Randbehandlung bei Wavelets für Faltungsnetzwerke

Felix Blanke

Geboren am 30. August 1999 in Frankfurt am Main

27. August 2021

Bachelorarbeit Mathematik

Betreuer: Prof. Dr. Jochen Garcke

Zweitgutachter: Prof. Dr. Christian Bauckhage

Institut für Numerische Simulation

Betreuer am Fraunhofer SCAI: Moritz Wolter

Fraunhofer-Institut für Algorithmen und Wissenschaftliches Rechnen

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Inhaltsverzeichnis

Abkürzungen

CelebA Large-scale Celeb Faces Attributes (Datensatz)

CNN Faltungsnetzwerk (*englisch*: Convolutional Neural Network)

DCT Diskrete Cosinus-Transformation

DWT Diskrete Wavelet-Transformation

FFHQ Flickr Faces High Quality (Datensatz)

FFT Schnelle Fourier-Transformation (*englisch*: Fast Fourier Transform)

FIR-Filter Filter mit endlicher Impulsantwort (*englisch*: Finite Impulse Response Filter)

FWT Schnelle Wavelet-Transformation (*englisch*: Fast Wavelet Transform)

GAN Generative Adversarial Network

GPU Grafikprozessor (*englisch*: Graphical Processing Unit)

IFWT Inverse Schnelle Wavelet-Transformation (*englisch*: Inverse Fast Wavelet Transform)

LSUN Large-scale Scene UNderstanding bedrooms (Datensatz)

Einführung

In den computergestützten Wissenschaften sind Transformationen eines Signals in seine Frequenzbestandteile allgegenwärtig. Sie ermöglichen es, verschiedene Frequenzbänder eines Signals voneinander zu trennen. Die wohl bekannteste dieser Transformationen ist die Fouriertransformation.

Allerdings zeigt sich in der Bildverarbeitung ein zentraler Nachteil Fourier-transformierter Signale: Das transformierte Signal enthält keinerlei Ortsinformationen mehr. Dies ist ein großes Problem, wenn beispielsweise die Frequenzbänder von Vorder- und Hintergrund grundverschieden sind, da diese so nicht differenziert werden können.

Eine Lösung bieten die sogenannten Wavelets. Diese Standardmethode der Signalverarbeitung [SN96; Jl01; TM02] ermöglicht es, Signale im Frequenzbereich darzustellen, ohne alle Zeitinformationen zu verlieren. Der Begriff Wavelets leitet sich vom französischen *ondelettes*, wörtlich übersetzt „kleine Wellen“, ab. Wie der Name bereits andeutet, sind Wavelets wellenartige Funktionen, die sowohl in Zeit als auch Frequenz lokal sind.

Aufgrund dieser Vorzüge haben Wavelets vielfältige Anwendungsfälle im maschinellen Lernen. Sie werden beispielsweise zum Entfernen störender Artefakte wie Dunstschwaden aus Landschaftsfotografien [Fu+21], zur Generierung synthetischer Bilder [Gal+21], zur Erkennung von synthetisch generierten hyper-realistischen Bildern (DeepFakes) [Wol+21] oder in Pooling-Schichten in Faltungsnetzwerken [WG21; WL18] eingesetzt.

In der Regel sind die Signale, die beim maschinellen Lernen verarbeitet werden, endlich. Wie wir im Laufe dieser Arbeit besprechen werden, erfordert dies eine explizite Behandlung der Signalränder, um Wavelet-Transformationen anwenden zu können. In der Praxis geschieht die Randbehandlung oft durch eine Fortsetzung der Signale. Jedoch können dabei je nach Fortsetzungsmethode verschiedene Probleme auftreten, weshalb wir als Alternative zur Signalfortsetzung die Randbehandlung mittels sogenannter Randfilter besprechen werden. Diese bieten den Vorteil, dass einerseits weniger Randartefakte auftreten und andererseits unabhängig von der Wahl des Wavelets die Wavelet-Transformierte eines endlichen Signals dieselben Maße hat. Im Falle einer Verwendung von Faltungsnetzwerken muss dadurch bei einem Wechsel des verwendeten Wavelets nicht die Netzwerkarchitektur angepasst werden.

Die Beschreibung der Wavelet-Theorie bedient sich mathematischer Grundlagen in verschiedenen Bereichen wie z.B. der Signalverarbeitung und der Funktionalanalysis, weshalb im Rahmen dieser Arbeit nur ein Abriss dessen möglich ist. An einigen Stellen werden wir Konstruktionen nur skizzieren können, um den Rahmen dieser Arbeit nicht zu sprengen. Insbesondere die signaltheoretischen Grundlagen in den Abschnitten 1.1 und 1.2 führen wir mit dem klaren Fokus auf die spätere Anwendung zur Konstruktion einer orthonormalen Wavelet-Basis ein. Auch werden Verallgemeinerungen wie biorthonormale Filterbänke und Wavelets nicht behandelt.

Aufbau der Arbeit

Diese Arbeit gliedert sich in vier Kapitel. Im ersten Kapitel arbeiten wir die theoretischen Grundlagen der Wavelet-Theorie auf. Dazu führen wir zunächst Filter (Abschnitt 1.1) und darauf aufbauend Filterbänke (Abschnitt 1.2) ein. Wir beschreiben Anforderungen an Filterbänke, um mit diesen eine in Zeit und Frequenz lokale Orthonormalbasis von $L^2(\mathbb{R})$ zu konstruieren – die sogenannte Wavelet-Basis. Weiter lernen wir mit der Schnellen Wavelet-Transformation eine effiziente Möglichkeit des Basiswechsels in diese Basis kennen. Dabei gehen wir davon aus, dass die Leserin und der Leser mit grundlegenden Konzepten der Funktionalanalysis wie Hilberträumen vertraut ist.

Wir betrachten in Kapitel 2 Möglichkeiten, die im ersten Kapitel entwickelte Theorie auf endliche Signale anwendbar zu machen. Dazu führen insbesondere sogenannte Gram-Schmidt-Randfilter als Möglichkeit der Randbehandlung ein und vergleichen diese mit der Randbehandlung durch symmetrische Fortsetzung anhand von Visualisierungen.

Im dritten Kapitel untersuchen wir dann Anwendungen der Gram-Schmidt-Randfilter. Dazu zeigen wir zunächst, dass Wavelet-Transformationen auf zweidimensionale Signale anwendbar sind, indem wir diese entlang jeder Achse transformieren. Anschließend erweitern wir einen Anwendungsfall von Wavelets im maschinellen Lernen mittels Gram-Schmidt-Randfilter auf längere Wavelets und zeigen, dass somit deutliche Verbesserungen gegenüber Haar-Wavelets erzielt werden können.

Schließlich geben wir in Kapitel 4 eine kurze Zusammenfassung, eine Diskussion der Anwendung von Randfiltern im maschinellen Lernen und einen Ausblick.

Eigener Anteil

Folgende wesentliche Beiträge finden sich in dieser Arbeit:

- Aufarbeitung der Grundlagen der Wavelet-Theorie (Kapitel 1),
- Zusammenführung verschiedener Ansätze [HV94; Jl01] zur Konstruktion von Gram-Schmidt-Randfiltern (Kapitel 2),
- Implementation der eindimensionalen und separablen zweidimensionalen Schnellen Wavelet-Transformation und Wavelet-Paket-Transformation mit Randbehandlung durch Gram-Schmidt-Randfilter in Python (vgl. Abschnitt 3.1),
- Vergleich der Gram-Schmidt-Randfilter mit der Randbehandlung durch symmetrische Fortsetzung im Eindimensionalen anhand von Zeit-Frequenz-Darstellungen,
- Erweiterung der Ansätze von Wolter u. a. [Wol+21] zur Erkennung von DeepFakes auf beliebige orthonormale Wavelets durch Randbehandlung mittels Gram-Schmidt-Randfiltern; experimentelle Untersuchung des Ansatzes für verschiedene Wavelets und Datensätze und Vergleich der Ergebnisse mit Wolter u. a. [Wol+21],
- Diskussion der Anwendung der Gram-Schmidt-Randfilter im maschinellen Lernen sowie Ausblick.

Danksagung

An dieser Stelle möchte ich meine Dankbarkeit für die Unterstützung, die ich erfahren durfte, ausdrücken. Zunächst bedanke ich mich bei Herrn Prof. Dr. Jochen Garcke für die Betreuung und die Möglichkeit, an so einem forschungsnahen Thema arbeiten zu können. Des Weiteren möchte ich Herrn Prof. Dr. Christian Bauckhage für die Übernahme der Zweitkorrektur danken. Dank geht auch an das Fraunhofer SCAI dafür, dass ich das Rechencluster für die Experimente nutzen durfte, und an meinen Betreuer am Fraunhofer SCAI, Moritz Wolter, für die anregenden Diskussionen und die Unterstützung.

Außerdem möchte ich meinen Dank gegenüber allen Personen ausdrücken, die an der Entwicklung freier Software wie LaTeX, *Python* oder *NumPy* mitwirken, ohne die die Erstellung meiner Arbeit kaum möglich gewesen wäre.

Nicht zuletzt gilt mein herzlicher Dank meiner Familie, insbesondere meinem Vater Thomas Blanke, und Marena Richter für die Geduld und die konstruktiven Korrekturanmerkungen.

1. Grundlagen

Den theoretischen Ausgangspunkt für diese Arbeit bilden Signale $f \in L^2(\mathbb{R}^d)$, also auf $\mathbb{R}^d$ quadratintegrierbare Funktionen. Das Ziel dieses Kapitels ist, für $L^2(\mathbb{R}^d)$ eine Orthonormalbasis aus Wavelet-Funktionen zu konstruieren. Dies sind Funktionen, die sowohl in der Zeit als auch in der Frequenz lokal sind. Dafür wenden wir uns zunächst zeitdiskreten Signalen zu und führen in den Abschnitten 1.1 und 1.2 einige Konzepte aus der Signalverarbeitung ein. Dazu orientieren wir an Strang und Nguyen [SN96]. In Abschnitt 1.3 werden wir dann die Brücke zurück zu kontinuierlichen Signalen schlagen und mit diesen Mitteln die Wavelet-Basis sowie ein schnelles Verfahren zu ihrer Berechnung beschreiben.

Wir wählen zur Beschreibung von Signalen $L^2(\mathbb{R}^d)$ aus zwei Gründen. Einerseits bildet $L^2(\mathbb{R}^d)$ mit dem Skalarprodukt

$$\langle f, g \rangle := \int_{\mathbb{R}^d} f(x)g(x)\,\mathrm{d}x \quad \text{für } f, g \in L^2(\mathbb{R}^d)$$

einen Hilbertraum [Wer18]. Dies ermöglicht es, Werkzeuge der Funktionalanalysis anzuwenden. Andererseits entspricht eine endliche induzierte L^2-Norm in der Signaltheorie einem Signal mit endlicher Energie [Mer20].

Wir beschränken uns für dieses Kapitel auf die Verarbeitung eindimensionaler Signale. Hier ist als Intuition die Vorstellung eines Audiosignals hilfreich. In Kapitel 3 erweitern wir die Theorie auf zweidimensionale Signale, um auch Bilder untersuchen zu können.

1.1. Filter

Definition 1.1.1. Ein (diskretes) *Signal* $\mathbf{x}$ ist ein Element aus dem Folgenraum

$$l^2(\mathbb{Z}) := \left\{ \mathbf{x} \colon \mathbb{Z} \to \mathbb{R} \;\middle|\; \sum_{n \in \mathbb{Z}} |\mathbf{x}(n)|^2 < \infty \right\}.$$

Wir schreiben kurz l^2 für $l^2(\mathbb{Z})$. Der verwendete Folgenbegriff nutzt $\mathbb{Z}$ zur Indizierung, damit die Folge sowohl für steigenden als auch fallenden Index nicht endet.

Bemerkung. Analog zum theoretischen Signalbegriff aus der Einleitung dieses Kapitels entspricht eine Folge aus l^2 in der Signaltheorie einem diskreten Signal mit endlicher Energie [Mer20].

Wir können diesen diskreten Signalbegriff als das Ergebnis einer Auswertung eines kontinuierlichen Signals $f \in L^2(\mathbb{R}^d)$ auf einem regelmäßigen Gitter interpretieren [SN96], wie in Abbildung 1.1 schematisch dargestellt ist.

Ein wichtiges Signal ist der sogenannte Einheitsimpuls δ, auch Dirac-Impuls genannt:

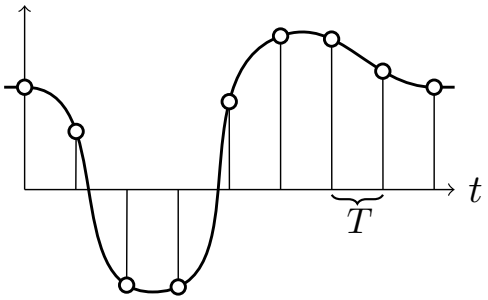


Abbildung 1.1.: Beispielsignal auf $\mathbb{R}$ sowie diskretisiertes Signal zur Sampling-Rate T

Definition 1.1.2. Der *Einheitsimpuls* $\delta \in l^2$ ist definiert als

$$\delta(n) := \begin{cases} 1 & n = 0, \\ 0 & \text{sonst.} \end{cases}$$

Definition 1.1.3. Ein *Filter* $\mathbf{H}\colon l^2 \to l^2$ ist ein linearer Operator, der ein Signal mittels Faltung mit einem festen Signal $\mathbf{h} \in l^2$ filtert, also in ein anderes Signal transformiert. Es gilt also für ein Signal $\mathbf{x} \in l^2$

$$[\mathbf{H}\mathbf{x}](n) = [\mathbf{h} * \mathbf{x}](n) = \sum_{k\in\mathbb{Z}} \mathbf{h}(k)\mathbf{x}(n-k).$$

Das feste Signal $\mathbf{h}$ wird „Impulsantwort“ genannt, da die Anwendung eines Filters auf den Einheitsimpuls dieses Signal erzeugt. Die Länge eines Filters definieren wir als die Länge seiner Impulsantwort.

Definition 1.1.4. Ein bedeutender Spezialfall sind die Filter mit endlicher Impulsantwort (*englisch*: Finite Impulse Response Filter, FIR-Filter).

Diese Filter sind wichtig, da wir einerseits Filter über ihre Impulsantwort codieren wollen, was in Anwendungen nur für Filter mit endlicher Impulsantwort möglich ist, und andererseits sonst die Berechnung der Faltung mit der Impulsantwort nicht in endlicher Zeit durchführbar wäre.

Definition 1.1.5. Wir nennen einen Filter *kausal*, wenn seine Impulsantwort für alle negativen Indizes null ist.

Beispiel 1.1.6. Zwei Filter, auf die wir im Laufe dieser Arbeit wiederholt zurückkommen werden, sind das *Gleitende Mittel* $\mathbf{H}_0$ und die *Gleitende Differenz* $\mathbf{H}_1$. Das Gleitende Mittel ist ein kausaler FIR-Filter mit der Impulsantwort

$$\mathbf{h}_0(n) = \begin{cases} \frac{1}{2} & n = 0 \text{ oder } n = 1, \\ 0 & \text{sonst.} \end{cases} \tag{1.1}$$

Somit entspricht die Ausgabe des Filters dem Mittelwert des aktuellen Signalwerts $\mathbf{x}(n)$ und des vorherigen Wertes $\mathbf{x}(n-1)$:

$$\mathbf{H}_0\mathbf{x}(n) = \frac{1}{2}\mathbf{x}(n) + \frac{1}{2}\mathbf{x}(n-1)$$

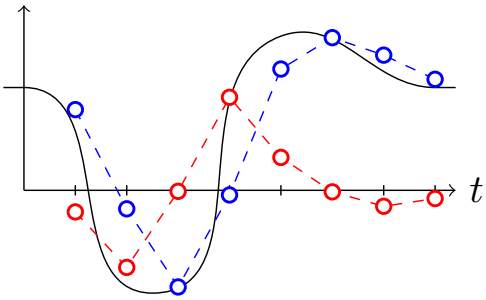


Abbildung 1.2.: Anwendung des Gleitenden Mittels (blau) und der Gleitenden Differenz (rot) auf das Beispielsignal aus Abbildung 1.1.

Die Gleitende Differenz hat die Impulsantwort

$$\mathbf{h}_1(n) = \begin{cases} \frac{1}{2} & n = 0, \\ -\frac{1}{2} & n = 1, \\ 0 & \text{sonst.} \end{cases} \tag{1.2}$$

und ist somit ebenfalls ein kausaler FIR-Filter. Die Ausgabe des Filters ist die Differenz des aktuellen Signalwerts $\mathbf{x}(n)$ und des vorherigen Werts $\mathbf{x}(n-1)$:

$$\mathbf{H}_1\mathbf{x}(n) = \frac{1}{2}\mathbf{x}(n) - \frac{1}{2}\mathbf{x}(n-1)$$

Eine Visualisierung der Anwendung beider Filter auf das Beispielsignal aus Abbildung 1.1 findet sich in Abbildung 1.2.

Filter sind ein zentrales Objekt dieser Arbeit. Die bisherige Darstellung eines Filters arbeitet im „Zeitbereich“: ein Filter wird mit seiner Impulsantwort assoziiert. Dann entspricht die Anwendung des Filters der Faltung des Eingabesignals mit der Impulsantwort.

Dem gegenüber gibt es noch zwei weitere Darstellungsarten für einen Filter, die wir in dieser Arbeit verwenden werden: die Darstellung im „Frequenzbereich“ mittels der diskreten Fouriertransformation und die Darstellung als unendliche sogenannte Toeplitz-Matrix (vgl. Abschnitt 1.1.2).

1.1.1. Filterdarstellung mittels Fouriertransformation

Definition 1.1.7. Die Fourier-transformierte eines Signals $\mathbf{x} \in l^2$ ist

$$X(\omega) = \sum_{k\in\mathbb{Z}} \mathbf{x}(k)\mathrm{e}^{-\mathrm{i}k\omega}.$$

Lemma 1.1.8. *Für $\mathbf{x} \in l^2$ existiert eine eindeutige Fourier-transformierte $X \in L^2([-\pi,\pi])$.*

Beweis. Dies folgt direkt aus dem Satz von Fischer-Riesz [z.B. Bea04, Theorem 13.12]. □

Beispiel 1.1.9. Die Frequenzantwort des Gleitenden Mittels $\mathbf{H}_0$ ist

$$H_0(\omega) = \sum_{k\in\mathbb{Z}} \mathbf{h}_0(k)\mathrm{e}^{-\mathrm{i}k\omega} = \frac{1}{2} + \frac{1}{2}\mathrm{e}^{-\mathrm{i}\omega} = \frac{\mathrm{e}^{\mathrm{i}\omega/2} + \mathrm{e}^{-\mathrm{i}\omega/2}}{2}\mathrm{e}^{-\mathrm{i}\omega/2} = \cos\left(\frac{\omega}{2}\right)\mathrm{e}^{-\mathrm{i}\omega/2}.$$

Beispiel 1.1.10. Die Frequenzantwort der Gleitenden Differenz $\mathbf{H}_1$ ist

$$H_1(\omega) = \sum_{k\in\mathbb{Z}} \mathbf{h}_1(k)\mathrm{e}^{-\mathrm{i}k\omega} = \frac{1}{2} - \frac{1}{2}\mathrm{e}^{-\mathrm{i}\omega} = \frac{\mathrm{e}^{\mathrm{i}\omega/2} - \mathrm{e}^{-\mathrm{i}\omega/2}}{2}\mathrm{e}^{-\mathrm{i}\omega/2} = \mathrm{i}\sin\Big(\frac{\omega}{2}\Big)\mathrm{e}^{-\mathrm{i}\omega/2}.$$

Lemma 1.1.11 (Strang und Nguyen [SN96, S. 5])**.** *Die Fourier-transformierte eines gefilterten Signals entspricht der Multiplikation des Fourier-transformierten Eingabesignals mit der Fouriertransformation der Impulsantwort des Filters.*

Somit können wir einen Filter mit der Impulsantwort $\mathbf{h}$ auch mit der sogenannten *Frequenzantwort*

$$H(\omega) = \sum_{k\in\mathbb{Z}} \mathbf{h}(k)\mathrm{e}^{-\mathrm{i}k\omega},$$

also der Fourier-transformierten dieser Impulsantwort, assoziieren.

1.1.2. Filterdarstellung mittels Toeplitz-Matrizen

Definition 1.1.12. Eine (unendliche) Toeplitz-Matrix $A = (a_{ij})_{i,j\in\mathbb{Z}}$ ist eine Matrix, deren Diagonalen alle den gleichen Eintrag haben, also

$$a_{ij} = a_{kl} \quad \text{für alle } i,j,k,l \in \mathbb{Z} \text{ mit } i-j = k-l.$$

Lemma 1.1.13. *Die Anwendung eines Filters* $\mathbf{H}$ *auf ein diskretes Signal* $\mathbf{x} \in l^2$ *entspricht dem (unendlichen) Matrix-Vektor-Produkt der Toeplitz-Matrix* $H = (h_{ij})_{i,j\in\mathbb{Z}}$ *mit* $\mathbf{x}$*, wobei* $h_{ij} := \mathbf{h}(i-j)$*.*

Beweis. Wir können das Matrix-Vektor-Produkt $H\mathbf{x}$ schreiben als

$$H\mathbf{x} = \begin{bmatrix} \ddots & & & & \\ \cdots & \mathbf{h}(0) & \mathbf{h}(-1) & \mathbf{h}(-2) & \cdots \\ \cdots & \mathbf{h}(1) & \mathbf{h}(0) & \mathbf{h}(-1) & \cdots \\ \cdots & \mathbf{h}(2) & \mathbf{h}(1) & \mathbf{h}(0) & \cdots \\ & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ \mathbf{x}(-1) \\ \mathbf{x}(0) \\ \mathbf{x}(1) \\ \vdots \end{bmatrix}$$

$$= \begin{bmatrix} & & \vdots & & \\ \cdots + & \mathbf{h}(0)\mathbf{x}(-1) & + \mathbf{h}(-1)\mathbf{x}(0) & + \mathbf{h}(-2)\mathbf{x}(1) & + \cdots \\ \cdots + & \mathbf{h}(1)\mathbf{x}(-1) & + \mathbf{h}(0)\mathbf{x}(0) & + \mathbf{h}(-1)\mathbf{x}(1) & + \cdots \\ \cdots + & \mathbf{h}(2)\mathbf{x}(-1) & + \mathbf{h}(1)\mathbf{x}(0) & + \mathbf{h}(0)\mathbf{x}(1) & + \cdots \\ \cdots + & \mathbf{h}(3)\mathbf{x}(-1) & + \mathbf{h}(2)\mathbf{x}(0) & + \mathbf{h}(1)\mathbf{x}(1) & + \cdots \\ & & \vdots & & \end{bmatrix}.$$

Für ein beliebiges $n \in \mathbb{Z}$ ist also der n-te Eintrag von $H\mathbf{x}$

$$[H\mathbf{x}](n) = \sum_{j\in\mathbb{Z}} h_{nj}\mathbf{x}(j) = \sum_{j\in\mathbb{Z}} \mathbf{h}(n-j)\mathbf{x}(j) = [\mathbf{h} * \mathbf{x}](n),$$

was das Ergebnis der Anwendung von $\mathbf{h}$ auf $\mathbf{x}$ ist. □

Somit können wir einen Filter $\mathbf{H}$ mit der zugehörigen Toeplitz-Matrix H identifizieren. Ist $\mathbf{H}$ kausal, so hat H die Gestalt einer unendlichen unteren Dreiecksmatrix; ist $\mathbf{H}$ ein FIR-Filter, so enthält jede Zeile von H endlich viele Einträge, nämlich die Einträge der Impulsantwort in umgekehrter Reihenfolge. Somit hat die Toeplitz-Matrixdarstellung von kausalen FIR-Filtern, auf die wir uns überwiegend beschränken werden, die Form einer unendlichen unteren Dreiecks-Bandmatrix.

Bemerkung. Für einen Filter $\mathbf{H}$ haben wir mit der Impulsantwort, der Frequenzantwort und der Toeplitzmatrix drei verschiedene Darstellungen kennengelernt. Für den Rest der Arbeit einigen wir uns auf die folgende Notationen: die Impulsantwort notieren wir als fettgedruckten Kleinbuchstaben, z.B. $\mathbf{h}$, und die Frequenzantwort als kursivgedruckten Großbuchstaben, z.B. H. Für die Toeplitz-Matrix überladen wir die Notation und schreiben diese – genau wie den Filter selbst – als fettgedruckten Großbuchstaben, z.B. $\mathbf{H}$.

1.1.3. Hoch- und Tiefpassfilter

Wie der Name Filter bereits andeutet, dienen Filter dazu, gewisse unerwünschte Anteile aus einem Signal zu entfernen. Die beiden für die Wavelettheorie relevanten Filterarten sind die sogenannten Hoch- und Tiefpassfilter. Diese entfernen niedrigfrequente bzw. hochfrequente Signalanteile und lassen die verbleibenden Signalanteile unverändert. Dazu unterteilen wir den Frequenzbereich $[-\pi, \pi]$ in zwei gleiche Teile: das Frequenzinterval $(-\frac{\pi}{2}, \frac{\pi}{2})$ bildet die niedrigen Frequenzen, der Rest die hohen Frequenzen.

Wie ein Filter $\mathbf{H}$ auf verschiedene Frequenzanteile wirkt, können wir in der Frequenzantwort H des Filters ablesen. Das Entfernen einer Frequenz ω durch den Filter entspricht $|H(\omega)| = 0$; das Beibehalten der Frequenz hingegen $|H(\omega)| = 1$.

Somit können wir ideale Hoch- und Tiefpassfilter definieren:

Definition 1.1.14. Der *ideale Tiefpassfilter* $\mathbf{H}_0$ ist ein Filter, der das Frequenzband $(-\frac{\pi}{2}, \frac{\pi}{2})$ unverändert lässt und die restlichen Frequenzen entfernt. Der *ideale Hochpassfilter* $\mathbf{H}_1$ wiederum ist ein Filter, der das Frequenzband $(-\frac{\pi}{2}, \frac{\pi}{2})$ entfernt und die restlichen Frequenzen unverändert lässt. Die Frequenzantworten von $\mathbf{H}_0$ und $\mathbf{H}_1$ haben die Form

$$H_0(\omega) = \begin{cases} 1 & \omega \in (-\frac{\pi}{2}, \frac{\pi}{2}), \\ 0 & \text{sonst;} \end{cases} \qquad H_1(\omega) = \begin{cases} 0 & \omega \in (-\frac{\pi}{2}, \frac{\pi}{2}), \\ 1 & \text{sonst.} \end{cases} \tag{1.3}$$

Lemma 1.1.15 (Strang und Nguyen [SN96, S. 45ff.])**.** *Die idealen Hoch- und Tiefpassfilter haben die Impulsantworten*

$$\mathbf{h}_0(n) = \frac{\sin \frac{\pi n}{2}}{\pi n} = \begin{cases} \frac{1}{2} & n = 0, \\ 0 & n \textit{ gerade}, n \neq 0, \\ (-1)^{k+1} \frac{1}{\pi n} & n = 2k+1; \end{cases} \qquad \mathbf{h}_1(n) = (-1)^n \mathbf{h}_0(n) \tag{1.4}$$

und sind damit keine FIR-Filter.

Bemerkung. Da die idealen Hoch- und Tiefpassfilter keine FIR-Filter sind, sind sie für Anwendungsfälle ungeeignet. Auch endliche Approximationen der idealen Filter, z.B.

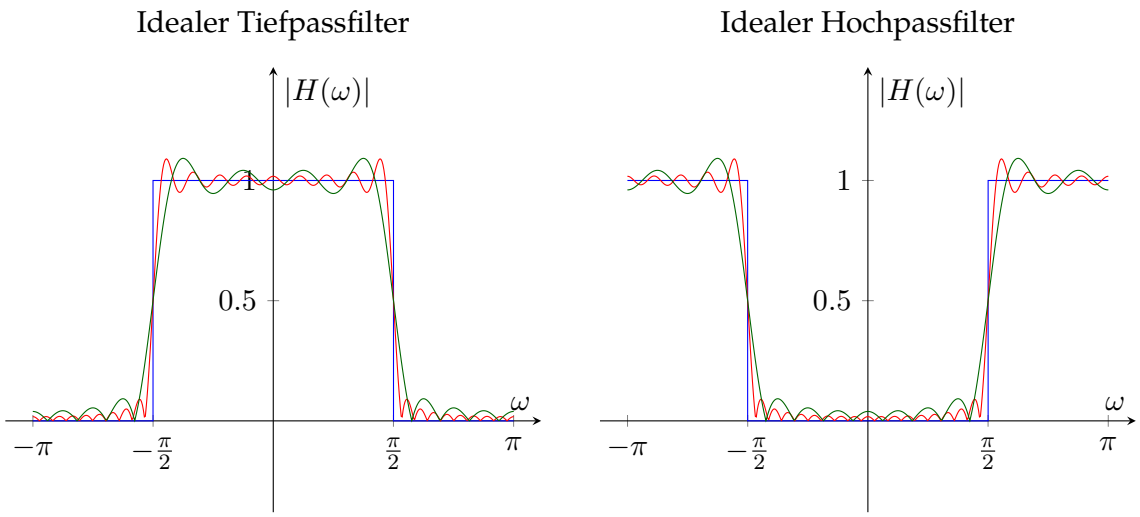


Abbildung 1.3.: Absolute Frequenzantwort des idealen Tiefpassfilters (links) und Hochpassfilters (rechts). Dargestellt sind jeweils die absolute Frequenzantwort des idealen Filters (blau) sowie der Approximation mit den ersten 10 (grün) und 20 (rot) Termen. Gut zu erkennen sind die Ausschläge an den Sprungstellen $\pm\pi/2$, die als Gibbs'sches Phänomen beschrieben werden.

durch Einschränkung auf die ersten N Filterkoeffizienten, sind aufgrund des Gibbs'schen Phänomens [vgl. Bea04, Abschnitt 14A; SN96, S. 46ff.] keine optimale Lösung, wie in Abbildung 1.3 veranschaulicht wird. Aus diesem Grund wurden verschiedene Verfahren zur Konstruktion von nicht-idealen FIR-Hoch- und Tiefpassfiltern vorgeschlagen [vgl. SN96, Abschnitt 2.3].

Unsere Mindestanforderung an nicht-ideale Tiefpassfilter ist, dass das niedrigfrequenteste Signal, also das konstante Signal, unverändert bleibt und das hochfrequenteste diskrete Signal, also das alternierende Signal $\mathbf{x}(n) = (-1)^n$, auf 0 gesetzt wird. Alternativ formuliert muss für die Frequenzantwort H gelten, dass

$$|H(0)| = 1, \qquad |H(\pm\pi)| = 0.$$

Diese Mindestanforderung gilt für nicht-ideale Hochpassfilter genau umgekehrt:

$$|H(0)| = 0, \qquad |H(\pm\pi)| = 1.$$

Dies sind Minimalkriterien. Wir werden im Laufe dieser Arbeit weitere Bedingungen kennenlernen, die ein Filter erfüllen muss, um zur Konstruktion von Wavelets geeignet zu sein.

Beispiel 1.1.16. Wir haben bereits jeweils ein Beispiel für einen nicht-idealen Tief- bzw. Hochpassfilter kennengelernt: das Gleitende Mittel und die Gleitende Differenz. Ersteres bildet den Durchschnitt zweier aufeinanderfolgender Signalwerte. Somit werden – bildlich gesprochen – schnelle Änderungen im Signal „herausgemittelt", wodurch sich ein Tiefpassfilter ergibt. Als Hochpassfilter agiert die Gleitende Differenz gegensätzlich. Indem

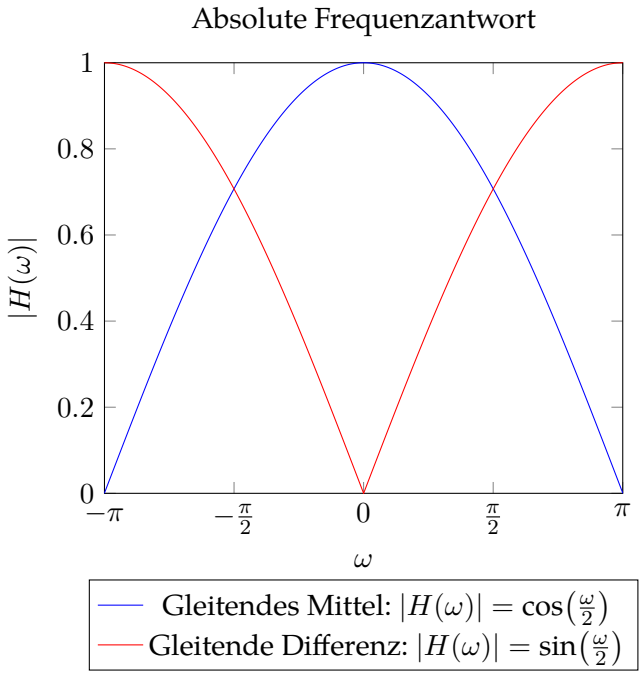


Abbildung 1.4.: Betrag der Frequenzantwort des Gleitenden Mittels (blau) und der Gleitenden Differenz (rot). Wir erkennen, dass die Bezeichnungen „Tiefpassfilter“ bzw. „Hochpassfilter“ jeweils gerechtfertigt sind.

Differenzen aufeinanderfolgender Signalwerte gebildet werden, werden Änderungen im Signal herausgestellt und das „Grundniveau“ des Signals entfernt.

Eine Visualisierung des Betrags der Frequenzantworten beider Filter wird in Abbildung 1.4 dargestellt. Wir erkennen, dass die Filter die Mindestanforderungen an Tief- bzw. Hochpassfilter erfüllen.

1.2. Filterbänke

Ein häufiges Schema von Anwendungen der Signalverarbeitung ist die *Analyse* eines Signals. Dabei wird ein Signal in eine Form transformiert, die eine Extraktion relevanter Informationen oder eine Weiterverarbeitung erleichtert. Ein Beispiel einer solchen Analyse haben wir mit den Hoch- und Tiefpassfiltern bereits kennengelernt. Hier wird ein Signal auf seinen hoch- bzw. niedrigfrequenten Anteil reduziert. Allerdings gehen bei dieser Reduktion Informationen verloren, da gewisse Frequenzbereiche aus dem Signal entfernt werden. Das ursprüngliche Signal ist somit nicht rekonstruierbar.

In einigen Anwendungsfällen ist es jedoch notwendig, dass kein Informationsverlust bei der Analyse auftritt. Es soll also eine andere Darstellung des Signals mit wünschenswerten Eigenschaften gefunden werden, aus der das Signal wieder rekonstruiert werden kann. Diesen Rekonstruktionsvorgang nennen wir *Synthese*.

Wie besprochen, erfüllen einzelne Hoch- und Tiefpassfilter diese Eigenschaft per Definition nicht, da gewisse Frequenzbereiche aus dem Signal entfernt werden. Nehmen wir allerdings ein passendes Paar aus Hoch- und Tiefpassfilter zusammen, verlieren wir keine Informationen. Stattdessen teilen wir das Signal lediglich in einen hoch- und einen niedrigfrequenten Anteil auf, die wir wieder zum Ursprungssignal zusammensetzen können. Damit kommen wir zum Konzept einer *Filterbank*. Diese nutzt für die Analyse und Synthese jeweils ein geeignetes Filterpaar, wobei das Analysepaar aus einem Hoch- und einem Tiefpassfilter besteht. Das Analysepaar nennen wir dabei auch *Analysebank*, das Synthesepaar entsprechend *Synthesebank*. Die Analyse findet statt, indem wir beide Filter der Analysebank auf das Signal anwenden und somit das Signal in einen hoch- und einen niedrigfrequenten Signalanteil aufteilen. Für die Synthese wenden wir auf diese beiden Signalanteile jeweils einen Filter der Synthesebank an und addieren dies. In Abbildung 1.5 ist dies schematisch dargestellt.

Im folgenden Abschnitt werden wir Kriterien für die Wahl geeigneter Filter kennenlernen, sodass der Syntheseteil einer Filterbank den Analyseteil invertiert.[1]

1.2.1. Sampling in der Filterbank

Diese erste Konzeption einer Filterbank hat noch ein entscheidendes Problem: Für ein Eingabesignal mit endlicher Länge[2] erzeugen die Filter der Analysebank jeweils ein mindestens genauso langes gefiltertes Signal.

Wendet man beide an, so verdoppelt sich der benötigte Speicherplatz, obwohl der Informationsgehalt konstant geblieben ist. Die Lösung für dieses Problem ist das Downsampling: wir verwerfen jeden zweiten Eintrag der Filter-Ergebnisse.

Definition 1.2.1. Der *Downsampling*-Operator $(\downarrow 2)$ entfernt die ungeraden Komponenten aus einem Signal $\mathbf{x}$. Für $\mathbf{v} := (\downarrow 2)\mathbf{x}$ gilt $\mathbf{v}(k) = \mathbf{x}(2k)$. Wir erhalten somit

$$\mathbf{v} = (\downarrow 2)\begin{bmatrix}\cdots & \mathbf{x}(-1) & \mathbf{x}(0) & \mathbf{x}(1) & \mathbf{x}(2) & \cdots\end{bmatrix} = \begin{bmatrix}\cdots & \mathbf{x}(-2) & \mathbf{x}(0) & \mathbf{x}(2) & \cdots\end{bmatrix}.$$

Lemma 1.2.2. *Sei* $\mathbf{H}$ *ein Filter mit Impulsantwort* $\mathbf{h}$ *und* $\mathbf{x}$ *ein Signal. Dann gilt*

$$[(\downarrow 2)\mathbf{H}\mathbf{x}](n) = \sum_{k\in\mathbb{Z}} \mathbf{h}(2n-k)\mathbf{x}(k).$$

Beweis. Sei $\mathbf{v} := \mathbf{H}\mathbf{x} = \sum_{k\in\mathbb{Z}} \mathbf{h}(n-k)\mathbf{x}(k)$. Dann ist

$$[(\downarrow 2)\mathbf{v}](n) = \mathbf{v}(2n) = \sum_{k\in\mathbb{Z}} \mathbf{h}(2n-k)\mathbf{x}(k). \qquad \square$$

Die Synthese soll bei der Rekonstruktion ein Signal in der Länge des Ausgangssignals erzeugen. Um die Halbierung der Signallänge durch Downsampling zu kompensieren, wenden wir den Upsampling-Operator an.

[1] Genau genommen verlangen wir nicht direkt Invertierbarkeit, sondern erlauben bei der Rekonstruktion eine Verzögerung im Signal, um kausale Filter zu ermöglichen. Mehr dazu später.

[2] Wie genau man einen Filter auf ein endliches Signal anwenden kann, besprechen wir in Kapitel 2.

Definition 1.2.3. Der *Upsampling*-Operator $(\uparrow 2)$ ist das Gegenstück zum Downsampling-Operator und fügt zwischen allen Komponenten eines Signals $\mathbf{y}$ eine 0 ein. Für $\mathbf{u} \coloneqq (\uparrow 2)\mathbf{y}$ haben wir

$$\begin{cases} \mathbf{u}(2k) = \mathbf{y}(k) \\ \mathbf{u}(2k+1) = 0 \end{cases}.$$

Wir erhalten also

$$(\uparrow 2)\begin{bmatrix} \cdots & \mathbf{y}(-1) & \mathbf{y}(0) & \mathbf{y}(1) & \cdots \end{bmatrix} = \begin{bmatrix} \cdots & \mathbf{y}(-1) & 0 & \mathbf{y}(0) & 0 & \mathbf{y}(1) & 0 & \cdots \end{bmatrix}.$$

Bemerkung. (i) Für den Rest der Arbeit werden wir diese beiden Operatoren als Teil der Analysebank bzw. der Synthesebank sehen. Die Analysebank besteht somit aus der Anwendung der Analysefilter gefolgt vom Downsampling-Operator; in der Synthesebank folgen auf den Upsampling-Operator die Synthesefilter.

(ii) Für die Konstruktion einer Wavelet-Basis in Abschnitt 1.3 werden wir die Analysebank einer Filterbank mehrfach iteriert anwenden. Jedoch verwerfen wir aktuell die Hälfte der Signaleinträge und reduzieren dadurch die Energie des Analyseergebnisses erheblich, insbesondere bei wiederholter Anwendung. Diesen Energieverlust werden wir kompensieren, indem wir die Ausgabe der Analysebank um den Faktor $\sqrt{2}$ skalieren und somit die Energie verdoppeln.

Diesen Faktor von $\sqrt{2}$ werden wir direkt in die verwendeten Hoch- und Tiefpassfilter integrieren. Statt der Tiefpassfilter $\mathbf{H}_0$ und Hochpassfilter $\mathbf{H}_1$ der Analysebank verwenden wir die Filter

$$\mathbf{C} \coloneqq \sqrt{2}\mathbf{H}_0 \quad \text{und} \quad \mathbf{D} \coloneqq \sqrt{2}\mathbf{H}_1.$$

Im Kontext von Filterbanken bezeichnen wir diese skalierten Filter auch als Hoch- bzw. Tiefpassfilter und notieren diese stets mit $\mathbf{C}$ für den Tiefpass- und $\mathbf{D}$ für den Hochpassfilter. Die entsprechenden Synthesefilter werden wir mit $\tilde{\mathbf{C}}$ für den Tiefpasskanal und $\tilde{\mathbf{D}}$ für den Hochpasskanal schreiben.

1.2.2. Matrixdarstellung von Filterbänken

Sei $\mathbf{L} \coloneqq (\downarrow 2)\mathbf{C}$ und $\mathbf{B} \coloneqq (\downarrow 2)\mathbf{D}$. Dann können wir $\mathbf{L}$ bzw. $\mathbf{B}$ als unendliche Matrizen schreiben, indem wir aus den Toeplitz-Matrizen von $\mathbf{C}$ bzw. $\mathbf{D}$ die ungerade indizierten Zeilen löschen. Nehmen wir beide Matrizen zusammen, erhalten wir die Darstellung der Analysebank als unendliche Matrix

$$\mathbf{H}_t \coloneqq \begin{bmatrix} \mathbf{L} \\ \mathbf{D} \end{bmatrix} = \begin{bmatrix} & & & \vdots & & & \\ \cdots & \mathbf{c}(0) & \mathbf{c}(-1) & \mathbf{c}(-2) & \mathbf{c}(-3) & \mathbf{c}(-4) & \cdots \\ \cdots & \mathbf{c}(2) & \mathbf{c}(1) & \mathbf{c}(0) & \mathbf{c}(-1) & \mathbf{c}(-2) & \cdots \\ & & & \vdots & & & \\ \cdots & \mathbf{d}(0) & \mathbf{d}(-1) & \mathbf{d}(-2) & \mathbf{d}(-3) & \mathbf{c}(-4) & \cdots \\ \cdots & \mathbf{d}(2) & \mathbf{d}(1) & \mathbf{d}(0) & \mathbf{d}(-1) & \mathbf{c}(-2) & \cdots \\ & & & \vdots & & & \end{bmatrix}.$$

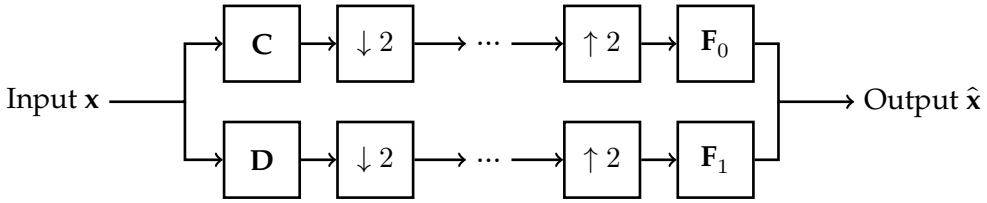


Abbildung 1.5.: Schema einer Filterbank. Wir erkennen die Analysebank (links) und die Synthesebank (rechts) mit den normierten Tief- und Hochpassfiltern $\mathbb{C}$ und **D**. Die Analysebank teilt das Signal **x** in einen niedrig- und einen hochfrequenten Anteil mit je halber Länger auf; die Synthesebank rekombiniert diese Anteile zum Output $\hat{\mathbf{x}}$. Die Punkte zwischen beiden Teilen deuten eine Weiterverarbeitung wie z.B. Kompression an. Für die Rekonstruktion nehmen wir keine Weiterverarbeitung an (können die Lücke gedanklich schließen) und erwarten $\hat{\mathbf{x}}(n) = \mathbf{x}(n-l)$ für $l \geq 0$.

Für ein Signal **x** und $a := \mathbf{L}\mathbf{x}$, $b := \mathbf{B}\mathbf{x}$ gilt in Block-Notation

$$\mathbf{H}_t\mathbf{x} = \begin{bmatrix}\mathbf{L}\\ \mathbf{B}\end{bmatrix}\mathbf{x} = \begin{bmatrix}\mathbf{a}\\ \mathbf{b}\end{bmatrix}.$$

Eine weitere Matrix-Darstellung der Analysebank ist die Block-Toeplitz-Matrix:

Definition 1.2.4. Sei $\mathbf{L} := (\downarrow 2)\mathbf{C}$ und $\mathbf{B} := (\downarrow 2)\mathbf{D}$. Dann erhalten wir die unendliche Matrix $\mathbf{H}_b$, indem wir die Zeilen von **L** und **B** verschränken. Die geraden Zeilen von $\mathbf{H}_b$ bilden die Zeilen von **L**; die ungeraden die Zeilen von **B**. Es gilt

$$\begin{cases}(\mathbf{H}_b)_{2i,j} = (\mathbf{L})_{i,j},\\ (\mathbf{H}_b)_{2i+1,j} = (\mathbf{B})_{i,j}.\end{cases}$$

$\mathbf{H}_b$ hat somit die Form

$$\mathbf{H}_b = \begin{bmatrix}
\ddots & & & & & & & \\
\cdots & \mathbf{c}(0) & \mathbf{c}(-1) & \mathbf{c}(-2) & \mathbf{c}(-3) & \mathbf{c}(-4) & \mathbf{c}(-5) & \cdots \\
\cdots & \mathbf{d}(0) & \mathbf{d}(-1) & \mathbf{d}(-2) & \mathbf{d}(-3) & \mathbf{c}(-4) & \mathbf{d}(-5) & \cdots \\
\cdots & \mathbf{c}(2) & \mathbf{c}(1) & \mathbf{c}(0) & \mathbf{c}(-1) & \mathbf{c}(-2) & \mathbf{c}(-3) & \cdots \\
\cdots & \mathbf{d}(2) & \mathbf{d}(1) & \mathbf{d}(0) & \mathbf{d}(-1) & \mathbf{c}(-2) & \mathbf{d}(-3) & \cdots \\
 & & & & & & & \ddots
\end{bmatrix}.$$

Wir bezeichnen $\mathbf{H}_b$ als *Block-Toeplitz-Matrix*, da die 2×2-Block-Diagonalen von $\mathbf{H}_b$ konstant sind.

1.2.3. Perfekte Rekonstruierbarkeit und Orthonormale Filterbänke

Definition 1.2.5. Sei **x** ein Signal und $\mathbf{a} = \mathbf{L}\mathbf{x}$ und $\mathbf{b} = \mathbf{B}\mathbf{x}$ die Ausgaben der Analysebank. Die Synthesebank ergibt dann

$$\hat{\mathbf{x}} := \tilde{\mathbf{C}}(\uparrow 2)\mathbf{a} + \tilde{\mathbf{D}}(\uparrow 2)\mathbf{b} = \tilde{\mathbf{C}}(\uparrow 2)(\downarrow 2)\mathbf{C}\mathbf{x} + \tilde{\mathbf{D}}(\uparrow 2)(\downarrow 2)\mathbf{D}\mathbf{x}.$$

Wir sagen, dass eine Filterbank *perfekte Rekonstruktion* ermöglicht, wenn die Synthesebank die Analysebank bis auf eine Verzögerung von $l \geq 0$ invertiert. Das heißt, dass für alle $n \in \mathbb{Z}$ folgt:

$$\mathbf{x}(n) = \hat{\mathbf{x}}(n - l).$$

Bemerkung. Wir räumen bei der Invertierung eine Verzögerung ein, damit wir die Kausalität der verwendeten Synthesefilter sicherstellen können. Angenommen, wir haben als Synthesefilter FIR-Filter, die eine perfekte Rekonstruktion ohne Verzögerung einräumen. Sind diese keine kausalen Filter, existiert aufgrund der endlichen Impulsantwort ein $l \geq 0$, sodass eine Vezögerung um l die Filter kausal werden lässt. Allerdings führt dies auch zu einer Verzögerung des rekonstruierten Signals um l.

Im folgenden werden wir aus den Analysefiltern Synthesefilter konstruieren. Die Berechnungen dazu sind einfacher, wenn die konstruierten Synthesefilter nicht kausal sind. Wir merken uns aber, dass sich durch Einfügen einer Verzögerung Kausalität herstellen ließe.

Definition 1.2.6. Sei $\mathbf{H}$ ein Filter. Dann nennen wir den Filter $\mathbf{H}^T$ mit Impulsantwort

$$\mathbf{h}^T(k) = \mathbf{h}(-k)$$

den *transponierten Filter* zu $\mathbf{H}$. Die Toeplitz-Matrix von $\mathbf{H}^T$ ergibt sich durch Transponieren der Toeplitz-Matrix $\mathbf{H}$.

Definition 1.2.7. Wir nennen eine Filterbank *orthonormal*, wenn sie perfekte Rekonstruktion ermöglicht und

$$\tilde{\mathbf{C}} = \mathbf{C}^T \quad \tilde{\mathbf{D}} = \mathbf{D}^T,$$

also sich die Synthesefilter aus den Analysefiltern durch Transponieren ergeben.

Bemerkung. Zur Konstruktion einer orthonormalen Filterbank reicht es also aus, die Filter der Analysebank so zu bestimmen, dass die Filterbank perfekte Rekonstruktion ermöglicht. Die Synthesebank ergibt sich dann aus den transponierten Filtern.

Satz 1.2.8 (Strang und Nguyen [SN96, S. 147ff.])**.** *Sei* $\mathbf{c}$ *ein Tiefpassfilter mit gerader Länge* N*. Wählen wir*

$$\mathbf{d}(k) = (-1)^k \mathbf{c}(N - 1 - k), \tag{1.5}$$

so ist die entsprechende Filterbank genau dann orthonormal, wenn

$$\sum_{k \in \mathbb{Z}} \mathbf{c}(k)\mathbf{c}(k - 2n) = \delta_{n,0}. \tag{1.6}$$

Bemerkung. Mithilfe dieses Satzes ergibt sich eine orthonormale Filterbank sogar bereits aus der Wahl eines geeigneten Tiefpassfilters. Der Hochpassfilter folgt aus Gleichung (1.5) und die Synthesebank aus den transponierten Analysefiltern.

Beispiel 1.2.9. Wir betrachten unser laufendes Beispiel des Gleitenden Mittels und der Gleitenden Differenz. Wir wollen zeigen, dass beide Filter zusammen eine Filterbank bilden. Dazu skalieren wir die Impulsantworten mit $\sqrt{2}$ und erhalten

$$\sqrt{2}\mathbf{h_0} = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right), \quad \sqrt{2}\mathbf{h_1} = \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right).$$

Wir stellen fest, dass das Filterpaar die Gleichung (1.5) erfüllt. Weiter gilt

$$\sum_{k\in\mathbb{Z}} (\sqrt{2}\mathbf{h}_0(k))^2 = \frac{1}{\sqrt{2}^2} + \frac{1}{\sqrt{2}^2} = 1.$$

Außerdem überschneiden sich aufgrund der geringen Länge die geraden Verschiebungen von $\mathbf{h}_0$ nicht, sodass Gleichung (1.6) erfüllt ist. Die Filterbank mit skaliertem Gleitenden Mittel und Gleitender Differenz als Analysefiltern ist also orthonormal.

1.3. Wavelets

Wir haben im vorherigen mit Filterbänken ein diskretes Signal in einen hochfrequenten Teil („Details“) und einen niedrigfrequenten Teil („Grundniveau“) aufteilen können. Dieses Konzept wollen wir nutzen, um Signale in $L^2(\mathbb{R})$ anzunähern. Ausgehend von einem Grundniveau ergänzen wir auf verschiedenen Auflösungen, auch Skalen genannt, nach und nach mehr Details, um das Signal zu approximieren. Die Funktionen, mit denen wir die Details ergänzen, sind unsere Wavelets. Nehmen wir die Details auf allen noch so feinen Auflösungen zusammen, erhalten wir eine Basis von $L^2(\mathbb{R})$ – die Wavelet-Basis.

In diesem Abschnitt orientieren wir uns an Strang und Nguyen [SN96] und Daubechies [Dau92].

1.3.1. Multiskalenanalyse

Definition 1.3.1 (Multiskalenanalyse)**.** Eine Familie $(V_j)_{j\in\mathbb{Z}}$ von Unterräumen von $L^2(\mathbb{R})$ heißt *Multiskalenanalyse*, wenn sie eine aufsteigende Kette

$$\{0\} \subset \cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots \subset L^2(\mathbb{R}) \tag{1.7}$$

mit den folgenden Eigenschaften bilden:

(MS1) Vollständigkeit:

$$\overline{\bigcup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R}) \quad \text{und} \quad \bigcap_{j\in\mathbb{Z}} V_j = \{0\}, \tag{1.8}$$

wobei der Abschluss unter der $L^2(\mathbb{R})$-Norm gebildet wird,

(MS2) Verschiebungsinvarianz:

$$f(t) \in V_0 \iff f(t-k) \in V_0 \quad \text{für alle } k \in \mathbb{Z}, \tag{1.9}$$

(MS3) Skalierungsinvarianz:

$$f(t) \in V_j \iff f(2t) \in V_{j+1} \quad \text{für alle } j \in \mathbb{Z}, \tag{1.10}$$

(MS4) Verschiebungsinvariante Basis: Es existiert eine Funktion $\phi \in V_0$, sodass

$$\{\phi(t-k) \mid k \in \mathbb{Z}\}$$

eine Orthonormalbasis von V_0 ist.

Die Funktion ϕ wird *Skalierungsfunktion* genannt. Das orthogonale Komplement W_j zweier aufeinanderfolgender Unterräume der Multiskalenanalyse

$$V_{j+1} = V_j \oplus W_j, \tag{1.11}$$

bezeichnen wir als *Waveletunterraum*.

Multiskalenanlysen sind das zentrale funktionalanalytische Konzept der Wavelet-Theorie. Die Unterräume V_j entsprechen allen Funktionen, die bis zu einer Auflösung von 2^j approximiert haben. Waveletunterräume W_j stellen die eingangs beschriebenen Details, die mit jeder Skala ergänzt werden, dar.

Bemerkung. Für $f \in L^2(\mathbb{R})$ sei $\mathcal{P}_{V_j}(f)$ die orthogonale Projektion von f auf V_j. Dann impliziert Punkt (MS1) einerseits, dass die Multiskalenanalyse nicht redundant ist und andererseits, dass

$$\lim_{j\to\infty} \mathcal{P}_{V_j}(f) = f.$$

Lemma 1.3.2 (Eigenschaften der Multiskalenanalyse)**.** *Sei $\{V_j\}_{j\in\mathbb{Z}}$ eine Multiskalenanalyse. Dann gilt*

(i) $W_j \perp W_k$ *für alle* $k \neq j$,

(ii) $V_{j+1} = W_j \oplus \cdots \oplus W_1 \oplus W_0 \oplus V_0$ *für alle* $j \geq 0$,

(iii) $f(t) \in W_j \iff f(2t) \in W_{j+1}$ *für alle* $j \in \mathbb{Z}$,

(iv) für alle $j \in \mathbb{Z}$ *ist* $\{\phi_{jk} \mid k \in \mathbb{Z}\}$ *mit* $\phi_{jk}(t) := 2^{j/2}\phi(2^j t - k)$ *eine Orthonormalbasis von* V_j.

Beweis. (i) Ohne Beschränkung der Allgemeinheit sei k der kleinere der Indizes k und j. Dann gilt $W_k \subset V_j \perp W_j$.

(ii) folgt direkt aus der wiederholten Anwendung der Definition der Waveletunterräume.

(iii) folgt aus der der Definition der Waveletunterräume sowie Punkt (MS3).

(iv) Das dies eine Basis ist folgt direkt aus den Eigenschaften (MS2) bis (MS4). Mittels Substitution lässt sich die Orthonormalität der Basis direkt auf Punkt (MS4) zurückführen:

$$\|2^{j/2}\phi(2^j t - k)\|^2_{L^2(\mathbb{R})} = \int_{\mathbb{R}} 2^j \phi(2^j t - k)^2 \,\mathrm{d}t = \int_{\mathbb{R}} \phi(t)^2 \,\mathrm{d}t = \|\phi\|^2_{L^2(\mathbb{R})} = 1$$

Für $k \neq k'$ erhalten wir per Substitution

$$\begin{aligned}\langle 2^{j/2}\phi(2^j t-k), 2^{j/2}\phi(2^j t-k')\rangle_{L^2(\mathbb{R})} &= \int_{\mathbb{R}} 2^j \phi(2^j t-k)\phi(2^j t-k')\,\mathrm{d}t \\ &= \int_{\mathbb{R}} \phi(t-k)\phi(t-k')\,\mathrm{d}t \\ &= \langle \phi(t-k), \phi(t-k')\rangle_{L^2(\mathbb{R})} = 0\end{aligned}$$

□

Satz 1.3.3 (Zwei-Skalen-Gleichung). *Sei $\{V_j\}_{j\in\mathbb{Z}}$ eine Multiskalenanalyse. Dann existieren Koeffizienten $\mathbf{c} \in l^2$, sodass die Skalierungsfunktion ϕ die Zwei-Skalen-Gleichung erfüllt:*

$$\phi(t) = \sqrt{2}\sum_{k\in\mathbb{Z}} \mathbf{c}(k)\phi(2t-k) \qquad \textit{mit } \mathbf{c}(k) \in \mathbb{R}. \qquad \text{(Zwei-Skalen-Gleichung)}$$

Für die Koeffizienten $\mathbf{c}(k)$ gilt

$$\sum_{k\in\mathbb{Z}} \mathbf{c}(k)\mathbf{c}(k-2m) = \delta_{m,0}. \tag{1.12}$$

Beweis nach Strang und Nguyen [SN96]. Es gilt $\phi \in V_0 \subset V_1$. Somit können wir ϕ in der Orthonormalbasis $\{\sqrt{2}\phi(2t-k)\}$ von V_1 darstellen und erhalten die Zwei-Skalen-Gleichung. Die Gleichung (1.12) bekommen wir, indem wir die Zwei-Skalen-Gleichungen für $\phi(t)$ und $\phi(t-m)$ multiplizieren, anschließend integrieren und die Orthonormalität der Basis nutzen:

$$\begin{aligned}\delta_{m,0} = \int_{\mathbb{R}} \phi(t)\phi(t-m)\,\mathrm{d}t &= \int_{\mathbb{R}} 2\Big(\sum_{k\in\mathbb{Z}} \mathbf{c}(k)\phi(2t-k)\Big)\Big(\sum_{k\in\mathbb{Z}} \mathbf{c}'(k)\phi(2(t-m)-k)\Big)\,\mathrm{d}t \\ &= \int_{\mathbb{R}} 2\Big(\sum_{k\in\mathbb{Z}} \mathbf{c}(k)\phi(2t-k)\Big)\Big(\sum_{k\in\mathbb{Z}} \mathbf{c}(k-2m)\phi(2t-k)\Big)\,\mathrm{d}t \\ &= \sum_{k\in\mathbb{Z}} \mathbf{c}(k)\mathbf{c}(k-2m)\end{aligned}$$

□

Satz 1.3.4 (Waveletbasis, [Dau92, Theorem 5.1.1]). *Sei $\{V_j\}_{j\in\mathbb{Z}}$ eine Multiskalenanalyse. Dann existiert ein $w \in W_0$, sodass $\{w(t-k) \mid k \in \mathbb{Z}\}$ eine Orthonormalbasis von W_0 ist und*

$$\{\phi(t-k) \mid k \in \mathbb{Z}\} \quad \cup \quad \bigcup_{j\geq 0}\{w_{jk}(t) \mid k \in \mathbb{Z}\} \qquad \textit{mit} \qquad w_{jk}(t) := 2^{j/2}w(2^j t-k) \tag{1.13}$$

eine Orthonormalbasis von $L^2(\mathbb{R})$ ist. Die Funktion w hat die Form

$$w(t) = \sqrt{2}\sum_{k\in\mathbb{Z}} \mathbf{d}(k)\phi(2t-k). \qquad \text{(Wavelet-Gleichung)}$$

Eine mögliche Konstruktion von w ergibt sich, wenn wir $\mathbf{d}$ wie in (1.5) wählen als

$$\mathbf{d}(k) := (-1)^k \mathbf{c}(N-1-k) \qquad \textit{mit } N \textit{ gerade}, \tag{1.14}$$

wobei $\mathbf{c}$ die Koeffizienten aus der Zwei-Skalen-Gleichung sind.

Die Funktion w nennen wir „Mutterwavelet“, da alle Waveletunterräume von skalierten Translationen dieser Funktion aufgespannt werden.

Bemerkung. Die Zwei-Skalen-Gleichung und ihr Pendant, die Wavelet-Gleichung, sind zentral für die Wavelet-Theorie, da sie uns erlauben, Skalierungsfunktionen bzw. Wavelets einer Skala in der Basis der nächst-höheren Skala zu schreiben – mit Basiskoeffizienten, die nicht von der verwendeten Skala abhängen. Dieser Zusammenhang zwischen den verschiedenen Skalen ist elementar für die Wavelet-Theorie und erlaubt uns beispielsweise die Konstruktion schneller Verfahren zum Basiswechsel zwischen den Skalen in den folgenden Abschnitten.

Lemma 1.3.5. *Sei $\{V_j\}_{j\in\mathbb{Z}}$ eine Multiskalenanalyse mit einer Skalierungsfunktion ϕ, deren Fouriertransformation beschränkt und in $\omega = 0$ stetig sowie ungleich Null ist. Weiter seien* **c** *die Koeffizienten aus der Zwei-Skalen-Gleichung und* **d** *die Koeffizienten aus der Wavelet-Gleichung nach der Konstruktion* (1.14)*. Dann ist* **c** *ein Tiefpassfilter, der zusammen mit* **d** *eine orthonormale Filterbank bildet.*

Beweis. Man kann zeigen, dass unter diesen Bedingungen an die Skalierungsfunktion **c** ein mit $\sqrt{2}$ skalierter Tiefpassfilter ist [VK95, Problem 4.3]. Das heißt, dass

$$|C(0)| = \sqrt{2} \quad \text{und} \quad |C(\pm\pi)| = 0.$$

Da wir **d** wie in (1.5) wählen, folgt aufgrund von Theorem 1.2.8 die Orthonormalität aus (1.12). □

Bemerkung. Die an die Skalierungsfunktion gestellten Bedingungen sind in Anwendungsfällen stets erfüllt [VK95, S. 226].

Wir haben gezeigt, wie wir ausgehend von einer Multiskalenanalyse eine Orthonormalbasis von $L^2(\mathbb{R})$ konstruieren können. Zentrales Element ist dabei die Wahl einer passenden Skalierungsfunktion, aus der sich alles weitere ergibt.

Allerdings ist es nicht einfach, eine geeignete Skalierungsfunktion zu finden. So sind beispielsweise die Skalierungsfunktionen zweier wichtiger Klassen von Wavelets, die wir im Abschnitt 1.3.4 einführen, nicht in einer geschlossenen Form darstellbar [Dau92].

Ein vielversprechenderer Ansatz ist es, einen geeigneten Tiefpassfilter **c** zu wählen, den wir als Koeffizienten in die Zwei-Skalen-Gleichung einsetzen. Damit erhalten wir eine Fixpunktgleichung, die wir nach der Skalierungsfunktion lösen können – sofern der Tiefpassfilter gewisse Eigenschaften erfüllt, die die Lösbarkeit sicherstellen.

Wir wollen diesen Ansatz exemplarisch an einem Tief- und Hochpassfilterpaar durchrechnen:

Beispiel 1.3.6 (Haar-Wavelets)**.** Wir wählen als Tiefpassfilter das Gleitende Mittel $\mathbf{h}_0 = (1/2, 1/2)$ und wollen daraus die Skalierungsfunktion ϕ konstruieren. Zunächst skalieren wir $\mathbf{h}_0$ mit $\sqrt{2}$, um den Tiefpassfilter der Filterbank

$$\mathbf{c} = \sqrt{2}\mathbf{h}_0 = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$$

zu erhalten Aus Gleichung (1.14) ergibt sich mit $N = 2$ dann der Hochpassfilter

$$\mathbf{d}(k) := (-1)^k \mathbf{c}(1-k) = \Big(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\Big) = \sqrt{2}\Big(\frac{1}{2}, -\frac{1}{2}\Big),$$

welchen wir als skalierte Gleitende Differenz wiedererkennen. Wir setzen $\mathbf{c}$ in die Zwei-Skalen-Gleichung ein und erhalten die Fixpunktgleichung

$$\phi(t) = \phi(2t) + \phi(2t-1).$$

Bei dieser einfachen Gleichung können wir eine Lösung direkt ablesen: die Rechteckfunktion

$$\phi(t) = \phi_0 := \mathbb{1}_{[0,1)}(t) = \begin{cases} 1 & 0 \leq t < 1, \\ 0 & \text{sonst.} \end{cases}$$

Wir rechnen nach, dass dies tatsächlich eine Lösung der Zwei-Skalen-Gleichung ist:

$$\phi(2t) + \phi(2t-1) = \mathbb{1}_{[0,\frac{1}{2})}(t) + \mathbb{1}_{[\frac{1}{2},1)}(t) = \mathbb{1}_{[0,1)} = \phi(t).$$

Mit der Wavelet-Gleichung können wir nun das zu ϕ zugehörige Mutterwavelet konstruieren:

$$w(t) = \sqrt{2}\sum_{k\in\mathbb{Z}} \mathbf{d}(k)\phi(2t-k) = \phi(2t) - \phi(2t-1) = \begin{cases} 1 & 0 \leq t < \frac{1}{2}, \\ -1 & \frac{1}{2} \leq t < 1, \\ 0 & \text{sonst.} \end{cases}$$

Die aus ϕ und w abgeleiteten Wavelets werden nach Haar [Haa10] als *Haar-Wavelets* bezeichnet. In Abbildung 1.6 sind ϕ und w abgebildet.

Die Haar-Wavelets bilden aufgrund der geringen Länge der zugrundeliegenden Filter die einfachsten Wavelets. Nichtsdestotrotz haben sie große praktische Relevanz. Dies liegt unter anderem daran, dass sie aufgrund der geringen Länge auf endlichen Signalen keine Randbehandlung erfordern (vgl. Kapitel 2).

Bei den Haar-Wavelets konnten wir die Skalierungsfunktion ϕ, die die Zwei-Skalen-Gleichung löst, direkt ermitteln. Im Allgemeinen können wir ϕ mit einer Fixpunktiteration ermitteln. Die dazu notwendige Konstruktion wollen wir kurz skizzieren. Details zu dieser Konstruktion finden sich in Strang und Nguyen [SN96, Abschnitt 7.2].

Wir beginnen mit der Rechteckfunktion ϕ_0, die wir auch im Beispiel nutzten, und setzen wiederholt in die Zwei-Skalen-Gleichung ein. Somit erhalten wir die rekursive Folge

$$\phi^{(i+1)}(t) = \sqrt{2}\sum_{k\in\mathbb{Z}} \mathbf{c}(k)\phi^{(i)}(2t-k) \qquad \text{mit } \phi^{(0)} = \phi_0.$$

Nun kann man zeigen, dass man diese Fixpunktiteration äquivalent als Iteration eines (unendlichen) Matrix-Vektor-Produkts

$$a^{(i+1)} = \mathbf{T}a^{(i)} = \mathbf{T}^i a^{(0)} \qquad \text{mit } \mathbf{T} = (\downarrow 2)\mathbf{C}\mathbf{C}^T.$$

umschreiben kann. Dann folgt, dass die ursprünglichen Fixpunktiteration genau dann in $L^2(\mathbb{R})$ gegen eine Skalierungsfunktion konvergiert, wenn für die Eigenwerte der Matrix $\mathbf{T}$ gilt:

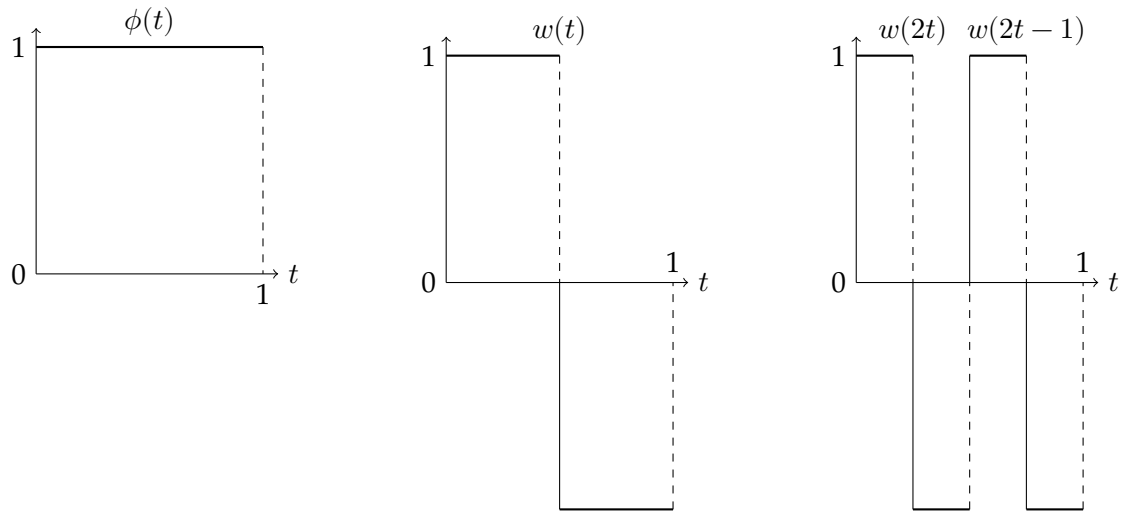


Abbildung 1.6.: Skalierungsfunktion (links), Mutterwavelet (Mitte) und Wavelets der ersten Skala (rechts) der Haar-Wavelets.

(i) für alle Eigenwerte λ bis auf exakt einen ist $|\lambda| < 1$;

(ii) für diesen einen Eigenwert λ_0 gilt $|\lambda_0| = 1$.

Für Tiefpassfilter, für die obiges Kriterium erfüllt ist, existiert also stets eine passende Skalierungsfunktion, die wir sogar mit der Fixpunktiteration approximieren können. Alle Tiefpassfilter, die wir in dieser Arbeit verwenden, erfüllen dieses Kriterium.

1.3.2. Schnelle Wavelet-Transformation

Wir haben in Punkt (ii) von Lemma 1.3.2 gesehen, dass wir die Unterräume V_j einer Multiskalenanalyse in eine direkte Summe von V_0 und Wavelet-Unterräumen zerlegen können. Ebenso haben wir die beiden orthonormalen Basen

$$\{\phi_{Jk} \mid k \in \mathbb{Z}\} \qquad \text{sowie} \qquad \{\phi_{0k} \mid k \in \mathbb{Z}\} \cup \bigcup_{i=0}^{J} \{w_{jk} \mid k \in \mathbb{Z}\}$$

von V_J konstruiert. Eine Basiswechsel von der Skalierungsfunktionsbasis in die Waveletbasis bezeichnen wir als Diskrete Wavelet-Transformation (DWT). Im folgenden werden wir ein effizientes Verfahren zur Berechnung der DWT kennenlernen: die Schnelle Wavelet-Transformation (*englisch*: Fast Wavelet Transform, FWT). Zuerst beschrieben wurde die FWT von Mallat [Mal89]. Wir orientieren uns für die Herleitung an Strang und Nguyen [SN96]. Dabei nutzen wir die Notation $y_{j,\_}$ für eine Folge $\{y_{j,k}\}_{k\in\mathbb{Z}}$.

Wir betrachten zunächst den Schritt von V_1 zu $V_0 \oplus W_0$. Die weiteren Schritte folgen dann induktiv aufgrund der Eigenschaften der Multiskalenanalyse. Für $f_1(t) \in V_1$ seien $a_{1,\_}$ die Koeffizienten der Basisdarstellung

$$f_1(t) = \sum_{k\in\mathbb{Z}} a_{1k}\phi_{1k}(t) = \sqrt{2}\sum_{k\in\mathbb{Z}} a_{1k}\phi(2t-k).$$

Wir wollen einen Basiswechsel konstruieren. Also suchen wir $a_{0,-}$ und $b_{0,-}$, sodass

$$f_1(t) = \sum_{k\in\mathbb{Z}} a_{1k}\phi_{1k}(t) = \sum_{k\in\mathbb{Z}} a_{0,k}\phi_{0,k}(t) + \sum_{k\in\mathbb{Z}} b_{0,k}w_{0,k}(t) = \sum_{k\in\mathbb{Z}} a_{0,k}\phi(t-k) + \sum_{k\in\mathbb{Z}} b_{0,k}w(t-k).$$

Zur Berechnung der Koeffizienten betrachten wir die Zwei-Skalen-Gleichung für $\phi(t-n)$ und die Wavelet-Gleichung für $w(t-n)$ mit $n \in \mathbb{Z}$. Wir substituieren $k = l-2n$ und erhalten

$$\begin{aligned}
\phi_{0,n}(t) &= \sqrt{2}\sum_{l\in\mathbb{Z}} \mathbf{c}(l-2n)\phi(2(t-n)-(l-2n)) = \sqrt{2}\sum_{l\in\mathbb{Z}} \mathbf{c}(l-2n)\phi_{1,l}(t);\\
w_{0,n}(t) &= \sqrt{2}\sum_{l\in\mathbb{Z}} \mathbf{d}(l-2n)\phi(2(t-n)-(l-2n)) = \sqrt{2}\sum_{l\in\mathbb{Z}} \mathbf{d}(l-2n)\phi_{1,l}(t).
\end{aligned}$$

Betrachten wir das Skalarprodukt von $\phi_{0,n}$ bzw. $w_{0,n}$ mit f_1, so erhalten wir aufgrund der Orthonormalität der Basis einen Ausdruck für die Koeffizienten a_{0n} bzw. b_{0n}:

$$\begin{aligned}
a_{0n} &= \langle \phi_{0n}, f_1\rangle_{L^2(\mathbb{R})} = \sum_{l\in\mathbb{Z}} \mathbf{c}(l-2n)a_{1l},\\
b_{0n} &= \langle w_{0n}, f_1\rangle_{L^2(\mathbb{R})} = \sum_{l\in\mathbb{Z}} \mathbf{d}(l-2n)a_{1l}
\end{aligned} \qquad \text{mit } n \in \mathbb{Z}.$$

Wir vergleichen dies mit Lemma 1.2.2 und stellen fest, dass sich die Koeffizienten $a_{0,-}$ und $b_{0,-}$ direkt als Ausgabe einer Filterbank mit $a_{1,-}$ als Eingabe ergeben. Allerdings müssen wir beachten, dass das Vorzeichen der Filterindizes gegensätzlich zu Lemma 1.2.2 ist. Daher besteht die Analysebank der Filterbank aus den transponierten Filtern $\mathbf{C}^T$ und $\mathbf{D}^T$ mit Impulsantworten

$$\mathbf{c}^T(k) = \mathbf{c}(-k) \qquad \text{und} \qquad \mathbf{d}^T(k) = \mathbf{d}(-k).$$

Obige Überlegungen können wir analog für alle $j \in \mathbb{Z}$ anwenden, um ausgehend von der Basisdarstellung in V_j die Koeffizienten $a_{j-1,-}$ und $b_{j-1,-}$ zu berechnen. Wenden wir das Verfahren anschließend auf $a_{j-1,-}$ an, erhalten wir eine Darstellung von $a_{j,-}$ in der Basis von $V_{j-2} \oplus W_{j-2} \oplus W_{j-1}$. Dies können wir nun wiederholen, um nach und nach die Koeffizienten weiterer Waveletunterräume zu berechnen. Insgesamt ergibt sich die Schnelle Wavelet-Transformation. In Abbildung 1.7 findet sich eine schematische Darstellung des Ablaufs der FWT.

Bemerkung. (i) In der Praxis wird die FWT nach einer festen Anzahl J von Rekursionsschritten abgebrochen. Diese Anzahl nennen wir *Skalen*. So gehen wir z.B. bei einer dreiskaligen FWT von $a_{3,-}$ aus und berechnen die Koeffizienten $b_{2,-}$, $b_{1,-}$, $b_{0,-}$ und brechen mit $a_{0,-}$ ab.

(ii) Sofern die verwendeten Filter eine endliche Länge haben, entstehen in jedem Rekursionsschritt aus dem Koeffizientenvektor $a_{j,-}$ mit endlicher Länge zwei Vektoren $a_{j-1,-}$ und $b_{j-1,-}$ mit jeweils der halben Länge.

(iii) Kehren wir die Konstruktion der FWT um, d.h. bestimmen ausgehend von $a_{j,-}$ und $b_{j,-}$ die Koeffizienten $a_{j+1,-}$, so erhalten wir die Inverse Schnelle Wavelet-Transformation (*englisch*: Inverse Fast Wavelet Transform, IFWT). Dies entspricht einer wiederholten Anwendung der Synthesebank der in der FWT benutzten Filterbank. Das Schema ist in Abbildung A.1 skizziert.

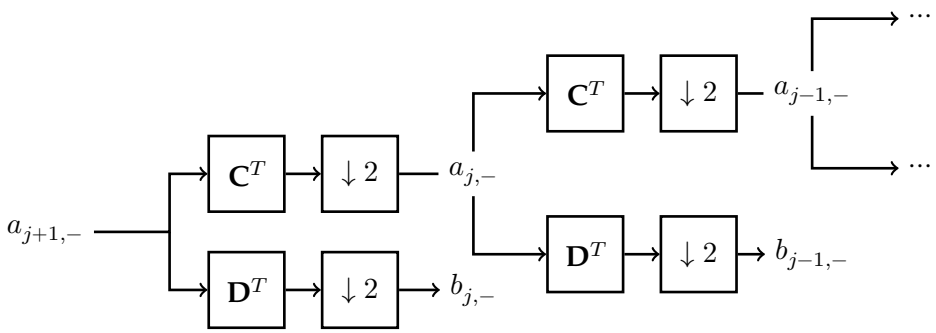


Abbildung 1.7.: Schema der Schnellen Wavelet-Transformation

(iv) In der Konstruktion sind die Basiskoeffizienten $a_{j,k}$ der Ausgangspunkt der FWT. Wollen wir für ein Signal $f \in L^2(\mathbb{R})$ (bzw. eine Diskretisierung des Signals) die FWT durchführen, müssten wir also zunächst die Basiskoeffizienten der orthogonalen Projektion von f nach V_j berechnen. Falls wir das Signal f gegeben haben, können wir dazu das Skalarprodukt von f mit den Basiselementen $\phi_{j,k}$ berechnen. Haben wir jedoch ein diskretisiertes Signal gegeben, müssen wir die Basiskoeffizienten approximieren; verschiedene Ansätze dazu werden z.B. von Strang und Nguyen [SN96, S. 232f.] vorgestellt.

Allerdings wird diese Feinheit in der Praxis sehr oft übergangen, so auch bei Wolter u. a. [Wol+21] und *PyWavelets* [Lee+19], einer Python-Bibliothek zur Berechnung von Wavelet-Transformationen. Da dies die Referenzen sind, mit denen wir uns in dieser Arbeit vergleichen werden, werden auch wir die FWT direkt auf das Signal statt die Koeffizienten anwenden.

1.3.3. Wavelet-Paket-Transformation

Wir betrachten die schematische Darstellung der Schnellen Wavelet-Transformation in Abbildung 1.7. Wenn wir die berechneten Basiskoeffizienten als Knoten auffassen, erzeugt die FWT einen gerichteten Binärbaum. Die Wurzel bilden die Eingangskoeffizienten $a_{J,-}$. Die Tiefe der Knoten im Baum entspricht der Skala der Koeffizienten. Um die Knoten der nächsten Skala zu erhalten, wird die Filterbank nur auf die Skalierungsfunktions-Koeffizienten $a_{j,-}$ an; die Wavelet-Koeffizienten $b_{j,-}$ sind Blätter des Binärbaumes. Somit besteht der entstehende Binärbaum aus einer linearen Anzahl von Knoten.

Wir wollen nun mit der *Wavelet-Paket-Transformation* eine weitere Wavelet-Transformation einführen. Dazu vervollständigen wir den Binärbaum der FWT, indem wir die Filterbank auch auf die Wavelet-Koeffizienten $b_{j,-}$ die Filterbank anwenden. Dann sind die Koeffizienten der j-ten Ebene des vollständigen Baumes das Ergebnis der j-skaligen Wavelet-Paket-Transformation. Haben die Eingangskoeffizienten eine Länge von L, so erzeugt die j-skalige Wavelet-Paket-Transformation 2^j Koeffizienten der Länge $L/2^J$. Man kann zeigen, dass dies einem Basiswechsel in die sogenannte *Walsh-Basis* entspricht [SN96, S. 72f.]. Die Basisfunktionen ergeben sich aus der Skalierungsfunktion ϕ durch eine dem Pfad im vollständigen Baum entsprechenden rekursiven Anwendung der Zwei-Skalen-Gleichung

und der Wavelet-Gleichung. Entspricht eine Kante dem Tiefpassfilter, wenden wir die Zwei-Skalen-Gleichung an, für den Hochpassfilter die Wavelet-Gleichung.

1.3.4. Daubechies-Wavelets und Symlets

Wir wollen im folgenden die Konstruktion zweier verwandter Wavelet-Familien skizzieren: der *Daubechies-Wavlets* und der *Symlets*. Die Daubechies-Wavelets sind nach Daubechies [Dau88] benannt, die diese zuerst beschrieb. Die Symlets sind eine symmetrischere Abwandlung der Daubechies-Wavelets. Für die Herleitung richten wir uns nach Strang und Nguyen [SN96] und Daubechies [Dau92].

Beide Familien entstehen, indem wir die Analysefilter der Filterbank als sogenannte *maxflat-Filter* konstruieren. Das sind kausale FIR-Filter, deren Frequenzantwort bei 0 und $\pm\pi$ maximal flach sind. Dadurch nähert sich die Frequenzantwort der Frequenzantwort des idealen Hoch- bzw. Tiefpassfilters an. Wir wollen die Filterkoeffizienten eines solchen maxflat-Tiefpassfilters $\mathbf{C}$ der Länge $2p$ konstruieren. Der zugehörige Hochpassfilter ergibt sich aus Gleichung (1.5).

Da wir eine orthonormale Filterbank erhalten wollen, müssen die Filterkoeffizienten $\mathbf{c}$ die p Bedingungen aus Gleichung (1.6) erfüllen. Die verbleibenden p Freiheitsgrade verwenden wir dazu, $C(\omega)$ bei $\omega = 0$ und $\omega = \pi$ maximal flach zu wählen. Dies entspricht den p Bedingungen

$$C(\pi) = C'(\pi) = \cdots = C^{(p-1)}(\pi) = 0,$$

also einer p-fachen Nullstelle von $C(\omega)$ bei π. Aus Symmetriegründen gilt dies für die Ableitungen dann auch für $\omega = 0$. Wir können somit $C(\omega)$ schreiben als

$$C(\omega) = \sum_{n=0}^{2p-1} \mathbf{c}(n)\mathrm{e}^{-\mathrm{i}n\omega} = \left(\frac{1+\mathrm{e}^{-\mathrm{i}\omega}}{2}\right)^p R(\mathrm{e}^{-\mathrm{i}\omega}),$$

wobei R ein Polynom vom Grad $p-1$ ist. Nun kann man für das trigonometrische Polynom $|R(z)|^2$ eine konkrete Form herleiten [vgl. Dau92, S. 171f.]. Dann erhalten wir die Koeffizienten des Polynoms R, indem wir $|R(z)|^2$ spektral faktorisieren in

$$|R(z)|^2 = R(z)\overline{R(z)}.$$

Diese Faktorisierung ist jedoch nicht eindeutig. Wählen wir diese so, dass alle Wurzeln von R im oder auf dem Einheitskreis liegen, erhalten wir die Daubechies-Wavelets. Bemerkenswert ist, dass wir für den Spezialfall $p = 1$ die bereits bekannten Haar-Wavelets erhalten. Allerdings sind die Daubechies-Filter für größere Längen sehr asymmetrisch. Für $p > 1$ ist es nicht möglich, die Wurzeln so zu wählen, dass ein symmetrischer Filter resultiert [Dau92]. Die Wahl der Wurzeln, die einem symmetrischen Filter am nächsten kommt, ergeben die Symlets. In Abbildung A.2 sind die Skalierungsfunktionen und Mutterwavelets für Daubechies-Wavelets und Symlets der Längen $2p = 8, 16, 24$ dargestellt.

2. Randbehandlung bei Wavelets

In numerischen Anwendungsfällen, insbesondere mit Bezug zum maschinellen Lernen, haben die Signale oft endliche Längen. Bilder etwa, ein weitverbreiteter Anwendungsfall der Signalverarbeitung, sind zweidimensionale diskrete Signale mit endlicher Breite und Höhe. Die in Kapitel 1 eingeführte Theorie basiert wesentlich auf der Anwendung von Filtern auf ein Signal. Dies führt allerdings bei einem endlichen Signal zu Problemen: die Berechnung der Faltung mit einem kausalen FIR-Filter $\sum_{k=0}^{N-1} \mathbf{h}(k)\mathbf{x}(n-k)$ ist am Signalrand nicht möglich ist, da auf undefinierte Signaleinträge zugegriffen werden müsste.

Verfahren zur Realisierung der Filteranwendung an den Signalrändern, bezeichnen wir als *Randbehandlung*. Wir lernen in diesem Kapitel zwei Ansätze dazu kennen: Signalfortsetzung (Abschnitt 2.1) und Randfilter (Abschnitt 2.2). Anschließend führen wir in Abschnitt 2.3 mithilfe von Randfiltern eine Matrix-basierte Formulierung der FWT ein. In Abschnitt 2.4 besprechen wir eine Möglichkeit, Signale in Abhängigkeit von Zeit und Frequenz zu visualisieren, und nutzen diese, um die beiden Randbehandlungsansätze zu vergleichen.

2.1. Randbehandlung durch Signalfortsetzung

Ein naheliegender Ansatz für die Randbehandlung ist, das endliche Signal zu einem unendlichen Signal fortzusetzen und die entwickelten Methoden auf dieses fortgesetzte Signal anzuwenden. Dazu betrachten wir, wie eine Filterbank auf das endliche Signal angewandt werden kann, da daraus die anderen Anwendungsfälle wie die DWT abgeleitet werden können. Wir wollen dies mit einem Beispiel einführen, auf das wir in späteren Abschnitten zurückkommen werden.

Beispiel 2.1.1. Angenommen, wir wollen die FWT mit FIR-Filtern der Länge N auf ein endliches Signal $\mathbf{x}$ der Länge L anwenden. Sei $\mathbf{H}_b$ die Block-Toeplitz-Matrix der Filterbank der Transformation. Dann müssten wir eigentlich das Matrix-Vektor-Produkt $\mathbf{H}_b\mathbf{x}$ bilden. Dies ist jedoch unmöglich, da $\mathbf{H}_b$ eine unendliche Matrix ist.

Um Abhilfe zu schaffen, setzen wir $\mathbf{x}$ zu einem unendlichen Signal in $l^2(\mathbb{Z})$ fort, indem wir an die Signalränder Nullen anfügen. Dadurch erhalten wir das unendliche Signal $\tilde{\mathbf{x}}$ mit

$$\tilde{\mathbf{x}}(k) = \begin{cases} 0 & k < 0 \\ \mathbf{x}(k) & 0 \leq k < L \\ 0 & k \geq L \end{cases}.$$

Diese Fortsetzung ist äquivalent zum Entfernen aller Spalten von $\mathbf{H}_b$ mit Index kleiner 0 bzw. größer $L-1$. Somit erhalten wir eine Matrix mit passender endlicher Spaltenanzahl

und können das Matrix-Vektor-Produkt bilden. Allerdings enthält die angepasste Matrix noch unendlich viele Zeilen, sodass das Filterergebnis unendlich lang ist.

Wir erinnern uns daran, dass $\mathbf{H}_b$ eine Block-Toeplitz-Matrix ist und die Filterbank aus FIR-Filtern besteht. Daher sind nur endlich viele (Block-)Diagonalen von $\mathbf{H}_b$ nicht Null und alle Zeilen der zugeschnittenen Matrix unterhalb bzw. oberhalb eines gewissen Indizes bestehen nur aus Nullen. Entfernen wir diese „Null-Zeilen“, bleibt eine endliche $K \times L$-Matrix $\tilde{\mathbf{H}}_b$ über, die wir mit dem Signal multiplizieren können.

Für eine Filterlänge $N = 6$ und Signallänge $L = 8$ erhalten wir also

$$\tilde{\mathbf{H}}_b\mathbf{x} = \begin{bmatrix} \mathbf{c}(4) & \mathbf{c}(5) & & & & & & \\ \mathbf{d}(4) & \mathbf{d}(5) & & & & & & \\ \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) & & & & \\ \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) & & & & \\ \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) & & \\ \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) & & \\ & & \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) \\ & & \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) \\ & & & & \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) \\ & & & & \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) \\ & & & & & & \mathbf{c}(0) & \mathbf{c}(1) \\ & & & & & & \mathbf{d}(0) & \mathbf{d}(1) \end{bmatrix} \begin{bmatrix} \mathbf{x}(0) \\ \mathbf{x}(1) \\ \mathbf{x}(2) \\ \mathbf{x}(3) \\ \mathbf{x}(4) \\ \mathbf{x}(5) \\ \mathbf{x}(6) \\ \mathbf{x}(7) \end{bmatrix} = \begin{bmatrix} \mathbf{y}(0) \\ \mathbf{y}(1) \\ \mathbf{y}(2) \\ \mathbf{y}(3) \\ \mathbf{y}(4) \\ \mathbf{y}(5) \\ \mathbf{y}(6) \\ \mathbf{y}(7) \\ \mathbf{y}(8) \\ \mathbf{y}(9) \\ \mathbf{y}(10) \\ \mathbf{y}(11) \end{bmatrix}. \quad (2.1)$$

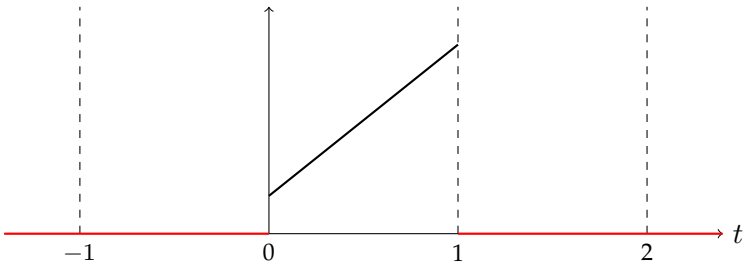
Die resultierende Matrix hat in diesem Fall $K = 12$ Zeilen, womit das Ergebnis der Filter-Anwendung länger als das Ausgangssignal ist.

Die Randbehandlungsmethode in diesem Beispiel arbeitet mit einer Fortsetzung des Signals mit Nullen. Eine solche Fortsetzung mit einem konstanten Wert bezeichnet man auch als *Padding*. Daneben gibt es noch andere Verfahren der Signalfortsetzung [vgl. KV89]. In Abbildung 2.1 sind vier häufig verwendete Methoden der Signalfortsetzung skizzenhaft dargestellt. All diese Verfahren sind im Ablauf sehr ähnlich zum Eingangsbeispiel. Lediglich die Matrix $\tilde{\mathbf{H}}_b$ wird soweit nach links und rechts fortgesetzt, dass keiner der FIR-Filter in den Zeilen „durchgeschnitten“ wird, wie dies im Beispiel geschehen ist.

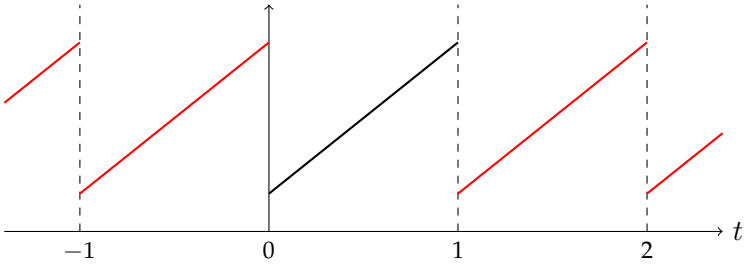
Bei diesen Fortsetzungen des Signals kommt es an den Signalenden zu Unstetigkeitsstellen in der Fortsetzung oder in seiner Ableitung. Diese Unstetigkeiten führen in der FWT zu großen Koeffizienten und damit zu Randartefakten [Jl01, S. 144].

Darüber hinaus haben wir ein weiteres Problem einiger dieser Signalfortsetzungen bereits im Eingangsbeispiel kennengelernt. Mit der dort eingeführten Konstruktion[1] erzeugt die Anwendung eines FWT-Schritts auf ein Signal für alle orthonormalen Wavelets mit Ausnahme des Haar-Wavelets ein längeres Signal, wobei die Verlängerung proportional zur Filterlänge ist [Jl01]. Dies liegt daran, dass auch Zeilen, bei denen nur noch ein Teil des Filters mit dem Eingangssignal „überlappt“, für das Ausgangssignal relevant sind, sodass sich das Signal an beiden Enden verlängert. Dies gilt aufgrund der kurzen Länge nicht für das Haar-Wavelet. Wichtig ist, dass das volle verlängerte Ausgangssignal für die perfekte Rekonstruierbarkeit notwendig ist und nicht einfach gekürzt werden kann.

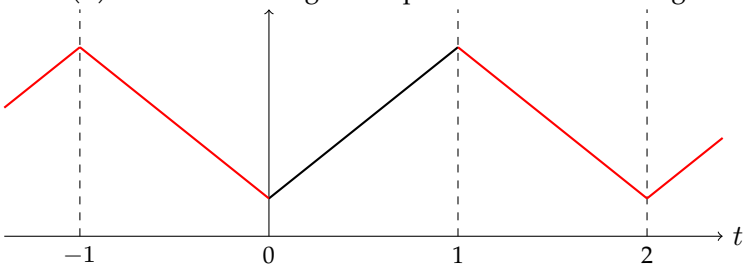
[1] Es gibt für manche Varianten wie etwa die periodische Fortsetzung bessere Konstruktionen, die dieses Problem vermeiden [Jl01, Abschnitt 10.4].



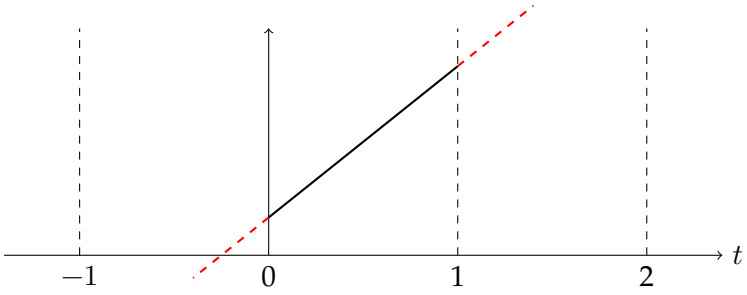
(a) Randbehandlung durch Padding mit Nullen.



(b) Randbehandlung durch periodische Fortsetzung.



(c) Randbehandlung durch symmetrische Fortsetzung.



(d) Randbehandlung durch Extrapolation.

Abbildung 2.1.: Vier Arten der Randbehandlung eines Beispielsignals mit beschränktem Träger. Das Beispielsignal ist schwarz, die Fortsetzung rot.

2.2. Randbehandlung mittels Randfiltern

Alle bisher behandelten Ansätze zur Randbehandlung liefen darauf hinaus, das endliche Signal auf verschiedene Arten zu einem unendlichen Signal fortzusetzen. Die Randbehandlung, die wir im folgenden einführen, ist grundverschieden. Nicht das Signal wird an die Filter angepasst, sondern die Filter an das Signal.

Dies geschieht, indem wir eine Eigenschaft, die bislang alle Filter implizit hatten, vernachlässigen: die *Zeitinvarianz*. Die Matrixdarstellung der Filter waren Toeplitz-Matrizen, hatten also konstante Diagonaleinträge. Sie haben also alle Stellen des Signals gleich gefiltert. Wir wollen nun die Filter an den Signalrändern anpassen, um trotz endlicher Signallänge Randeffekte zu vermeiden. Die so angepassten Filter sind nicht mehr zeitinvariant; ihre Matrixdarstellung wird einige veränderte Zeilen beinhalten.

Die hier besprochene Konstruktion der sogenannten Gram-Schmidt-Randfilter basiert auf Herley und Vetterli [HV94]; wir orientieren uns darüber hinaus an Jensen und la Cour-Harbo [Jl01, Abschnitt 10.3] und Strang und Nguyen [SN96, Abschnitt 8.5]. Wir passen zunächst die Filter an den Rändern an und leiten anschließend eine daraus resultierende Basis von $L^2([0,1])$ her. Wie wir feststellen werden, ist das Verfahren sehr effizient und bietet gute Kontrolle über die Filterkoeffizienten.

Vergleichbare Ansätze sind z.B. von Meyer [Mey91] und Cohen, Daubechies und Vial [CDV93] vorgeschlagen worden. Der Hauptunterschied zum hier vorgestellten Verfahren ist, dass dabei zunächst die Waveletbasis angepasst wird, um eine Basis von $L^2([0,1])$ zu konstruieren, und daraus die Filterkoeffizienten abgeleitet werden. Dies bietet den Vorteil, dass die Eigenschaften der Basis besser zu kontrollieren sind und dadurch z.B. bei Daubechies-Wavelets die maxflat-Eigenschaft erhalten werden kann. Allerdings ist die Filterberechnung dadurch deutlich aufwendiger [Jl01].

2.2.1. Gram-Schmidt-Randfilter

Wie bei der Signalfortsetzung ist es unser Ziel, aus der unendlichen Block-Toeplitz-Matrix $\mathbf{H}_b$ eine endliche Matrix zu konstruieren, mit der wir ein endliches Signal multiplizieren können. Allerdings soll das Problem, dass sich ein Signal durch das Filtern verlängert, vermieden werden. Gleichzeitig wollen wir die Orthogonalität der Filterbank beibehalten. Somit wollen wir also für ein Signal der Länge L eine quadratische, orthonormale $L \times L$-Matrix $\mathbf{H}_L$ konstruieren.

Der Grund dafür, dass beim Filtern mit Signalfortsetzung das Signal verlängert wurde, sind die „unvollständigen“ Zeilen am oberen und unteren Rand der Matrix. Daher ist unser Ausgangspunkt die L-spaltige Matrix

$$\mathbf{H}_{\text{in}}^{(d)} := \left[\begin{array}{c|cccccccc|c} & \mathbf{c}(0) & \mathbf{c}(1) & \dots & \dots & \mathbf{c}(N-2) & \mathbf{c}(N-1) & & & \\ & \mathbf{d}(0) & \mathbf{d}(1) & \dots & \dots & \mathbf{d}(N-2) & \mathbf{d}(N-1) & & & \\ & & \ddots & \ddots & & & \ddots & \ddots & & \\ & & & \ddots & \ddots & & & \ddots & \ddots & \\ & & & & \mathbf{c}(0) & \mathbf{c}(1) & \dots & \dots & \mathbf{c}(N-2) & \mathbf{c}(N-1) \\ & & & & \mathbf{d}(0) & \mathbf{d}(1) & \dots & \dots & \mathbf{d}(N-2) & \mathbf{d}(N-1) \\ \underbrace{\quad}_{d \text{ Nullen}} & & & & & & & & & \underbrace{\quad}_{d \text{ Nullen}} \end{array}\right],$$

(2.2)

die für $d \geq 0$ aus $\mathbf{H}_b$ hervorgeht, indem $L - 2d$ Spalten mit „vollständigen“ Zeilen herausgeschnitten werden und auf beiden Seiten weitere d Spalten hinzugenommen werden, die dann nur Nullen enthalten. Die d weiteren Spalten dienen der Optimierung der Randfilter, worauf wir am Ende dieses Abschnittes zurückkommen werden; bis dahin wählen wir $d = 0$ und notieren $\mathbf{H}_{\text{in}} := \mathbf{H}_{\text{in}}^{(0)}$. Dadurch, dass wir nur vollständige Zeilen auswählen, ist $\mathbf{H}_{\text{in}}^{(d)}$ nicht quadratisch.[2] Man kann sich leicht überlegen, dass $\mathbf{H}_{\text{in}}^{(d)}$ aus L Spalten und $(L - N + 2 - 2d)$ Zeilen besteht.

Beispiel 2.2.1. Für eine Filterlänge $N = 6$ und Signallänge $L = 8$ erhalten wir

$$\text{für } d = 0 \qquad \mathbf{H}_{\text{in}} = \begin{bmatrix} \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) & & \\ \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) & & \\ & & \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) \\ & & \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) \end{bmatrix},$$

$$\text{für } d = 1 \qquad \mathbf{H}_{\text{in}}^{(1)} = \begin{bmatrix} 0 & \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) & 0 \\ 0 & \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) & 0 \end{bmatrix}.$$

Da die Matrix aus einer orthonormalen Filterbank hervorgeht, gilt auch

$$(\mathbf{H}_{\text{in}}^{(d)}) \cdot (\mathbf{H}_{\text{in}}^{(d)})^T = \mathbf{I}$$

Somit bilden die Zeilen von $\mathbf{H}_{\text{in}}^{(d)}$ ein Orthonormalsystem. Wir wollen nun $\mathbf{H}_L$ aus $\mathbf{H}_{\text{in}}^{(d)}$ konstruieren, indem wir oben und unten weitere Zeilen hinzufügen, ohne die Orthonormalität zu verletzen. Das bedeutet, dass wir die Zeilen von $\mathbf{H}_{\text{in}}^{(d)}$ zu einer Orthonormalbasis von $\mathbb{R}^L$ vervollständigen müssen. Die oben hinzugefügten Zeilen nennen wir *linke Randfilter*; die unten hinzugefügten Zeilen *rechte Randfilter*. Wir wollen gleich viele linke wie rechte Randfilter. Eine kurze Rechnung ergibt, dass wir jeweils

$$\frac{N-2}{2} + d$$

Randfilter hinzufügen müssen.

Bevor wir zur Konstruktion kommen, müssen wir Annahmen an die Filterlänge N und die Signallänge L stellen:

(i) L muss gerade sein, damit es gleich viele Hoch- wie Tiefpasszeilen gibt;

(ii) N muss gerade sein, damit es gleich viele linke wie rechte Randfilter gibt;

(iii) $L \gg N$, damit die linken und rechten Randfilter sich nicht schneiden können.[3]

[2] Außer für den Sonderfall der Haar-Wavelets. Für diesen ist $\mathbf{H}_{\text{in}}$ bereits die gesuchte Matrix.

[3] Diese Annahme ist im maschinellen Lernen oft sowieso erfüllt, da die Filterlänge im Vergleich zur Signallänge häufig vernachlässigbar klein ist. Zu Problemen kann es allerdings kommen, wenn viele Skalen bei der DWT verwendet werden.

An sich ist die Konstruktion einer Matrix $\mathbf{H}_L$ aus $\mathbf{H}_{\text{in}}$ nicht schwierig. Da die Zeilen von $\mathbf{H}_{\text{in}}$ ein Orthonormalsystem bilden, sind sie auch linear unabhängig voneinander. Wir können also die Zeilen von $\mathbf{H}_{\text{in}}$ beliebig zu einer Basis von $\mathbb{R}^L$ vervollständigen, die wir anschließend mit dem Gram-Schmidt-Verfahren orthonormalisieren. Somit könnten wir die Basiselemente als Zeilen von $\mathbf{H}_L$ wählen und hätten unsere Anforderungen erfüllt.

Lemma 2.2.2 (Herley und Vetterli [HV94])**.** *Sei* $\mathbf{H}_{\text{in}}$ *wie in* (2.2) *definiert mit* $d = 0$ *und gerader Filterlänge* N*. Dann gilt für Vektoren, die zu allen Zeilen von* $\mathbf{H}_{\text{in}}$ *orthogonal sind, dass höchstens die jeweils äußeren* $N - 2$ *Einträge nicht Null sind.*

Das vorherige Lemma zeigt, dass alle Randfilter nur an den Rändern von Null verschiedene Einträge haben. Wir würden jedoch eine Lösung bevorzugen, bei der die oben hinzugefügten Zeilen nur am linken Rand Einträge haben, die ungleich Null sind, und analog die unten hinzugefügten Zeilen nur am rechten Rand. Dies ist allerdings im Allgemeinen nicht erfüllt, weshalb wir etwas überlegter vorgehen müssen. Dazu ziehen wir bei der Basisvervollständigung nur noch Elemente in Betracht, die ausschließlich an einem der beiden Enden von der Null verschiedene Einträge haben. Dafür richten wir unseren Blick erneut auf die Matrix $\tilde{\mathbf{H}}_b$ aus (2.1). Die mittleren Zeilen von $\tilde{\mathbf{H}}_b$ bilden $\mathbf{H}_{\text{in}}$. Wir nehmen jeweils die ersten $(N-2)/2$ Zeilen oberhalb und unterhalb hinzu und erhalten eine quadratische Matrix

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{\text{li}} \\ \mathbf{H}_{\text{in}} \\ \mathbf{H}_{\text{re}} \end{bmatrix} \in \mathbb{R}^{L \times L},$$

wobei

$$\mathbf{H}_{\text{li}}, \mathbf{H}_{\text{re}} \in \mathbb{R}^{\frac{N-2}{2} \times L}$$

die oben bzw. unten hinzugenommen Zeilen sind. Aufgrund der Konstruktion sind nur die ersten bzw. letzten $N-2$ Einträge von $\mathbf{H}_{\text{li}}$ bzw. $\mathbf{H}_{\text{re}}$ von Null verschieden, sodass wir diese schreiben können als

$$\mathbf{H}_{\text{li}} = \begin{bmatrix} \mathbf{L} & \mathbf{0} \end{bmatrix}, \quad \mathbf{H}_{\text{re}} = \begin{bmatrix} \mathbf{0} & \mathbf{R} \end{bmatrix} \qquad \text{mit } \frac{N-2}{2} \times (N-2)\text{-Matrizen } \mathbf{L} \text{ und } \mathbf{R}. \tag{2.3}$$

Beispiel 2.2.3. Für eine Filterlänge $N = 6$ und Signallänge $L = 8$ müssen wir oben und unten jeweils $N/2 - 1 = 2$ Zeilen an $\mathbf{H}_{\text{in}}$ anfügen. So erhalten wir

$$\begin{bmatrix} \begin{bmatrix} \mathbf{L} & \mathbf{0} \end{bmatrix} \\ \mathbf{H}_{\text{in}} \\ \begin{bmatrix} \mathbf{0} & \mathbf{R} \end{bmatrix} \end{bmatrix} = \left[\begin{array}{cccc|cccc} \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) & & & & \\ \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) & & & & \\ \hline \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) & & \\ \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) & & \\ & & \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) \\ & & \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) \\ \hline & & & & \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) \\ & & & & \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) \end{array}\right]$$

Satz 2.2.4 (Randfilter für $d = 0$)**.** *Sei* $\mathbf{H}_{\text{in}}$ *wie in* (2.2) *und sei* $\mathbf{L}$ *und* $\mathbf{R}$ *wie in* (2.3)*. Sei* $\mathbf{L}'$ *und* $\mathbf{R}'$ *die mittels Gram-Schmidt-Verfahren orthonormalisierten Zeilen von* $\mathbf{L}$ *bzw.* $\mathbf{R}$*. Dann ist*

$$\begin{bmatrix} [\mathbf{L}' \quad \mathbf{0}] \\ \mathbf{H}_{\text{in}} \\ [\mathbf{0} \quad \mathbf{R}'] \end{bmatrix}$$

eine orthonormale $L \times L$*-Matrix.*

Beweis. Man kann zeigen, dass $\mathbf{H}$ vollen Rang hat [vgl. Her+93]. Die Zeilen von $\mathbf{H}$ bilden also eine Basis von $\mathbb{R}^L$.

Wir haben uns bereits überlegt, dass die Zeilen von $\mathbf{H}_{\text{in}}$ ein Orthonormalsystem bilden. Aufgrund der Orthonormalität der zugrundeliegenden Filterbank sind die Zeilen von $\mathbf{H}_{\text{li}}$ und $\mathbf{H}_{\text{re}}$ orthogonal zu den Zeilen von $\mathbf{H}_{\text{in}}$. Auch sind die Zeilen von $\mathbf{H}_{\text{li}}$ und $\mathbf{H}_{\text{re}}$ jeweils orthogonal zueinander, da $L \gg N$ ist und sich somit die Spalten mit Nicht-Null-Einträgen beider Matrizen nicht überschneiden. Allerdings sind die Zeilen von $\mathbf{H}_{\text{li}}$ bzw. $\mathbf{H}_{\text{re}}$ noch nicht normiert und untereinander noch nicht orthogonal.

Wenden wir das Gram-Schmidt-Verfahren auf die Zeilen von $\mathbf{H}_{\text{li}}$ an, spannen die Zeilen anschließend weiterhin denselben Unterraum auf. Somit bleiben die Zeilen orthogonal zu den Zeilen von $\mathbf{H}_{\text{in}}$ und $\mathbf{H}_{\text{re}}$. Gleichzeitig sind für alle Zeilen von $\mathbf{H}_{\text{li}}$ nur höchstens die ersten $N - 2$ Einträge von Null verschieden. Daher ist es äquivalent, das Gram-Schmidt-Verfahren nur auf die ersten $N-2$ Eintäge der Zeilen, also auf $\mathbf{L}$, anzuwenden und so diese Teile der Zeilen zu orthonormalisieren. Analog können wir dies für $\mathbf{H}_{\text{re}}$ bzw. $\mathbf{R}$ zeigen. □

Wir erhalten somit ein Verfahren zur Berechnung der linken und rechten Randfilter, bei dem die konstruierten Randfilter nur an jeweils einem Rand von Null verschiedene Einträge haben:

Algorithmus 1 Gram-Schmidt-Randfilter für $d = 0$

1: Konstruiere $(L - N + 2) \times L$-Matrix $\mathbf{H}_{\text{in}}$ wie in (2.2).
2: Konstruiere $((N-2)/2) \times (N-2)$-Matrizen $\mathbf{L}$ und $\mathbf{R}$ wie in (2.3).
3: Orthonormalisiere mittels Gram-Schmidt-Verfahren die Zeilen von $\mathbf{L}$ bzw. $\mathbf{R}$ und speichere diese in $\mathbf{L}'$ bzw. $\mathbf{R}'$.
4: **return** $L \times L$-Orthonormalmatrix $\mathbf{H}_L = \begin{bmatrix} [\mathbf{L}' \quad \mathbf{0}] \\ \mathbf{H}_{\text{in}} \\ [\mathbf{0} \quad \mathbf{R}'] \end{bmatrix}$

Lemma 2.2.5 (Laufzeit der Randfilterberechnung)**.** *Die asymptotische Laufzeit der Gram-Schmidt-Randfilterberechnung für* $d = 0$*, eine Signallänge* L *und Filterlänge* N *ist*

$$\mathcal{O}(LN + N^3).$$

Betrachten wir N *als konstant, ist die Konstruktion also linear in der Signallänge.*

Beweis. Wir überlegen uns den Aufwand für jeden der vier Schritte des Algorithmus. Der Aufwand zur Konstruktion von $\mathbf{L}$ und $\mathbf{R}$ entspricht der Anzahl ihrer Einträge, also jeweils

$\mathcal{O}(N^2)$. Wir können $\mathbf{H}_{\text{in}}$ und $\mathbf{H}_L$ als dünnbesetzte Matrizen realisieren. Dann entspricht der Aufwand der Konstruktion der Anzahl der von Null verschiedenen Einträge. Für $\mathbf{H}_{\text{in}}$ sind dies $\mathcal{O}(LN)$; für $\mathbf{H}_L$ sind es $\mathcal{O}(LN + 2N^2)$. Der Aufwand für die Orthonormalisierung ist jeweils $\mathcal{O}(N^3)$ [vgl. GV12, Abschnitt 5.2.8]. Somit ergibt sich ein Gesamtaufwand von

$$\mathcal{O}(LN + 2N^2 + N^3) = \mathcal{O}(LN + N^3). \qquad \square$$

Bemerkung. Ein Vorteil dieser Berechnung von Randfiltern ist, dass sie unabhängig von der Signallänge L ist: Sowohl die Anzahl der Randfilter als auch die Orthonormalisierung hängt nur von den gewählten Filtern ab. Möchte man also eine Filterbank auf Signale verschiedener Längen anwenden, muss man die Konstruktion der Randfilter nicht wiederholen.

2.2.2. Optimierung der Randfilter

Wir haben im vorherigen Abschnitt Gram-Schmidt-Randfilter für $d = 0$ konstruiert. Allerdings waren wir dabei darauf festgelegt, jeweils $(N-2)/2$ linke bzw. rechte Randfilter zu bilden. Wir werden nun sehen, wie wir diese Anzahl erhöhen und die dadurch gewonnenen Freiheitsgrade zur Optimierung der Randfilter nutzen können. Wir orientieren uns dazu an Herley und Vetterli [HV94].

Sei $d > 0$. Dann enthält die Matrix $\mathbf{H}_{\text{in}}^{(d)}$ am linken und rechten Ende jeweils d Spalten aus Nullen. Schneiden wir diese ab, erhalten wir die Matrix $\mathbf{H}_{\text{in}}$ – allerdings für eine reduzierte Signallänge $L-2d$. Darauf können wir die Konstruktion der Randfilter für $d = 0$ anwenden. Fügen wir die resultierenden orthonormalisierten Randfiltermatrizen $\mathbf{L}'$ und $\mathbf{R}'$ zentriert über $\mathbf{H}_{\text{in}}^{(d)}$ an, erhalten wir eine $L - 2d \times L$-Matrix, die links und rechts d Nullspalten hat. Wir ergänzen nun in der oberen linken und unteren rechten Ecke eine $d \times d$-Einheitsmatrix und bekommen so die $L \times L$-Matrix

$$\mathbf{T} := \begin{bmatrix} \mathbf{I}_d & & \\ & [\mathbf{L}' \quad \mathbf{0}] & \\ & \mathbf{H}_{\text{in}}^{(d)} & \\ & [\mathbf{0} \quad \mathbf{R}'] & \\ & & \mathbf{I}_d \end{bmatrix}. \tag{2.4}$$

Beispiel 2.2.6. Für eine Filterlänge $N = 6$ und Signallänge $L = 8$ müssen wir oben und unten jeweils $N/2 - 1 + d = 2$ Zeilen an $\mathbf{H}_{\text{in}}^{(d)}$ anfügen. Für $d = 1$ erhalten wir

$$\mathbf{T} = \left[\begin{array}{c|cccccccc} 1 & & & & & & & & \\ \hline & \boxed{\begin{matrix}\mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) \\ \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5)\end{matrix}} & & & & & & \\ & \mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) & \mathbf{c}(4) & \mathbf{c}(5) & & \\ & \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3) & \mathbf{d}(4) & \mathbf{d}(5) & & \\ & & & \boxed{\begin{matrix}\mathbf{c}(0) & \mathbf{c}(1) & \mathbf{c}(2) & \mathbf{c}(3) \\ \mathbf{d}(0) & \mathbf{d}(1) & \mathbf{d}(2) & \mathbf{d}(3)\end{matrix}} & & & \\ & & & & & & & 1 \end{array}\right]$$

Wir überlegen uns, dass wir mit dieser Konstruktion tatsächlich $d + (N-2)/2$ Randfilter zu der Matrix $\mathbf{H}_{\text{in}}^{(d)}$ erzeugt haben, da die Zeilen der Einheitsmatrix orthogonal zu den Null-Einträgen unterhalb der Matrix sind. Jedoch haben die Zeilen der Einheitsmatrix keine guten Filtereigenschaften. Wir können aber das folgende Lemma nutzen, um Eigenschaften der Randfilter zu verbessern:

Lemma 2.2.7 (Herley und Vetterli [HV94])**.** *Sei $d \geq 0$ und $\mathbf{T}$ wie in Gleichung (2.4). Dann ist genau dann $\mathbf{T}'$ eine Ergänzung von $\mathbf{H}_{\text{in}}^{(d)}$ mit orthonormalen linken und rechten Randfiltern zu einer quadratischen Matrix, wenn quadratische orthonormale Matrizen $\mathbf{U}_l$ und $\mathbf{U}_r$ der Größe $(N-2)/2 + d$ existieren, sodass*

$$\mathbf{T}' = \begin{bmatrix} \mathbf{U}_l & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{U}_r \end{bmatrix} \mathbf{T}.$$

Bemerkung. Wir können also die in Gleichung (2.4) konstruierten Randfilter in alle anderen Randfilter transformieren, indem wir sie mit einer orthonormalen Matrix multiplizieren. Somit ist es möglich, auch Eigenschaften der Randfilter wie etwa die Frequenzselektivität zu optimieren, indem wir passende Orthonormalmatrizen wählen.

2.2.3. Waveletbasis auf dem Intervall

Es ist möglich, für ein geschlossenes Intervall I aus Gram-Schmidt-Randfiltern eine Wavelet-Basis von $L^2(I)$ zu konstruieren. Wir wollen die Konstruktion an dieser Stelle kurz skizzieren. Für Details sei auf Herley und Vetterli [HV94] verwiesen.

Wir gehen von einer Multiskalenanalyse $\{V_j\}_{j\in\mathbb{Z}}$ von $L^2(\mathbb{R})$ mit zugehöriger Skalierungsfunktion ϕ und Mutterwavelet w aus. Für eine geeignet feine Skala J betrachten wir dann analog zur Multiskalenanalyse die Zerlegung

$$L^2(I) = \hat{V}_J \oplus \bigoplus_{j=J}^{\infty} \hat{W}_j.$$

Dabei entspricht $\hat{V}_J$ dem Unterraum V_J und $\hat{W}_j$ für $j \geq J$ den Waveletunterraum W_j.

Wir wollen Basen dieser Unterräume konstruieren. Zunächst übernehmen wir in die Basen von $\hat{V}_J$ bzw. $\hat{W}_j$ genau die Elemente von V_J bzw. W_j, deren Träger komplett in I liegt. Allerdings fehlen uns dadurch Basisfunktionen mit Träger an den Intervallrändern. Durch iterierte Anwendung der DWT unter Verwendung der Randfilter erzeugen wir Rand-Skalierungsfunktionen und Rand-Wavelets, die die Basen vervollständigen.

Das Ergebnis dieser Konstruktion ist eine orthonormale Waveletbasis von $L^2(I)$. Allerdings verlieren wir analog zum diskreten Fall die Zeitinvarianz-Eigenschaft an den Rändern. So werden die Rand-Skalierungsfunktionen und Rand-Wavelets nicht mehr durch Skalierung und Translation aus der Skalierungsfunktion bzw. dem Mutterwavelet erzeugt. Stattdessen ergibt die iterierte Anwendung der DWT die zu $\hat{V}_J$ gehörenden Rand-Skalierungsfunktionen und zu $\hat{W}_J$ gehörenden Rand-Wavelets. Die Rand-Wavelets höherer Skalen ergeben sich dann durch Skalierung.

Wie eingangs beschrieben, erlaubt diese Konstruktion nur beschränkte Kontrolle über die konstruierte Waveletbasis. Zwar kann man zeigen, dass die konstruierten Rand-Skalierungsfunktionen außerhalb eines ε-Balls vom Rand genauso stetig wie die zugrundeliegende Skalierungsfunktion sind [HV94]. Jedoch werden Eigenschaften wie die maximale Flachheit im Allgemeinen bei dieser Konstruktion[4] am Rand nicht erhalten.

2.3. Schnelle Wavelet-Transformation mit dünnbesetzten Matrizen

Wir haben mit den Gram-Schmidt-Randfiltern eine Möglichkeit kennengelernt, die Anwendung einer Filterbank auf ein endliches Signal durch eine quadratische Matrix darzustellen. Dies wollen wir im folgenden dazu nutzen, für die in Abschnitt 1.3.2 eingeführte FWT eine elegante, matrixbasierte Formulierung für endliche Signale zu konstruieren. Dabei orientieren wir uns an Strang und Nguyen [SN96].

Wir erinnern uns daran, dass die FWT die DWT zwischen verschiedenen Skalen berechnet, indem eine orthonormale Filterbank wiederholt angewandt wird. Seien $\mathbf{C}^T$ und $\mathbf{D}^T$ die Analysefilter dieser Filterbank und seien $a_{j,-}$ die Koeffizienten der Basisdarstellung zur Skala j. Da wir die FWT auf endliche Signale anwenden wollen, nehmen wir an, dass wir $a_{j,-}$ als endlichen Vektor der Länge L schreiben können und dass die Filterbank FIR-Filter der Länge N verwendet. Da wir Gram-Schmidt-Randfilter nutzen wollen, nehmen wir außerdem an, dass N und L die dafür notwendigen Bedingungen erfüllen.

Mithilfe der Gram-Schmidt-Randfilter können wir die Filterbank als $L \times L$-Matrix $\mathbf{H}_L$ schreiben. Bei der Konstruktion dieser Matrix sind wir von der Block-Toeplitz-Darstellung der Filterbank ausgegangen, sodass die Zeilen von $\mathbf{H}_L$ abwechselnd dem Tief- und dem Hochpassfilter zuzuordnen sind. Wir setzen dabei die abwechselnde Zuordnung von $\mathbf{H}_{\text{in}}^{(d)}$ auf die angefügten Randfilter fort.

Wir wollen die Verschränkung der Zeilen der beiden Filter in $\mathbf{H}_L$ auflösen. Dazu fassen wir alle Zeilen des Tiefpassfilters in der Matrix $\mathbf{R}_L$ und alle Zeilen des Hochpassfilters in $\mathbf{S}_L$ zusammen. Beide Matrizen sind $(L/2) \times L$-Matrizen. Ordnen wir die Matrizen übereinander an, können wir die Anwendung der Filterbank auf die Koeffizienten $a_{j,-}$ in Blockform schreiben:

$$\hat{\mathbf{H}}_L a_{j,-} = \begin{bmatrix} \mathbf{R}_L \\ \mathbf{S}_L \end{bmatrix} a_{j,-} = \begin{bmatrix} a_{j-1,-} \\ b_{j-1,-} \end{bmatrix},$$

wobei $a_{j-1,-}$ und $b_{j-1,-}$ jeweils $L/2$ lang sind.

Wir gehen zur nächsten Skala über und wiederholen diese Schritte. So konstruieren wir die $(L/2) \times (L/2)$-Matrix $\hat{\mathbf{H}}_{L/2}$, die wir auf $a_{j-1,-}$ anwenden, um $a_{j-2,-}$ und $b_{j-2,-}$ zu erhalten. Auch diese Skala können wir durch eine $L \times L$-Matrix ausdrücken, indem wir $\hat{\mathbf{H}}_{L/2}$ um eine passende Einheitsmatrix erweitern, mit der wir nur $b_{j-1,-}$ multiplizieren. In

[4] im Gegensatz zu z.B. Cohen, Daubechies und Vial [CDV93]

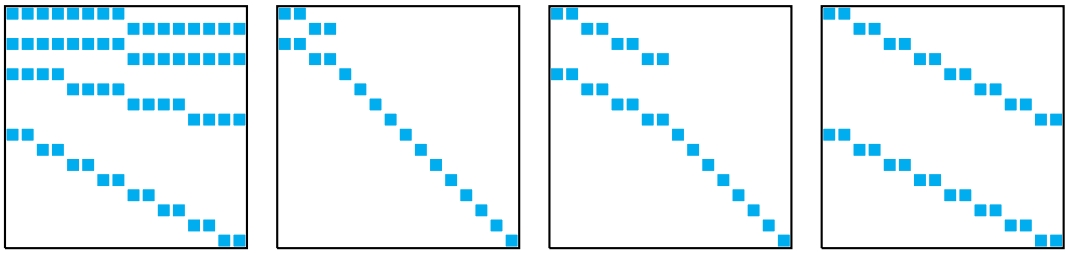


Abbildung 2.2.: Struktur der Analysematrizen der FWT für Haar Wavelets auf verschiedenen Skalen [nach WG21]. Einträge der dünnbesetzten Matrizen, die nicht Null sind, sind farblich markiert. Die Analysematrix (links) ist das Produkt der Analysematrizen der einzelnen Skalen, die jeweils die FWT-Operation auf einer Skala codieren.

Blockschreibweise gilt dann

$$\begin{bmatrix} a_{j-2,-} \\ b_{j-2,-} \\ b_{j-1,-} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{H}}_{L/2} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{L/2} \end{bmatrix} \begin{bmatrix} a_{j-1,-} \\ b_{j-1,-} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{H}}_{L/2} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{L/2} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{H}}_{L} \end{bmatrix} a_{j,-}.$$

Wir können also die DWT über zwei Skalen als Produkt von Matrizen berechnen. Dieses Muster können wir auf weitere Skalen fortsetzen und erhalten so

$$\begin{bmatrix} a_{j-l,-} \\ b_{j-l,-} \\ b_{j-l+1,-} \\ \vdots \\ b_{j-1,-} \end{bmatrix} = \Big(\prod_{i=0}^{l-1} A_i\Big) a_{j,-} \quad \text{mit} \quad A_i := \begin{bmatrix} \hat{\mathbf{H}}_{L/2^i} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{L-L/2^i} \end{bmatrix}$$

Somit ergibt sich eine Matrix-Formulierung der FWT, indem wir die Eingabekoeffizienten mit dem Produkt der Analysematrizen A_i der einzelnen Skalen multiplizieren. In Abbildung 2.2 ist dies für die Haar-Wavelets auf drei Skalen schematisch dargestellt.

Satz 2.3.1 (Laufzeit der FWT)**.** *Für eine Filterlänge N und eine Signallänge $L = 2^J$ hat die J-skalige schnelle Wavelet-Transformation eine Laufzeit in $\mathcal{O}(NL)$.*

Beweis. Die Analysematrizen A_i sind dünnbesetzte Matrizen. Der Aufwand der Matrixmultiplikation entspricht also asymptotisch der Anzahl der von Null verschiedenen Einträge der Analysematrizen.

In der Analysematrix der ersten Skala enthält jede Zeile höchstens N Einträge; dies ergibt insgesamt maximal NL Einträge. Bei der nächsten Skala müssen wir nur die obere Hälfte der Matrix zählen, da die Identität in der unteren Hälfte keine Operationen kostet, und haben somit $NL/2$ Einträge. Insgesamt ergibt sich für J Skalen

$$NL + \frac{1}{2}NL + \frac{1}{4}NL + \cdots + 2^{-(J-1)}NL < 2NL = \mathcal{O}(NL). \qquad \square$$

Bemerkung. Für konstante Filterlänge N ist die FWT mit einer linearen Laufzeit asymptotisch schneller als die Schnelle Fourier-Transformation (*englisch*: Fast Fourier Transform, FFT), die eine asymptotische Laufzeit von $\mathcal{O}(L \log L)$ hat [SN96].

2.4. Untersuchung mittels Zeit-Frequenz-Darstellungen

Wir wollen die Randbehandlung mit Gram-Schmidt-Randfiltern näher untersuchen und beispielhaft mit der symmetrischen Fortsetzung, einer in der Bildverarbeitung häufig verwendeten Signalfortsetzung, vergleichen. Dazu visualisieren wir für beide Randbehandlungsverfahren für verschiedene Wavelets die DWT zweier endlicher Testsignale. Für die Auswahl der Testsignale und die Visualisierungsmethodik orientieren wir uns an Jensen und la Cour-Harbo [Jl01].

2.4.1. Verwendete Testsignale

Das erste Testsignal $\mathbf{x}_0$ ist ein sogenannter *Chirp* (vom englischen (*to*) *chirp*, „zwitschern“). Damit bezeichnet man ein Signal, dessen Frequenz sich im Zeitverlauf verändert (vgl. Abbildung 2.3). Hier verwenden wir einen Chirp, dessen Frequenz linear mit dem Faktor 128 steigt. Wir erhalten die Funktionsvorschrift

$$f_0(t) := \sin(128\pi t^2), \tag{2.5}$$

die an den Punkten $\{0, 1, \dots, 2^{10}\}$ ausgewertet $\mathbf{x}_0$ ergibt.

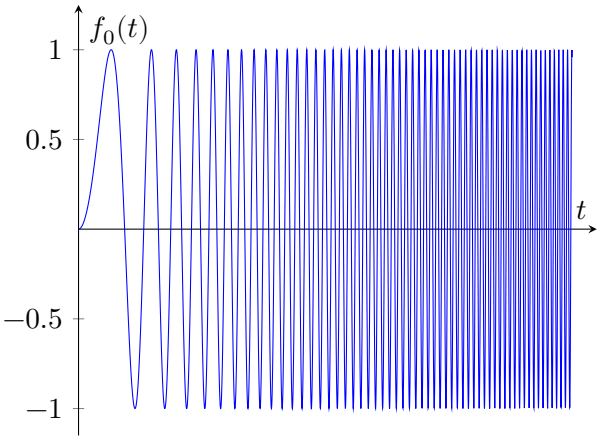


Abbildung 2.3.: Beispiel eines linearen Chirps

Das zweite Testsignal $\mathbf{x}_1$ besteht aus zwei Teilen. Der erste Teil soll sehr lokal im Frequenzbereich sein. Dazu wählen wir die Summe dreier Sinus-Wellen mit fester Frequenz

$$f_1(t) := \sin(\omega_0 t) + \sin(2\omega_0 t) + \sin(3\omega_0 t) \qquad \text{mit } \omega_0 := 405.5419,$$

die wir ebenfalls auf den Punkten $\{0, 1, \dots, 2^{10}\}$ auswerten. Der zweite Teil des Testsignals

soll im Zeitbereich sehr lokal sein. Dafür wählen wir das Signal

$$\mathbf{x}_1(n) = f_1(n) + \begin{cases} 25 & n = 299 \\ 1 & 499 \leq n < 700 \\ 15 & n = 900 \\ 0 & \text{sonst} \end{cases} \qquad \text{für } n \in \{0, 1, \ldots, 2^{10}\}.$$

Das zweite Testsignal $\mathbf{x}_1$ ergibt sich dann aus der Summe beider Teile. In Abbildung A.3 sind Spektogramme beider Testsignale dargestellt.

2.4.2. Zeit-Frequenz-Darstellungen mittels Diskreter Wavelet-Transformation

Für die Visualisierung nutzen wir sogenannte Zeit-Frequenz-Darstellungen. Damit bezeichnen wir eine Darstellung der Energie eines Signals in Abhängigkeit der Zeit und der Frequenz. Ein Beispiel dafür im Kontext der Fouriertransformation sind Spektogramme. Wir wollen im folgenden aus der Wavelet-Basis-Darstellung solche Zeit-Frequenz-Darstellungen konstruieren.

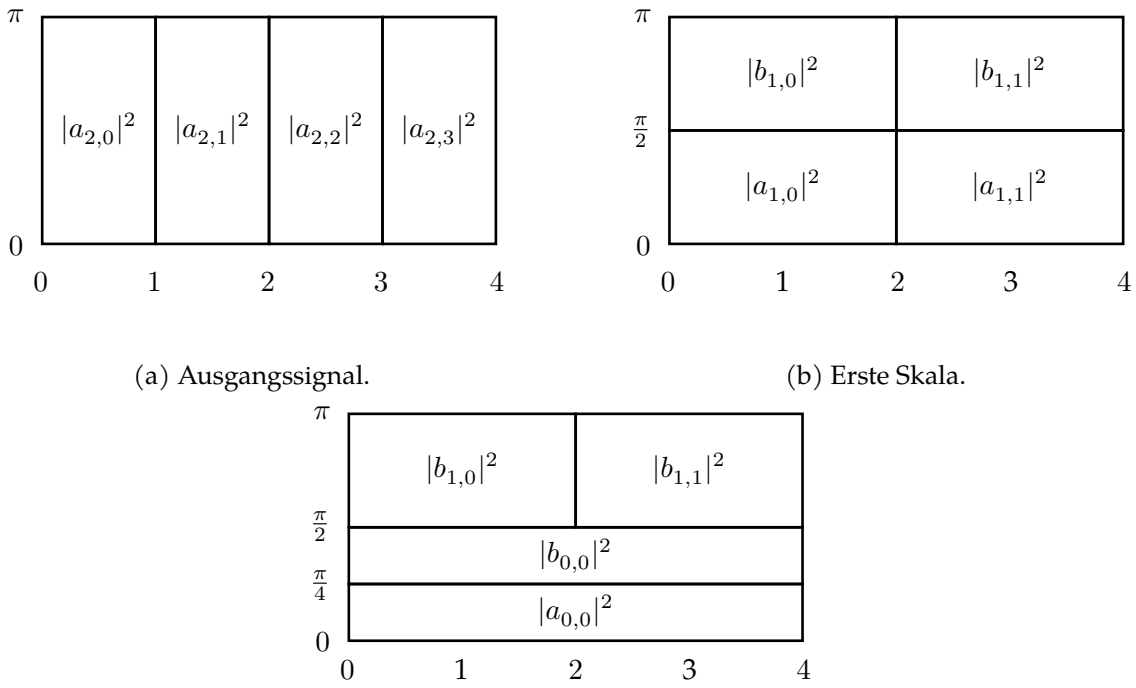
Dafür überlegen wir uns, dass die Energie-Anteile eines diskreten Signals sehr fein in der Zeitdimension aufgetrennt sind. Allerdings sind noch keine Frequenzinformation enthalten (vgl. Abbildung 2.4a). Wenden wir nun die DWT einmal an, teilen wir das Signal in einen hochfrequenten und einen niedrigfrequenten Anteil auf. Somit verdoppeln wir die Auflösung in der Frequenzdimension. Im Gegenzug halbiert sich die Länge der einzelnen Anteile und somit auch die Auflösung in der Zeitdimension (vgl. Abbildung 2.4b). Mit der nächsten Iteration der FWT geschieht dies erneut für den niedrigfrequenten Anteil: er wird in zwei Anteile aufgeteilt, womit wir die Frequenzauflösung im niedrigfrequenten Bereich verdoppeln und die Zeitauflösung halbieren (vgl. Abbildung 2.4c).

Dies führen wir für alle weiteren Skalen der DWT fort. Die so resultierende Kachelung bildet die Zeit-Frequenz-Darstellung der DWT des Signals. Wie in Abbildung 2.4 dargestellt, entspricht jede Kachel einem Koeffizienten der DWT des Signals. Die Energieanteile, die auf eine Kachel entfallen, sind dann das Betragsquadrat des Koeffizienten. Grafisch können wir dies darstellen, indem wir die Kacheln entsprechend ihrem Energieanteil einfärben. Wir wählen eine Einfärbung in Graustufen, wobei dunklere Farben höheren Energieanteilen entsprechen. Zur besseren Visualisierung normalisieren wir den Energieanteil $|s|^2$ logarithmisch durch

$$\log_e(|s|^2 + c_0).$$

Wir orientieren uns an Jensen und la Cour-Harbo [Jl01] und wählen für die Darstellung $c_0 = 1$. In Abbildung A.4a sind die Zeit-Frequenz-Darstellungen der DWT der in Abschnitt 2.4.1 konstruierten Testsignale abgebildet.

Bemerkung. Wir sind in der Beschreibung der Zeit-Frequenz-Darstellungen implizit davon ausgegangen, dass sich das Signal bei der Anwendung der Filterbank nicht verlängert. Dies ist für eine Randbehandlung mit Randfiltern per Konstruktion stets erfüllt. Verwenden wir jedoch andere Randbehandlungsverfahren, verlängert sich das Signal durch die wiederholte Anwendung der Filterbank. Diese Koeffizienten fügen wir als weitere Kacheln am linken und rechten Rand der Darstellung an.



(a) Ausgangssignal.

(b) Erste Skala.

(c) Zweite Skala.

Abbildung 2.4.: Zeit-Frequenz-Darstellungen der verschiedenen Skalen während der FWT eines Signals der Länge 4 [nach Jl01]. Auf der horizontalen Achse verläuft die Zeitdimension; auf der vertikalen die Frequenzdimension. Die einzelnen Felder entsprechen der Energie eines einzelnen Koeffizienten.

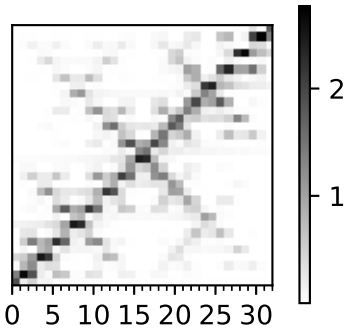
2.4.3. Zeit-Frequenz-Darstellungen mittels Wavelet-Paket-Transformation

Die bisher beschriebene Art der Zeit-Frequenz-Darstellungen hat den Nachteil, dass die Frequenz- bzw. Zeitauflösung zwischen den Frequenzbändern variiert. Die Frequenzauflösung steigt mit fallender Frequenz, wohingegen die Zeitauflösung sinkt. Aufgrund der heisenbergschen Unschärferelation [SN96, S. 67] ist es nicht möglich, beide Dimensionen gleichzeitig fein aufzulösen, aber wir können die Auflösung beider Dimensionen im ganzen Bildbereich konstant halten. Anschaulich gesprochen sollen die entstehenden Kacheln nicht mehr mit fallender Frequenz immer flacher und länger werden, sondern im ganzen Bildbereich die gleiche Form bewahren.

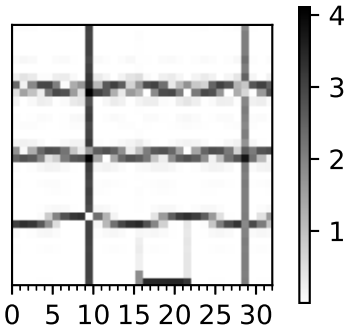
Dies erreichen wir, indem wir statt der DWT die Wavelet-Paket-Transformation berechnen (vgl. Abschnitt 1.3.3). In jeder Skala wenden wir die Filterbank auf alle Koeffizienten der vorherigen Skala an. Für alle Koeffizienten verdoppelt sich die Frequenzauflösung und halbiert sich die Zeitauflösung. Somit verändert sich die Form aller Kacheln und nicht, wie bei der DWT, nur die der Kacheln der untersten Zeile. Wählen wir für eine Ausgangssignallänge $L = 2^{2J}$ die Skalenanzahl als J, so erhalten wir eine regelmäßige $J \times J$-Kachelung; die Auflösung ist also in beiden Dimensionen gleich.

Bemerkung. Im Gegensatz zur DWT müssen wir bei der Wavelet-Paket-Transformation

darauf achten, in welcher Reihenfolge wir die Koeffizienten entlang der Frequenzachse anordnen. Bei der DWT konnten wir die aus dem Verfahren entstehende natürliche Anordnung wählen. Dies ist jedoch bei der Wavelet-Paket-Transformation nicht möglich, da es bei der Anwendung der Filterbank auf die hochfrequenten Anteile der vorherigen Skala zu Spiegelungen des Signals im Frequenzbereich kommt. Dies lässt sich mit einer Umordnung entlang der Frequenzachse korrigieren, der sogenannten Frequenzordnung. Diese Umordnung ist jedoch ausschließlich bei Visualisierungen notwendig. Für Details sei auf Jensen und la Cour-Harbo [Jl01, Abschnitt 9.3] verwiesen.



(a) Zeit-Frequenz-Darstellung des ersten Testsignals aus Abschnitt 2.4.1 (linearer Chirp).



(b) Zeit-Frequenz-Darstellung des zweiten Testsignals aus Abschnitt 2.4.1 (lokal in Zeit und Frequenz).

Abbildung 2.5.: Fünfskalige Zeit-Frequenz-Darstellungen mittels Wavelet-Paket-Transformation mit Haar-Wavelets.

2.4.4. Untersuchung von Testsignalen

Wir wollen die Randbehandlung mittels Gram-Schmidt-Randfiltern mit der Randbehandlung durch symmetrische Fortsetzung vergleichen. Dazu betrachten wir für beide Randbehandlungsverfahren die Zeit-Frequenz-Darstellungen der fünfskaligen Wavelet-Paket-Transformation der in Abschnitt 2.4.1 konstruierten Testsignale. Diese Visualisierung der Testsignale wird auch von Jensen und la Cour-Harbo [Jl01, Abschnitt 9.3] diskutiert.

Zur Berechnung der Randbehandlung mittels Gram-Schmidt-Randfiltern nutzen wir die in Abschnitt 3.1 besprochene Python-Implementierung, für die symmetrische Fortsetzung die Python-Bibliothek *PyWavelets* [Lee+19].

In Abbildungen 2.5a und 2.5b ist die Zeit-Frequenz-Darstellung mittels Wavelet-Paket-Transformation für das Haar-Wavelet abgebildet. Wir sehen klar die zu erwarteten Charakteristika der Testsignale, die sich bereits in Spektogrammen in Abbildung A.3 gezeigt hatten: Beim linearen Chirp erkennen wir, dass die Signalenergie mit zunehmender Zeit linear in der Frequenz steigt. Gleichzeitig erkennen wir gegenläufig zu diesem linearen Trend laufende Artefakte. Diese stammen daher, dass die Filter der Haar-Wavelets nicht ideal sind [Jl01, S. 112f.].

Im zweiten Testsignal erkennen wir sehr scharf die in Zeit bzw. Frequenz lokalen Anteile. Die einzelnen von Null verschiedenen Einträge bei $n = 299$ und 900 sehen wir als Energieanteile in allen Frequenzen. Dies ist zu erwarten, da die Frequenzantwort des Einheitssignals konstant gleich 1 ist. Auch den konstanten Abschnitt zwischen 499 und 700 erkennen wir an der Energie im untersten Frequenzband in diesem Abschnitt. An den Rändern des konstanten Abschnitts sehen wir aufgrund der Sprungstellen Artefakte in höheren Frequenzbereichen.

Auch wenn diese Visualisierung mittels der Haar-Wavelets hilfreich für das Verständnis der Testsignale und der zu erwartenden Effekte ist, ist sie nicht dazu geeignet, Randbehandlungen zu vergleichen. Dies liegt schlicht daran, dass aufgrund der kurzen Filterlänge bei Haar-Wavelets keine Randbehandlung notwendig ist. Daher wollen wir uns im weiteren Verlauf längeren Wavelets zuwenden. Unsere Wahl im Rahmen dieser Arbeit sind Wavelets der Daubechies- und Symlet-Familie, die wir beide in Abschnitt 1.3.4 beschrieben haben. Wir erinnern uns, dass beide Familien einer ähnlichen Konstruktion entstammen und in dieser Hinsicht Symlets „weniger asymmetrische“ Daubechies-Wavelets sind.

In Abbildungen 2.6 und 2.7 sehen wir für die Randbehandlung durch Gram-Schmidt-Randfilter bzw. symmetrische Fortsetzung jeweils die Zeit-Frequenz-Darstellungen beider Testsignale mittels fünfskaliger Wavelet-Paket-Transformation für Wavelets der Daubechies- und der Symlet-Familie mit den Längen 8, 16 und 24. Für die aufgrund symmetrischer Fortsetzung verlängerten Signale ist der ursprüngliche Signalbereich durch blaue Linien markiert. Zur besseren Darstellung wurden auch bei der Darstellung mit Randfiltern entsprechend Kacheln an den Rändern ergänzt; diese enthalten aber keine Energieanteile.

Wir betrachten die Darstellungen des ersten Testsignals. Zunächst stellen wir fest, dass im Vergleich zu den Haar-Wavelets mit steigender Filterlänge deutlich weniger gegenläufige Artefakte auftreten. Das liegt daran, dass mit steigender Länge die Daubechies- und Symlet-Filter sich idealen Filter annähern [Jl01, S. 113]. Wir bemerken außerdem, dass bei den Daubechies-Wavelets mit steigender Länge die Energieanteile deutlich breiter um den linearen Trend streuen. Grund hierfür ist die starke Asymmetrie der Daubechies-Filter [Jl01, Abschnitt 9.4.3]. Folgerichtig sehen wir diesen Effekt bei den deutlich weniger asymmetrischen Symlets kaum. Insgesamt kommt es beim ersten Testsignal zu wenig Randartefakten. Jedoch führt der Asymmetrie-Effekt längerer Daubechies-Wavelets dazu, dass die Energieanteile des oberen und unteren Frequenzbereichs sich außerhalb des ursprünglichen Signalbereichs verlagern.

In den Darstellungen des zweiten Testsignals erkennen wir den Asymmetrie-Effekt wieder. So streuen die zeitlokalen Anteile bei $n = 299$ und 900 für längere Daubechies-Wavelets so stark, dass diese für Daubechies-Wavelets der Länge 12 kaum ausmachbar sind.[5] Auch die Energieanteile des konstanten Signalanteils von $n = 499$ bis 700 rücken mit steigender Länge der Daubechies-Wavelets näher an den Rand. Allerdings können wir ebenfalls feststellen, dass sich diese Streuung nur auf die Zeit-Dimension auswirkt; die in der Frequenz lokalen Anteile streuen nicht vermehrt. Für Symlets sind diese Effekte aufgrund der höheren Symmetrie kaum zu bemerken.

Wir erkennen bei den Darstellungen des zweiten Testsignals große Unterschiede an

[5] Es gibt Methoden, diese Streuung für die bessere visuelle Interpretierbarkeit explizit zu korrigieren [vgl. Jl01, Abschnitt 9.4.3].

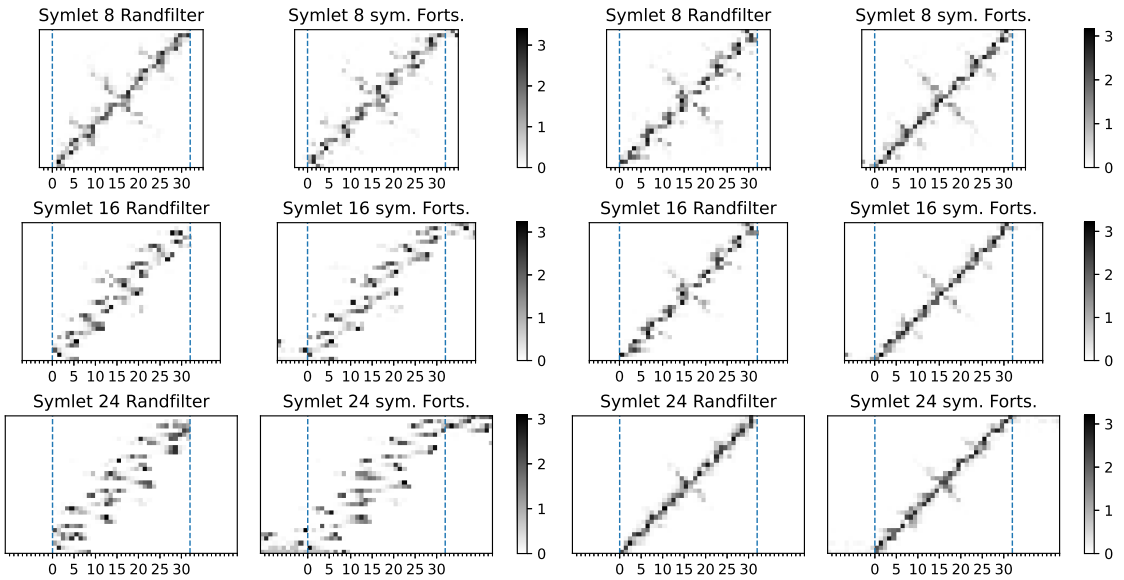


Abbildung 2.6.: Zeit-Frequenz-Darstellungen des ersten Testsignals (linearer Chirp) für verschiedene Wavelets. Gegenübergestellt sind jeweils zwei fünfskalige Wavelet-Packet-Zerlegungen: mit Gram-Schhmidt-Randfiltern und mit symmetrischer Signalfortsetzung in der PyWavelets-Implementation [Lee+19]. Die blauen Linien geben den Zeitbereich des Eingabesignals an.

den Rändern. Bei der symmetrischen Fortsetzung sind die Energieanteile außerhalb des ursprünglichen Signalbereichs sehr ausgeprägt. Für die Symlets sind die Energieanteile dort eher diffus verteilt, wohingegen wir bei den Daubechies-Wavelets klar die Fortsetzung der in der Frequenz lokalen Anteile ausmachen können. Dem gegenüber entstehen bei der Randbehandlung durch Gram-Schmidt-Randfilter deutlich weniger Randartefakte. So kommt es bei den Daubechies-Wavelets zu einer Verringerung der Energieanteile am linken Signalrand. Bei den Symlets stellen wir eine leichte Streuung in der Frequenzdimension am linken Rand fest.

Insgesamt ist zu beobachten, dass Symlets aufgrund der höheren Symmetrie besser die Zeitposition der Signalteile erhalten. Andererseits konnten wir für die Testsignale eine klare Reduktion der Randartefakte durch Gram-Schmidt-Randfilter ermitteln.

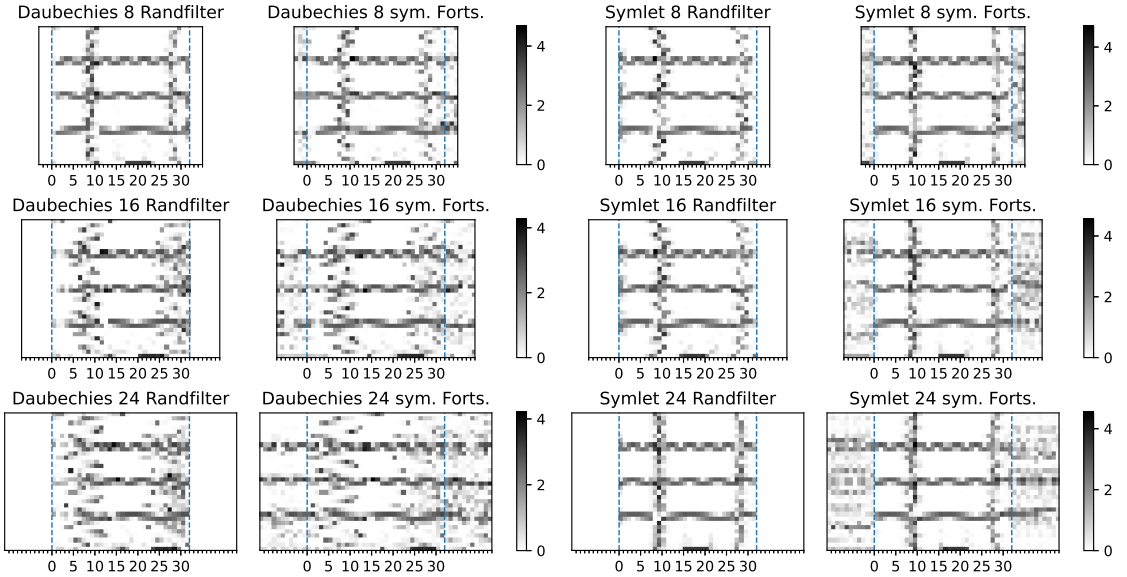


Abbildung 2.7.: Zeit-Frequenz-Darstellungen des zweiten Testsignals (lokal in Zeit und Frequenz) für verschiedene Wavelets. Gegenübergestellt sind jeweils zwei fünfskalige Wavelet-Packet-Zerlegungen: Mit Gram-Schmidt-Randfiltern und mit symmetrischer Signalfortsetzung in der PyWavelets-Implementation [Lee+19]. Die blauen Linien geben den Zeitbereich des Eingabesignals an.

3. Numerische Experimente

In diesem Kapitel beschäftigen wir uns mit der Anwendung der Gram-Schmidt-Randfilter zur Erkennung von DeepFake-Bildern (vgl. Abschnitt 3.3). Dazu besprechen wir in Abschnitt 3.1 zunächst die Implementation der Gram-Schmidt-Randfilter in Python. Um diese überhaupt auf Bilder, also zweidimensionale Signale endlicher Länge, anwenden zu können, führen wir in Abschnitt 3.2 ein Verfahren ein, mit dem wir die eindimensionalen Wavelet-Transformation entlang jeder Achse separat anwenden können. Anschließend verwenden wir dies, um Visualisierungen der 2D-Wavelet-Paket-Transformation zu untersuchen.

In Abschnitt 3.3 erweitern wir einen Ansatz von Wolter u. a. [Wol+21] zur Erkennung von DeepFakes. Dieser verwendet Haar-Wavelets, um Randartefakte zu vermeiden. Wir tun dies, indem wir anstatt Haar-Wavelets längere Wavelets der Daubechies- und Symlet-Familie mit Randbehandlung durch Gram-Schmidt-Randfilter einsetzen. Dies ermöglicht einerseits die Verwendung längerer Wavelets mit deutlich reduzierten Randartefakten (vgl. Abschnitt 3.2). Andererseits müssen wir das Faltungsnetzwerklayout nicht in Abhängigkeit der Wavelet-Länge anpassen, da die Signallänge durch die Randfilter beibehalten wird.

3.1. Implementierung

In dieser Arbeit verwenden wir für die Berechnung von Wavelet-Tranformationen mit Gram-Schmidt-Randfiltern auf Basis der Matrix-Darstellung der Transformationen eine eigene Implementation in der Programmiersprache *Python*.[1] Die Matrizen werden dabei als dünnbesetzte Tensoren der *PyTorch*-Bibliothek [Pas+19] umgesetzt. Dies ermöglicht einerseits eine Beschleunigung der Berechnungen durch Nutzung von Grafikprozessoren (*englisch*: Graphical Processing Units, GPUs) und andererseits den Einsatz automatischen Differenzierens zum Trainieren von Faltungsnetzwerken (*englisch*: Convolutional Neural Networks, CNNs).

Die Implementierung baut auf den Python-Bibliotheken *Pytorch Wavelet Toolbox* [Wol21] und *PyWavelets* [Lee+19] auf. Die Implementation der Randfilterberechnung basiert auf Jensen und la Cour-Harbo [Jl01].

3.2. Separable 2D-Wavelet-Transformationen

Die in Kapitel 1 und 2 besprochenen Wavelet-Transformationen agieren nur auf eindimensionalen (endlichen) Signalen. Wir wollen diese Transformationen nun auf einfache Art

[1] Dem Autor ist keine andere öffentlich verfügbare Implementation von (Gram-Schmidt-)Randfiltern in Python bekannt.

auf zweidimensionale Signale erweitern. Dazu wenden wir entlang beider Achsen des Signals separat eine eindimensionale Wavelet-Transformation an, weshalb diese Art der zweidimensionalen Wavelet-Transformation auch *separabel* genannt wird. Für die Konstruktion orientieren wir uns an Jensen und la Cour-Harbo [Jl01, Abschnitt 6.1]. Im folgenden sprechen wir allgemein von Wavelet-Transformation, da alle Konstruktionsschritte gleichermaßen auf die FWT und die Wavelet-Paket-Transformation anwendbar sind.

Wir betrachten ein endliches zweidimensionales Signal $\mathbf{X} \in \mathbb{R}^{M \times N}$ mit M Zeilen und N Spalten. Zunächst wollen wir eine eindimensionale Wavelet-Transformation entlang der Spalten von $\mathbf{X}$ durchführen. Wir haben gesehen, dass wir mithilfe von Gram-Schmidt-Randfiltern die Wavelet-Transformation als quadratische Matrix darstellen können. Für die Transformation der Spalten erhalten wir somit die $M \times M$-Matrix $\mathbf{W}_s$. Für jede Spalte ergibt sich durch Multiplikation mit $\mathbf{W}_s$ die Wavelet-Transformation der Spalte. Somit ist das Matrix-Produkt

$$\mathbf{Y}_s := \mathbf{W}_s \mathbf{X}$$

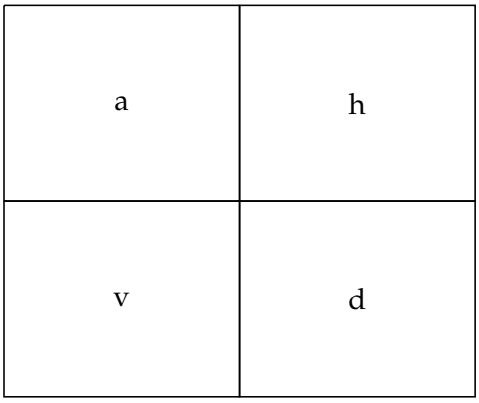
eine Matrix, deren Spalten die Wavelet-Transformation der Spalten von $\mathbf{X}$ sind. Nun wollen wir die Wavelet-Transformation entlang der Zeilen von $\mathbf{Y}_s$ durchführen. Dies ist äquivalent zur Wavelet-Transformation entlang der Spalten von $\mathbf{Y}_s^T$. Sei $\mathbf{W}_z$ die $N \times N$-Matrix der Zeilen-Wavelet-Transformation. Dann ist

$$\mathbf{Y} := (\mathbf{W}_z \mathbf{Y}_s^T)^T = \mathbf{W}_s \mathbf{X} \mathbf{W}_z^T$$

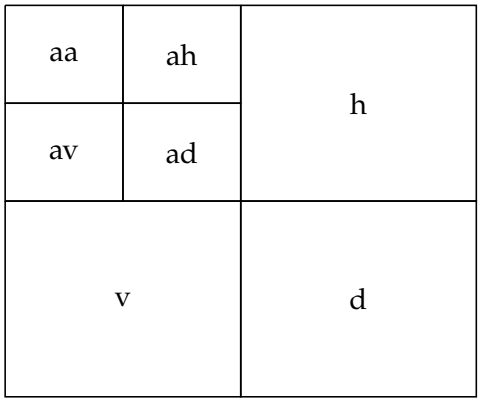
das Ergebnis der separablen 2D-Wavelet-Transformation. Aufgrund der Assoziativität der Matrix-Multiplikation spielt es keine Rolle, ob wir bei der Herleitung zunächst entlang der Spalten oder der Zeilen transformieren.

Wir konnten die Anzahl an Skalen und die verwendeten Wavelets separat pro Achse wählen. Somit ist es denkbar, dass wir entlang der Achsen verschiedene Wavelet-Transformationen berechnen. In dieser Arbeit beschränken wir uns jedoch auf den Fall, dass alle Parameter für beide Achsen identisch sind, eingeschlossen der Signallänge. Es gilt also stets $\mathbf{W}_z = \mathbf{W}_c$.

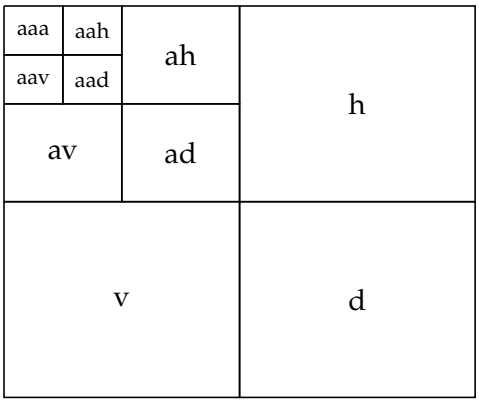
Bei der 2D-FWT wird das Signal entlang jeder Achse in einen hoch- und einen niedrigfrequenten Teil aufgeteilt; insgesamt ergeben sich also vier Teile. Der in beiden Achsen niedrigfrequente Teil heißt Approximationskoeffizient und wird mit „a“ notiert. Die anderen Teile entsprechen Richtungen von Kanten im Bild, die in diesem Teil hervorgehoben werden: ist der Teil einer Achse hoch- und der andere niedrigfrequent, werden horizontale („h“) bzw. vertikale („v“) Kanten im Bild hervorgehoben. Sind die Teile beider Achsen hochfrequent, entspricht dies diagonalen Kanten („d“). In Abbildung 3.1a ist dies schematisch dargestellt. Die zweite Skala der 2D-FWT erhalten wir, indem wir dies auf dem Approximationskoeffizienten der vorherigen Skala wiederholen (vgl. Abbildungen 3.1b und 3.1c). Wir notieren die so erhaltenen Koeffizienten, indem wir den Buchstaben der aktuellen Skala („a“, „d“ etc.) an die Buchstaben der vorherigen Skala anhängen. Wiederholen wir diese Zerteilung auf allen Koeffizienten der vorherigen Skala, entspricht dies einer 2D-Wavelet-Packet-Transformation (vgl. Abbildung 3.1d).



(a) Erste Skala einer 2D-FWT



(b) Zweite Skala einer 2D-FWT



(c) Dritte Skala einer 2D-FWT

aaa	aah	aha	ahh	haa	hah	hha	hhh
aav	aad	ahv	ahd	hav	had	hhv	hhd
ava	avh	ada	adh	hva	hvh	hda	hdh
avv	avd	adv	add	hvv	hvd	hdv	hdd
vaa	vah	vha	vhh	daa	dah	dha	dhh
vav	vad	vhv	vhd	dav	dad	dhv	dhd
vva	vvh	vda	vdh	dva	dvh	dda	ddh
vvv	vvd	vdv	vdd	dvv	dvd	ddv	ddd

(d) Dreiskalige 2D-Wavelet-Paket-Transformation in Frequenzordnung.

Abbildung 3.1.: Schemata zweidimensionaler Wavelet-Transformationen.

3.3. DeepFake-Erkennung

DeepFakes sind mit Methoden des DeepLearnings generierte, täuschend realistische Medieninhalte (hier Bilder). Die dafür verwendeten Architekturen werden als Generative Adversarial Networks (GANs) [Goo+14] bezeichnet, übersetzt etwa „erzeugende gegnerische Netzwerke“. Sie bestehen aus zwei Faltungsnetzwerken, die in einem Nullsummenspiel gegeneinander antreten. Eines der Netzwerke generiert Bildkandidaten (der „Generator“), die echten Bilddaten täuschend ähnlich sein sollen. Das andere Netzwerk versucht diese Kandidaten von echten Bilddaten zu unterscheiden (der „Diskriminator“). Der Generator wird dann darauf trainiert, Bildkandidaten zu erzeugen, die den Diskriminator täuschen.

Die Ergebnisse der GANs sind so überzeugend, dass Menschen erhebliche Schwierigkeiten haben, sie als synthetisch zu erkennen [Kar+18]. Als Demonstration sei die Interseite `whichfaceisreal.com` [WB19] erwähnt, auf der Nutzer:innen ein Bild aus dem Flickr Faces High Quality Datensatz [KLA19] und ein mit StyleGAN [KLA19] generiertes Bild angezeigt wird. Bei der Aufgabe, das richtige Bild auszuwählen, erzielen Nutzer:innen

selbst mit Übung nur eine Genauigkeit von 75 % [Sim19].

DeepFakes stellen zunehmend ein gesellschaftliches Problem dar, da sie unter anderem dazu verwendet werden können, Falschinformationen zu streuen und den öffentlichen Diskurs oder Wahlen zu beeinflussen [Wes19]. Daher ist es wichtig, robuste Verfahren zur Erkennung von DeepFakes zu entwickeln.

Wolter u. a. [Wol+21] schlagen dazu vor, einen Klassifikator auf den Wavelet-Paket-koeffizienten der Bilder zu trainieren. Da sie auf einem Datensatz an den Rändern im Frequenzbereich große Unterschiede zwischen den echten und GAN-generierten Daten feststellen und daher Randeffekte vermeiden wollen, verwenden sie Haar-Wavelets. Wir wollen diesen Ansatz mittels Gram-Schmidt-Randfilter auf längere Wavelets ausweiten. Da durch Gram-Schmidt-Randfilter die Signale unabhängig der Wavelet-Länge ihre Länge beibehalten, können wir für alle Wavelet-Längen die von Wolter u. a. [Wol+21] vorgeschlagene Architektur übernehmen.

Zunächst wollen wir jedoch ein besseres Verständnis dafür entwickeln, worin die Unterschiede zwischen mittels GANs generierten und echten Bildern liegen können. Dazu untersuchen wir in Abschnitt 3.3.1 die Visualisierung von 2D-Wavelet-Paketen.

Anschließend betrachten wir die Problemstellung, auf einem aus Orginalbildern und DeepFakes verschiedener GANs bestehenden Datensatz die Bildquelle zu klassifizieren. Unsere Ergebnisse werden wir neben Wolter u. a. [Wol+21] auch mit Frank u. a. [Fra+20] vergleichen, die für die gleiche Problemstellung einen auf der Diskreten Cosinus-Transformation (DCT) basierten Ansatz vorschlagen. In Abschnitt 3.3.4 werden wir dann untersuchen, wie gut Bilder eines GANs, das nicht in den Trainingsdaten repräsentiert ist, als DeepFakes erkannt werden.

3.3.1. Untersuchung des FFHQ-Datensatzes

Zunächst wollen wir ein Verständnis dafür entwickeln, wo der Unterschied zwischen Orginalbildern und DeepFakes liegen kann. Dazu betrachten wir 5.000 zufällig ausgewählte Bilder des Flickr Faces High Quality-Datensatzes [KLA19], der aus Portraitfotos echter Menschen besteht. Hierauf generieren wir mit StyleGAN-Architektur [KLA19] 5.000 DeepFakes und vergleichen diese mit den Orginalbildern. Dies machen wir ähnlich wie Wolter u. a. [Wol+21], die den FFHQ-Datensatz für log-skalierte Haar-Wavelets betrachtet haben.

Wir haben im vorherigen Abschnitt gesehen, wie wir Wavelet-Transformationen auf ein zweidimensionales Signal anwenden können. Dies nutzen wir, indem wir mithilfe der dreiskaligen separablen 2D-Wavelet-Paket-Transformation mittels Gram-Schmidt-Randfiltern die Bilder kanalweise transformieren. Anschließend bilden wir pro Datenquelle den Mittelwert und die Standardabweichung über alle Bilder und die Farbkanäle. Zudem berechnen wir für den Mittelwert und die Standardabweichung jeweils die absolute Differenz beider Datenquellen. In Abbildungen A.7 und A.8 sehen wir die Ergebnisse für das Daubechies-Wavelet sowie das Symlet der Länge 24 in Frequenzordnung.

Für beide Wavelets sind in der Differenz der absoluten Koeffizienten deutlich schemenhafte Gesichtszüge zu erkennen. Dies lässt darauf schließen, dass die GAN-generierten Bilder strukturelle Fehler im Gesichtsbereich machen. Zudem sehen wir, dass die Paketenergie der absoluten Differenz der mittleren Signale mit zunehmender Frequenz ansteigt.

Dies liegt daran, dass bei den GANs die mittlere Energie im hochfrequenten Bereich höher ist.

Weiter erkennen wir sowohl bei den Orginalbildern als auch bei den DeepFakes in der Standardabweichung Gesichtsumrandungen, die jedoch auf den Orginaldaten stärker sind. Hier können wir vermuten, dass die Varianz im Hintergrund bei orginalen Bildern höher als bei synthetischen ist.

Auch bei der Differenz der mittleren Koeffizienten sehen wir, dass die Ränder um die Gesichtsschemen ausgeprägt sind. Hier – also im Bildhintergrund – liegen somit im Durchschnitt Unterschiede vor.

Betrachten wir die Bilder hinsichtlich der Randbehandlung, so erkennen wir wenig Randartefakte. Wir erkennen an manchen Paketen tieferer Frequenzen Linien an den Rändern, die auf beiden Datenquellen in gleicher Form auftreten, jedoch beim Daubechies-Wavelets breiter sind als beim Symlet. Ähnlich wie in Abschnitt 2.4 können wir daher darauf schließen, dass diese Streifen asymmetriebedingt sind. Weiter erkennen wir in den Ecken der Pakete punktartige Artefakte. Diese treten ausgesprochen symmetrisch auf und zeigen die gleiche Breite wie die bereits bemerkten Streifen, weshalb wir hier vermuten können, dass dies ebenfalls eine Folge der Asymmetrien in Verbindung mit der separablen Transformation sind.

3.3.2. Datensätze und Parameterwahl

Um Vergleichbarkeit zu gewährleisten, stellen wir den Versuchsaufbau von Wolter u. a. [Wol+21] exakt nach. Lediglich bei der Berechnung der Wavelet-Paket-Transformation werden wir abweichend die in Abschnitt 3.1 beschriebene Implementation der Wavelet-Paket-Transformation mit Gram-Schmidt-Randfiltern verwenden. Wir haben auch mit Frank u. a. [Fra+20] eine hohe Vergleichbarkeit der Ergebnisse, da sich deren Versuchsaufbau von Wolter u. a. [Wol+21] nicht wesentlich unterscheidet.

Wir führen alle Ergebnisse dieses Abschnittes auf zwei Datensätzen aus Farbbildern durch: Large-scale Scene UNderstanding bedrooms (LSUN) [Yu+15] und Large-scale Celeb Faces Attributes (CelebA) [Liu+15]. Auf den Bildern des LSUN-Datensatzes sind Schlafzimmer abgebildet. Der CelebA-Datensatz enthält Bilder von Gesichtern prominenter Personen, die nach der Augenposition ausgerichtet wurden. In Abbildung A.5 ist pro Datensatz beispielhaft ein Element abgebildet.

Wir wählen aus beiden Datensätzen zufällig 150.000 Bilder aus, schneiden diese zu und skalieren sie jeweils auf eine Größe von 128×128 Pixeln. Für beide Datensätze erzeugen wir mit vortrainierten Modellen von Yu, Davis und Fritz [YDF19] ebenfalls in einer Größe von 128×128 Pixeln 150.000 Bilder mit den GAN-Architekturen CramerGAN [Bel+17], MMDGAN [Bin+18], ProGAN [Kar+18] und SN-DCGAN [Miy+18]. Pro Datensatz ist in Abbildung A.6 exemplarisch ein mit ProGAN generiertes Bild dargestellt.

Pro Datensatz teilen wir die insgesamt 750.000 Bilder zufällig in einen Trainingsdatensatz aus 500.000 Bildern, einen Validierungsdatensatz aus 100.000 Bildern sowie einen Test-Datensatz aus 150.000 Bildern auf, wobei wir jeweils aus jeder Quelle gleich viele Bilder wählen. Wir berechnen anschließend für alle Bilder pro Farbkanal die dreiskalige separable 2D-Wavelet-Paket-Transformation mit Wavelets der Daubechies- und Symlet-Familien der Längen 8, 16 und 24. Dabei verwenden wir zur Randbehandlung Gram-Schmidt-Randfilter

ohne Optimierungen. Außerdem erstellen wir eine Variante des Datensatzes, indem wir die Wavelet-Koeffzienten $\log_e$-skalieren.

Es soll die Bildquelle klassifiziert werden. Unterschieden werden also die Orginalbilder und vier GAN-Architekturen, was fünf Klassen ergibt. Hierzu trainieren wir auf allen Datensätzen eine lineare Regression und ein Faltungsnetzwerk. Dabei nutzen die von Wolter u. a. [Wol+21] vorgeschlagene Architektur, vgl. Tabelle A.1 für den genauen Aufbau der Architekturen. Vor dem Training normalisieren wir die Daten, indem wir je Farbkanal den Mittelwert der Trainingsdaten abziehen und durch die Standardabweichung der Trainingsdaten teilen. Zum Training der Modelle nutzen wir den Adam-Optimierer [KB15] mit einer Batchgröße von 512 und einer Lernrate von 0.001. Die lineare Regression trainieren wir auf dem LSUN-Datensatz zehn und auf dem CelebA-Datensatz acht Epochen; das Faltungsnetzwerk trainieren wir jeweils 50 Epochen lang. Im Sinne der Reproduzier- und Vergleichbarkeit trainieren wir alle Modelle fünfmal für die fixierten Zufallszahlenseeds 0 bis 4 und geben stets den Mittelwert und die Standardabweichung an.

3.3.3. Ergebnisse der Bildquellen-Klassifikation

Die Genauigkeit der Klassifikation der Bildquelle auf den Testdatensätzen ist in Tabelle 3.1 für die lineare Regression und in Tabelle 3.2 für die CNNs dargestellt. Die linke Spalte enthält jeweils die Ergebnisse für den CelebA-Datensatz und die rechte für den LSUN-Datensatz. Angegeben ist jeweils die maximale und mittlere Genauigkeit aus fünf Durchläufen sowie die Standardabweichung. Neben den Ergebnissen für Daubechies-Wavelets und Symlets der Längen 8, 16 und 24 sind zum Vergleich die Ergebnisse von Wolter u. a. [Wol+21] für das Haar-Wavelet sowie die unverarbeiteten Bilddaten angegeben. Für die CNNs sind ebenfalls die DCT-basierten Ergebnisse von Frank u. a. [Fra+20] dargestellt.

Auf unskalierten Wavelet-Paketkoeffizienten ist beim Daubechies-Wavelet der Länge 8 auf dem CelebA-Datensatz und dem Symlet der Länge 24 auf dem LSUN-Datensatz jeweils ein Ausreißer aufgetreten. Beide Male ist die Regression auf den Trainingsdaten konvergiert, aber hat nicht auf die Testdaten generalisiert. Beim Daubechies-Wavelet war die Testgenauigkeit des Ausreißers bei 78,72 %; beim Symlet 38,64 %. Dies schlägt sich in einer hohen Standardabweichung und geringeren mittleren Genauigkeit nieder. Vernachlässigt man die Ausreißer, erhält man über vier Durchläufe beim Daubechies-Wavelet eine Testgenauigkeit von $99{,}22 \pm 0{,}10$ % und beim Symlet von $83{,}54 \pm 2{,}74$ %.

Wir sehen insbesondere bei der Regression, aber auch beim CNN, auf den log-skalierten Daten erhebliche Verbesserungen gegenüber den log-skalierten Haar-Wavelets. Bei der Regression ist das beste mittlere Ergebnis auf CelebA 4,46 Prozentpunkte besser als das Haar-Wavelet; auf LSUN beträgt die Verbesserung 6,37 Prozentpunkte. Die einzige Konstellation, in der die log-skalierten Wavelets nicht besser als das log-skalierte Haar-Wavelet sind, ist das CNN mit dem Daubechies-Wavelet der Länge 24 auf LSUN. Die Verbesserung ist auf dem CelebA-Datensatz so ausgeprägt, dass das mittlere Ergebnis der linearen Regression der log-skalierten Symlets der Länge 16 sogar besser ist, als alle zitierten Vergleichswerte auf CelebA – die nicht-linearen Faltungsnetze eingeschlossen. Mit der CNN-Architektur verbessert sich dies noch im Maximum auf 99,68 %. Auch auf dem LSUN-Datensatz ist die lineare Regression auf dem besten log-skalierten Wavelet im Mittel

besser als die maximalen Vergleichswerte bis auf den DCT-basierten Ansatz. Auch mit der CNN-Architektur ist der DCT-basierte Ansatz noch besser, wobei die log-skalierten Wavelets im Gegensatz zum Haar-Wavelet kompetetive Ergebnisse erzielen.

Diese Verbesserung spiegelt sich bei den unskalierten Wavelet-Koeffizienten nicht wider. Hier gibt es geringfügige Verbesserungen gegenüber dem Haar-Wavelet, aber auch signifikante Verschlechterungen.

Für steigende Filterlänge erkennen wir für die Daubechies-Wavelets auf den log-skalierten Daten einen Abwärtstrend. Dies macht für das CNN auf LSUN zwischen den Längen 8 und 24 im Mittel sogar 3,25 Prozentpunkte aus. Für die Symlets können wir keinen Trend erkennen. Dies könnte auf die asymmetriebedingten Verschiebungseffekte zurückzuführen sein, die wir in Abschnitt 3.2 für die Daubechies-Wavelets gesehen hatten.

3.3.4. Ergebnisse der Erkennung eines unbekannten Generators

In der Praxis sind beim Training der Modelle nicht alle möglichen GAN-Architekturen bekannt oder es liegen Trainingsdaten nicht in ausreichender Anzahl vor, sodass eine Generalisierung auf unbekannte Quellen synthetischer Bilder notwendig ist. Wir wollen untersuchen, inwiefern die bisher besprochenen Modelle dies leisten können. Dabei orientieren wir uns wieder an Wolter u. a. [Wol+21], die auch dieses Experiment mit Haar-Wavelets durchgeführt haben. Zur Vergleichbarkeit wählen wir wieder den gleichen Experimentaufbau.

Wir bilden einen Trainingsdatensatz aus insgesamt 120.000 und einen Validierungsdatensatz aus insgesamt 12.000 zufällig ausgewählten Bildern. Jeweils die Hälfte wählen wir aus dem LSUN-Datensatz, die andere Hälfte zu gleichen Teilen aus Bildern, die mit CramerGAN [Bel+17], MMDGAN [Bin+18] und ProGAN [Kar+18] generiert wurden. Mit SN-DCGAN [Miy+18] generierte Bilder fügen wir lediglich zum Test-Datensatz hinzu, der aus 50.000 Bildern besteht, die zu gleichen Teilen zufällig aus allen fünf Datenquellen gewählt wurden. Die ausgewählten Daten bereiten wir wie in Abschnitt 3.3.2 beschrieben vor.

Die Aufgabe besteht nun darin, zu erkennen, ob ein Bild ein DeepFake ist oder nicht. Wir betrachten dazu die gleiche CNN-Architektur mit den gleichen Trainingshyperparametern wie in Abschnitt 3.3.2, wobei wir uns wie auch Wolter u. a. [Wol+21] auf log-skalierte Wavelets beschränken und aufgrund der geringeren Datenmenge 50 statt 20 Epochen trainieren. Die Ergebnisse sind in Tabelle 3.3 dargestellt. Auch hier zitieren wir wieder die Ergebnisse der Haar-Wavelets von Wolter u. a. [Wol+21].

Wir stellen klare Unterschiede zwischen den Symlets und den Daubechies-Wavelets fest. Bei den Daubechies-Wavelets ist keine Generalisierung gelungen. Für alle drei Längen liegt die mittlere Genauigkeit unterhalb der Genauigkeit von 50 %, die bei einem zufälligen Raten zu erwarten wäre. Bei den Symlets ist zwar eine Generalisierung gelungen, jedoch signifikant schlechter als mit den zitierten Haar-Wavelets. Zudem stellen wir hier mit steigender Länge einen deutlich negativen Trend fest.

Tabelle 3.1.: Ergebnisse der Quellenerkennung mittels linearer Regression auf den LSUN and CelebA Datensätzen. Angegeben ist die Genauigkeit auf den Testdaten. Sofern möglich ist die mittlere Genauigkeit sowie die Standardabweichung für fünf Durchläufe angegeben. Das Wavelet mit dem besten Ergebnis ist in kursiv hervorgehoben; das insgesamt beste Ergebnis ist fett gedruckt.

	Genauigkeit CelebA		LSUN	
Methode	max	$\mu \pm \sigma$	max	$\mu \pm \sigma$
Regression Pixel [Wol+21]	97,30 %	95,60 ± 1,62 %	80,50 %	77,31 ± 2,65 %
Regression Wavelets				
Haar [Wol+21]	99,20 %	99,03 ± 0,20 %	86,66 %	82,63 ± 2,15 %
Daubechies 8	99,32 %	95,12 ± 8,20 %	86,34 %	84,26 ± 1,46 %
Daubechies 16	99,22 %	98,87 ± 0,28 %	85,30 %	83,55 ± 2,01 %
Daubechies 24	*99,33 %*	99,25 ± 0,08 %	*86,71 %*	85,60 ± 1,20 %
Symlets 8	99,13 %	98,90 ± 0,28 %	85,53 %	81,78 ± 3,03 %
Symlets 16	99,09 %	98,99 ± 0,10 %	84,57 %	81,07 ± 2,87 %
Symlets 24	99,24 %	98,80 ± 0,43 %	86,52 %	74,56 ± 18,12 %
Regression $\log_e$-Wavelets				
Haar [Wol+21]	95,44 %	94,91 ± 0,53 %	91,50 %	91,20 ± 0,18 %
Daubechies 8	99,29 %	98,91 ± 0,32 %	**97,90** %	97,57 ± 0,54 %
Daubechies 16	98,95 %	98,33 ± 0,70 %	96,24 %	95,81 ± 0,31 %
Daubechies 24	98,69 %	98,17 ± 0,30 %	95,78 %	95,50 ± 0,33 %
Symlets 8	99,37 %	99,07 ± 0,27 %	97,34 %	96,86 ± 0,47 %
Symlets 16	**99,43** %	99,37 ± 0,06 %	97,36 %	97,09 ± 0,25 %
Symlets 24	99,30 %	99,22 ± 0,07 %	96,84 %	96,69 ± 0,17 %

Tabelle 3.2.: Ergebnisse der Quellenerkennung mittels CNN auf den LSUN and CelebA Datensätzen. Angegeben ist die Genauigkeit auf den Testdaten. Sofern möglich ist die mittlere Genauigkeit sowie die Standardabweichung für fünf Durchläufe angegeben. Das Wavelet mit dem besten Ergebnis ist in kursiv hervorgehoben; das insgesamt beste Ergebnis ist fett gedruckt.

	Genauigkeit CelebA		LSUN	
Methode	max	$\mu \pm \sigma$	max	$\mu \pm \sigma$
CNN-Pixel [Wol+21]	98,87 %	98,74 ± 0,14 %	97,06 %	95,02 ± 1,14 %
CNN-DCT [Fra+20]	99,07 %	-	**99,64 %**	-
CNN Wavelets				
Haar [Wol+21]	99,35 %	99,11 ± 0,18 %	93,61 %	92,34 ± 1,49 %
Daubechies 8	99,40 %	99,33 ± 0,06 %	*94,15* %	93,64 ± 0,60 %
Daubechies 16	99,23 %	99,06 ± 0,10 %	92,04 %	91,31 ± 0,61 %
Daubechies 24	99,14 %	98,86 ± 0,30 %	92,13 %	90,85 ± 0,72 %
Symlets 8	99,32 %	99,15 ± 0,17 %	93,27 %	91,27 ± 2,84 %
Symlets 16	99,30 %	99,09 ± 0,12 %	92,93 %	92,11 ± 0,91 %
Symlets 24	99,34 %	98,90 ± 0,50 %	93,83 %	92,40 ± 1,99 %
CNN $\log_e$-Wavelets				
Haar [Wol+21]	97,09 %	96,79 ± 0,29 %	97,14 %	96,89 ± 0,19 %
Daubechies 8	99,66 %	99,50 ± 0,12 %	*99,45* %	99,22 ± 0,23 %
Daubechies 16	99,05 %	98,91 ± 0,14 %	97,91 %	97,86 ± 0,05 %
Daubechies 24	98,31 %	97,49 ± 0,69 %	96,60 %	95,97 ± 0,55 %
Symlets 8	99,58 %	99,35 ± 0,27 %	99,42 %	99,16 ± 0,25 %
Symlets 16	**99,68 %**	99,47 ± 0,11 %	99,30 %	98,88 ± 0,50 %
Symlets 24	99,51 %	99,16 ± 0,46 %	99,24 %	98,92 ± 0,24 %

Tabelle 3.3.: Ergebnisse der GAN-Erkennung mittels CNN auf dem LSUN-Datensatz. Die CNNs wurden sowohl auf echten Daten als auch auf generierten Bildern von CramerGAN [Bel+17], MMDGAN [Bin+18] und ProGAN [Kar+18] trainiert, echte und generierte Bilder unterscheiden zu können. Von SN-DCGAN [Miy+18] generierte Bilder („unbekannte" Quelle) wurden erst zur Testmenge hinzugefügt, die pro Quelle 10.000 Bilder enthielt. Angegeben sind die Genauigkeit auf der gesamten Testmenge sowie auf den bekannten und unbekannten Quellen. Die mittlere Genauigkeit sowie die Standardabweichung sind für fünf Durchläufe angegeben. Das Wavelet mit dem besten Ergebnis ist fett hervorgehoben. [Darstellung nach Wol+21]

	LSUN Klassifikationsgenauigkeit auf Testmenge					
	unbekannte Quellen		bekannte Quellen		gesamt	
CNN-Methode	max	$\mu \pm \sigma$	max	$\mu \pm \sigma$	max	$\mu \pm \sigma$
Pixel [Wol+21]	61,7 %	46,5 ± 12,1 %	95,1 %	91,9 ± 2,1 %	87,1 %	82,8 ± 3,9 %
$\log_e$-Wavelets						
Haar [Wol+21]	**81,7** %	78,8 ± 1,8 %	98,6 %	98,4 ± 0,1 %	**95,2** %	94,5 ± 0,4 %
Daubechies 8	46,4 %	43,9 ± 1,8 %	**99,6** %	99,6 ± 0,0 %	89,0 %	88,4 ± 0,4 %
Daubechies 16	66,5 %	43,0 ± 12,2 %	98,6 %	97,4 ± 0,8 %	92,2 %	86,6 ± 3,0 %
Daubechies 24	59,9 %	47,7 ± 9,4 %	97,8 %	96,2 ± 1,4 %	90,2 %	86,5 ± 3,0 %
Symlets 8	76,0 %	73,9 ± 1,6 %	99,4 %	99,3 ± 0,2 %	94,6 %	94,2 ± 0,4 %
Symlets 16	73,8 %	69,0 ± 3,2 %	99,3 %	98,9 ± 0,3 %	94,2 %	92,9 ± 0,8 %
Symlets 24	64,8 %	57,1 ± 6,2 %	99,3 %	99,0 ± 0,3 %	92,3 %	90,6 ± 1,4 %

4. Diskussion und Ausblick

Im Zuge dieser Arbeit haben wir Wavelet-Transformationen und deren Anwendung auf endliche Signale besprochen. Dazu haben wir zunächst Wavelet-Transformationen auf unendlichen Signalen eingeführt und Ansätze zur Randbehandlung aufgezeigt, mit denen diese Transformationen auf endliche Signale anwendbar sind. Hierfür haben wir mit den Gram-Schmidt-Randfiltern eine Randbehandlungsmethode für orthonormale Wavelets vorgestellt und konstruiert. Wir haben diese mit der Randbehandlung durch symmetrische Fortsetzung verglichen und konnten eine Reduktion der Randartefakte feststellen.

Weiter haben wir die Gram-Schmidt-Randfilter verwendet, um einen Ansatz von Wolter u. a. [Wol+21] zur Erkennung von DeepFakes auf längere Wavelets zu erweitern. In umfangreichen numerischen Experimenten haben wir die Versuche von Wolter u. a. [Wol+21] für Wavelets der Daubechies- und Symlet-Familie nachgestellt. Durch die Verwendung von Randfiltern war es dabei nicht notwendig, die Netzwerkarchitektur in Abhängigkeit der Wavelet-Länge anzupassen.

Bei der Auswertung der Experimente zur Quellenerkennung konnten wir auf fast allen log-skalierten Wavelet-Paket-Koeffizienten signifikante und teils erhebliche Verbesserungen gegenüber den Vergleichswerten feststellen. So hat beispielsweise das beste Ergebnis auf dem CelebA-Datensatz nur knapp halb so viele Test-Samples falsch klassifiziert wie der beste Vergleichswert.

Jedoch setzen sich diese Verbesserungen gegenüber dem Haar-Wavelet bei den Erkennung eines unbekannten Generators nicht fort. Hier schneiden alle längeren Wavelets bei der Genauigkeit auf dem unbekannten GAN durchweg schlechter ab als das Haar-Wavelet. Dabei ist der Einbruch bei den symmetrischeren Symlets bedeutend weniger ausgeprägt als bei den Daubechies-Wavelets, die für alle drei Längen schlechter als naives Raten generalisieren. Es liegt die Vermutung nahe, dass dies mit der stärkeren Asymmetrie der Daubechies-Wavelets zusammenhängen könnte. Auf den Symlets können wir darüber hinaus mit steigender Länge einen Abwärtstrend bei der Genauigkeit auf der unbekannten Quelle feststellen. Somit müssen wir das Fazit ziehen, dass der Einsatz längerer Wavelets mit Gram-Schmidt-Randfiltern für die Generalisierung auf unbekannte Quellen nicht erfolgreich war.

Bei beiden hier besprochenen Versuchsabläufen müssen für jede Konstellation aus Wavelet und Skalierung aus den zugrundeliegenden CelebA- und LSUN-Datensätzen neue Trainingsdaten erzeugt werden. Aufgrund der erheblichen Größe der Datensätze ist dies mit einem hohen Ressourcenaufwand verbunden, weshalb wir in der Anzahl von untersuchten Wavelets beschränkt waren. Daher haben wir auch keine Vergleichsläufe mit anderen Randbehandlungsmethoden durchgeführt. Wir können somit den Einfluss der Gram-Schmidt-Randfilter nicht quantifizieren. Nichtsdestotrotz ist es plausibel, dass die Randfilter einen relevanten Unterschied ausgemacht haben. In den Visualisierungen konnten wir für lange Wavelets erhebliche Reduktionen der Randartefakte beobachten und

Wolter u. a. [Wol+21] beobachten, dass bei einem zu CelebA vergleichbaren Datensatz die Frequenzunterschiede vor allem an den Rändern auftreten.

Insgesamt sind die erzielten Ergebnisse vielversprechend, doch es bedarf weiterer Untersuchungen der Vor- und Nachteile von Gram-Schmidt-Randfiltern im maschinellen Lernen. Insbesondere systematische Vergleiche für verschiedene Längen und Wavelet-Familien mit anderen Randbehandlungsmethoden und in anderen Kontexten sind erforderlich. Hierzu ist es hilfreich, dass die Randfilter die Signalgröße unverändert lassen, wodurch wir sie mit wenig Aufwand in bestehende Architekturen integrieren können.

Ein weiteres interessantes Feld ist die Verbesserung der Randfilter in Anwendungsfällen. In Abschnitt 2.2.2 haben wir Möglichkeiten besprochen, wie die Randfilter durch Multiplikation mit orthonormalen Matrizen optimiert werden könnten. Darauf haben wir in unseren Experimenten verzichtet, jedoch wäre dies ein spannender Ansatz. Da wir den Raum der orthonormalen Matrizen parametrisieren können [HV94], wäre im Kontext von Faltungsnetzwerken sogar eine adaptive Implementierung denkbar, sodass die Randfilter im Trainingsprozess optimiert werden.

Wir sehen also, dass die Wavelet-Randbehandlung mit Randfiltern im Kontext des maschinellen Lernens offene Fragestellungen bietet, die Möglichkeiten der Weiterentwicklung geben.

A. Weitere Abbildungen und Tabellen

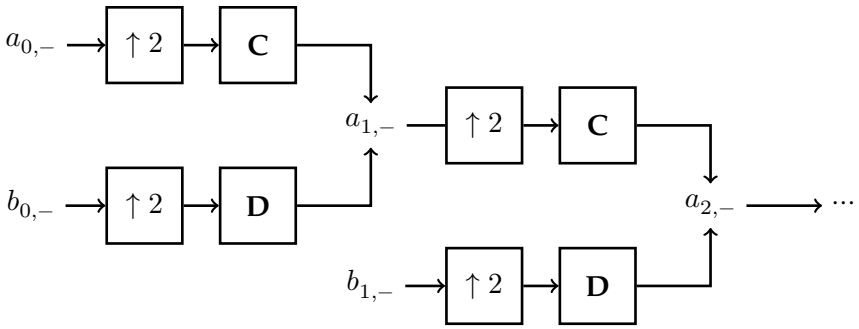
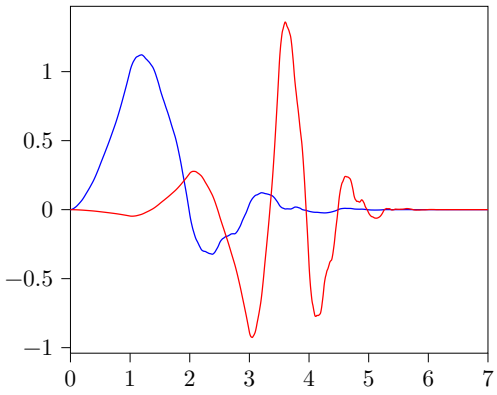


Abbildung A.1.: Schema der Inversen Schnellen Wavelet-Transformation

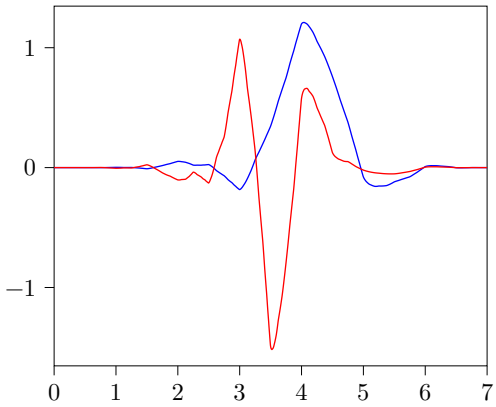
Tabelle A.1.: Die Regressions- und CNN-Architektur von Wolter u. a. [Wol+21], die wir für unsere vergleichenden Experimente in Abschnitt 3.3 nutzen. In Klammern sind für die Faltungskerne und lineare Schichten die Matrixgrößen angegeben.

Regression	
Linear	(49.152, # Klassen)

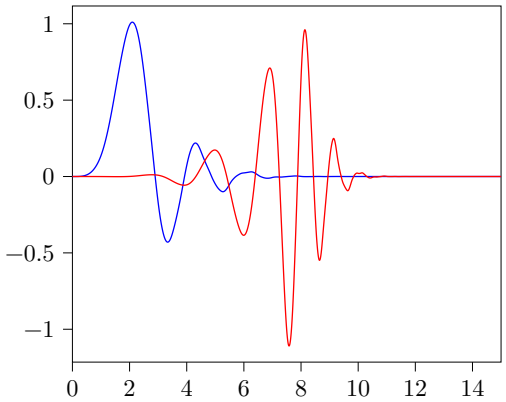
Wavelet-Packet-CNN	
Faltung ReLU, Faltung ReLU, Faltung ReLU,	(192×24×3×3), (24×24×6×6), (24×24×9×9),
Linear	(24, # Klassen)



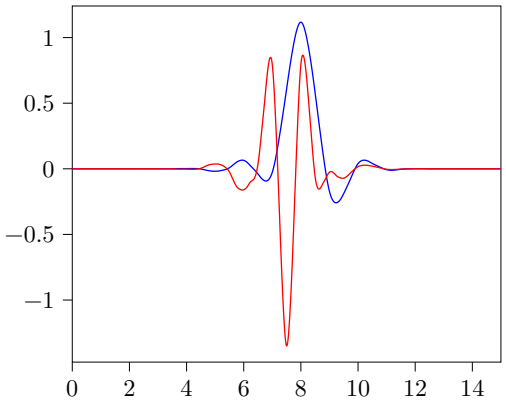
(a) Daubechies-Wavelet der Länge 4.



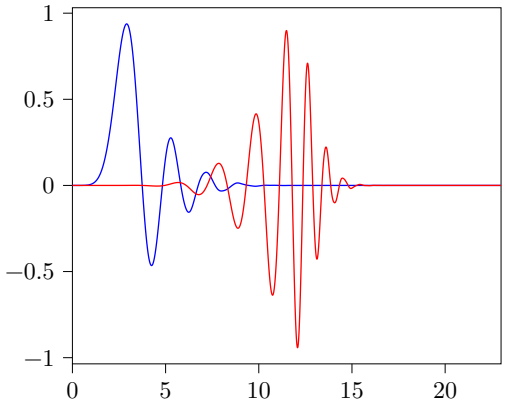
(b) Symlet der Länge 4.



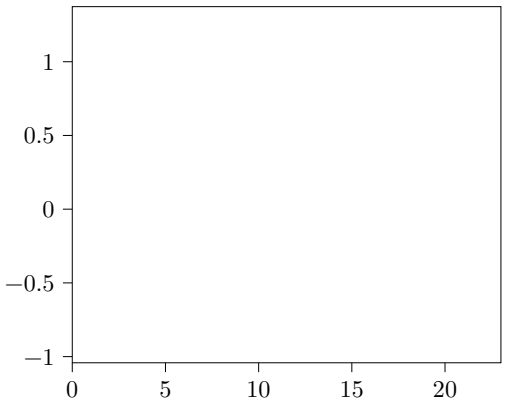
(c) Daubechies-Wavelet der Länge 8.



(d) Symlet der Länge 8.



(e) Daubechies-Wavelet der Länge 12.



(f) Symlet der Länge 12.

Abbildung A.2.: Skalierungsfunktionen (rot) und Mutterwavelets (blau) für Daubechies-Wavelets (oben) und Symlets (unten) der Längen 8, 16 und 24 (von links nach rechts).

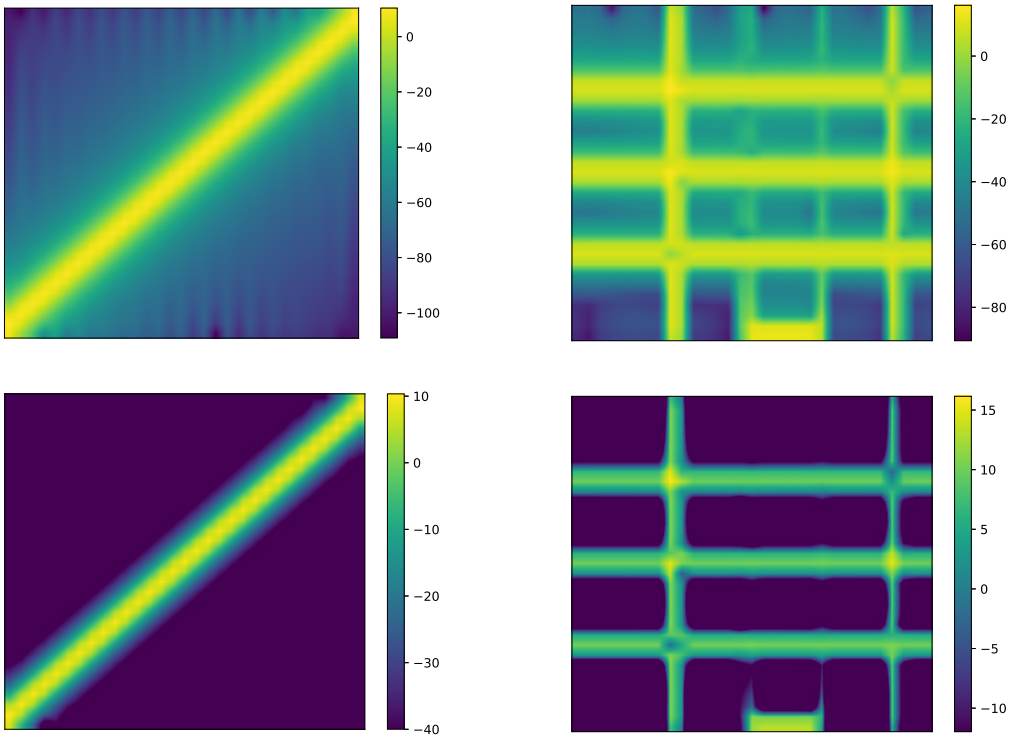
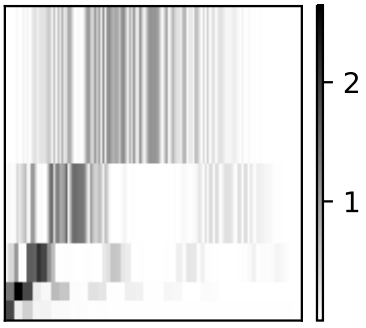
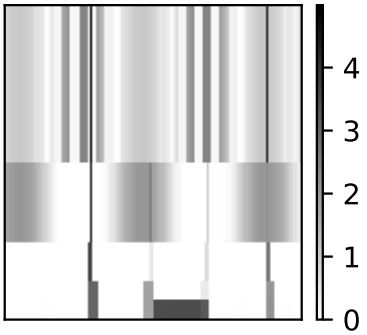


Abbildung A.3.: Spektogramm des ersten Testsignals (links) und des zweiten Testsignals (rechts). In der oberen Zeile sind die unverarbeiteten Spektrogramme angegeben; in der unteren Zeile wurde Rauschen unterhalb einer Stärke von -40 dBc entfernt. Die lineare Frequenzsteigerung (erstes Testsignal) sowie die lokalen Stellen in Zeit und Frequenz (zweites Testsignal) sind gut zu erkennen.



(a) Zeit-Frequenz-Darstellung des ersten Testsignals aus Abschnitt 2.4.1 (linearer Chirp).



(b) Zeit-Frequenz-Darstellung des zweiten Testsignals aus Abschnitt 2.4.1 (lokal in Zeit und Frequenz).

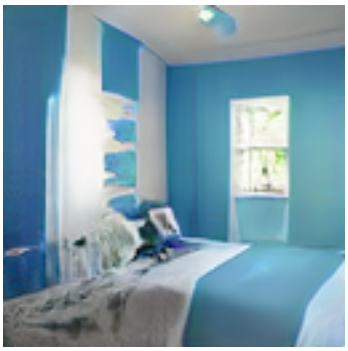
Abbildung A.4.: Fünfskalige Zeit-Frequenz-Darstellungen mittels DWT mit Haar-Wavelets.

(a) Orginalbild aus dem CelebA-Datensatz

(b) Orginalbild aus dem LSUN-Datensatz

Abbildung A.5.: Beispielbilder aus dem CelebA- und dem LSUN-Datensatz.

(a) Auf Basis des CelebA-Datensatzes generiertes Bild.



(b) Auf Basis des LSUN-Datensatzes generiertes Bild.

Abbildung A.6.: Mithilfe von ProGAN [Kar+18] generierte Bilder.

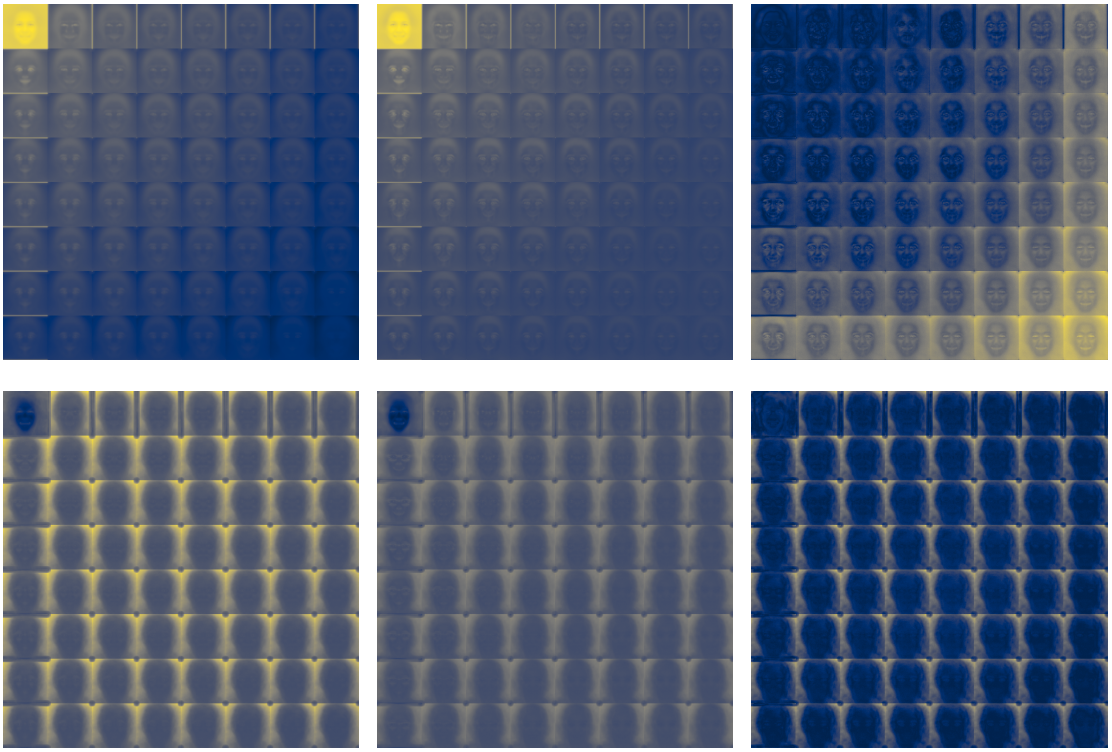


Abbildung A.7.: Dreiskalige $\log_e$-skalierte separable 2D-Wavelet-Paket-Transformation mit Symlets der Länge 24 und Gram-Schmidt-Randfiltern von Bildern des FFHQ-Datensatzes (links) und der StyleGAN-Architektur (mitte) mit den Maßen 1024×1024. Rechts ist die absolute Differenz beider dargestellt. Angegeben sind der Mittelwert (oben) und die Standardabweichung (unten) von jeweils 5000 Bildern.

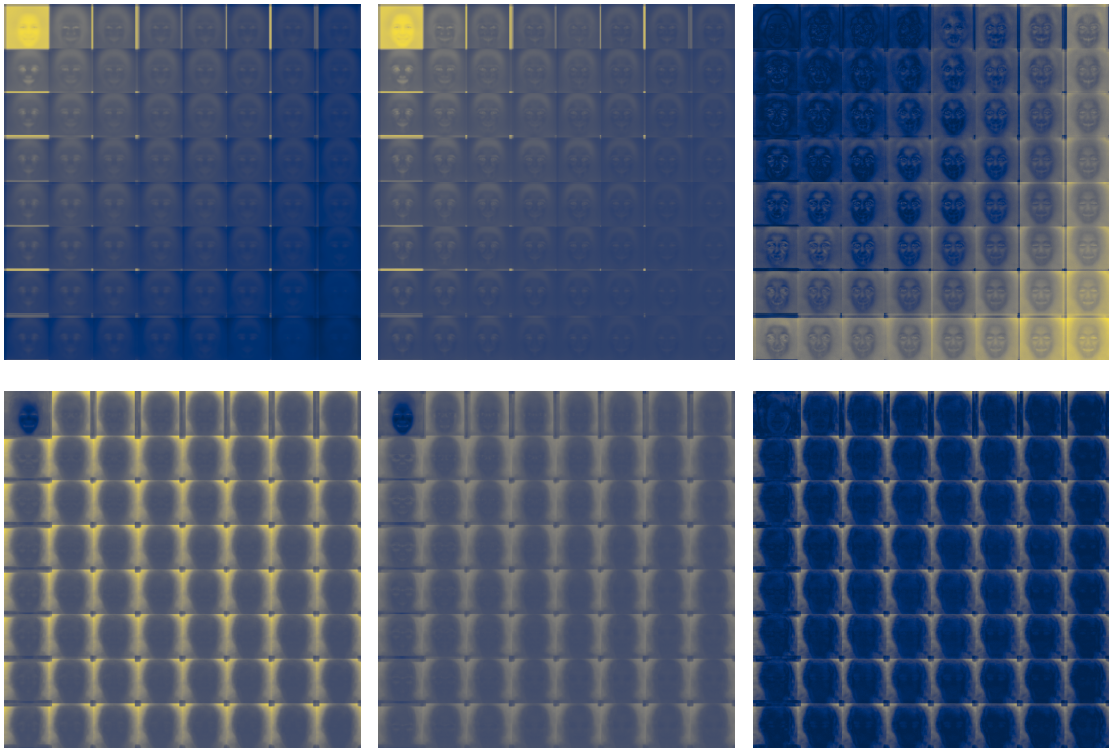


Abbildung A.8.: Dreiskalige $\log_e$-skalierte separable 2D-Wavelet-Paket-Transformation mit Daubechies-Wavelets der Länge 24 und Gram-Schmidt-Randfiltern von Bildern des FFHQ-Datensatzes (links) und der StyleGAN-Architektur (mitte) mit den Maßen 1024×1024. Rechts ist die absolute Differenz beider dargestellt. Angegeben sind der Mittelwert (oben) und die Standardabweichung (unten) von jeweils 5000 Bildern.

Literatur

[Bea04] Richard Beals. *Analysis: an introduction*. Cambridge u.a.: Cambridge Univ. Press, 2004.

[Bel+17] Marc G. Bellemare u. a. „The Cramer Distance as a Solution to Biased Wasserstein Gradients“. In: *CoRR* abs/1705.10743 (2017). arXiv: 1705.10743.

[Bin+18] Mikolaj Binkowski u. a. „Demystifying MMD GANs“. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[CDV93] Albert Cohen, Ingrid Daubechies und Pierre Vial. „Wavelets on the Interval and Fast Wavelet Transforms“. In: *Applied and Computational Harmonic Analysis* 1.1 (1993), S. 54–81.

[Dau88] Ingrid Daubechies. „Orthonormal bases of compactly supported wavelets“. In: *Communications on Pure and Applied Mathematics* 41.7 (1988), S. 909–996.

[Dau92] Ingrid Daubechies. *Ten Lectures on Wavelets*. SIAM, 1992.

[Fra+20] Joel Frank u. a. „Leveraging Frequency Analysis for Deep Fake Image Recognition“. In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Bd. 119. Proceedings of Machine Learning Research. PMLR, 2020, S. 3247–3258.

[Fu+21] Minghan Fu u. a. „DW-GAN: A Discrete Wavelet Transform GAN for NonHomogeneous Dehazing“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. Juni 2021, S. 203–212.

[Gal+21] Rinon Gal u. a. „SWAGAN: A Style-Based Wavelet-Driven Generative Model“. In: *ACM Trans. Graph.* 40.4 (Juli 2021).

[Goo+14] Ian Goodfellow u. a. „Generative Adversarial Nets“. In: *Advances in Neural Information Processing Systems*. Hrsg. von Z. Ghahramani u. a. Bd. 27. Curran Associates, Inc., 2014.

[GV12] Gene H. Golub und Charles F. Van Loan. *Matrix Computations*. 4. Aufl. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2012.

[Haa10] Alfred Haar. „Zur Theorie der orthogonalen Funktionensysteme“. In: *Mathematische Annalen* 69.3 (1910), S. 331–371.

[Her+93] Cormac Herley u. a. „Tilings of the time-frequency plane: construction of arbitrary orthogonal bases and fast tiling algorithms“. In: *IEEE Transactions on Signal Processing* 41.12 (1993), S. 3341–3359.

[HV94] Cormac Herley und Martin Vetterli. „Orthogonal time-varying filter banks and wavelet packets“. In: *IEEE Transactions on Signal Processing* 42.10 (1994), S. 2650–2663.

[Jl01] Arne Jensen und Anders la Cour-Harbo. *Ripples in mathematics: the discrete wavelet transform*. Springer Science & Business Media, 2001.

[Kar+18] Tero Karras u. a. „Progressive Growing of GANs for Improved Quality, Stability, and Variation“. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[KB15] Diederik P. Kingma und Jimmy Ba. „Adam: A Method for Stochastic Optimization“. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Hrsg. von Yoshua Bengio und Yann LeCun. 2015.

[KLA19] Tero Karras, Samuli Laine und Timo Aila. *A Style-Based Generator Architecture for Generative Adversarial Networks*. 2019. arXiv: 1812.04948 [cs.NE].

[KV89] Gunnar Karlsson und Martin Vetterli. „Extension of finite length signals for sub-band coding“. In: *Signal Processing* 17.2 (1989), S. 161–168.

[Lee+19] Gregory R. Lee u. a. „PyWavelets: A Python package for wavelet analysis“. In: *Journal of Open Source Software* 4.36 (2019), S. 1237.

[Liu+15] Ziwei Liu u. a. „Deep Learning Face Attributes in the Wild“. In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dez. 2015.

[Mal89] Stéphane Mallat. „A theory for multiresolution signal decomposition: the wavelet representation“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.7 (1989), S. 674–693.

[Mer20] Alfred Mertins. *Signaltheorie: Grundlagen der Signalbeschreibung, Filterbänke, Wavelets, Zeit-Frequenz-Analyse, Parameter- und Signalschätzung*. 4. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden, 2020.

[Mey91] Yves Meyer. „Ondelettes sur l'intervalle.“ Französisch. In: *Revista Matemática Iberoamericana* 7.2 (1991), S. 115–133.

[Miy+18] Takeru Miyato u. a. „Spectral Normalization for Generative Adversarial Networks“. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[Pas+19] Adam Paszke u. a. „PyTorch: An Imperative Style, High-Performance Deep Learning Library“. In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Hrsg. von Hanna M. Wallach u. a. 2019, S. 8024–8035.

[Sim19] Tom Simonite. *Artificial intelligence is coming for our faces*. wired. Juni 2019. URL: www.wired.com/story/artificial-intelligence-fake-fakes/ (besucht am 26. 08. 2021).

[SN96] Gilbert Strang und Truong Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.

[TM02] David S Taubman und Michael W Marcellin. „JPEG2000: Standard for interactive imaging“. In: *Proceedings of the IEEE* 90.8 (2002), S. 1336–1357.

[VK95] Martin Vetterli und Jelena Kovačević. *Wavelets and subband coding*. Prentice Hall, 1995.

[WB19] Jevin West und Carl Bergstrom. *Which Face Is Real?* 2019. URL: whichfaceisreal.com (besucht am 26. 08. 2021).

[Wer18] Dirk Werner. *Funktionalanalysis*. 8. Aufl. 2018. Springer-Lehrbuch. Berlin, Heidelberg: Springer Spektrum, 2018.

[Wes19] Mika Westerlund. „The Emergence of Deepfake Technology: A Review“. In: *Technology Innovation Management Review* 9 (Nov. 2019), S. 40–53.

[WG21] Moritz Wolter und Jochen Garcke. „Adaptive wavelet pooling for convolutional neural networks“. In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Bd. 130. Proceedings of Machine Learning Research. PMLR, Apr. 2021, S. 1936–1944.

[WL18] Travis Williams und Robert Li. „Wavelet pooling for convolutional neural networks“. In: *International Conference on Learning Representations*. 2018.

[Wol+21] Moritz Wolter u. a. *Wavelet-Packet Powered Deepfake Image Detection*. 2021. arXiv: 2106.09369 [cs.CV].

[Wol21] Moritz Wolter. „Frequency Domain Methods in Recurrent Neural Networks for Sequential Data Processing“. Diss. Rheinische Friedrich-Wilhelms-Universität Bonn, Juli 2021.

[YDF19] Ning Yu, Larry Davis und Mario Fritz. „Attributing Fake Images to GANs: Learning and Analyzing GAN Fingerprints“. In: *IEEE International Conference on Computer Vision (ICCV)*. 2019.

[Yu+15] Fisher Yu u. a. *LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop*. 2015. arXiv: 1506.03365 [cs.CV].

Abbildungsverzeichnis

Tabellenverzeichnis