



Professur für Mathematik
und Operations Research
85577 Neubiberg
Tel.: 089/6004-3931
Fax: 089/6004-2615

Black-Box-Modellierung mit Wavelet-Netzwerken

OBERLEUTNANT DANIEL POHL

Vorsitzender des Promotionsausschusses: Univ.-Prof. Dr.-Ing. Dieter Gerling
1. Berichterstatter: Univ.-Prof. Dr.rer.nat. Stefan Schäffler
2. Berichterstatter: Univ.-Prof. Dr.rer.nat. Mathias Richter

Tag der Prüfung: 26.05.2011

Mit der Promotion erlangter akademischer Grad:
Doktor-Ingenieur
(Dr.-Ing.)

Neubiberg, den 15. Juni 2011

Inhaltsverzeichnis

1	Einführung	4
2	Mathematische Grundlagen	6
2.1	Funktionsapproximation mit Taylor-Polynomen	6
2.2	Funktionsapproximation mit Fourier-Reihen	8
2.3	Wavelets	13
2.4	Wavelet-Frames	22
2.5	Ermittlung der Frameeigenschaft für einen konkreten Fall	25
2.6	Das lineare statistische Modell	34
2.6.1	Kleinste-Quadrate-Schätzung	36
2.6.2	Varianz-Schätzung	37
3	Wavelet-Netzwerke	38
3.1	Die Problemstellung	39
3.2	Die Struktur eines Wavelet-Netzwerkes	40
3.3	Probleme bei der Konstruktion von Wavelet-Netzwerken	42
3.4	Implementierungen in dieser Arbeit	42
3.5	Teilaufgabengliederung	43
4	Standardalgorithmen	45
4.1	Erstellung einer Wavelet-Bibliothek	45
4.2	Regressorauswahl	48
4.2.1	Sukzessive Regressorauswahl mit schrittweiser Orthogonalisierung	50
4.2.2	Rückwärtselimination von Regressoren	55
4.2.3	Akaike's Final Prediction Error	59
4.3	Regressoroptimierung mit Levenberg-Marquardt	60
4.4	Konditionsprobleme mit Standardalgorithmen	63
5	Verbesserung von Standardalgorithmen	73
5.1	Eine Cluster-Bibliothek basierend auf dem K-means-Algorithmus	73
5.2	Ein robustifizierter Algorithmus zur Vorwärts-Regressorauswahl	76
5.3	Eine robustifizierte Version des Levenberg-Marquardt-Algorithmus	77
5.4	Ergebnisse der robustifizierten Algorithmen	79
6	Ein neues, statistisches Gütekriterium	86
6.1	Vereinbarungen	86
6.2	Statistischer Raum und Hypothesen	89
6.3	Die Modellunsicherheit einer Hypothese H_{I_r}	94
6.4	Bewertung von Hypothesen	96

7	Algorithmen zum neuen Gütekriterium	100
7.1	Eine rekursiv erzeugte Wavelet-Bibliothek	100
7.1.1	Der eindimensionale Fall	101
7.1.2	Der mehrdimensionale Fall	104
7.2	Betrachtungen zum Mexikanerhut-Wavelet	105
7.2.1	Grundlegende Integrale	107
7.2.2	Mehrdimensionale Integrale	109
7.2.3	Das Skalarprodukt zweier Wavelets	111
7.2.4	Rechenaufwand für das Skalarprodukt zweier Wavelets	116
7.3	Effiziente Matrixinversion	118
7.4	Das Prinzip der sukzessiven Regressorauswahl	121
7.5	Sukzessive Regressorauswahl mit Konditionsbetrachtung	123
7.5.1	Rahmenalgorithmus	123
7.5.2	Effiziente Berechnung der Kriteriumsfunktion	124
7.5.3	Algorithmische Aufbereitung	128
7.6	Rückwärtselimination von Regressoren	130
7.7	Regressorelimination mit Konditionsbetrachtung	132
7.7.1	Rahmenalgorithmus	132
7.7.2	Effiziente Berechnung der Kriteriumsfunktion	133
7.7.3	Algorithmische Aufbereitung	137
7.8	Erweiterung der Algorithmen	139
7.9	Ergebnisse der statistischen Algorithmen	142
7.10	Verbesserung der Varianz-Schätzung	148
8	Fazit und Ausblick	150
	Literaturverzeichnis	152
	Danksagung	153

1 Einführung

Phänomenologische Modellierung oder auch Black-Box-Modellierung ist eine Ingenieur-Methodik, die erst mit der Verbreitung effizienter Rechentechnik wahre Bedeutung erlangt hat. Sie ist immer dann sinnvoll, wenn die Reaktion eines technischen oder natürlichen Systems auf gegebene Eingangsgrößen zwar im Wesentlichen deterministisch ist, aber eine klassische Modellierung zu umständlich wäre oder nicht möglich ist. Mit klassisch ist hier eine Modellierung gemeint, die auf den Gesetzen der Physik basiert und von dort aus deduktiv vorgeht. Grundlage einer Black-Box-Modellbildung sind mehrere Messungen von Ein- und Ausgangsgrößen am zu modellierenden System, die dessen wesentliches Verhalten aufnehmen. Reale Black-Box-Modellierungsprobleme lassen sich oft auf das folgende mathematische Problem zurückführen: Finde eine Funktion, die möglichst gut durch gegebene Stützstellen verläuft und sich dazwischen „vernünftig“ verhält. Damit sind wir bei einer klassischen mathematischen Problematik. Es gilt aus einer möglichst reichhaltigen Grundmenge an Funktionen auf Grundlage der Stützstellen und Zusatzkriterien die beste Funktion auszuwählen. Diese Aufgabe stellt grundsätzlich ein Spannungsfeld dar. Ist die Grundmenge sehr eingeschränkt, lässt sich leicht ein Gütekriterium aufstellen und eine beste Funktion finden. Lässt man z.B. nur lineare Funktionen zu, so ist der mittlere quadratische Fehler ein einfaches und gutes Gütekriterium und mit einem linearen Ausgleichsproblem findet man effizient und eindeutig die beste Funktion. Nimmt man eine reichhaltigere Grundmenge, so werden meist komplexere Gütekriterien benötigt. Die Suche nach der besten Funktion gestaltet sich schwieriger und oft findet man nur ein Suboptimum. Einer der klassischen Kompromisse in diesem Spannungsfeld sind Neuronale Netzwerke. Mit ihnen kann man eine reichhaltige Grundmenge an Funktionen abdecken und mit effizienten Algorithmen ein suboptimales Ergebnis erhalten. Die große Schwäche ist das Gütekriterium. Das äußert sich besonders im Problem des sogenannten Overtrainings, also darin, dass eigentlich schon gute Resultate sich bei weiteren Iterationen des Suchalgorithmus wieder verschlechtern.

Wavelet-Netzwerke wurden erstmals 1992 von Qingha Zhang und A. Benveniste [1] als Weiterentwicklung von Neuronalen Netzen eingeführt. Sie sind vielversprechend für Black-Box-Modellierungsprobleme. Thema dieser Dissertation ist die Konstruktion jener Wavelet-Netzwerke. Dabei liegt der Fokus auf den dafür verwendeten Algorithmen und deren mathematischen Grundlagen. Alle hier vorgestellten Verfahren wurden implementiert und auf Beispielprobleme angewendet. Ihre Ergebnisse werden anschließend ausgewertet. In den ersten beiden Kapiteln dieser Dokumentation wird gezeigt, auf welchen theoretischen Grundlagen Wavelet-Netzwerke basieren. Es folgt die Einführung von Standardalgorithmen und die Analyse ihrer Probleme in Kapitel 4. In einem ersten Lösungsansatz werden im darauf folgenden Kapitel heuristisch motivierte Weiterentwicklungen vorgestellt mit dem Ziel, die Robustheit der Modellbildung zu verbessern. Herzstück dieser Dissertation ist die Einführung eines ganz neuen,

statistisch motivierten Gütekriteriums für Wavelet-Netzwerke in Kapitel 6. Dieses Gütekriterium wird in Kapitel 7 mit Hilfe von eigens dafür angefertigten, effizienten Konstruktionsalgorithmen umgesetzt. Dabei wird sich sein Nutzen klar herausstellen. Wir wünschen viel Vergnügen bei der Lektüre.

2 Mathematische Grundlagen

Die mathematischen Grundlagen in den ersten drei Abschnitten dieses Kapitels sind zu großen Teilen direkt oder in angepasster Form aus [7] übernommen. In diesen einleitenden Teilen geht es um eine der wichtigsten Fragen der angewandten Mathematik, nämlich der Approximation gegebener Funktionen durch Linearkombinationen einer gegebenen Menge von Basisfunktionen. In diesem Zusammenhang gibt es drei klassische Vorgehensweisen:

- Approximation durch Taylor-Polynome: Die Basisfunktionen sind Polynome
- Approximation durch Fourier-Reihen: Die Basisfunktionen sind Schwingungen
- Approximation durch Wavelets: Die Basisfunktionen sind hierbei durch spezielle Eigenschaften definiert, um die Nachteile der ersten beiden Approximationsarten zu umgehen.

Im Folgenden sollen der Taylor-Ansatz und die Approximation durch Fourier-Reihen kurz skizziert werden und auf entsprechende Eigenschaften eingegangen werden. Es folgt die Einführung von Wavelets, auf denen unser Hauptaugenmerk liegen wird. Weiterhin wollen wir den wichtigen Begriff des Wavelet-Frames einführen, weil dieser eine theoretische Grundlage für die Konstruktion eines Wavelet-Netzwerkes darstellt. Zum Schluss wird für einen konkreten Fall die Frame-Eigenschaft einer sogenannten Waveletfamilie gezeigt werden. Dieses letzte Ergebnis soll im Abschnitt 4.1 wieder aufgegriffen werden.

2.1 Funktionsapproximation mit Taylor-Polynomen

Zuerst wollen wir die Approximation von Funktionen durch Polynome genauer betrachten.

Satz 2.1

Seien $I \subseteq \mathbb{R}$ ein Intervall, das aus mindestens zwei Punkten besteht.

Sei $n \in \mathbb{N}_0$ und

$$f : I \rightarrow \mathbb{R}, x \mapsto f(x)$$

eine $(n + 1)$ -mal stetig differenzierbare Funktion in I , dann gilt für $a, x \in I$:

$$f(x) = \sum_{k=0}^n \frac{f^{(k)}(a)}{k!} (x - a)^k + R_{n+1}(x),$$

ferner existiert zu $a, x \in I$ ein ξ zwischen a und x (also $\xi \in [a, x]$ oder $\xi \in [x, a]$) mit:

$$R_{n+1}(x) = \frac{f^{(n+1)}(\xi)}{(n + 1)!} (x - a)^{n+1}.$$

In dieser Aussage wird somit eine gegebene Funktion f mit geeigneten Eigenschaften durch folgendes Polynom approximiert:

$$\sum_{k=0}^n \frac{f^{(k)}(a)}{k!} (x - a)^k$$

Der Approximationsfehler ist durch das Restglied

$$R_{n+1}(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} (x - a)^{n+1}$$

beschrieben. Auffällig an dieser Art der Approximation ist die Tatsache, dass einerseits die Funktion f hinreichend oft stetig differenzierbar sein muss und andererseits, dass der „Entwicklungspunkt“ a eine entscheidende Rolle spielt. Genauer gesagt: Eine in einer Umgebung eines Punktes $a \in \mathbb{R}$ n -mal stetig differenzierbare Funktion f kann in dieser Umgebung bis auf einen Fehler der Ordnung $o(|x - a|^n)$ durch das Taylor-Polynom n -ter Ordnung

$$T_n[f, a] : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto \sum_{k=0}^n \frac{f^{(k)}(a)}{k!} (x - a)^k$$

approximiert werden:

Satz 2.2

Seien $I \subseteq \mathbb{R}$ ein Intervall, das aus mindestens zwei Punkten besteht, $n \in \mathbb{N}_0$, $a \in I$ und

$$f : I \rightarrow \mathbb{R}, \quad x \mapsto f(x)$$

eine n -mal stetig differenzierbare Funktion in I , dann gilt für alle $x \in I$

$$f(x) = \sum_{k=0}^n \frac{f^{(k)}(a)}{k!} (x - a)^k + o(|x - a|^n) \quad \text{für } x \rightarrow a.$$

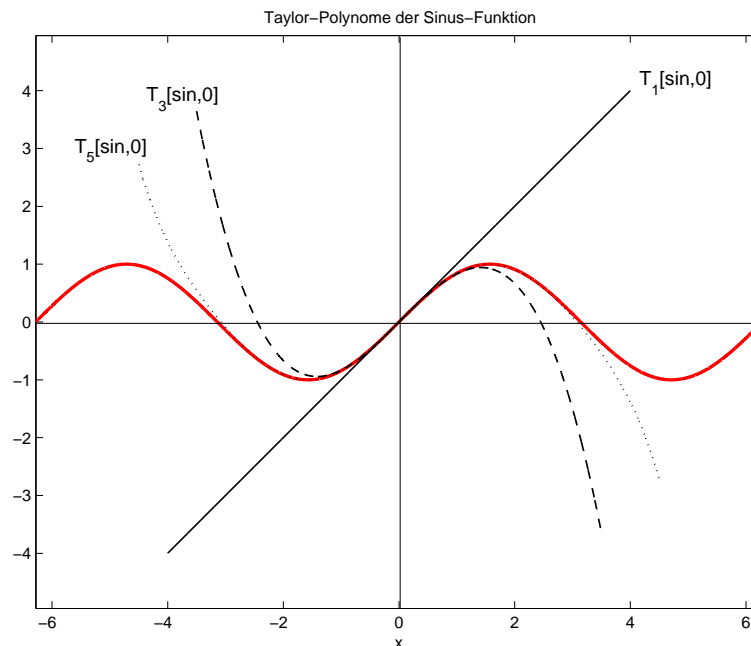
Mit anderen Worten: Für $x \rightarrow a$ konvergiert die Differenz (und damit der Fehler der Approximation)

$$f(x) - \sum_{k=0}^n \frac{f^{(k)}(a)}{k!} (x - a)^k$$

schneller gegen Null als die Funktion

$$|x - a|^n.$$

Dies ist die Stärke der Approximation von Funktionen durch den Ansatz von Taylor. Die große Schwäche zeigt sich, wenn man einen anderen Punkt $x \in I$, $x \neq a$ betrachtet, wie das folgende Beispiel $f = \sin$ und $a = 0$ zeigt.



Um dieser extremen „Lokalität“ zu entgehen, kann man versuchen, von Polynomen auf Potenzreihen überzugehen. Diese bestehen aber aus unendlich vielen Summanden und was noch schlimmer ist - Die Lokalität kann in einigen Fällen trotz dieser unendlich vielen Summanden erhalten bleiben.

2.2 Funktionsapproximation mit Fourier-Reihen

Fourier-Reihen bilden das wichtigste Werkzeug der Mathematik zur Approximation von Funktionen durch Schwingungen; daher ist die Theorie der Fourier-Reihen zum Beispiel in der Kommunikationstechnik unentbehrlich.

Definition 2.3

Seien $L > 0$ eine reelle Zahl und

$$f : \mathbb{R} \rightarrow \mathbb{C}$$

eine Funktion mit

$$f(x + L) = f(x) \quad \text{für alle } x \in \mathbb{R},$$

dann heißt die Funktion f L -periodisch bzw. periodisch mit der Periode L .

Für jede L -periodische Funktion gilt offensichtlich

$$f(x + nL) = f(x) \quad \text{für alle } n \in \mathbb{Z} \quad \text{und alle } x \in \mathbb{R}.$$

Durch die Transformation

$$F : \mathbb{R} \rightarrow \mathbb{C}, \quad x \mapsto f\left(\frac{L}{2\pi}x\right)$$

wird aus einer L -periodischen Funktion f eine 2π -periodische Funktion F . Aus F gewinnt man die Funktion f durch

$$f(x) = F\left(\frac{2\pi}{L}x\right) \quad \text{für alle } x \in \mathbb{R}$$

zurück. Wir werden uns daher im Folgenden auf 2π -periodische Funktionen beschränken.

▷ **Beispiele:**

- Die reellen trigonometrischen Polynome der Ordnung n gegeben durch

$$f : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto \frac{a_0}{2} + \sum_{k=1}^n [a_k \cos(kx) + b_k \sin(kx)], \quad a_0, a_i, b_i \in \mathbb{R}, \quad i \in \mathbb{N}$$

sind 2π -periodische Funktionen. Da mit partieller Integration gezeigt werden kann, dass

$$\begin{aligned} \int_0^{2\pi} \cos(kx) \sin(lx) dx &= 0 \quad \text{für alle } k, l \in \mathbb{N}_0 \\ \int_0^{2\pi} \cos(kx) \cos(lx) dx &= \int_0^{2\pi} \sin(kx) \sin(lx) dx = 0 \quad \text{für alle } k, l \in \mathbb{N}_0, \quad k \neq l \\ \int_0^{2\pi} \cos^2(kx) dx &= \int_0^{2\pi} \sin^2(kx) dx = \pi \quad \text{für alle } k \in \mathbb{N}, \end{aligned}$$

sind für

$$f : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto \frac{a_0}{2} + \sum_{k=1}^n [a_k \cos(kx) + b_k \sin(kx)], \quad a_0, a_i, b_i \in \mathbb{R}, \quad i \in \mathbb{N}$$

die Konstanten

$$a_0, a_i, b_i \in \mathbb{R}, \quad i \in \mathbb{N}$$

durch f eindeutig bestimmt. Zum Beispiel erhält man aus der Gleichung

$$f(x) \cdot \cos(mx) = \frac{a_0}{2} \cos(mx) + \sum_{k=1}^n [a_k \cos(kx) \cos(mx) + b_k \sin(kx) \cos(mx)]$$

mit $m \in \mathbb{N}_0$ durch Integration:

$$\int_0^{2\pi} f(x) \cos(mx) dx = \pi a_m \quad \text{bzw.} \quad a_m = \frac{1}{\pi} \int_0^{2\pi} f(x) \cos(mx) dx.$$

Analog erhält man für $n \in \mathbb{N}$:

$$\int_0^{2\pi} f(x) \sin(nx) dx = \pi b_n \quad \text{bzw.} \quad b_n = \frac{1}{\pi} \int_0^{2\pi} f(x) \sin(nx) dx.$$

- Verwendet man in der Abbildungsvorschrift

$$f : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto \frac{a_0}{2} + \sum_{k=1}^n [a_k \cos(kx) + b_k \sin(kx)], \quad a_0, a_i, b_i \in \mathbb{R}, \quad i \in \mathbb{N}$$

die bekannten Formeln

$$\begin{aligned} \cos(kx) &= \frac{1}{2} (e^{ikx} + e^{-ikx}) \quad k \in \mathbb{N}_0 \\ \sin(kx) &= \frac{1}{2i} (e^{ikx} - e^{-ikx}) \quad k \in \mathbb{N}_0, \end{aligned}$$

so ergibt sich für alle $x \in \mathbb{R}$:

$$f(x) = \sum_{k=1}^n c_{-k} e^{-ikx} + \sum_{k=0}^n c_k e^{ikx} =: \sum_{k=-n}^n c_k e^{ikx}$$

mit $c_0 = \frac{a_0}{2}$, $c_{-k} = \frac{1}{2} (a_k + ib_k)$ und $c_k = \frac{1}{2} (a_k - ib_k)$, $k \in \mathbb{N}$. Diese Beobachtung führt zu der Idee, auch komplexwertige trigonometrische Funktionen

$$f : \mathbb{R} \rightarrow \mathbb{C}, \quad x \mapsto \sum_{k=-n}^n c_k e^{ikx}, \quad c_k \in \mathbb{C}, \quad k \in \{-n, \dots, -1, 0, 1, \dots, n\},$$

die auch 2π -periodisch sind, zu betrachten.

◁

Um nun bei komplexwertigen trigonometrischen Funktionen

$$f : \mathbb{R} \rightarrow \mathbb{C}, \quad x \mapsto \sum_{k=-n}^n c_k e^{ikx}, \quad c_k \in \mathbb{C}, \quad k \in \{-n, -n+1, \dots, -1, 0, 1, \dots, n\},$$

durch Integration zu zeigen, dass die Koeffizienten $c_k \in \mathbb{C}$, $k \in \{-n, -n+1, \dots, -1, 0, 1, \dots, n\}$ eindeutig bestimmt sind, benötigt man das Riemann-Integral für komplexwertige Funktionen. Seien dazu $u, v : [a, b] \rightarrow \mathbb{R}$ Riemann-integrierbare Funktionen, dann heißt auch die komplexwertige Funktion

$$\phi : [a, b] \rightarrow \mathbb{C}, \quad x \mapsto u(x) + iv(x)$$

Riemann-integrierbar und es gilt

$$\int_a^b \phi(x) dx := \int_a^b u(x) dx + i \int_a^b v(x) dx.$$

Speziell für Funktionen

$$\phi : [a, b] \rightarrow \mathbb{C}, \quad x \mapsto e^{imx}, \quad m \in \mathbb{Z} \setminus \{0\}$$

ergibt sich

$$\int_a^b e^{imx} dx = \frac{1}{im} e^{imx} \Big|_a^b$$

Es folgt für $a = 0$ und $b = 2\pi$:

$$\int_0^{2\pi} e^{imx} dx = 0.$$

Für

$$f : \mathbb{R} \rightarrow \mathbb{C}, \quad x \mapsto \sum_{k=-n}^n c_k e^{ikx}, \quad c_k \in \mathbb{C}, \quad k \in \{-n, -n+1, \dots, -1, 0, 1, \dots, n\}$$

gilt

$$f(x)e^{-imx} = \sum_{k=-n}^n c_k e^{i(k-m)x} \quad \text{für alle } x \in \mathbb{R}$$

und somit durch Integration

$$c_m = \frac{1}{2\pi} \int_0^{2\pi} f(x)e^{-imx} dx \quad c_m \in \mathbb{C}, \quad m \in \{-n, -n+1, \dots, -1, 0, 1, \dots, n\}.$$

Im Folgenden soll eine gegebene 2π -periodische Riemann-integrierbare Funktion g durch eine Folge von komplexen trigonometrischen Polynomen approximiert werden. Daher definiert man

Definition 2.4

Sei $g : \mathbb{R} \rightarrow \mathbb{C}$ eine 2π -periodische und über dem Intervall $[0, 2\pi]$ Riemann-integrierbare Funktion, dann heißen die Zahlen

$$c_m = \frac{1}{2\pi} \int_0^{2\pi} g(x)e^{-imx} dx \quad c_m \in \mathbb{C}, \quad m \in \mathbb{Z}$$

Fourier-Koeffizienten von g und mit

$$\mathcal{F}_n[g] : \mathbb{R} \rightarrow \mathbb{C}, \quad x \mapsto \sum_{k=-n}^n c_k e^{ikx}$$

die Funktionenfolge $(\mathcal{F}_n[g])_{n \geq 0}$ Fourier-Reihe von f .

Im Allgemeinen konvergiert die Fourier-Reihe einer Funktion g weder gleichmäßig noch punktweise gegen g . Daher muss man auf einen anderen Konvergenzbegriff, dem der Konvergenz in quadratischen Mittel, ausweichen.

Definition 2.5

Seien

$$f, f_n : \mathbb{R} \rightarrow \mathbb{C}$$

für alle $n \in \mathbb{N}_0$ 2π -periodische und über dem Intervall $[0, 2\pi]$ Riemann-integrierbare Funktionen, dann heißt die Funktionenfolge $(f_n)_{n \geq 0}$ konvergent im quadratischen Mittel gegen f , falls

$$\lim_{n \rightarrow \infty} \int_0^{2\pi} |f(x) - f_n(x)|^2 dx = 0.$$

Bei diesem Konvergenzbegriff wird die quadratische Abweichung der zu betrachtenden Funktion und ihrer entsprechenden Approximation über dem ganzen Intervall $[0, 2\pi]$ integriert.

Satz 2.6

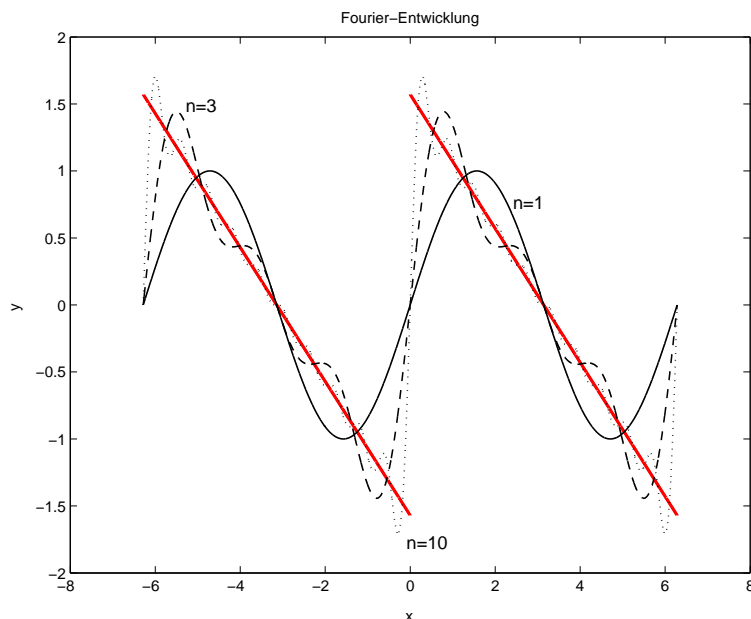
Sei $g : \mathbb{R} \rightarrow \mathbb{C}$ eine 2π -periodische und über dem Intervall $[0, 2\pi]$ Riemann-integrierbare Funktion, dann konvergiert die Fourier-Reihe von g im quadratischen Mittel gegen g und für die Fourierkoeffizienten c_k gilt:

$$\lim_{n \rightarrow \infty} \sum_{k=-n}^n |c_k|^2 = \frac{1}{2\pi} \int_0^{2\pi} |f(x)|^2 dx.$$

Die folgende Abbildung zeigt die 2π -periodische Funktion g mit

$$g_{|[0,2\pi]} : [0, 2\pi] \rightarrow \mathbb{R}, \quad x \mapsto \frac{\pi}{2} - \frac{1}{2}x$$

und die Funktionen $\mathcal{F}_1[g]$, $\mathcal{F}_3[g]$ und $\mathcal{F}_{10}[g]$ im Intervall $[-2\pi, 2\pi]$.



Interessant ist, dass die Fourier-Reihe an keiner Stelle gegen die zu approximierende Funktion konvergiert. Die Approximation von Funktionen durch Fourier-Reihen ist somit eine extrem „nichtlokale“ Methode.

Lässt man bei den Fourier-Reihen neben ganzzahligen Frequenzen in e^{ikx} , $k \in \mathbb{Z}$, auch reelle Frequenzen zu, so kommt man zu Fourier-Transformation einer geeigneten Funktion $g : \mathbb{R} \rightarrow \mathbb{C}$:

$$\hat{g} : \mathbb{R} \rightarrow \mathbb{C}, \quad \alpha \mapsto \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} g(t)e^{-i\alpha t} dt.$$

Die Funktionswerte der Fourier-Transformierten \hat{g} sind Analoga zu den Koeffizienten c_k der Fourier-Reihe, wie an der Umkehrformel

$$g(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{g}(\alpha)e^{i\alpha x} d\alpha$$

zu erkennen ist. Auch hier besteht das Problem der „Nichtlokalität“: Eine Frequenz α beschreibt kein lokales Verhalten von g (Maxima, Minima, Sprungstellen) so, wie ein Argument x von g kein lokales Verhalten von \hat{g} beschreibt.

2.3 Wavelets

Fasst man die bisher betrachteten Methoden zur Approximation von Funktionen zusammen, so wird bei der Taylor-Entwicklung die Funktion f als Linearkombination der Basisfunktionen

$$\{1, (x - a), (x - a)^2, (x - a)^3, \dots\}$$

approximiert bzw. dargestellt, während beim Fourier-Ansatz die Funktion g als Linearkombination der Basisfunktionen

$$\{\dots, e^{i(-2)x}, e^{i(-x)}, 1, e^{ix}, e^{i2x}, \dots\}.$$

approximiert bzw. dargestellt wird. Wie die Fourier-Transformation zeigt, kann es auch überabzählbar viele Basisfunktionen

$$\{e^{i\alpha x}; \alpha \in \mathbb{R}\}$$

geben. In diesem Fall kann man nicht mehr von Linearkombinationen sprechen, sondern man erhält statt den (un)endlichen Summen

- Taylor: $\sum_{k=0}^n \frac{f^{(k)}(a)}{k!} (x - a)^k$ bzw. $\sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!} (x - a)^k$ mit den Koeffizienten $\frac{f^{(k)}(a)}{k!}$.
- Fourier: $\sum_{k=-n}^n c_k e^{ikx}$ bzw. $\sum_{k=-\infty}^{\infty} c_k e^{ikx}$ mit den Koeffizienten $c_k = \frac{1}{2\pi} \int_0^{2\pi} g(x) e^{-ikx} dx$

das Integral

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{g}(\alpha) e^{i\alpha x} d\alpha$$

als Darstellung für g . Die Grundidee der Wavelet-Transformation basiert nun auf der Methodik der Fourier-Transformation mit dem Ziel, auch „Lokalität“ in der Darstellung der Funktion g durch überabzählbar viele Basisfunktionen zu erhalten. Bei der Fourier-Transformation erhält man die Basisfunktionen $e_{\alpha}(x) = e^{i\alpha x}$ durch Variation einer Grundfunktion

$$\phi : \mathbb{R} \rightarrow \mathbb{C}, \quad x \mapsto e^{ix}$$

vermöge

$$e_{\alpha} : \mathbb{R} \rightarrow \mathbb{C}, \quad x \mapsto \phi(\alpha x).$$

Bei der Wavelet-Transformation geht man völlig analog von einem Mutter-Wavelet

$$\psi : \mathbb{R} \rightarrow \mathbb{C}$$

aus, für das im Prinzip die folgenden Eigenschaften gelten sollen:

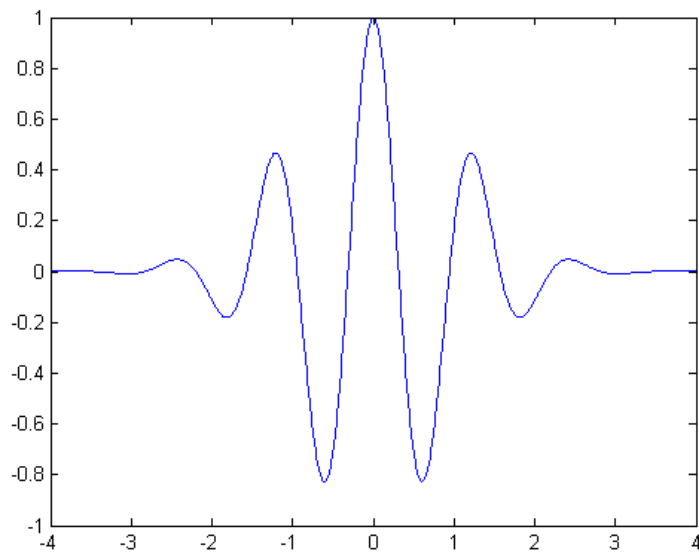
- $\int_{-\infty}^{\infty} |x| |\psi(x)| dx < \infty$
- $\int_{-\infty}^{\infty} \psi(x) dx = 0$

- $\|\psi\| = 1$, also: $\int_{-\infty}^{\infty} \psi(x)\overline{\psi(x)}dx = 1$.

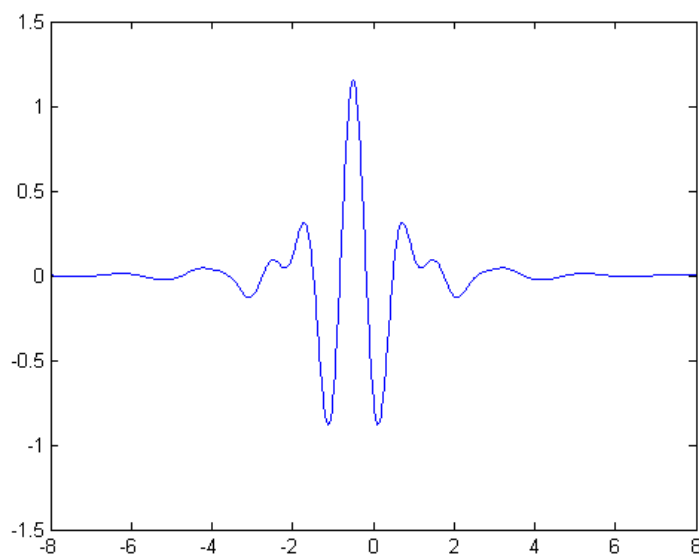
Diese Eigenschaften implizieren, dass ψ asymptotisch gegen Null konvergiert bzw. kompakten Träger hat.

▷ **Beispiele:**

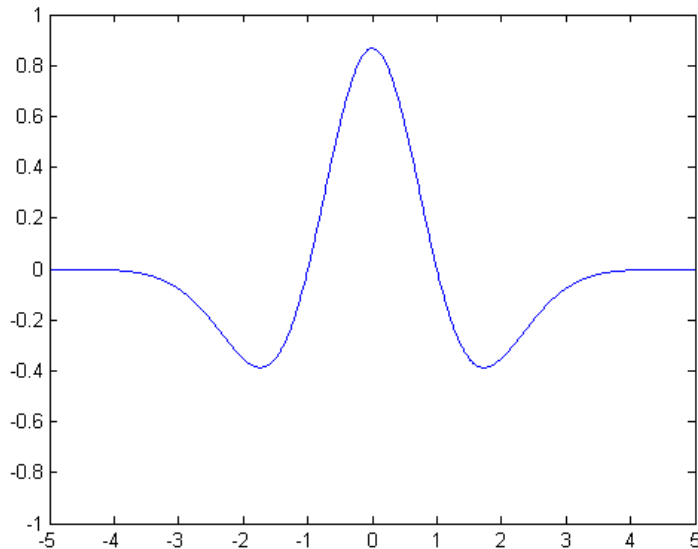
- Morlet-Wavelet:



- Meyer-Wavelet:



- Mexikanerhut-Wavelet:



Ausgehend von einem Mutter-Wavelet der obigen Form erhält man nun die Menge der Basisfunktionen durch

$$\left\{ \frac{1}{\sqrt{|a|}} \psi \left(\frac{x-b}{a} \right) =: \psi_{a,b}; a \in \mathbb{R} \setminus \{0\}, b \in \mathbb{R} \right\}.$$

Der Parameter a steuert die Größe des Intervalls, in dem das Wavelet

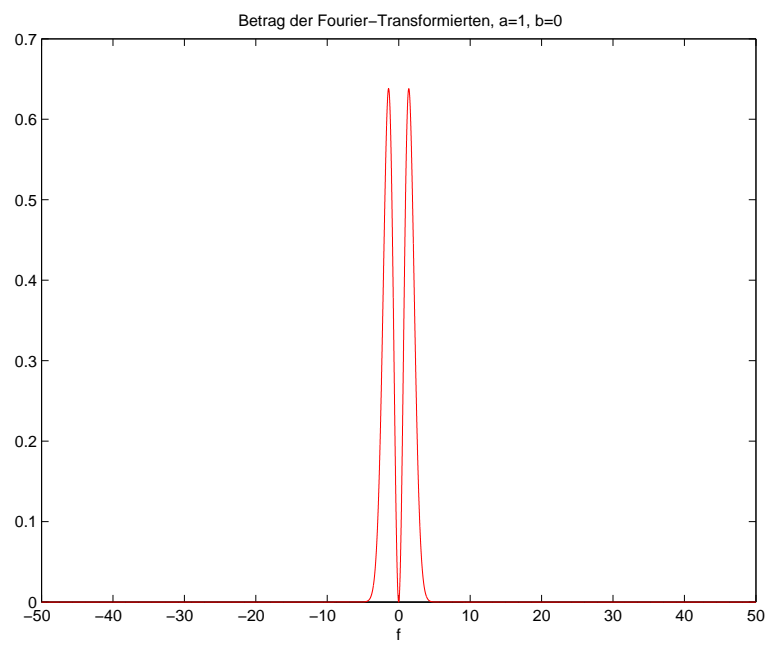
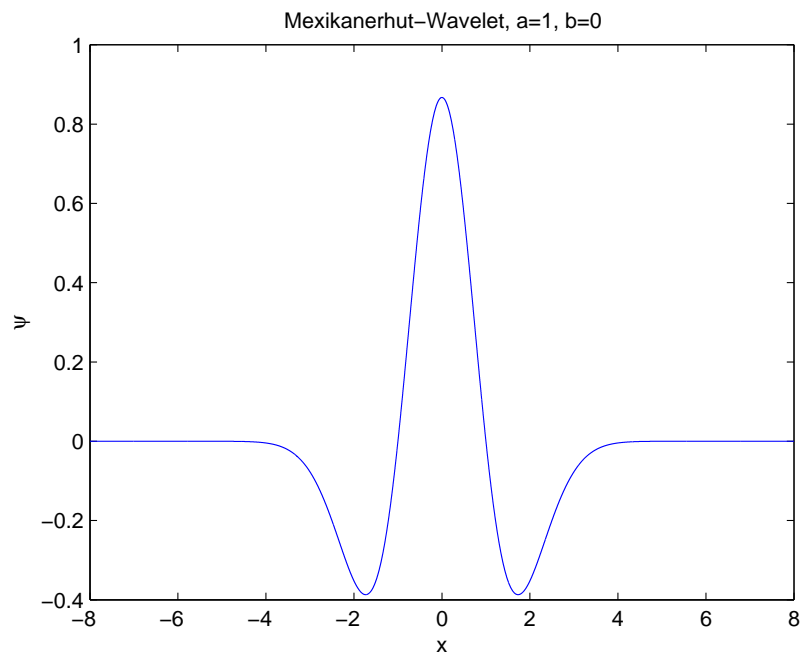
$$\frac{1}{\sqrt{|a|}} \psi \left(\frac{x-b}{a} \right)$$

relevante Funktionswerte ungleich Null besitzt. Der Parameter b verschiebt dieses Intervall auf der x -Achse. Der Skalierungsfaktor

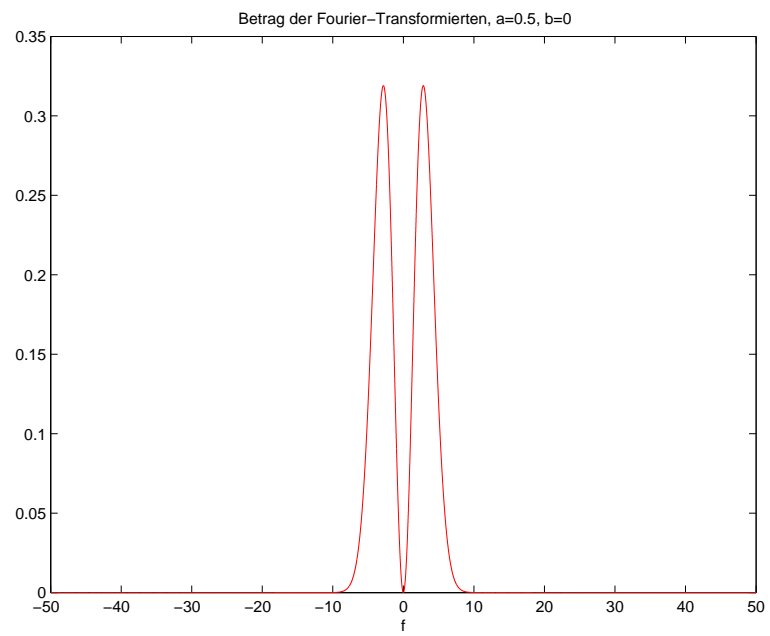
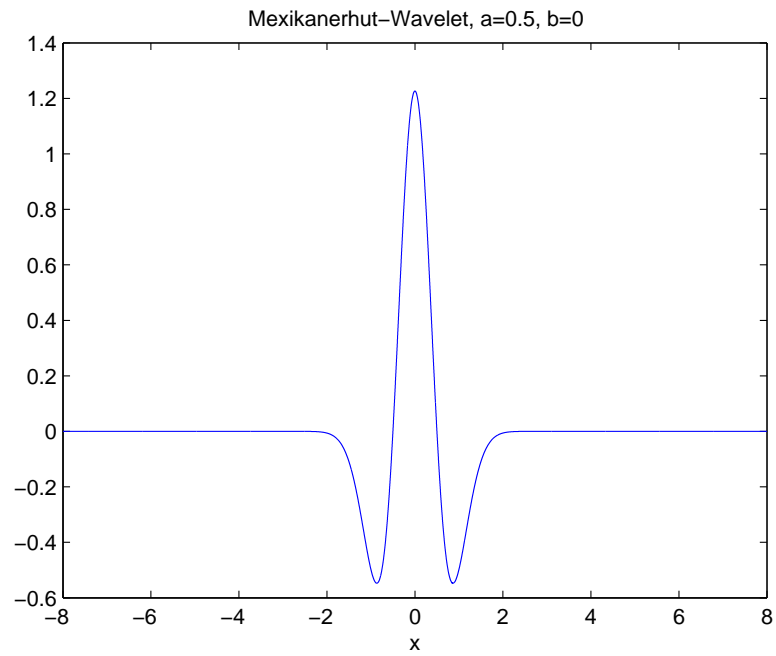
$$\frac{1}{\sqrt{|a|}}$$

garantiert, dass die obigen Bedingungen für das Mutter-Wavelet auch für die Basisfunktionen gelten. Die Wirkung der Parameter a, b soll an einigen Beispielen anhand des Mexikanerhut-Wavelets im Zeit- und im Frequenzbereich (Fourier-Transformierte) erläutert werden:

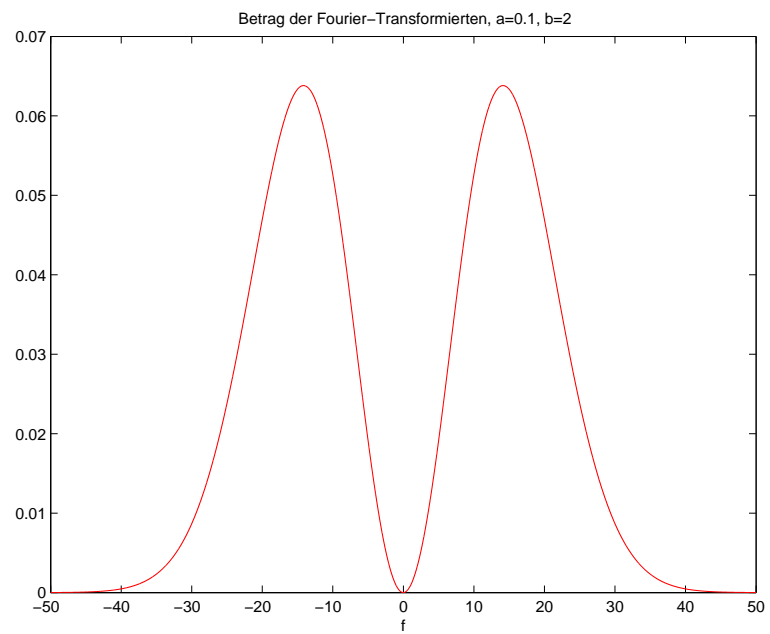
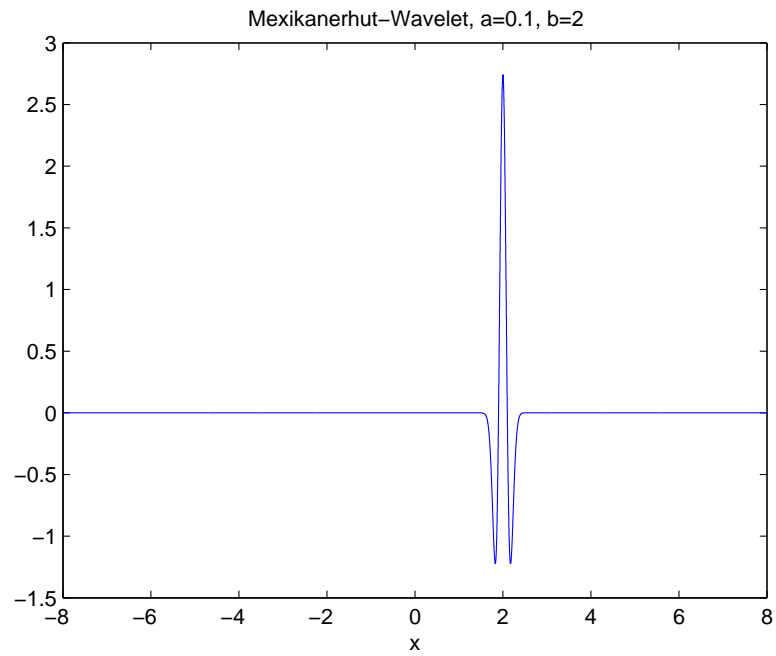
- $a = 1, b = 0$ (Mutter-Wavelet):



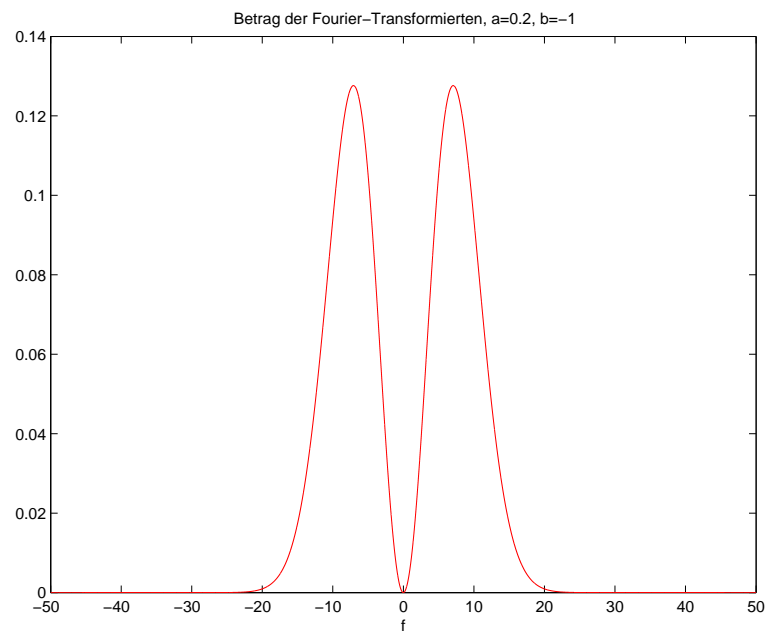
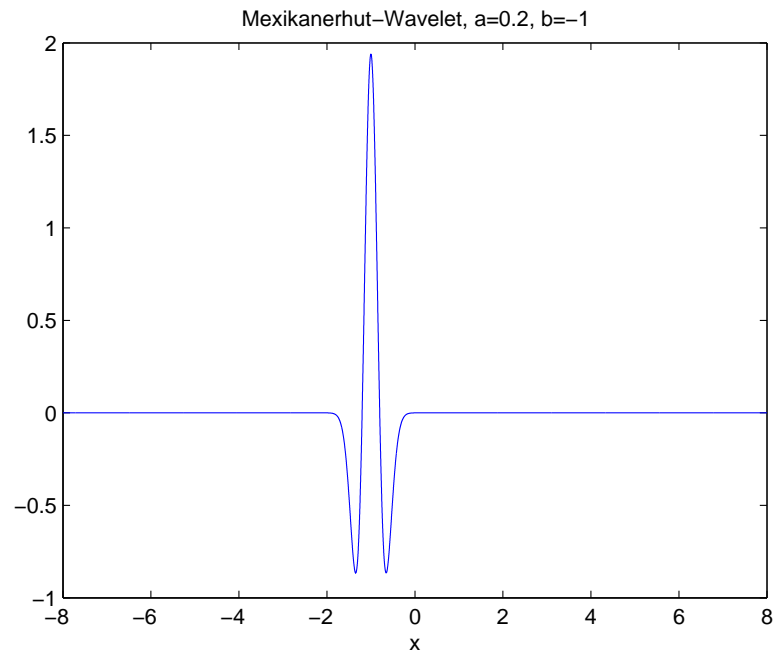
- $a = \frac{1}{2}, b = 0$:



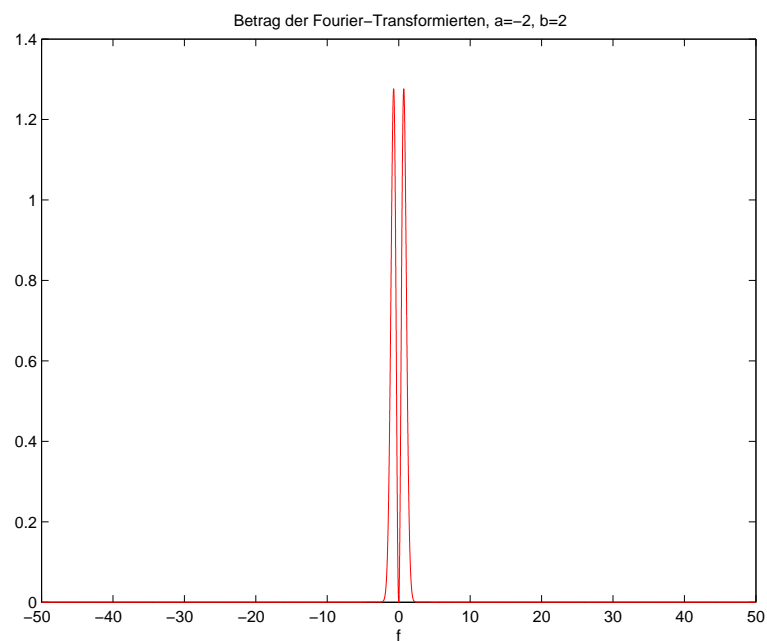
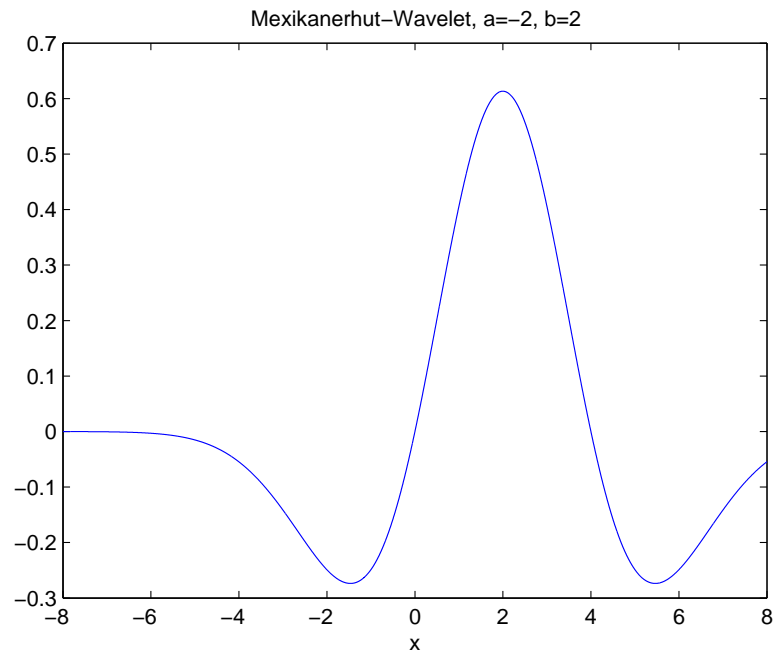
- $a = \frac{1}{10}, b = 2$:



- $a = \frac{1}{5}, b = -1$:



- $a = -2, b = 2$:



Mit den Parametern a, b kann man lokale und globale Eigenschaften von g wesentlich gezielter erfassen - allerdings um den Preis einer doppelt parametrisierten Menge von Basisfunktionen.

Analog zur Fourier-Transformation erhält man nun Formeln für die Wavelet-Transformation und ihre Umkehrung: Zu $g : \mathbb{R} \rightarrow \mathbb{C}$ ergibt sich die Wavelet-

Transformierte

$$\mathcal{W}g : \mathbb{R} \setminus \{0\} \times \mathbb{R} \rightarrow \mathbb{C}, \quad (a, b) \mapsto \int_{-\infty}^{\infty} g(t) \overline{\psi_{a,b}(t)} dt \quad (2.1)$$

und die Umkehrformel (an allen Stetigkeitsstellen von g)

$$g(x) = \frac{1}{C_\psi} \int_{\mathbb{R} \setminus \{0\} \times \mathbb{R}} \frac{1}{a^2} \mathcal{W}g(a, b) \psi_{a,b}(x) da db,$$

wobei C_ψ eine Konstante ist, die nur von ψ abhängt.

Für den weiteren Gebrauch wollen wir an dieser Stelle noch eine Aussage über die Fourier-Transformierte von $\psi_{a,b}$ herleiten. Für die Fouriertransformation gelten die folgenden Beziehungen für Verschiebung bzw. Streckung im Zeitbereich:

$$\begin{aligned} f(x) &\circ\!\!\!\circ\!\!\!\bullet \hat{f}(\xi) \\ f(x-t) &\circ\!\!\!\circ\!\!\!\bullet e^{-i\xi t} \hat{f}(\xi) \\ f\left(\frac{x}{a}\right) &\circ\!\!\!\circ\!\!\!\bullet |a| \hat{f}(a\xi) \end{aligned}$$

daraus lassen sich die folgenden Beziehungen ableiten:

$$\begin{aligned} \psi(x) &\circ\!\!\!\circ\!\!\!\bullet \hat{\psi}(\xi) \\ g(x) := \psi\left(\frac{x}{a}\right) &\circ\!\!\!\circ\!\!\!\bullet |a| \hat{\psi}(a\xi) = \hat{g}(\xi) \\ \psi\left(\frac{x-t}{a}\right) &= g(x-t) \circ\!\!\!\circ\!\!\!\bullet e^{-i\xi t} \hat{g}(\xi) = e^{-i\xi t} |a| \hat{\psi}(a\xi) \end{aligned}$$

und daraus folgt letztendlich:

$$\psi_{a,b} = |a|^{-1/2} \psi\left(\frac{x-t}{a}\right) \circ\!\!\!\circ\!\!\!\bullet e^{-i\xi t} |a|^{1/2} \hat{\psi}(a\xi) \quad (2.2)$$

2.4 Wavelet-Frames

Dieser Abschnitt ist in wesentlichen Teilen direkt oder in angepasster Form aus Kapitel 4 von [8] übernommen. Es soll dabei lediglich ein Abriss der komplexen Materie gegeben werden, der keinen Anspruch auf Vollständigkeit erhebt, sondern nur den roten Faden der Argumentation aufzeigen soll. Für detaillierte Ausführungen und Beweise der theoretischen Grundlagen wird auf [8] verwiesen.

Die kontinuierliche Wavelet-Transformation ermöglicht es jede Funktion aus $L^2(\mathbb{R}^d, \mathbb{R})$ in den Wavelet-Raum abzubilden und zurück. Jedes Wavelet mit Parametern a und t lässt sich als ein Vektor im Wavelet-Raum betrachten. Dieser Raum ist aber in zwei Dimensionen überabzählbar unendlich. Das bedeutet zunächst noch keinen Gewinn für unsere Anwendung. Es stellt sich die Frage, ob

nicht auch eine geringere Anzahl von Wavelets ausreichend wäre um beliebige Funktionen des $L^2(\mathbb{R}, \mathbb{R})$ -Funktionsraumes darzustellen. Die Antwort ist ja. Den entscheidenden Schritt bringen sogenannte Wavelet-Frames.

Aufs knappste zusammengefasst: Ein Frame ist eine Kollektion $a. := (a_i | i \in I)$ von Vektoren eines Hilbertraums X , die so reichhaltig ist, dass kein $x \in X$ auf allen a_i senkrecht steht. Im unendlichdimensionalen Fall ist das nicht so leicht sicherzustellen. Die a_i brauchen nicht linear unabhängig (geschweige denn orthonormiert) zu sein; in diesem Sinne sind Frames im Allgemeinen redundant. Man könnte auch sagen, ein Frame ist weniger als eine Basis eines Vektorraumes. Man kann zwar jeden Punkt des Vektorraumes als Linearkombination der Framevektoren darstellen, aber im Allgemeinen nicht eindeutig.

Beschränken wir unsere Betrachtungen zunächst auf den eindimensionalen Fall. Sei ψ ein Mutter-Wavelet. Bei der kontinuierlichen Wavelet-Transformation wurden alle Wavelets aus der Menge

$$\left\{ \psi_{a,b} | \psi_{a,b} = a^{-1/2} \psi\left(\frac{x-t}{a}\right) : a \in \mathbb{R} \setminus \{0\}, t \in \mathbb{R} \right\} \quad (2.3)$$

genutzt um eine beliebige Funktion des Funktionsraumes L^2 darzustellen. Jetzt wollen wir uns auf eine Teilmenge davon beschränken indem wir nur noch folgende diskrete Werte für a_m und $t_{m,n}$ zulassen:

$$a_m = \sigma^m \quad \text{und} \quad t_{m,n} = n\sigma^m\beta$$

wobei $\sigma > 1$ und $\beta > 0$ vorgegebene Konstanten sind und $m, n \in \mathbb{Z}$ alle ganzzahligen Werte durchlaufen. Wir erhalten so die abzählbar unendliche Familie

$$\psi. := (\psi_{m,n} | m, n \in \mathbb{Z}) \quad (2.4)$$

wobei

$$\psi_{m,n}(x) := \sigma^{-m/2} \psi\left(\frac{x - n\sigma^m\beta}{\sigma^m}\right) = \sigma^{-m/2} \psi(\sigma^{-m}x - n\beta)$$

Man kann nun sagen die Familie $\psi.$ spannt einen Vektorraum von Funktionen auf. Das beeindruckende daran ist, dass unter gewissen Umständen dieser Vektorraum dicht im L^2 Funktionsraum liegt bezüglich der L^2 -Norm. Mit anderen Worten: $\psi.$ ist dann ein Frame des L^2 -Funktionsraumes oder zumindest ein Frame eines Funktionsraumes, der dicht in L^2 liegt. Beweisen lässt sich dies mit Hilfe der Frame Theorie, die hier aber nicht im Detail behandelt werden soll. Wir beschränken uns auf die folgende Definition und den folgenden Satz unter Verweis auf [8].

Definition 2.7

Sei der Frameoperator $T : f \mapsto Tf$ definiert über die Formel

$$\begin{aligned} Tf(m, n) &:= \langle f, \psi_{m,n} \rangle = \mathcal{W}f(a_m, t_{m,n}) \quad ((m, n) \in \mathbb{Z}^2) \\ &= \int_{-\infty}^{\infty} f(x) \overline{\psi_{a,b}(x)} dx \end{aligned}$$

Der folgende Satz ist ein aus Frametheorie und diskreter Wavelet-Transformation abgeleiteter Spezialfall. Er ist hier von zentraler Bedeutung:

Satz 2.8

Die Waveletfamilie ψ . ist genau dann ein Frame des L^2 , wenn es Konstanten $0 < A \leq B$ gibt, sodass

$$A \|f\|^2 \leq \|Tf\|^2 \leq B \|f\|^2 \quad \forall f \in L^2$$

dabei bezeichnet $\|f\|$ die $L^2(\mathbb{R})$ -Norm und $\|Tf\|$ die $l^2(\mathbb{Z}^2)$ -Norm.

Dieser Satz liefert bei einer Waveletfamilie ψ . eine Handhabe für den Beweis der Frameeigenschaft. Genau darauf werden wir im folgenden hinarbeiten.

Sei $\hat{\psi} : \mathbb{R} \rightarrow \mathbb{C}$ die Fouriertransformierte von ψ , also

$$\hat{\psi}(\alpha) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \psi(x) e^{-i\alpha x} dx \tag{2.5}$$

Der Zoomschritt $\sigma > 1$ sei gegeben. Ein Mutter-Wavelet ψ heißt für die Zwecke der nachfolgenden Betrachtungen zulässig, wenn die Fourier-Transformierte $\hat{\psi}$ die nachstehenden Bedingungen (a) und (b) erfüllt.

(a) Es gibt Konstanten $\alpha > 0, \rho > 0$ und C mit

$$|\hat{\psi}(\xi)| \leq \begin{cases} C|\xi|^\alpha & (|\xi| \leq 1) \\ \frac{C}{|\xi|^{1+2\rho}} & (|\xi| \geq 1) \end{cases} \tag{2.6}$$

Diese Bedingung ist an sich harmlos und dient in erster Linie zur Einführung der Konstanten α, ρ und C . Ist zum Beispiel $t\psi \in L^1$ und ψ' von beschränkter Variation, so gelten Abschätzungen der Form (2.6) mit $\alpha = 1$ und $\rho = \frac{1}{2}$.

(b) Es gibt eine Konstante $A' > 0$ mit

$$\sum_{m=-\infty}^{\infty} |\hat{\psi}(\sigma^m \xi)|^2 \geq A' \quad (1 \leq |\xi| \leq \sigma) \quad (2.7)$$

Weil die linke Seite von (2.7) gegenüber der Streckung $\xi \mapsto \sigma\xi$ invariant ist, kann man sich auf den Prüfbereich $1 \leq |\xi| \leq \sigma$ beschränken. Die Bedingung drückt aus, dass die Nullstellen von $\hat{\psi}$ nicht „logarithmisch konspirieren“ dürfen. Insbesondere darf der Träger von $\hat{\psi}$ nicht in einem einzigen Intervall $]b, \sigma b[$ Platz haben.

Die Konstanten α, ρ, C und A wollen wir im folgenden als Parameter von ψ bezeichnen. Wir kommen nun zum zweiten zentralen Satz dieses Abschnitts, der aus [8], Satz 4.10 übernommen ist.

Satz 2.9

Der Zoomschritt $\sigma > 1$ sei vorgegeben. Es sei ψ ein zulässiges Wavelet mit Parametern α, ρ, C und A . Dann gibt es Konstanten β_0, B' und C' , so dass folgendes zutrifft: Ist der Grundschrift $\beta < \beta_0$, so ist die Familie $\psi_{\cdot} = (\psi_{m,n} | (m, n) \in \mathbb{Z}^2)$ ein Frame mit Frame-Konstanten

$$A = \frac{2\pi}{\beta} (A' - C' \beta^{1+\rho}), \quad B = \frac{2\pi}{\beta} (B' + C' \beta^{1+\rho})$$

Damit ist gezeigt, dass es theoretisch möglich ist, jede Funktion des $L^2(\mathbb{R}, \mathbb{R})$ mit einer abzählbar unendlichen Familie ψ_{\cdot} beliebig genau zu approximieren. Man kann sogar noch einen Schritt weiter gehen und sagen, dass sich jede Funktion des $L^2(\mathbb{R}, \mathbb{R})$ beliebig genau (im Sinne der L^2 -Norm) als Linearkombination von endlich vielen Funktionen aus ψ_{\cdot} darstellen lässt. Im nächsten Abschnitt werden wir einen konkreten Fall betrachten.

2.5 Ermittlung der Frameeigenschaft für einen konkreten Fall

Wir betrachten folgende konkrete Wavelet-Familie ψ_{\cdot} wie in (2.4) mit Mexikanerhut Mutter-Wavelet

$$\psi : \mathbb{R} \rightarrow \mathbb{R} \quad \text{mit} \quad \psi(x) = \frac{2}{\sqrt{3}} \pi^{-1/4} (1 - x^2) e^{-x^2/2}$$

und Konstanten:

$$\sigma = 2 \quad , \quad \beta = 2$$

Die zu ψ gehörige Fouriertransformierte lautet:

$$\hat{\psi} : \mathbb{R} \rightarrow \mathbb{R} \quad \text{mit} \quad \hat{\psi}(\xi) = \frac{2}{\sqrt{3}} \pi^{-1/4} \xi^2 e^{-\xi^2/2} \quad (2.8)$$

Für die Herleitung wird auf [8] verwiesen.

Satz 2.9 trifft keine Aussage über die Größe von β_0 . Es bleibt uns hier nichts anderes übrig als tief in den Beweis von Satz 2.9 einzusteigen um zu einer Aussage zu unserem konkreten Fall zu gelangen. Um das Ergebnis dieses (sehr technischen) Abschnittes vorweg zu nehmen: Ja, ψ ist ein Frame des $L^2(\mathbb{R})$. Folglich ist $\beta_0 > 2$. Das ist ein für die Praxis brauchbarer Wert wie wir in den späteren Kapiteln noch sehen werden. Wäre z.B. $\beta_0 = 0,01$, dann hätte die Framemeigenschaft von Waveletfamilien allenfalls theoretischen Charme, aber keinen praktischen Nutzen für die Verwendung bei Wavelet-Netzwerken. Wir stützen uns bei den folgenden Ausführungen im wesentlichen auf [8], Abschnitt 4.4 .

Wir stehen vor der Aufgabe den Term

$$\|Tf\|^2 = \sum_{m,n} |Tf(m,n)|^2 = \sum_{m,n} |Wf(\sigma^m, n\sigma^m\beta)|^2$$

möglichst genau abzuschätzen. Dazu benötigen wir folgende grundlegenden Ergebnisse aus der Fourieranalysis:

Satz 2.10

Seien $f, g : \mathbb{R} \rightarrow \mathbb{C}$ zwei Funktionen mit Fouriertransformierten \hat{f} und \hat{g} analog zu (2.5). Dann gilt:

$$\begin{aligned} \langle f, g \rangle &= \langle \hat{f}, \hat{g} \rangle \\ \Leftrightarrow \int_{-\infty}^{\infty} f(x)\overline{g(x)}dx &= \int_{-\infty}^{\infty} \hat{f}(\xi)\overline{\hat{g}(\xi)}d\xi \end{aligned}$$

Auch der folgende Satz über Fourierreihen soll hier nur zitiert werden.

Satz 2.11

Es sei $f : \mathbb{R} \rightarrow \mathbb{C}$ eine periodische Funktion mit Periode $L > 0$, und es sei $\int_0^L |f(x)|^2 dx < \infty$. Dann gilt

$$f(x) = \sum_{k=-\infty}^{\infty} c_k e^{2k\pi ix/L}, \quad \hat{f}(k) := c_k := \frac{1}{L} \int_0^L f(x) e^{-2k\pi ix/L} dx$$

und es ist

$$\sum_{k=-\infty}^{\infty} |c_k|^2 = \frac{1}{L} \int_0^L |f(x)|^2 dx$$

Der eigentliche Beweis von Satz 2.9 beginnt hier mit:

$$\begin{aligned}\mathcal{W}f(a, t) &= \int f(x) \overline{\psi_{a,t}(x)} dx \\ &= |a|^{1/2} \int \hat{f}(\xi) \overline{\hat{\psi}(a\xi)} e^{ib\xi} d\xi\end{aligned}$$

Dabei gilt die erste Gleichheit nach (2.1), die zweite nach (2.2) und Satz 2.10. Wir setzen zur Abkürzung:

$$g(\xi) := \hat{f}(\xi) \overline{\hat{\psi}(a\xi)} \quad (2.9)$$

Damit ergibt sich, $t \neq 0$ vorausgesetzt:

$$\mathcal{W}f(a, nt) = |a|^{1/2} \int_0^{2\pi/t} e^{int\xi} \sum_l g\left(\xi + l\frac{2\pi}{t}\right) d\xi \quad (2.10)$$

Die Funktion

$$G(\xi) := \sum_l g\left(\xi + l\frac{2\pi}{t}\right) \quad (2.11)$$

ist periodisch mit Periode $\frac{2\pi}{t}$. Nach den Formeln aus Satz 2.11 können wir daher (2.10) interpretieren als

$$\mathcal{W}f(a, nt) = |a|^{1/2} \cdot \frac{2\pi}{t} \hat{G}(-n)$$

und durch summieren über n ergibt sich

$$\sum_n |\mathcal{W}f(a, nt)|^2 = |a| \left(\frac{2\pi}{t}\right)^2 \sum_n |\hat{G}(n)|^2 = |a| \frac{2\pi}{t} \int_0^{2\pi/t} |G(\xi)|^2 d\xi \quad (2.12)$$

Hier wurde zuletzt die in Satz 2.11 angeführte sogenannte Parsevalsche Formel für Periodenlänge $\frac{2\pi}{t}$ benützt.

Wir untersuchen nun das letzte Integral genauer:

$$\begin{aligned}\int_0^{2\pi/t} |G(\xi)|^2 d\xi &= \int_0^{2\pi/t} \sum_{k,l} g\left(\xi + l\frac{2\pi}{t}\right) \overline{g\left(\xi + k\frac{2\pi}{t}\right)} d\xi \\ &= \sum_{k,l} \int_0^{2\pi/t} g\left(\xi + l\frac{2\pi}{t}\right) \overline{g\left(\xi + k\frac{2\pi}{t}\right)} d\xi\end{aligned}$$

Hier substituieren wir $\xi' := \xi + l\frac{2\pi}{t}$ und erhalten weiter

$$\begin{aligned}\int_0^{2\pi/t} |G(\xi)|^2 d\xi &= \sum_{k,l} \int_{2l\pi/t}^{2(l+1)\pi/t} g(\xi') \overline{g\left(\xi' + (k-l)\frac{2\pi}{t}\right)} d\xi' \\ &= \sum_k \int g(\xi) \overline{g\left(\xi + k\frac{2\pi}{t}\right)} d\xi\end{aligned}$$

Tragen wir das in (2.12) ein, so können wir

$$\sum_n |\mathcal{W}f(a, nt)|^2 = \frac{2\pi|a|}{t} \sum_k \int g(\xi) \overline{g\left(\xi + k\frac{2\pi}{t}\right)} d\xi$$

notieren. Wir setzen hier

$$a := \sigma^m, \quad t := \sigma^m \beta \quad (m \in \mathbb{Z})$$

und summieren auch noch über m . Es ergibt sich

$$\|Tf\|^2 = \sum_{m,n} |\mathcal{W}f(\sigma^m, n\sigma^m\beta)|^2 = \frac{2\pi}{\beta} \sum_{k,m} Q_{km} \quad (2.13)$$

wobei die Q_{km} nach Definition (2.9) von g folgendermaßen aussehen:

$$Q_{km} := \int \hat{f}(\xi) \overline{\hat{\psi}(\sigma^m\xi)} \overline{\hat{f}\left(\xi + k\frac{2\pi}{\sigma^m\beta}\right)} \hat{\psi}\left(\sigma^m\xi + k\frac{2\pi}{\beta}\right) d\xi$$

Es wird sich herausstellen, dass in (2.13) die Terme mit $k = 0$ den Löwenanteil ausmachen. Wir schreiben daher

$$\|Tf\|^2 = \frac{2\pi}{\beta} \left(\int |\hat{f}(\xi)|^2 \sum_m |\hat{\psi}(\sigma^m\xi)|^2 d\xi + Q \right) \quad (2.14)$$

wobei die sämtlichen Terme Q_{km} mit $k \neq 0$ im Rest Q zusammengefasst sind. Um den Hauptteil und den Rest gegeneinander ausspielen zu können, benötigen wir das folgende Lemma:

Lemma 2.12

Es sei ψ ein zulässiges Wavelet mit Parametern α, ρ, C und A' . Dann gibt es erstens ein B' mit

$$\sum_m |\hat{\psi}(\sigma^m\xi)|^2 \leq B' \quad \forall \xi \in \mathbb{R}$$

und zweitens gilt

$$|Q| \leq C' \beta^{1+\rho} \|f\|^2 \quad (2.15)$$

mit einem C' , das unabhängig von β ist.

Nach Definition von A' haben wir dann

$$\frac{2\pi}{\beta} (A' - C' \beta^{1+\rho}) \|f\|^2 \leq \|Tf\|^2 \leq \frac{2\pi}{\beta} (B' + C' \beta^{1+\rho}) \|f\|^2$$

wie behauptet. Damit ist Satz 2.9, modulo das Lemma, bewiesen.

Für unser eigentliches Ziel, nämlich die Berechnung der Frameeigenschaft für das konkrete Problem des Mexikanerhut-Wavelets wie am Anfang des Kapitels beschrieben, ist es unumgänglich auch noch in den Beweis des Lemma 2.12 einzusteigen. Das soll jetzt getan werden.

Um die Summe $\sum_m |\hat{\psi}(\sigma^m \xi)|^2$ nach oben abzuschätzen, müssen wir die Terme mit $m < 0$ und diejenigen mit $m \geq 0$ getrennt behandeln und dabei jeweils die passende Ungleichung für $\hat{\psi}$ verwenden. Es ergibt sich

$$\begin{aligned} \sum_m |\hat{\psi}(\sigma^m \xi)|^2 &\leq \sup_{1 \leq |\xi| \leq \sigma} \sum_m |\hat{\psi}(\sigma^m \xi)|^2 \\ &\leq C^2 \left(\sum_{m < 0} (\sigma^{m+1})^{2\alpha} + \sum_{m \geq 0} \frac{1}{(\sigma^m)^{2(1+2\rho)}} \right) =: B' \end{aligned}$$

Nun zu (2.15). Wir fassen Q_{km} als ein Skalarprodukt auf, wobei wir die vorhandenen Faktoren noch passend aufteilen. Nach der Schwarzschen Ungleichung ergibt sich

$$\begin{aligned} |Q_{km}| &\leq \left(\int |\hat{f}(\xi)|^2 |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi + 2k\pi/\beta)| d\xi \right)^{1/2} \\ &\quad \left(\int |\hat{f}(\xi + 2k\pi/(\sigma^m \beta))|^2 |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi + 2k\pi/\beta)| d\xi \right)^{1/2} \end{aligned}$$

Substituieren wir hier im zweiten Faktor $\xi' := \xi + 2k\pi/(\sigma^m \beta)$, so folgt:

$$\begin{aligned} |Q_{km}| &\leq \left(\int |\hat{f}(\xi)|^2 |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi + 2k\pi/\beta)| d\xi \right)^{1/2} \times \\ &\quad \left(\int |\hat{f}(\xi)|^2 |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi - 2k\pi/\beta)| d\xi \right)^{1/2} \end{aligned}$$

Die $|Q_{km}|$ sind nun über alle $k \neq 0$ und alle m zu summieren. Für die innere Summe über m verwenden wir die Schwarzsche Ungleichung in der Form

$$\sum_m (\sqrt{x_m} \cdot \sqrt{y_m}) \leq \sqrt{\sum_m x_m} \cdot \sqrt{\sum_m y_m}$$

Es ergibt sich

$$\begin{aligned} |Q| &\leq \sum_{k \neq 0} \left(\int |\hat{f}(\xi)|^2 \sum_m |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi + 2k\pi/\beta)| d\xi \right)^{1/2} \times \\ &\quad \left(\int |\hat{f}(\xi)|^2 \sum_m |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi - 2k\pi/\beta)| d\xi \right)^{1/2} \end{aligned} \quad (2.16)$$

Die Summen über m werden nun nach oben abgeschätzt mit Hilfe der Funktion

$$q(s) := \sup_{\xi} \sum_m |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi + s)| \quad (2.17)$$

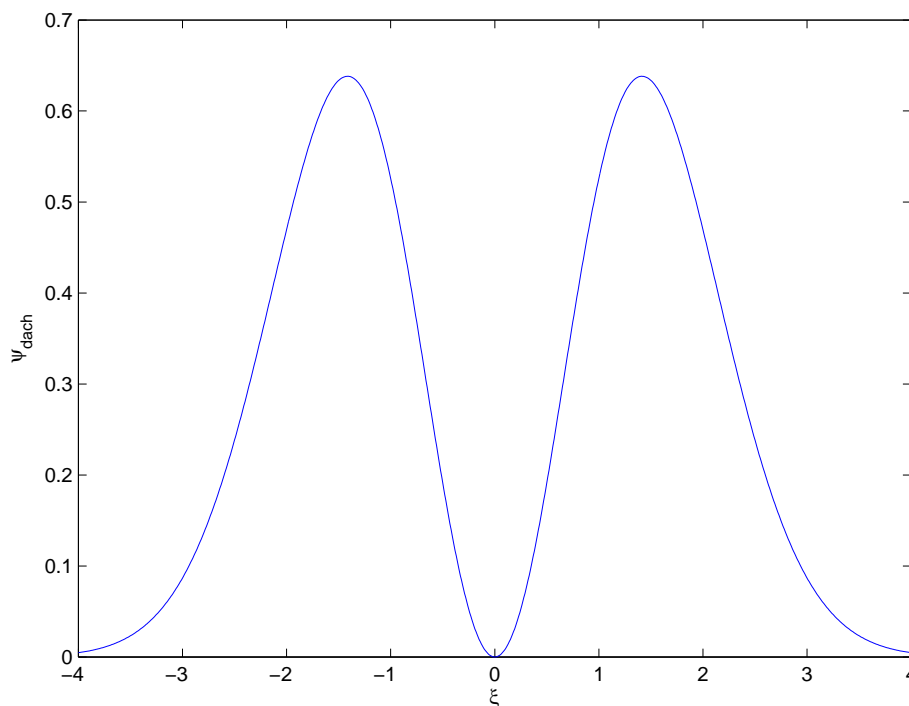


Abbildung 1: Fouriertransformierte des Mutterwavelets $\hat{\psi}(\xi)$

Dabei würde es an sich genügen, das Supremum über $1 \leq |\xi| \leq \sigma$ zu erstrecken. Die Ungleichung (2.16) geht damit über in

$$|Q| \leq \|f\|^2 \sum_{k \neq 0} \sqrt{q(2k\pi/\beta)q(-2k\pi/\beta)} \quad (2.18)$$

An dieser Stelle brechen wir den Beweis des Lemma 2.12 ab. Die bis hierhin geleistete Vorarbeit genügt um nun mit dem konkreten Problem zu beginnen. Die Zulässigkeitsbedingung (2.6) ist für das Mexikanerhut-Wavelet ψ mit Fouriertransformierter $\hat{\psi}(\xi)$ wie in (2.8) trivialer Weise erfüllt. In Abbildung 1 ist die Fouriertransformierte $\hat{\psi}$ zu sehen. Sie ist stets reell und positiv. Für kleine Werte von $|\xi|$ geht sie asymptotisch gegen die Funktion $h(x) = \gamma x^2$ mit $\gamma = \frac{2}{\sqrt{3}}\pi^{-1/4}$. Für große Werte von $|\xi|$ dominiert der Term $e^{-x^2/2}$, d.h. die Funktion fällt sehr schnell zu Null ab. Weiterhin lässt sich elementar-analytisch berechnen, dass $\hat{\psi}$ bei $\pm\sqrt{2}$ Maxima vom Wert

$$\hat{\psi}_{max} = \hat{\psi}(\sqrt{2}) = \gamma \cdot 2e^{-1} \approx 0,63814 \quad (2.19)$$

annimmt. Wir wollen nun im Folgenden Konstanten A' und B' möglichst scharf abschätzen, mit denen gilt:

$$A' \leq \sum_{m=-\infty}^{\infty} |\hat{\psi}(\sigma^m \xi)|^2 \leq B' \quad (1 \leq |\xi| \leq \sigma)$$

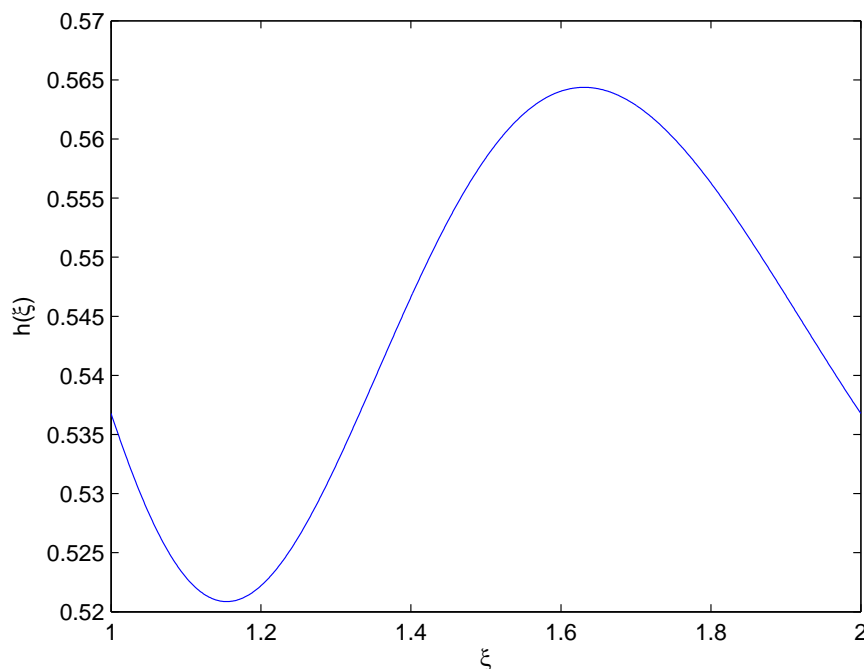


Abbildung 2: Graph der Hilfsfunktion $h(\xi)$

A' wird für die Zulässigkeitsbedingung (b) in (2.7) benötigt. Dabei verwenden wir Matlab als numerisches Hilfsmittel. Zunächst können wir uns überlegen, dass es genügt, den Laufindex m von -100 bis 10 laufen zu lassen. Eine grobschlüchtige Abschätzung bestätigt, dass wir dadurch an der unteren Grenze einen Fehler von weitaus weniger als 2^{-100} in Kauf nehmen und an der oberen Grenze weitaus weniger als e^{-1000} . Wir können diese Fehler folglich vernachlässigen. Die Funktion

$$h(\xi) = \sum_{m=-100}^{10} |\hat{\psi}(\sigma^m \xi)|^2$$

wurde für unser konkretes Problem an 1000 äquidistanten Stellen im Intervall $[1, 2]$ ausgewertet. Der Graph von $h(\xi)$ ist in Abbildung 2 zu sehen. Am Graphen erkennt man bereits, dass das Auffinden des Minimums A' und des Maximums B' ein gut konditioniertes Problem ist. Rundet man die Funktionswerte von $h(\xi)$ auf vier Nachkommastellen, so erhält man an 21 nebeneinanderliegenden ξ -Stützwerten den kleinsten Funktionswert von $A' = 0,5209$ und an 15 nebeneinanderliegenden ξ -Stützstellen den größten Funktionswert $B' = 0,5644$. Es kann folglich von einer Genauigkeit auf vier Nachkommastellen ausgegangen werden. Mit diesen Konstanten A' und B' folgt aus (2.14), dass

$$\frac{2\pi}{\beta} \left(A' \int |\hat{f}(\xi)|^2 d\xi + Q \right) \leq \|Tf\|^2 \leq \frac{2\pi}{\beta} \left(B' \int |\hat{f}(\xi)|^2 d\xi + Q \right)$$

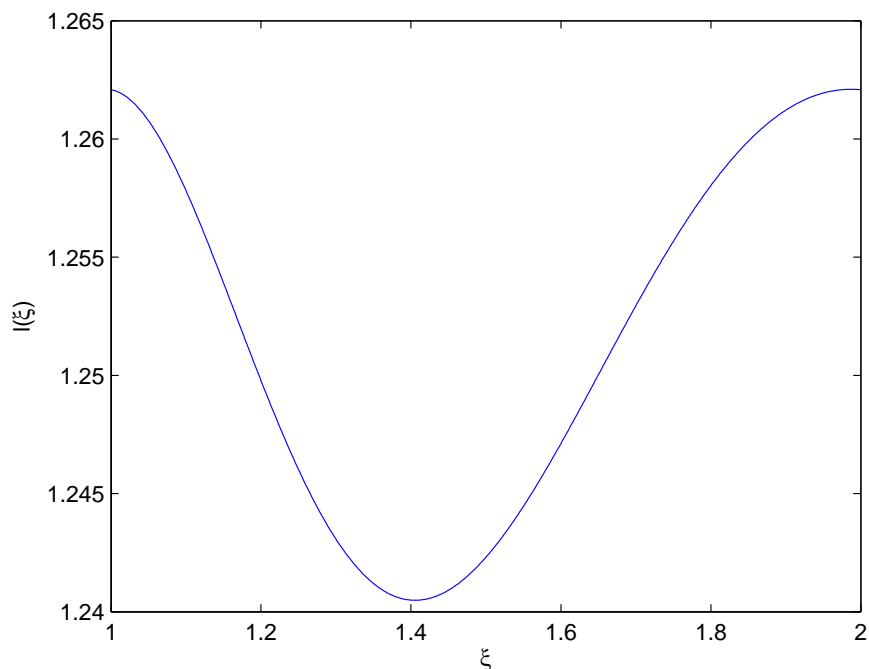


Abbildung 3: Graph der Hilfsfunktion $l(\xi)$

und mit Satz 2.10 erhalten wir

$$\frac{2\pi}{\beta} (\|f\|^2 A' + Q) \leq \|Tf\|^2 \leq \frac{2\pi}{\beta} (\|f\|^2 B' + Q) \quad (2.20)$$

Auf ähnliche Weise wie B' soll nun der Term

$$\sup_{\xi} \sum_{m=-\infty}^{\infty} |\hat{\psi}(\sigma^m \xi)|$$

berechnet werden. Dazu führt man die Funktion $l(\xi)$ ein und sucht

$$l_{max} = \sup_{\xi} l(\xi) \quad \text{mit} \quad l(\xi) = \sum_{m=-100}^{10} |\hat{\psi}(\sigma^m \xi)|$$

Durch grobe Abschätzungen kann hier wiederum gezeigt werden, dass die weggelassenen Summanden einen Einfluss von weniger als ca. 2^{-100} haben und somit völlig zurecht weggelassen wurden. $l(\xi)$ wurde wiederum an 1000 äquidistanten Stellen zwischen 1 und 2 berechnet und in Abbildung 3 dargestellt. Der auf vier Nachkommastellen gerundete maximale Funktionswert $l_{max} = 1,2621$ wurde an 40 nebeneinanderliegenden dieser 1000 Stützstellen angenommen, was uns eine Genauigkeit von l_{max} auf vier Nachkommastellen versichert.

Die bis hierher geleistete Vorarbeit lässt nun die folgenden zwei Abschätzungen für $q(s)$ aus (2.17) zu:

$$\begin{aligned}
 q(s) &= \sup_{\xi} \sum_m |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi + s)| \\
 &\leq \sup_{\xi} \sum_m |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}_{max}| \\
 &= |\hat{\psi}_{max}| \sup_{\xi} \sum_m |\hat{\psi}(\sigma^m \xi)| \\
 &= \hat{\psi}_{max} \cdot l_{max} \\
 &\approx \frac{2}{\sqrt{3}} \pi^{-1/4} \frac{2}{e} \cdot 1,2621 \\
 &\approx q_{max} = 0,8054
 \end{aligned}$$

Diese Abschätzung gilt für alle s ohne Einschränkung. Es kann immer noch von einer Genauigkeit auf 4 Nachkommastellen ausgegangen werden. Wenn wir zusätzlich $s \geq \sqrt{2}$ annehmen, dann können wir noch eine bessere Abschätzung vornehmen, weil $\hat{\psi}(\xi)$ für $\xi \geq \sqrt{2}$ monoton fällt und zwar:

$$\begin{aligned}
 q(s) &= \sup_{\xi} \sum_m |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(\sigma^m \xi + s)| \\
 &\leq \sup_{\xi} \sum_m |\hat{\psi}(\sigma^m \xi)| |\hat{\psi}(s)| \\
 &= |\hat{\psi}(s)| \sup_{\xi} \sum_m |\hat{\psi}(\sigma^m \xi)| \\
 &= \hat{\psi}(s) \cdot l_{max} \\
 &= \hat{\psi}(s) \cdot 1,2621
 \end{aligned}$$

Um nun $|Q|$ in (2.18) abschätzen können, werden wir endlich konkret und führen D' ein mit

$$\begin{aligned}
 D' &= \sum_{k \neq 0} \sqrt{q(2k\pi/\beta)q(-2k\pi/\beta)} \\
 &= 2 \sum_{k=1}^{\infty} \sqrt{q(k\pi)q(-k\pi)} \\
 &\leq 2 \sum_{k=1}^{\infty} \sqrt{\hat{\psi}(k\pi) \cdot l_{max} \sqrt{q_{max}}} \\
 &= 2 \sqrt{l_{max} \sqrt{q_{max}}} \sum_{k=1}^{\infty} \sqrt{\hat{\psi}(k\pi)} \\
 &= 2 \sqrt{l_{max} \sqrt{q_{max}}} \sqrt{\frac{2}{\sqrt{3}} \pi^{-1/4}} \sum_{k=1}^{\infty} (k\pi) e^{-\frac{(k\pi)^2}{4}}
 \end{aligned}$$

$$\begin{aligned}
 &\approx 2\sqrt{l_{max}}\sqrt{q_{max}}\sqrt{\frac{2}{\sqrt{3}}\pi^{-1/4}\sum_{k=1}^2(k\pi)e^{-\frac{(k\pi)^2}{4}}} \\
 &\approx 2\sqrt{1,2621}\sqrt{0,8052}\sqrt{\frac{2}{\sqrt{3}}\pi^{-1/4}\cdot 0,26675} \\
 &= 0,5009
 \end{aligned}$$

Man kann dabei durchaus noch von drei signifikanten Nachkommastellen ausgehen. Wir haben in dieser Rechnung die Abschätzungen für $q(s)$ verwandt. Der Term $e^{-(k\pi)^2/4}$ geht für $k > 2$ so schnell gegen Null, dass für eine Berechnung von vier signifikanten Nachkommastellen in der Summe nur die ersten beiden Glieder berücksichtigt werden müssen. Nun können wir an (2.20) anknüpfen und unsere gewonnenen Ergebnisse verwenden.

$$\begin{aligned}
 &\frac{2\pi}{\beta} (\|f\|^2 A' - |Q|) \leq \|Tf\|^2 \leq \frac{2\pi}{\beta} (\|f\|^2 B' + |Q|) \\
 \Rightarrow &\frac{2\pi}{\beta} (\|f\|^2 A' - \|f\|^2 D') \leq \|Tf\|^2 \leq \frac{2\pi}{\beta} (\|f\|^2 B' + \|f\|^2 D') \\
 \Rightarrow &\|f\|^2 \frac{2\pi}{\beta} (0,5209 - 0,5009) \leq \|Tf\|^2 \leq \|f\|^2 \frac{2\pi}{\beta} (0,5644 + 0,5009) \\
 \Rightarrow &\|f\|^2 \cdot 0,06 \leq \|Tf\|^2 \leq \|f\|^2 \cdot 3,35
 \end{aligned}$$

Damit ist gezeigt, dass die Wavelet-Familie ψ , so, wie sie am Anfang dieses Absatzes definiert wurde, die Frameeigenschaft erfüllt. Dabei wurden teilweise sehr grobe Abschätzungen vorgenommen und wie man sieht, hat es zum Schluss nur ganz knapp funktioniert. Wenn die Wavelets dichter zusammengedrückt werden, also $\beta < 2$ verwendet wird, bleibt die Frameeigenschaft erhalten. Wenn man bei der Abschätzung von D' bei den kleinen Werten von k die $q(s)$ nicht nur abschätzen sondern direkt ausrechnen würde, dann hätte man das Potential auch noch für Werte $\beta > 2$ die Frameeigenschaft nachzuweisen. Die Intuition legt nahe, dass $\beta_0 > 3$ ist. Ein Beweis dafür soll hier nicht geführt werden. Wir haben hier gezeigt, dass das β_0 in Satz 2.9 in unserem konkreten Fall größer als 2 ist und somit in einem „vernünftigen“ Bereich liegt. Dadurch wird nahe gelegt, dass auch bei analog konstruierten mehrdimensionalen Wavelet-Familien bei einer „vernünftigen“ Wahl von β (in dieser Arbeit wurde in allen Problemen $\sigma = 2$ und $\beta = 1$ verwendet) eine Frameeigenschaft gegeben ist. Wir werden in Abschnitt 4.1 auf diese Erkenntnis zurückgreifen.

2.6 Das lineare statistische Modell

Im Zuge der Black-Box Modellierung werden wir Modelle auswählen und deren Parameter anpassen. Bei der Berechnung der Parameter werden wir das lineare statistische Modell nutzen. In diesem Abschnitt orientieren wir uns stark an [10], Kapitel 17, wählen aber teils andere Bezeichner zum Zwecke der Konsistenz dieser Arbeit. Beim linearen statistischen Modell geht man davon aus, dass

beobachtbare Daten linear von gewissen Parametern abhängen. Bei konkreten Messungen treten jedoch zufällige Messfehler auf, sodass die Koeffizienten nicht unmittelbar bestimmt werden können. Lineare statistische Modelle dienen unter anderem dazu, trotz der Störung durch Messfehler o.Ä. auf Grundlage der gemessenen Daten die Koeffizienten und damit den linearen Zusammenhang zu bestimmen. Ausgangspunkt der linearen statistischen Modelle ist ein statistischer Raum

$$(\mathbb{R}^n, \mathcal{B}^n, \mathbb{P}_{\underline{Y}, \theta \in \Theta}).$$

Das Besondere des statistischen Raumes $(\mathbb{R}^n, \mathcal{B}^n, \mathbb{P}_{\underline{Y}, \theta \in \Theta})$ besteht nun in der Tatsache, dass über $\mathbb{P}_{\underline{Y}, \theta \in \Theta}$ nur der Erwartungswert von \underline{Y} in Form einer Linearkombination der Vektoren $\underline{v}_1, \dots, \underline{v}_r$ mit unbekanntem Linearfaktoren $\tilde{\underline{u}} \in \mathbb{R}^r$ und lediglich die Kovarianzmatrix von \underline{Y} in Form eines Vielfachen der Einheitsmatrix mit unbekanntem Koeffizienten $\tilde{\sigma}^2 > 0$ bekannt sind. Daher modelliert man \underline{Y} durch

$$\underline{Y} = \mathbf{V}\tilde{\underline{u}} + \underline{E},$$

mit einer Matrix $\mathbf{V} = (\underline{v}_1, \dots, \underline{v}_r) \in \mathbb{R}^{n,r}$ und einer n -dimensionalen reellen Zufallsvariablen \underline{E} . In diesem linearen Modell parametrisiert $\theta = (\tilde{\underline{u}}, \tilde{\sigma}^2) \in \mathbb{R}^r \times \mathbb{R}_+$ also die Familie der Zufallsvariablen $\underline{Y} = \underline{Y}(\theta)$ und damit nur mittelbar die Familie der induzierten Verteilungen $\mathbb{P}_{\underline{Y}, \theta \in \Theta}$. Wir fassen zusammen

Definition 2.13

Es sei $\mathbf{V} \in \mathbb{R}^{n,r}$ eine Matrix, $\tilde{\underline{u}} \in \mathbb{R}^r$ und $\tilde{\sigma}^2 > 0$. Ferner sei \underline{E} eine n -dimensionale Zufallsvariable mit

$$\mathbb{E}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)}(\underline{E}) = \underline{0} \quad \text{und} \quad \text{Cov}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)}(\underline{E}) = \tilde{\sigma}^2 \mathbf{I}$$

Dann heißt ein statistischer Raum

$$(\mathbb{R}^n, \mathcal{B}^n, \mathbb{P}_{\underline{Y}, \theta \in \Theta}), \quad \theta = (\tilde{\underline{u}}, \tilde{\sigma}^2) \in \mathbb{R}^r \times \mathbb{R}_+ = \Theta,$$

mit

$$\underline{Y} = \mathbf{V}\tilde{\underline{u}} + \underline{E}$$

lineares Modell

Die Matrix $\mathbf{V} \in \mathbb{R}^{n,r}$ wird als Designmatrix bezeichnet und besteht aus den Spalten $\underline{v}_1, \dots, \underline{v}_r \in \mathbb{R}^n$. Diese wiederum heißen Regressoren. Die Forderungen an \underline{E} stellen sicher, dass

$$\mathbb{E}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)}(\underline{Y}) = \mathbf{V}\tilde{\underline{u}} \quad \text{und} \quad \text{Cov}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)}(\underline{Y}) = \tilde{\sigma}^2 \mathbf{I}$$

gilt. Die Komponenten Y_1, \dots, Y_n von \underline{Y} haben also i.A. unterschiedliche Erwartungswerte, sind paarweise unkorreliert und haben die gemeinsame Varianz $\tilde{\sigma}^2$.

2.6.1 Kleinste-Quadrate-Schätzung

Bei linearen statistischen Modellen ist der Verteilungstyp der Zufallsvariablen \underline{Y} mit

$$\underline{Y} = \mathbf{V}\tilde{\underline{u}} + \underline{E}$$

im Allgemeinen nicht bekannt. Daher stützt man sich bei der Schätzung des unbekannt Parameters $\tilde{\underline{u}} \in \mathbb{R}^r$ auf einen stärker heuristischen Zugang. Dabei wählt man folgende Schätzfunktion \underline{u} für $\tilde{\underline{u}}$ unter der Voraussetzung, dass die Designmatrix \mathbf{V} vollen Rang r besitzt. Sei \underline{y} nun eine Realisierung von \underline{Y} und \underline{e} die zugehörige Realisierung von \underline{E} . Dann ist

$$\underline{u} : \mathbb{R}^n \rightarrow \mathbb{R}^r, \quad \underline{y} \mapsto \arg \min_{\underline{u} \in \mathbb{R}^r} \{ \|\underline{e}\|_2^2 \},$$

wobei $\arg \min_{\underline{u} \in \mathbb{R}^r} \{ \|\underline{e}\|_2^2 \}$ den eindeutig bestimmten Vektor $\underline{u}(\underline{y}) \in \mathbb{R}^r$ bezeichnet, für den die Funktion

$$\|\underline{e}(\underline{y})\|_2^2 = \|\underline{y} - \mathbf{V}\underline{u}\|_2^2$$

minimal wird. Durch partielle Differentiation nach den Komponenten von \underline{u} und durch Lösung eines linearen Gleichungssystems erhält man die explizite Form

$$\underline{u} : \mathbb{R}^n \rightarrow \mathbb{R}^r, \quad \underline{y} \mapsto (\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T \underline{y}.$$

somit ist \underline{u} eine messbare Funktion und wegen

$$\mathbb{E}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)}(\underline{u}) = \mathbb{E}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)} [(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T \underline{Y}] = \tilde{\underline{u}}$$

auch erwartungstreu. Ferner ist \underline{u} eine lineare Funktion in \underline{Y} . Damit haben wir gezeigt:

Satz 2.14

Sei $(\mathbb{R}^n, \mathcal{B}^n, \mathbb{P}_{\underline{Y}, \theta \in \mathbb{R}^r \times \mathbb{R}_+})$ mit $\underline{Y} = \mathbf{V}\tilde{\underline{u}} + \underline{E}$ ein lineares Modell.

Hat die Designmatrix \mathbf{V} vollen Rang r , so ist

$$\underline{u} : \mathbb{R}^n \rightarrow \mathbb{R}^r, \quad \underline{y} \mapsto (\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T \underline{y},$$

eine erwartungstreue Schätzfunktion für $\tilde{\underline{u}}$ mit

$$\underline{u}(\underline{y}) = \arg \min_{\underline{u} \in \mathbb{R}^r} \{ \|\underline{e}\|_2^2 \} = \arg \min_{\underline{u} \in \mathbb{R}^r} \{ \|\underline{y} - \mathbf{V}\underline{u}\|_2^2 \}$$

Auf Grund dieser Eigenschaften wird die Schätzfunktion \underline{u} auch als „Kleinste-Quadrate-Schätzfunktion“ bezeichnet.

2.6.2 Varianz-Schätzung

Da nach Voraussetzung die einzelnen Komponenten E_1, \dots, E_n des Fehlers \underline{E} unkorreliert sind und identische Varianz besitzen, bietet sich als Schätzfunktion für $\tilde{\sigma}^2$ die Funktion

$$\bar{S}^2 : \mathbb{R}^n \rightarrow \mathbb{R}_+, \quad \underline{y} \mapsto \frac{1}{n} \sum_{i=1}^n e_i^2 = \frac{1}{n} \|\underline{e}\|_2^2 = \frac{1}{n} \|\underline{y} - \mathbf{V}\tilde{\underline{u}}\|_2^2$$

an. Diese Schätzfunktion hängt aber vom unbekanntem Parameter $\tilde{\underline{u}}$ ab. Daher ersetzen wir $\tilde{\underline{u}}$ durch den Kleinste-Quadrate-Schätzer g aus Satz 2.14 und erhalten so einen Schätzer \hat{S}^2 mit

$$\hat{S}^2 : \mathbb{R}^n \rightarrow \mathbb{R}_+, \quad \underline{y} \mapsto \frac{1}{n} \|\underline{y} - \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T \underline{y}\|_2^2$$

Für $\mathbb{E}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)}(\hat{S}^2)$ gilt:

$$\begin{aligned} \mathbb{E}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)}(\hat{S}^2) &= \frac{1}{n} \mathbb{E}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)} (\underline{Y}^T [\mathbf{I} - \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T] [\mathbf{I} - \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T] \underline{Y}) \\ &= \frac{1}{n} \text{Spur} [\text{Cov}_{(\tilde{\underline{u}}, \tilde{\sigma}^2)} ([\mathbf{I} - \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T] \underline{Y})] \\ &= \frac{1}{n} \text{Spur} (\tilde{\sigma}^2 [\mathbf{I} - \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T]) \\ &= \frac{1}{n} \tilde{\sigma}^2 (n - r) \end{aligned}$$

da $[\mathbf{I} - \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T]$ die lineare Projektion auf $\text{Bild}(\mathbf{V})^\perp$ darstellt. Um eine erwartungstreue Schätzfunktion σ^2 für $\tilde{\sigma}^2$ zu erhalten, modifizieren wir \hat{S}^2 zu

$$\sigma^2 : \mathbb{R}^n \rightarrow \mathbb{R}_+, \quad \underline{y} \mapsto \frac{1}{n - r} \|\underline{y} - \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T \underline{y}\|_2^2 \quad (2.21)$$

Dabei dient σ^2 als Maß für die Güte der Schätzung des Erwartungswertes $\mathbf{V}\tilde{\underline{u}}$ von \underline{Y} durch $\mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T \underline{y}$.

3 Wavelet-Netzwerke

Die Wavelet-Theorie ist ein inzwischen weiterentwickeltes Gebiet der Mathematik, für das es zahlreiche Anwendungen gibt, wie z.B. in der Numerik und Signalverarbeitung. Doch obwohl diese reichhaltige Theorie effiziente Algorithmen zur Analyse, Approximation und Kompression von Funktionen oder Signalen hervorgebracht hat, blieb die Anwendung dieser Algorithmen normalerweise auf Probleme von niedriger Dimension d beschränkt. Mit Dimension ist hier die Dimension des Definitionsbereichs des Signals oder der Funktion gemeint. Typischerweise wurde dabei nur mit $d = 1$ oder $d = 2$ gearbeitet. Grund dafür war, dass die Konstruktion und Verwaltung einer höherdimensionalen Wavelet-Basis, also z.B. einer Basis des $L^2(\mathbb{R}^d, \mathbb{R})$ zu hohem Rechenaufwand mit sich gebracht hätte. Um auch höherdimensionale Probleme bearbeiten zu können, ist es wünschenswert, eine Technik zu finden, deren Komplexität nicht so sehr von der Dimensionalität abhängt. Es scheint, dass (künstliche) Neuronale Netze für solche Zwecke besonders geeignet sind. Mehrere Studien haben sich mit der Approximation von höherdimensionalen Funktionen mit Neuronalen Netzen befasst. Neuronale Netze sind eine besondere Rechenarchitektur, bei der Knoten (Neuronen) in verschiedenen Lagen miteinander verbunden sind. Der Name stammt von den biologischen neuronalen Systemen, welche die Inspiration lieferten. Allerdings funktionieren die heutigen künstlichen Neuronalen Netze etwas anders als die biologischen. Obwohl einige theoretischen Untersuchungen das Potential von Neuronalen Netzen zur universalen Funktionsapproximation bestätigen, so mangelt es doch an einer effizienten konstruktiven Methode um zum einen die Struktur des jeweiligen Netzwerkes zu bestimmen und zum anderen dessen Parameter. Besonders die robuste Modellbildung stellt ein Problem dar. Durch die Ähnlichkeit der diskreten inversen Wavelet Transformation und Neuronalen Netzen mit einer vergrabenen Schicht (hidden layer) kamen als erste Q. Zhang und A. Benveniste [1] auf die Idee die Neuronale Netzwerkstruktur und Wavelets zu verbinden. Es folgten Weiterentwicklungen wie [2],[3]oder [4]. In [1] wurde das Wavelet-Netzwerk eingeführt als eine Klasse von sukzessiv arbeitenden Netzen, die aus Wavelets besteht. Diese Art von Netzen bringt verschiedene Vorteile mit sich. Z.B. stellt die Wavelet-Theorie bessere Theoreme über die allgemeine Approximationsfähigkeit bereit als die von konventionellen Neuronalen Netzen. Zudem liefert die Wavelet-Transformation nützliche Anhaltspunkte für die Konstruktion eines Wavelet-Netzwerkes. Viele Autoren haben unabhängig von einander die Zusammenhänge zwischen der Wavelet Theorie und Neuronalen Netzwerken untersucht. In [12] hat Martin Prescher sich mit verschiedenen Netzwerk-Klassen und deren Gemeinsamkeiten und Unterschieden auseinandergesetzt. In dieser Dissertation gehen wir direkt von den Ergebnissen der Wavelet-Theorie zum Wavelet-Netzwerk, ohne uns tiefer mit Neuronalen Netzwerken zu beschäftigen.

3.1 Die Problemstellung

Wavelet-Netzwerke werden verwendet, um Problemstellungen der folgenden Art zu bearbeiten:

Gegeben ist ein Trainingsdatensatz der Form:

$$\begin{aligned}\mathbb{O}_1^N &= \{(x_1, y_1), \dots, (x_N, y_N)\} \\ x_i &\in \mathbb{R}^d, i = 1 \dots N \\ y_i &\in \mathbb{R}, i = 1 \dots N\end{aligned}$$

Er besteht aus N Trainingspunkten, wobei die x -Werte aus \mathbb{R}^d stammen und die y Werte immer aus \mathbb{R} . Würde man ein analoges Problem betrachten mit y -Werten aus dem \mathbb{R}^k , so könnte das Problem immer in k Trainingsdatensätze der obigen Art zerlegt werden. Ziel ist es nun eine Funktion

$$f : \mathbb{R}^d \rightarrow \mathbb{R}$$

zu finden, welche die Trainingsdaten möglichst gut approximiert und gewisse gewünschte Eigenschaften besitzt. Man betreibt folgende Modellbildung:

Sei $\tilde{f} : \mathbb{R}^d \rightarrow \mathbb{R}$ eine unbekannte Funktion, dann gelte:

$$y_i = \tilde{f}(x_i) + e_i \quad \forall (x_i, y_i) \in \mathbb{O}_1^N \quad (3.22)$$

wobei die x_i entweder als definierte Stützpunkte angesehen werden können oder als Realisierung einer Zufallsvariablen X mit einer bestimmten, aber im Allgemeinen nicht bekannten Verteilung. Die e_i fasst man als Realisierungen einer von X unabhängigen Zufallsvariablen E auf mit $\mathbb{E}(E) = 0$. Auch die y_i muss man dem entsprechend als Realisierungen von Zufallsvariablen auffassen. Die Funktion \tilde{f} wird auch Regressionsfunktion genannt. Dabei gehen wir davon aus, dass sie einem bestimmten Funktionenraum angehört (z.B. den stetigen, quadratisch integrierbaren Funktionen). Die eigentliche Aufgabe besteht nun darin, ausschließlich mit Hilfe der Trainingsdaten \mathbb{O}_1^N eine möglichst gute Näherung f für die Funktion \tilde{f} zu finden. Möglichst gut bedeutet in diesem Fall mit einem möglichst geringen Abstand $\left\| \tilde{f} - f \right\|_p$ in einer L_p -Norm. Da uns zur Lösung des Problems aber nur die Trainingsdaten zur Verfügung stehen, müssen wir auf ein genügsames, zumeist heuristisch motiviertes Gütekriterium zurückgreifen. So ein Kriterium ist zum Beispiel der mittlere quadratische Fehler.

$$MSE = \frac{1}{N} \sum_{k=1}^N [y_k - f(x_k)]^2 \quad (3.23)$$

Dieses Kriterium für die Güte der Approximationsfunktion f wird uns noch konkret in den Standardalgorithmen in Kapitel 4 begleiten. In Kapitel 5 werden wir es etwas erweitern und in Kapitel 6 einen gänzlich neuen Ansatz zur Konstruktion einer Kriteriumsfunction aufzeigen. f soll in dieser Arbeit stets ein Wavelet-Netzwerk sein.

Beispiel 3.1

Für die bessere Vorstellung sollen hier denkbare Anwendungsbeispiele für ein Problem dieser Struktur aufgezeigt werden. Letztere erheben keinen Anspruch auf wissenschaftliche Genauigkeit, sondern dienen nur zur Motivation und Inspiration.

1. Nehmen wir an, wir haben ein technisches System (z.B. eine Gleichstrommaschine), dessen Verhalten wir Black-Box-modellieren wollen, weil es vom normalen Verhalten einer Gleichstrommaschine abweicht. Als Eingangsgrößen nehmen wir den Strom I und die Drehzahl n . Ausgangsgröße ist die Klemmenspannung U . Man würde also Messreihen aufnehmen, wobei man I und n in bestimmten Bereichen nach einem regulären Muster einstellt und dann U misst. Das wäre also ein Beispiel für definierte Stützpunkte x_i . Die e_i können als Messfehler aufgefasst werden.
2. Ein weiteres Beispiel für definierte Stützpunkte ist ein Graustufenbild, wie es in [5] behandelt wird. Ein solches Bild lässt sich immer als abgetastete Funktion des $L^2(\mathbb{R}^2)$ auffassen. Die Stützstellen liegen dabei auf einem regulären Gitter.
3. Probleme, in denen die x_i nur als Realisierung einer Zufallsvariablen X mit irgendeiner (nicht notwendiger Weise bekannten) Verteilung aufgefasst werden können, gewinnt man z.B. aus der Natur. Wenn man vermutet, dass eine (schwer oder nicht messbare) Umweltgröße sich durch funktionalen Zusammenhang aus anderen (leichter messbaren) Umweltgrößen ergibt, so kann man Trainingsdatenpaare (x_i, y_i) durch Messung sowohl der x_i als auch der y_i erhalten. Ein triviales Beispiel wäre ein Barometer. Die x_i sind der gegenwärtige Luftdruck und die y_i sind ein Maß für die Sonneneinstrahlung am darauf folgenden Tag.

3.2 Die Struktur eines Wavelet-Netzwerkes

Sei $\psi : \mathbb{R}^d \rightarrow \mathbb{R}$ ein Mutter-Wavelet und sei $w_i \in \mathbb{R}, a_i \in \mathbb{R}_+^d, t_i \in \mathbb{R}^d$ für $i = 1 \dots n$, dann ist

$$f_{wav}(x) = \sum_{i=1}^n w_i \psi [a_i \star (x - t_i)] \quad (3.24)$$

ein Wavelet-Netzwerk, wobei \star für das komponentenweise Produkt von zwei Vektoren steht. t_i wird auch als Translation und a_i als Dilation der Wavelets bezeichnet. Die w_i sind die Gewichte der einzelnen Wavelets. Mit d bezeichnen wir die Dimension des Problems. Man sieht also ganz deutlich, dass das Wavelet-Netzwerk eine Überlagerung von Wavelets der Dimension d ist. Aber Achtung!

Die hier gewählte Schreibweise weicht in der Rolle der a_i ganz entscheidend von der Schreibweise in Abschnitt 2.4 ab. In der eindimensionalen Wavelet-Familie (2.3) haben wir durch Skalare a geteilt. Das ist die gängige Vorgehensweise in der Wavelet-Theorie. Für mehrdimensionale Wavelet-Netzwerke sind die a_i Vektoren und die multiplikative Schreibweise wird sich in der Folge als günstiger erweisen.

Ein zu (3.24) alternatives Konzept wurde in [1] und [5] verfolgt. Hier lautete der Ansatz:

$$f_{wav}(x) = \sum_{i=1}^n w_i \psi [\mathbf{D}_i \mathbf{R}_i (x - t_i)]$$

wobei die t_i wie in (3.24) die Translation bewirken. Die Diagonalmatrix $\mathbf{D}_i = \text{diag}(a_i)$ hat die gleiche Funktion wie die komponentenweise Multiplikation mit den a_i im oberen Konzept. Allein die Orthonormalmatrix \mathbf{R} bewirkt hier den wesentlichen Unterschied, dass die Wavelets auch gedreht werden können. Das bringt zusätzliche Flexibilität in das Netzwerk. Für Dimensionen höher als 2 ist aber die Erstellung und Anpassung einer solchen Orthonormalmatrix sehr mühsam und aufwändig. Da in dieser Arbeit ein allgemeines, für d Dimensionen konzipiertes Netzwerk umgesetzt werden sollte, wurde auf das erste, einfachere Konzept wie in [2] zurückgegriffen.

Erweiterte Wavelet-Netzwerke

Wavelet-Netzwerke der eben eingeführten Form sind eine Linearkombination aus verschiedenartig gestreckt, gestaucht und verschobenen Versionen eines Mutter-Wavelets ψ . Der Grundgedanke einer Erweiterung ist simpel. Zum eigentlichen Wavelet-Netzwerk wird noch ein affiner Term hinzugefügt, sodass

$$f(x) = f_{wav}(x) + f_{aff}(x), \quad (3.25)$$

wobei f_{wav} (3.24) entspricht und $f_{aff}(x) = a_{aff}^T x + c_{aff}$ ist. Dabei ist $a_{aff} \in \mathbb{R}^d$ und $c_{aff} \in \mathbb{R}$. Die Definition des affinen Terms f_{aff} wurde durch [3] inspiriert. Dort wurde genau genommen ein linearer Term zum Wavelet-Netzwerk hinzugefügt, d.h. den konstanten Anteil c_{aff} haben wir hier eigenmächtig ergänzt. Wir wählen in dieser Arbeit das Konzept des erweiterten Wavelet-Netzwerkes, weil es nur geringe Mehrkosten im Sinne von Rechenaufwand und zusätzlichen Parametern mit sich bringt, unter Umständen aber die Funktionenbeschreibung wesentlich verbessern kann. Zudem ist der affine Anteil f_{aff} was Konditionsbetrachtungen angeht äußerst unbedenklich. Alle Algorithmen, die in dieser Dissertation eingeführt werden, sollen erweiterte Wavelet-Netzwerke im Sinne von (3.25) erzeugen, bzw. anpassen. Wir werden aber bei den Algorithmen sehen, dass die Erweiterung der Netzwerke um einen affinen Term nur wenige zusätzliche Überlegungen erfordert.

3.3 Probleme bei der Konstruktion von Wavelet-Netzwerken

Die Erstellung eines Wavelet-Netzwerkes ist, wie die Erstellung eines Neuronalen Netzes, grundsätzlich ein Fall von parameterloser Optimierung. Parameterlose Optimierung bedeutet mitnichten, dass keine Parameter vorhanden sind. Vielmehr bedeutet es, dass a priori nicht feststeht, wie viele Parameter das Netzwerk letztendlich beinhalten soll. Für die Erstellung eines Wavelet-Netzwerkes muss zunächst einmal ein Mutter-Wavelet ψ ausgewählt werden. Damit werden schon bestimmte Eigenschaften der Funktion f festgelegt. Ist ψ z.B. k -mal stetig differenzierbar, so ist es auch f . Danach muss geklärt werden, welche und wie viele Wavelets in das Netzwerk aufgenommen werden und wie die Wavelets an die Trainingsdaten angepasst werden. Die Anzahl der Wavelets ist direkt korreliert mit der Anzahl der Parameter. Das geht aus (3.24) hervor. Die Anzahl der Parameter wiederum ist ein wichtiger Indikator für die Robustheit des Netzwerkes. Insbesondere darf ein Wavelet-Netzwerk nicht mehr Parameter haben als es Stützpunkte N in \mathbb{O}_1^N gibt, weil das Netzwerk sich sonst beliebig an die Trainingspunkte anpassen kann. Das Problem der richtigen Wahl der Parameteranzahl nennt sich auch das Bias-Varianz Dilemma. Je mehr Parameter das Netzwerk hat um so besser kann es sich an die Trainingsdaten anpassen. Die Gefahr, dass sich das Netzwerk nicht an die eigentliche Regressionsfunktion \tilde{f} anpasst, sondern sich auf mehr oder weniger beliebige Weise an die Daten anpasst, steigt ebenso. Man kann auch sagen: Je mehr Parameter, desto schlechter die Kondition der Modellbildung. In [12] hat sich Martin Prescher ausgiebig mit dem Bias-Varianz Dilemma befassen. Wir werden noch auf dieses Problem zurückkommen. Bei der Anpassung der Wavelets an die Trainingsdaten können ebenfalls Effekte schlechter Kondition auftreten, was wir im nächsten Kapitel gleich sehen werden.

3.4 Implementierungen in dieser Arbeit

Schwerpunkt dieser Dissertation soll es sein, verschiedene Algorithmen in einem C#-Programm umzusetzen, die zu gegebenen Trainingsdatensätzen \mathbb{O}_1^N Wavelet-Netzwerke f erstellen. Drei Kapitel dieser Dissertation widmen sich diesen Algorithmen:

- Kapitel 4: Standardalgorithmen
- Kapitel 5: Verbesserung von Standardalgorithmen
- Kapitel 7: Algorithmen zum neuen Gütekriterium

Dabei sollten alle Algorithmen möglichst allgemein und für grundsätzlich beliebiges d , also für Probleme beliebiger Dimensionalität geschrieben werden. Zudem sollten mehrere Beispielpunkte erstellt werden. Zielsetzung war, die Funktionalität aber auch die Probleme verschiedener implementierter Algorithmen zur Erstellung eines Wavelet-Netzwerkes überprüfen und gegeneinander

vergleichen zu können. Das erforderte es, solche Algorithmen effizient zu implementieren und deren unterschiedliche Ergebnisse so anschaulich wie möglich zu präsentieren. Fast alle Beispiele wurden auf $d = 2$ Dimensionen beschränkt, weil sich Funktionen $f : \mathbb{R}^d \rightarrow \mathbb{R}$ für höhere Dimensionen als 2 im Allgemeinen schlecht veranschaulichen lassen. Als Mutter-Wavelet wurde in allen Fällen die C^∞ -Funktion

$$\psi : \mathbb{R}^d \rightarrow \mathbb{R} \quad \text{mit} \quad \psi(x) = C_d \cdot (d - \|x\|_2^2) \exp\left(-\frac{\|x\|_2^2}{2}\right) \quad (3.26)$$

genutzt. Sie wird auch Mexikanerhut-Wavelet genannt. Der Vorfaktor C_d ist nur von der Dimension d abhängig und ist so gewählt, dass $\|\psi\|_2 = 1$ erreicht wird. Dabei ist hier die $L^2(\mathbb{R}^d, \mathbb{R})$ -Norm gemeint.

3.5 Teilaufgabengliederung

Es ist sinnvoll und zweckmäßig die Erstellung eines Wavelet-Netzwerkes zu einem gegebenen Trainingsdatensatz in 3 Schritte zu unterteilen:

1. Erstellen einer Wavelet-Bibliothek
2. Regressorauswahl
3. Regressoroptimierung

Die einzelnen Schritte umfassen dabei folgende Aufgaben

1. Erstellen einer Wavelet-Bibliothek:

Eine Wavelet-Bibliothek ist eine Menge von verschiedenartig gestreckt bzw. gestaucht und verschobenen Versionen ein- und des selben Mutter-Wavelets ψ . Wir beschränken uns in dieser Arbeit auf das Mexikanerhut-Wavelet aus (3.26). Ein Wavelet ψ_i in der Bibliothek ist dabei eine Funktion

$$\psi_i : \mathbb{R}^d \rightarrow \mathbb{R} \quad \text{mit} \quad \psi_i(x) = w_i \psi[a_i \star (x - t_i)] \quad (3.27)$$

Wie schon in (3.24) steht dabei \star für das komponentenweise Produkt zweier Vektoren. Jedes Wavelet $\psi_i = \psi_{\theta_i}$ ist dabei charakterisiert durch seinen Parametervektor θ_i mit

$$\theta_i = (w_i, a_i^T, t_i^T)^T, \quad (3.28)$$

wobei a_i und t_i jeweils d -dimensionale Spaltenvektoren sind. Die Gewichte w_i spielen in der Wavelet-Bibliothek noch keine entscheidende Rolle. Für die anschließenden Algorithmen kann es aber sinnvoll sein, die Wavelets in der Bibliothek zu normieren. Üblicher Weise wählt man zunächst

$$w_i = \Pi(a_i)^{\frac{1}{2}}$$

Der Ausdruck $\Pi(a_i)$ steht für das Produkt aller Komponenten des Vektors a_i . Damit normiert man jedes Wavelet ψ_i auf $\|\psi_i\|_2 = 1$. Hierbei steht $\|\cdot\|$ für die $L^2(\mathbb{R}^d, \mathbb{R})$ -Norm. Wir werden im Folgenden aber auch andere Normierungen nutzen.

2. Regressorauswahl:

In diesem Schritt geht es darum, aus der nun vorhandenen Wavelet-Bibliothek die am besten geeigneten Wavelets ψ_i auszuwählen. Dabei werden auch die Gewichte w_i der ausgewählten Wavelets angepasst. Die restlichen Parameter der θ_i , also die a_i und t_i werden in diesem Schritt noch nicht angetastet. Es muss im Zuge der Wavelet-Auswahl auch festgelegt werden, wie viele Regressoren oder Wavelets ausgewählt werden. Grundsätzlich besteht die Möglichkeit, Wavelets nacheinander auszuwählen und sukzessive dem Wavelet-Netzwerk hinzuzufügen oder anders herum alle Wavelets dem Netzwerk hinzuzufügen und dann eines nach dem anderen zu entfernen bis eine optimale Zahl erreicht ist. Die Parameter von f_{aff} aus (3.25) werden hierbei auch mit angepasst.

3. Regressoroptimierung:

Im vorherigen Schritt wurde das Wavelet-Netzwerk gewissermaßen grob initialisiert. Es wurden Wavelets ausgewählt und die Gewichte wurden so angepasst, dass die Funktion f aus (3.25) sich möglichst gut an die Trainingspunkte anschmiegt. In diesem Schritt werden jetzt alle Parameter der ausgewählten Wavelets möglichst mit einer quasi-Newton-Methode optimiert. Es eignen sich hierfür ein Gauß-Newton-Algorithmus oder ein Levenberg-Marquardt-Algorithmus bzw. abgewandelte Formen davon.

4 Standardalgorithmen

Zunächst einmal sollen für die eben beschriebenen Teilschritte 1, 2 und 3 Standardalgorithmen implementiert werden um eine Grundlage zu schaffen. Die Algorithmen zu Schritt 1 und 2 sind dabei Implementierungen der Pseudoalgorithmen aus [2]. Teilschritt 3 wurde mit dem Levenberg-Marquardt-Verfahren implementiert. Die grundlegende Funktionsweise dieser Algorithmen soll im Folgenden dargelegt werden.

4.1 Erstellung einer Wavelet-Bibliothek

Die Wavelet Bibliothek W wird als eine Menge von Regressorkandidaten angesehen. Sie sollte eine endliche Anzahl an Wavelets enthalten, die so klein wie möglich ist, um Regressorauswahlalgorithmen effizient anzuwenden und so groß wie nötig, um genügend Flexibilität zur Funktionendarstellung zu gewährleisten. Bei gegebenem Mutter-Wavelet ψ geschieht die Konstruktion von W , indem man eine endliche Untermenge der kontinuierlich parametrisierten Funktionenfamilie

$$\left\{ \left(\prod_{i=1}^d a_i \right)^{\frac{1}{2}} \cdot \psi [a \star (x - t)] : a \in \mathbb{R}_+^d, t \in \mathbb{R}^d \right\}$$

auswählt. Wir orientieren uns hierbei an der Idee des Wavelet-Frames wie er in Abschnitt 2.4 eingeführt wurde, nutzen aber gleich die in (3.24) eingeführte, veränderte Schreibweise. Indem wir

$$a = (\sigma^{-m}, \dots, \sigma^{-m})^T \quad \text{und} \quad t = n\sigma^m\beta$$

setzen, erhalten wir die abzählbar unendliche Familie

$$\psi. := (\psi_{m,n} | m \in \mathbb{Z}, n \in \mathbb{Z}^d)$$

wobei

$$\psi_{m,n}(x) := \sigma^{-dm/2} \psi\left(\frac{x - n\sigma^m\beta}{\sigma^m}\right) = \sigma^{-dm/2} \psi(\sigma^{-m}x - n\beta) \quad (4.29)$$

ist. In Abschnitt 2.4 haben wir für den eindimensionalen Fall festgehalten, dass unter gewissen Voraussetzungen diese Familie ein Frame des $L^2(\mathbb{R}, \mathbb{R})$ bildet. Das bedeutet, dass man prinzipiell jede Funktion dieses Raumes beliebig genau als Linearkombination der Funktionen aus $\psi.$ darstellen kann. In Kapitel 10 von [11] hat Ingrid Daubechies gezeigt, dass eine mehrdimensionale Wavelet-Familie, konstruiert wie in (4.29) mit einem radialsymmetrischen Mutter-Wavelet, bei geeigneter Wahl von σ und β einen Wavelet-Frame bildet. Beispielhaft haben wir in Abschnitt 2.5 die Frameeigenschaft für den konkreten Fall nachgewiesen, dass ψ das eindimensionale Mexikanerhut-Wavelet ist und $\sigma = 2$, $\beta = 2$ sind. Eine weitere Betrachtung dieser theoretischen Frameeigenschaften ist hier nicht lohnenswert. Abzählbar unendlich viele Wavelets sind nämlich zu viele für eine Wavelet Bibliothek.

An dieser Stelle hört die exakte Mathematik auf und die Heuristik beginnt. Man trifft hier die Annahme, dass die zu Grunde liegende Funktion f aus (3.22) eine gewisse Regularität besitzt. Man nimmt also an, dass man ganz kleine Wavelets, also Wavelets mit kleinen Werten für m nicht benötigt. Sehr große Wavelets, also Wavelets mit großem m , die einen größeren Bereich überspannen, als den Bereich der Trainingsdaten, sind auch nicht sinnvoll. Somit können die Werte für m auf eine endliche Teilmenge von \mathbb{Z} begrenzt werden. Man kann sich überlegen: Wenn ψ wie oben ein Frame bildet, dann ist auch für eine beliebige Konstante $c \in \mathbb{R}_+$ und

$$a = (c^{-1}\sigma^{-m}, \dots, c^{-1}\sigma^{-m})^T \quad \text{und} \quad t = cn\sigma^m\beta$$

die abzählbar unendliche Familie

$$\psi.^* := (\psi_{m,n} | m \in \mathbb{Z}, n \in \mathbb{Z}^d)$$

mit

$$\psi_{m,n}^*(x) := c^{-d/2}\sigma^{-dm/2}\psi\left(\frac{x - cn\sigma^m\beta}{c\sigma^m}\right) = c^{-d/2}\sigma^{-dm/2}\psi\left(\frac{\sigma^{-m}}{c}x - n\beta\right) \quad (4.30)$$

ein Frame des $L^2(\mathbb{R}^d, \mathbb{R})$. Die beiden Familien ψ und $\psi.^*$ gehen ja gewissermaßen durch Streckung oder Stauchung ineinander über. Da wir nun diesen zusätzlichen Freiheitsgrad eingeführt haben, können wir festlegen, dass die größten Wavelets in unserer Bibliothek, das heißt die Wavelets mit der größten Dilation σ^m bei $m = 0$ entstehen. Das d -dimensionale Mexikanerhut-Mutterwavelet ist ein rotationssymmetrisches Wavelet mit dem gesamten \mathbb{R}^d als Träger. Allerdings fällt es sehr schnell ab für $\|x\|_2 \rightarrow \infty$. Aus diesem Grund macht es Sinn einen effektiven Radius r_{eff} zu definieren, der für einen effektiven Träger steht. Der effektive Radius wurde hier so gewählt, dass für das Mutter-Wavelet ψ wie in (3.26) gilt

$$r_{eff} = \max_{r \in \mathbb{R}_+} [\psi(re_1) \geq 0.05\psi(0)], \quad (4.31)$$

wobei e_1 ein beliebiger Einheitsvektor ist. Im eindimensionalen Fall ist $r_{eff} \approx 3,23$. Zurück zur Wavelet Bibliothek: Die Konstante c wurde nun so gewählt, dass die größten Wavelets (mit $m = 0$) alle Trainingsdaten mit ihrem effektiven Träger überdecken können. Es wird noch ein Maß für die Ausdehnung der Trainingsdaten benötigt. Alle x -Werte der Trainingsdaten lassen sich stets mit einem Hyperquader umschließen. Die Länge der Diagonale des Hyperquaders sei d_{diag} . Das motivierte folgende Formel für c :

$$c = d_{diag}/r_{eff} \quad (4.32)$$

Wie groß σ und β gewählt wird und bis zu welcher Schicht m_{min} Wavelets verschiedener Dilationen in die Bibliothek aufgenommen werden, muss vom Anwender vorgegeben werden. Standardmäßig wurden im Programm $\sigma = 2$, $\beta = 1$ und $m_{min} = -5$ vorgegeben. Wir sind also jetzt soweit, dass wir uns einschränken auf: $m \in \{m_{min}, \dots, 0\}$. Was uns noch fehlt sind die Translationen n . Dazu folgende Vorgehensweise: Es macht nur Sinn Wavelets aus $\psi.^*$ in die Bibliothek

aufzunehmen, die mit ihrem effektiven Träger auch Trainingsdaten überdecken. Zu diesem Zweck gehen wir Trainingspunkt für Trainingspunkt in \mathbb{O}_1^N durch und fügen in allen Ebenen m jene Wavelets zur Bibliothek hinzu, die mit ihrem effektiven Träger den Trainingspunkt überdecken. Mathematisch geschrieben bedeutet das folgendes: Für ein rotationssymmetrisches Mutter-Wavelet ψ mit effektivem Radius r_{eff} , $\sigma > 1, \beta > 0$, $m_{min} \in \mathbb{Z}$ und der Konstanten c wie in (4.32) wird eine Wavelet Bibliothek der Form

$$W_{wav} = \left\{ \psi_{m,n}^* \in \psi \cdot \left| m_{min} \leq m \leq 0 \wedge \exists (x_i, y_i) \in \mathbb{O}_1^N : \frac{\sigma^{-m}}{c} x_i - n\beta \leq r_{eff} \right. \right\} \quad (4.33)$$

erstellt. Da der praktisch umgesetzte Algorithmus die Trainingspunkte nacheinander durchgeht und die zugehörigen Wavelets der Bibliothek hinzufügt, kommt es sehr häufig vor, dass ein Wavelet, das in irgendeinem Schritt hinzugefügt werden soll, bereits in der Bibliothek vorhanden ist, weil es schon einen anderen Trainingspunkt mit seinem effektiven Träger überdeckt. Das heißt, wenn man ein Wavelet der Bibliothek hinzufügen will, muss man erst in der Bibliothek suchen, ob es vielleicht schon vorhanden ist. Hierzu wurde ein schneller Suchalgorithmus mit Hilfe des Hashing-Verfahrens implementiert. In der Wavelet-Bibliothek liegen zum Schluss $L \in \mathbb{N}$ Wavelets. Diese Bibliothek lässt sich jetzt folgendermaßen schreiben:

$$W_{wav} = \{ \psi_i : \mathbb{R}^d \rightarrow \mathbb{R} \mid \psi_i(x) = w_i \psi [a_i \star (x - t_i)] \quad , i = 1 \dots L \} \quad (4.34)$$

wobei die a_i und t_i stets d -dimensionale Vektoren sind und \star das komponentenweise Produkt zweier Vektoren beschreibt.

Bis hierher sind wir im Wesentlichen [2] gefolgt. Da wir aber ein erweitertes Wavelet-Netzwerk gemäß (3.25) erstellen wollen, müssen wir darüber hinaus den affinen Anteil f_{aff} in (3.25) repräsentieren können. f_{aff} lässt sich auf folgende Art darstellen:

$$\begin{aligned} f_{aff}(x) &= \psi_{0,aff}(x) + \psi_{1,aff}(x) + \dots + \psi_{d,aff}(x) \\ &= w_{0,aff} + w_{1,aff} e_1^T x + \dots + w_{d,aff} e_d^T x \end{aligned}$$

,wobei d die Dimension des Problems ist und e_i , $i = 1 \dots d$ die kanonischen Einheitsvektoren des \mathbb{R}^d bezeichnen. Die $\psi_{i,aff}$ mit $i = 0 \dots d$ bezeichnen wir im Folgenden als affine Terme. Davon inspiriert definieren wir uns die Teilbibliothek der affinen Terme

$$W_{aff} = \{ \psi_{i,aff} \mid \psi_{0,aff}(x) = w_{0,aff} \wedge \psi_{i,aff}(x) = w_{i,aff} e_i^T x \quad \forall i = 1 \dots d \} \quad (4.35)$$

wobei die $w_{i,aff}$ mit $i = 0 \dots d$ zunächst auf 1 gesetzt werden können, wie es bereits bei den Gewichten der Wavelets getan wurde. Wie wir in den kommenden Regressorauswahlalgorithmen sehen werden, ermöglicht diese Darstellung der affinen Terme eine Gleichbehandlung mit den Wavelets. Wir können die Teilbibliotheken also zusammenfassen zu

$$W = W_{wav} \cup W_{aff} \quad (4.36)$$

wobei W_{wav} wie in (4.34) und W_{aff} wie in (4.35) aussehen. Im Folgenden werden die Funktionen in W wieder durchgehend indiziert. Wenn also von einem $\psi_i \in W$ die Rede ist, dann kann sich dahinter sowohl ein Wavelet als auch ein affiner Term verbergen. Wir wollen dann allgemein den Begriff Regressoren verwenden. In den folgenden Algorithmen wird grundsätzlich eine erweiterte Wavelet-Bibliothek in Form von (4.36) betrachtet, wenn nicht ausdrücklich anders verlangt.

4.2 Regressorauswahl

Wie bereits bei der Teilaufgabengliederung in Abschnitt 3.5 beschrieben, erfolgt nach der Erstellung einer Wavelet-Bibliothek eine Regressorauswahl. Hierzu wurden zunächst zwei Standardalgorithmen implementiert

1. Sukzessive Regressorauswahl mit schrittweiser Orthogonalisierung
2. Rückwärtselimination von Regressoren

Beiden Algorithmen liegen einige gemeinsame Gedanken zu Grunde. Unser Ausgangspunkt ist eine erweiterte Wavelet-Bibliothek der Form (4.36). Um die weiteren Berechnungen leichter zu machen, normieren wir die Regressoren um.

$$W_{neu} = \left\{ \psi_{i,neu} : \psi_{i,neu}(x) = u_i \psi_{i,alt}(x), u_i = \left(\sum_{k=1}^N [\psi_{i,alt}(x_k)]^2 \right)^{-\frac{1}{2}}, i = 1 \dots L \right\} \quad (4.37)$$

Das heißt, die Gewichte w_i der Regressoren werden mit den so berechneten u_i multipliziert und das Ergebnis wird wieder in den w_i gespeichert. Wenn ψ_i ein Wavelet ist, bedeutet das konkret:

$$\psi_{i,neu} = w_i \psi [a_i \star (x - t_i)], w_i = \left[\sum_{k=1}^N (\psi [a_i \star (x_k - t_i)])^2 \right]^{-\frac{1}{2}}$$

wobei x_1, \dots, x_N für die x -Werte der Trainingsdaten in \mathbb{O}_1^N stehen. Die Reihenfolge der Nummerierung der ψ_i ist dabei willkürlich. Es gilt nun, die „besten“ Regressoren aus einer endlichen Menge von Regressorkandidaten auszuwählen. Das ist ein klassisches Problem der Mathematik. In unserem Fall sind die Regressoren alle Wavelets und affine Terme aus W . Die Aufgabe besteht nun darin, die „besten“ $M \leq L$ Regressoren aus W , basierend auf den Trainingsdaten \mathbb{O}_1^N auszuwählen, um die Approximationsfunktion

$$f_M(x) = \sum_{i \in I} u_i \psi_i(x) \quad (4.38)$$

zu bilden. Dabei ist I eine M -elementige Teilmenge der Indizes $\{1, 2, \dots, L\}$ und $u_i \in \mathbb{R}$.

Problemstellung der Regressorauswahl

Bei gegebener Wavelet-Bibliothek W und Trainingsdaten \mathbb{O}_1^N , sei \mathcal{I}_M die Menge aller M -elementigen Teilmengen von $\{1, 2, \dots, L\}$, $M \leq L$. Mit dem Ansatz des mittleren quadratischen Fehlers aus (3.23) besteht das Problem nun darin, jenes $I \in \mathcal{I}_M$ zu finden, welches folgende Funktion minimiert:

$$J(I) = \min_{u_i, i \in I} \frac{1}{N} \sum_{k=1}^N \left(y_k - \sum_{i \in I} u_i \psi_i(x_k) \right)^2 \quad (4.39)$$

Die Minimierung in u_i führt dabei auf das klassische Problem der Minimierung der Quadrate, also das Lineare Ausgleichsproblem, für jedes $I \in \mathcal{I}_M$. Im Unterabschnitt 4.2.3 über Akaike's Final Prediction Error werden wir uns damit beschäftigen, die Größe M zu bestimmen. Im Moment ist es aus didaktischen Gründen besser, M als gegeben anzusehen. Für eine verständliche Darstellung der Algorithmen ist es sinnvoll, folgende Spaltenvektoren zu definieren:

$$v_i = \begin{bmatrix} \psi_i(x_1) \\ \vdots \\ \psi_i(x_N) \end{bmatrix}$$

wobei $\psi_i \in W$ und x_1, \dots, x_N die x -Werte der Trainingsdaten \mathbb{O}_1^N sind. Die ψ_i wurden gemäß (4.37) so normiert, dass die v_i jetzt Einheitsvektoren sind und somit gilt:

$$v_i^T v_i = 1, \quad i = 1 \dots L$$

Nun werden alle v_i , $i = 1 \dots L$ zu einer Matrix \mathbf{V} zusammengefasst:

$$\mathbf{V} = \{v_1, \dots, v_L\}$$

Ebenso definieren wir uns den Vektor der y -Werte:

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad (4.40)$$

wobei y_1, \dots, y_N die y -Werte der Trainingsdaten \mathbb{O}_1^N sind. Sei $\text{span}(\{v_i : i \in I\})$ der Unterraum des \mathbb{R}^N der durch Linearkombination der Vektoren v_i , $i \in I$ aufgespannt wird. Mit den nun eingeführten Notationen ist das Problem der Minimierung von (4.39) äquivalent dazu, M Vektoren v_i zu finden, welche den euklidischen Abstand zwischen Vektor y und dem Unterraum $\text{span}(\{v_i : i \in I\})$ minimieren. Der euklidische Abstand zwischen einem Vektor $y \in \mathbb{R}^N$ und einem Unterraum des \mathbb{R}^N ist definiert als der euklidische Abstand zwischen y und dessen Orthogonalprojektion auf den Unterraum. Im Prinzip könnten die optimalen M Regressoren gefunden werden, indem man alle M -Elementigen Teilmengen von W untersucht. In der Praxis gibt es aber meist zu viele solche Teilmengen, als dass dieses Vorgehen in erträglicher Rechenzeit abgeschlossen werden könnte. Deshalb muss man in der Praxis auf suboptimale, heuristische Algorithmen zurück greifen. Im Folgenden werden zwei dieser Algorithmen vorgestellt.

4.2.1 Sukzessive Regressorauswahl mit schrittweiser Orthogonalisierung

Diesem Algorithmus liegt die Idee zu Grunde, einen Regressor nach dem anderen sukzessive auszuwählen, sodass man in M Schritten die M Regressoren ψ_i mit $i \in I$ erhält. Im ersten Schritt wird das Wavelet ψ_{l_1} ausgewählt, das am besten zu den Trainingsdaten \mathbb{O}_1^N passt. In jedem der folgenden Schritte $i = 2, \dots, M$ wird immer jenes Wavelet ausgewählt, welches zusammen mit den zuvor ausgewählten am besten zu den Trainingsdaten passt. Wenn wir mit den Vektoren arbeiten, bedeutet es, dass man im i -ten Schritt jenes v_{l_i} auswählt, das zusammen mit den schon vorher ausgewählten Vektoren den Unterraum des \mathbb{R}^N aufspannt, der den kleinsten euklidischen Abstand zu y hat. Im Prinzip könnte man die Aufgabe lösen, indem man im i -ten Schritt $L - i + 1$ Lineare Ausgleichsprobleme löst. Für eine genaue Beschreibung des Linearen Ausgleichsproblems wird auf [9] verwiesen. Das ist aber in der Regel zu rechenaufwändig. Deshalb wird eine modifizierte Version dieses Algorithmus verwendet, die deutlich weniger Rechenaufwand benötigt. Sie stammt aus [2].

Nehmen wir an, dass in Schritt i des Algorithmus die $i - 1$ bereits ausgewählten Regressoren den Vektoren $v_{l_1}, \dots, v_{l_{i-1}}$ entsprechen. Um nun den i -ten Regressor auszuwählen, müssen wir den Abstand von y zum Unterraum $\text{span}(v_{l_1}, \dots, v_{l_{i-1}}, v_j)$ für jedes $j = 1, \dots, L$ und $j \neq l_1, \dots, l_{i-1}$ berechnen. Um eine effizientere Berechnung zu ermöglichen, orthogonalisieren wir die v_j zu den schon vorher ausgewählten Vektoren.

Nehmen wir zunächst an, dass die Vektoren $v_{l_1}, \dots, v_{l_{i-1}}$ bereits orthonormalisiert sind und in $w_{l_1}, \dots, w_{l_{i-1}}$ umbenannt wurden. Dann gilt: $\text{span}(v_{l_1}, \dots, v_{l_{i-1}}, v_j) = \text{span}(w_{l_1}, \dots, w_{l_{i-1}}, v_j)$. Wir berechnen nun für jedes $j = 1, \dots, L$ mit $j \neq l_1, \dots, l_{i-1}$:

$$p_j = v_j - [(v_j^T w_{l_1})w_{l_1} + \dots + (v_j^T w_{l_{i-1}})w_{l_{i-1}}] \quad (4.41)$$

$$q_j = (p_j^T p_j)^{-\frac{1}{2}} p_j \quad (4.42)$$

Nun wollen wir jenes v_j oder besser gesagt jenes q_j suchen, welches folgende Funktion minimiert:

$$\begin{aligned} J(v_j) &= J(q_j) \\ &= [y - (\tilde{u}_{l_1} w_{l_1} + \dots + \tilde{u}_{l_{i-1}} w_{l_{i-1}} + \tilde{u}_j q_j)]^T [y - (\tilde{u}_{l_1} w_{l_1} + \dots + \tilde{u}_{l_{i-1}} w_{l_{i-1}} + \tilde{u}_j q_j)] \\ &= [y - \mathbf{W}_j U_j]^T [y - \mathbf{W}_j U_j] \end{aligned}$$

mit Matrix $\mathbf{W}_j = (w_{l_1}, \dots, w_{l_{i-1}}, q_j)$

$$\text{und Vektor } U_j = (\tilde{u}_{l_1}, \dots, \tilde{u}_{l_{i-1}}, \tilde{u}_j)^T = (\mathbf{W}_j^T \mathbf{W}_j)^{-1} \mathbf{W}_j^T y = \mathbf{W}_j^T y \quad (4.43)$$

Der Vektor U_j ist das Ergebnis eines Linearen Ausgleichsproblems wie in [9] beschrieben. Die letzte Gleichung folgt aus der Orthonormalität der $w_{l_1}, \dots, w_{l_{i-1}}, q_j$. Damit ergibt sich weiter:

$$\begin{aligned} J(v_j) &= y^T y + U_j^T \mathbf{W}_j^T \mathbf{W}_j U_j - 2U_j^T \mathbf{W}_j^T y \\ &= y^T y + U_j^T U_j - 2U_j^T \mathbf{W}_j^T y \end{aligned}$$

In (4.43) haben wir $U_j = \mathbf{W}_j^T y$ erhalten. Deshalb gilt:

$$\begin{aligned} J(v_j) &= y^T y + U_j^T U_j - 2U_j^T U_j \\ &= y^T y - U_j^T U_j \\ &= y^T y - (\tilde{u}_{l_1}^2 + \cdots + \tilde{u}_{l_{i-1}}^2 + \tilde{u}_j^2) \end{aligned}$$

Folglich ist die Minimierung von $J(v_j)$ gleichbedeutend mit der Maximierung von $\tilde{u}_{l_1}^2 + \cdots + \tilde{u}_{l_{i-1}}^2 + \tilde{u}_j^2$. Aus (4.43) wissen wir auch:

$$\begin{aligned} \tilde{u}_{l_k} &= w_{l_k}^T y, \quad k = 1, \dots, i-1 \\ \tilde{u}_j &= q_j^T y \end{aligned}$$

Demzufolge ist $\tilde{u}_{l_1}^2 + \cdots + \tilde{u}_{l_{i-1}}^2$ unabhängig von q_j . Daraus schließen wir, dass die Minimierung von $J(v_j)$ gleichbedeutend ist mit der Maximierung von

$$\tilde{u}_j^2 = (q_j^T y)^2$$

Nach M Iterationen sind die Werte von l_1, \dots, l_M bestimmt. Dann ist

$$y = [w_{l_1}, \dots, w_{l_M}] [\tilde{u}_{l_1}, \dots, \tilde{u}_{l_M}]^T + \gamma_M = [v_{l_1}, \dots, v_{l_M}] [u_{l_1}, \dots, u_{l_M}]^T + \gamma_M$$

wobei γ_M üblicherweise als Residuenvektor bezeichnet wird. Es gilt also

$$[w_{l_1}, \dots, w_{l_M}] [\tilde{u}_{l_1}, \dots, \tilde{u}_{l_M}]^T = [v_{l_1}, \dots, v_{l_M}] [u_{l_1}, \dots, u_{l_M}]^T \quad (4.44)$$

Das Wavelet-Netzwerk soll sich am Ende ergeben zu

$$f_M(x) = \sum_{i=1}^M u_i \psi_{l_i}(x)$$

also müssen wir letztlich noch die Werte der u_{l_i} aus den \tilde{u}_{l_i} bestimmen. Wenn wir in (4.41) und (4.42) $j = l_i$ verwenden, so erhalten wir:

$$\begin{aligned} v_{l_i} &= [(v_{l_i}^T w_{l_1})w_{l_1} + \cdots + (v_{l_i}^T w_{l_{i-1}})w_{l_{i-1}}] + p_{l_i} \\ &= [(v_{l_i}^T w_{l_1})w_{l_1} + \cdots + (v_{l_i}^T w_{l_{i-1}})w_{l_{i-1}}] + (p_{l_i}^T p_{l_i})^{\frac{1}{2}} q_{l_i} \\ &= [(v_{l_i}^T w_{l_1})w_{l_1} + \cdots + (v_{l_i}^T w_{l_{i-1}})w_{l_{i-1}}] + (p_{l_i}^T p_{l_i})^{\frac{1}{2}} w_{l_i} \\ &= [a_{1i}w_{l_1} + \cdots + a_{i-1i}w_{l_{i-1}}] + a_{ii}w_{l_i} \end{aligned}$$

wobei

$$\begin{aligned} a_{ki} &= v_{l_i}^T w_{l_k}, \quad k = 1, \dots, i-1 \\ a_{ii} &= (p_{l_i}^T p_{l_i})^{\frac{1}{2}} \end{aligned}$$

Man kann nun schreiben:

$$[w_{l_1}, \dots, w_{l_M}] \mathbf{A} = [v_{l_1}, \dots, v_{l_M}] \quad (4.45)$$

wobei A folgende obere Dreiecksmatrix ist:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & & a_{1M} \\ 0 & a_{22} & a_{23} & \cdots & & a_{2M} \\ 0 & 0 & a_{33} & \cdots & & a_{3M} \\ \vdots & \vdots & \ddots & \ddots & & \vdots \\ 0 & 0 & \dots & 0 & a_{M-1M-1} & a_{M-1M} \\ 0 & 0 & \dots & & 0 & a_{MM} \end{bmatrix}$$

Dann können die u_{l_i} errechnet werden, indem man das folgende dreieckförmige Gleichungssystem löst, welches durch Kombination von (4.44) und (4.45) entsteht:

$$\mathbf{A} [u_{l_1}, \dots, u_{l_M}]^T = [\tilde{u}_{l_1}, \dots, \tilde{u}_{l_M}]^T \quad (4.46)$$

Man beachte, dass in (4.41) die Projektionen

$$(v_j^T w_{l_1})w_{l_1} + \dots + (v_j^T w_{l_{i-2}})w_{l_{i-2}}$$

in den vorhergehenden Schritten bereits berechnet wurden. Nur der letzte Term $(v_j^T w_{l_{i-1}})w_{l_{i-1}}$ muss im gegenwärtigen Schritt noch berechnet werden. Das vermindert die benötigten Rechenoperationen des Algorithmus erheblich. Der Algorithmus lässt sich in Pseudocode folgendermaßen darstellen:

Algorithmus 4.1*Sukzessive Regressorauswahl mit schrittweiser Orthogonalisierung***Schritt 1:** *erzeuge*

$$I_1 = \{1, 2, \dots, L\}$$

finde

$$l_1 = \arg \max_{j \in I_1} (v_j^T y)^2$$

und setze

$$\tilde{u}_{l_1} = v_{l_1}^T y$$

$$w_{l_1} = v_{l_1}$$

$$a_{11} = 1$$

$$p_j^{(1)} = v_j, \quad j = 1, \dots, L, \quad j \neq l_1$$

Schritt i ($i = 2, \dots, M$):

$$I_i = I_{i-1} - l_{i-1}$$

berechne nun für jedes $j \in I_i$

$$p_j^{(i)} = p_j^{(i-1)} - (v_j^T w_{l_{i-1}}) w_{l_{i-1}}$$

$$I_i := I_i \setminus \{j : p_j^{(i)} = 0\}$$

finde

$$l_i = \arg \max_{j \in I_i} \frac{[(p_j^{(i)})^T y]^2}{(p_j^{(i)})^T p_j^{(i)}}$$

und setze

$$w_{l_i} = \left[(p_{l_i}^{(i)})^T p_{l_i}^{(i)} \right]^{-\frac{1}{2}} p_{l_i}^{(i)}$$

$$\tilde{u}_{l_i} = w_{l_i}^T y$$

$$a_{ki} = v_{l_i}^T w_{l_k}, \quad k = 1, \dots, i-1$$

$$a_{ii} = \left[(p_{l_i}^{(i)})^T p_{l_i}^{(i)} \right]^{-\frac{1}{2}}$$

Schritt $M+1$: *löse Gleichung (4.46) um die u_i , $i = 1, \dots, M$ zu erhalten und konstruiere*

$$f_M(x) = \sum_{i=1}^M u_i \psi_{l_i}(x)$$

Dieser aus [2] übernommene Standardalgorithmus wurde nicht eins zu eins implementiert sondern in etwas abgewandelter Form. In den ersten $d + 1$ Schritten wurde keine eigentliche Regressorauswahl vorgenommen sondern es wurden stets die affinen Terme aus W_{aff} hinzugefügt. Diese Terme können unter Umständen die Funktion bereits gut beschreiben und sie „kosten“ zusammen nur $d + 1$ Parameter im Wavelet-Netzwerk nach (3.25) und zwar für die Gewichte $w_{i,aff}$, $i = 0 \dots d$. Jedes echte Wavelet „kostet“ bereits $2d + 1$ Parameter - jeweils d für Translation t und Dilation a und einen für das Gewicht w . Man könnte auch sagen: weil die affinen Terme wesentlich „billiger“ als Wavelets sind, werden sie bei der Regressorauswahl bevorzugt. Zudem wurde abweichend von dieser Beschreibung bei der Implementierung des Algorithmus bei allen Indizes mit 0 angefangen. Des weiteren wurden Termteile, die in der Beschreibung wiederholt vorkommen, wie $(p_{l_i}^{(i)})^T p_{l_i}^{(i)}$ natürlich nur einmal pro Schritt berechnet. Da M , anders als oben dargestellt, nicht a priori gegeben ist, muss der Algorithmus nicht nur mit M Schritten sondern mit $i_{max} \geq M$ Schritten durchlaufen werden. Es gilt

$$i_{max} = \max_{i \in \mathbb{N}} (np(i) \leq N)$$

$$np(i) = d + 1 + (2d + 1)(i - d - 1), \quad i \geq d + 1$$

Dabei steht $np(i)$ für die Anzahl der Parameter des letztendlich entstehenden Wavelet-Netzwerkes, die sich ergibt, wenn i Regressoren aus W ausgewählt werden. In $i \geq d + 1$ Schritten werden $d + 1$ Parameter für die affinen Terme benötigt und $2d + 1$ Parameter für jedes der $i - d - 1$ Wavelets. Man schließt somit gleich von vornherein aus, dass Wavelet-Netzwerke entstehen, die mehr Parameter haben als es Trainingspunkte gibt.

$$\Rightarrow (2d + 1)(i - d - 1) \leq N - d - 1$$

$$\Rightarrow i \leq \frac{N - d - 1}{2d + 1} + d + 1$$

$$\Rightarrow i_{max} = \left\lfloor \frac{N - d - 1}{2d + 1} \right\rfloor + d + 1$$

Im Unterabschnitt 4.2.3 über Akaike's Final Prediction Error werden wir ein heuristisches Kriterium einführen mit dem am Ende festgelegt wird, wie groß M ist. Dazu werden wir noch den Residuenvektor γ_i in Schritt i des Algorithmus benötigen. Für diesen gilt:

$$\begin{aligned} \gamma_0 &= y \\ \gamma_i &= y - (v_{l_1} u_{l_1} + \dots + v_{l_i} u_{l_i}) \\ &= y - (w_{l_1} \tilde{u}_{l_1} + \dots + w_{l_i} \tilde{u}_{l_i}) \\ \Rightarrow \gamma_i &= \gamma_{i-1} - w_{l_i} \tilde{u}_{l_i}, \quad i = 1, \dots, i_{max} \end{aligned} \tag{4.47}$$

Nach i_{max} Schritten ist \mathbf{A} eine $i_{max} \times i_{max}$ -Matrix und $\tilde{U} = (\tilde{u}_1, \dots, \tilde{u}_{i_{max}})^T$ ein i_{max} -Spaltenvektor. Wenn am Ende aber nur die ersten $M \leq i_{max}$ Regressoren

ausgewählt werden sollen, dann reduziert man \mathbf{A} auf seine oberen linken $M \times M$ Elemente und \tilde{U} auf seine ersten M Einträge und löst damit zum Schluss (4.46).

4.2.2 Rückwärtselimination von Regressoren

Im Gegensatz zum vorhergehenden Verfahren will man das Approximationsergebnis in Form von (4.38) erreichen, indem man mit allen Regressoren, also allen Wavelets und affinen Termen, anfängt und dann in jedem Schritt einen eliminiert. Dabei wird versucht, das Residuum $\|\gamma\|_2$ so wenig wie möglich zu vergrößern. Benutzt man alle Regressoren aus W für die Approximationsfunktion, dann lautet diese:

$$f_l(x) = \sum_{i=1}^L u_i \psi_i(x)$$

Dabei ergeben sich die u_i als Lösung eines Linearen Ausgleichsproblems wie in [9] beschrieben zu

$$(u_1, \dots, u_L)^T = [(v_1, \dots, v_L)^T (v_1, \dots, v_L)]^{-1} (v_1, \dots, v_L)^T y \quad (4.48)$$

Hierbei ist zu beachten, dass die Inversion der Matrix $(v_1, \dots, v_L)^T (v_1, \dots, v_L)$ nicht möglich ist, falls sie singular ist. Dieser Fall tritt insbesondere dann ein, wenn mehr Regressoren als Trainingspunkte vorhanden sind, also wenn $L > N$. Falls die Matrix singular sein sollte, bricht der Algorithmus mit einer Fehlermeldung ab. Die Residuen

$$\gamma_L(k) = y_k - f_l(x_k), \quad k = 1 \dots N$$

können wieder in vektorieller Form geschrieben werden als

$$\gamma_L = y - (v_1, \dots, v_L)(u_1, \dots, u_L)^T \quad (4.49)$$

Führt man (4.48) und (4.49) zusammen, erhält man

$$\gamma_L^T \gamma_L = y^T y - y^T \mathbf{V}_0 (\mathbf{V}_0^T \mathbf{V}_0)^{-1} \mathbf{V}_0^T y$$

wobei die Matrix $\mathbf{V}_0 = (v_1, \dots, v_L)$ ist. Wenn wir nun ein Wavelet, sagen wir ψ_j , aus $f_L(x)$ entfernen, kann die gleiche Berechnung durchgeführt werden und wir erhalten ein Ergebnis der Form:

$$\gamma_{L-1}^T \gamma_{L-1} = y^T y - y^T C(v_j | \mathbf{V}_0) [C(v_j | \mathbf{V}_0)^T C(v_j | \mathbf{V}_0)]^{-1} C(v_j | \mathbf{V}_0)^T y$$

Hierbei bedeutet der Operator C das Komplement einer Matrix. Sei z.B. $\mathbf{U} = [U_1, U_2, U_3]$, dann ist $C(U_2 | \mathbf{U}) = [U_1, U_3]$. Folglich steigt die Summe der Residuenquadrate durch die Elimination von ψ_i aus $f_L(x)$ um

$$\begin{aligned} J(\psi_j) &= \gamma_{L-1}^T \gamma_{L-1} - \gamma_L^T \gamma_L \\ &= y^T \mathbf{V}_0 (\mathbf{V}_0^T \mathbf{V}_0)^{-1} \mathbf{V}_0^T y - y^T C(v_j | \mathbf{V}_0) [C(v_j | \mathbf{V}_0)^T C(v_j | \mathbf{V}_0)]^{-1} C(v_j | \mathbf{V}_0)^T y \end{aligned} \quad (4.50)$$

Entfernt man aus $f_L(x)$ das Wavelet ψ_j , welches (4.50) minimiert, so erhält man $f_{L-1}(x)$. Dieser Schritt wird dann wiederholt ausgeführt, sodass sich folgender Algorithmus ergibt:

Algorithmus 4.2 *Rückwärtselimination von Regressoren*

Schritt 0: setze $\mathbf{V}_0 = (v_1, \dots, v_L)$

Schritt i ($i = 1, \dots, L - M$): Sei $I_i = \{j : j = 1, \dots, L \text{ und } j \neq l_1, \dots, l_{i-1}\}$.
finde

$$l_i = \arg \max_{j \in I_i} y^T C(v_j | \mathbf{V}_{i-1}) [C(v_j | \mathbf{V}_{i-1})^T C(v_j | \mathbf{V}_{i-1})]^{-1} C(v_j | \mathbf{V}_{i-1})^T y$$

setze $\mathbf{V}_i = C(v_{l_i} | \mathbf{V}_{i-1})$

Schritt $L - M + 1$: Sei $I_{L-M+1} = \{j : j = 1, \dots, L \text{ und } j \neq l_1, \dots, l_{L-M}\}$
erzeuge

$$f_M(x) = \sum_{j \in I_{L-M+1}} u_j \psi_j(x)$$

wobei die u_j die Komponenten des Vektors u sind, der sich ergibt zu

$$u = (\mathbf{V}_{L-M}^T \mathbf{V}_{L-M})^{-1} \mathbf{V}_{L-M}^T y$$

Der für diesen Algorithmus benötigte Rechenaufwand ist beträchtlich. Zum Beispiel müssten im i -ten Schritt $L - i + 1$ Matrizen invertiert werden. Der Rechenaufwand für Matrixinversionen kann auf folgende Weise reduziert werden: Für eine beliebige Matrix $\mathbf{U} = [U_1, U_2, U_3]$ bei der U_1, U_2, U_3 Unterblöcke von \mathbf{U} sind, ist

$$\mathbf{U}^T \mathbf{U} = \begin{bmatrix} U_1^T U_1 & U_1^T U_2 & U_1^T U_3 \\ U_2^T U_1 & U_2^T U_2 & U_2^T U_3 \\ U_3^T U_1 & U_3^T U_2 & U_3^T U_3 \end{bmatrix}$$

Wir gehen in unserem Fall davon aus, dass U_2 immer ein Spaltenvektor ist, wogegen U_1 und U_3 Matrizen sind. Angenommen $(\mathbf{U}^T \mathbf{U})^{-1}$ wurde bereits berechnet und genau so partitioniert wie $\mathbf{U}^T \mathbf{U}$:

$$(\mathbf{U}^T \mathbf{U})^{-1} = \begin{bmatrix} \lambda_{11} & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & \lambda_{22} & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & \lambda_{33} \end{bmatrix}$$

Dann gilt:

$$\begin{aligned} ([U_1, U_3]^T [U_1, U_3])^{-1} &= \begin{bmatrix} U_1^T U_1 & U_1^T U_3 \\ U_3^T U_1 & U_3^T U_3 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \lambda_{11} & \lambda_{13} \\ \lambda_{31} & \lambda_{33} \end{bmatrix} - \lambda_{22}^{-1} \begin{bmatrix} \lambda_{12} \\ \lambda_{32} \end{bmatrix} \begin{bmatrix} \lambda_{21} & \lambda_{23} \end{bmatrix} \end{aligned} \quad (4.51)$$

Diese Beziehung kann mit elementaren Mitteln verifiziert werden. Benutzt man die Beziehung in (4.51), so muss nur $\mathbf{V}_0^T \mathbf{V}_0$ einmal invertiert werden. Die Matrix $[(C(v_j|\mathbf{V}_i)^T C(v_j|\mathbf{V}_i))]^{-1}$ kann aus den Unterblöcken von $(\mathbf{V}_i^T \mathbf{V}_i)^{-1}$ gewonnen werden. Soweit sind wir wiederum der Theorie nach [2] gefolgt.

Wie schon zuvor stehen wir auch bei diesem Algorithmus vor dem Problem, dass M nicht a priori bekannt ist. Deshalb werden alle Wavelets nacheinander eliminiert, sodass am Ende nur noch die $d + 1$ affinen Terme übrig bleiben. In jedem Schritt i muss $\gamma_i^T \gamma_i$ berechnet werden. Wofür, werden wir noch im Unterabschnitt Akaike's Final Prediction Error sehen. Bei der praktischen Umsetzung des Algorithmus wurde wie folgt vorgegangen: Die Invertierung der $L \times L$ -Matrix $(\mathbf{V}_i^T \mathbf{V}_i)^{-1}$ erfolgte in zwei Schritten. Da die Matrix auf jeden Fall symmetrisch, positiv semidefinit ist und für die Inversion auf jeden Fall als positiv definit angenommen werden muss, bietet sich zunächst eine Cholesky-Zerlegung an. Das Verfahren wird in [9] genau erklärt. Dabei wird eine $L \times L$ untere Dreiecksmatrix $\hat{\mathbf{L}}$ erzeugt, sodass gilt

$$\hat{\mathbf{L}} \hat{\mathbf{L}}^T = \mathbf{V}_0^T \mathbf{V}_0 \quad (4.52)$$

Die Inverse der unteren Dreiecksmatrix $\hat{\mathbf{L}}$ ist wieder eine untere Dreiecksmatrix $\hat{\mathbf{L}}^{-1}$. Das macht die Inversion sehr einfach. Mit (4.52) ergibt sich

$$(\mathbf{V}_0^T \mathbf{V}_0)^{-1} = (\hat{\mathbf{L}} \hat{\mathbf{L}}^T)^{-1} = (\hat{\mathbf{L}}^T)^{-1} \hat{\mathbf{L}}^{-1} = (\hat{\mathbf{L}}^{-1})^T \hat{\mathbf{L}}^{-1}$$

Das nächste Problem ist, dass in jedem der i Schritte

$$l_i = \arg \max_{j \in I_i} y^T C(v_j|\mathbf{V}_{i-1}) [C(v_j|\mathbf{V}_{i-1})^T C(v_j|\mathbf{V}_{i-1})]^{-1} C(v_j|\mathbf{V}_{i-1})^T y \quad (4.53)$$

berechnet werden muss, was $|I_i| = L - i + 1$ Berechnungen des Terms hinter $\arg \max$ erfordert. Da die Anzahl der Regressorkandidaten L mitunter sehr groß sein kann, ist an dieser Stelle eine numerisch effiziente Berechnung dieses Terms notwendig. Für die bessere Verständlichkeit der folgenden Ausführungen beschränken wir uns auf den ersten Schritt $i = 1$. Die nachfolgenden Schritte sind vom Prinzip her analog, aber die Indizierung wird etwas komplizierter. Hierzu führen wir einige neue Größen ein. Es sei

$$\begin{aligned} b &= (b_1, \dots, b_L)^T = \mathbf{V}_0^T y \\ b^{(-j)} &= (b_1, \dots, b_{j-1}, 0, b_{j+1}, \dots, b_L)^T \end{aligned} \quad (4.54)$$

das heißt bei $b^{(-j)}$ wurde die j -te Komponente von b zu Null gesetzt. Sei

$$\begin{aligned} \mathbf{\Lambda} &= \begin{bmatrix} \lambda_{11} & \cdots & \lambda_{1L} \\ \vdots & \ddots & \\ \lambda_{L1} & & \lambda_{LL} \end{bmatrix} = (\Lambda_1, \dots, \Lambda_L) = (\mathbf{V}_0^T \mathbf{V}_0)^{-1} \\ \Lambda_j &= (\lambda_{1j}, \dots, \lambda_{Lj})^T, \quad j = 1, \dots, L \\ \Lambda_j^{(-k)} &= (\lambda_{1j}, \dots, \lambda_{k-1j}, 0, \lambda_{k+1j}, \dots, \lambda_{Lj})^T, \quad j, k = 1, \dots, L \\ \mathbf{\Lambda}^{(-j)} &= (\Lambda_1^{(-j)}, \dots, \Lambda_{j-1}^{(-j)}, 0, \Lambda_{j+1}^{(-j)}, \dots, \Lambda_L^{(-j)}), \quad j = 1, \dots, L \end{aligned}$$

$$= \begin{bmatrix} \lambda_{11} & \cdots & \lambda_{1j-1} & 0 & \lambda_{1j+1} & \cdots & \lambda_{1L} \\ \vdots & \ddots & & 0 & \vdots & \ddots & \\ \lambda_{j-11} & & \lambda_{j-1j-1} & 0 & \lambda_{j-1j+1} & & \lambda_{j-1L} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \lambda_{j+11} & \cdots & \lambda_{j+1j-1} & 0 & \lambda_{j+1j+1} & \cdots & \lambda_{j+1L} \\ \vdots & \ddots & \vdots & 0 & \vdots & \ddots & \\ \lambda_{L1} & & \lambda_{Lj-1} & 0 & \lambda_{Lj+1} & & \lambda_{LL} \end{bmatrix} \quad (4.55)$$

oder einfach gesagt: In $\mathbf{\Lambda}^{(-j)}$ ist wie $\mathbf{\Lambda}$, nur wurden die j -te Zeile und Spalte zu Null gesetzt. Wir führen weiterhin ein:

$$\mathbf{\Phi}_j = [C(v_j|\mathbf{V}_0)^T C(v_j|\mathbf{V}_0)]^{-1} = \begin{bmatrix} \phi_{11} & \cdots & \phi_{1L-1} \\ \vdots & \ddots & \\ \phi_{L-11} & & \phi_{L-1L-1} \end{bmatrix}_j, \quad j = 1, \dots, L$$

Nun führen wir die Matrix $\phi_j^{(+k)}$ ein, die alle Elemente aus ϕ_j enthält, bei der aber in die k -te Zeile und Spalte Nullen eingefügt wurden.

$$\mathbf{\Phi}_j^{(+k)} = \begin{bmatrix} \phi_{11} & \cdots & \phi_{1k-1} & 0 & \phi_{1k} & \cdots & \phi_{1L-1} \\ \vdots & \ddots & & 0 & \vdots & \ddots & \\ \phi_{k-11} & & \phi_{k-1k-1} & 0 & \phi_{k-1k} & & \phi_{k-1L-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \phi_{k1} & \cdots & \phi_{kk-1} & 0 & \phi_{kk} & \cdots & \phi_{kL-1} \\ \vdots & \ddots & \vdots & 0 & \vdots & \ddots & \\ \phi_{L-11} & & \phi_{L-1k-1} & 0 & \phi_{L-1k} & & \phi_{L-1L-1} \end{bmatrix}_j, \quad j, k = 1, \dots, L \quad (4.56)$$

Wir wollen (4.54) und (4.56) verwenden um den Term hinter $\arg \min$ in Gleichung (4.53) darzustellen. Dann gilt

$$\begin{aligned} & y^T C(v_j|\mathbf{V}_0) [C(v_j|\mathbf{V}_0)^T C(v_j|\mathbf{V}_0)]^{-1} C(v_j|\mathbf{V}_0)^T y \\ &= b^{(-j)T} \mathbf{\Phi}_j^{(+j)} b^{(-j)} \\ &= b^{(-j)T} \left[\mathbf{\Lambda}^{(-j)} - \lambda_{jj}^{-1} \Lambda_j^{(-j)} \Lambda_j^{(-j)T} \right] b^{(-j)} \quad (\text{wegen (4.51)}) \\ &= b^{(-j)T} \mathbf{\Lambda}^{(-j)} b^{(-j)} - \lambda_{jj}^{-1} b^{(-j)T} \Lambda_j^{(-j)} \Lambda_j^{(-j)T} b^{(-j)} \\ &= b^{(-j)T} \mathbf{\Lambda} b^{(-j)} - \lambda_{jj}^{-1} (\Lambda_j^T b^{(-j)})^2 \\ &= b^{(-j)T} (\mathbf{\Lambda} b - \Lambda_j b_j) - \lambda_{jj}^{-1} (\Lambda_j^T b^{(-j)})^2 \end{aligned}$$

Diese Berechnungen waren alle für den ersten Schritt mit $i = 1$ gedacht. Alle weiteren Schritte berechnen sich im Prinzip analog. Betrachtet man die letzte Zeile, so wird der Nutzen all dieser Umformungen deutlich. Der Term hinter $\arg \min$ in (4.53) sieht kompliziert aus und muss im i -ten Schritt für alle $L - i + 1$ der v_j berechnet werden. Durch die Umformungen benötigen wir nun für diese $L - i + 1$ einzelnen Termberechnungen nur Vektoroperationen und

keine Matrixoperationen mehr. $\mathbf{\Lambda}$ wird jeweils aus dem vorherigen Schritt $i - 1$ übernommen. Der Vektor b wird durch Herausstreichen eines Eintrages aus Schritt $i - 1$ übernommen. $\mathbf{\Lambda}b$ muss nur einmal in Schritt i berechnet werden. Die Matrix in (4.51) wird nur einmal am Ende des i -ten Schrittes wirklich ausgerechnet um als $\mathbf{\Lambda}$ für den nächsten Schritt $i + 1$ Verwendung zu finden. Wir benötigen noch

$$\gamma_i^T \gamma_i = y^T y - y^T C(v_j | \mathbf{V}_{i-1}) [C(v_j | \mathbf{V}_{i-1})^T C(v_j | \mathbf{V}_{i-1})]^{-1} C(v_j | \mathbf{V}_{i-1})^T y$$

Da der hintere Term bereits, wie oben gezeigt, berechnet wurde, bleibt der Rechenaufwand vernachlässigbar.

4.2.3 Akaike's Final Prediction Error

In den beiden Regressorauswahlalgorithmen wurde bisher noch nicht festgelegt, wie viele Wavelets letztendlich verwendet werden sollen. Die affinen Terme werden jedoch immer verwendet. Man benötigt in jedem Fall eine Kriteriumsfunktion, die auf der einen Seite die Abweichung des Modells von den Trainingsdaten, also die Residuenvektoren γ berücksichtigt, auf der anderen Seite aber auch die Komplexität des Modells bewertet. In dieser Arbeit wurde wie in [2] eine heuristische Kriteriumsfunktion verwendet - Die sogenannte Akaike's final prediction error criterion (FPE):

$$J_{FPE}(f) = \frac{1 + n_p/N}{1 - n_p/N} \frac{1}{2N} \sum_{k=1}^N [f(x_k) - y_k]^2 \quad (4.57)$$

Dabei ist f ein erweitertes Wavelet-Netzwerk und n_p dessen Anzahl an Parametern. Es gilt folglich

$$n_p = d + 1 + (2d + 1)n_{wav}$$

Es werden $d + 1$ Parameter für die affinen Terme benötigt und $2d + 1$ für jedes der n_{wav} Wavelets. In einem Schritt i der Algorithmen ist also

$$J_{FPE}(f_i) = \frac{1 + n_p(i)/N}{1 - n_p(i)/N} \frac{1}{2N} \gamma_i^T \gamma_i$$

Dieses Kriterium wird fortlaufend in jedem Schritt i der Algorithmen berechnet. Am Ende wird jenes f_i ausgewählt, welches $J_{FPE}(f_i)$ minimiert.

4.3 Regressoroptimierung mit Levenberg-Marquardt

In den vorangegangenen Regressorauswahlalgorithmen wurde ein erweitertes Wavelet-Netzwerk f wie in (3.25) erstellt, das die Trainingsdaten bereits gut approximiert. Diese Funktion f gilt es nun weiter zu verbessern. Bis jetzt wurden nur Regressoren ausgewählt und ihre Gewichte wurden direkt oder indirekt als Ergebnis eines Linearen Ausgleichsproblems optimal angepasst. Die ausgewählten Regressoren werden nun als gegeben angesehen, aber ihre Parameter werden zur weiteren Anpassung freigegeben. Hierzu wird ein iteratives Verfahren verwendet. Ein Iterationszyklus besteht dabei aus

1. Anpassung der affinen Terme mit Hilfe eines Linearen Ausgleichsproblems.
2. Sukzessive Anpassung der Wavelets mit Levenberg-Marquardt Schritten. Das heißt, alle Wavelets werden einzeln und nacheinander angepasst.

Diese Iterationszyklen wurden mehrmals durchlaufen; typischer Weise 50 mal. Bei den affinen Termen bildet man zunächst den Residuenvektor γ und die Matrix \mathbf{A}

$$\gamma = \begin{bmatrix} y_1 - f_{wav}(x_1) \\ \vdots \\ y_N - f_{wav}(x_N) \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 1 & x_1^T \\ \vdots & \vdots \\ 1 & x_N^T \end{bmatrix}$$

wobei $(x_i, y_i) \in \mathbb{O}_1^N$. Die Funktion f_{wav} umfasst alle Wavelets und hat eine Form wie in (3.24). Damit ergeben sich die Gewichte der affinen Terme aus W_{aff} wie in (4.35) als Lösung des Linearen Ausgleichsproblems zu

$$w = \begin{bmatrix} w_{0,aff} \\ w_{1,aff} \\ \vdots \\ w_{d,aff} \end{bmatrix} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \gamma$$

Bei der numerischen Umsetzung wird $\mathbf{A}^T \mathbf{A}$ natürlich nicht invertiert sondern Cholesky-zerlegt mit anschließender Vorwärts- und Rückwärtssubstitution (Details siehe [9]).

Die Theorie zur Anpassung der Wavelets ist etwas umfangreicher. Betrachten wir zunächst die generelle Funktionsweise des Levenberg-Marquardt-Algorithmus. Sei θ ein Vektor und $F(\theta)$ eine stetig differenzierbare, vektorwertige Funktion. Dann ist

$$\eta(\theta) = \|F(\theta)\|_2^2 = F(\theta)^T F(\theta) \quad (4.58)$$

Das Problem lautet: finde

$$\eta_{min} = \eta(\theta_{min}) = \min_{\theta} \eta(\theta) = \min_{\theta} \|F(\theta)\|_2^2$$

Dieses Problem soll iterativ gelöst werden, indem man mit einem θ_0 beginnt und eine Folge (θ_i) konstruiert, für die gilt: $\|F(\theta_{i+1})\|_2^2 < \|F(\theta_i)\|_2^2 \quad \forall i \in \mathbb{N}$. Sei ferner \mathbf{JF} die Jakobimatrix von F . Dann gilt

$$\begin{aligned} g_i &= 2\mathbf{JF}(\theta_i)^T F(\theta_i) = \nabla\eta(\theta_i) \\ \mathbf{H}_i &= 2\mathbf{JF}(\theta_i)^T \mathbf{JF}(\theta_i) \approx \nabla^2\eta(\theta_i) \end{aligned} \quad (4.59)$$

Sei weiterhin \mathbf{I} die Einheitsmatrix. Dann berechnen wir

$$\begin{aligned} s_i^{(0)} &= [\mathbf{H}_i + 2\lambda\mathbf{I}]^{-1} g_i \\ &= [\mathbf{JF}(\theta_i)^T \mathbf{JF}(\theta_i) + \lambda\mathbf{I}]^{-1} \mathbf{JF}(\theta_i)^T F(\theta_i) \end{aligned} \quad (4.60)$$

Dabei ist λ eine sehr kleine Größe, die sicher stellen soll, dass die Matrixinversion, oder besser die Cholesky-Zerlegung mit anschließender Vorwärts- und Rückwärtssubstitution, immer gut genug konditioniert ist. Sonst würden wir scheitern, sobald $\mathbf{JF}(\theta_i)^T \mathbf{JF}(\theta_i)$ nicht vollen Rang hätte. $s_i^{(0)}$ ist dabei die Schrittweite. Jetzt wird normalerweise das Armijo-Goldstein-Verfahren angewandt. Hier soll nur eine vereinfachte Form beschrieben werden. Nach der Berechnung von $s_i^{(0)}$ wird geprüft, ob

$$\eta(\theta_i - s_i^{(0)}) < \eta(\theta_i)$$

Falls nein, wird die Schrittweite so oft halbiert, also $s_i^{(k+1)} = s_i^{(k)}/2$, bis

$$\eta(\theta_i - s_i^{(k+1)}) < \eta(\theta_i)$$

Dann wird abgebrochen und

$$\theta_{i+1} = \theta_i - s_i^{(k+1)}$$

gesetzt. Damit wird sichergestellt, dass in jedem Schritt ein Abstieg erfolgt. Das Levenberg-Marquardt-Verfahren ist ein weit verbreitetes Verfahren, das effiziente Berechnungen zulässt und in der Regel hinreichend schnell konvergiert. Kommen wir zurück zu unserem erweiterten Wavelet-Netzwerk f und betrachten wir ein einzelnes, darin enthaltenes Wavelet ψ_θ mit $\theta^T = (w, a^T, t^T)^T$. Dann gilt

$$\psi_\theta = w\psi[a \star (x - t)] \quad (4.61)$$

Wenn wir nun ψ_θ aus f entfernen, erhalten wir eine Funktion $f^{(-)}$, sodass

$$f = f^{(-)} + \psi_\theta$$

Wir schließen nun den Kreis und setzen für $F(\theta)$ in (4.58)

$$F(\theta) = \begin{bmatrix} y_1 - f(x_1) \\ \vdots \\ y_N - f(x_N) \end{bmatrix} = \begin{bmatrix} y_1 - f^{(-)}(x_1) - \psi_\theta(x_1) \\ \vdots \\ y_N - f^{(-)}(x_N) - \psi_\theta(x_N) \end{bmatrix}$$

Dann benötigen wir noch $\mathbf{JF}(\theta)$ und erhalten

$$\mathbf{JF}(\theta) = - \begin{bmatrix} \nabla_{\theta}^T \psi_{\theta}(x_1) \\ \vdots \\ \nabla_{\theta}^T \psi_{\theta}(x_N) \end{bmatrix}$$

Hierbei müssen wir die einzelnen Gradienten wiederum herunterbrechen zu

$$\nabla_{\theta} \psi_{\theta}(x_i) = \begin{bmatrix} \frac{\partial \psi_{\theta}}{\partial w} \\ \frac{\partial \psi_{\theta}}{\partial a} \\ \frac{\partial \psi_{\theta}}{\partial t} \end{bmatrix} = \begin{bmatrix} \psi[a \star (x_i - t)] \\ w \nabla_x \psi[a \star (x_i - t)] \star (x_i - t) \\ -w \nabla_x \psi[a \star (x_i - t)] \star a \end{bmatrix}$$

Letztendlich müssen wir noch einen Schritt tiefer gehen und $\nabla_x \psi(x^*)$ berechnen, wobei ψ das Mutter-Wavelet aus (3.26) ist.

$$\begin{aligned} \nabla_x \psi(x^*) &= \nabla_x \left[C_d (d - \|x^*\|_2^2) \exp\left(-\frac{\|x^*\|_2^2}{2}\right) \right] \\ &= \nabla_x \left[C_d (d - x^{*T} x^*) \exp\left(-\frac{x^{*T} x^*}{2}\right) \right] \\ &= C_d \left[(d - x^{*T} x^*) \exp\left(-\frac{x^{*T} x^*}{2}\right) (-x^*) + (-2x^*) \exp\left(-\frac{x^{*T} x^*}{2}\right) \right] \\ &= C_d (-d - 2 + x^{*T} x^*) \exp\left(-\frac{x^{*T} x^*}{2}\right) x^* \end{aligned}$$

Dabei ist C_d eine Konstante, die nur von der Dimension d abhängt. Wir werden C_d noch unter (7.105) berechnen. Damit ist alles so weit heruntergebrochen, wie benötigt für die algorithmische Umsetzung. $\mathbf{JF}(\theta)$ muss in jedem Iterationsschritt vollständig neu berechnet werden. Bei $F(\theta)$ muss nur der Anteil des gerade betrachteten Wavelets neu berechnet werden. Hier lässt sich mit etwas Geschicklichkeit viel Rechenzeit einsparen. Es ist möglich jedes Wavelet mehreren Iterationsschritten zu unterziehen, bevor man zum nächsten Wavelet übergeht. Diese Praxis hat sich aber nicht bewährt.

4.4 Konditionsprobleme mit Standardalgorithmen

In diesem Abschnitt werden einige Beispiele betrachtet und die Probleme, die dort entstehen.

Problem 1: Ein gutartiges Problem

Sei $\mathbb{O}_1^N = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ein Trainingsdatensatz und sei

$$\tilde{f} : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{mit} \quad \tilde{f}(x) = (x_1^2 - x_2^2) \sin(5x_1)$$

Die 441 Vektoren x_i liegen dabei auf einem regulären Gitter mit einem Gitterabstand von 0,1 auf dem Quadrat $[-1, 1] \times [-1, 1]$. Weiterhin gilt: $y_i = f(x_i)$. Die Trainingsdaten sind dargestellt in Abbildung 4. Nach dem Trainingsdatensatz wurde eine Wavelet-Bibliothek erstellt, wie in Abschnitt 4.1 beschrieben. Sie hat eine Form wie (4.33) mit den vorgegebenen Konstanten $\sigma = 2$, $\beta = 1$ und $m_{min} = -5$. Auch affine Terme wurden hinzugefügt, wie in diesem Abschnitt erklärt. Weiterhin erfolgte eine Vorwärtsauswahl von Regressoren, wie in Abschnitt 4.2 beschrieben. Dabei wurden 67 Wavelets ausgewählt. Das Resultat sehen wir in Abbildung 5. Die Wurzel des mittleren quadratischen Fehlers

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N} \sum_{k=1}^N [y_k - f(x_k)]^2} \quad (4.62)$$

beträgt ca. 0,008, was nicht viel ist. Die Funktion sieht augenscheinlich glatt aus und ist offensichtlich eine gute Näherung an die Trainingsdaten, bis auf die vier markanten Spitzen. Diese vier Spitzen sind ein Ausdruck schlechter Kondition bei der Modellbildung. Wir benennen dieses Phänomen mit dem Wort „Spiking“. Führen wir jetzt noch 50 Iterationen des in Abschnitt 4.3 beschriebenen Levenberg-Marquardt-Algorithmus aus, so erhalten wir das Ergebnis in Abbildung 6. Es ist zu erkennen, dass sich die Glattheit der Kurve etwas verbessert hat. $RMSE \approx 0,002$, d.h. dieses Kriterium hat sich noch einmal deutlich verbessert. Zwei der Spikes sind fast ganz zurück gegangen, aber die anderen beiden sind geblieben. Auch bei noch mehr Iterationsschritten ändert sich daran grundsätzlich nichts. Das heißt, wir haben das Konditionsproblem beibehalten. Zugegebenermaßen könnte man hier das Problem leicht umgehen. Da die Funktion offensichtlich „gutartig“ ist, werden keine so kleinen Wavelet-Kandidaten benötigt. D.h. man könnte bei der Erzeugung der Wavelet-Bibliothek m_{min} einfach größer wählen und würde damit das Spiking vermeiden.

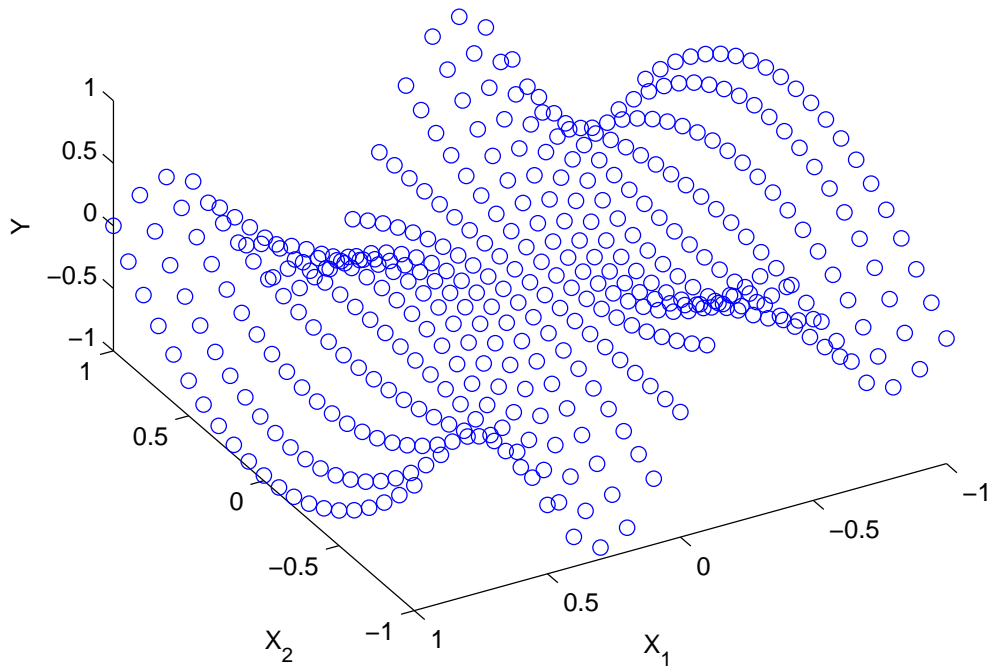
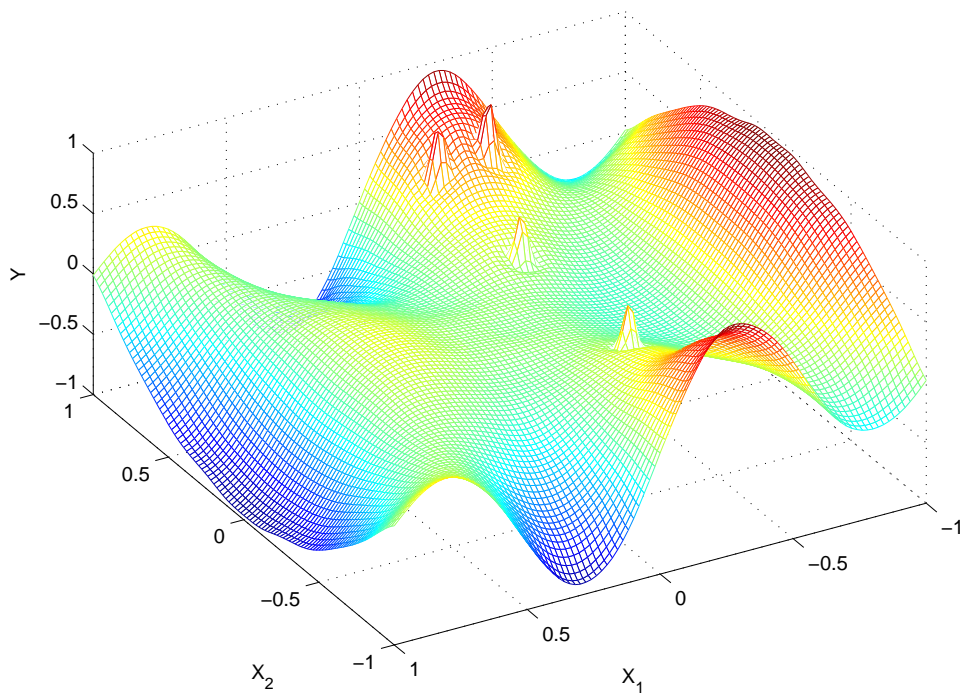


Abbildung 4: Trainingsdaten Problem 1

Abbildung 5: f nach Standard-Vorwärtsregressorauswahl bei Problem 1

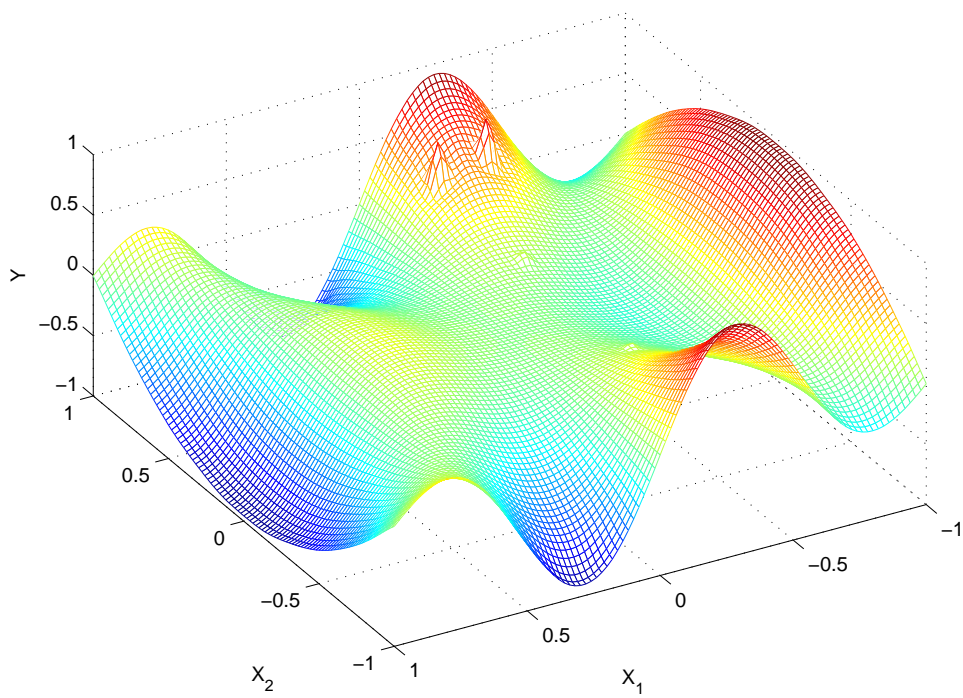


Abbildung 6: f nach Standard-Vorwärtsregressorauswahl und 50 Levenberg-Marquardt-Schritten bei Problem 1

Problem 2: Ein realistisches Problem

Sei $\mathbb{O}_1^N = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ein Trainingsdatensatz und sei

$$\tilde{f} : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{mit} \quad \tilde{f}(x) = \sin(x_1^2) \cos(x_2)$$

Die 500 Vektoren x_i sind dabei paarweise unabhängige Realisierungen eines Zufallsvektors X , der auf der zweidimensionalen Standardnormalverteilung basiert. Um genau zu sein, wurde folgendermaßen vorgegangen:

- x_1, \dots, x_{100} sind Realisierungen von $X_1 : X_1 \sim N((0, 0), \mathbf{I}_2)$
- x_{101}, \dots, x_{200} sind Realisierungen von $X_2 : X_2 \sim N((0, 1), \mathbf{I}_2)$
- x_{201}, \dots, x_{300} sind Realisierungen von $X_3 : X_3 \sim N((0, -1), \mathbf{I}_2)$
- x_{301}, \dots, x_{400} sind Realisierungen von $X_4 : X_4 \sim N((1, 0), \mathbf{I}_2)$
- x_{401}, \dots, x_{500} sind Realisierungen von $X_5 : X_5 \sim N((-1, 0), \mathbf{I}_2)$

Dabei steht \mathbf{I}_2 für die 2×2 Einheitsmatrix. Zusätzlich führen wir Fehlerterme $e_i \in \mathbb{R}$, $i = 1 \dots 500$ ein, mit

- e_1, \dots, e_{500} sind paarweise unabhängige Realisierungen einer Zufallsvariablen $E : E \sim N(0, 0.1^2)$

Weiterhin gilt: $y_i = \tilde{f}(x_i) + e_i$. Die Trainingsdaten sind in Abbildung 7 dargestellt. Nach dem Trainingsdatensatz wurde, genau wie in Problem 1, eine Wavelet-Bibliothek mit den vorgegebenen Konstanten $\sigma = 2$, $\beta = 1$ und $m_{min} = -5$ erzeugt. Genau wie in Problem 1 erfolgte danach eine Vorwärtswahl von Regressoren. Dabei wurden 45 Wavelets ausgewählt. Das Resultat sehen wir in Abbildung 8. Die Wurzel des mittleren quadratischen Fehlers ergibt sich nach (4.62) zu $RMSE \approx 0,12$, was an sich nicht schlecht ist. Unter der hypothetischen Annahme $f = \tilde{f}$ würden wir auf Grund der Fehlerterme e_i mit einem $RMSE = 0,1$ rechnen. Bei bloßer Betrachtung des Wertebereichs in Abbildung 8 fällt auf, dass dieser das Intervall $[-1, 1]$ bei weitem sprengt. Auch hier haben wir wieder das Spiking Problem. Anders als in Problem 1 ist hier nicht offensichtlich, dass man keine kleinen Wavelets, d.h. Wavelets mit kleiner Ausdehnung, benötigt. Zudem sehen wir jetzt, dass auch große Wavelets zum Spiking neigen können. Offensichtlich ist $RMSE$ hier kein hinreichend gutes Maß für die Güte der Approximation f an \tilde{f} . Deshalb haben wir ein Evaluations-Set \mathbb{O}_2^N konstruiert und zwar genau so, wie das Trainingsset. Es basiert auf den gleichen Verteilungen und der gleichen Funktion \tilde{f} , nur mit anderen, zufälligen Realisierungen für die x_i und e_i , $i = 1, \dots, 500$. Auch hier wurde die Wurzel des mittleren quadratischen Fehlers mit dem Wavelet-Netzwerk f analog zu (4.62) berechnet. Damit ergab sich: $Eval - RMSE \approx 0,47$. $Eval - RMSE$ ist ein zweckmäßiges Maß für die Güte des Wavelet-Netzwerkes f , weil letzteres an den Stellen ausgewertet wird, die relevant sind. Oder andersherum argumentiert: An

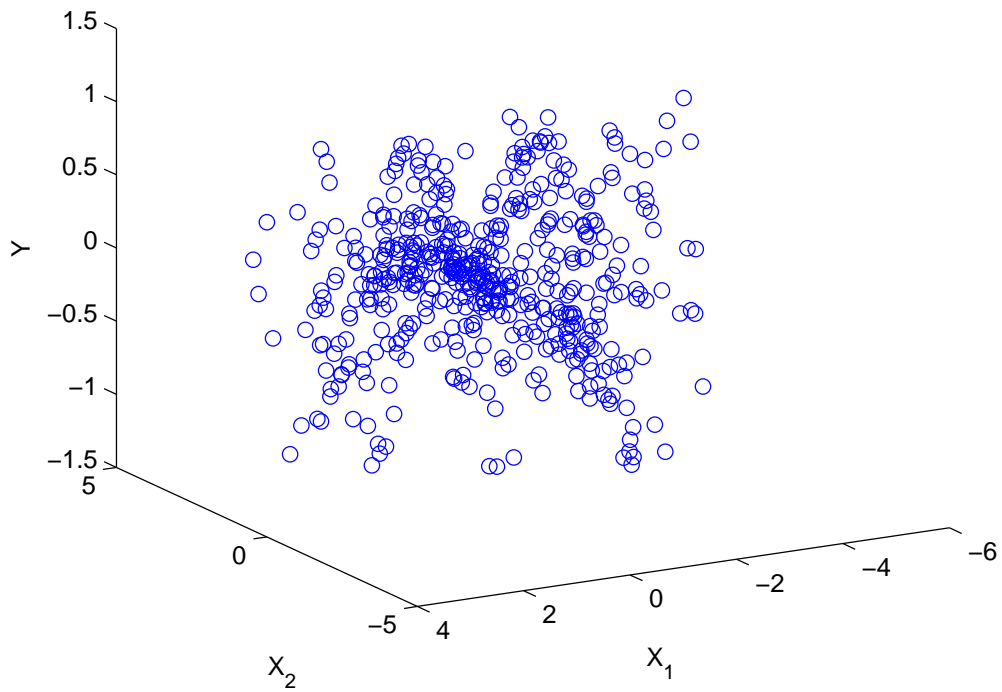


Abbildung 7: Trainingsdaten Problem 2

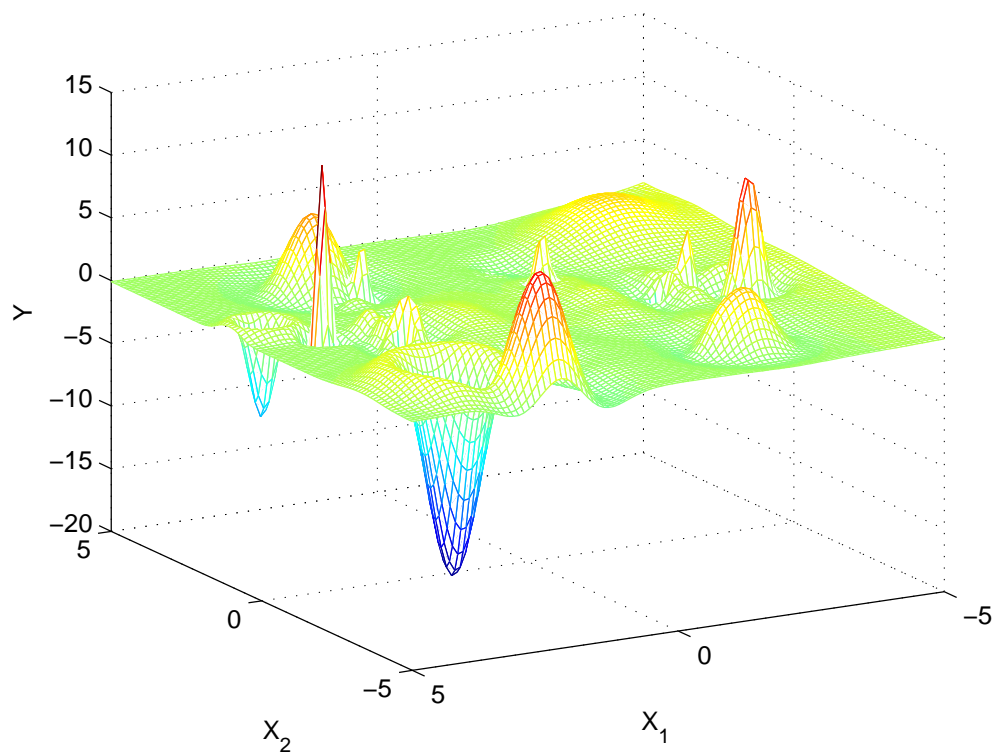


Abbildung 8: f nach Standard-Vorwärtsregressorauswahl bei Problem 2

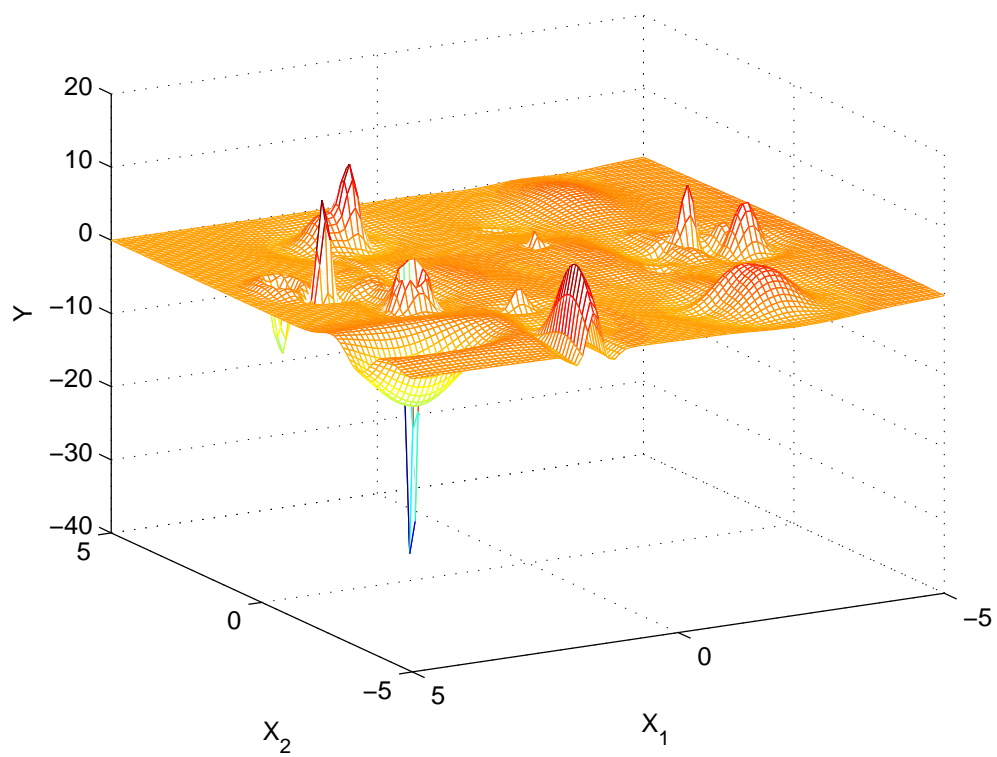


Abbildung 9: f nach Standard-Vorwärtsregressorauswahl und 50 Levenberg-Marquardt-Schritten bei Problem 2

x -Werten, die nie oder kaum angenommen werden, weil dort nur eine geringe Verteilungsdichte der Zufallsvariablen X vorherrscht, darf f ruhig stärker von \tilde{f} abweichen. Führen wir jetzt wieder, genau wie in Problem 1 beschrieben, 50 Iterationen des Levenberg-Marquardt Algorithmus aus, so erhalten wir das Ergebnis in Abbildung 9. $RMSE \approx 0,08$ und $Eval - RMSE \approx 0,40$. Damit haben sich diese beiden Kriterien leicht verbessert. Man sieht aber in der Grafik, dass sich das Spiking teilweise noch weiter ausgeprägt hat. Führt man weitere Iterationsschritte durch, dann verstärkt sich das Spiking weiter. Auch $Eval - RMSE$ wird dann wieder schlechter und steigt über eins. Das heißt, wir haben mit den Standardalgorithmen ein immenses Konditionsproblem.

Problem 3: Ein schwieriges Problem

Sei $\mathbb{O}_1^N = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ein Trainingsdatensatz und sei

$$\tilde{f} : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{mit} \quad \tilde{f}(x) = \begin{cases} 0 & \text{für } x_1 + x_2 \leq 0 \\ 1 & \text{sonst} \end{cases}$$

Die 441 Vektoren x_i liegen dabei genau wie in Problem 1 auf einem regulären Gitter mit einem Gitterabstand von 0,1 auf dem Quadrat $[-1, 1] \times [-1, 1]$. Weiterhin gilt: $y_i = \tilde{f}(x_i)$. Die Trainingsdaten sind dargestellt in Abbildung 10. Nach dem Trainingsdatensatz wurde, genau wie in Problem 1 und 2, eine Wavelet-Bibliothek mit den vorgegebenen Konstanten $\sigma = 2$, $\beta = 1$ und $m_{min} = -5$ erzeugt. Genau wie in Problem 1 und 2 erfolgte danach eine Vorwärtsauswahl von Regressoren. Dabei wurden 68 Wavelets ausgewählt. Das Resultat sehen wir in Abbildung 11. Die Wurzel des mittleren quadratischen Fehlers ergibt sich nach (4.62) zu $RMSE \approx 0,032$. Mit einem besonders guten Resultat war bei diesem Problem nicht zu rechnen, weil \tilde{f} eine Sprungfunktion ist und f immer eine beliebig oft stetig differenzierbare Funktion. Die Form der anzupassenden Funktion korreliert in keinsten Weise mit der Form der Regressoren, also der Form von radialsymmetrischen C^∞ -Wavelets. Allerdings ist das Resultat noch schlechter als erhofft. Man kann extremes Spiking an der Sprungkante beobachten. Auf den Halbebenen auf denen \tilde{f} konstant ist, wird die Funktion durch das Wavelet-Netzwerk f gut beschrieben. Führen wir jetzt wieder, genau wie in Problem 1 beschrieben, 50 Iterationen des Levenberg-Marquardt-Algorithmus aus, so erhalten wir das Ergebnis in Abbildung 12. $RMSE \approx 0,010$. Damit hat sich die Zielfunktion um Faktor 3 verbessert. Das Spiking Problem ist aber genau so erhalten geblieben. Auch hier zeigt sich deutlich, dass die Minimierung von MSE bzw. von $RMSE$ allein auf ein sehr schlecht konditioniertes Modellbildungsproblem führt.

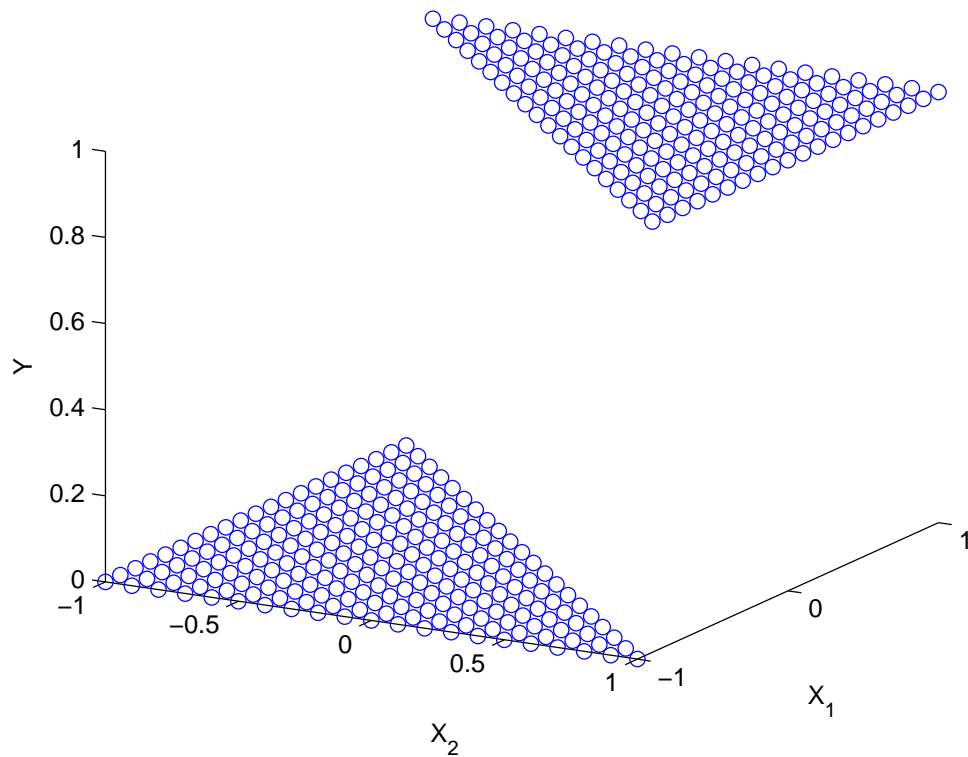


Abbildung 10: Trainingsdaten Problem 3

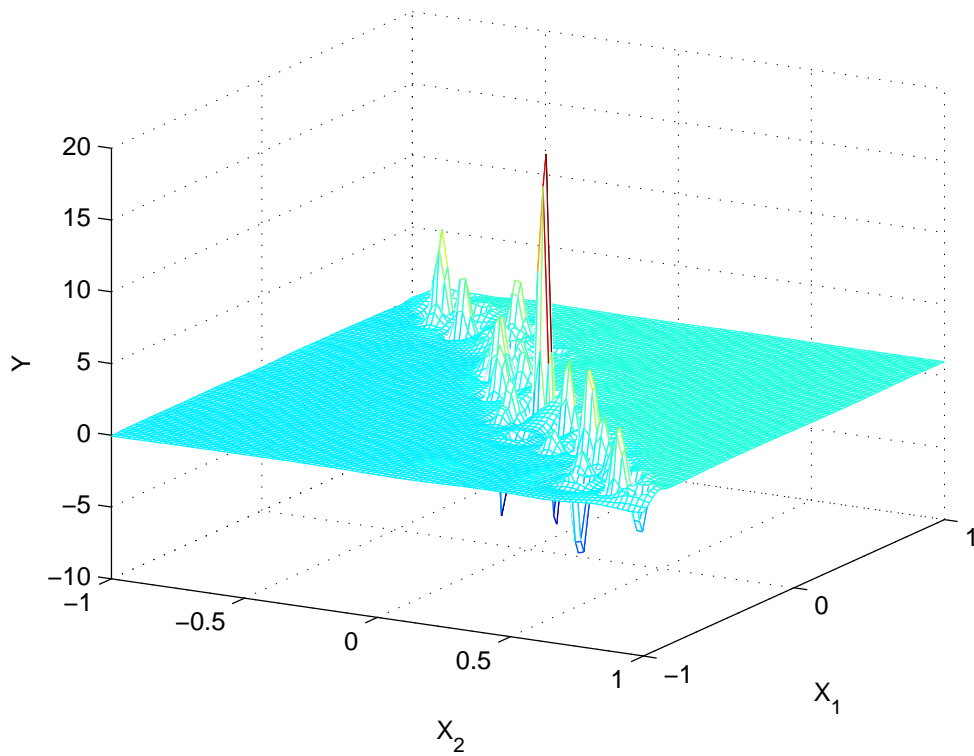


Abbildung 11: f nach Standard-Vorwärtsregressorauswahl bei Problem 3

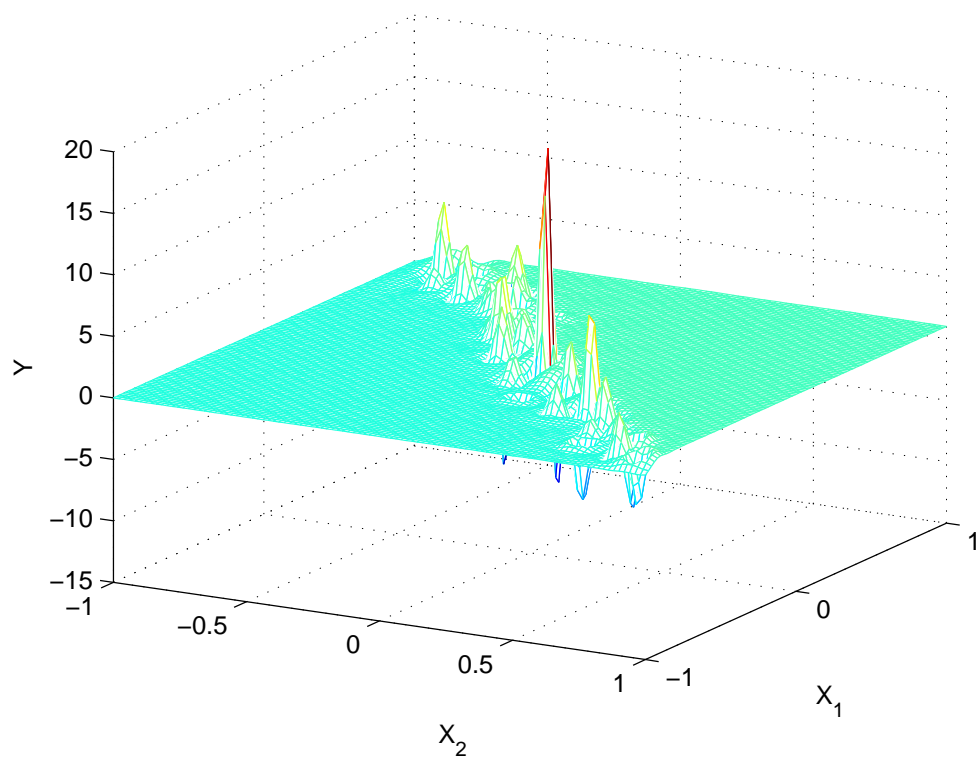


Abbildung 12: f nach Standard-Vorwärtsregressorauswahl und 50 Levenberg-Marquardt-Schritten bei Problem 3

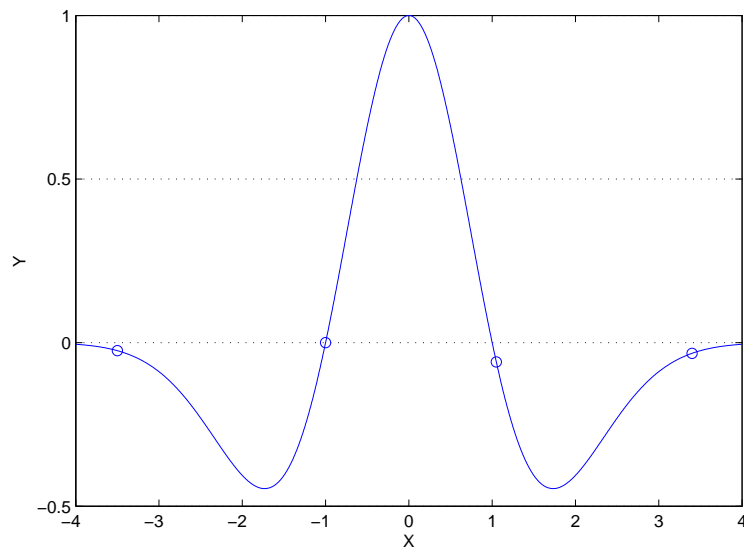


Abbildung 13: Entstehung von Spiking

Grund für das Spiking

Der Grund für das „Spiking“-Phänomen lässt sich am besten anhand von Abbildung 13 erklären. In dieser Skizze sieht man vier Trainingsdatenpunkte, die alle recht nahe bei der 0-Linie liegen. Wenn nun ein Regressorkandidat genau an den Stützstellen, das heißt an den x -Werten der Trainingsdaten, sehr kleine Werte hat, dann entsteht eine gewisse Hebelwirkung. Das Wavelet wird „herausgehobelt“. Man könnte auch sagen: Die Modellbildung ist schlecht konditioniert. Es entsteht auf jeden Fall ein schlechtes Modell, denn wie man leicht nachvollziehen kann, wäre im gezeigten Beispiel die Nullfunktion eine wesentlich bessere Näherung an die Daten als dieses Wavelet.

5 Verbesserung von Standardalgorithmen

Die gerade beobachteten Konditionsprobleme machen die Notwendigkeit robuster Algorithmen für die Erstellung eines erweiterten Wavelet-Netzwerkes deutlich. Deshalb wollen wir in einem ersten Ansatz jeweils eine Lösung für jedes der drei in der Teilaufgabengliederung unter 3.5 beschriebenen Teilprobleme erstellen. Dementsprechend sollen hier die folgenden neuen Algorithmen eingeführt werden:

1. eine Cluster-Bibliothek basierend auf dem K-means Algorithmus
2. ein robustifizierter Algorithmus zur Vorwärts-Regressorauswahl mit nachfolgender Redundanzreduzierung mit Hilfe der Rückwärtselimination von Regressoren
3. eine robustifizierte Version des Levenberg-Marquardt-Algorithmus

5.1 Eine Cluster-Bibliothek basierend auf dem K-means-Algorithmus

Bislang wurde bei der Erstellung der Wavelet-Bibliothek wie in 4.1 vorgegangen. Dabei wurden Wavelets verschiedener Translationen und Dilationen jeweils auf einem regulären Gitter angeordnet. Hier soll ein ganz neuer Ansatz verfolgt werden.

Nehmen wir an, wir wollen die Trainingsdaten \mathbb{O}_1^N in M Cluster ($M \leq N$) aufteilen, d.h. in M lokale Bereiche zusammenfassen. Nehmen wir weiter an, wir haben M Cluster mit den Clusterzentren

$$Z = \{z_1, z_2, \dots, z_M\}, \quad z_i \in \mathbb{R}^d$$

Diese Clusterzentren können z.B. mit den x -Werten zufällig ausgewählter Trainingspunkte initialisiert werden. Jeder Trainingsdatenpunkt ist genau einem der Cluster zugeordnet. Diese Zuordnung wird beschrieben durch die Zuordnungsfunktion

$$\eta : \{1, \dots, N\} \rightarrow \{1, \dots, M\} \quad \text{mit} \quad \eta(i) = j \quad (5.63)$$

Das bedeutet der i -te Trainingspunkt gehört zum j -ten Cluster.

Das Ziel besteht darin, den mittleren quadratischen Abstand zwischen den x -Werten der Trainingsdaten und den Clusterzentren der jeweils diesen Trainingspunkten zugeordneten Cluster zu minimieren. Die Aufgabe lautet also: finde

$$\arg \min_{Z, \eta} \sum_{i=1}^N \|x_i - z_{\eta(i)}\|^2$$

Dem entsprechend führen wir die Zielfunktion

$$L(Z, \eta) = \sum_{i=1}^N \|x_i - z_{\eta(i)}\|^2 \quad (5.64)$$

ein. Natürlich wäre auch hier eine erschöpfende globale Optimierung undenkbar aufwändig. Wir wollen deshalb ein iteratives Verfahren verwenden, das uns zu einer suboptimalen Lösung führt.

Angenommen Z sei fest vorgegeben, dann sollte die optimale Zuordnungsfunktion (5.63) jedem Punkt das nächstgelegene Clusterzentrum zuweisen, also

$$\eta(i) = \arg \min_{j \in \{1, \dots, M\}} \|x_i - z_j\|_2$$

Wenn hingegen $\eta(\cdot)$ als fest angenommen wird, dann sollten die Clusterzentren z_j das arithmetische Mittel der Trainingsdaten sein, die zum j -ten Cluster gehören, also

$$z_j = \frac{\sum_{i: \eta(i)=j} x_i}{N_j}$$

wobei N_j die Anzahl der zum j -ten Cluster zugeordneten Trainingsdatenpunkte ist.

Der K-means-Algorithmus führt im Wesentlichen immer wieder alternierend zwei Schritte durch

1. Zu einer gegebenen festen Menge Z von Clusterzentren wird $\eta(\cdot)$ optimiert, indem jeder Trainingsdatenpunkt dem (Im Sinne der euklidischen Norm) nächsten Clusterzentrum zugewiesen wird.
2. Die Clusterzentren werden erneuert als arithmetisches Mittel der dem Cluster zugeordneten Trainingsdatenpunkte.

Der Algorithmus konvergiert immer, da die Zielfunktion (5.64) in jedem Schritt fällt und nach unten beschränkt ist. Normalerweise konvergiert er auch schnell genug. Der Algorithmus ist am Ziel, wenn die Zielfunktion nicht weiter abnimmt. Hier noch einmal eine kompakte Darstellung des Algorithmus

Algorithmus 5.1 *K-means*

Sei $M \leq N$ und \mathbb{O}_1^N gegeben

Schritt 0: *initialisiere*

$$Z = (z_1, \dots, z_M), \quad z_i \in \mathbb{R}^d$$

mit zufällig gewählten, paarweise verschiedenen x_i aus \mathbb{O}_1^N

Schritt i , Teil 1: *erzeuge oder aktualisiere*

$$\eta : \eta(k) = \arg \min_{j \in \{1, \dots, M\}} \|x_k - z_j\|_2$$

Schritt i , Teil 2: *aktualisiere*

$$Z = \left\{ z_j \mid N_j \neq 0 \wedge z_j = \frac{\sum_{k:\eta(k)=j} x_k}{N_j} \right\}$$

wobei $N_j = |\eta^{-1}(\{j\})|$ die Anzahl der zum j -ten Cluster zugeordneten Trainingsdatenpunkte ist

Abbruchbedingung: *breche den Algorithmus ab, sobald die Zielfunktion*

$$L(Z, \eta) = \sum_{k=1}^N \|x_k - z_{\eta(k)}\|_2^2$$

nicht mehr weiter sinkt

Für die Erstellung einer Wavelet-Bibliothek wird dieser Cluster-Algorithmus nun folgendermaßen benutzt: Der K-means-Algorithmus, wie eben beschrieben, wird $\lfloor N/2 \rfloor$ mal durchlaufen, jeweils mit $M = 1, \dots, \lfloor N/2 \rfloor$. Immer nach einem Durchlauf des K-means-Algorithmus wird für jedes Cluster j , das mindestens 2 Punkte enthält

$$d_j = \max_{k:\eta(k)=j} \|x_k - z_j\|_2$$

berechnet. Danach wird ein neues Wavelet ψ_j der Form

$$\psi_j(x) = w_j \psi[a_j \star (x - t_j)] \quad \text{mit} \quad a_j = \begin{bmatrix} c^{-1} d_j^{-1} \\ \vdots \\ c^{-1} d_j^{-1} \end{bmatrix}, t_j = z_j$$

hinzugefügt, sofern ein solches Wavelet nicht bereits in der Bibliothek vorhanden ist. Dabei wurde $c = 2, 5$ verwendet. Die w_j spielen noch keine Rolle und wurden deshalb willkürlich zu eins gesetzt.

Vorteile der Cluster-Bibliothek:

- Auf Grund der Konstruktionsweise entstehen erst gar keine Regressorkandidaten, die bei der Regressorauswahl zum Spiking neigen würden.
- Bei Problemen hoher Dimension d entstehen bei einer Wavelet-Bibliothek auf einem regulären Gitter zu viele Regressorkandidaten. Bei einer Cluster-Bibliothek ist die Anzahl der Regressorkandidaten per se nicht von der Dimensionalität d des Problems abhängig.

Nachteile der Cluster-Bibliothek:

- Die theoretischen Erkenntnisse der Wavelet-Theorie werden nicht verwendet. Die Frame-Eigenschaft einer Wavelet-Familie auf einem regulären Gitter kann nicht als Grundlage angesehen werden wie im Standardverfahren. Insbesondere Funktionen, die am Rand ihres relevanten, mit Trainingsdaten bestückten Bereiches zu besonders großen oder kleinen Werten streben, können mit der Cluster-Bibliothek oft weniger gut dargestellt werden als mit der Standardmethode.

5.2 Ein robustifizierter Algorithmus zur Vorwärts-Regressorauswahl

Der hier beschriebene Algorithmus ist eine leicht abgewandelte Form des unter 4.2 beschriebenen Algorithmus zur sukzessiven Regressorauswahl mit schrittweiser Orthogonalisierung. Die Kenntnis von Algorithmus 4.1 wird hier vorausgesetzt. Die zentrale Stelle der Regressorauswahl in Algorithmus 4.1 ist die Zeile: finde

$$l_i = \arg \max_{j \in I_i} \frac{\left[(p_j^{(i)})^T y \right]^2}{(p_j^{(i)})^T p_j^{(i)}} \quad (5.65)$$

An genau dieser Stelle muss eingegriffen werden, um Wavelets mit schlechter Kondition bei der Regressorauswahl zu unterdrücken. Die affinen Terme wurden, wie bereits beschrieben, gleich als erstes ausgewählt. Wir können uns bei den folgenden Betrachtungen folglich ausschließlich auf die Auswahl von Wavelets konzentrieren. Die Grundlage des Algorithmus ist eine zuvor erstellte Wavelet-Bibliothek der Form (4.34), also ausgeschrieben:

$$W_{wav} = \{ \psi_i : \mathbb{R}^d \rightarrow \mathbb{R} | \psi_i(x) = w_i \psi [a_i \star (x - t_i)] \quad , i = 1, \dots, L \}$$

Wir definieren uns zu den Wavelets ψ_i nun ungewichtete Norm-Wavelets

$$\psi_i^* : \mathbb{R}^d \rightarrow \mathbb{R} | \psi_i^*(x) = \psi [a_i \star (x - t_i)] \quad , i = 1, \dots, L$$

Weiterhin definieren wir zu jedem Wavelet ψ_i eine heuristisch motivierte Konditionszahl κ_i mit

$$\kappa_i = \sum_{j=1}^N \psi_i^*(x_j)^2 \quad , i = 1, \dots, L$$

,wobei die x_i aus \mathbb{O}_1^N stammen. Ein Spiking-anfälliges Wavelet ψ_i , wie in Abbildung 13, hat ein kleines κ_i . Ein Wavelet ψ_j , das Stützpunkte dort hat, wo auch betragsmäßig hohe Werte des Wavelets liegen, erzielt ein höheres κ_j . Solch ein Wavelet neigt auch weniger zum Spiking. Man beachte, dass die κ_i , $i = 1, \dots, L$ nur einmal zu Beginn des Algorithmus berechnet werden müssen. Sie können gewissermaßen als Konditionsindikator der Wavelets ψ_i aus W_{wav} betrachtet werden. Nun ändern wir in Algorithmus 4.1 die Gleichung (5.65) folgendermaßen ab:

$$l_i = \arg \max_{j \in I_i} \frac{\left[(p_j^{(i)})^T y \right]^2}{(p_j^{(i)})^T p_j^{(i)}} \ln(\kappa_i) \quad (5.66)$$

Mit dem Zusatzterm $\ln(\kappa_i)$ in (5.66) wird erreicht, dass Wavelets, die zum Spiking neigen würden, benachteiligt werden. Die Motivation für den eingebauten Logarithmus ist Intuition. Der Logarithmus ist für kleine Argumente sehr steil und für große Argumente sehr flach. Bei größeren, gutartigen Wavelets sollen sich Unterschiede in den κ_i nicht so sehr auf die Regressorauswahl auswirken. Die bestmögliche Anpassung an die Trainingsdaten ist hier das wichtigste Ziel. Bei Wavelets mit kleinem κ_i tritt durch die Steilheit des Logarithmus die Anpassung an die Trainingsdaten in den Hintergrund. Die Vermeidung von Spiking ist hier vorrangig. Dieser abgewandelte Algorithmus zur Vorwärts-Auswahl von Regressoren wird, wie wir noch sehen werden, gute Resultate zeigen.

Es ist in fast jedem Fall sinnvoll, nach der Vorwärtsauswahl von Regressoren zusätzlich den Standardalgorithmus zur Rückwärtselimination, wie beschrieben in Abschnitt 4.2, auszuführen. Bei der sukzessiven Auswahl von Regressoren kann es dazu kommen, dass auch solche Wavelets genommen werden, die sich später wieder als entbehrlich erweisen. Die bei der sukzessiven Regressorauswahl gewonnenen Vektoren v_i sind in jedem Fall linear unabhängig. Das heißt die Rückwärtselimination ist auch immer ausführbar. Die Zielfunktion (4.57) wird durch dieses Vorgehen nochmals minimiert. Unter Umständen können noch einmal über ein Viertel der Regressoren entfallen. Natürlich steigt in jedem Fall der mittlere quadratische Fehler, wie definiert in (3.23), aber die Verringerung der Parameter in f kann zu einer höheren Robustheit führen.

5.3 Eine robustifizierte Version des Levenberg-Marquardt-Algorithmus

Der hier beschriebene Algorithmus ist eine abgewandelte Form des unter 4.3 beschriebenen Algorithmus zur Regressoroptimierung. Die Kenntnis dieses Algorithmus wird hier vorausgesetzt. Der Eingriff erfolgt bei Gleichung (4.58), die da lautet:

$$\eta(\theta) = \|F(\theta)\|_2^2 = F(\theta)^T F(\theta)$$

Hier fügen wir eine Konditionsstrafe $p(\theta)$ und eine Konstante c ein, und ersetzen die Gleichung durch

$$\eta(\theta) = \|F(\theta)\|_2^2 + cp(\theta) = F(\theta)^T F(\theta) + cp(\theta)$$

Wie beim Standardalgorithmus bereits beschrieben, umfasst θ alle Parameter eines Wavelets wie in Gleichung (4.61), also

$$\psi_\theta = w\psi [a \star (x - t)]$$

Wir führen nun, genau wie im robustifizierten Algorithmus zur Regressorauswahl, ein ungewichtetes Norm-Wavelet ein.

$$\psi_\theta^* = \psi [a \star (x - t)]$$

Sei nun

$$p(\theta) = \frac{1}{q(\theta)} = \frac{1}{\sum_{i=1}^N \psi_\theta^*(x_i)^4}$$

Dabei stammen die x_i aus \mathbb{O}_1^N . Die Motivation dieser Konditionsstrafe ist ganz ähnlich zu Stande gekommen wie bei der Regressorauswahl. $q(\theta)$ ist groß für breite Wavelets, die viele Trainingspunkte mit den Bereichen überdecken, wo sie hohe Werte haben. Diese Wavelets neigen nicht zum Spiking. Wavelets die zum Spiking neigen, besitzen kleinere Werte $q(\theta)$. Die Funktion $y = 1/x$ ist steil bei kleinen x -Werten und flach bei großen x -Werten. Das heißt, bei „gutartigen“ Wavelets steht die Verminderung des mittleren quadratischen Fehlers MSE im Vordergrund. Bei Wavelets die zum Spiking neigen steht die Verminderung der Konditionsstrafe $p(\theta)$ im Mittelpunkt. Ursprünglich stand in der Summe der Term $\psi_\theta^*(x_i)^2$. Die vierte Potenz zeigt aber ein besseres Verhalten. Wir benötigen weiterhin den Gradienten und eine Näherung an die Hessematrix der Konditionsstrafenfunktion. Es gilt:

$$\begin{aligned} \nabla q(\theta) &= \sum_{i=1}^N [4\psi_\theta^*(x_i)^3 \nabla \psi_\theta^*(x_i)] \\ \nabla p(\theta) &= -\nabla q(\theta) p(\theta)^2 \\ \nabla^2 p(\theta) &\approx \nabla q(\theta) 2p(\theta) \nabla^T q(\theta) p^2(\theta) = 2p(\theta)^3 \nabla q(\theta) \nabla^T q(\theta) \end{aligned}$$

Der Nabla-Operator bezieht sich dabei jeweils auf θ . Um diese Größen auch tatsächlich zu berechnen, müssen wir noch $\nabla \psi_\theta^*(x_i)$ herunterbrechen:

$$\nabla_\theta \psi_\theta^*(x_i) = \begin{bmatrix} \frac{\partial \psi_\theta^*}{\partial a} \\ \frac{\partial \psi_\theta^*}{\partial t} \end{bmatrix} = \begin{bmatrix} 0 \\ \nabla_x \psi [a \star (x_i - t)] \star (x_i - t) \\ -\nabla_x \psi [a \star (x_i - t)] \star a \end{bmatrix}$$

Das weitere Herunterbrechen $\nabla_x \psi(x^*)$ wurde bereits im Standardalgorithmus unter 4.3 gezeigt. Mit diesem Wissen im Hinterkopf werden in (4.59) der Gradient von η und die Näherung an die Hessematrix von η aus dem Standardalgorithmus folgendermaßen ergänzt:

$$\begin{aligned} g_i &= 2\mathbf{J}\mathbf{F}(\theta_i)^T F(\theta_i) - \nabla q(\theta)p(\theta)^2 = \nabla \eta(\theta_i) \\ \mathbf{H}_i &= 2\mathbf{J}\mathbf{F}(\theta_i)^T \mathbf{J}\mathbf{F}(\theta_i) + 2p(\theta)^3 \nabla q(\theta) \nabla q(\theta)^T \approx \nabla^2 \eta(\theta_i) \end{aligned} \quad (5.67)$$

Das weitere Vorgehen entspricht genau dem Vorgehen im Standardalgorithmus, wie beschrieben in 4.3.

5.4 Ergebnisse der robustifizierten Algorithmen

In diesem Abschnitt werden die gleichen Beispiele betrachtet wie unter 4.4 mit den Standardalgorithmen.

Problem 1: Das gutartige Problem

Der Trainingsdatensatz und die Funktion \tilde{f} wurden bereits im Abschnitt 4.4 eingeführt, in welchem die Resultate der Standardalgorithmen besprochen wurden. Der Trainingsdatensatz ist in Abbildung 4 graphisch dargestellt. Zur Verbesserung der Ergebnisse gegenüber den Standardalgorithmen, wurden hier zwei der drei verbesserten Verfahren angewendet. Die Wavelet-Bibliothek wurde noch herkömmlich erstellt, wie beschrieben in 4.1. Sie hat eine Form wie (4.33) mit den vorgegebenen Konstanten $\sigma = 2$, $\beta = 1$ und $m_{min} = -5$. Auch affine Terme wurden, wie in diesem Abschnitt beschrieben, hinzugefügt.

Bei einem Trainingsdatensatz auf einem regulären Gitter ist es sinnvoller auch eine Wavelet-Bibliothek auf einem regulären Gitter zu verwenden als einen Cluster-Algorithmus. Danach erfolgte die neue, in 5.2 eingeführte, robustifizierte Vorwärtsauswahl von Regressoren. Dabei wurden 64 Wavelets ausgewählt. Das Resultat sehen wir in Abbildung 14. Es ergibt sich eine Wurzel des mittleren quadratischen Fehlers von $RMSE \approx 0,012$. Die Funktion sieht augenscheinlich glatt aus und ist offensichtlich eine gute Näherung an die Trainingsdaten. Das Spiking wurde erfolgreich unterdrückt. Führt man noch eine Rückwärtselimination von Regressoren durch, vermindert sich die Zahl der Wavelets von 64 auf 60. $RMSE$ steigt nur leicht, aber nicht in den ersten drei Nachkommastellen. Augenscheinlich ändert sich nichts. Führen wir jetzt noch 50 Iterationen des in 5.3 beschriebenen, verbesserten Levenberg-Marquardt-Algorithmus aus, so erhalten wir das Ergebnis in Abbildung 15. Es ist zu erkennen, dass sich die Glattheit der Kurve weiter verbessert hat. $RMSE \approx 0,002$, d.h. wir erreichen hier die gleiche Güte wie mit den Standardalgorithmen. Spiking ist nicht zu erkennen. Auch bei noch mehr Iterationsschritten ändert sich daran grundsätzlich nichts. Das heißt, wir sind bei diesem Problem mit den verbesserten Algorithmen erfolgreich. In diesem Fall gibt es nichts zu beanstanden.

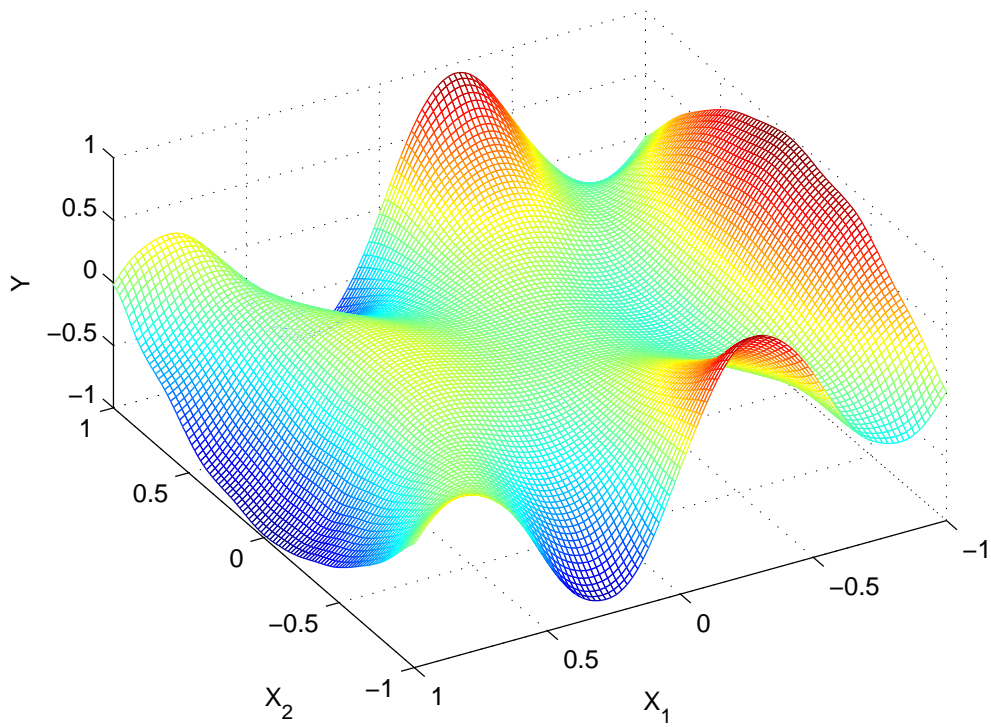


Abbildung 14: f nach robuster Vorwärtsregressorauswahl bei Problem 1

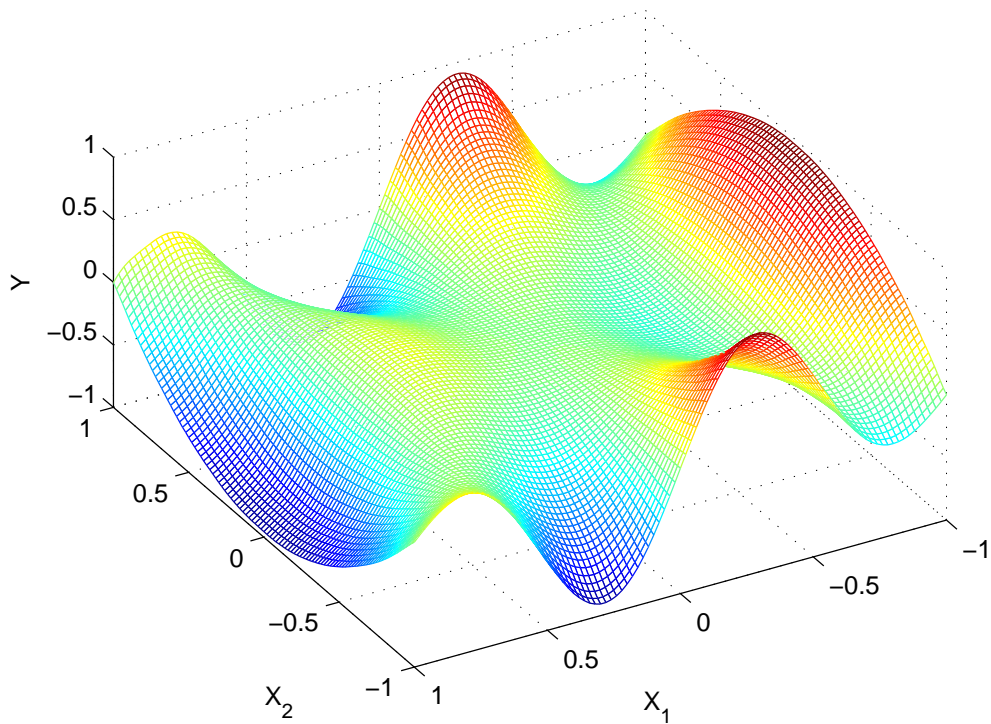


Abbildung 15: f nach robuster Vorwärtsregressorauswahl + Rückwärtselimination und 50 verbesserten Levenberg-Marquardt-Schritten bei Problem 1

Problem 2: Das realistische Problem

Hier war $\mathbb{D}_1^N = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ein Trainingsdatensatz mit $N = 500$ Trainingsdatenpunkten. Weiterhin war

$$\tilde{f} : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{mit} \quad \tilde{f}(x) = \sin(x_1^2) \cos(x_2)$$

Die 500 Vektoren x_i waren dabei Realisierungen eines Zufallsvektors X , der auf der Standardnormalverteilung basierte. Die genaue Verteilung wurde bereits unter 4.4 beschrieben. Des weiteren gab es bei dem Problem zufällige Fehlerterme $e_i \in \mathbb{R}$, $i = 1 \dots 500$, die unabhängige Realisierungen einer normalverteilten Zufallsvariablen mit Erwartungswert 0 und Varianz $0,1^2$ waren. Es gilt der Zusammenhang: $y_i = f(x_i) + e_i$. Die Trainingsdaten sind dargestellt in Abbildung 7.

Bei diesem Problem kamen alle drei robustifizierten Algorithmen zur Anwendung. Nach dem Trainingsdatensatz wurde, anders als in allen bisherigen Problemen, eine Cluster-Bibliothek erzeugt. Der zugehörige Algorithmus wurde in 5.1 beschrieben. Besonders bei statistisch verteilten Trainingsdaten ist ein solcher Ansatz nahe liegend. Die erzeugte Bibliothek umfasst 710 Wavelets.

Im nächsten Schritt wurde eine robuste Vorwärtsregressorauswahl gemäß 5.2 ausgeführt. Dabei wurden 36 Wavelets ausgewählt. Im Anschluss erfolgte noch eine Rückwärtselimination von Regressoren. Dabei wurde die Anzahl der Wavelets noch einmal auf 22 reduziert. Die Wurzel des mittleren quadratischen Fehlers $RMSE$ stieg dadurch leicht von 0,22 auf 0,24. Der mittlere Quadratische Fehler beim Auswertungsset $Eval - RMSE$ stieg dabei von 0,273 auf 0,278. Die Verschlechterungen durch den Wegfall von Wavelets sind also durchaus zu vernachlässigen. Unter der hypothetischen Annahme $f = \tilde{f}$ würden wir auf Grund der Fehlerterme e_i mit einem $RMSE = 0,1$ rechnen. Das Resultat sehen wir in Abbildung 16. Die Konturen der Funktion sind bereits gut erkennbar. Auch der Wertebereich ist im Rahmen. Spiking ist nicht zu erkennen.

Führen wir wiederum 50 Schritte des verbesserten Levenberg-Marquardt-Algorithmus durch, so erhalten wir das Ergebnis in Abbildung 17. Dabei fällt $RMSE$ auf 0,147 und $Eval - RMSE$ auf 0,212. Die Funktion sieht augenscheinlich besser aus als zuvor. Der mittlere quadratische Fehler des Auswertungsdatensatzes ist ungefähr doppelt so hoch wie sein theoretisches Minimum aufgrund der Fehlerterme e_i . Das ist bei der gegebenen Funktion \tilde{f} und der Verteilung der Trainingsdaten ein zufriedenstellendes Resultat. Dass das Wavelet-Netzwerk f in den Ecken der Darstellung sehr hohe bzw. niedrige Werte annimmt, stört dabei nicht. Wie man in Abbildung 18 erkennt, gibt es an diesen Stellen keine Trainingsdaten. Da der Auswertungsdatensatz auf der gleichen Verteilung beruht, sind diese Stellen aber auch nicht relevant. Das Ergebnis kann sich also sehen lassen.

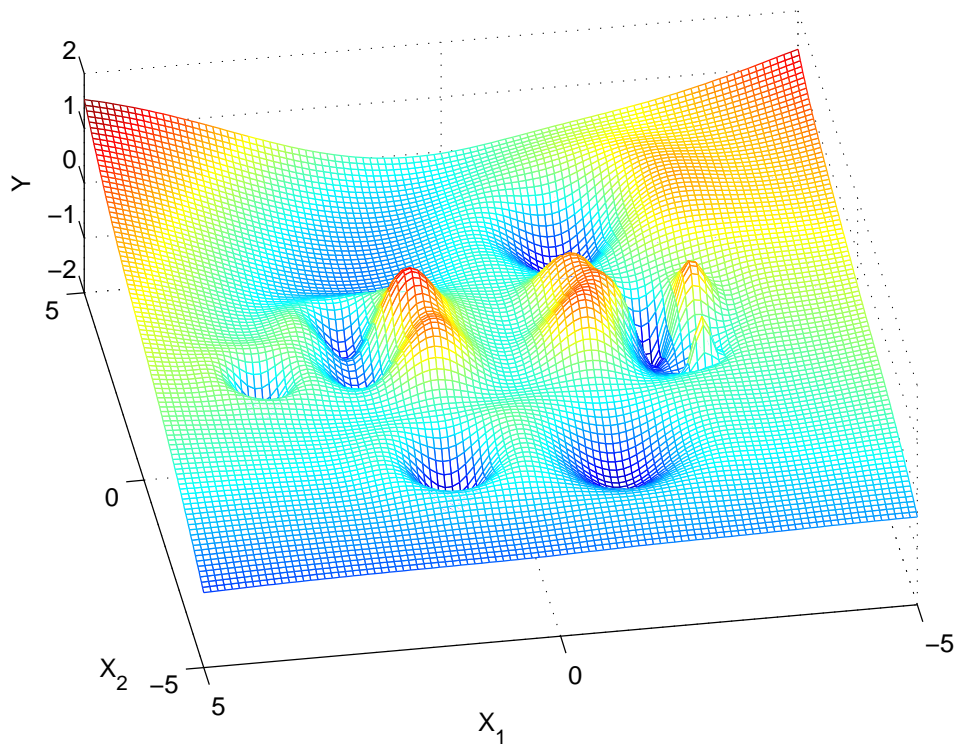


Abbildung 16: f nach robuster Vorwärtsregressorauswahl + Rückwärtselimination bei Problem 2

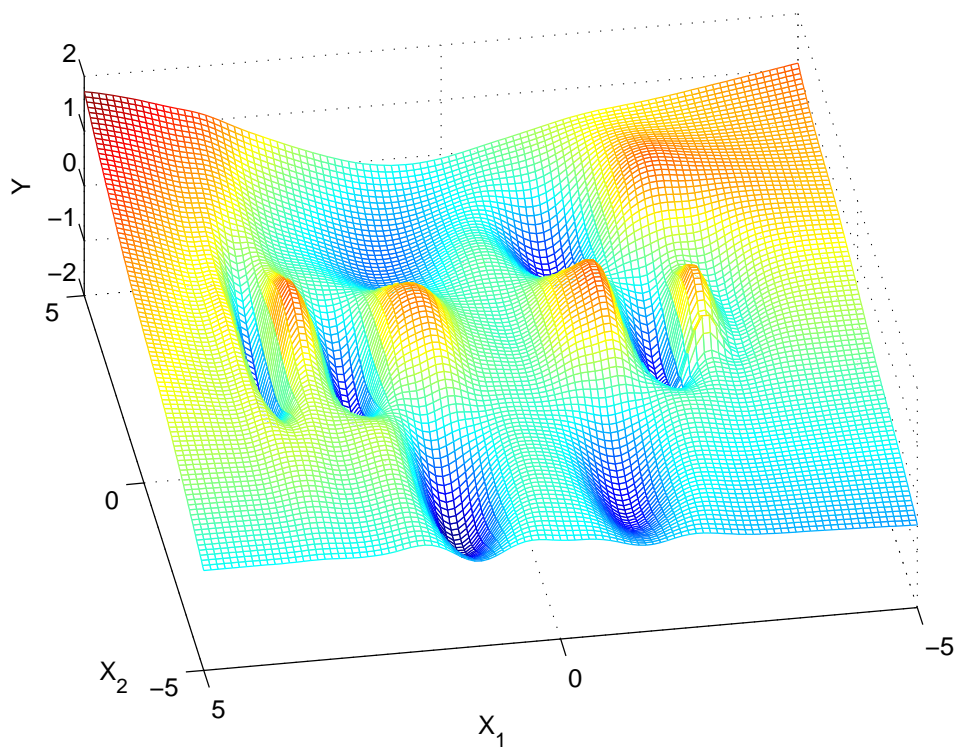


Abbildung 17: f nach robuster Vorwärtsregressorauswahl + Rückwärtselimination und 50 verbesserten Levenberg-Marquardt-Schritten bei Problem 2

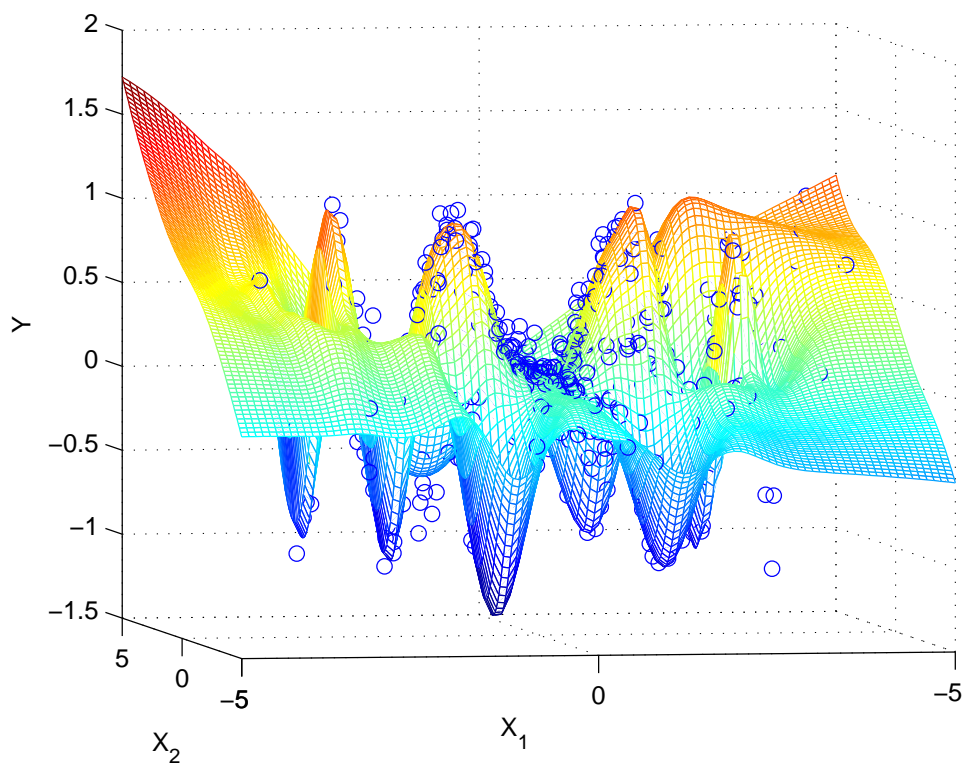


Abbildung 18: wie Abbildung 17 nur mit Trainingsdaten

Problem 3: Das schwierige Problem

Bei diesem Problem war $\mathbb{O}_1^N = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ein Trainingsdatensatz auf einem regulären Gitter mit einem Gitterabstand von 0,1 auf dem Quadrat $[-1, 1] \times [-1, 1]$ und

$$\tilde{f} : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{mit} \quad \tilde{f}(x) = \begin{cases} 0 & \text{für } x_1 + x_2 \leq 0 \\ 1 & \text{sonst} \end{cases}$$

Weiterhin gilt: $y_i = f(x_i)$

Die Trainingsdaten sind dargestellt in Abbildung 7.

Analog zu Problem 1 fanden zwei der drei robustifizierten Algorithmen Anwendung. Nach dem Trainingsdatensatz wurde eine herkömmliche Wavelet-Bibliothek mit den vorgegebenen Konstanten $\sigma = 2$, $\beta = 1$ und $m_{min} = -5$ erzeugt. Bei Trainingsdaten auf einem regulären Gitter ist ein Cluster-Algorithmus nicht zweckmäßig. Daher wurde auf die herkömmliche Wavelet-Bibliothek zurückgegriffen.

Darauf folgte aber eine robuste Vorwärtsregressorauswahl, wie in 5.2 beschrieben. Dabei wurden 70 Wavelets ausgewählt. Bei der folgenden Rückwärtselimination wurde diese Zahl auf 61 reduziert. Der mittlere quadratische Fehler *RMSE* stieg von 0,035 auf 0,040. Das Resultat sehen wir in Abbildung 19. Dafür, dass \tilde{f} eine Sprungfunktion ist und f immer eine stetige Funktion sein muss, ist das Resultat verblüffend gut. Vor allem wenn man bedenkt, wie es mit den Standardalgorithmen in Abbildung 11 aussah, ist die Verbesserung wesentlich. Spiking ist so gut wie nicht vorhanden. Bis auf eine leichte Welligkeit bei den Flächen approximiert f die Stufenfunktion sehr gut.

50 Schritte des robustifizierten Levenberg-Marquardt-Algorithmus führen zu dem Ergebnis in Abbildung 20. *RMSE* $\approx 0,006$. Wie man sieht, kommt das Spiking-Problem trotz des robusten Algorithmus wieder zurück. Allerdings nicht in so ausgeprägter Form wie beim Standardalgorithmus. Es wurde folglich eine Verbesserung erzielt, auch wenn das Optimum noch nicht erreicht ist. Zumindest sind die Flächen glatter geworden. Bei der Bewertung bleibt zu beachten, dass das Mexikanerhut-Wavelet prinzipiell ungeeignet ist um eine Stufenfunktion anzupassen. Es besteht allerdings die Vermutung, dass sich der Levenberg-Marquardt-Algorithmus noch besser robustifizieren lässt.

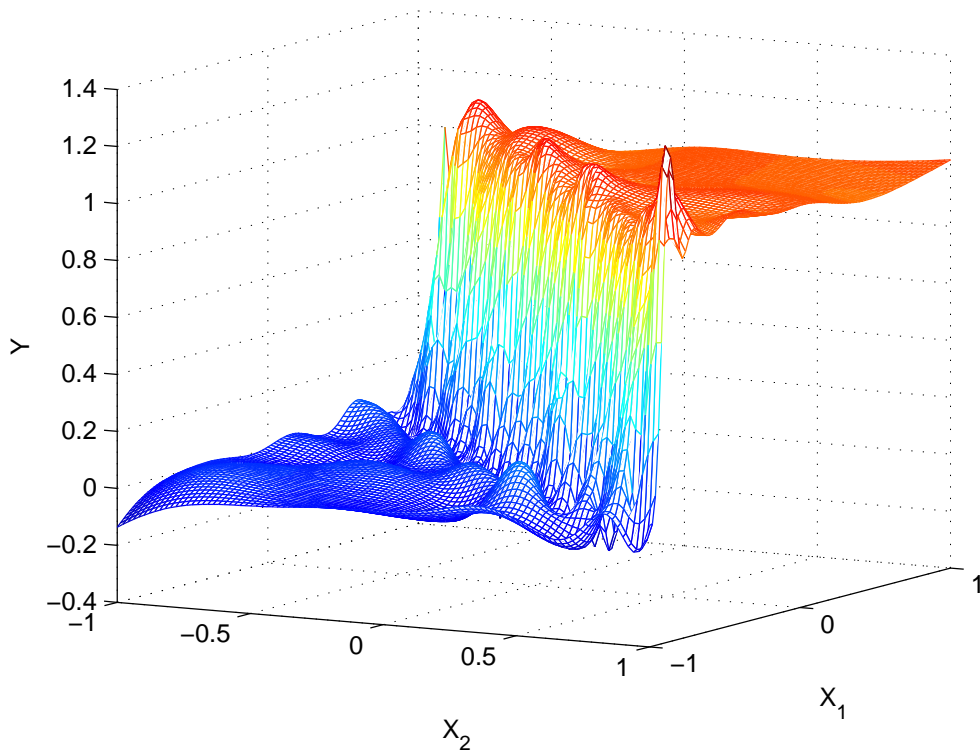


Abbildung 19: f nach robuster Vorwärtsregressorauswahl + Rückwärtselimination bei Problem 3

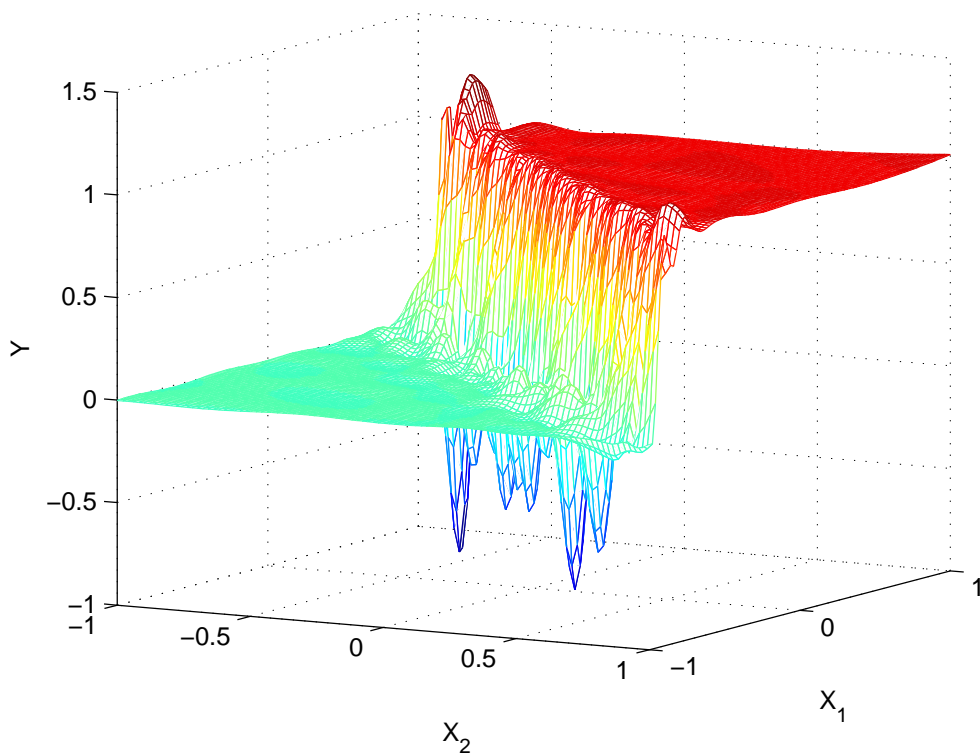


Abbildung 20: f nach robuster Vorwärtsregressorauswahl + Rückwärtselimination und 50 verbesserten Levenberg-Marquardt-Schritten bei Problem 3

6 Ein neues, statistisches Gütekriterium

Kapitel 6 und 7 bilden das Herzstück dieser Dissertation. Darin soll ein grundlegend neues Verfahren zur Erstellung von Wavelet-Netzwerken vorgestellt werden. Es basiert auf einem neuartigen, statistisch fundierten Gütekriterium zur Regressorauswahl. Die bisher betrachteten Verfahren zur Erstellung von Wavelet-Netzwerken bestanden stets aus drei Schritten.

1. Erstellung einer Wavelet-Bibliothek
2. Regressorauswahl
3. Regressoroptimierung

Dabei wurden im Zuge der Regressorauswahl heuristische Kriterien zur Auswahl des nächst besten oder Abwahl des nächst schlechtesten Wavelets herangezogen. Die Standardalgorithmen zielten ausschließlich auf eine maximale Verbesserung oder minimale Verschlechterung des mittleren quadratischen Fehlers ab. Bei den verbesserten Algorithmen wurde dieses Kriterium mit einfachen Mitteln leicht modifiziert. Die anschließende Regressoroptimierung erfolgte jeweils nach der gleichen, oder eine ganz ähnlichen Kriteriumsfunktion. Leider stand die Robustheit der Modellbildung auch bei den robustifizierten Algorithmen auf keiner soliden mathematischen Grundlage.

Deshalb werden wir in diesem Kapitel ein ganz neues, statistisches Gütekriterium einführen und im nächsten Kapitel zwei dazugehörige Algorithmen zur Regressorauswahl vorstellen. Dabei wird sich herausstellen, dass wir bei geschickter programmtechnischer Umsetzung fast keine Effizienzeinbußen gegenüber den bisherigen Algorithmen in Kauf nehmen müssen. Einziger Wermutstropfen an diesem neuen Konzept ist, dass das neue Gütekriterium keine (hinreichend effiziente) algorithmische Implementierung einer Regressoroptimierung zulässt. Das heißt, einmal ausgewählte Wavelets können nur noch in ihrer Gewichtung variiert, aber nicht mehr gestreckt oder gestaucht werden. Diese Einschränkung kann jedoch durch eine neue, umfangreichere Wavelet Bibliothek hinreichend kompensiert werden, die ebenfalls im nächsten Kapitel eingeführt werden soll. Zunächst aber konzentrieren wir unsere Betrachtungen auf das neuartige, statistische Gütekriterium.

6.1 Vereinbarungen

Das statistische Modell

Wir haben in Kapitel 3 die Grundlagen für die hier betrachteten Wavelet-Netzwerke gelegt. Insbesondere wurde unter 3.1 auf die Problemstellung und unter 3.2 auf die Struktur von Wavelet-Netzwerken eingegangen. Aus didaktischen Gründen werden hier einige dieser Konzepte noch einmal wiederholt, wobei neue Betrachtungen gleich einfließen.

Sei $(\Omega, \mathcal{F}, \mathcal{P})$ ein Wahrscheinlichkeitsraum. Gegeben ist ein Trainingsdatensatz der Form:

$$\begin{aligned} \mathbb{O}_1^N &= \{(x_1, y_1), \dots, (x_N, y_N)\} \\ x_i &\in \mathbb{R}^d, i = 1 \dots N \\ y_i &\in \mathbb{R}, i = 1 \dots N \end{aligned}$$

Dabei legen wir die Modellannahme zu Grunde, dass es eine uns nicht bekannte, wahre Funktion $\tilde{f} : \mathbb{R}^d \rightarrow \mathbb{R}$ gibt und ein $\omega \in \Omega$, sodass

$$y_i = \tilde{f}(x_i) + \Delta y_i, \quad i = 1 \dots N \tag{6.68}$$

Hierbei sollen die Δy_i als Realisierungen von unabhängigen, jeweils $N(0, \tilde{\sigma}^2)$ verteilten Zufallsvariablen $\Delta Y_i : \Omega \rightarrow \mathbb{R}$ angesehen werden, also $\Delta y_i = \Delta Y_i(\omega)$. Somit können auch die y_i als Realisierungen von Zufallsvariablen $Y_i : \Omega \rightarrow \mathbb{R}$ angesehen werden. Dieses Grundkonzept werden wir später noch weiter ausprägen.

Bei den folgenden Betrachtungen werden wir viele verschiedene, aber verwandte Bezeichner antreffen, wie z.B. ΔY_i und Δy_i . Um dabei nicht den Überblick zu verlieren, legen wir Bezeichnerkonventionen fest und führen Bezeichnerfamilien ein.

Bezeichnerkonventionen:

Bezeichner für	Schreibweise	Beispiel
Zufallsgrößen	Großbuchstaben	ΔY_i
Realisierungen	Kleinbuchstaben	Δy_i
Vektoren (außer $x_i \in \mathbb{R}^d$)	mit Unterstrich	$\underline{\Delta y}$
wahre Größen	Kleinbuchstaben mit Tilde	\tilde{y}
Abweichungen von wahren Größen	mit Delta	ΔY_i
Matrizen	Fett gedruckt	V

Die Bezeichnerfamilie $[y]$:

	Skalare $i = 1 \dots N$	Vektoren
Zufallsgrößen	$Y_i = \underbrace{\tilde{y}_i}_{\tilde{f}(x_i)} + \underbrace{\Delta Y_i}_{\sim N(0, \tilde{\sigma}^2)}$	$\underline{Y} = \underbrace{\tilde{y}} + \underbrace{\underline{\Delta Y}}$ $\begin{bmatrix} Y_1 \\ \vdots \\ Y_N \end{bmatrix} = \begin{bmatrix} \tilde{y}_1 \\ \vdots \\ \tilde{y}_N \end{bmatrix} + \begin{bmatrix} \Delta Y_1 \\ \vdots \\ \Delta Y_N \end{bmatrix}$
Realisierungen	$\underbrace{y_i}_{Y_i(\omega)} = \tilde{y}_i + \underbrace{\Delta y_i}_{\Delta Y_i(\omega)}$	$\underline{y} = \underline{\tilde{y}} + \underline{\Delta y}$ $\underline{y}_{(\omega)} = \underline{\tilde{y}} + \underline{\Delta Y}_{(\omega)}$

(6.69)

Die Varianz $\tilde{\sigma}^2 \in \mathbb{R}^+$ betrachten wir ebenfalls als einen wahren, uns nicht bekannten Parameter. Wir suchen nun einen möglichst guten Schätzer $f : \mathbb{R}^d \rightarrow \mathbb{R}$ für \tilde{f} , den wir allein aus dem Trainingsdatensatz \mathbb{O}_1^N ermitteln müssen. Deshalb kann man in unserem statistischen Modell die Funktion f selbst als Realisierung

einer Zufallsfunktion $F : \Omega \rightarrow L^2(\mathbb{R}^d, \mathbb{R})$ ansehen. Es bietet sich wieder an, eine Bezeichnerfamilie $[f]$ zu gründen, die viele Parallelen zur Bezeichnerfamilie $[y]$ aufweist.

Die Bezeichnerfamilie $[f]$:

	Skalare $(x \in \mathbb{R}^d)$	Funktionen $\in L^2(\mathbb{R}^d, \mathbb{R})$	
Zufallsgrößen	$F(x) = \tilde{f}(x) + \Delta F(x)$	$F = \tilde{f} + \Delta F$	(6.70)
Realisierungen	$\underbrace{f(x)}_{F(\omega)(x)} = \tilde{f}(x) + \underbrace{\Delta f(x)}_{\Delta F(\omega)(x)}$	$\underbrace{f}_{F(\omega)} = \tilde{f} + \underbrace{\Delta f}_{\Delta F(\omega)}$	

Zur Veranschaulichung empfiehlt es sich etwas vorzugreifen und einen Blick auf den unteren Teil von Abbildung 21 zu werfen. Wir werden später noch genauer auf die Grafik eingehen. An dieser Stelle sei nur noch darauf hingewiesen, dass zwar

$$\tilde{y}_i = \tilde{f}(x_i), \quad i = 1 \dots N \tag{6.71}$$

aber im allgemeinen

$$y_i \neq f(x_i), \quad i = 1 \dots N \quad \text{und}$$

$$\Delta y_i \neq \Delta f(x_i), \quad i = 1 \dots N$$

Die Wavelet-Bibliothek

Die Begriffe Wavelet-Familie und Wavelet-Frame wurden in Kapitel 2, Mathematische Grundlagen eingeführt. Sei ψ . eine (abzählbar unendliche) Wavelet-Familie und

$$W_{wav} \subset \psi.$$

$$W_{wav} = \{\psi_1, \dots, \psi_L\}$$

$$\psi_i : \mathbb{R}^d \rightarrow \mathbb{R}, \quad i = 1 \dots L$$

$$\|\psi_i\|_2 = 1, \quad i = 1 \dots L$$

Die Wavelet-Bibliothek stellt eine endliche, aber große Menge von Wavelets zur Verfügung, sodass $span(W_{wav}) \subset L^2(\mathbb{R}^d, \mathbb{R})$ uns genug Flexibilität gibt, um verschiedenartigste Schätzer-Funktionen f als Linearkombination der ψ_i darzustellen. Wir definieren weiterhin

$$\mathbf{V} = (\underline{v}_1, \dots, \underline{v}_L) \quad \text{mit} \quad \underline{v}_i = \begin{bmatrix} \psi_i(x_1) \\ \vdots \\ \psi_i(x_N) \end{bmatrix} \quad i = 1 \dots L \tag{6.72}$$

6.2 Statistischer Raum und Hypothesen

Da $|W_{wav}| = L$ im allgemeinen sehr groß, und insbesondere größer als unsere Anzahl von Trainingsdatenpunkten sein wird, müssen wir eine Teilmenge der Wavelet-Bibliothek W_{wav} auswählen. Der Schätzer f soll schließlich als Linearkombination dieser Teilmenge gebildet werden. Dazu nutzen wir das Konzept des Statistischen Raumes. Zu unserem Messraum (Ω, \mathcal{F}) sei \mathcal{W} eine Menge von Wahrscheinlichkeitsmaßen auf \mathcal{F} . Der Zufallsvektor

$$\underline{Y} : \Omega \rightarrow \mathbb{R}^N$$

sei $\mathcal{F} - \mathcal{B}^N$ -messbar, wobei \mathcal{B}^N für die N -dimensionale Borelsche σ -Algebra steht. Sei ferner

$$\Theta = \left\{ (\tilde{f}, \tilde{\sigma}^2) \mid \tilde{f} \in L^2(\mathbb{R}^d, \mathbb{R}), \tilde{\sigma}^2 \in \mathbb{R}^+ \right\} \quad (6.73)$$

eine Parametermenge. Seien nun die $\mathbb{P}_{\underline{Y}, \theta}$ mit $\theta \in \Theta$ Wahrscheinlichkeitsmaße auf $(\mathbb{R}^N, \mathcal{B}^N)$, mit denen die Komponenten Y_1, \dots, Y_N von \underline{Y} paarweise unabhängig und

$$Y_i \sim N(\tilde{f}(x_i), \tilde{\sigma}^2) \quad \text{für } i = 1 \dots N$$

sind. Anders formuliert, sind unter dem Wahrscheinlichkeitsmaß $\mathbb{P}_{\underline{Y}, \theta}$ die

$$Y_i = \tilde{f}(x_i) + \Delta Y_i \quad \text{mit } \Delta Y_i \sim N(0, \tilde{\sigma}^2) \quad (6.74)$$

und ΔY_i paarweise unabhängig.

Das Grundkonzept des statistischen Raumes ist nun die Annahme, dass es ein wahres $\tilde{\theta} \in \Theta$ gibt und damit eine wahre Verteilung $\mathbb{P}_{(\underline{Y}, \tilde{\theta})}$ die \underline{Y} zu Grunde liegt. Diese kennen wir aber nicht. Deshalb stellen wir Hypothesen auf, also Annahmen über $\tilde{\theta}$, die wir dann mit einem Gütekriterium gegeneinander vergleichen.

Definition 6.1 Hypothese H_{I_r}

Sei $(\mathbb{R}^N, \mathcal{B}^N, \mathbb{P}_{\underline{Y}, \theta})$ der Statistische Raum des Zufallsvektors $\underline{Y} = (Y_1, \dots, Y_N)^T$ mit den Y_i nach (6.74) und Θ aus (6.73). Sei ferner $W_{wav} = \{\psi_1, \dots, \psi_L\}$ eine Wavelet-Bibliothek und $I_r \subset \{1, \dots, L\}$ mit $|I_r| = r \leq L$. Dann stellen wir die Hypothese H_{I_r} auf, dass

$$\tilde{f} \in \text{span} \{ \psi_i \mid i \in I_r \}$$

Anders ausgedrückt: Für $\tilde{\theta} = (\tilde{f}, \tilde{\sigma}^2) \in \Theta$ gilt: Für alle $i \in I_r$ gibt es ein $\tilde{u}_i \in \mathbb{R}$, sodass

$$\tilde{f}(x) = \sum_{i \in I_r} \tilde{u}_i \psi_i(x)$$

und damit

$$Y_i = \tilde{f}(x_i) + \Delta Y_i \quad i = 1 \dots N$$

mit unabhängigen und identisch $N(0, \tilde{\sigma}^2)$ verteilten Zufallsvariablen ΔY_i .

Wir gehen also in jeder Hypothese davon aus, dass die wahre Funktion \tilde{f} in einem sehr eingeschränkten Funktionenraum liegt. Entscheidend dabei ist, dass jede Hypothese ihre eigenen Parameter \tilde{u}_i und $\tilde{\sigma}^2$ hat, die aber nicht bekannt sind.

Anders formuliert: Eine Hypothese H_{I_r} lässt nicht nur ein einziges $\tilde{\theta}$ zu sondern eine, wenn auch sehr eingeschränkte, Teilmenge von Θ . Unter einer Hypothese H_{I_r} werden wir den Schätzer f auch aus dem $\text{span}\{\psi_i | i \in I_r\}$ wählen. Dann müssen nur noch geeignete Schätzer u_i für die wahren Parameter \tilde{u}_i gefunden werden. Auch für $\tilde{\sigma}^2$ werden wir einen Schätzer finden. Es sei noch erwähnt, dass der Parameterraum Θ bei Weitem nicht vollständig durch Hypothesen abgedeckt wird, da die $\tilde{f} \in L^2(\mathbb{R}^d, \mathbb{R})$ dafür aus einem viel zu großen Funktionenraum kommen. Man könnte den Parameterraum Θ auch gleich weiter einschränken, indem man $\tilde{f} \in \text{span}(W_{wav})$ fordert. Aber selbst in diesem Fall würde Θ im Allgemeinen nicht vollständig durch Hypothesen abgedeckt werden können.

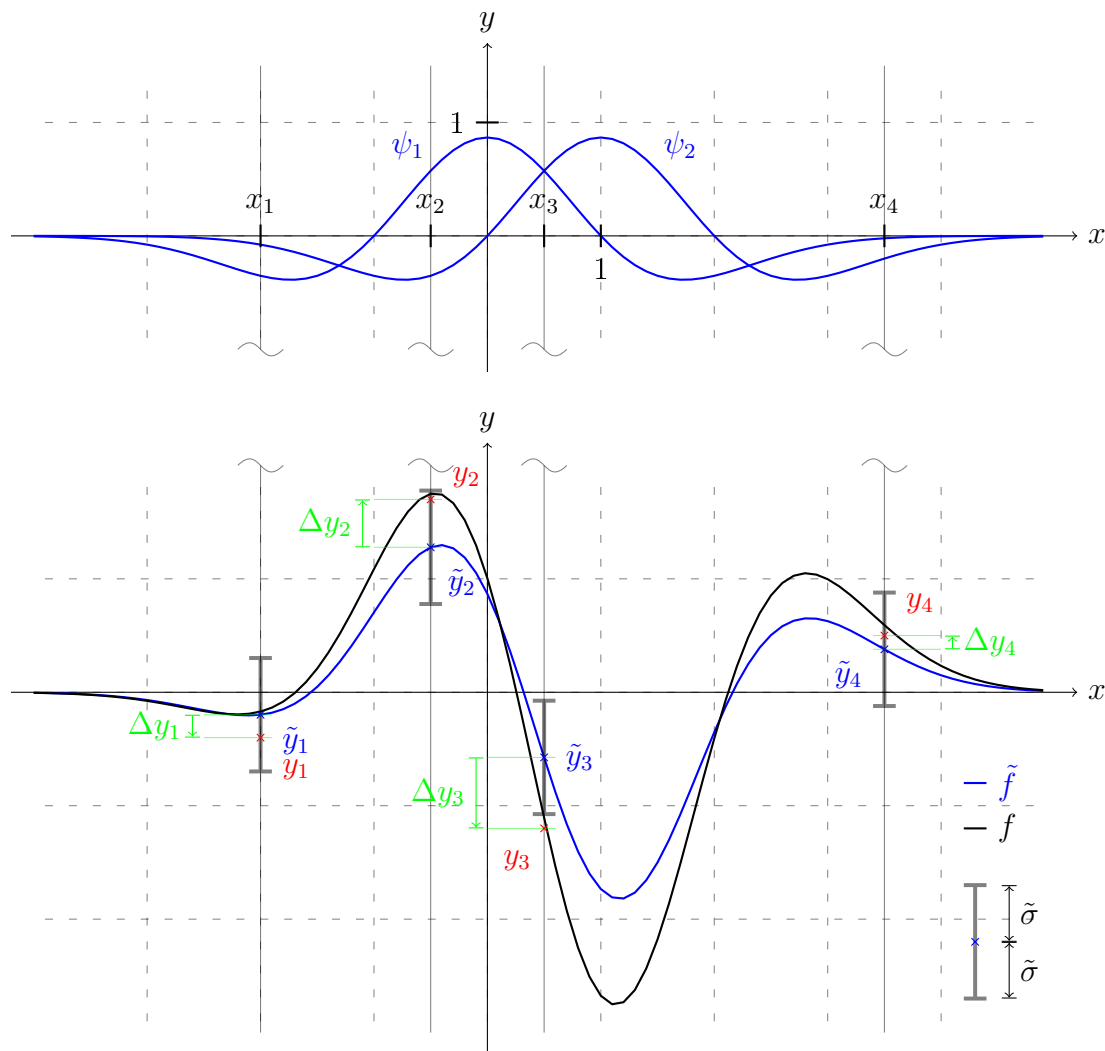


Abbildung 21: Beispiel einer einfachen Hypothese H_{I_2} mit zwei Wavelets

Beispiel 6.2

Um das Konzept der Hypothesen etwas anschaulicher zu machen, wurde in Abbildung 21 ein einfaches Beispiel graphisch dargestellt. Gegeben ist ein ein-dimensionales Problem ($d = 1$) mit vier Punkten, zu sehen als rote Kreuze. Der Trainingsdatensatz ist

$$\mathbb{O}_1^4 = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\}$$

Gesucht ist eine Funktion f , die den funktionalen Zusammenhang zwischen den vier x - und y - Werten auf sinnvolle Weise nachbilden soll. Es werden hierzu verschiedene Hypothesen aufgestellt, die in einem späteren Schritt verglichen werden sollen. Genau eine dieser Hypothesen ist in Abbildung 21 zu sehen. Diese Hypothese nennen wir hier

$$H_{I_2} \quad \text{mit} \quad I_2 = \{1, 2\}$$

Das heißt, wir nehmen an, dass die wahre Funktion \tilde{f} , eine Linearkombination der Wavelets ψ_1 und ψ_2 ist. Im konkreten Beispiel handelt es sich um Mexikanerhut - Wavelets. ψ_1 entspricht dem Mutter-Wavelet ψ und ψ_2 ist um den Wert $t = 1$ nach rechts verschoben. Die konkrete Wahl der Wavelets ist für das Hypothesenkonzept aber nicht von Bedeutung. Wichtig ist, dass \tilde{f} nun durch zwei (gedachte) Parameter \tilde{u}_1 und \tilde{u}_2 bestimmt ist mit

$$\tilde{f} = \tilde{u}_1 \psi_1 + \tilde{u}_2 \psi_2$$

Die Funktion \tilde{f} ist nur eine gedachte Funktion, die zu keinem Zeitpunkt genau bestimmt werden kann. Sie ist zu Vorstellungszwecken in die Grafik eingezeichnet worden. Ebenso gedacht ist die wahre Standardabweichung $\tilde{\sigma}$, die durch Fehlerbalken in der Grafik angedeutet wurde. Wir benötigen die beiden gedanklichen Konstrukte aber um die y -Werte aus unseren vier Trainingsdatenpunkten als Realisierungen von Zufallsvariablen aus einem statistischen Modell gemäß (6.68) anzusehen. Anders verhält es sich mit dem Schätzer f . Zu einem gegebenen Trainingsdatensatz \mathbb{O}_1^N und einer Hypothese H_{I_r} wird eine konkrete Funktion

$$f = u_1 \psi_1 + u_2 \psi_2$$

zugeordnet. Dazu werden die Größen u_1 und u_2 so gewählt, dass

$$\sum_1^4 [y_i - f(x_i)]^2$$

minimal wird. Sollte sich unsere Beispielhypothese H_{I_2} am Ende als beste herausstellen, so würde die in der Grafik dargestellte Funktion f als Lösung ausgegeben.

Da die Indizierung der $\psi_i \in W_{wav}$ beliebig vertauschbar ist, werden wir zunächst zur Vereinfachung festlegen, dass o.B.d.A

$$\begin{aligned} I_r &= \{1, \dots, r\} \\ \Leftrightarrow \tilde{f} &\in \text{span} \{ \psi_1, \dots, \psi_r \} \\ \Leftrightarrow \tilde{f}(x) &= \sum_{i=1}^r \tilde{u}_i \psi_i(x) \end{aligned}$$

Sei weiterhin mit den \underline{v}_i aus (6.72)

$$\mathbf{V}_r = \mathbf{V}_{I_r} = (\underline{v}_1, \dots, \underline{v}_r)$$

Dabei hat \mathbf{V}_r stets vollen Rang, weil Hypothesen H_{I_r} , bei denen die $\underline{v}_i \quad i = 1 \dots r$ linear abhängig sind, gar nicht erst betrachtet werden sollen. Mit der Definition der \underline{v}_i in (6.72) folgt insbesondere, dass die $\psi_i, \quad i = 1 \dots r$ linear unabhängig sein müssen. Denn wären die ψ_i linear abhängig, so wären es auch die \underline{v}_i . Unter der Hypothese H_{I_r} hat der Schätzer $f : \mathbb{R}^d \rightarrow \mathbb{R}$ für \tilde{f} die nun folgende Form:

$$f(x) = \sum_{i=1}^r u_i \psi_i(x)$$

Die u_i werden dabei allein aus dem Trainingsdatensatz \mathbb{O}_1^N gewonnen, weshalb wir sie wiederum als Realisierungen von Zufallsvariablen $U_i : \Omega \rightarrow \mathbb{R}$ auffassen können. Wir gründen wieder eine Bezeichnerfamilie $[u]$ in Analogie zur Bezeichnerfamilie $[y]$ aus (6.69)

Die Bezeichnerfamilie $[u]$:

	Skalare $i = 1 \dots r$	Vektoren	
Zufallsgrößen	$U_i = \tilde{u}_i + \Delta U_i$	$\underbrace{\underline{U}} = \underbrace{\tilde{\underline{u}}}_{\begin{bmatrix} \tilde{u}_1 \\ \vdots \\ \tilde{u}_r \end{bmatrix}} + \underbrace{\Delta \underline{U}}_{\begin{bmatrix} \Delta U_1 \\ \vdots \\ \Delta U_r \end{bmatrix}}$	(6.75)
Realisierungen	$\underbrace{u_i}_{U_i(\omega)} = \tilde{u}_i + \underbrace{\Delta u_i}_{\Delta U_i(\omega)}$	$\underbrace{\underline{u}}_{\underline{U}(\omega)} = \underbrace{\tilde{\underline{u}}}_{\tilde{\underline{u}}} + \underbrace{\Delta \underline{u}}_{\Delta \underline{U}(\omega)}$	

Wir fassen hier kurz zusammen:

$$\begin{aligned} f(x) &= \sum_{i=1}^r u_i \psi_i(x) \\ \tilde{f}(x) &= \sum_{i=1}^r \tilde{u}_i \psi_i(x) \\ \Delta f(x) &= \sum_{i=1}^r \Delta u_i \psi_i(x) \end{aligned} \tag{6.76}$$

Diese Gleichungen sind hier für die Realisierungen aus [f] und [u] formuliert, gelten aber in identischer Form auch für die entsprechenden Zufallsgrößen.

Wir definieren weiterhin das Residuum γ_i an einem Trainingsdatenpunkt und den Residuenvektor $\underline{\gamma}$ und berechnen mit (6.72)

$$\begin{aligned} \gamma_i &= y_i - f(x_i) \\ \underline{\gamma} &= \begin{bmatrix} \gamma_1 \\ \vdots \\ \gamma_N \end{bmatrix} = \underline{y} - \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_N) \end{bmatrix} \\ &= \underline{y} - \sum_{i=1}^r \begin{bmatrix} u_i \psi_i(x_1) \\ \vdots \\ u_i \psi_i(x_N) \end{bmatrix} = \underline{y} - \sum_{i=1}^r u_i \underline{v}_i = \underline{y} - \mathbf{V}_r \underline{u} \end{aligned} \quad (6.77)$$

Wir wollen \underline{u} nun so wählen, dass $\|\underline{\gamma}\|_2$ minimal wird, also

$$\begin{aligned} \underline{u} &= \operatorname{argmin} \|\underline{\gamma}\|_2 \\ &= \operatorname{argmin} [\underline{\gamma}^T \underline{\gamma}] \\ &= \operatorname{argmin} [\underline{y}^T \underline{y} - 2\underline{y}^T \mathbf{V}_r \underline{u} + \underline{u}^T \mathbf{V}_r^T \mathbf{V}_r \underline{u}] \\ \Leftrightarrow \underline{0} &= \nabla_{\underline{u}} [\underline{y}^T \underline{y} - 2\underline{y}^T \mathbf{V}_r \underline{u} + \underline{u}^T \mathbf{V}_r^T \mathbf{V}_r \underline{u}] \\ \Leftrightarrow \underline{0} &= -2\mathbf{V}_r^T \underline{y} + 2\mathbf{V}_r^T \mathbf{V}_r \underline{u} \\ \Leftrightarrow \underline{u} &= (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \underline{y} \end{aligned} \quad (6.78)$$

Probleme dieser Art bezeichnet man als lineare Ausgleichsprobleme. \underline{u} ist ein Maximum-Likelihood-Schätzer für \tilde{u} unter der Hypothese H_{I_r} , was wir hier aber nicht beweisen wollen. Mit (6.77) folgern wir noch, dass

$$\underline{\gamma} = \underline{y} - \mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \underline{y}$$

und damit

$$\|\underline{\gamma}\|_2^2 = \underline{y}^T \underline{y} - \underline{y}^T \mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \underline{y} \quad (6.79)$$

Wir fahren fort mit (6.71) und (6.76) und schließen, dass

$$\tilde{\underline{y}} = \begin{bmatrix} \tilde{f}(x_1) \\ \vdots \\ \tilde{f}(x_N) \end{bmatrix} = \sum_{i=1}^r \begin{bmatrix} \tilde{u}_i \psi_i(x_1) \\ \vdots \\ \tilde{u}_i \psi_i(x_N) \end{bmatrix} = \sum_{i=1}^r \tilde{u}_i \underline{v}_i = \mathbf{V}_r \tilde{\underline{u}}$$

Daraus folgt unmittelbar, dass

$$\tilde{\underline{u}} = (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \tilde{\underline{y}}$$

und mit (6.78) folgt letztlich, dass

$$\Delta \underline{u} = (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \Delta \underline{y} \quad (6.80)$$

Die Gleichung gilt in identischer Weise für die zugehörigen Zufallsgrößen der Bezeichnerfamilien [y] und [u]

Ein Schätzer für $\tilde{\sigma}^2$

Wir haben bis jetzt zu einem gegebenen Trainingsdatensatz \mathbb{O}_1^N unter der Hypothese H_{I_r} den Schätzer \underline{u} bestimmt. Damit sind ebenso der Schätzer f und der Residuenvektor $\underline{\gamma}$ bestimmt. In unserem statistischen Modell kommen Messfehler ΔY_i mit dem Erwartungswert Null und der Varianz $\tilde{\sigma}^2$ vor. Diese Varianz $\tilde{\sigma}^2$ können wir unter der Hypothese H_{I_r} durch eine sogenannte Stichprobenvarianz oder auch empirische Varianz σ^2 schätzen. Mit (6.69) ergibt sich

$$\sigma^2 = \text{Var}_{emp}(Y_i) = \text{Var}_{emp}(\Delta Y_i) = \frac{\|\underline{\gamma}\|_2^2}{N-r} \quad (6.81)$$

mit $\|\underline{\gamma}\|_2^2$ aus (6.79). Die Formel für die Stichprobenvarianz wurde in Abschnitt 2.6.2 hergeleitet. Sie basiert auf dem linearen statistischen Modell. Dabei ist σ^2 ein erwartungstreuer Schätzer für $\tilde{\sigma}^2$ innerhalb einer Hypothese H_{I_r} . Es sei vorweg genommen, dass diese Stichprobenvarianz im Rahmen der Hypothesenauswahl Probleme bereiten kann. Auf letztere Probleme und deren Lösung kommen wir in Abschnitt 7.10 zurück.

6.3 Die Modellunsicherheit einer Hypothese H_{I_r}

In diesem Abschnitt wollen wir ein Kriterium für die Robustheit eines Modells nach der Hypothese H_{I_r} beschreiben. Grob gesagt, bezeichnen wir hier ein Modell als robust, wenn der Schätzer f im Mittel nur wenig von der wahren Funktion \tilde{f} abweicht, also $\mathbb{E}(\|\Delta F\|_2^2)$ klein ist. Mit Hilfe von (6.76) und der $L^2(\mathbb{R}^d, \mathbb{R})$ -Norm berechnen wir

$$\begin{aligned} \|\Delta f\|_2^2 &= \int_{\mathbb{R}^d} [\Delta f(x)]^2 d\lambda(x) \\ &= \int_{\mathbb{R}^d} \left[\sum_{i=1}^r \Delta u_i \psi_i(x) \right]^2 d\lambda(x) \\ &= \int_{\mathbb{R}^d} \left[\sum_{i=1}^r \Delta u_i \psi_i(x) \right] \cdot \left[\sum_{j=1}^r \Delta u_j \psi_j(x) \right] d\lambda(x) \\ &= \sum_{i=1}^r \sum_{j=1}^r \Delta u_i \Delta u_j \underbrace{\int_{\mathbb{R}^d} \psi_i(x) \psi_j(x) d\lambda(x)}_{p_{ij}} \end{aligned}$$

Dabei steht λ für das d -dimensionale Lebesgue-Maß. Um diese Gleichung noch etwas schöner schreiben zu können, definieren wir uns

$$\mathbf{P}_r = \begin{pmatrix} p_{11} & \cdots & p_{1r} \\ \vdots & \ddots & \vdots \\ p_{r1} & \cdots & p_{rr} \end{pmatrix} \quad \text{mit} \quad p_{ij} = \langle \psi_i, \psi_j \rangle = \int_{\mathbb{R}^d} \psi_i(x) \psi_j(x) dx$$

wobei $\langle \cdot, \cdot \rangle$ das Skalarprodukt in $L^2(\mathbb{R}^d, \mathbb{R})$ ist. Mit (6.75) erhalten wir schließlich

$$\|\Delta f\|_2^2 = \Delta \underline{u}^T \mathbf{P}_r \Delta \underline{u}$$

An dieser Gleichung sieht man sofort, dass \mathbf{P}_r positiv semidefinit ist. Weil die ψ_i $i = 1 \dots r$ linear unabhängig sind, folgt mit (6.76) sogar, dass \mathbf{P}_r positiv definit sein muss. Betrachten wir die zugehörigen Zufallsgrößen der Bezeichnerfamilien $[f]$ und $[u]$, so erhalten wir analog

$$\|\Delta F\|_2^2 = \Delta U^T \mathbf{P}_r \Delta U$$

Diese Art der doppelten Betrachtung sollten wir bei allen weiteren Überlegungen im Hinterkopf behalten. Das nächste Etappenziel wird es sein den Erwartungswert $\mathbb{E}(\|\Delta F\|_2^2)$ zu berechnen, aber bis dahin ist noch ein gewisser Weg. Stammen die ψ_i aus einer orthonormalen Wavelet-Basis ψ_\cdot , dann ist $\mathbf{P}_r = \mathbf{E}$, also die Einheitsmatrix. Orthonormale Wavelet-Basen können z.B. mit dem Haar-Wavelet, den Battle-Lemarié-Wavelets, den Daubechies-Wavelets oder dem Meyer-Wavelet konstruiert werden. Das für unsere Bedürfnisse besonders geeignete Mexikanerhut-Wavelet kann hingegen keine orthonormale Wavelet-Basis bilden. Deshalb ist \mathbf{P}_r für uns leider keine Einheitsmatrix. Wir wenden nun (6.80) an und erhalten

$$\|\Delta F\|_2^2 = \Delta \underline{Y}^T \underbrace{\mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T}_{\mathbf{B}_N} \Delta \underline{Y} \quad (6.82)$$

Die $N \times N$ -Matrix \mathbf{B}_N ist dabei symmetrisch, positiv semidefinit, weil \mathbf{P}_r symmetrisch, positiv definit ist. Damit ist \mathbf{B}_N diagonalisierbar. Es gibt also zu \mathbf{B}_N einen vollständigen Satz von paarweise orthogonalen, normierten Eigenvektoren $\underline{o}_1, \dots, \underline{o}_N$, die man zur orthonormalen Matrix

$$\mathbf{O} = (\underline{o}_1, \dots, \underline{o}_N)$$

zusammenfasst und eine Diagonalmatrix

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_N \end{bmatrix}$$

mit den zugehörigen Eigenwerten $\lambda_1, \dots, \lambda_N$ in entsprechender Reihenfolge, so dass

$$\mathbf{B}_N = \mathbf{O} \mathbf{D} \mathbf{O}^T$$

Nun betrachten wir den von $\Delta \underline{Y}$ abhängigen Zufallsvektor

$$\underline{\Phi} = \begin{bmatrix} \Phi_1 \\ \vdots \\ \Phi_N \end{bmatrix} = \mathbf{O}^T \Delta \underline{Y}$$

Da die Komponenten von $\Delta \underline{Y}$, also $\Delta Y_1, \dots, \Delta Y_N$ paarweise unabhängig und je $N(0, \tilde{\sigma}^2)$ verteilt sind und \mathbf{O}^T eine orthonormale Matrix ist, sind auch die Zufallsvariablen Φ_1, \dots, Φ_N paarweise unkorreliert und $N(0, \tilde{\sigma}^2)$ verteilt. Für einen Beweis sei auf [10] verwiesen. Nun haben wir alle Bausteine zusammen um $\mathbb{E}(\|\Delta F\|_2^2)$ zu berechnen. Dabei beginnen wir mit (6.82):

$$\begin{aligned} \mathbb{E}(\|\Delta F\|_2^2) &= \mathbb{E}(\Delta \underline{Y}^T \mathbf{B}_N \Delta \underline{Y}) \\ &= \mathbb{E}(\Delta \underline{Y}^T \mathbf{O} \mathbf{O}^T \Delta \underline{Y}) \\ &= \mathbb{E}(\underline{\Phi}^T \mathbf{D} \underline{\Phi}) \\ &= \sum_{i=1}^N \lambda_i \mathbb{E}(\Phi_i^2) \\ &= \tilde{\sigma}^2 \text{Spur}(\mathbf{D}) \end{aligned}$$

Da man Matrizen in der Spur vertauschen kann, ergeben sich folgende Vereinfachungen

$$\begin{aligned} \text{Spur}(\mathbf{D}) &= \text{Spur}(\mathbf{O} \mathbf{O}^T) \\ &= \text{Spur}(\mathbf{B}_N) \\ &= \text{Spur}[(\mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T)] \\ &= \text{Spur}[\mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1}] \end{aligned}$$

Somit erhalten wir schließlich

$$\mathbb{E}(\|\Delta F\|_2^2) = \tilde{\sigma}^2 \text{Spur}[\mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1}] \quad (6.83)$$

Damit ist das Etappenziel erreicht und ein Kriterium für die Modellunsicherheit der Hypothese H_{I_r} gefunden. Dabei kann der Ausdruck $\text{Spur}[\mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1}]$ direkt, und später auch effizient, berechnet werden. $\tilde{\sigma}^2$ ist unbekannt und wird durch die empirische Varianz σ^2 geschätzt werden müssen.

6.4 Bewertung von Hypothesen

Unser Ziel ist es nun jede Hypothese H_{I_r} durch ein einziges Gütekriterium zu bewerten um schließlich die beste Hypothese auswählen zu können. In dieses Gütekriterium müssen zwei Aspekte einfließen

- Möglichst gute Anpassung des Modells an die Trainingsdaten
- Möglichst gute Modellsicherheit

Wir haben es also mit einer Ausprägung des Bias-Varianz-Dilemmas zu tun. Die Anpassung an die Trainingsdaten, genauer gesagt, die Anpassung der wahren Funktion f an die Trainingsdaten spiegelt sich in $\tilde{\sigma}^2$ wieder. Die Modellsicherheit, also die Anpassung des Schätzers f an die wahre Funktion \tilde{f} messen wir, wie eben gezeigt, mit $E(\|\Delta F\|_2^2)$. Da diese zwei Größen aus „verschiedenen Welten“ stammen, müssen wir in die Trickkiste greifen um beide sinnvoll miteinander zu verbinden. Hier kommt das Konzept des Pseudo-Trägers zum Einsatz.

Das Konzept des Pseudo-Trägers

Hinter dem Konzept des Pseudo-Trägers stehen zwei Überlegungen. Zum einen wollen wir eine Funktion f finden, die nichts anderes als eine Linearkombination von gestreckten, bzw. gestauchten und verschobenen Versionen eines Mutter-Wavelets ist. Dieses Mutter-Wavelet ist bei uns das d -dimensionale Mexikanerhut-Wavelet aus (3.26), also

$$\psi : \mathbb{R}^d \rightarrow \mathbb{R} \quad \text{mit} \quad \psi(x) = C_d \cdot (d - \|x\|_2) \exp\left(-\frac{\|x\|_2^2}{2}\right)$$

Dieses Mutter-Wavelet fällt für $\|x\| \rightarrow \infty$ sehr schnell gegen Null ab. Es hat also einen quasi kompakten Träger. Folglich wird auch f stets einen quasi kompakten Träger besitzen.

Die zweite Überlegung ist, dass nur solche Wavelets ψ_i aus W_{wav} ausgewählt werden, die genügend Trainingspunkte x_i im Bereich ihres quasi kompakten Trägers aufweisen. Daraus folgt, dass unser Schätzer f außerhalb des Bereiches der Trainingsdaten sehr schnell gegen Null abfallen wird. Deshalb betrachten wir die grobe Umgebung der Trainingsdaten als Pseudo-Träger unseres Wavelet-Netzwerkes f . Natürlich ist das kein mathematisch genaues Konzept. Es ist aber leider alternativlos. Sei $T \subset \mathbb{R}^d$ dieser Pseudo-Träger. T sei borelsch und sinnvoller Weise zumeist kompakt. Insbesondere sollten alle x_i aus \mathbb{O}_1^N in T liegen. Die gute Nachricht ist, dass wir für die folgenden Betrachtungen lediglich die Größe $\tau = |T| = \lambda(T)$ benötigen, wobei λ das d -dimensionale Lebesgue-Maß darstellt. Über die konkrete Gestalt des Pseudo-Trägers müssen wir uns also keine Gedanken machen. Wir müssen lediglich seine Größe τ grob abschätzen und sie zusammen mit dem Trainingsdatensatz \mathbb{O}_1^N vorgeben.

Zusammenführung beider Aspekte

Wir wollen im Folgenden feststellen, wie stark eine Abweichungsfunktion Δf des Schätzers f von der wahren Funktion \tilde{f} an einem einzelnen Punkt im Mittel ausschlägt. Wie im letzten Abschnitt beschrieben, sind signifikante Ausschläge nur im Bereich des Pseudo-Trägers zu erwarten. Sei $\dot{X} : \Omega \rightarrow \mathbb{R}^d$ eine Zufallsvariable, die auf dem Pseudo-Träger $T \subset \mathbb{R}^d$ gleichverteilt und von den bisher betrachteten Zufallsvariablen unabhängig ist. Man kann sich \dot{X} als einen statistischen Auswertedatenpunkt vorstellen.

Uns interessiert dabei der Erwartungswert

$$\begin{aligned}\mathbb{E}_{\dot{X}} \left[\left(\Delta f(\dot{X}) \right)^2 \right] &= \frac{1}{\lambda(T)} \int_{T \subset \mathbb{R}^d} [\Delta f(x)]^2 d\lambda(x) \\ &= \frac{1}{\tau} \int_{T \subset \mathbb{R}^d} [\Delta f(x)]^2 d\lambda(x)\end{aligned}$$

Dabei ist λ das d -dimensionale Lebesgue-Maß. Nun müssen wir eine Näherung machen, die aber verantwortbar ist, weil wir erwarten, dass Δf außerhalb des Pseudo-Trägers sehr schnell gegen Null geht.

$$\begin{aligned}\mathbb{E}_{\dot{X}} \left[\left(\Delta f(\dot{X}) \right)^2 \right] &\approx \frac{1}{\tau} \int_{\mathbb{R}^d} [\Delta f(x)]^2 d\lambda(x) \\ &= \frac{1}{\tau} \|\Delta f\|_2^2\end{aligned}$$

Diese Betrachtung gilt wie gewohnt in analoger Weise für die Zufallsgröße ΔF , also

$$\mathbb{E}_{\dot{X}} \left[\left(\Delta F(\dot{X}) \right)^2 \right] \approx \frac{1}{\tau} \|\Delta F\|_2^2$$

Bilden wir jetzt noch den Erwartungswert über ΔF , so erhalten wir mit (6.83)

$$\begin{aligned}\mathbb{E} \left[\left(\Delta F(\dot{X}) \right)^2 \right] &= \mathbb{E}_{\Delta F} \left(\mathbb{E}_{\dot{X}} \left[\left(\Delta F(\dot{X}) \right)^2 \right] \right) \\ &\approx \frac{1}{\tau} \mathbb{E} \left[\|\Delta F\|_2^2 \right] \\ &= \frac{1}{\tau} \tilde{\sigma}^2 \text{Spur} \left[\mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \right]\end{aligned}$$

Wir haben nun ein Maß für die mittlere Abweichung des Schätzers f von der wahren Funktion \tilde{f} an einem zufälligen Punkt. Nun gehen wir noch einen Schritt weiter und definieren eine Zufallsvariable $\Delta \dot{Y} : \Omega \rightarrow \mathbb{R}$, die $N(0, \tilde{\sigma}^2)$ -verteilt und von den bisherigen Zufallsvariablen unabhängig ist. Sei damit die abhängige Zufallsvariable

$$\dot{Y} = \tilde{f}(\dot{X}) + \Delta \dot{Y}$$

Das Paar \dot{X}, \dot{Y} kann man als einen statistischen Auswertedatensatz ansehen, mit dem wir nun endlich die Güte der Hypothese H_{I_r} testen wollen. Die Finale Frage ist: Wie stark werden weitere Realisierungen von Datensätzen unter der Hypothese H_{I_r} im Mittel von dem gefundenen Schätzer abweichen.

Diese Frage definiert uns das Ideal-Kriterium \tilde{K} mit

$$\begin{aligned}
 \tilde{K} &= \mathbb{E} \left[\left(\dot{Y} - F(\dot{X}) \right)^2 \right] \\
 &= \mathbb{E} \left[\left(\tilde{f}(\dot{X}) + \Delta \dot{Y} - \tilde{f}(\dot{X}) - \Delta F(\dot{X}) \right)^2 \right] \\
 &= \mathbb{E} \left[\left(\Delta \dot{Y} - \Delta F(\dot{X}) \right)^2 \right] \\
 &= \mathbb{E} \left[\left(\Delta \dot{Y} \right)^2 \right] - 2 \mathbb{E} \left[\Delta \dot{Y} \Delta F(\dot{X}) \right] + \mathbb{E} \left[\left(\Delta F(\dot{X}) \right)^2 \right] \\
 &= \underbrace{\text{Var}(\Delta \dot{Y})}_{=\tilde{\sigma}^2} - 2 \underbrace{\mathbb{E} \left[\Delta \dot{Y} \right]}_{=0} \mathbb{E} \left[\Delta F(\dot{X}) \right] + \underbrace{\mathbb{E} \left[\left(\Delta F(\dot{X}) \right)^2 \right]}_{\approx \frac{1}{\tau} \tilde{\sigma}^2 \text{Spur}(\mathbf{P}_r(\mathbf{V}_r^T \mathbf{V}_r)^{-1})}
 \end{aligned}$$

Wenn wir nun zusammenfassen und $\tilde{\sigma}^2$ durch den erwartungstreuen Schätzer σ^2 schätzen, so erhalten wir

$$\begin{aligned}
 \tilde{K} &\approx \tilde{\sigma}^2 \left(1 + \frac{1}{\tau} \text{Spur} \left[\mathbf{P}_r(\mathbf{V}_r^T \mathbf{V}_r)^{-1} \right] \right) \\
 &\approx \sigma^2 \left(1 + \frac{1}{\tau} \text{Spur} \left[\mathbf{P}_r(\mathbf{V}_r^T \mathbf{V}_r)^{-1} \right] \right)
 \end{aligned}$$

Mit diesem Ergebnis können wir uns zufrieden geben und definieren das real berechenbare Kriterium $K = K(H_{I_r})$ als

$$K = \sigma^2 \left(1 + \frac{1}{\tau} \text{Spur} \left[\mathbf{P}_r(\mathbf{V}_r^T \mathbf{V}_r)^{-1} \right] \right) \quad (6.84)$$

Mit dieser Formel für das Kriterium K sind wir am Ziel dieses Kapitels angekommen. Der Zweite Summand in der Klammer lässt sich als Modellunsicherheitsfaktor auffassen, den wir mit MUF bezeichnen wollen. Es gilt also

$$K = \sigma^2 (1 + MUF) \quad (6.85)$$

mit

$$MUF = \frac{1}{\tau} \text{Spur} \left[\mathbf{P}_r(\mathbf{V}_r^T \mathbf{V}_r)^{-1} \right]$$

Die empirische Varianz σ^2 gibt in etwa wieder, wie gut sich die Funktion f an die Trainingsdaten anpasst. Der Faktor MUF ist ein Maß für die Robustheit des Modells an sich. Er macht eine Aussage darüber, wie sich die Funktion f zwischen den Trainingsdaten verhält, wenn die Trainingsdaten selbst wackeln. Wir sehen weiterhin, dass die Varianz-Schätzung von kritischer Bedeutung ist, weil sie auch mit dem Modellunsicherheitsfaktor multipliziert wird. Am Ende des nächsten Kapitels werden wir darauf zurück kommen. Nun erst einmal zum nächsten Schritt! Die Aufgabe der Regressorauswahl muss es sein, eine Hypothese H_{I_r} zu finden, die K minimiert. Im nächsten Kapitel werden dazu geeignete, effiziente Algorithmen vorgestellt.

7 Algorithmen zum neuen Gütekriterium

In diesem Kapitel wird es darum gehen, die Idee des neuen, statistisch motivierten Gütekriteriums K aus Kapitel 6 in Algorithmen zur Konstruktion von Wavelet-Netzwerken umzusetzen. Dabei werden wir unsere Anstrengungen auf das Mexikanerhut-Wavelet konzentrieren, wie es in (3.26) eingeführt wurde. Alle Algorithmen werden zunächst für reine Wavelet-Netzwerke beschrieben und später auf das Konzept eines erweiterten Wavelet-Netzwerkes angepasst. Die Ergebnisse der hier eingeführten Algorithmen sollen präsentiert werden um Stärken aber auch Schwachpunkte dieses neuen Ansatzes aufzuzeigen. Im letzten Abschnitt dieses Kapitels werden wir noch eine kleine, finale Modifikation des Gütekriteriums vornehmen, welches in Kapitel 6 eingeführt wurde. Damit werden wir eine Schwäche des Konzeptes kompensieren.

7.1 Eine rekursiv erzeugte Wavelet-Bibliothek

In diesem Abschnitt wollen wir einen erneuten Ansatz zur Konstruktion einer umfangreichen Wavelet-Bibliothek wagen. In Abschnitt 4.1 sind wir im Wesentlichen vom Wavelet-Frame

$$\psi. := (\psi_{m,n} | m \in \mathbb{Z}, n \in \mathbb{Z}^d)$$

mit

$$\psi_{m,n}(x) := \sigma^{-dm/2} \psi(\sigma^{-m}x - n\beta)$$

ausgegangen. Danach haben wir die Parameter m und n auf sinnvolle Weise beschränkt, sodass eine endliche aber hinreichend große Wavelet - Bibliothek entstand. Auffällig daran ist, dass im mehrdimensionalen Fall stets nur rotationssymmetrische Wavelets entstanden. Das genügt nach [11], Kapitel 10 auch, um die Frame - Eigenschaft von $\psi.$ zu gewährleisten. Auch bei der Einführung einer Cluster-Bibliothek in Abschnitt 5.1 haben wir uns auf rotationssymmetrische Wavelets beschränkt. An dieser Stelle wollen wir einen Algorithmus einführen, der eine breitere Palette von Wavelets zur Verfügung stellt. Ausgangspunkt unserer Überlegungen ist ein d -dimensionales Mutter-Wavelet. Wir ziehen hier zur Veranschaulichung wieder das Mexikanerhut-Wavelet aus (3.26) heran, mit

$$\psi : \mathbb{R}^d \rightarrow \mathbb{R} \quad \text{mit} \quad \psi(x) = C_d(d - \|x\|_2^2) e^{-\frac{1}{2}\|x\|_2^2}$$

Dabei ist C_d eine Konstante, die nur von der Dimension d abhängt. Wir werden C_d später, unter (7.105) noch konkret berechnen. Mit diesem Mutter-Wavelet wollen wir eine Wavelet-Bibliothek

$$W_{wav} = \{\psi_1, \dots, \psi_L\}$$

aufbauen. Jedes darin enthaltene Wavelet hat die Form

$$\psi_i : \mathbb{R}^d \rightarrow \mathbb{R} \quad \text{mit} \quad \psi_i(x) = \Pi(a_i)^{\frac{1}{2}} \psi[a_i \star (x - t_i)] \quad (7.86)$$

Dabei sind $a_i \in \mathbb{R}_+^d$, die $t_i \in \mathbb{R}^d$, $\Pi(\cdot)$ steht für das Produkt aller Komponenten eines Vektors und \star für das komponentenweise Produkt zweier Vektoren. Die Gewichte w_i haben wir implizit schon auf $w_i = \Pi(a_i)^{\frac{1}{2}}$ gesetzt, um $\|\psi_i\|_2 = 1$ zu gewährleisten. Ziel dieses neuerlichen Ansatzes ist es, bei den a_i mehr Variationen in den verschiedenen Dimensionen ins Spiel zu bringen als bisher.

7.1.1 Der eindimensionale Fall

Der Fall $d = 1$ ist kein Argument für diesen neuen Algorithmus. Es wird darin keine Verbesserung gegenüber den schon behandelten Wavelet-Bibliotheken erzielt. Wir betrachten ihn aus rein didaktischen Gründen. Hat man das Prinzip im eindimensionalen nachvollzogen, lässt sich die Konstruktionsweise sehr leicht in mehrere Dimensionen übertragen. Ansatz der Überlegungen ist wieder ein Wavelet-Frame. Betrachten wir dazu Abbildung 22. Darin sehen wir vier Abbilder von ein und dem selben Koordinatensystem. Für alle eingezeichneten Wavelets ψ_i gilt

$$\|\psi_i\|_2 = 1$$

Jedes Wavelet ψ_i ist somit durch das Parameterpaar $(a_i, t_i) \in \mathbb{R}^2$ eindeutig bestimmt zu

$$\psi_i = a_i^{\frac{1}{2}} \psi [a_i(x - t_i)] \quad (7.87)$$

Die Idee des Wavelet-Frames war es nun, die Parameterpaare (a_i, t_i) mit einem $\sigma > 1$ und $\beta > 0$ derart zu beschränken, dass nur noch

$$a_i = \sigma^{-m} \quad \text{und} \quad t_i = n\sigma^m\beta, \quad m, n \in \mathbb{Z}$$

und somit nur noch Wavelets der Form

$$\psi_i = \psi_{m,n} = \sigma^{-\frac{m}{2}} \psi (\sigma^{-m}x - n\beta)$$

zulässig waren. An diesem Konzept orientieren wir uns auch hier, wandeln es aber leicht ab. Dabei wollen wir uns auf den Fall $\sigma = 2$ beschränken, β aber variabel lassen. Wir legen uns hier fest auf

$$a_i = c^{-1} \cdot 2^{-m} \quad (7.88)$$

und ebenso darauf, dass zwei benachbarte, Wavelets ψ_k und ψ_l mit gleicher Dilation $a_k = a_l$ den Abstand

$$|t_k - t_l| = c \cdot 2^m \beta \quad (7.89)$$

haben sollen, wobei c eine noch zu bestimmende Konstante ist.

Wir betrachten nun die Werte x_1, \dots, x_N aus dem gegebenen Trainingsdatensatz \mathbb{O}_1^N . Dazu finden wir stets zwei Werte $x_{min}, x_{max} \in \mathbb{R}$, sodass

$$x_{min} \leq x_i \leq x_{max} \quad \text{für alle} \quad i = 1 \dots N.$$

Nun legen wir fest, dass sich an diesen beiden Stellen zwei benachbarte Wavelets im Sinne von Gleichung (7.89) befinden sollen, also ψ_1 mit $t_1 = x_{min}$ und ψ_2 mit

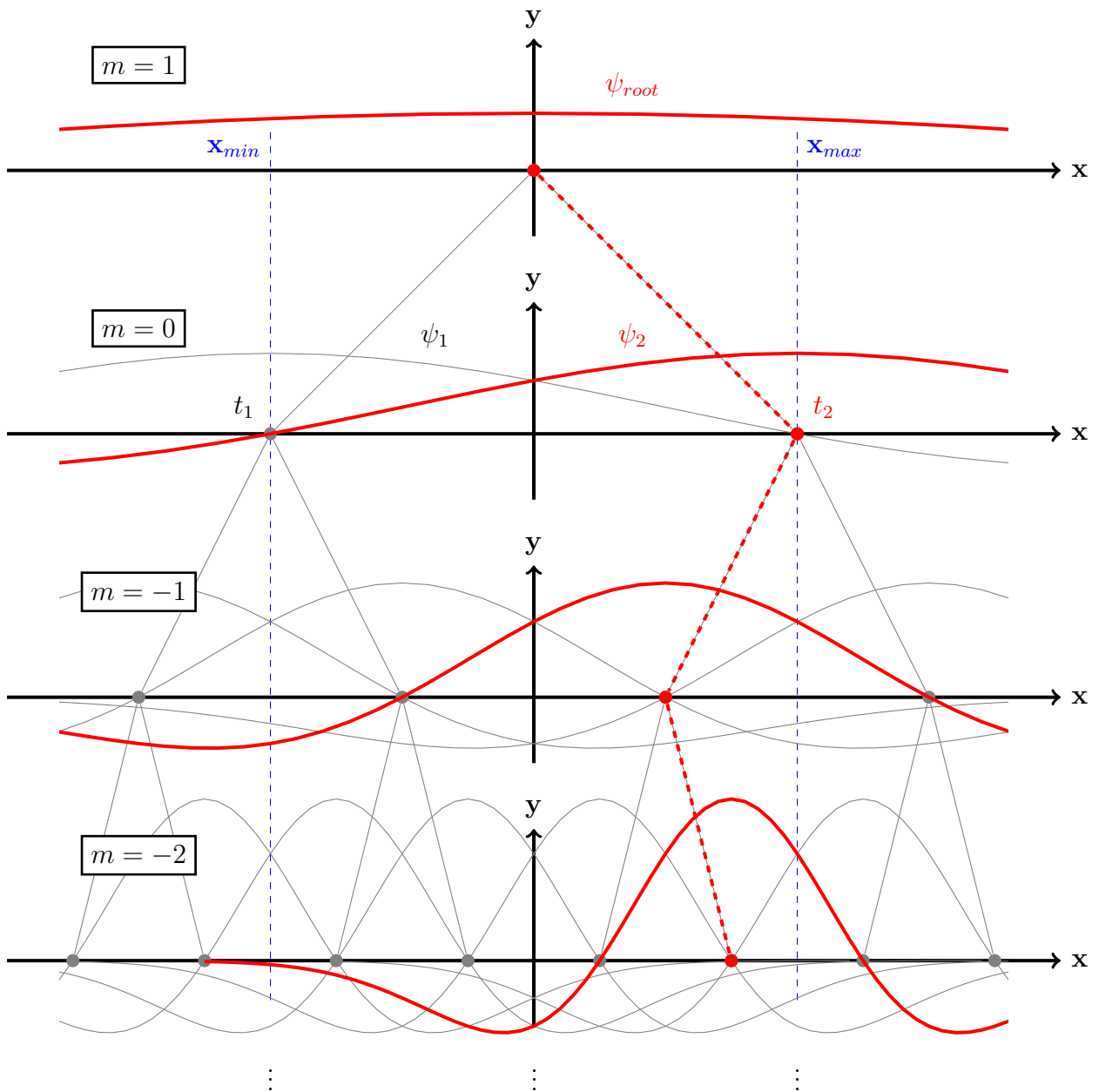


Abbildung 22: Eine eindimensionale, rekursiv erstellte Wavelet-Bibliothek

$t_2 = x_{max}$. Alle Wavelets mit gleicher Dilation $a(m)$ gemäß (7.88) bezeichnen wir als eine Schicht. ψ_1 und ψ_2 sollen die Schicht mit $m = 0$ bilden. Damit können wir Gleichung (7.89) auflösen zu

$$c = \frac{x_{max} - x_{min}}{\beta}.$$

Für das Beispiel in Abbildung 22 wurde $\beta = 1$ gewählt. Ausgehend von ψ_1 und ψ_2 kann nun rekursiv eine Wavelet-Bibliothek aufgebaut werden. Dabei wird ein Rekursionsschritt wie folgt durchgeführt:

Sei ψ_i ein Wavelet der Schicht m , bestimmt durch seine Parameter a_i und t_i . Dann werden ausgehend von ψ_i zwei benachbarte, neue Ableger-Wavelets ψ_k und ψ_l der Schicht $m - 1$ erzeugt, mit

$$a_k = a_l = c^{-1} \cdot 2^{-(m-1)} \quad \text{und} \quad t_{k/l} = t_i \pm \frac{1}{2}c \cdot 2^{m-1}\beta.$$

Man erkennt, dass (7.88) und (7.89) erfüllt sind. Den Rekursionsschritt kann man in gleicher Weise wieder mit den Ablegern ψ_k und ψ_l durchführen und so weiter. Die Rekursion wird abgebrochen, wenn eine Schicht $m = m_{min}$ erreicht ist. Zudem erkennt man, dass sich ψ_1 und ψ_2 als Ableger eines einzelnen Wavelets ψ_{root} der Lage $m = 1$ auffassen lassen mit

$$a_{root} = c^{-1} \cdot 2^{-1} \quad \text{und} \quad t_{root} = \frac{x_{min} + x_{max}}{2}.$$

Es sei erwähnt, dass es sinnvoll ist, der Bibliothek nur Wavelets hinzuzufügen, die Trainingsdaten in bestimmten Bereichen aufweisen. Damit ist gemeint, dass man zum Beispiel nur Wavelets ψ_i gemäß (7.87) zulässt, für die gilt:

$$\exists(x_j, y_j) \in \mathbb{O}_1^N : \psi_i(x_j) > \frac{1}{2} \max[\psi_i(\mathbb{R})] \quad (7.90)$$

Dabei ist $\max[\psi_i(\mathbb{R})] = \psi_i(t_i)$, also der maximale Wert, den ψ_i annehmen kann. Der Faktor $1/2$ ist hier willkürlich gewählt, wurde aber auch im Programm so umgesetzt. Dieses Vorgehen zielt lediglich darauf ab, Wavelets die wenige Trainingsdatenpunkte abdecken oder zum Spiking neigen, in der Bibliothek vermeiden. Die Wavelet-Bibliothek bleibt kleiner, wird aber nicht schlechter.

Die wesentlichen Überlegungen formulieren wir nun in algorithmischer Form:

Algorithmus 7.1 *Rekursives Erstellen einer Wavelet Bibliothek*

Seien $x_{min}, x_{max} \in \mathbb{R}$ gegeben, mit $x_{min} \leq x_i \leq x_{max}$ für alle x_i aus einem Trainingsdatensatz \mathbb{O}_1^N . Sei ferner ein Mutter-Wavelet $\psi : \mathbb{R} \rightarrow \mathbb{R}$ gegeben, ein $\beta \in \mathbb{R}^+$ und ein $m_{min} \in \mathbb{Z}$.

Rekursionsanfang: Sei

$$c = \frac{x_{max} - x_{min}}{\beta}$$

Dann ist

$$\psi_{root} = a_{root}^{\frac{1}{2}} \psi [a_{root}(x - t_{root})]$$

mit

$$a_{root} = c^{-1} \cdot 2^{-1} \quad \text{und} \quad t_{root} = \frac{x_{min} + x_{max}}{2}.$$

das Wavelet der Schicht $m = 1$.

Rekursionsschritt Sei

ψ_i ein Wavelet der Schicht m , eindeutig bestimmt durch seine Parameter a_i und t_i . Wenn $m > m_{min}$, dann füge zwei Wavelets ψ_k und ψ_l der Schicht $m - 1$ mit

$$a_k = a_l = c^{-1} \cdot 2^{-(m-1)} \quad \text{und} \quad t_{k/l} = t_i \pm \frac{1}{2} c \cdot 2^{m-1} \beta.$$

der Wavelet-Bibliothek hinzu. Wende dann den Rekursionsschritt auf ψ_k und ψ_l an.

7.1.2 Der mehrdimensionale Fall

Im mehrdimensionalen Fall wollen wir d -dimensionale Wavelets ψ_i , parametrisiert wie in (7.86), erzeugen. Jedes Wavelet ist dabei eindeutig bestimmt durch seine Parameter

$$a_i = \begin{bmatrix} a_{i,1} \\ \vdots \\ a_{i,d} \end{bmatrix} \quad \text{und} \quad t_i = \begin{bmatrix} t_{i,1} \\ \vdots \\ t_{i,d} \end{bmatrix}.$$

Ein $\beta > 0$ sei wiederum vorgegeben. Um nun Paare von (a_i, t_i) zu erzeugen, betrachten wir zunächst jede Dimension einzeln. Das heißt, wir erzeugen zunächst Paare $(a_{i,1}, t_{i,1})$, genau wie im eindimensionalen Fall. Zur ersten Komponente der x -Werte $x_1, \dots, x_N \in \mathbb{R}^d$ aus dem Trainingsdatensatz \mathbb{O}_1^N gibt es stets zwei Werte $x_{min,1}$ und $x_{max,1}$ mit

$$x_{min,1} \leq x_{i,1} \leq x_{max,1} \quad \text{für alle} \quad i = 1 \dots N$$

Ganz analog zu Algorithmus 7.1 errechnen wir dann ein c_1 und damit ein erstes Wertepaar $(a_{root,1}, t_{root,1})$. Auch die Rekursionsschritte verlaufen völlig analog. Wir erhalten dadurch eine Komponentenmenge

$$K_1 = \{(a_{i,1}, t_{i,1}) | i = 1 \dots p_1\}$$

Mit der gleichen Vorgehensweise erhalten wir in den anderen d Dimensionen Komponentenmengen K_2, \dots, K_d . Wir bilden nun das Kartesische Produkt

$$K = K_1 \times K_2 \times \dots \times K_d$$

dieser Mengen. In jedem Element von K stecken nun alle Informationen, die ein d -dimensionales Wavelet ψ_i bestimmen. Dabei haben wir alle möglichen Kombinationen von Dilationen und Translationen aus der Betrachtung in den einzelnen Dimensionen hergestellt. Bei der programmtechnischen Umsetzung kann man den Algorithmus 7.1 so erweitern, dass man einen ersten Baum für die erste Dimension erstellt (wie in Abbildung 22) und an jeden Knoten des Baumes, einschließlich der Wurzel, einen Baum für die zweite Dimension anhängt und an jeden Knoten der angehängten Bäume wiederum einen Baum für die dritte Dimension und so weiter. Auf diese Weise lässt sich der gesamte Algorithmus geschickt in eine einzige rekursive Funktion packen. Auf die Ausführung aller technischen Details sei hier verzichtet, da auf diesem Algorithmus nicht der Schwerpunkt liegen soll. Die Idee sollte jedoch klar geworden sein. Analog zu (7.90) ist es ratsam, der Wavelet-Bibliothek auch diesmal nur Wavelets hinzuzufügen, die Trainingsdaten in bestimmten Bereichen aufweisen. Das erreicht man wiederum mit einer Forderung wie:

$$\exists(x_i, y_i) \in \mathbb{O}_1^N : \psi_i(x_i) > \frac{1}{2} \max [\psi_i(\mathbb{R}^d)]$$

Dabei ist $\max [\psi_i(\mathbb{R}^d)] = \psi_i(t_i)$, analog zu (7.90). Wir vermeiden somit von vornherein Wavelets, die zum Spiking neigen.

7.2 Betrachtungen zum Mexikanerhut-Wavelet

Dieser Abschnitt befasst sich ausschließlich mit dem mehrdimensionalen Mexikanerhut-Wavelet. Das Mutter-Wavelet hat die Form

$$\psi : \mathbb{R}^d \rightarrow \mathbb{R} \quad \text{mit} \quad \psi(x) = C_d(d - \|x\|_2^2)e^{-\frac{1}{2}\|x\|_2^2} \quad (7.91)$$

Dabei ist C_d eine nur von der Dimension d abhängige Konstante, die dafür sorgt, dass

$$\|\psi\|_2 = 1$$

ist, wobei $\|\cdot\|_2$ die $L^2(\mathbb{R}^d, \mathbb{R})$ -Norm darstellt. Von diesem gemeinsamen Mutter-Wavelet werden viele verschiedene gestreckte, bzw. gestauchte und verschobene Versionen $\psi_i : \mathbb{R}^d \rightarrow \mathbb{R}$ benötigt.

Dazu brauchen wir die Parameter

$$a_i = \begin{bmatrix} a_{i1} \\ \vdots \\ a_{id} \end{bmatrix} \in \mathbb{R}_+^d \quad \text{und} \quad t_i = \begin{bmatrix} t_{i1} \\ \vdots \\ t_{id} \end{bmatrix} \in \mathbb{R}^d$$

sodass

$$\psi_i : \mathbb{R}^d \rightarrow \mathbb{R} \quad \text{mit} \quad \psi_i(x) = \Pi(a_i)^{\frac{1}{2}} \psi [a_i(x - t_i)] \quad (7.92)$$

ist. Dabei haben wir schon einige neue Schreibweisen vorweggenommen, die wir für die weiteren Betrachtungen offiziell einführen wollen. Seien l und m zwei Spaltenvektoren gemäß

$$l = \begin{bmatrix} l_1 \\ \vdots \\ l_d \end{bmatrix} \in \mathbb{R}^d \quad \text{und} \quad m = \begin{bmatrix} m_1 \\ \vdots \\ m_d \end{bmatrix} \in \mathbb{R}^d$$

Dann definieren wir das Produkt von zwei Spaltenvektoren komponentenweise, also

$$lm = l \cdot m = \begin{bmatrix} l_1 \\ \vdots \\ l_d \end{bmatrix} \cdot \begin{bmatrix} m_1 \\ \vdots \\ m_d \end{bmatrix} = \begin{bmatrix} l_1 m_1 \\ \vdots \\ l_d m_d \end{bmatrix}$$

Darauf aufbauend definieren wir die Potenz eines Spaltenvektors ebenso komponentenweise, also zum Beispiel

$$l^3 = l \cdot l \cdot l = \begin{bmatrix} l_1^3 \\ \vdots \\ l_d^3 \end{bmatrix}$$

Das Produkt der Komponenten eines Vektors wollen wir schreiben als

$$\Pi(l) = \prod_{i=1}^d l_i$$

Die Summe der Komponenten eines Vektors schreiben wir als

$$[l] = \sum_{i=1}^d l_i$$

Somit ist nicht nur (7.92) erklärt sondern auch eine für uns praktische Schreibweise geschaffen. Durch den Vorfaktor $\Pi(a_i)^{\frac{1}{2}}$ in (7.92) ist sichergestellt, dass

$$\|\psi_i\|_2 = 1$$

Ziel dieses Kapitels ist es $p_{ij} \in \mathbb{R}$ als Funktion von a_i, t_i, a_j und t_j zu berechnen sodass

$$p_{ij} = p_{ij}(a_i, t_i, a_j, t_j) = \langle \psi_i, \psi_j \rangle = \int_{\mathbb{R}^d} \psi_i(x) \psi_j(x) d\lambda(x) \quad (7.93)$$

wobei λ für das d -dimensionale Lebesgue-Maß steht. Dieses uneigentliche Integral existiert stets, weil es das Skalarprodukt im $L^2(\mathbb{R}^d, \mathbb{R})$ ist. Eine numerische Integration wäre hier sehr zeitaufwendig und würde zu unverhältnismäßigen Verzögerungen in jedem Algorithmus führen, der die p_{ij} benötigt. Zudem ist eine numerische Integration immer nur eine Näherung an den wahren Wert des Integrals. In diesem Kapitel soll deshalb das Integral in (7.93) analytisch gelöst werden, sodass am Ende ein algebraischer Ausdruck für p_{ij} zur Verfügung steht. Dieser Ausdruck ist nicht leicht herzuleiten und wird sehr komplex sein. Wir werden jedoch am Ende sehen, dass er recht effizient berechenbar ist. Die Herleitung von p_{ij} erklären wir sukzessive in mehreren Schritten. Am Anfang werden einfache Integrale berechnet, mit Hilfe derer dann immer komplexere Integrale ausgewertet werden bis schließlich das Integral des Skalarproduktes p_{ij} aufgelöst werden kann. Am Ende jedes Abschnittes steht eine Zusammenfassung der wichtigen Ergebnisse.

7.2.1 Grundlegende Integrale

In diesem Abschnitt berechnen wir Integrale der Form

$$\int_{-\infty}^{\infty} x^n e^{-\frac{1}{2}x^2} dx$$

für die ganzzahligen Werte $n = 0 \dots 4$ als Grundlage für die weiteren Überlegungen.

Wir beginnen mit

$$\begin{aligned} \int_{-\infty}^{\infty} e^{-\frac{1}{2}x^2} dx &= \sqrt{\int_{x_1 \in \mathbb{R}} \int_{x_2 \in \mathbb{R}} e^{-\frac{1}{2}x_1^2} e^{-\frac{1}{2}x_2^2} dx_2 dx_1} \\ &= \sqrt{\int_{\underline{x} \in \mathbb{R}^2} e^{-\frac{1}{2}\|\underline{x}\|_2^2} d\underline{x}} \\ &= \sqrt{\int_{r=0}^{\infty} \int_{\varphi=0}^{2\pi} r \cdot e^{-\frac{1}{2}r^2} d\varphi dr} \\ &= \sqrt{2\pi \int_{r=0}^{\infty} r \cdot e^{-\frac{1}{2}r^2} dr} \\ &= \sqrt{2\pi \left[-e^{-\frac{1}{2}r^2} \right]_0^{\infty}} \\ &= \sqrt{2\pi} \end{aligned} \tag{7.94}$$

Im nächsten Schritt wollen wir die Integrale

$$\int_{-\infty}^{\infty} x^2 e^{-\frac{1}{2}x^2} dx \quad \text{und} \quad \int_{-\infty}^{\infty} x^4 e^{-\frac{1}{2}x^2} dx$$

berechnen. Dies geschieht mittels Produktintegration mit der Formel

$$\int u dv = uv - \int v du$$

Damit und mit (7.94) berechnet sich wie folgt

$$\begin{aligned} \int_{-\infty}^{\infty} x^2 e^{-\frac{1}{2}x^2} dx &= \int_{-\infty}^{\infty} \underbrace{(-x)}_u \cdot \underbrace{(-x \cdot e^{-\frac{1}{2}x^2})}_{dv} dx \\ &= \left[\underbrace{-x}_u \cdot \underbrace{e^{-\frac{1}{2}x^2}}_v \right]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} \underbrace{-1}_{du} \cdot \underbrace{e^{-\frac{1}{2}x^2}}_v dx \\ &= \sqrt{2\pi} \end{aligned}$$

Dieses Ergebnis können wir im nächsten Schritt nutzen und erhalten auf analoge Weise

$$\begin{aligned} \int_{-\infty}^{\infty} x^4 e^{-\frac{1}{2}x^2} dx &= \int_{-\infty}^{\infty} \underbrace{(-x^3)}_u \cdot \underbrace{(-x \cdot e^{-\frac{1}{2}x^2})}_{dv} dx \\ &= \left[\underbrace{-x^3}_u \cdot \underbrace{e^{-\frac{1}{2}x^2}}_v \right]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} \underbrace{-3x^2}_{du} \cdot \underbrace{e^{-\frac{1}{2}x^2}}_v dx \\ &= 3\sqrt{2\pi} \end{aligned}$$

Die letzten beiden Integrale

$$\int_{-\infty}^{\infty} x \cdot e^{-\frac{1}{2}x^2} dx = \int_{-\infty}^{\infty} x^3 e^{-\frac{1}{2}x^2} dx = 0$$

sind Null, weil die Integranden ungerade Funktionen sind. Die Ergebnisse dieses Abschnittes sind also

$$\int_{-\infty}^{\infty} x^n e^{-\frac{1}{2}x^2} dx = \sqrt{2\pi} \cdot \begin{cases} 1 & \text{für } n = 0 \\ 0 & \text{für } n = 1 \\ 1 & \text{für } n = 2 \\ 0 & \text{für } n = 3 \\ 3 & \text{für } n = 4 \end{cases} \quad (7.95)$$

7.2.2 Mehrdimensionale Integrale

Wir gehen nun über zur nächsten Komplexitätsstufe und stellen uns ein Repertoire von mehrdimensionalen Hilfsintegralen zur Verfügung, die wir später benötigen werden. Sei

$$u = \begin{bmatrix} u_1 \\ \vdots \\ u_d \end{bmatrix} \in \mathbb{R}^d \quad \text{und} \quad f : \mathbb{R}^d \rightarrow \mathbb{R}$$

Wir werden hier Integrale der Form

$$\int_{u \in \mathbb{R}^d} f(u) e^{-\frac{1}{2}\|u\|_2^2} du$$

betrachten. Dabei ist du eine Kurzschreibweise für $d\lambda(u)$ mit dem d -dimensionalen Lebesgue-Maß λ . Bei den hier betrachteten Integralen können wir stets in Riemann-Integrale auflösen. Das heißt im folgenden wird stets gelten

$$\begin{aligned} & \int_{u \in \mathbb{R}^d} f(u) e^{-\frac{1}{2}\|u\|_2^2} du \\ &= \int_{u_1=-\infty}^{\infty} \cdots \int_{u_d=-\infty}^{\infty} f(u) \cdot e^{-\frac{1}{2}u_d^2} du_d \cdots e^{-\frac{1}{2}u_1^2} du_1 \end{aligned}$$

Mit den Ergebnissen des letzten Abschnittes erhalten wir das erste benötigte Integral

$$\int_{u \in \mathbb{R}^d} e^{-\frac{1}{2}\|u\|_2^2} du = \prod_{i=1}^d \int_{u_i=-\infty}^{\infty} e^{-\frac{1}{2}u_i^2} du_i = \sqrt{2\pi}^d$$

Wir führen nun die beiden Variablen

$$a = \begin{bmatrix} a_1 \\ \vdots \\ a_d \end{bmatrix} \in \mathbb{R}_+^d \quad \text{und} \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_d \end{bmatrix} \in \mathbb{R}^d$$

ein. Mit den am Anfang dieses Kapitels eingeführten Schreibweisen berechnen wir für die ganzzahligen Werte $n = 0 \dots 4$ nun

$$\begin{aligned} & \int_{u \in \mathbb{R}^d} [au^n] e^{-\frac{1}{2}\|u\|_2^2} du \\ &= \sum_{i=1}^d \int_{u \in \mathbb{R}^d} a_i u_i^n e^{-\frac{1}{2}\|u\|_2^2} du \end{aligned}$$

$$\begin{aligned}
 &= \sum_{i=1}^d a_i \left[\underbrace{\int_{u_i=-\infty}^{\infty} u_i^n e^{-\frac{1}{2}u_i^2} du_i}_{\text{siehe (7.95)}} \prod_{\substack{j=1 \\ j \neq i}}^d \underbrace{\int_{u_j=-\infty}^{\infty} e^{-\frac{1}{2}u_j^2} du_j}_{\sqrt{2\pi}} \right] \\
 &= [a] \sqrt{2\pi}^d \cdot \begin{cases} 1 & \text{für } n = 0 \\ 0 & \text{für } n = 1 \\ 1 & \text{für } n = 2 \\ 0 & \text{für } n = 3 \\ 3 & \text{für } n = 4 \end{cases} \quad (7.96)
 \end{aligned}$$

Wir benötigen für den späteren Gebrauch noch Integrale der Form

$$\int_{u \in \mathbb{R}^d} [au^p] [bu^q] e^{-\frac{1}{2}\|u\|_2^2} du$$

Dabei interessieren uns genau genommen nur die drei Fälle $(p, q) = (1, 1)$, $(p, q) = (1, 2)$, und $(p, q) = (2, 2)$.

$$\begin{aligned}
 &\int_{u \in \mathbb{R}^d} [au^p] [bu^q] e^{-\frac{1}{2}\|u\|_2^2} du \\
 &= \sum_{i=1}^d \sum_{j=1}^d \int_{u \in \mathbb{R}^d} a_i b_j u_i^p u_j^q e^{-\frac{1}{2}\|u\|_2^2} du \\
 &= \sum_{\substack{i,j=1 \\ i \neq j}}^d \int_{u \in \mathbb{R}^d} a_i b_j u_i^p u_j^q e^{-\frac{1}{2}\|u\|_2^2} du + \underbrace{\sum_{i=1}^d \int_{u \in \mathbb{R}^d} a_i b_i u_i^{p+q} e^{-\frac{1}{2}\|u\|_2^2} du}_{\text{siehe (7.96)}} \\
 &= \sum_{\substack{i,j=1 \\ i \neq j}}^d a_i b_j \left[\underbrace{\int_{u_i=-\infty}^{\infty} u_i^p e^{-\frac{1}{2}u_i^2} du_i}_{\substack{0 \text{ für } p=1 \\ \sqrt{2\pi} \text{ für } p=2}} \underbrace{\int_{u_j=-\infty}^{\infty} u_j^q e^{-\frac{1}{2}u_j^2} du_j}_{\substack{0 \text{ für } q=1 \\ \sqrt{2\pi} \text{ für } q=2}} \prod_{\substack{k=1 \\ i \neq k \neq j}}^d \underbrace{\int_{u_k=-\infty}^{\infty} e^{-\frac{1}{2}u_k^2} du_k}_{\sqrt{2\pi}} \right] \\
 &+ [ab] \sqrt{2\pi}^d \cdot \begin{cases} 1 & \text{für } p+q=2 \\ 0 & \text{für } p+q=3 \\ 3 & \text{für } p+q=4 \end{cases} \\
 &= \sqrt{2\pi}^d \cdot \begin{cases} [ab] & \text{für } p=1 \wedge q=1 \\ 0 & \text{für } p=1 \wedge q=2 \\ [a][b] + 2[ab] & \text{für } p=2 \wedge q=2 \end{cases}
 \end{aligned}$$

Wir fassen die wichtigen Ergebnisse dieses Abschnittes wieder zusammen

$$\begin{aligned}
 \int_{u \in \mathbb{R}^d} e^{-\frac{1}{2}\|u\|_2^2} du &= \sqrt{2\pi}^d \\
 \int_{u \in \mathbb{R}^d} [au^n] e^{-\frac{1}{2}\|u\|_2^2} du &= \sqrt{2\pi}^d \cdot \begin{cases} 0 & \text{für } n = 1 \\ [a] & \text{für } n = 2 \end{cases} \\
 \int_{u \in \mathbb{R}^d} [au^p] [bu^q] e^{-\frac{1}{2}\|u\|_2^2} du &= \sqrt{2\pi}^d \cdot \begin{cases} [ab] & \text{für } p = 1 \wedge q = 1 \\ 0 & \text{für } p = 1 \wedge q = 2 \\ [a] [b] + 2 [ab] & \text{für } p = 2 \wedge q = 2 \end{cases}
 \end{aligned} \tag{7.97}$$

7.2.3 Das Skalarprodukt zweier Wavelets

Nachdem uns nun komplexere Hilfsintegrale zur Verfügung stehen, nehmen wir uns die p_{ij} vor. Konkret heißt das

$$\begin{aligned}
 p_{ij} &= \langle \psi_i, \psi_j \rangle \\
 &= \int_{x \in \mathbb{R}^d} \psi_i(x) \psi_j(x) dx
 \end{aligned}$$

Die ψ_i und ψ_j können wir gemäß (7.92) ausdrücken und erhalten weiter

$$\begin{aligned}
 p_{ij} &= \int_{x \in \mathbb{R}^d} \Pi(a_i)^{\frac{1}{2}} \psi[a_i(x-t_i)] \Pi(a_j)^{\frac{1}{2}} \psi[a_j(x-t_j)] dx \\
 &= \Pi(a_i)^{\frac{1}{2}} \Pi(a_j)^{\frac{1}{2}} \cdot \\
 &\quad \int_{x \in \mathbb{R}^d} C_d(d - \|a_i(x-t_i)\|_2^2) e^{-\frac{1}{2}\|a_i(x-t_i)\|_2^2} C_d(d - \|a_j(x-t_j)\|_2^2) e^{-\frac{1}{2}\|a_j(x-t_j)\|_2^2} dx \\
 &= C_d^2 \Pi(a_i a_j)^{\frac{1}{2}} \cdot \\
 &\quad \int_{x \in \mathbb{R}^d} (d - [a_i^2(x-t_i)^2]) \cdot (d - [a_j^2(x-t_j)^2]) e^{-\frac{1}{2}[[a_i^2(x-t_i)^2] + [a_j^2(x-t_j)^2]]} dx
 \end{aligned} \tag{7.98}$$

An dieser Stelle widmen wir unsere gesamte Aufmerksamkeit dem letzten Exponenten. Es gilt diesen auf eine Form zu bringen, sodass wir die zuvor berechneten Teilintegrale anwenden können. Konkret wollen wir das x auf geeignete Weise so substituieren, dass

$$-\frac{1}{2} [u^2 + c] \stackrel{!}{=} -\frac{1}{2} [[a_i^2(x-t_i)^2] + [a_j^2(x-t_j)^2]] \tag{7.99}$$

Dabei soll c ein beliebiger konstanter Vektor sein. Dazu substituieren wir konkret

$$x := \delta(u - \mu)$$

mit geeigneten $\delta, \mu \in \mathbb{R}^d$ und die vorherige Forderung wird zu

$$\begin{aligned} [u^2 + c] &\stackrel{!}{=} [a_i^2 [\delta(u - \mu) - t_i]^2 + a_j^2 [\delta(u - \mu) - t_j]^2] \\ &= [a_i^2 [\delta u - (\delta\mu + t_i)]^2 + a_j^2 [\delta u - (\delta\mu + t_j)]^2] \\ &= \left[\underbrace{(a_i^2 + a_j^2) \delta^2}_{\stackrel{!}{=}1} \cdot u^2 - 2\delta \underbrace{[a_i^2 (\delta\mu + t_i) + a_j^2 (\delta\mu + t_j)]}_{\stackrel{!}{=}0} \cdot u \right. \\ &\quad \left. + \underbrace{a_i^2 (\delta\mu + t_i)^2 + a_j^2 (\delta\mu + t_j)^2}_c \right] \end{aligned} \quad (7.100)$$

Durch Koeffizientenvergleich können nun die Werte δ , μ und c bestimmt werden. Durch Vergleich des Koeffizienten vor u^2 erhalten wir

$$\delta = (a_i^2 + a_j^2)^{-\frac{1}{2}} \quad (7.101)$$

Dabei erkennt man, dass alle Komponenten des Vektors δ echt positiv sind. Den Wert μ erhalten wir durch Vergleich des Koeffizienten vor u

$$\begin{aligned} 0 &\stackrel{!}{=} \delta [a_i^2 (\delta\mu + t_i) + a_j^2 (\delta\mu + t_j)] \\ &= \underbrace{(a_i^2 + a_j^2)}_1 \delta^2 \mu + \delta (a_i^2 t_i + a_j^2 t_j) \\ \Leftrightarrow \mu &= -\delta (a_i^2 t_i + a_j^2 t_j) \end{aligned}$$

Damit können wir auch den Ausdruck für c noch etwas vereinfachen zu

$$\begin{aligned} c &= a_i^2 (\delta\mu + t_i)^2 + a_j^2 (\delta\mu + t_j)^2 \\ &= \delta^2 [a_i^2 + a_j^2] [a_i^2 (\delta\mu + t_i)^2 + a_j^2 (\delta\mu + t_j)^2] \\ &= \delta^2 [a_i^4 (\delta\mu + t_i)^2 + a_j^4 (\delta\mu + t_j)^2 \\ &\quad + a_i^2 a_j^2 (\delta\mu + t_i)^2 + a_i^2 a_j^2 (\delta\mu + t_j)^2] \\ &= \delta^2 [a_i^4 (\delta\mu + t_i)^2 + 2a_i^2 a_j^2 (\delta\mu + t_i) (\delta\mu + t_j) + a_j^4 (\delta\mu + t_j)^2 \\ &\quad + a_i^2 a_j^2 (\delta\mu + t_i)^2 - 2a_i^2 a_j^2 (\delta\mu + t_i) (\delta\mu + t_j) + a_i^2 a_j^2 (\delta\mu + t_j)^2] \\ &= \delta^2 \left[\left(\underbrace{a_i^2 (\delta\mu + t_i) + a_j^2 (\delta\mu + t_j)}_{=0 \text{ (siehe 7.100)}} \right)^2 + a_i^2 a_j^2 (\delta\mu + t_i - \delta\mu - t_j)^2 \right] \\ &= \delta^2 a_i^2 a_j^2 t^2 \end{aligned} \quad (7.102)$$

wobei $\bar{t} = t_j - t_i$ ist. Wir haben nun die Substitution der x durch u fertig vorbereitet und die Hilfsgrößen δ , μ und c bestimmt. Damit wollen wir an das letzte Integral in (7.98) anknüpfen. Wenn wir die Integrationsvariable x substituieren, folgern wir aus (7.99), dass

$$\begin{aligned} x &= \delta(u - \mu) \\ \Rightarrow dx &= \Pi(\delta)du \\ \text{also } \int_{x \in \mathbb{R}^d} f[x]dx &= \int_{u \in \mathbb{R}^d} f[x(u)]\Pi(\delta)du \end{aligned}$$

Der Faktor $\Pi(\delta)$ kommt daher, weil sich die Integrale über \mathbb{R}^d in d in einander geschachtelte Riemann-Integrale auflösen lassen, worin je eine Komponente von x durch die zugehörige Komponente von $\delta(u - \mu)$ substituiert wird. Wir greifen nun Gleichung (7.98) wieder auf und schreiben weiter

$$\begin{aligned} p_{ij} &= C_d^2 \Pi \left(a_i^{\frac{1}{2}} a_j^{\frac{1}{2}} \right) \cdot \\ &\int_{u \in \mathbb{R}^d} (d - \lfloor a_i^2 (x - t_i)^2 \rfloor) \cdot (d - \lfloor a_j^2 (x - t_j)^2 \rfloor) e^{-\frac{1}{2} \lfloor u^2 + c \rfloor} \Pi(\delta) du \\ &= C_d^2 \Pi \left(\delta a_i^{\frac{1}{2}} a_j^{\frac{1}{2}} \right) e^{-\frac{1}{2} \lfloor c \rfloor} \cdot \left[\underbrace{\int_{u \in \mathbb{R}^d} d^2 e^{-\frac{1}{2} \lfloor u^2 \rfloor} du}_{=d^2 \sqrt{2\pi}^d \text{ nach (7.97)}} \right. \\ &\quad + \int_{u \in \mathbb{R}^d} \underbrace{-d \lfloor a_i^2 (x - t_i)^2 + a_j^2 (x - t_j)^2 \rfloor}_{= \lfloor u^2 + c \rfloor \text{ nach (7.99)}} e^{-\frac{1}{2} \lfloor u^2 \rfloor} du \\ &\quad \left. + \underbrace{\int_{u \in \mathbb{R}^d} \lfloor a_i^2 (x - t_i)^2 \rfloor \cdot \lfloor a_j^2 (x - t_j)^2 \rfloor e^{-\frac{1}{2} \lfloor u^2 \rfloor} du}_{=h} \right] \\ &= C_d^2 \Pi \left(\delta a_i^{\frac{1}{2}} a_j^{\frac{1}{2}} \right) e^{-\frac{1}{2} \lfloor c \rfloor} \cdot \\ &\quad \left[\underbrace{d^2 \sqrt{2\pi}^d - d \int_{u \in \mathbb{R}^d} \lfloor u^2 \rfloor e^{-\frac{1}{2} \lfloor u^2 \rfloor} du}_{=d \sqrt{2\pi}^d \text{ nach (7.97)}} - d \cdot \lfloor c \rfloor \underbrace{\int_{u \in \mathbb{R}^d} e^{-\frac{1}{2} \lfloor u^2 \rfloor} du}_{=\sqrt{2\pi}^d \text{ nach (7.97)}} + h \right] \\ &= C_d^2 \Pi \left(\delta a_i^{\frac{1}{2}} a_j^{\frac{1}{2}} \right) e^{-\frac{1}{2} \lfloor c \rfloor} \cdot \left[-d \cdot \lfloor c \rfloor \sqrt{2\pi}^d + h \right] \end{aligned} \tag{7.103}$$

Bis hierhin haben wir schon wesentliche Teile von p_{ij} berechnet. Wir müssen nur noch h auswerten.

$$h = \int_{u \in \mathbb{R}^d} \underbrace{[a_i^2 (x - t_i)^2] \cdot [a_j^2 (x - t_j)^2]}_{p(u)} e^{-\frac{1}{2}[u^2]} du$$

Dabei konzentrieren wir uns zunächst einmal auf $p(u)$, eine Art Polynom. Wir werden diesen Ausdruck durch Ausmultiplizieren jetzt so in Summanden zerlegen, dass die einzelnen Summanden integrierbar sind.

$$\begin{aligned} p(u) &= [a_i^2 (x - t_i)^2] \cdot [a_j^2 (x - t_j)^2] \\ &= [a_i^2 [\delta(u - \mu) - t_i]^2] \cdot [a_j^2 [\delta(u - \mu) - t_j]^2] \\ &= [a_i^2 [\delta u - (\delta\mu + t_i)]^2] \cdot [a_j^2 [\delta u - (\delta\mu + t_j)]^2] \end{aligned}$$

Dabei können wir gleich noch die Ausdrücke $\delta\mu + t_i$ und $\delta\mu + t_j$ anders darstellen, sodass kein μ mehr darin vorkommt. Mit $\bar{t} = t_j - t_i$ gilt

$$\delta\mu + t_i = -\delta^2 (a_i^2 t_i + a_j^2 t_j) + \delta^2 (a_i^2 + a_j^2) t_i = \delta^2 (a_j^2 t_i - a_i^2 t_j) = -\delta^2 a_j^2 \bar{t}$$

Analog gilt

$$\delta\mu + t_j = -\delta^2 (a_i^2 t_i + a_j^2 t_j) + \delta^2 (a_i^2 + a_j^2) t_j = \delta^2 (a_i^2 t_j - a_i^2 t_i) = \delta^2 a_i^2 \bar{t}$$

Nach diesem kurzen Einschub berechnen wir $p(u)$ weiter

$$\begin{aligned} p(u) &= [a_i^2 [\delta u + \delta^2 a_j^2 \bar{t}]^2] \cdot [a_j^2 [\delta u - \delta^2 a_i^2 \bar{t}]^2] \\ &= [\delta^2 a_i^2 u^2 + 2\delta^3 a_i^2 a_j^2 \bar{t} u + \delta^4 a_i^2 a_j^4 \bar{t}^2] \\ &\quad \cdot [\delta^2 a_j^2 u^2 - 2\delta^3 a_i^2 a_j^2 \bar{t} u + \delta^4 a_i^4 a_j^2 \bar{t}^2] \\ p(u) &= [\delta^2 a_i^2 u^2] \cdot [\delta^2 a_j^2 u^2] \\ &\quad + [\delta^2 a_i^2 u^2] \cdot [-2\delta^3 a_i^2 a_j^2 \bar{t} u] \\ &\quad + [\delta^2 a_i^2 u^2] \cdot [\delta^4 a_i^4 a_j^2 \bar{t}^2] \\ &\quad + [2\delta^3 a_i^2 a_j^2 \bar{t} u] \cdot [\delta^2 a_j^2 u^2] \\ &\quad + [2\delta^3 a_i^2 a_j^2 \bar{t} u] \cdot [-2\delta^3 a_i^2 a_j^2 \bar{t} u] \\ &\quad + [2\delta^3 a_i^2 a_j^2 \bar{t} u] \cdot [\delta^4 a_i^4 a_j^2 \bar{t}^2] \\ &\quad + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \cdot [\delta^2 a_j^2 u^2] \\ &\quad + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \cdot [-2\delta^3 a_i^2 a_j^2 \bar{t} u] \\ &\quad + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \cdot [\delta^4 a_i^4 a_j^2 \bar{t}^2] \end{aligned}$$

Wir haben nun $p(u)$ so in neun Summanden zerlegt, dass wir das Integral

$$h = \int_{u \in \mathbb{R}^d} p(u) e^{-\frac{1}{2}[u^2]} du$$

in neun Integrale aufteilen können von denen jedes Einzelne eine bereits bekannte Form gemäß (7.97) besitzt. Damit können wir nun h berechnen

$$\begin{aligned} h &= \sqrt{2\pi}^d \cdot \left([\delta^2 a_i^2] \cdot [\delta^2 a_j^2] + 2 [\delta^4 a_i^2 a_j^2] + 0 \right. \\ &\quad + [\delta^2 a_i^2] \cdot [\delta^4 a_i^4 a_j^2 \bar{t}^2] + 0 \\ &\quad + [-4\delta^6 a_i^4 a_j^4 \bar{t}^2] + 0 \\ &\quad + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \cdot [\delta^2 a_j^2] + 0 \\ &\quad \left. + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \cdot [\delta^4 a_i^4 a_j^2 \bar{t}^2] \right) \\ &= \sqrt{2\pi}^d \cdot \left(\left([\delta^2 a_i^2] + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \right) \cdot \left([\delta^2 a_j^2] + [\delta^4 a_i^4 a_j^2 \bar{t}^2] \right) \right. \\ &\quad \left. + 2 [\delta^4 a_i^2 a_j^2] - 4 [\delta^6 a_i^4 a_j^4 \bar{t}^2] \right) \end{aligned}$$

Da wir nun h berechnet haben, können wir an (7.103) anschließen und endlich p_{ij} ausdrücken

$$\begin{aligned} p_{ij} &= C_d^2 \sqrt[4]{\Pi(\delta^4 a_i^2 a_j^2)} \sqrt{2\pi}^d e^{-\frac{1}{2}|c|} \cdot \left[-d \cdot |c| + 2 [\delta^4 a_i^2 a_j^2] - 4 [\delta^6 a_i^4 a_j^4 \bar{t}^2] \right. \\ &\quad \left. + \left([\delta^2 a_i^2] + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \right) \cdot \left([\delta^2 a_j^2] + [\delta^4 a_i^4 a_j^2 \bar{t}^2] \right) \right] \end{aligned} \quad (7.104)$$

An dieser Stelle fällt auf relativ einfache Weise der Parameter C_d aus (7.91) ab, denn für $\psi_1 = \psi_2 = \psi$ ist

$$p_{12} = \langle \psi, \psi \rangle = 1$$

In diesem Spezialfall ist

$$\bar{t} = t_1 = t_2 = \underline{0} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \quad \text{und} \quad a_1 = a_2 = \underline{1} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Damit berechnen sich δ^2 nach (7.101) und c nach (7.102) zu

$$\delta^2 = (a_1^2 + a_2^2)^{-1} = \begin{bmatrix} 1/2 \\ \vdots \\ 1/2 \end{bmatrix} \quad \text{und} \quad c = \delta^2 a_1^2 a_2^2 \bar{t}^2 = \underline{0} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

Wenn wir das nun in Formel (7.104) einsetzen, erhalten wir

$$\begin{aligned} 1 = p_{12} &= C_d^2 \sqrt[4]{\left(\frac{1}{4}\right)^d} \sqrt{2\pi}^d \cdot \left[2\frac{d}{4} + \left(\frac{d}{2}\right) \cdot \left(\frac{d}{2}\right) \right] \\ &= C_d^2 \sqrt{\pi}^d \cdot \frac{d^2 + 2d}{4} \end{aligned}$$

Umgestellt nach C_d ergibt sich

$$C_d = \pi^{-\frac{d}{4}} \cdot \frac{2}{\sqrt{d^2 + 2d}} \quad (7.105)$$




Zuletzt fassen wir alle Teile zusammen und erhalten das zentrale Ergebnis dieses Abschnittes

Das Skalarprodukt p_{ij} zweier d -dimensionaler, normierter Mexikanerhut-Wavelets ψ_i und ψ_j ist

$$\begin{aligned} p_{ij} &= \langle \psi_i, \psi_j \rangle \\ &= C_d^2 \sqrt[4]{\Pi(\delta^4 a_i^2 a_j^2)} \sqrt{2\pi^d} e^{-\frac{1}{2} [\delta^2 a_i^2 a_j^2 \bar{t}^2]} \\ &\quad \cdot \left[-d \cdot [\delta^2 a_i^2 a_j^2 \bar{t}^2] + 2 [\delta^4 a_i^2 a_j^2] - 4 [\delta^6 a_i^4 a_j^4 \bar{t}^2] \right. \\ &\quad \left. + \left([\delta^2 a_i^2] + [\delta^4 a_i^2 a_j^4 \bar{t}^2] \right) \cdot \left([\delta^2 a_j^2] + [\delta^4 a_i^4 a_j^2 \bar{t}^2] \right) \right] \end{aligned} \quad (7.106)$$

wobei $a_i, t_i, a_j, t_j \in \mathbb{R}^d$ die zugehörigen Dilations- und Translationsvektoren sind und $\bar{t} = t_j - t_i$, $\delta = (a_i^2 + a_j^2)^{-\frac{1}{2}}$ und $C_d = \pi^{-\frac{d}{4}} \cdot \frac{2}{\sqrt{d^2 + 2d}}$ ist.

7.2.4 Rechenaufwand für das Skalarprodukt zweier Wavelets

Im letzten Abschnitt haben wir einen algebraischen Ausdruck für das Skalarprodukt p_{ij} zweier normierter Mexikanerhut-Wavelets gefunden. Dieser Ausdruck sieht zunächst sehr komplex aus. Der numerische Rechenaufwand lässt sich dennoch stark begrenzen. Die Konstante $C_d^2 \cdot \sqrt{2\pi^d}$ hängt nur von der Dimension d ab und muss daher nur ein einziges Mal pro Trainingsdatensatz berechnet werden. Wie die komplexeren Terme in der Formel für p_{ij} berechnet werden, zeigt Abbildung 23. Dabei sind die gegebenen Größen blaue Rechtecke . Zwischengrößen sind erdfarbene Kreise oder Ellipsen  und Ergebnisterme sind lachsfarbene Ellipsen . Alle Ergebnisterme werden in der Formel für p_{ij} benötigt. Zur Berechnung einer Zwischen- oder Ausgangsgröße sind genau d Multiplikationen notwendig. Bei der Berechnung von \bar{t}^2 sind zusätzlich d Subtraktionen durchzuführen. Bei δ^2 sind es statt Multiplikationen d Divisionen und ebenso viele Additionen. Insgesamt sind also zum Erreichen der Ergebnisterme $12d$ Multiplikationen/Divisionen und $2d$ Additionen/Subtraktionen notwendig. Für die Summenbildung der Ergebnisterme ($[\cdot]$) werden $7d$ Additionen benötigt und für das Produkt $\Pi(\delta^4 a_i^2 a_j^2)$ sind es d Multiplikationen. Der restliche Rechenaufwand beläuft sich auf eine Berechnung von $\sqrt[4]{\cdot}$, eine Potenz von e , acht Multiplikationen und vier Additionen.

Zusammengefasst ergibt sich also

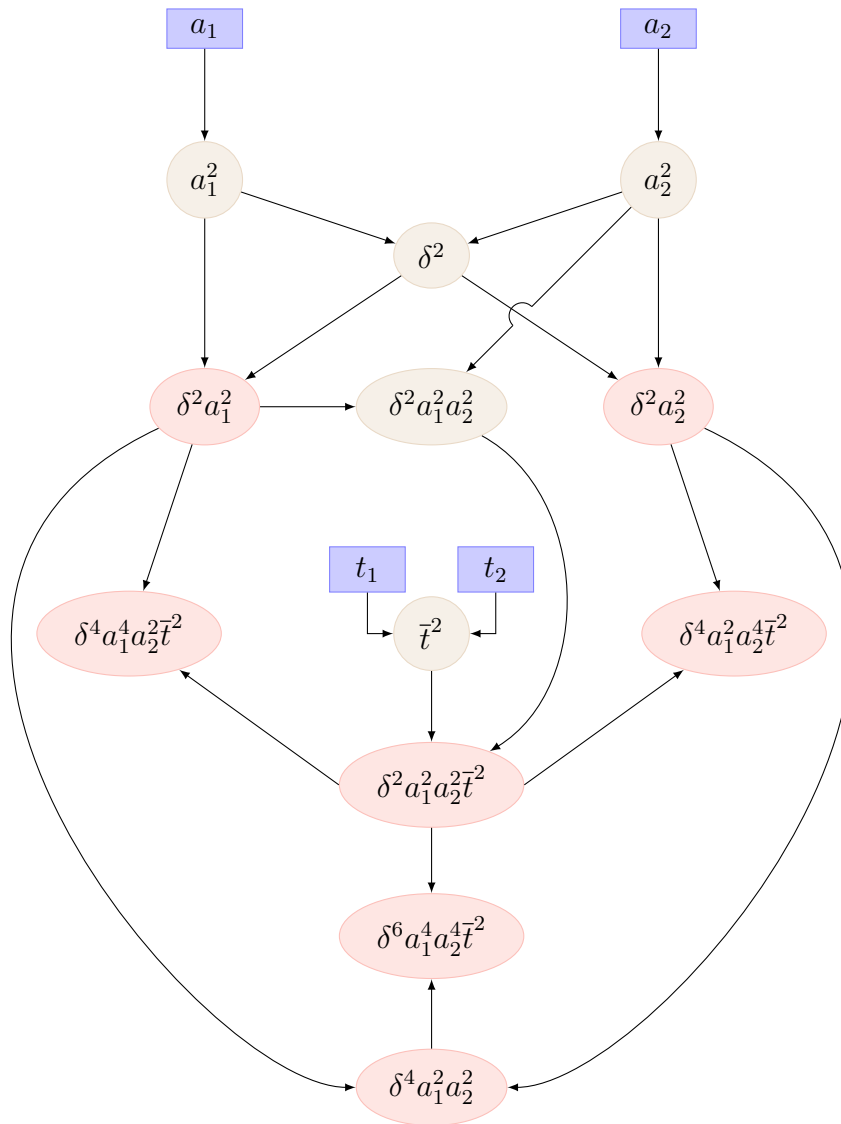


Abbildung 23: Schritte zur Berechnung von p_{ij}

Der numerische Aufwand zur Berechnung des Skalarproduktes zweier Wavelets $p_{ij} = \langle \psi_i, \psi_j \rangle$ umfasst

- $12d + 8$ Multiplikationen/Divisionen
- $9d + 4$ Additionen/Subtraktionen
- eine vierte Wurzel
- eine Potenz von e

7.3 Effiziente Matrixinversion

Die numerische Inversion einer invertierbaren $r \times r$ -Matrix \mathbf{S}_r benötigt im allgemeinen eine Anzahl von Rechenschritten der Ordnung $O(r^3)$, ist also sehr rechenaufwendig. Im Abschnitt 7.5 werden wir sehr viele Inversionen von speziellen Matrizen \mathbf{S}_{r+1} mit größeren Werten für r durchführen müssen, wodurch wir sehr schnell ausufernde Rechenzeiten erhalten würden. An dieser Stelle soll gezeigt werden, wie unter den besonderen Rahmenbedingungen, die wir in den Abschnitten 7.5 und 7.7 noch vorfinden werden, der Rechenaufwand für eine Matrixinversion zunächst auf die Ordnung $O(r^2)$ reduziert werden kann. Sei

$$\mathbf{V}_r \in \mathbb{R}^{n \times r} \quad \text{mit} \quad \mathbf{V}_r = (\underline{v}_1, \dots, \underline{v}_r)$$

wobei die \underline{v}_i linear unabhängige Spaltenvektoren sind mit

$$\underline{v}_i \in \mathbb{R}^n, \quad i = 1 \dots r \quad \text{und} \quad r < n$$

Somit hat \mathbf{V}_r vollen Rang und die $r \times r$ -Matrix \mathbf{S}_r mit

$$\mathbf{S}_r = \mathbf{V}_r^T \mathbf{V}_r$$

ist symmetrisch, positiv definit und somit invertierbar. Wir nehmen nun an, dass die Matrix $\mathbf{\Lambda}_r = \mathbf{S}_r^{-1}$ bereits berechnet wurde. Wir erhalten nun einen zusätzlichen Spaltenvektor \underline{v}_{r+1} und damit eine Matrix

$$\mathbf{V}_{r+1} = (\mathbf{V}_r, \underline{v}_{r+1}) = (\underline{v}_1, \dots, \underline{v}_r, \underline{v}_{r+1})$$

mit der wir wiederum eine Matrix \mathbf{S}_{r+1} berechnen können mit

$$\mathbf{S}_{r+1} = \mathbf{V}_{r+1}^T \mathbf{V}_{r+1} = \begin{pmatrix} \mathbf{S}_r & \mathbf{V}_r^T \underline{v}_{r+1} \\ \underline{v}_{r+1}^T \mathbf{V}_r & \underline{v}_{r+1}^T \underline{v}_{r+1} \end{pmatrix}$$

Unter diesen speziellen Voraussetzungen gilt es nun die inverse Matrix $\mathbf{\Lambda}_{r+1} = \mathbf{S}_{r+1}^{-1}$ effizient zu berechnen. Zunächst substituieren wir

$$\begin{aligned} \underline{a} &= \mathbf{V}_r^T \underline{v}_{r+1} \\ \underline{b} &= \underline{v}_{r+1}^T \underline{v}_{r+1} \end{aligned}$$

sodass

$$\mathbf{S}_{r+1} = \begin{pmatrix} \mathbf{S}_r & \underline{a} \\ \underline{a}^T & \underline{b} \end{pmatrix}$$

Für $\mathbf{\Lambda}_{r+1}$ verfolgen wir den Ansatz

$$\mathbf{\Lambda}_{r+1} = \begin{pmatrix} \mathbf{\Lambda}_r & \underline{0} \\ \underline{0}^T & 0 \end{pmatrix} + \underbrace{\begin{pmatrix} \underline{c} \\ \underline{d} \end{pmatrix}}_{\underline{k}_{r+1}} \cdot \underbrace{\begin{pmatrix} \underline{c} \\ \underline{d} \end{pmatrix}^T}_{\underline{k}_{r+1}^T} \quad (7.107)$$

wobei \underline{c} ein r -elementiger Spaltenvektor ist und $\underline{0}$ ein r -elementiger Nullvektor. Somit suchen wir (berechenbare) Ausdrücke für \underline{c} und \underline{d} .

Man beachte, dass d nicht Null sein kann. Da $\underline{k}_{r+1} \cdot \underline{k}_{r+1}^T = -\underline{k}_{r+1} \cdot (-\underline{k}_{r+1}^T)$ können wir o.B.d.A festlegen, dass $d < 0$ ist. Es muss nun gelten

$$\begin{aligned} \mathbf{E}_{r+1} &= \mathbf{S}_{r+1} \cdot \mathbf{\Lambda}_{r+1} = \begin{pmatrix} \mathbf{S}_r & \underline{a} \\ \underline{a}^T & b \end{pmatrix} \cdot \left[\begin{pmatrix} \mathbf{\Lambda}_r & \underline{0} \\ \underline{0}^T & 0 \end{pmatrix} + \begin{pmatrix} \underline{c}\underline{c}^T & \underline{c}d \\ \underline{c}^T d & dd \end{pmatrix} \right] \\ &= \begin{pmatrix} \mathbf{E}_r & \underline{0} \\ \underline{a}^T \mathbf{\Lambda}_r & 0 \end{pmatrix} + \begin{pmatrix} \mathbf{S}_r \underline{c}\underline{c}^T + \underline{a}\underline{c}^T d & \mathbf{S}_r \underline{c}d + \underline{a}dd \\ \underline{a}^T \underline{c}\underline{c}^T + b\underline{c}^T d & \underline{a}^T \underline{c}d + bdd \end{pmatrix} \end{aligned}$$

Dabei steht \mathbf{E} jeweils für die Einheitsmatrix. Nun betrachten wir die einzelnen Komponenten dieser Matrixgleichung und erhalten die vier Einzelgleichungen

$$\begin{aligned} \text{I} &: \underline{0} = \mathbf{S}_r \underline{c}\underline{c}^T + \underline{a}\underline{c}^T d \\ \text{II} &: \underline{0} = \mathbf{S}_r \underline{c}d + \underline{a}dd \\ \text{III} &: \underline{0}^T = \underline{a}^T \mathbf{\Lambda}_r + \underline{a}^T \underline{c}\underline{c}^T + b\underline{c}^T d \\ \text{IV} &: 1 = \underline{a}^T \underline{c}d + bdd \end{aligned}$$

Gesucht sind \underline{c} und d . Der Rest ist bekannt. Das Gleichungssystem ist somit überbestimmt. Erweitert man Gleichung II mit \underline{c}^T/d von rechts, so erhält man Gleichung I. Gleichung I muss nicht weiter betrachtet werden, da sie automatisch erfüllt ist, sobald Gleichung II erfüllt ist. Wir formen Gleichung II weiter um und erhalten

$$\begin{aligned} \text{II} &\Leftrightarrow \underline{0} = \mathbf{S}_r \underline{c} + \underline{a}d \\ &\Leftrightarrow \underline{c} = -d \mathbf{\Lambda}_r \underline{a} \quad (\text{IIa}) \end{aligned}$$

Wenn $\text{IV} \cdot \underline{c}^T$ bedeutet dass beide Seiten von Gleichung IV von rechts mit \underline{c}^T multipliziert werden und IIa^T bedeutet, dass beide Seiten von Gleichung IIa transponiert werden, dann erkennt man, dass

$$(\text{IV} \cdot \underline{c}^T - \text{IIa}^T) / d = \text{III}$$

Das bedeutet Gleichung III muss nicht mehr betrachtet werden, weil sie automatisch erfüllt ist, sobald die Gleichungen II und IV erfüllt sind. Nun setzen wir Gleichung IIa in IV ein und erhalten

$$\begin{aligned} \text{IIa in IV} &: 1 = -\underline{a}^T d \mathbf{\Lambda}_r \underline{a} d + bdd \\ &\Leftrightarrow d^2 = \frac{1}{b - \underline{a}^T \mathbf{\Lambda}_r \underline{a}} \\ &\Leftrightarrow d = \frac{-1}{\sqrt{b - \underline{a}^T \mathbf{\Lambda}_r \underline{a}}} \quad (\text{V}) \end{aligned}$$

weil wir uns bereits auf $d < 0$ festgelegt haben. Somit können wir V berechnen und danach IIa um d und \underline{c} zu bestimmen. Zusammengefasst erhalten wir

$$\underline{k}_{r+1} = \begin{pmatrix} \underline{c} \\ d \end{pmatrix} = \frac{1}{\sqrt{b - \underline{a}^T \mathbf{\Lambda}_r \underline{a}}} \cdot \begin{pmatrix} \mathbf{\Lambda}_r \underline{a} \\ -1 \end{pmatrix} \quad (7.108)$$

Es soll an dieser Stelle auch erwähnt werden, was passiert wenn die Matrix \mathbf{V}_{r+1} nicht vollen Rang hat. Wir betrachten dazu den Term

$$\begin{aligned} b - \underline{a}^T \mathbf{\Lambda}_r \underline{a} &= \underline{v}_{r+1}^T \underline{v}_{r+1} - \underline{v}_{r+1}^T \mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \underline{v}_{r+1} \\ &= \|\underline{v}_{r+1} - \mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \underline{v}_{r+1}\|_2^2 \geq 0 \end{aligned}$$

Anschaulich gesprochen, ziehen wir die Projektion von \underline{v}_{r+1} auf den durch \mathbf{V}_r aufgespannten Unterraum des \mathbb{R}^n von \underline{v}_{r+1} selbst ab und bilden dann die Norm. Diese Norm ist genau dann Null, wenn \underline{v}_{r+1} selbst in dem durch \mathbf{V}_r aufgespannten Unterraum liegt, also genau dann wenn \mathbf{V}_{r+1} singulär ist. In diesem Fall würde bei der Berechnung von \underline{k}_{r+1} eine Division durch Null auftreten. Wie wir sehen, ist bereits jetzt die Anzahl der benötigten Rechenoperationen nur noch von der Ordnung $O(r^2)$.

Wir fassen die Ergebnisse dieses Abschnittes wie folgt zusammen:

Algorithmus 7.2

Sei $r + 1 < n$ und sowohl $\mathbf{V}_r \in \mathbb{R}^{n \times r}$ mit $\mathbf{V}_r = (\underline{v}_1, \dots, \underline{v}_r)$ als auch $\mathbf{V}_{r+1} \in \mathbb{R}^{n \times r+1}$ mit $\mathbf{V}_{r+1} = (\mathbf{V}_r, \underline{v}_{r+1}) = (\underline{v}_1, \dots, \underline{v}_{r+1})$ haben vollen Rang. Sei ferner

$$\begin{aligned} \mathbf{\Lambda}_r &= (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \quad \text{gegeben und} \\ \mathbf{\Lambda}_{r+1} &= (\mathbf{V}_{r+1}^T \mathbf{V}_{r+1})^{-1} \quad \text{gesucht.} \end{aligned}$$

Dann brauchen wir keine Matrixinversion durchzuführen sondern berechnen

1. $\underline{a}_r = \mathbf{V}_r^T \underline{v}_{r+1}$ und $b = \underline{v}_{r+1}^T \underline{v}_{r+1}$
2. $\underline{k}_{r+1} = \frac{1}{\sqrt{b - \underline{a}_r^T \mathbf{\Lambda}_r \underline{a}_r}} \cdot \begin{pmatrix} \mathbf{\Lambda}_r \underline{a}_r \\ -1 \end{pmatrix}$
3. $\mathbf{\Lambda}_{r+1} = \begin{pmatrix} \mathbf{\Lambda}_r & \underline{0} \\ \underline{0}^T & 0 \end{pmatrix} + \underline{k}_{r+1} \underline{k}_{r+1}^T$

Sollte \mathbf{V}_{r+1} einmal nicht vollen Rang haben, dann (und nur dann) ist $b - \underline{a}_r^T \mathbf{\Lambda}_r \underline{a}_r = 0$ und $\mathbf{\Lambda}_{r+1}$ existiert nicht.

7.4 Das Prinzip der sukzessiven Regressorauswahl

Ausgangspunkt der anstehenden Betrachtung ist eine Wavelet-Bibliothek wie unter Abschnitt 3.5 beschrieben, also:

$$W_{wav} = \{\psi_1, \dots, \psi_L\}$$

Im Idealfall würde man die Potenzmenge $\mathcal{P}(W_{wav})$ bilden. Jede r -elementige Teilmenge von W_{wav} kann durch eine Indexmenge $I_r \subset \{1, \dots, L\}$ mit $|I_r| = r \leq L$ beschrieben werden. Solange dabei $r < N$ ist, also nicht mehr Wavelets in der Teilmenge sind als es Trainingsdatenpunkte gibt, lässt sich jeder solchen Teilmenge eine Hypothese H_{I_r} gemäß Definition 6.1 zuordnen. Zu jeder Hypothese könnten wir weiterhin das Kriterium K gemäß (6.84) berechnen und letztlich die Hypothese mit dem geringsten Kriterium auswählen. Fertig wäre das optimale Wavelet-Netzwerk. Leider ist dieses Vorgehen für größere Werte von N und L nicht praktikabel, weil die Anzahl der auszuwertenden Hypothesen und somit der Rechenaufwand exponentiell mit N , bzw. L wächst.

Fall	Anzahl der Hypothesen H_{I_r}
$L \leq N$	$= 2^L$
$L > N$	$> 2^N$

Ein einfacher, pragmatischer Ansatz, dem Problem des ausufernden Rechenaufwands zu begegnen, ist das Verfahren der sukzessiven Regressorauswahl. Dabei werden die Wavelets eines nach dem anderen hinzugewählt. Auf jeder Stufe des Verfahrens geht man davon aus, dass bereits r Wavelets ausgewählt sind, die als gesetzt betrachtet werden können. Welche Wavelets der Bibliothek W_{wav} ausgewählt wurden, lässt sich am besten mit einer injektiven Funktion

$$\pi : \{1, \dots, r\} \rightarrow \{1, \dots, L\}$$

festhalten. Dabei wurde das Wavelet $\psi_{\pi(1)}$ als erstes und das Wavelet $\psi_{\pi(r)}$ als letztes ausgewählt. Nun wird unter den noch nicht ausgewählten Wavelets jenes dazu genommen, welches zusammen mit den schon gesetzten Wavelets die beste Hypothese $H_{I_{r+1}}$ abgibt. Wir erweitern also den Definitionsbereich von π auf $\{1, \dots, r+1\}$ und legen $\pi(r+1)$ fest. Somit sind wir von Stufe r zu Stufe $r+1$ gelangt. Wenn keine Verbesserung der bestehenden Hypothese mehr erzielt werden kann, wird abgebrochen. Zur Veranschaulichung hier ein Beispiel

Beispiel 7.3

Wir betrachten den vereinfachten Fall einer Wavelet-Bibliothek mit vier Wavelets, also

$$W_{wav} = \{\psi_1, \psi_2, \psi_3, \psi_4\}$$

1. Stufe: Vergleich der Hypothesen

$$H_{\{1\}} \Rightarrow K = 5.6$$

$$H_{\{2\}} \Rightarrow K = 3.2$$

$$H_{\{3\}} \Rightarrow K = 4.6$$

$$H_{\{4\}} \Rightarrow K = 4.3$$

Auswahl der besten Hypothese: $H_{\{2\}}$, also $\pi(1) := 2$

2. Stufe: Vergleich der Hypothesen

$$H_{\{2,1\}} \Rightarrow K = 3.5$$

$$H_{\{2,3\}} \Rightarrow K = 3.0$$

$$H_{\{2,4\}} \Rightarrow K = 1.7$$

Auswahl der besten Hypothese: $H_{\{2,4\}}$, also $\pi(2) := 4$

3. Stufe: Vergleich der Hypothesen

$$H_{\{2,4,1\}} \Rightarrow K = 1.8$$

$$H_{\{2,4,3\}} \Rightarrow K = 2.3$$

Keine Verbesserung gegenüber $H_{\{2,4\}}$ möglich.

Ergebnis: $H_{\{2,4\}}$, d.h. die Wavelets ψ_2 und ψ_4 wurden ausgewählt. $\pi(1) = 2$ und $\pi(2) = 4$.

Für wie viele verschiedene Hypothesen H_{L_r} die Kriteriumsfunktion K letztendlich berechnet werden muss, steht nicht a priori fest, sondern hängt davon ab, wie viele Wavelets gewählt werden. Bezeichnen wir jede Berechnung von K als einen Schritt, dann sind auf einer Stufe r demzufolge $L - r$ Schritte nötig, bevor man zur nächsten Stufe kommt. Werden am Ende \tilde{r} von L Wavelets ausgewählt, dann ist die folgende Anzahl von Schritten, bzw. Auswertungen nötig:

$$\sum_{r=0}^{\tilde{r}} L - r$$

Das sind größenordnungsmäßig $L \cdot \tilde{r}$ Auswertungen. Natürlich ist nicht zu erwarten, dass die Methode der sukzessiven Regressorauswahl die global bestbewertete Hypothese hervorbringt, wie es bei einer Auswertung aller denkbaren Hypothesen der Fall wäre. Sie führt aber im Allgemeinen zu hinreichend guten Lösungen unter vertretbarem Aufwand.

7.5 Sukzessive Regressorauswahl mit Konditionsbetrachtung

Dieser Abschnitt baut direkt auf das statistische Kriterium für die Güte von Wavelet-Netzwerken auf, wie es in Kapitel 6 eingeführt wurde. Ziel ist es, das eben beschriebene Prinzip der sukzessiven Regressorauswahl mit dem Güte-Kriterium K aus (6.84) effizient umzusetzen. Ausgangspunkt ist eine Wavelet-Bibliothek W_{wav} , wie sie vorzugsweise durch Algorithmus 7.1 erstellt wurde. Nun betrachten wir die Formel

$$K_r = \sigma_r^2 \left(1 + \frac{1}{\tau} \text{Spur} [\mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1}] \right) \quad (7.109)$$

Dabei ist hier

$$\mathbf{V}_r = (\underline{v}_1, \dots, \underline{v}_r) \quad \text{mit} \quad \underline{v}_i = \begin{bmatrix} \psi_{\pi(i)}(x_1) \\ \vdots \\ \psi_{\pi(i)}(x_n) \end{bmatrix}, \quad i = 1 \dots r \quad (7.110)$$

$$\text{und} \quad \mathbf{P}_r = \begin{pmatrix} 1 & \langle \psi_{\pi(1)}, \psi_{\pi(2)} \rangle & \cdots & \langle \psi_{\pi(1)}, \psi_{\pi(r)} \rangle \\ \langle \psi_{\pi(2)}, \psi_{\pi(1)} \rangle & 1 & \cdots & \langle \psi_{\pi(2)}, \psi_{\pi(r)} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \psi_{\pi(r)}, \psi_{\pi(1)} \rangle & \langle \psi_{\pi(r)}, \psi_{\pi(2)} \rangle & \cdots & 1 \end{pmatrix} \quad (7.111)$$

Dabei bezeichnet $\langle \psi_i, \psi_j \rangle$ das Skalarprodukt p_{ij} zweier Wavelets, wie in Abschnitt 7.2 berechnet. Die Diagonalelemente sind eins, weil die Wavelets normiert sind. Für die Stichprobenvarianz σ_r^2 gilt

$$\sigma_r^2 = \frac{\|\underline{\gamma}_r\|_2^2}{N - r}$$

mit $\underline{\gamma}_r = \underline{y} - \mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \underline{y}$

Dabei ist $\underline{\gamma}_r$ der Residuenvektor, der sich als Lösung des Linearen Ausgleichproblems ergibt.

7.5.1 Rahmenalgorithmus

Unser Algorithmus zur sukzessiven Regressorauswahl soll den Folgenden Rahmen haben

```

Algorithmus 7.4 Rahmenalgorithmus Regressorauswahl
 $K_{\text{aussen}} := \infty;$ 
 $K_{\text{innen}} := \infty;$ 
 $r := 0;$  – Abgeschlossene Stufe
– äußere Schleife über die Stufen  $r$ :
loop
– innere Schleife über die noch nicht gewählten Wavelets
for  $i = 1 \dots L$   $i \neq \pi(1), \dots, i \neq \pi(r)$  loop
 $K_{\text{test}} := \text{Kriteriumsfunktion}[\pi(1), \dots, \pi(r), i];$  – gemäß (7.109)
if  $K_{\text{test}} < K_{\text{innen}}$  then
 $\pi(r+1) := i;$ 
 $K_{\text{innen}} := K_{\text{test}};$ 
end if;
end loop;
if  $K_{\text{innen}} < K_{\text{aussen}}$  then
 $r := r + 1;$ 
 $K_{\text{aussen}} := K_{\text{innen}};$ 
else
Algorithmus_zu_Ende();
end if;
end loop;

```

Grundsätzlich ist es kein Problem, zu einem gegebenen Trainingsdatensatz \mathbb{O}_1^N und einer ausgewählten Menge von r Wavelets, indiziert durch $I_r \subset \{1, \dots, L\}$ mit $|I_r| = r \leq L$, einen Ausdruck der Form (7.109) auszuwerten. Allerdings ist der Rechenaufwand zur Auswertung von K_r erheblich, besonders bei größeren Werten von r . Hierbei schlägt vor allem die Inversion der $r \times r$ -Matrix $\mathbf{V}_r^T \mathbf{V}_r$ zu Buche. Dieser Rechenaufwand lässt sich jedoch erheblich reduzieren, wenn man im Rahmen der sukzessiven Regressorauswahl einige Größen geschickt wiederverwendet, die bereits zuvor berechnet wurden. Wie genau das vonstatten geht werden wir nun ausführen.

7.5.2 Effiziente Berechnung der Kriteriumsfunktion

Wir gehen nun davon aus, dass wir im Zuge der sukzessiven Regressorauswahl bereits die Stufe r abgeschlossen haben, d.h. wir haben bereits die r Wavelets $\psi_{\pi(1)}, \dots, \psi_{\pi(r)}$ in genau dieser Reihenfolge ausgewählt. Damit ist bereits eine Matrix \mathbf{V}_r mit vollem Rang gemäß (7.110) vorhanden, sowie eine Matrix \mathbf{P}_r gemäß (7.111). Weiterhin gehen wir davon aus, dass eine Matrix $\mathbf{\Lambda}_r$ bereits berechnet wurde und vorliegt, für die gilt

$$\mathbf{\Lambda}_r = (\mathbf{V}_r^T \mathbf{V}_r)^{-1}$$

Diese Matrix ist symmetrisch und positiv definit. Im eben beschriebenen Rahmenalgorithmus werden in der inneren Schleife alle noch nicht ausgewählten

Wavelets i durchlaufen. Dabei wird in jedem Durchlauf eine Kriteriumsfunktion \dot{K}_{r+1} der Form

$$\dot{K}_{r+1} = \dot{\sigma}_{r+1}^2 \left(1 + \frac{1}{\tau} \text{Spur} \left[\dot{\mathbf{P}}_{r+1} (\dot{\mathbf{V}}_{r+1}^T \dot{\mathbf{V}}_{r+1})^{-1} \right] \right)$$

berechnet. Die Punkte über den jeweiligen Größen bedeuten, dass diese von dem jeweiligen i in der inneren Schleife abhängen. Wir wählen diese Schreibweise um eine Doppelindizierung zu vermeiden und die Lesbarkeit zu erleichtern. Wir schreiben also

$$\dot{\psi} = \psi_i$$

wenn wir nun einen Vektor $\dot{\underline{p}}_r$ einführen mit

$$\dot{\underline{p}}_r = \begin{bmatrix} \langle \dot{\psi}, \psi_{\pi(1)} \rangle \\ \vdots \\ \langle \dot{\psi}, \psi_{\pi(r)} \rangle \end{bmatrix}$$

,dann erhalten wir für jedes Wavelet i in der inneren Schleife des Rahmenalgorithmus eine Matrix $\dot{\mathbf{P}}_{r+1}$ mit

$$\dot{\mathbf{P}}_{r+1} = \begin{bmatrix} \mathbf{P}_r & \dot{\underline{p}}_r \\ \dot{\underline{p}}_r^T & 1 \end{bmatrix}$$

Dabei liegt die Matrix \mathbf{P}_r schon berechnet vor und hat eine Form wie in (7.111). Auch bei der Berechnung von $\dot{\underline{p}}_r$ machen wir uns zu Nutze, dass bereits eine Stufe vorher, beim gleichen i in der inneren Schleife, ein $\dot{\underline{p}}_{r-1}$ berechnet wurde. Damit ist

$$\dot{\underline{p}}_r = \begin{bmatrix} \dot{\underline{p}}_{r-1} \\ \langle \dot{\psi}, \psi_{\pi(r)} \rangle \end{bmatrix}$$

und das Skalarprodukt zweier Wavelets muss in jedem Durchlauf der inneren Schleife effektiv nur einmal ausgewertet werden. Auf die Berechnung dieses Skalarproduktes und den damit verbundenen Rechenaufwand wurde in Abschnitt 7.2 eingegangen. Natürlich müssen dafür alle bereits erfolgten Auswertungen gespeichert werden. Wenden wir uns nun der notwendigen Matrixinversion in der inneren Schleife zu. Zunächst führen wir den Vektor $\dot{\underline{v}}$ und die Matrix $\dot{\mathbf{V}}_{r+1}$ ein mit

$$\dot{\underline{v}} = \begin{bmatrix} \dot{\psi}(x_1) \\ \vdots \\ \dot{\psi}(x_N) \end{bmatrix} \quad \text{und} \quad \dot{\mathbf{V}}_{r+1} = (\mathbf{V}_r, \dot{\underline{v}})$$

Dabei müssen die Vektoren $\dot{\underline{v}}$ für jedes Wavelet nur ein mal vor Beginn des Algorithmus berechnet werden und liegen somit für den wiederholten Gebrauch vor. Nun wollen wir die Matrix

$$\dot{\mathbf{\Lambda}}_{r+1} = (\dot{\mathbf{V}}_{r+1}^T \dot{\mathbf{V}}_{r+1})^{-1}$$

berechnen. Dazu greifen wir auf die Ergebnisse aus dem Abschnitt 7.3 über effiziente Matrixinversion zurück. Wir führen ein:

$$1.) \quad \underline{\dot{a}}_r = \mathbf{V}_r^T \underline{\dot{v}} = \begin{pmatrix} \underline{\dot{a}}_{r-1} \\ \underline{v}_r^T \underline{\dot{v}} \end{pmatrix} \quad \text{und} \quad \underline{\dot{b}} = \underline{\dot{v}}^T \underline{\dot{v}}$$

wobei nur $\underline{v}_r^T \underline{\dot{v}}$ in jedem Durchlauf der inneren Schleife berechnet werden muss. Alle anderen Größen können aus der letzten Stufe übernommen werden. Nun benötigen wir

$$2.) \quad \underline{\dot{k}}_{r+1} = \frac{1}{\sqrt{\underline{\dot{b}} - \underline{\dot{a}}_r^T \mathbf{\Lambda}_r \underline{\dot{a}}_r}} \cdot \begin{pmatrix} \mathbf{\Lambda}_r \underline{\dot{a}}_r \\ -1 \end{pmatrix}$$

Dabei können wir den Ausdruck $\mathbf{\Lambda}_r \underline{\dot{a}}_r$ noch geschickter berechnen, wenn wir uns auf der letzten Stufe beim gleichen Wavelet in der inneren Schleife den Term $\mathbf{\Lambda}_{r-1} \underline{\dot{a}}_{r-1}$ gespeichert haben und ebenso den Vektor \underline{k}_r am Ende der letzten Stufe. Es gilt nämlich

$$\mathbf{\Lambda}_r \underline{\dot{a}}_r = \left[\begin{pmatrix} \mathbf{\Lambda}_{r-1} & \underline{0} \\ \underline{0}^T & 0 \end{pmatrix} + \underline{k}_r \underline{k}_r^T \right] \cdot \begin{pmatrix} \underline{\dot{a}}_{r-1} \\ \underline{v}_r^T \underline{\dot{v}} \end{pmatrix} = \begin{pmatrix} \mathbf{\Lambda}_{r-1} \underline{\dot{a}}_{r-1} \\ 0 \end{pmatrix} + \underline{k}_r \underline{k}_r^T \underline{\dot{a}}_r$$

Dabei haben wir (7.107) verwendet. Auf diese Weise sparen wir sehr viel Rechenaufwand in der inneren Schleife des Rahmenalgorithmus. Der notwendige Aufwand zur Berechnung eines $\underline{\dot{k}}_{r+1}$ ist somit nur linear von r abhängig und nicht quadratisch. Nun könnten wir also

$$3.) \quad \dot{\mathbf{\Lambda}}_{r+1} = \begin{pmatrix} \mathbf{\Lambda}_r & \underline{0} \\ \underline{0}^T & 0 \end{pmatrix} + \underline{\dot{k}}_{r+1} \underline{\dot{k}}_{r+1}^T$$

berechnen, was wir aber de facto in der inneren Schleife nicht tun müssen, wie wir gleich sehen. Dazu betrachten wir nun den Term

$$\begin{aligned} & \text{Spur} \left[\dot{\mathbf{P}}_{r+1} (\dot{\mathbf{V}}_{r+1}^T \dot{\mathbf{V}}_{r+1})^{-1} \right] \\ &= \text{Spur}(\dot{\mathbf{P}}_{r+1} \dot{\mathbf{\Lambda}}_{r+1}) \\ &= \text{Spur} \left[\begin{pmatrix} \mathbf{P}_r & \underline{\dot{p}}_r \\ \underline{\dot{p}}_r^T & 1 \end{pmatrix} \cdot \left\{ \begin{pmatrix} \mathbf{\Lambda}_r & \underline{0} \\ \underline{0}^T & 0 \end{pmatrix} + \underline{\dot{k}}_{r+1} \underline{\dot{k}}_{r+1}^T \right\} \right] \\ &= \text{Spur} \left[\begin{pmatrix} \mathbf{P}_r \mathbf{\Lambda}_r & \underline{0} \\ \underline{\dot{p}}_r^T \mathbf{\Lambda}_r & 0 \end{pmatrix} + \dot{\mathbf{P}}_{r+1} \underline{\dot{k}}_{r+1} \underline{\dot{k}}_{r+1}^T \right] \\ &= \text{Spur}(\mathbf{P}_r \mathbf{\Lambda}_r) + \text{Spur}(\dot{\mathbf{P}}_{r+1} \underline{\dot{k}}_{r+1} \underline{\dot{k}}_{r+1}^T) \\ &= \underbrace{\text{Spur}(\mathbf{P}_r \mathbf{\Lambda}_r)}_{\text{bekannt}} + \underline{\dot{k}}_{r+1}^T \dot{\mathbf{P}}_{r+1} \underline{\dot{k}}_{r+1} \end{aligned}$$

Das bedeutet, dass wir das Ergebnis von $\text{Spur}(\mathbf{P}_r \mathbf{\Lambda}_r)$ am Ende jeder Stufe speichern müssen. Die Berechnung von $\underline{\dot{k}}_{r+1}^T \dot{\mathbf{P}}_{r+1} \underline{\dot{k}}_{r+1}$ würde größenordnungsmäßig

$r^2/2$ Operationen erfordern, womit sie zum größten Zeitfaktor in der inneren Schleife des Algorithmus werden würde. Wir können den Rechenaufwand aber erheblich reduzieren. Dazu schlüsseln wir den Term noch weiter auf zu

$$\begin{aligned}\underline{\dot{k}}_{r+1}^T \dot{\mathbf{P}}_{r+1} \underline{\dot{k}}_{r+1} &= \frac{1}{\underline{\dot{b}} - \underline{\dot{a}}_r^T \underline{\Lambda}_r \underline{\dot{a}}_r} \cdot \left(\begin{array}{cc} \underline{\dot{a}}_r^T \underline{\Lambda}_r & -1 \end{array} \right) \left(\begin{array}{c} \underline{\mathbf{P}}_r \quad \underline{\dot{p}}_r \\ \underline{\dot{p}}_r^T \quad 1 \end{array} \right) \left(\begin{array}{c} \underline{\Lambda}_r \underline{\dot{a}}_r \\ -1 \end{array} \right) \\ &= \frac{1}{\underline{\dot{b}} - \underline{\dot{a}}_r^T \underline{\Lambda}_r \underline{\dot{a}}_r} \cdot \left[\underline{\dot{a}}_r^T \underline{\Lambda}_r \underline{\mathbf{P}}_r \underline{\Lambda}_r \underline{\dot{a}}_r - 2 \underline{\dot{a}}_r^T \underline{\Lambda}_r \underline{\dot{p}}_r + 1 \right]\end{aligned}$$

und darin noch weiter

$$\begin{aligned}\underline{\dot{a}}_r^T \underline{\Lambda}_r \underline{\mathbf{P}}_r \underline{\Lambda}_r \underline{\dot{a}}_r &= \underline{\dot{a}}_r^T \left[\left(\begin{array}{cc} \underline{\Lambda}_{r-1} & \underline{0} \\ \underline{0}^T & \underline{0} \end{array} \right) + \underline{k}_r \underline{k}_r^T \right] \underline{\mathbf{P}}_r \left[\left(\begin{array}{cc} \underline{\Lambda}_{r-1} & \underline{0} \\ \underline{0}^T & \underline{0} \end{array} \right) + \underline{k}_r \underline{k}_r^T \right] \underline{\dot{a}}_r \\ &= \underbrace{\underline{\dot{a}}_{r-1}^T \underline{\Lambda}_{r-1} \underline{\mathbf{P}}_{r-1} \underline{\Lambda}_{r-1} \underline{\dot{a}}_{r-1}}_{1^*} + 2 \underline{\dot{a}}_r^T \underline{k}_r \underbrace{\underline{k}_r^T \underline{\mathbf{P}}_r}_{2^*} \left(\begin{array}{c} \underline{\Lambda}_{r-1} \underline{\dot{a}}_{r-1} \\ \underline{0} \end{array} \right) \\ &\quad + \underbrace{\underline{\dot{a}}_r^T \underline{k}_r \underline{k}_r^T \underline{\mathbf{P}}_r \underline{k}_r \underline{k}_r^T \underline{\dot{a}}_r}_{3^*}\end{aligned}$$

Dabei wurde der Term 1^* zum jeweiligen Wavelet bereits in der letzten Stufe berechnet. Die Terme 2^* und 3^* mussten nur einmalig nach Abschluss der letzten Stufe berechnet werden, stehen also in der inneren Schleife bereits zur Verfügung. Somit sind in der inneren Schleife nur noch Additionen und Multiplikationen von Vektoren notwendig. Aufwendige Matrixoperationen können vermieden werden. Wichtig ist nun noch die Stichprobenvarianz $\dot{\sigma}_{r+1}^2$ in jedem Durchlauf der inneren Schleife zu bestimmen. Dabei ist

$$\dot{\sigma}_{r+1}^2 = \frac{\left\| \underline{\dot{\gamma}}_{r+1} \right\|_2^2}{N - (r + 1)}$$

wobei $\underline{\dot{\gamma}}_{r+1}$ der Residuenvektor ist, der sich folgendermaßen als Lösung eines Linearen Ausgleichproblems ergibt

$$\underline{\dot{\gamma}}_{r+1} = \underline{y} - \dot{\mathbf{V}}_{r+1} (\dot{\mathbf{V}}_{r+1}^T \dot{\mathbf{V}}_{r+1})^{-1} \dot{\mathbf{V}}_{r+1}^T \underline{y}$$

Damit ergibt sich

$$\begin{aligned}\left\| \underline{\dot{\gamma}}_{r+1} \right\|_2^2 &= \underline{y}^T \underline{y} - \underline{y}^T \dot{\mathbf{V}}_{r+1} \underline{\Lambda}_{r+1} \dot{\mathbf{V}}_{r+1}^T \underline{y} \\ &= \underline{y}^T \underline{y} - \underline{y}^T (\underline{\mathbf{V}}_r, \underline{\dot{v}}) \left[\left(\begin{array}{cc} \underline{\Lambda}_r & \underline{0} \\ \underline{0}^T & \underline{0} \end{array} \right) + \underline{k}_{r+1} \underline{k}_{r+1}^T \right] \left(\begin{array}{c} \underline{\mathbf{V}}_r^T \\ \underline{\dot{v}}^T \end{array} \right) \underline{y} \\ &= \underline{y}^T \underline{y} - \underline{y}^T \underline{\mathbf{V}}_r \underline{\Lambda}_r \underline{\mathbf{V}}_r^T \underline{y} - \underline{y}^T \dot{\mathbf{V}}_{r+1} \underline{k}_{r+1} \underline{k}_{r+1}^T \dot{\mathbf{V}}_{r+1}^T \underline{y} \\ &= \left\| \underline{\gamma}_r \right\|_2^2 - (\underline{k}_{r+1}^T \dot{\mathbf{V}}_{r+1}^T \underline{y})^2\end{aligned}$$

Dabei ist $\left\| \underline{\gamma}_r \right\|_2^2$ aus der letzten Stufe bekannt und

$$\dot{\mathbf{V}}_{r+1}^T \underline{y} = \left(\begin{array}{c} \underline{\mathbf{V}}_r^T \underline{y} \\ \underline{\dot{v}}^T \underline{y} \end{array} \right)$$

wobei $\mathbf{V}_r \underline{y}$ bereits aus der letzten Stufe bekannt ist und $\underline{\dot{v}}^T \underline{y}$ für jedes Wavelet nur einmal vor dem Algorithmus berechnet werden muss. Damit ist auch der Rechenaufwand zur Ermittlung von $\left\| \underline{\dot{\gamma}}_{r+1} \right\|_2^2$ sehr gering. Haben wir nun all diese Teilterme berechnet, fügen wir die Komponenten nur noch zusammen und erhalten in jedem Durchlauf der inneren Schleife ein Kriterium

$$\dot{K}_{r+1} = \dot{\sigma}_{r+1}^2 \left(1 + \frac{1}{\tau} \text{Spur} \left[\dot{\mathbf{P}}_{r+1} (\dot{\mathbf{V}}_{r+1}^T \dot{\mathbf{V}}_{r+1})^{-1} \right] \right)$$

7.5.3 Algorithmische Aufbereitung

Nun wollen wir die eben gemachten Überlegungen noch in eine Form umschreiben, die eine direkte algorithmische Umsetzung erlaubt. Grundlage ist der Rahmenalgorithmus aus Abschnitt 7.5.1. Dabei fügen wir an drei Stellen Berechnungen ein und zwar

1. Berechnungen im Vorfeld des Algorithmus
2. Berechnungen für alle Wavelets in der inneren Schleife
3. Berechnungen nach Abschluss der inneren Schleife

Berechnungen im Vorfeld des Algorithmus

Die folgenden Berechnungen sollten im Vorfeld des Algorithmus gemacht werden, damit im Algorithmus selbst auf die bereits berechneten Größen zurückgegriffen werden kann. Als erstes berechnen wir einmalig

$$\left\| \underline{\gamma}_0 \right\|_2^2 := \underline{y}^T \underline{y}$$

Danach berechnen wir für jedes Wavelet $\dot{\psi} = \psi_1, \dots, \psi_L$ die Ausdrücke

$$\underline{\dot{v}} := \begin{bmatrix} \dot{\psi}(x_1) \\ \vdots \\ \dot{\psi}(x_N) \end{bmatrix} \quad \text{und damit} \quad \underline{\dot{v}}^T \underline{y} \quad \text{sowie} \quad \dot{b} := \underline{\dot{v}}^T \underline{\dot{v}}$$

Berechnungen für alle Wavelets in der inneren Schleife

Die folgenden Berechnungen können erst ab $r = 1$ durchgeführt werden. Die Auswahl des allerersten Wavelets muss gesondert betrachtet werden. In der inneren Schleife, in der wir über alle noch nicht ausgewählten Wavelets $\dot{\psi} = \psi_i$ laufen, berechnen wir

$$\dot{p}_r := \left[\left\langle \dot{\psi}, \dot{p}_{r-1} \right\rangle \right]$$

Wir berechnen weiterhin

$$\underline{\dot{a}}_r := \begin{pmatrix} \underline{\dot{a}}_{r-1} \\ \underline{v}_r^T \underline{\dot{v}} \end{pmatrix}$$

und

$$\mathbf{\Lambda}_r \underline{\dot{a}}_r := \begin{pmatrix} \mathbf{\Lambda}_{r-1} \underline{\dot{a}}_{r-1} \\ 0 \end{pmatrix} + \underline{k}_r \underline{k}_r^T \underline{\dot{a}}_r$$

Damit berechnen wir weiter

$$\begin{aligned} \underline{\dot{a}}_r^T \mathbf{\Lambda}_r \mathbf{P}_r \mathbf{\Lambda}_r \underline{\dot{a}}_r &= \underline{\dot{a}}_{r-1}^T \mathbf{\Lambda}_{r-1} \mathbf{P}_{r-1} \mathbf{\Lambda}_{r-1} \underline{\dot{a}}_{r-1} + 2 \underline{\dot{a}}_r^T \underline{k}_r \underline{k}_r^T \mathbf{P}_r \begin{pmatrix} \mathbf{\Lambda}_{r-1} \underline{\dot{a}}_{r-1} \\ 0 \end{pmatrix} \\ &+ \underline{\dot{a}}_r^T \underline{k}_r \underline{k}_r^T \mathbf{P}_r \underline{k}_r \underline{k}_r^T \underline{\dot{a}}_r \end{aligned}$$

und damit weiter

$$\text{Spur}(\dot{\mathbf{P}}_{r+1} \dot{\mathbf{\Lambda}}_{r+1}) := \underbrace{\text{Spur}(\mathbf{P}_r \mathbf{\Lambda}_r)}_{\text{bekannt}} + \frac{1}{\dot{b} - \underline{\dot{a}}_r^T \mathbf{\Lambda}_r \underline{\dot{a}}_r} \cdot \left[\underline{\dot{a}}_r^T \mathbf{\Lambda}_r \mathbf{P}_r \mathbf{\Lambda}_r \underline{\dot{a}}_r - 2 \underline{\dot{a}}_r^T \mathbf{\Lambda}_r \underline{\dot{p}}_r + 1 \right]$$

Es folgt die Berechnung von

$$\underline{\dot{k}}_{r+1} := \frac{1}{\sqrt{\dot{b} - \underline{\dot{a}}_r^T \mathbf{\Lambda}_r \underline{\dot{a}}_r}} \cdot \begin{pmatrix} \mathbf{\Lambda}_r \underline{\dot{a}}_r \\ -1 \end{pmatrix}$$

Beim Ausprogrammieren des Algorithmus reicht es zunächst aus, in der inneren Schleife nur den Vorfaktor zu berechnen, da der Vektor $\mathbf{\Lambda}_r \underline{\dot{a}}_r$ bereits berechnet vorliegt. Nun benötigen wir noch

$$\left\| \underline{\dot{\gamma}}_{r+1} \right\|_2^2 := \left\| \underline{\dot{\gamma}}_r \right\|_2^2 - \left(\underline{\dot{k}}_{r+1}^T \begin{bmatrix} \mathbf{V}_r^T \underline{y} \\ \underline{\dot{v}}_r^T \underline{y} \end{bmatrix} \right)^2$$

und können damit schließlich

$$\dot{K}_{r+1} := \frac{\left\| \underline{\dot{\gamma}}_{r+1} \right\|_2^2}{N - (r + 1)} \left(1 + \frac{1}{\tau} \text{Spur} \left[\dot{\mathbf{P}}_{r+1} \dot{\mathbf{\Lambda}}_{r+1} \right] \right)$$

berechnen. Von den Ergebnissen der eben dargestellten acht Berechnungen müssen die ersten vier, also $\underline{\dot{p}}_r$, $\underline{\dot{a}}_r$, $\mathbf{\Lambda}_r \underline{\dot{a}}_r$ und $\underline{\dot{a}}_r^T \mathbf{\Lambda}_r \mathbf{P}_r \mathbf{\Lambda}_r \underline{\dot{a}}_r$ zu jedem Wavelet fest gespeichert werden, weil sie auf der nächsten Stufe bei eben diesem Wavelet wieder benötigt werden. Die anderen vier Größen, also $\underline{\dot{k}}_{r+1}$, $\text{Spur}(\dot{\mathbf{P}}_{r+1} \dot{\mathbf{\Lambda}}_{r+1})$, $\left\| \underline{\dot{\gamma}}_{r+1} \right\|_2^2$ und \dot{K}_{r+1} können verworfen werden, falls es bereits eine Hypothese mit kleinerem Kriterium K gab. Nur wenn die gerade aktuelle Hypothese die bislang beste ist, also wenn \dot{K}_{r+1} das bislang kleinste gefundene Kriterium ist, müssen diese vier Größen solange gespeichert bleiben bis, eine noch bessere Hypothese gefunden wird. Für den allerersten Durchlauf der inneren Schleife, also den Fall $r = 0$ werden die ersten drei Größen einfach noch nicht berechnet und bleiben somit Vektoren der Dimension Null. Die restlichen Berechnungen bleiben sinngemäß.

Berechnungen nach Abschluss der inneren Schleife

Grundlage ist wieder unser Rahmenalgorithmus. Direkt nach dem Abschluss der inneren Schleife müssen zwei Fälle unterschieden werden. Entweder es wurde kein Wavelet gefunden für welches $K_{r+1} < K_r$ war. In diesem Fall wird der Algorithmus beendet. Falls jedoch ein solches Wavelet $\dot{\psi} = \psi_{\pi(r+1)}$ gefunden wurde, müssen noch einige Größen berechnet, bzw. übernommen werden bevor es zur nächsten Stufe geht. Dabei beziehen sich im Folgenden alle Größen mit Punkt auf die Größen dieses zuletzt ausgewählten Wavelets.

$$\begin{aligned}\underline{v}_{r+1} &:= \dot{v} \\ \mathbf{V}_{r+1}^T \underline{y} &:= \begin{bmatrix} \mathbf{V}_r^T \underline{y} \\ \dot{v}^T \underline{y} \end{bmatrix} \\ \mathbf{P}_{r+1} &:= \begin{bmatrix} \mathbf{P}_r & \dot{p}_r \\ \dot{p}_r^T & 1 \end{bmatrix} \\ \underline{k}_{r+1} &:= \dot{k}_{r+1} \\ \mathbf{\Lambda}_{r+1} &:= \begin{pmatrix} \mathbf{\Lambda}_r & 0 \\ 0^T & 0 \end{pmatrix} + \underline{k}_{r+1} \underline{k}_{r+1}^T \\ \text{Spur}(\mathbf{P}_{r+1} \mathbf{\Lambda}_{r+1}) &:= \text{Spur}(\mathbf{P}_r \mathbf{\Lambda}_r) + \underline{k}_{r+1}^T \mathbf{P}_{r+1} \underline{k}_{r+1} \\ \|\underline{\gamma}_{r+1}\|_2^2 &:= \|\dot{\gamma}_{r+1}\|_2^2 \\ K_{r+1} &:= \dot{K}_{r+1}\end{aligned}$$

Somit haben wir alle Größen errechnet, die wir auf der nächsten Stufe in den Durchläufen der inneren Schleife benötigen. Auch die Zwischenergebnisse $\mathbf{P}_{r+1} \underline{k}_{r+1}$ und $\underline{k}_{r+1}^T \mathbf{P}_{r+1} \underline{k}_{r+1}$ sollten zu diesem Zweck explizit gespeichert werden.

7.6 Rückwärtselimination von Regressoren

Bisher haben wir einen Algorithmus beschrieben, der nacheinander Wavelets auswählt. Ein einmal ausgewähltes Wavelet war dabei für immer gesetzt. Nun kann es aber vorkommen, dass ein am Anfang gewähltes Wavelet überflüssig wird, weil später dazu gewählte Wavelets die Funktion schon besser beschreiben. Um bereits ausgewählte Wavelets auch wieder aussortieren zu können, führen wir nun einen Algorithmus zur Regressorelimination ein. Dieses Vorgehen ist in gewisser Weise komplementär zur sukzessiven Regressorauswahl. Die Voraussetzung dabei ist, dass wir bereits r Wavelets bzw. Regressoren aus der Wavelet-Bibliothek

$$W_{wav} = \{\psi_1, \dots, \psi_L\}$$

ausgewählt haben. Das heißt, wir haben bereits eine Ausgangshypothese H_{I_r} gemäß Definition 6.1, der auch schon ein Kriterium K_r zugeordnet ist. Ebenso gehen wir davon aus, dass die vorhandenen Wavelets in einer Reihenfolge

ausgewählt wurden, welche durch die injektive Funktion

$$\pi : \{1, \dots, r\} \rightarrow \{1, \dots, L\}$$

beschrieben wird. Auf jeder Stufe des Eliminations-Algorithmus versuchen wir nun ein Wavelet zu eliminieren und zwar so, dass das Kriterium K möglichst klein wird. So gelangen wir von Stufe r auf Stufe $r - 1$. Wurde zum Beispiel das Wavelet $\psi_{\pi(2)}$ eliminiert, so muss auch der entsprechende Eintrag aus der Funktion π gestrichen werden, und die anderen Einträge rücken nach, also

$$\pi(i) := \pi(i + 1) \forall i \in \{2, \dots, r - 1\}$$

Danach wird der Definitionsbereich von π auf $\{1, \dots, r - 1\}$ reduziert. Zur Veranschaulichung ziehen wir wiederum ein einfaches Beispiel heran

Beispiel 7.5

Wir betrachten den vereinfachten Fall einer Wavelet-Bibliothek mit vier Wavelets, also

$$W_{wav} = \{\psi_1, \psi_2, \psi_3, \psi_4\}$$

Daraus wurden die Wavelets ψ_2, ψ_3, ψ_4 in dieser Reihenfolge ausgewählt, also $\pi(1) = 2; \pi(2) = 3; \pi(3) = 4$ und wir haben die Ausgangshypothese

$$H_{\{2,3,4\}} \Rightarrow K = 2.3$$

3. Stufe: Vergleich der Hypothesen

$$H_{\{2,3\}} \Rightarrow K = 3.0$$

$$H_{\{2,4\}} \Rightarrow K = 1.7$$

$$H_{\{3,4\}} \Rightarrow K = 2.6$$

Auswahl der besten Hypothese: $H_{\{2,4\}}$, also $\pi(1) := 2$ und $\pi(2) := 4$

2. Stufe: Vergleich der Hypothesen

$$H_{\{2\}} \Rightarrow K = 3.2$$

$$H_{\{4\}} \Rightarrow K = 4.3$$

Keine Verbesserung gegenüber $H_{\{2,4\}}$ möglich.

Ergebnis: $H_{\{2,4\}}$, d.h. die Wavelets ψ_2 und ψ_4 wurden ausgewählt. $\pi(1) = 2$ und $\pi(2) = 4$.

Wenn wir die Rückwärtselimination von Regressoren verwenden, um das Ergebnis einer vorausgegangenen sukzessiven Regressorauswahl zu verbessern, dann ist der Rechenaufwand wenig kritisch.

7.7 Regressorelimination mit Konditionsbetrachtung

Damit dieser Algorithmus kompatibel mit dem in Unterabschnitt 7.5 beschriebenen Auswahlverfahren ist, muss die gleiche Kriteriumsfunktion zu Grunde gelegt werden, also

$$K_r = \sigma_r^2 \left(1 + \frac{1}{\tau} \text{Spur} [\mathbf{P}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1}] \right) \quad (7.112)$$

genau wie in (7.109).

7.7.1 Rahmenalgorithmus

Dabei soll unser Algorithmus zur Rückwärtselimination den folgenden Rahmen haben

Algorithmus 7.6 *Rahmenalgorithmus Regressorelimination*
 – r, K_r und π sind bereits gegeben
 $K_{\text{aussen}} := K_r$;
 $K_{\text{innen}} := K_r$
 – äußere Schleife, die r rückwärts zählt:
 loop
 – innere Schleife über alle ausgewählten Wavelets
 for $j = 1 \dots r$ loop
 $K_{\text{test}} := \text{Kriteriumsfunktion}[\pi(1), \dots, \pi(j-1), \pi(j+1), \dots, \pi(r)]$;
 – gemäß (7.112)
 if $K_{\text{test}} < K_{\text{innen}}$ then
 $j_{\text{best}} := j$
 $K_{\text{innen}} := K_{\text{test}}$;
 end if;
 end loop;
 if $K_{\text{innen}} < K_{\text{aussen}}$ then
 $\pi(k) := \pi(k+1)$ für $k = j_{\text{best}}, \dots, r-1$
 $r := r-1$;
 $K_{\text{aussen}} := K_{\text{innen}}$;
 else
 Algorithmus_zu_Ende();
 end if;
end loop;

Wie bereits bei der sukzessiven Regressorauswahl ist es grundsätzlich kein Problem den Ausdruck K_{test} gemäß (7.112) auszuwerten. Allerdings kann auch hier der Rechenaufwand den Rahmen sprengen, da insbesondere die Inversion der $r \times r$ -Matrix $\mathbf{V}_r^T \mathbf{V}_r$ für große r zu rechenintensiv wird. Dieser Rechenaufwand lässt sich jedoch erheblich reduzieren, wenn wir zum einen auf Größen zurückgreifen, die bei der vorangegangenen Regressorauswahl schon berechnet wurden

und zum anderen auch Größen innerhalb dieses Algorithmus geschickt wiederverwenden. Darüber hinaus ist es auch möglich die mitgeführten Größen der sukzessiven Regressorauswahl so weiterzuführen, dass im Anschluss an eine Regressorelimination wieder neue Regressoren ausgewählt werden können. Unser Fernziel ist also, dass sukzessive Regressorauswahl und Regressoreliminierung alternierend ausgeführt werden können um somit ein möglichst gutes Ergebnis zu finden.

7.7.2 Effiziente Berechnung der Kriteriumsfunktion

Wir gehen nun davon aus, dass wir uns auf Stufe r befinden, d.h. wir haben bereits r Wavelets $\psi_{\pi(1)}, \dots, \psi_{\pi(r)}$ in einer Reihenfolge die durch die Funktion π bestimmt ist. Damit ist bereits eine Matrix \mathbf{V}_r mit vollem Rang gemäß (7.110) vorhanden, sowie eine Matrix \mathbf{P}_r gemäß (7.111). Weiterhin gehen wir davon aus, dass eine Matrix $\mathbf{\Lambda}_r$ mit

$$\mathbf{\Lambda}_r = (\mathbf{V}_r^T \mathbf{V}_r)^{-1}$$

bereits berechnet wurde und vorliegt. Diese Matrix ist symmetrisch und positiv definit. Im oben beschriebenen Rahmenalgorithmus wird in der inneren Schleife über alle bereits ausgewählten Wavelets j geschliffen. Dabei wird in jedem Durchlauf eine Kriteriumsfunktion \tilde{K}_{r-1} der Form

$$\tilde{K}_{r-1} = \tilde{\sigma}_{r-1}^2 \left(1 + \frac{1}{\tau} \text{Spur} \left[\tilde{\mathbf{P}}_{r-1} (\tilde{\mathbf{V}}_{r-1}^T \tilde{\mathbf{V}}_{r-1})^{-1} \right] \right) \quad (7.113)$$

berechnet. Die Tilden über den jeweiligen Größen bedeuten, dass diese von dem jeweiligen j in der inneren Schleife abhängen. Wir wählen diese Schreibweise um eine Doppelindizierung zu vermeiden und die Lesbarkeit zu erleichtern. Wir schreiben also

$$\tilde{\psi} = \psi_{\pi(j)}$$

Für die weiteren Berechnungen wollen wir noch zwei neue Schreibweisen einführen, die besonders nützlich sind, wenn es darum geht ein Wavelet zu eliminieren. Fangen wir mit der Matrix \mathbf{V}_r aus (7.110) an und führen dazu die Matrix \mathbf{V}_r^0 sowie \mathbf{V}_r^- ein mit

$$\begin{aligned} \mathbf{V}_r &= (\underline{v}_1, \dots, \underline{v}_r) \\ \mathbf{V}_r^0 &= (\underline{v}_1, \dots, \underline{v}_{j-1}, \mathbf{0}, \underline{v}_{j+1}, \dots, \underline{v}_r) \\ \mathbf{V}_r^- &= (\underline{v}_1, \dots, \underline{v}_{j-1}, \underline{v}_{j+1}, \dots, \underline{v}_r) \end{aligned}$$

In \mathbf{V}_r^0 ersetzen wir alle Einträge der j -ten Spalte durch Nullen. Bei \mathbf{V}_r^- entfernen wir die j -te Spalte ganz. Natürlich hängen die beiden Ausdrücke vom jeweiligen j ab, also davon, welches Wavelet ausgeschlossen werden soll. Wir könnten somit auch schreiben

$$\tilde{\mathbf{V}}_{r-1} = \mathbf{V}_r^-$$

Bei den quadratischen $r \times r$ -Matrizen wie \mathbf{P}_r oder $\mathbf{\Lambda}_r$ gehen wir ähnlich vor. Nur dass wir hier die j -te Zeile und Spalte durch Nullen ersetzen oder Löschen. Wenn also

$$\mathbf{P}_r = \begin{pmatrix} p_{11} & \cdots & p_{1r} \\ \vdots & \ddots & \vdots \\ p_{r1} & \cdots & p_{rr} \end{pmatrix}$$

dann ist

$$\mathbf{P}_r^0 = \begin{pmatrix} p_{11} & \cdots & p_{1j-1} & 0 & p_{1j+1} & \cdots & p_{1r} \\ \vdots & \ddots & \vdots & \underline{0} & \vdots & \ddots & \vdots \\ p_{j-11} & \cdots & p_{j-1j-1} & 0 & p_{j-1j+1} & \cdots & p_{j-1r} \\ 0 & \underline{0}^T & 0 & 0 & 0 & \underline{0}^T & 0 \\ p_{j+11} & \cdots & p_{j+1j-1} & 0 & p_{j+1j+1} & \cdots & p_{j+1r} \\ \vdots & \ddots & \vdots & \underline{0} & \vdots & \ddots & \vdots \\ p_{r1} & \cdots & p_{rj-1} & 0 & p_{rj+1} & \cdots & p_{rr} \end{pmatrix}$$

und

$$\mathbf{P}_r^- = \begin{pmatrix} p_{11} & \cdots & p_{1j-1} & p_{1j+1} & \cdots & p_{1r} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ p_{j-11} & \cdots & p_{j-1j-1} & p_{j-1j+1} & \cdots & p_{j-1r} \\ p_{j+11} & \cdots & p_{j+1j-1} & p_{j+1j+1} & \cdots & p_{j+1r} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ p_{r1} & \cdots & p_{rj-1} & p_{rj+1} & \cdots & p_{rr} \end{pmatrix}$$

Ganz analog verhält es sich mit $\mathbf{\Lambda}_r$ und den dazugehörigen Matrizen $\mathbf{\Lambda}_r^0$ und $\mathbf{\Lambda}_r^-$. Wenn wir die Definition von \mathbf{P}_r gemäß (7.111) betrachten und nun ein Wavelet eliminieren, dann wird klar, dass

$$\tilde{\mathbf{P}}_{r-1} = \mathbf{P}_r^-$$

Allerdings ist es für $\mathbf{\Lambda}$ nicht so einfach, den

$$\tilde{\mathbf{\Lambda}}_{r-1} = \left(\tilde{\mathbf{V}}_{r-1}^T \tilde{\mathbf{V}}_{r-1} \right)^{-1} = \left(\mathbf{V}_r^{-T} \mathbf{V}_r^- \right)^{-1} \neq \mathbf{\Lambda}_r^-$$

Wenn nun

$$\mathbf{\Lambda}_r = \begin{pmatrix} \lambda_{11} & \cdots & \lambda_{1r} \\ \vdots & \ddots & \vdots \\ \lambda_{r1} & \cdots & \lambda_{rr} \end{pmatrix} = (\underline{\lambda}_1, \dots, \underline{\lambda}_r) \quad \text{mit} \quad \underline{\lambda}_k = \begin{bmatrix} \lambda_{1k} \\ \vdots \\ \lambda_{rk} \end{bmatrix}, k = 1 \dots r$$

dann übernehmen wir die schon für Matrizen eingeführten beiden Schreibweisen auch für Vektoren. Somit sind

$$\underline{\lambda}_k^0 = \begin{bmatrix} \lambda_{1k} \\ \vdots \\ \lambda_{j-1k} \\ 0 \\ \lambda_{j+1k} \\ \vdots \\ \lambda_{rk} \end{bmatrix} \quad \text{und} \quad \underline{\lambda}_k^- = \begin{bmatrix} \lambda_{1k} \\ \vdots \\ \lambda_{j-1k} \\ \lambda_{j+1k} \\ \vdots \\ \lambda_{rk} \end{bmatrix} \quad \text{für } k = 1 \dots r$$

Diese Schreibweisen sollen natürlich auch für andere Vektoren entsprechend gelten. Wie wir bereits bei den Standardalgorithmen in Gleichung (4.51) gesehen haben, hat das Entfernen des Wavelets $\psi_{\pi(j)}$ folgende Auswirkung auf die Matrix Λ :

$$\tilde{\Lambda}_{r-1} = \left(\tilde{\mathbf{V}}_{r-1}^T \tilde{\mathbf{V}}_{r-1} \right)^{-1} = \Lambda_r^- - \lambda_{jj}^{-1} \cdot \underline{\lambda}_j^- \cdot \underline{\lambda}_j^{-T}$$

Um nun in der inneren Schleife des Rahmenalgorithmus für alle $j = 1 \dots r$ den Ausdruck \tilde{K}_{r-1} effizient auszuwerten, müssen wir genauer auf die Ausdrücke $\tilde{\sigma}_{r-1}$ und $\text{Spur}(\tilde{\mathbf{P}}_{r-1} \tilde{\Lambda}_{r-1})$ eingehen. Dabei fangen wir mit $\tilde{\sigma}_{r-1}$ an. Wir wissen, dass

$$\tilde{\sigma}_{r-1} = \frac{\left\| \tilde{\underline{\gamma}}_{r-1} \right\|_2^2}{N - (r - 1)} \quad (7.114)$$

Wenn

$$\underline{b}_r = \begin{bmatrix} b_1 \\ \vdots \\ b_r \end{bmatrix} = \mathbf{V}_r^T \underline{y}$$

,dann gilt

$$\tilde{\underline{b}}_{r-1} = \tilde{\mathbf{V}}_{r-1}^T \underline{y} = \mathbf{V}_r^{-T} \underline{y} = \underline{b}_r^-$$

Damit lässt sich nun die Summe der Residuenquadrate wie folgt berechnen

$$\begin{aligned} \left\| \tilde{\underline{\gamma}}_{r-1} \right\|_2^2 &= \underline{y}^T \underline{y} - \underline{y}^T \tilde{\mathbf{V}}_{r-1} \left(\tilde{\mathbf{V}}_{r-1}^T \tilde{\mathbf{V}}_{r-1} \right)^{-1} \tilde{\mathbf{V}}_{r-1}^T \underline{y} \\ &= \underline{y}^T \underline{y} - \tilde{\underline{b}}_{r-1}^T \tilde{\Lambda}_{r-1} \tilde{\underline{b}}_{r-1} \\ &= \underline{y}^T \underline{y} - \underline{b}_r^{-T} \left(\Lambda_r^- - \lambda_{jj}^{-1} \cdot \underline{\lambda}_j^- \cdot \underline{\lambda}_j^{-T} \right) \underline{b}_r^- \\ &= \underline{y}^T \underline{y} - \underline{b}_r^{-T} \Lambda_r^- \underline{b}_r^- + \lambda_{jj}^{-1} \left(\underline{\lambda}_j^{-T} \underline{b}_r^- \right)^2 \end{aligned} \quad (7.115)$$

Dabei ist

$$\underline{b}_r^{-T} \Lambda_r^- \underline{b}_r^- = \underline{b}_r^T \Lambda_r \underline{b}_r - 2b_j \lambda_j^{-T} \underline{b}_r^- - b_j^2 \lambda_{jj}$$

Wenn wir das nun in (7.115) einsetzen, dann erhalten wir

$$\begin{aligned} \left\| \tilde{\underline{\gamma}}_{r-1} \right\|_2^2 &= \underline{y}^T \underline{y} - \underline{b}_r^T \Lambda_r \underline{b}_r + 2b_j \lambda_j^{-T} \underline{b}_r^- + b_j^2 \lambda_{jj} + \lambda_{jj}^{-1} \left(\underline{\lambda}_j^{-T} \underline{b}_r^- \right)^2 \\ &= \left\| \underline{\gamma}_r \right\|_2^2 + 2b_j \lambda_j^{-T} \underline{b}_r^- + b_j^2 \lambda_{jj} + \lambda_{jj}^{-1} \left(\underline{\lambda}_j^{-T} \underline{b}_r^- \right)^2 \end{aligned}$$

Somit können wir nun $\tilde{\sigma}_{r-1}$ gemäß (7.114) berechnen. Um \tilde{K}_{r-1} aus (7.113) bestimmen zu können, muss noch der folgende Ausdruck berechnet werden:

$$\begin{aligned}
 \text{Spur} \left[\tilde{\mathbf{P}}_{r-1} (\tilde{\mathbf{V}}_{r-1}^T \tilde{\mathbf{V}}_{r-1})^{-1} \right] &= \text{Spur}(\tilde{\mathbf{P}}_{r-1} \tilde{\mathbf{\Lambda}}_{r-1}) \\
 &= \text{Spur} \left[\mathbf{P}_r^- \cdot (\mathbf{\Lambda}_r^- - \lambda_{jj}^{-1} \underline{\lambda}_j^- \underline{\lambda}_j^{-T}) \right] \\
 &= \text{Spur}(\mathbf{P}_r^- \mathbf{\Lambda}_r^-) - \lambda_{jj}^{-1} \text{Spur}(\mathbf{P}_r^- \underline{\lambda}_j^- \underline{\lambda}_j^{-T}) \\
 &= \text{Spur}(\mathbf{P}_r^0 \mathbf{\Lambda}_r^0) - \lambda_{jj}^{-1} \underline{\lambda}_j^{-T} \mathbf{P}_r^- \underline{\lambda}_j^- \\
 &= \text{Spur}(\mathbf{P}_r \mathbf{\Lambda}_r) - 2 \underline{p}_j^T \underline{\lambda}_j + p_{jj} \lambda_{jj} - \lambda_{jj}^{-1} \underline{\lambda}_j^{-T} \mathbf{P}_r^- \underline{\lambda}_j^-
 \end{aligned}$$

Der einzig rechenintensive Teil an dem letzten Ausdruck ist die Auswertung von $\underline{\lambda}_j^{-T} \mathbf{P}_r^- \underline{\lambda}_j^-$. Das ist aber weitaus weniger kritisch als bei der Vorwärts - Auswahl von Regressoren, da hier in der inneren Schleife nur die bereits ausgewählten Wavelets durchlaufen werden und nicht alle Wavelets der Bibliothek.

Mitgeführte Größen

Bei der Vorwärtsregressorauswahl haben wir zu jedem Wavelet der Wavelet-Bibliothek W_{wav} die Vektoren $\dot{\underline{p}}_r$, $\dot{\underline{a}}_r$, $\mathbf{\Lambda}_r \dot{\underline{a}}_r$ und den Skalar $\dot{\underline{a}}_r^T \mathbf{\Lambda}_r \mathbf{P}_r \mathbf{\Lambda}_r \dot{\underline{a}}_r$ berechnet. Mit jedem neu dazu gewählten Wavelet mussten diese Größen für alle Wavelets aktualisiert werden. Da wir im Endeffekt den Algorithmus zur sukzessiven Regressorauswahl und den Algorithmus zur Regressoreliminierung alternierend ausführen wollen, müssen wir alle Größen für jedes Wavelet $\psi_i \in W_{wav}$ anpassen, sobald wir ein Wavelet $\psi_{\pi(j)}$ eliminiert haben. Dabei streichen wir den j -ten Eintrag aus

$$\dot{\underline{p}}_r = \begin{bmatrix} \langle \dot{\psi}, \psi_{\pi(1)} \rangle \\ \vdots \\ \langle \dot{\psi}, \psi_{\pi(r)} \rangle \end{bmatrix} \quad \text{sodass} \quad \dot{\underline{p}}_{r-1} = \dot{\underline{p}}_r^-$$

Ganz ähnlich verhält es sich mit $\dot{\underline{a}}_r$, denn

$$\dot{\underline{a}}_{r-1} = \mathbf{V}_{r-1} \dot{\underline{v}} = \mathbf{V}_r^- \dot{\underline{v}} = \dot{\underline{a}}_r^- \quad \text{für} \quad i = 1 \dots L$$

Dabei sei noch einmal daran erinnert, dass der Punkt über einer Größe nur eine Abkürzung für einen Index i ist. Demnach berechnen wir für alle $i = 1 \dots L$

$$\begin{aligned}
 \mathbf{\Lambda}_{r-1} \dot{\underline{a}}_{r-1} &= (\mathbf{\Lambda}_r^- - \lambda_{jj}^{-1} \underline{\lambda}_j^- \underline{\lambda}_j^{-T}) \cdot \dot{\underline{a}}_r^- \\
 &= \mathbf{\Lambda}_r^- \dot{\underline{a}}_r^- - \lambda_{jj}^{-1} \underline{\lambda}_j^- \underline{\lambda}_j^{-T} \dot{\underline{a}}_r^- \\
 &= (\mathbf{\Lambda}_r \dot{\underline{a}}_r)^- - \underline{\lambda}_j^- \dot{a}_{r,j} - \lambda_{jj}^{-1} \underline{\lambda}_j^- \underline{\lambda}_j^{-T} \dot{\underline{a}}_r^- \\
 &= (\mathbf{\Lambda}_r \dot{\underline{a}}_r)^- - \underline{\lambda}_j^- \cdot \underbrace{(\dot{a}_{r,j} + \lambda_{jj}^{-1} \underline{\lambda}_j^{-T} \dot{\underline{a}}_r^-)}_c
 \end{aligned}$$

Dabei steht $\dot{a}_{r,j}$ für den j -ten Eintrag von $\underline{\dot{a}}_r$. Es fehlt noch die Größe

$$\begin{aligned} \dot{a}_{r-1}^T \mathbf{\Lambda}_{r-1} \dot{\mathbf{P}}_{r-1} \mathbf{\Lambda}_{r-1} \underline{\dot{a}}_{r-1} &= [(\mathbf{\Lambda}_r \underline{\dot{a}}_r)^- - \underline{\lambda}_j^- \dot{c}]^T \mathbf{P}_r^- [(\mathbf{\Lambda}_r \underline{\dot{a}}_r)^- - \underline{\lambda}_j^- \dot{c}] \\ &= (\mathbf{\Lambda}_r \underline{\dot{a}}_r)^{-T} \mathbf{P}_r^- (\mathbf{\Lambda}_r \underline{\dot{a}}_r)^- - 2\dot{c} \underline{\lambda}_j^{-T} \mathbf{P}_r^- (\mathbf{\Lambda}_r \underline{\dot{a}}_r)^- + \dot{c}^2 \underline{\lambda}_j^{-T} \mathbf{P}_r^- \underline{\lambda}_j^- \end{aligned}$$

Dabei ist

$$(\mathbf{\Lambda}_r \underline{\dot{a}}_r)^{-T} \mathbf{P}_r^- (\mathbf{\Lambda}_r \underline{\dot{a}}_r)^- = (\mathbf{\Lambda}_r \underline{\dot{a}}_r)^T \mathbf{P}_r (\mathbf{\Lambda}_r \underline{\dot{a}}_r) - 2(\mathbf{\Lambda}_r \underline{\dot{a}}_r)_j \underline{p}_j^{-T} (\mathbf{\Lambda}_r \underline{\dot{a}}_r)^- - (\mathbf{\Lambda}_r \underline{\dot{a}}_r)_j^2 p_{jj}$$

Man beachte, dass die rechenintensiven Ausdrücke $\underline{\lambda}_j^{-T} \mathbf{P}_r^-$ und $\underline{\lambda}_j^{-T} \mathbf{P}_r^- \underline{\lambda}_j^-$ nicht vom Wavelet ψ abhängen und somit nur einmal berechnet werden müssen. Wie man sieht, reduziert sich der Rechenaufwand pro Wavelet auf vektorwertige Operationen. Das heißt, eine Multiplikation oder Addition von $r \times r$ -Matrizen kann vermieden werden.

7.7.3 Algorithmische Aufbereitung

Nun müssen wir die eben gemachten Überlegungen noch in eine Form bringen, die eine direkte algorithmische Umsetzung erlaubt. Wir wollen also den Rahmenalgorithmus aus Abschnitt 7.7.1 mit Leben füllen und betrachten dazu die folgenden drei Aspekte:

1. Voraussetzungen für den Algorithmus
2. Berechnungen in der inneren Schleife
3. Berechnungen nach Abschluss der inneren Schleife

Voraussetzungen für den Algorithmus

Der Eliminations-Algorithmus setzt voraus, dass bereits eine Auswahl von r Wavelets $\psi_{\pi(1)}, \dots, \psi_{\pi(r)}$ aus der Wavelet-Bibliothek W_{wav} stattgefunden hat. Dabei legt die Funktion π eine Reihenfolge fest. Zudem müssen wir verlangen, dass die folgenden Ausdrücke bereits berechnet vorliegen:

$$\mathbf{V}_r \quad \text{gemäß (7.110)}$$

$$\mathbf{\Lambda}_r = (\mathbf{V}_r^T \mathbf{V}_r)^{-1}$$

$$\mathbf{P}_r \quad \text{gemäß (7.111)}$$

$$K_r \quad \text{gemäß (7.112)}$$

$$\text{Spur}(\mathbf{P}_r \mathbf{\Lambda}_r)$$

$$\|\underline{\gamma}_r\|_2^2$$

Weiterhin gehen wir davon aus, dass zu jedem Wavelet $\psi_i \in W_{wav}$ die folgenden Vektoren berechnet wurden:

$$\begin{aligned} \underline{\dot{p}}_r \\ \underline{\dot{a}}_r \\ \mathbf{\Lambda}_r \underline{\dot{a}}_r \\ \underline{\dot{a}}_r^T \mathbf{\Lambda}_r \mathbf{P}_r \mathbf{\Lambda}_r \underline{\dot{a}}_r \end{aligned}$$

Diese drei Vektoren und der Skalar werden zwar für die Regressorelimination nicht benötigt, müssen aber mitgeführt werden, um anschließend wieder neue Regressoren effizient hinzuwählen zu können.

Berechnungen in der inneren Schleife

Sei

$$\underline{b}_r = \begin{bmatrix} b_1 \\ \vdots \\ b_r \end{bmatrix} = \mathbf{V}_r^T \underline{y}$$

Folgende Berechnungen müssen in der inneren Schleife für alle $j = 1 \dots r$ ausgeführt werden.

$$\begin{aligned} \text{Spur} \left(\tilde{\mathbf{P}}_{r-1} \tilde{\mathbf{\Lambda}}_{r-1} \right) &:= \text{Spur} \left(\mathbf{P}_r \mathbf{\Lambda}_r \right) - 2\underline{p}_j^T \underline{\lambda}_j + p_{jj} \lambda_{jj} - \lambda_{jj}^{-1} \underline{\lambda}_j^{-T} \mathbf{P}_r^{-1} \underline{\lambda}_j^- \\ \left\| \tilde{\underline{\gamma}}_{r-1} \right\|_2^2 &:= \left\| \underline{\gamma}_r \right\|_2^2 + 2b_j \underline{\lambda}_j^{-T} \underline{b}_r^- + b_j^2 \lambda_{jj} + \lambda_{jj}^{-1} \left(\underline{\lambda}_j^{-T} \underline{b}_r^- \right)^2 \\ \tilde{K}_{r-1} &:= \frac{\left\| \tilde{\underline{\gamma}}_{r-1} \right\|_2^2}{N - (r - 1)} \left(1 + \frac{1}{\tau} \text{Spur} \left[\tilde{\mathbf{P}}_{r-1} \tilde{\mathbf{\Lambda}}_{r-1} \right] \right) \end{aligned}$$

Wenn diese drei Größen für ein j berechnet wurden, können sie sofort wieder verworfen werden, falls es bereits eine bessere Hypothese mit kleinerem Kriterium K gab. Nur wenn die gerade aktuelle Hypothese die bislang beste ist, also wenn \tilde{K}_{r-1} das bislang kleinste gefundene Kriterium ist, müssen die drei Größen so lange gespeichert werden, bis eine noch bessere Hypothese gefunden wird.

Berechnungen nach Abschluss der inneren Schleife

Wir befinden uns nun im Rahmenalgorithmus am Ende der inneren Schleife. Dort werden zwei Fälle unterschieden. Entweder es wurde ein Wavelet $\psi_{\pi(j)}$ gefunden, dessen Entfernung ein Kriterium $\tilde{K}_{r-1} < K_r$ hervorgebracht hat oder eben nicht. Falls nicht, wird der Algorithmus beendet. Falls jedoch ein solches Wavelet $\psi_{\pi(j)}$ gefunden wurde, müssen noch einige Größen berechnet und übernommen werden, um die Eliminierung abzuschließen und alle mitgeführten Größen auf den aktuellen Stand zu bringen. Dabei beziehen wir uns im Folgenden

mit unseren Kurzschreibweisen wie \tilde{X} , X^0 und X^- stets auf dieses j :

$$\begin{aligned}\mathbf{V}_{r-1} &:= \mathbf{V}_r^- \\ \mathbf{P}_{r-1} &:= \mathbf{P}_r^- \\ \mathbf{\Lambda}_{r-1} &:= \mathbf{\Lambda}_r^- - \lambda_{jj}^{-1} \cdot \underline{\lambda}_j^- \cdot \underline{\lambda}_j^{-T} \\ \text{Spur}(\mathbf{P}_{r-1}\mathbf{\Lambda}_{r-1}) &:= \text{Spur}\left(\tilde{\mathbf{P}}_{r-1}\tilde{\mathbf{\Lambda}}_{r-1}\right) \\ \|\underline{\gamma}_{r-1}\|_2^2 &:= \|\tilde{\underline{\gamma}}_{r-1}\|_2^2 \\ K_{r-1} &:= \tilde{K}_{r-1}\end{aligned}$$

Weiterhin führen wir die folgenden Größen für alle Wavelets $\psi_i \in W_{wav}$, also für alle $i = 1 \dots L$ fort:

$$\begin{aligned}\dot{\underline{p}}_{r-1} &:= \dot{\underline{p}}_r^- \\ \dot{\underline{a}}_{r-1} &:= \dot{\underline{a}}_r^- \\ \mathbf{\Lambda}_{r-1}\dot{\underline{a}}_{r-1} &:= (\mathbf{\Lambda}_r\dot{\underline{a}}_r)^- - \underline{\lambda}_j^- \cdot \underbrace{(\dot{\underline{a}}_{r,j} + \lambda_{jj}^{-1}\underline{\lambda}_j^{-T}\dot{\underline{a}}_r^-)}_{\dot{c}} \\ \dot{\underline{a}}_{r-1}^T \mathbf{\Lambda}_{r-1} \dot{\mathbf{P}}_{r-1} \mathbf{\Lambda}_{r-1} \dot{\underline{a}}_{r-1} &:= (\mathbf{\Lambda}_r\dot{\underline{a}}_r)^T \mathbf{P}_r (\mathbf{\Lambda}_r\dot{\underline{a}}_r) - 2(\mathbf{\Lambda}_r\dot{\underline{a}}_r)_j \underline{p}_j^{-T} (\mathbf{\Lambda}_r\dot{\underline{a}}_r)^- \\ &\quad - (\mathbf{\Lambda}_r\dot{\underline{a}}_r)_j^2 p_{jj} - 2\dot{c}\underline{\lambda}_j^{-T} \mathbf{P}_r^- (\mathbf{\Lambda}_r\dot{\underline{a}}_r)^- + \dot{c}^2 \underline{\lambda}_j^{-T} \mathbf{P}_r^- \underline{\lambda}_j^-\end{aligned}$$

Somit haben wir alle Größen berechnet, die für das weitere Vorgehen des Eliminations-Algorithmus und eines eventuell anschließenden Regressorauswahl-Algorithmus benötigen. Im Falle einer anschließenden, weiteren Regressorauswahl, müssen die vier Größen $\dot{\underline{p}}_r$, $\dot{\underline{a}}_r$, $\mathbf{\Lambda}_r\dot{\underline{a}}_r$ und $\dot{\underline{a}}_r^T \mathbf{\Lambda}_r \mathbf{P}_r \mathbf{\Lambda}_r \dot{\underline{a}}_r$ im ersten Durchlauf nicht mehr berechnet werden, da sie bereits vorliegen. Im Folgenden werden die Vorwärtsauswahl und die Regressorelimination immer alternierend ausführen, bis die Kriteriumsfunktion K weder durch Hinzuwahl eines neuen, noch durch Abwahl eines bereits ausgewählten Regressors mehr verbessert werden kann.

7.8 Erweiterung der Algorithmen

Bislang wurde nur das Konzept aus Kapitel 6 umgesetzt mit einer reinen Wavelet-Bibliothek und zwei zueinander kompatiblen Algorithmen zur Vorwärtsregressorauswahl und zur Rückwärtselimination. In diesem Abschnitt wollen wir beide Algorithmen auch für erweiterte Wavelet-Netzwerke nutzen, die in Abschnitt 3.2 eingeführt worden sind. Außerdem werden wir zwei zusätzliche Stellgrößen in die Algorithmen einzuführen, um die Robusheit noch besser zu kontrollieren.

Erweiterung des Wavelet-Netzwerkes

Bereits die Standardalgorithmen in Kapitel 4 und die verbesserten Algorithmen in Kapitel 5 wurden auf erweiterte Wavelet-Netzwerke angewendet. Das soll nun auch hier möglich werden. Wir wollen also eine Funktion

$$f(x) = f_{wav}(x) + f_{aff}(x)$$

wie eingeführt in (3.25) erhalten. Hierbei ist f_{wav} ein reines Wavelet-Netzwerk, wie bislang betrachtet und $f_{aff}(x) = a_{aff}^T x + c_{aff}$. Um dieses Konstrukt in die bestehenden Algorithmen zu integrieren, führen wir die Teilbibliothek der affinen Terme ein, wie schon in (4.35) beschrieben, also

$$W_{aff} = \{\psi_{i,aff} | \psi_{0,aff}(x) = w_{0,aff} \wedge \psi_{i,aff}(x) = w_{i,aff} e_i^T x \forall i = 1 \dots d\}$$

Das konkrete Ziel ist es dabei, die $\psi_{i,aff}$ in den Algorithmen genau so verwenden wie ordinäre Wavelets. In beiden Algorithmen mit dem neuen Gütekriterium K ergibt sich aber ein wesentliches Problem, und zwar genau an der Stelle wo das Skalarprodukt p_{ij} zweier Wavelets in $L^2(\mathbb{R}^d, \mathbb{R}^d)$ berechnet werden soll. Keiner der affinen Terme befindet sich in diesem Funktionenraum. Wie also soll man p_{ij} wählen, falls ein oder zwei „unechte Wavelets“ im Spiel sind? Betrachten wir zunächst den Fall eines echten Wavelets ψ_j und eines affinen Terms. Wir betrachten das Integral

$$\int_{\mathbb{R}^d} \psi_{0,aff}(x) \psi_j(x) dx = w_{0,aff} \cdot \int_{\mathbb{R}^d} \psi_j(x) dx = 0$$

, wobei dx wie gewohnt als Kurzschreibweise für $d\lambda(x)$ dient. Weiterhin gilt für $i = 1 \dots d$:

$$\begin{aligned} \int_{\mathbb{R}^d} \psi_{i,aff}(x) \psi_j(x) dx &= \int_{\mathbb{R}^d} w_{i,aff} e_i^T x \cdot w_j \psi[a_j \star (x - t_j)] dx \\ &= \int_{\mathbb{R}^d} w_{i,aff} e_i^T (x + t_j) \cdot w_j \psi[a_j \star x] dx \\ &= \int_{\mathbb{R}^d} w_{i,aff} e_i^T x \cdot w_j \psi[a_j \star x] dx = 0 \end{aligned}$$

Die letzte Gleichheit gilt deshalb, weil das Mexikanerhut-Mutter-Wavelet ψ in allen d Komponenten von x eine gerade Funktion ist. Mit diesen Betrachtungen liegt es nahe, $p_{ij} = 0$ zu wählen, wenn ψ_i für einen affinen Term steht und ψ_j für ein echtes Wavelet. Im Falle von zwei affinen Termen wäre ein entsprechendes Integral nicht definiert. In diesem Fall lehnen wir uns etwas aus dem Fenster und wählen ebenfalls $p_{i,j} = 0$. Ist also W eine erweiterte Wavelet-Bibliothek gemäß

$$W = W_{wav} \cup W_{aff}$$

und sind $\psi_i, \psi_j \in W$, dann wählen wir in den beiden betrachteten Algorithmen jeweils

$$p_{ij} = \begin{cases} \langle \psi_i, \psi_j \rangle & \text{für } \psi_i \in W_{wav} \text{ und } \psi_j \in W_{wav} \\ 0 & \text{sonst} \end{cases}$$

Welche Konsequenz hat dieses Vorgehen anschaulich? Wackelt man an den Trainingsdatenpunkten, dann wackeln auch die Gewichte der Wavelets und der affinen Terme. Allerdings schlägt sich nur die Stärke des Wackelns der echten Wavelets im Modellunsicherheitsfaktor MUF nieder, der in (6.85) eingeführt wurde, nicht aber das Wackeln der affinen Terme. Dieses Manko ist jedoch vertretbar, berücksichtigt man, dass die Anpassung der affinen Terme fast immer sehr gut konditioniert ist. Bei der Implementierung der Algorithmen sind wir genau so vorgegangen wie hier beschrieben.

Modellunsicherheitsfaktor und minimale Standardabweichung

Wie wir im nächsten Abschnitt sehen werden, kann es vorkommen, dass sich ein Wavelet-Netzwerk „zu gut“ an gegebene Trainingsdaten \mathbb{O}_1^N anpasst. Man kann auch sagen, es findet ein Overfitting statt. Dieses Verhalten ist unerwünscht, weil dabei trotz guter Anpassung des Wavelet-Netzwerkes an die Trainingsdaten ein schlechteres Modell entsteht. Betrachten wir deshalb die Kriteriumsfunction K gemäß Gleichung (6.85), also

$$K = \sigma^2 (1 + MUF)$$

mit der Stichprobenvarianz σ^2 und dem Modellunsicherheitsfaktor MUF . An beiden Größen wollen wir nun ansetzen. Ein Ansatz besteht darin, den Modellunsicherheitsfaktor schlichtweg zu begrenzen, indem man in den Algorithmen fordert, dass

$$MUF \leq MUF_{max}$$

Bei der sukzessiven Regressorauswahl wächst MUF mit jedem zusätzlichen Wavelet. Bei der Regressoreliminierung wird MUF mit jedem aussortierten Wavelet auch wieder kleiner. Die Regressorauswahl wird also beendet, wenn man MUF_{max} überschreiten würde. Etwas anders wollen wir bei der Beschränkung von σ^2 vorgehen. Eine solche Beschränkung kann unter Umständen Sinn machen, wenn man davon ausgeht, dass die wahre Varianz $\tilde{\sigma}^2$ nicht unter einer gewissen Schranke liegt, also

$$\tilde{\sigma}^2 \geq \tilde{\sigma}_{min}^2$$

gilt. In diesem Fall haben wir das Kriterium K leicht abgewandelt zu

$$K^* = \sigma^2 + \min(\sigma^2, \tilde{\sigma}_{min}^2) \cdot MUF$$

Mit diesem abgewandelten Kriterium wird zwar eine weitere Anpassung an die Trainingsdaten stets ermöglicht, es wird aber verhindert, dass ein großer Modellunsicherheitsfaktor durch eine zu klein geschätzte Stichprobenvarianz kompensiert wird. Beide Ansätze wurden im Demonstratornetzwerk implementiert.

7.9 Ergebnisse der statistischen Algorithmen

In dieser Sektion werden die gleichen Beispielprobleme betrachtet wie in Abschnitt 4.4 mit den Standardalgorithmen und in Abschnitt 5.4 mit den heuristisch verbesserten Algorithmen. Dadurch soll ein finaler Vergleich ermöglicht werden.

Problem 1: Das gutartige Problem

Für den Trainingsdatensatz und die zu Grunde liegende Funktion f sei wiederum auf Abschnitt 4.4 verwiesen, wo die Resultate der Standardalgorithmen beschrieben wurden. Der Trainingsdatensatz ist in Abbildung 4 graphisch dargestellt. Diesmal wurde die neu entwickelte, rekursiv erstellte, reichhaltige Wavelet-Bibliothek aus Abschnitt 7.1 mit den Parametern $\beta = 0.5$ und $m_{min} = -6$ eingesetzt. Orientiert man sich zu Vorstellungszwecken an Abbildung 22, entspricht dies der doppelten Dichte an Wavelets und einer Abstufung bis zur Ebene $m = -6$. Auch affine Terme wurden beigelegt. Eine so reichhaltige Bibliothek ist von Nutzen, da die Wavelets nur noch ausgewählt und in ihrer Gewichtung angepasst werden, nicht aber mehr gestreckt oder verschoben. Bei der folgenden alternierenden Regressorauswahl mit Konditionsbetrachtung wurden

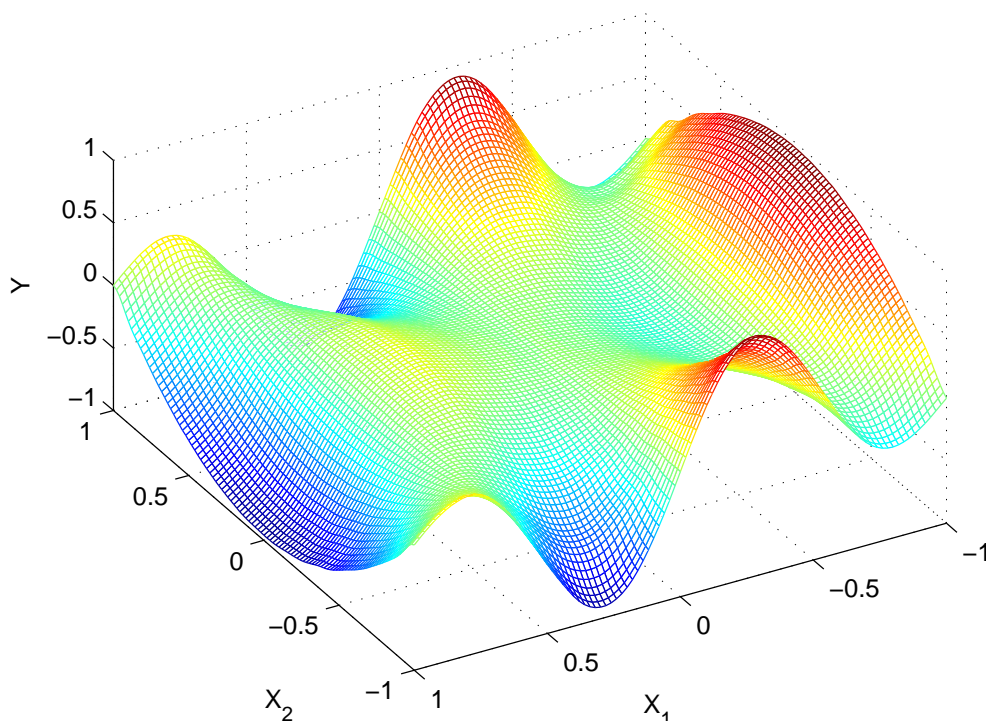
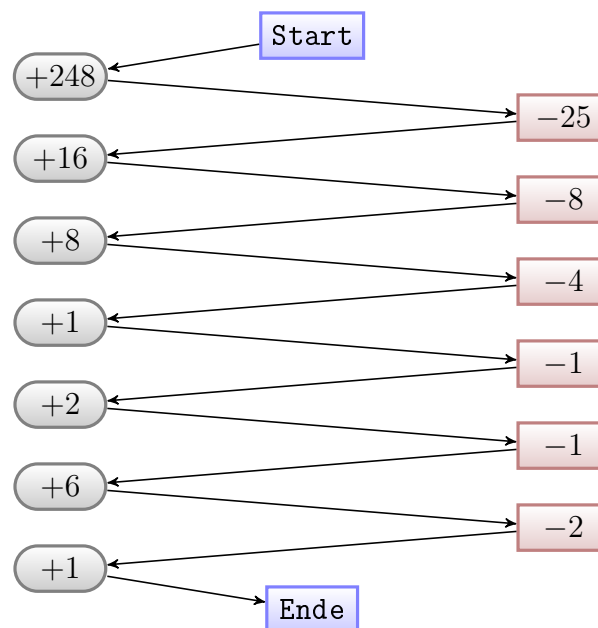


Abbildung 24: f nach statistischer Regressorauswahl bei Problem 1

168 Wavelets ausgewählt, bei einer Begrenzung des Modellunsicherheits-Faktors MUF auf 1. Das Ergebnis sehen wir in Abbildung 24. Es ergibt sich eine Wurzel des mittleren quadratischen Fehlers $RMSE \approx 0,0003$. Das heißt die Trainingsdaten wurden noch einmal um Faktor zehn genauer approximiert als bei den bisherigen Algorithmen. Lässt man die Beschränkung des Modellunsicherheits-Faktors fallen, wählt der Algorithmus 398 Regressoren aus, erreicht einen Wert von $MUF \approx 7,7$ und $RMSE \approx 3 * 10^{-8}$ und bricht die Auswahl von weiteren Wavelets auch nur aus numerischen Gründen ab. Optisch entsteht kaum ein Unterschied. Dieses Verhalten ist bedenklich, lässt sich aber damit erklären, dass die zu Grunde liegenden Trainingsdaten sich „zu gut“ durch Wavelets approximieren lassen. Entfernt man die Schicht der kleinsten Wavelets aus der Wavelet - Bibliothek, beschränkt sich also auf $m_{min} = -5$, werden 240 Wavelets ausgewählt. Dabei erhalten wir $RMSE \approx 0,00014$ und leider immer noch einen Modellunsicherheits-Faktor von $MUF \approx 7,3$. Die Funktion in Abbildung 24 wird augenscheinlich noch etwas glatter, also besser. Dass das Konzept der alternierenden Regressoraus- und Abwahl seine Berechtigung hat, zeigt sich sehr eindrucksvoll im folgenden Diagramm. Darin ist abgebildet, wie viele Wavelets der Algorithmus jeweils alternierend dazu gewählt und aussortiert hat:



Eine nochmalige Reduzierung der Wavelet-Bibliothek auf $m_{min} = -4$ reduziert schließlich die Modellunsicherheit auf $MUF \approx 2,2$ bei einer Auswahl von nur 62 Wavelets. Die Anpassung an die Trainingsdaten wird jedoch etwas schlechter auf $RMSE \approx 0,0030$. Was folgern wir aus diesem Verhalten? Hat eine Wavelet-Bibliothek zu viele und zu kleine Wavelets, dann ist sie in der Lage sich an fast alles beliebig gut anzupassen. Das ist nicht gewollt, da somit auch eine Anpassung an alle möglichen Fehler erfolgt. Daher sollte eine Wavelet-Bibliothek nur so viele Schichten wie nötig erhalten, was letztlich der Anwender vorgeben muss. Für eine bessere Lösung sei zunächst auf Abschnitt 7.10 vertröstet.

Problem 2: Das realistische Problem

Dieses Problem ist für unseren Algorithmus der alternierenden Regressorauswahl mit Konditionsbetrachtung wohl das relevanteste von den dreien, weil wir hierbei eine zu Grunde liegende Funktion \tilde{f} und normalverteilte Fehler haben. Dabei ist $\mathbb{O}_1^N = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ein Trainingsdatensatz mit $N = 500$ Punkten und

$$\tilde{f} : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{mit} \quad \tilde{f}(x) = \sin(x_1^2) \cos(x_2)$$

Die 500 Vektoren x_i waren zufällige Realisierungen eines Zufallsvektors X , wie unter Abschnitt 4.4 beschrieben. Die y_i wurden wieder erzeugt nach der Formel $\tilde{y}_i = \tilde{f}(x_i) + e_i$, wobei die e_i wiederum Realisierungen von paarweise unabhängigen $N(0; 0, 1^2)$ -verteilten Zufallsvariablen waren. Die Trainingsdaten sind dargestellt in Abbildung 7. Analog zum ersten Problem wurde auch hier eine reichhaltige, rekursiv erstellte Wavelet-Bibliothek mit den Parametern $\beta = 0.5$ und $m_{min} = -6$ zu Grunde gelegt. Diesmal wurde der alternierende Regressorauswahl-Algorithmus mit Konditionsbetrachtung ohne Einschränkungen angewendet. Dabei wurden 79 Wavelets ausgewählt. Der Modellunsicherheitsfaktor von $MUF \approx 0,2$ und eine Wurzel des mittleren quadratischen Fehlers von $RMSE \approx 0,082$ wurden erreicht. Das entspricht einer, aus den Trainingsdaten geschätzten, Standardabweichung von 0,9. Das Ergebnis sehen wir in Abbildung 25. Die Güte dieser Anpassung lässt sich mit dem Auswertedatensatz \mathbb{O}_2^N bestimmen, der nach der gleichen Funktion \tilde{f} und den gleichen Verteilungen wie der Trainingsdatensatz \mathbb{O}_1^N erstellt wurde, nur mit anderen Realisierungen für die (x_i, y_i) . Dabei erhielten wir $Eval - RMSE \approx 0,2$. Das stellt eine leichte Verbesserung gegenüber den bisherigen Algorithmen dar. Wir erkennen aber auch die rote Überhöhung am rechten Rand in Abbildung 25. Zeichnet man in den Graph des Wavelet - Netzwerkes noch die Trainingsdaten aus \mathbb{O}_1^N ein, so erhält man Abbildung 26. Dabei wird deutlich, dass im Bereich der Überhöhung fast keine Trainingsdaten vorhanden sind. In der Abbildung sieht man zwei der fünf Datenpunkte, welche diesen Hügel verursachen. Das Anpassungsproblem ist in diesem Bereich offensichtlich schlecht konditioniert. Dieses Verhalten des Algorithmus lässt die Schlussfolgerung zu, dass er keine geeigneteren Wavelets findet. Bei den gewählten Wavelets wiegt scheinbar die Verbesserung des Varianzschätzers σ^2 schwerer als die Verschlechterung des Modellunsicherheitsfaktors MUF . Dieses Resultat ist leider nicht gänzlich befriedigend. Allerdings sind im Auswerte-Datensatz, wegen der gleichen Verteilung, auch nur wenige Datenpunkte im Bereich des Hügels zu erwarten und somit nur geringe Fehler. In den Bereichen mit vielen Datenpunkten funktioniert die Anpassung glücklicher Weise sehr gut. Erweitert man die Wavelet-Bibliothek um zwei Ebenen ($m_{min} = -8$), so werden auch hier zu viele Wavelets ausgewählt und das Netzwerk passt sich „zu gut“ an die Trainingsdaten an. Wir werden dieses Problem in Abschnitt 7.10 wieder aufgreifen und lösen.

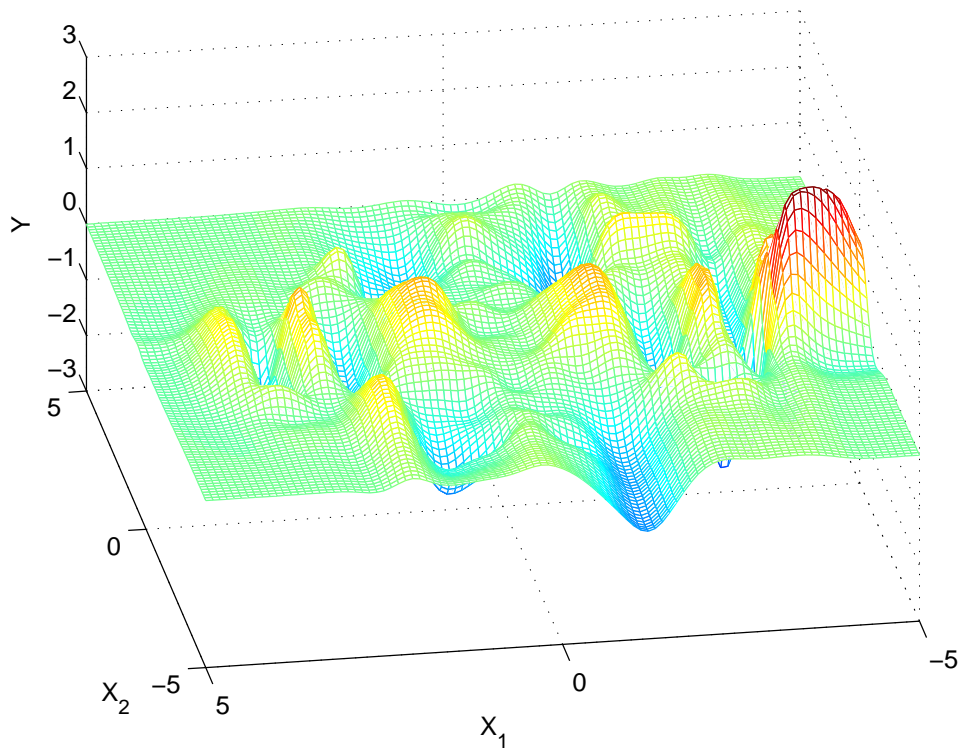


Abbildung 25: f nach statistischer Regressorauswahl bei Problem 2

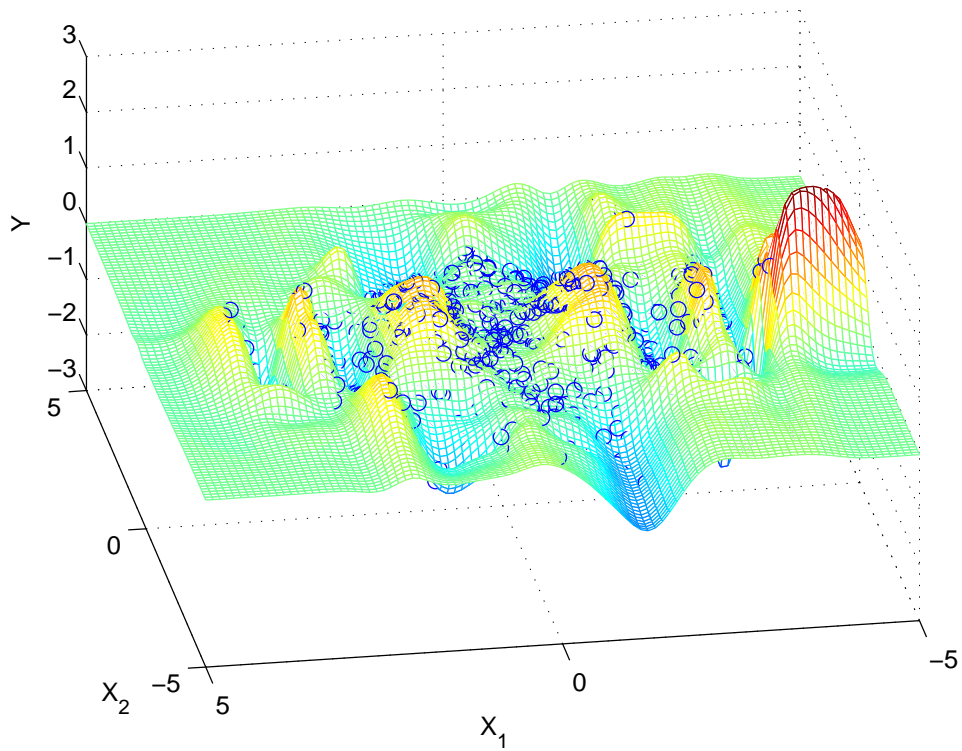
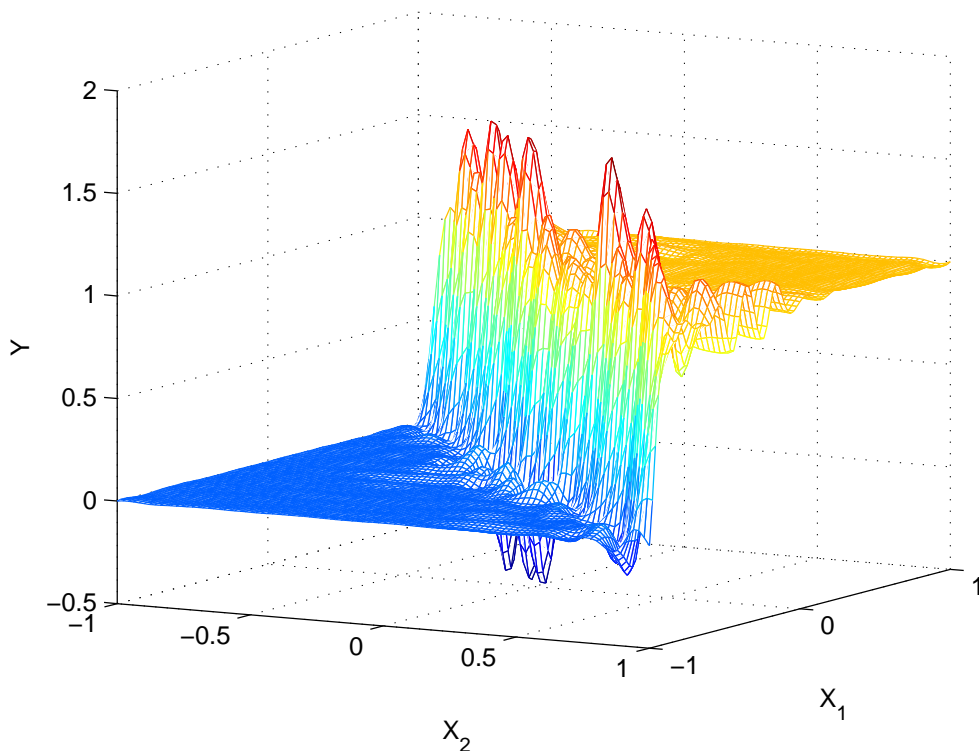
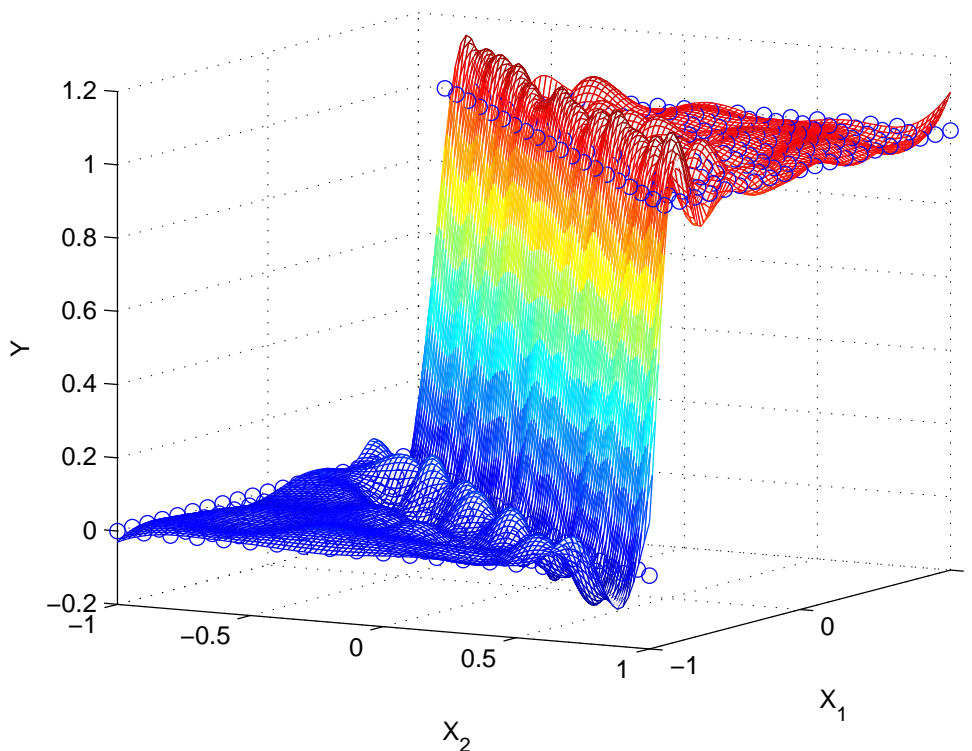


Abbildung 26: f nach statistischer Regressorauswahl bei Problem 2

Problem 3: Das schwierige Problem

Beim dritten und letzten Problem bestand die Aufgabe darin, eine Stufenfunktion mit einem Wavelet - Netzwerk anzupassen. Dabei lagen die x - Werte der Trainingsdatenpunkte auf einem regulären Gitter mit Gitterabstand 0,1 auf dem Quadrat $[-1,1] \times [-1,1]$. Die y - Werte waren 0 für $x_1 + x_2 \leq 0$ und 1 sonst. Der Trainingsdatensatz wurde graphisch dargestellt in Abbildung 10. Analog zu den letzten beiden Problemen wurde eine reichhaltige Wavelet-Bibliothek rekursiv erzeugt, wie in Abschnitt 7.1 beschrieben. Dabei wurden die Parameter $\beta = 0.5$ und $m_{min} = -6$ gewählt. Danach wurde eine alternierende Regressorauswahl mit Konditionsbetrachtung durchgeführt, wobei der Modellunsicherheitsfaktor MUF auf 1 begrenzt wurde. Dabei wurden 190 Wavelets ausgewählt und es wurde eine Wurzel des mittleren quadratischen Fehlers von $RMSE \approx 0,0013$ erzielt. Das Ergebnis sehen wir in Abbildung 27. Verzichtet man auf die Begrenzung der Modellunsicherheit, so bricht der Algorithmus erst nach 408 Wavelets mit $RMSE \approx 6 \cdot 10^{-8}$ bei einem Modellunsicherheits-Faktor von $MUF \approx 7$ aus numerischen Gründen ab. Augenscheinlich ändert sich fast nichts. Dieses Verhalten der Überanpassung deuten wir als Indiz dafür, dass wir zu viele Wavelets zur Verfügung gestellt haben. Nehmen wir aus der Wavelet-Bibliothek die Schicht der kleinsten Wavelets heraus und reduzieren uns somit

Abbildung 27: f nach statistischer Regressorauswahl bei Problem 3

Abbildung 28: f nach statistischer Regressorauswahl bei Problem 3

auf $m_{min} = -5$, ist das Verhalten ein ganz anderes. Bei der anschließenden alternierenden Regressorauswahl mit Konditionsbetrachtung und ohne weitere Einschränkungen, erhalten wir das Ergebnis in Abbildung 28. Dabei wurden die Trainingsdatenpunkte mit eingezeichnet. Diesmal wurden nur 67 Wavelets ausgewählt und eine Abweichung $RMSE \approx 0,074$ von den Trainingsdaten bei einem Modellunsicherheits-Faktor $MUF \approx 0.28$ ausgewählt. Das Ergebnis hat deutlich weniger Überschwinger, passt sich jedoch in den Ebenen auch schlechter an die Trainingsdaten an.

Probleme höherer Dimension

Das Programm wurde grundsätzlich für Probleme beliebiger Dimensionalität ausgelegt. Das Problem ist lediglich, dass sich eine Funktion $f : \mathbb{R}^d \rightarrow \mathbb{R}$ mit $d > 2$ graphisch nicht mehr darstellen lässt. Es wurde auch ein fünfdimensionales Modellproblem mit statistisch verteilten x -Werten und einem statistischen Fehler untersucht. Dieses Problem erzielte bereits bei den Standardalgorithmen hinreichend gute Ergebnisse und war daher von weniger Interesse in dieser Arbeit. Der limitierende Faktor bei Problemen hoher Dimensionalität ist zum einen die Wavelet-Bibliothek, zum anderen sind es die Trainingsdaten. Der Umfang einer Wavelet-Bibliothek auf einem regulären Gitter steigt

exponentiell mit der Dimension. Eine Cluster-Bibliothek ist eine erste Lösungsmöglichkeit. Allerdings können damit nicht alle Funktionen gut beschrieben werden. Bei Problemen höherer Dimension sollten auch genügend Trainingsdatenpunkte $(x_i, y_i) \in \mathbb{O}_1^N$ vorhanden sein um nicht in ein Sparse-Data-Problem zu kommen. Nur als Gedankenspiel: würde man im \mathbb{R}^5 in einen fünf dimensionalen Hyperquader die gleiche Trainingsdatendichte einbringen wollen, wie in den hier betrachteten \mathbb{R}^2 -Problemen, dann müssten bei einem regulären Gitter von Stützstellen x_i an jeder Kante in jeder Dimension 21 Datenpunkte eingeführt werden. Mit anderen Worten: man würde $21^5 \approx 4.000.000$ Trainingsdatenpunkte benötigen. Es gilt folglich weiterhin: „There’s no free lunch“.

7.10 Verbesserung der Varianz-Schätzung

Im letzten Kapitel haben wir gezeigt, dass die implementierten Algorithmen durchaus passable Ergebnisse erzielen. Ein Problem hat sich jedoch durch alle Beispielprobleme gezogen: Sobald man die Wavelet-Bibliothek zu groß wählt, also zu viele Schichten einrichtet, passt sich das Wavelet-Netzwerk „zu sehr“ an die Trainingsdaten an. Dabei werden zu viele Wavelets ausgewählt. Betrachten wir noch einmal die Kriteriumsfunktion K aus (6.85), also

$$K = \sigma^2 (1 + MUF)$$

Dabei kam σ^2 aus (6.81) gemäß

$$\sigma^2 = \text{Var}_{emp}(Y_i) = \text{Var}_{emp}(\Delta Y_i) = \frac{\|\underline{\gamma}\|_2^2}{N - r}$$

mit $\|\underline{\gamma}\|_2^2$ aus (6.79) gemäß

$$\|\underline{\gamma}\|_2^2 = \underline{y}^T \underline{y} - \underline{y}^T \mathbf{V}_r (\mathbf{V}_r^T \mathbf{V}_r)^{-1} \mathbf{V}_r^T \underline{y}$$

Bei der sukzessiven Regressorauswahl steigt MUF mit jedem zusätzlichen Wavelet, wogegen σ^2 fällt. Im Prinzip wählt der Algorithmus neue Wavelets hinzu, solange die geschätzte Varianz σ^2 schneller fällt als der Modellunsicherheitsfaktor MUF steigt. Die Beobachtungen aus dem letzten Abschnitt legen nahe, dass die Varianz-Schätzung nach unten verfälscht wird, wenn die Wavelet-Bibliothek zu groß ist. Dieses Phänomen lässt sich plausibel erklären. Befinden wir uns in einer Hypothese H_{I_r} , wie sie in Definition 6.1 eingeführt wurde, dann haben wir die „schöne Welt“ eines linearen statistischen Modells mit N Dimensionen und r Freiheitsgraden. Dieses Modell wurde in Abschnitt 2.6 eingeführt. Die r Freiheitsgrade sind dabei die Gewichte w_i der gewählten r Wavelets. Die d -dimensionalen Parameter-Vektoren a_i und t_i der Wavelets sind innerhalb einer Hypothese H_{I_r} fest. Betrachten wir hingegen die Menge aller Hypothesen und die dazugehörigen Auswahlalgorithmen, dann wählen wir ja eine bestimmte Hypothese und damit auch bestimmte Parameter a_i und t_i und zwar gerade so, dass K minimal wird. Betrachten wir eine Wavelet-Bibliothek

$$W_{wav} = \{\psi_i | i = 1 \dots L\}$$

, wie sie im Abschnitt 7.1 konstruiert wurde. Stellen wir uns nun vor, wir würden die Wavelets immer dichter zusammen rücken lassen und immer mehr Zwischenschichten einziehen. Dann würde W_{wav} auch immer mehr Wavelets enthalten, also im Grenzfall wieder zu einer kontinuierlich parametrisierten Wavelet-Familie werden. Der Algorithmus zur Regressorauswahl könnte in diesem Grenzfall die Parameter-Vektoren a_i und t_i völlig frei wählen. Er wählt genau jenes Wavelet mit den a_i und t_i aus, die zum besten Kriterium K führen. Somit hätten wir pro Wavelet ψ_i nicht nur einen freien Parameter w_i sondern gleich $2d + 1$ freie Parameter. Nun haben zwar keine kontinuierliche Wavelet-Familie und die a_i und t_i sind keine völlig frei wählbaren Parameter. Jedoch besteht Wahlfreiheit dieser Parameter innerhalb des Reservoirs von W_{wav} . Wir wollen daher a_i und t_i als semi-freie Parameter bezeichnen. Je mehr Wavelets eine Bibliothek enthält, als um so freier können sie angesehen werden. Unter einer Hypothese H_{I_r} findet durch die r freien Parameter w_i zwangsläufig eine „Überanpassung“ an die Trainingsdaten statt. Dieser Überanpassung durch die freien Parameter kompensiert man in der Stichprobenvarianz

$$\sigma^2 = \frac{\|\gamma\|_2^2}{N - r}$$

durch den Nenner $N - r$. Erst dadurch wird die Stichprobenvarianz im linearen statistischen Modell zu einem erwartungstreuen Schätzer. Ebenso findet im Zuge der Hypothesenauswahl eine Überanpassung an die Trainingsdaten durch selektive Auswahl der semi-freien Parameter-Vektoren a_i und t_i statt, also durch die selektive Auswahl einer bestimmten Hypothese. Als Lösungsansatz schlagen wir hier eine Abwandlung der Stichprobenvarianz σ^2 vor, die diesen zusätzlichen Parametern Rechnung trägt. Es sei

$$\sigma_*^2 = \frac{\|\gamma\|_2^2}{N - r(2d \cdot \xi + 1)} \quad (7.116)$$

mit $\xi \in [0, 1]$. Dabei ist ξ ein Maß für die Freiheit der semi-freien Parameter-Vektoren a_i und t_i . Wir sind auf Nummer sicher gegangen und haben einige Test mit $\xi = 1$ angestrengt. Das heißt, wir haben alle Komponenten der d -dimensionalen Vektoren a_i und t_i wie freie Parameter im linearen statistischen Modell angesehen. Dabei wurden sehr gute, robuste Ergebnisse erzielt. Es wurden weniger Wavelets ausgewählt und die Anpassung an die Trainingsdaten im Sinne des mittleren quadratischen Fehlers wurde schlechter. Auch bei großen Wavelet-Bibliotheken waren keine fehl- oder Überanpassungen zu verzeichnen. Eine Beschränkung des Modellunsicherheitsfaktors MUF nach oben oder der Varianz σ^2 nach unten wurde nicht mehr benötigt. Natürlich weicht uns bei diesem heuristischen Eingriff in die Formel für die Stichprobenvarianz das schöne mathematische Fundament des linearen statistischen Modells gehörig auf. Wir halten den Eingriff dennoch für wohl begründet, vertretbar und empfehlenswert. Der Änderungsaufwand an den bestehenden Algorithmen ist minimal. Mit diesem Resultat geben wir uns zufrieden und beenden den Arbeitsteil der Dissertation.

8 Fazit und Ausblick

Die Probleme bei der Verwendung von Standardalgorithmen zur Black-Box-Modellierung mit Wavelet-Netzwerken sind an den vorgestellten Beispielen sehr deutlich geworden. In einem ersten Ansatz wurde bei allen Konstruktionsschritten eingegriffen und es wurden robustere Alternativen vorgestellt. Damit konnten teilweise erhebliche Verbesserungen erzielt werden. Die Cluster-Bibliothek hat sich für einige Probleme bewährt, für andere weniger. Sie ist besonders für statistisch gestreute Datensätze zu empfehlen. Ein Nachteil ist der Verlust des theoretischen Unterbaus der regulären Bibliothek. Sehr gute Ergebnisse zeigte die robustifizierte Vorwärtsregressorauswahl. Dabei trat nur eine geringe Verschlechterung des mittleren quadratischen Fehlers gegenüber dem Standardalgorithmus auf. Auch der verbesserte Levenberg-Marquardt-Algorithmus verzeichnet nur geringe MSE-Einbußen und erhöht die Robustheit deutlich. Allerdings konnte er das Spiking im schwierigen Problem nicht verhindern. Diese ersten Verbesserungsansätze waren rein heuristisch motiviert. Mit der Einführung des neuen Gütekriteriums K wurde die robuste Regressorauswahl erstmals auf ein solides mathematisches Fundament gestellt. Auf diesem Fundament wurde ein kombinierter Algorithmus errichtet, der alternierend Regressoren hinzuwählen und abwählen kann. Er verbindet die Vorteile einer sukzessiven Regressorauswahl und einer Rückwärtselimination von Regressoren. Die geschickte algorithmische Umsetzung ermöglicht eine annähernd gleich effiziente Berechnung wie mit den Standard-Algorithmen trotz höherer Komplexität der Kriteriumsfunktion. Das Problem der verfälschten Varianz-Schätzung kann gut kompensiert werden. Die erzielten Testergebnisse überzeugen. Diese Arbeit bietet gleichwohl Ansätze für weitere Entwicklungen. Es wäre schön, einen Algorithmus zur Regressoroptimierung mit dem hier eingeführten Gütekriterium K zu gewinnen. Die Schwierigkeit ist dabei, dass in jedem Schritt zumindest ein Gradient und eine Approximation an die Hessematrix dieser Zielfunktion bestimmt werden müssen. Deshalb hat man hier mehr Einschränkungen als bei der Regressorauswahl. Weiterhin wäre es interessant, Wavelet-Netzwerke mit anderen Mutter-Wavelets zu testen. Der programmiertechnische Zusatzaufwand bei dem gegebenen Demonstrator hielt sich in Grenzen. Summa summarum sollte das Potential von Wavelet-Netzwerken deutlich geworden sein. Mit Hilfe des neuen Gütekriteriums K gibt es ein mathematisch fundiertes, konstruktives Verfahren, welches auch das Verhalten des Modells zwischen Trainingsdaten berücksichtigt. Hinzu kommen noch vorteilhafte mathematische Eigenschaften des fertig trainierten Wavelet-Netzwerkes, auf die in dieser Arbeit nicht eingegangen wurde. Diese Eigenschaften werden z.B. bei der Gesichtserkennung in [5] verwendet.

Dennoch: Black-Box-Modellierung, auch mit Wavelet-Netzwerken, ist kein Allheilmittel. An einer großen Anzahl von aussagekräftigen Trainingsdaten führt kein Weg vorbei. Probleme hoher Dimension bleiben stets schwierig. Ein für den Laien verwendbares Tool für Anwendung auf universelle Probleme ist nicht absehbar. Wann immer klassische Modellierung zumindest in Teile möglich ist,

sollte man versuchen diese einzubringen. Gleiches gilt für Informationen über Randbedingungen. Mit den richtigen Algorithmen sind Wavelet-Netzwerke sehr gut zur Black-Box-Modellierung geeignet. Hat man jedoch die Wahl, bleibt eine klassische Modellierung dem Black-Box-Ansatz vorzuziehen.

Literatur

- [1] Q. Zhang and A. Benveniste, *Wavelet networks* IEEE trans. on Neural Networks, vol. 3, pp. 889-898, Nov. 1992
- [2] Q. Zhang, *Using Wavelet Network in Nonparametric Estimation*, Technical Report 833, IRISA, May 1994
- [3] R. Galvao et al., *Linear-Wavelet Networks*, Int. J. Appl. Math. Comput. Sci., Vol. 14, No. 2, 221-232, 2004
- [4] Sheng-Tun Li and Shu-Ching Chen, *Function Approximation using Robust Wavelet Neural Networks* Proceedings of the 14th IEEE International Conference on tools with Artificial Intelligence (IC-TAI'02), 2002
- [5] V. Krueger, G. Sommer *Wavelet Networks for Face Processing*, JOSA A, Vol. 19, Issue 6, pp. 1112-1119
- [6] J. Zhang, *Wavelet Neural Networks for Function Learning*, IEEE Transactions on Signal Processing, Vol. 43, No. 6, June 1995
- [7] S. Schäffler, *Wavelets Teil 1*, internes Papier, 2007
- [8] Christian Blatter, *Wavelets - Eine Einführung*, Vieweg, Advanced Lectures in Mathematics, ISBN: 3-528-06947-3, 1998
- [9] Mathias Richter, *Numerik*, Skriptum zur Vorlesung, Herbst 2007
- [10] D. Meintrup und S. Schäffler, *Stochastik - Theorie und Anwendungen*, ISBN 3-540-21676-6, Springer Verlag, 2005
- [11] I. Daubechies, *Ten Lectures on Wavelets*, ISBN 0-89871-274-2, Society for Industrial and Applied Mathematics, 1992
- [12] M. Prescher, *Robuste Risiko-Optimierung mit Multi-Objective Neural Networks*, ISBN 978-3-8325-2079-3, Logos Verlag Berlin, 2008
- [13] D. Pohl, *Robuste Wavelet-Netzwerke*, Diplomarbeit, Dezember 2007

Danksagung

Größter Dank gebührt zweifellos meinem Doktorvater, Herrn Professor Dr. Dr. Stefan Schäffler. Er war es, der mich auf den rechten Weg der externen Promotion brachte, mich auch in schwierigen Phasen motivierte und unterstützte und mich letztlich dazu ermutigte diese Dissertation auch abzuschließen. Seiner Führung, seinen Beiträgen und seiner Arbeit verdanke ich es, heute eine Danksagung verfassen zu können. Gleich an zweiter Stelle danke ich Professor Dr. Mathias Richter für die sehr umfangreiche und zeitintensive Arbeit als Gutachter und für seinen fundierten Rat in allen mathematischen Detailfragen. Dr. Robert Schmied danke ich für die Hilfe bei unzähligen technischen Hürden, die mich ohne ihn wesentlich länger aufgehalten hätten. Weiterhin danke ich Dr. Rainer von Chossy für seine Kommentare und auch allen anderen am Institut, die direkt oder indirekt ein Stück zum Gelingen dieser Arbeit beigetragen haben.

Meinen Vorgesetzten im Systemunterstützungszentrum für Kampfflugzeuge danke ich dafür, sofern sie es wissentlich taten, dass sie meine Nutzung des Gleitzeit-systems zum Zwecke der Anwesenheitsminimierung auf Arbeit toleriert haben. Nur so gelang es mir regelmäßig an der Uni präsent zu sein.

Zu guter Letzt aber nicht minder geschätzt, danke ich meiner Mutter dafür, dass sie diese Arbeit zur Korrektur gelesen hat.