{B^n G_{n+1}^{-1}(B^n)^T}_{=I})^{-1} = G_{n+1}^{-1}(B^n)^T$$

Die ursprünglichen Daten können durch $x^{n+1} = P^n x^n + Q^n y^n$ rekonstruiert werden. Die Berechnung dieser Koeffizienten ist bei der Analyse und der Synthese exakt. Wo treten jetzt die Funktionen $\tilde{f}_n, \tilde{\varphi}^n$ und die Wavelets ψ^n auf?
Die Funktion $\tilde{f}_n$ ist aus $\tilde{V}_n$, einem Unterraum von V_{n+1}, der durch die Funktionen $\tilde{\varphi}_i^n$, $i = 1, .., \nu(n)$ erzeugt wird. Sie ist demnach gegeben durch:

$$\tilde{f}_n = \sum_{i=1}^{\nu(n)} x_i^n \tilde{\varphi}_i^n$$

Die Funktionen $\tilde{\varphi}^n$ ergeben sich hierbei durch die Matrix P^n als Linearkombinationen der φ^{n+1}:

$$\tilde{\varphi}^n = \varphi^{n+1} P^n$$

Addiert man die Funktion $\tilde{f}_n$ zu den Wavelets ψ^n, die den Fehler wiederspiegeln, ergibt dies die ursprüngliche Funktion f_{n+1}. Die Wavelets berechnen sich hierbei durch Matrix Q^n als Linearkombinationen der φ^{n+1}:

$$\psi^n = \varphi^{n+1} Q^n$$

Die Wavelets lassen sich hier nicht von vorne herein konstruieren. Sie ergeben sich, anders als bei den BLaC-Wavelets, erst durch die Beschaffenheit der Räume V_n.
Die Funktion f_n erhält man nun, indem man die Koeffizienten von $\tilde{\varphi}^n$ für die Funktionen φ^n verwendet. Hier entsteht auch der Approximationsfehler der nicht ineinander geschachtelten Räume, der hier aber nicht genauer untersucht werden soll.

Beispiel: In Abbildung 44 ist eine Region $R_{n,j}$ mit vier Dreiecken zu sehen. Rechts davon ist $\tilde{f}_n$ als Linearkombination der φ^{n+1} zu sehen, darunter die fehlenden Details in Form der Wavelets, die ebenfalls Linearkombinationen der φ^{n+1} sind. Ganz rechts oben befindet sich dann die zu $\tilde{f}_n$ isomorphe Funktion f_n.

8.3 Graphenstruktur der hierarchischen Triangulierungen

Wie bereits erwähnt, soll hier nun dargestellt werden, wie diese Hierarchie der Triangulierungen als Graph gespeichert werden kann. Hierfür benötigt dieser Graph N Level, einen für jeden Auflösungslevel der Triangulierung von Ω. Die Knoten eines Levels n entsprechen den Polygonregionen $R_{n,j}$. Jeder Knoten $R_{n,j}$ enthält zwei Listen mit Zeigern:

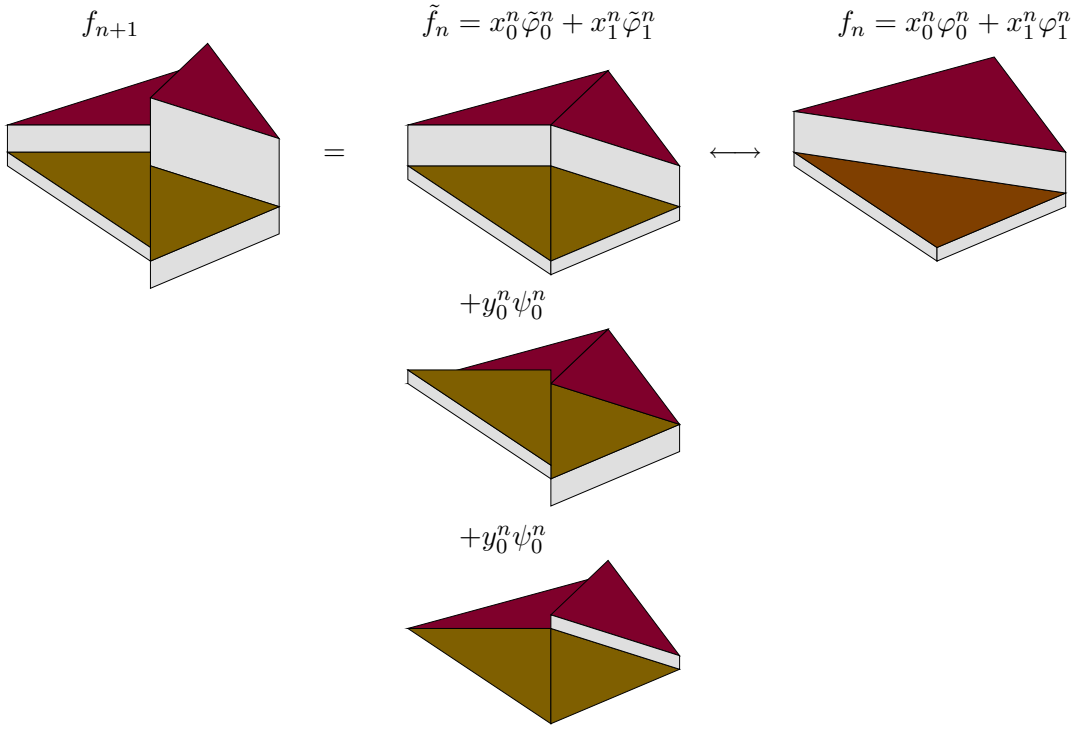


Abbildung 44: Wavelet-Dekomposition

1. Für die Kinderdreiecke $T_{n+1,k}$,
2. für die Elterndreiecke $T_{n,k}$.

Ausserdem enthält ein Knoten $R_{n,j}$ einen Zeiger auf einen Knoten $R_{m,j'}$ eines gröberen Levels $m \leq n$, falls $R_{m,j'}$ ein Kinderdreieck enthält, welches Elterndreieck von $R_{n,j}$ ist.
Werden feinere Triangulierungen z.B. durch ein 4-split-1-Verfahren erreicht, so hat jeder Knoten nur einen Zeiger auf ein Elterndreieck und vier Zeiger auf Kinderdreiecke. Als Graphenstruktur ergibt sich dann einfach ein Wald von Quadtree-Konstruktionen.

In den Abbildungen 45 und 46 ist ein Beispiel einer hierarchischen Triangulierung und ihrer Graphenstruktur zu sehen. Auf der linken Seite von Abbildung 45 ist eine Hierarchie von Triangulierungen von Level 3 bis zu Level 0 zu sehen. Die einzelnen Polygonregionen, die durch das Entfernen von Kanten entstehen, sind farbig markiert. Die gestrichelten Linien deuten an, wie die Region im nächsten Level neu trianguliert wird. Diese Regionen sind nochmal auf der rechten Seite der Abbildung zu sehen, dort mit den dazugehörigen Zeigern.
Der daraus resultierende Graph ist in Abbildung 46 zu sehen. Die gefüllten Kreise stehen für Dreiecke, und die abgerundeten Rechtecke markieren Gruppen von Dreiecken, die in den Polygonregionen zusammengehören. Ist ein Kreis nicht in einem solchen Rechteck, so ist das dazugehörige Dreieck nicht von der Neutriangulierung betroffen. Die Kanten des Graphen sind die oben erwähnten Zeiger auf Kinder- und Elterndreiecke und evtl. auf Knoten eines gröberen Levels.

Abgespeichert werden hierbei nach einer Dekomposition nur eine Liste der Dreiecke vom Level 0 und die Knoten, die den Regionen $R_{n,j}$ entsprechen. Dies entspricht der Speicherung der

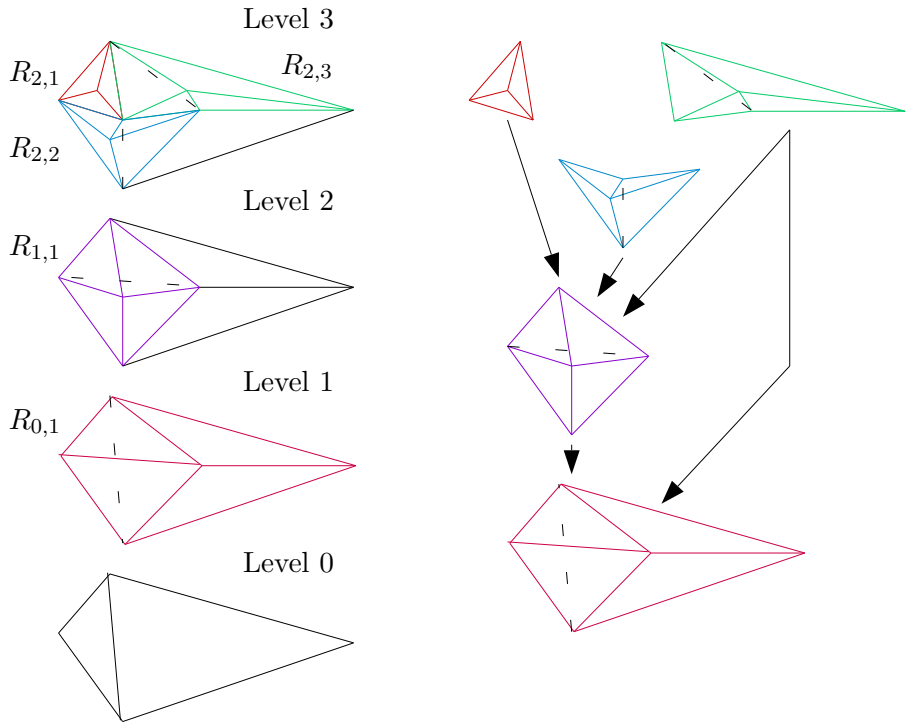


Abbildung 45: Beispiel einer hierarchischen Triangulierung und ihrer Verkettung

Glättekoeffizienten des niedrigsten Levels und der dazugehörigen Details.
Soll die Triangulierung zu einem bestimmten Level rekonstruiert werden, so werden zuerst die Dreiecke bei Level 0 gezeichnet. Um dann von einem Level n zum nächst höheren, also $n+1$, zu gelangen, betrachtet man die Knoten, die zu Level $n+1$ gespeichert wurden, und ersetzt die Dreiecke, auf die diese Knoten zeigen, durch die nächst feineren.

8.4 Algorithmen

Mit der in Kapitel 8.3 vorgestellten Graphenstruktur und den in Kapitel 8.2.1 und 8.2.2 vorgestellten Analyse-Matrizen ist es nun möglich folgende drei Algorithmen zu implementieren.

1. Wavelet Dekomposition auf den Triangulierungen
   - Starte mit den Koeffizienten x_N einer Funktion f bei der feinsten Triangulierung $T_{N,k}$
   - Berechne für jeden Auflösungslevel n und für jede Region $R_{n,j}$ mit den Analyse-Matrizen A und B die Glättungskoeffizienten x_n und die Detail- Koeffizienten y_n
   - Speichere diese statt der korrespondierenden feineren Koeffizienten x_{n+1}
   - Erhalte am Ende nur die Koeffizienten x_0 der Approximation von f in der gröbsten Auflösung und die Detailkoeffizienten für alle Regionen $R_{n,j}$ und für alle Level $n = 0, ..., N-1$
2. Rekonstruktion der Funktion in einer bestimmten Auflösung
   - Starte mit dem Output der Wavelet-Dekomposition

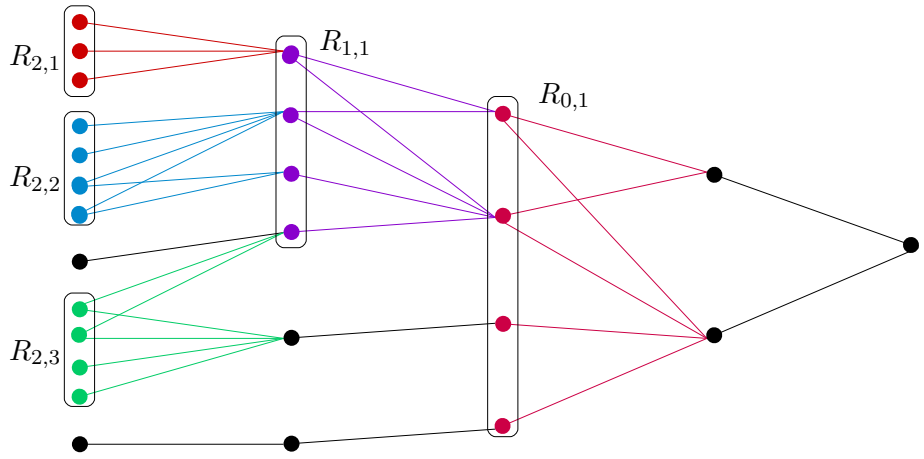


Abbildung 46: Der zugehörige Graph

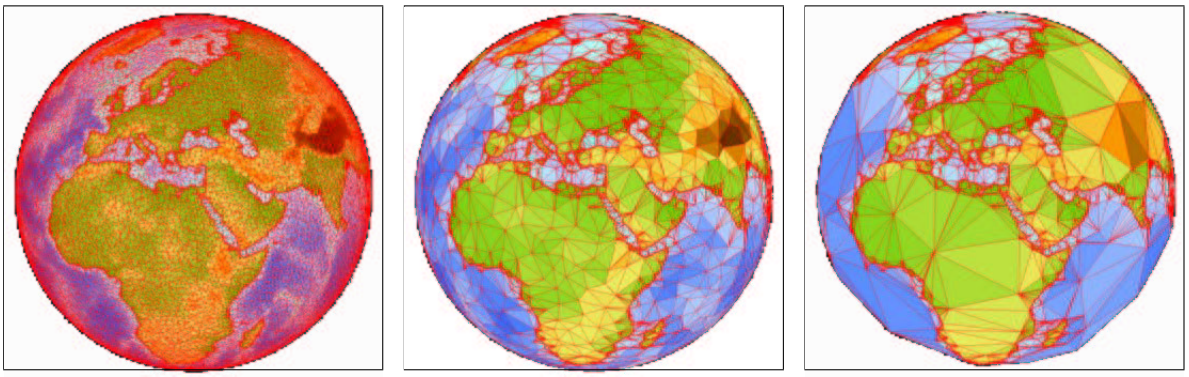
- Für jeden Auflösungslevel $k < n$ und für alle $R_{k,j}$ berechne mit den Synthese- Matrizen P und Q die Skalierungskoeffizienten x_{k+1} aus den gröberen x_k und aus den Detailkoeffizienten y_k

3. Fehlerkonstruktion zu vorgegebener Fehlerschranke e

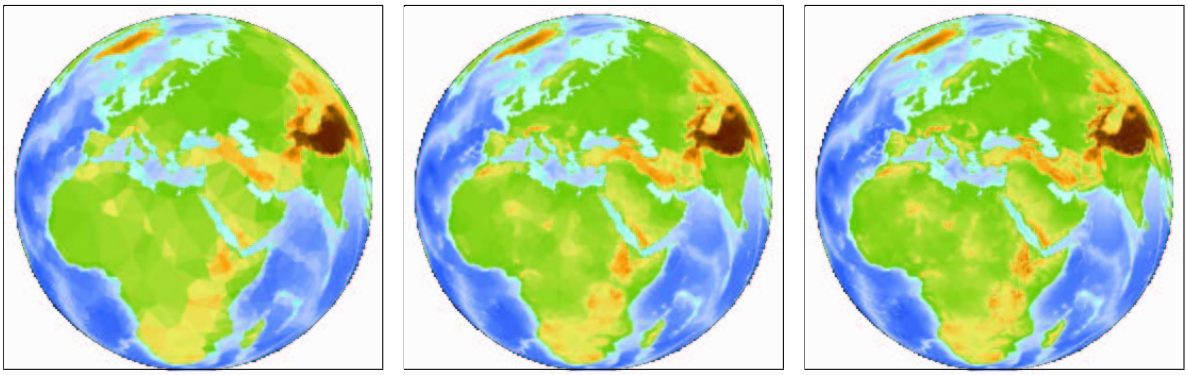
   - Starte mit dem Output der Wavelet-Dekomposition und einer Fehlerschranke ε
   - Wähle den Subgraph der Graphenstruktur, der Regionen $R_{n,j}$ enthält, die Detailkoeffizienten enthalten, deren Absolutwert grösser als ε ist.
   - Berechne mit den Matrizen P und Q auf diesen Regionen die Skalierungskoeffizienten x_{n+1}

8.5 Beispiele

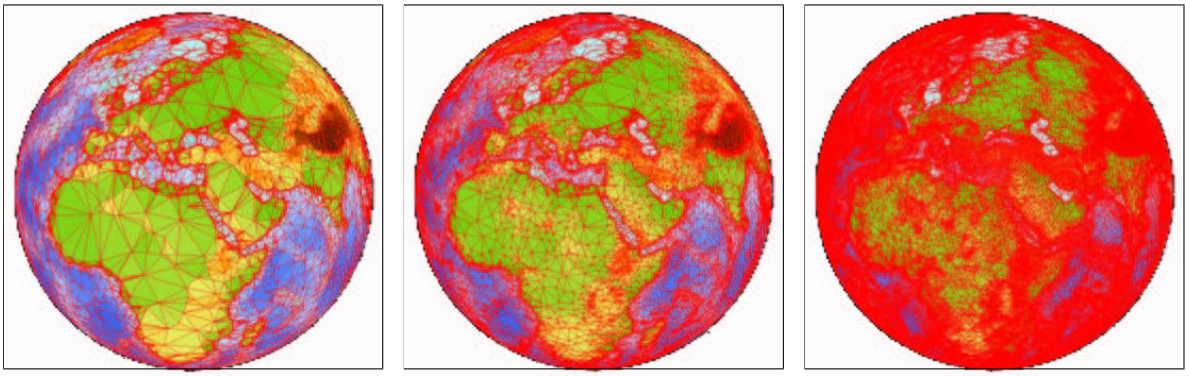
Die folgenden Beispiele sind im Wesentlichen dem Artikel von Bonneau in [18] entnommen. Die Abbildungen zeigen die partielle Rekonstruktion geometrischer Daten in verschiedenen Leveln. Die ursprüngliche Datenmenge besteht aus ungefähr 1.300 000 Kanten.



(a) Level 28, 122130 Kanten (b) Level 16, 46790 Kanten (c) Level 0, 44950 Kanten



(d) 60000 Kanten (von 1.3M) (e) 15000 Kanten (von 1.3M) (f) 122130 Kanten (von 1.3M)



(g) wie (d), mit Kanten (h) wie (e), mit Kanten (i)wie (f), mit Kanten

Literatur

[1] G.P. Bonneau, *Multiresolution Analysis with Non-Nested Spaces* aus A.Iske, E.Quak, M.S. Floater, *Tutorials on Multiresolution in Geometric Modelling*, Springer 1996

[2] G.P. Bonneau *BLAC-wavelets: a multiresolution analysis with non-nested spaces*, http://www-imagis.imag.fr/ Georges-Pierre.Bonneau/

[3] G.P. Bonneau, *Multiresolution Analysis on Irregular Surface Meshes*, http://www-imagis.imag.fr/ Georges-Pierre.Bonneau/

[4] G.P. Bonneau, A. Gerussi, *Hierarchical decomposition of datasets on irregular surface meshes*, http://www-imagis.imag.fr/ Georges-Pierre.Bonneau/

[5] R.E. Edwards, *Fourier Series, a modern Introduction, Vol. I*, Holt, Rinehart and Winston, 1967

[6] Otto Forster, *Analysis I-III*, Vieweg, 1996

[7] Heinz H. Gonska, *On approximation in spaces of continuous functions*, BULL. AUSTRAL. MATH. SOC. VOL. 28 (1983), 411-432

[8] Olaf Hansen, *Einführung in die Theorie und Anwendung der Wavelets*, Logos, 2000

[9] Janser, Luther, Otten, *Computergraphik und Bildverarbeitung*, Vieweg, 1996

[10] Koecher, *Lineare Algebra und analytische Geometrie*, Springer, 1997

[11] Christian Kurmann, *Interaktive Visualisierung von Volumendaten unter Einbeziehung einer exponentiellen Dämpfung und optimierter Farbraumkonvertierung*, ETH Zürich, Computer Graphics, Diplomarbeit SS 1996

[12] Louis/Maaß/Rieder, *Wavelets*, Teubner, 1998

[13] Y.Meyer, *Wavelets and Operators*, Cambridge University Press, 1992

[14] W. Rudin, *Real and Complex Analysis*, McGraw-Hill Book Company, 1966

[15] J. Stoer, *Numerische Mathematik 1 und 2*, Springer, 1999

[16] Stollnitz, DeRose, Salesin, *Wavelets for Computer Graphics*, Morgan Kaufman, 1996

[17] G. Strang, *Introduction to Applied Mathematics*, Wellesley-Cambridge-press, 1996

[18] http://www-imagis.imag.fr/ Georges-Pierre.Bonneau/

[19] www.JPEG2000.org