Maschinelles Lernen durch Funktionsrekonstruktion mit verallgemeinerten dünnen Gittern

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch–Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich–Wilhelms–Universität Bonn

vorgelegt von

Jochen Garcke

aus

Osterholz-Scharmbeck

Bonn 2004

Angefertigt mit Genehmigung der Mathematisch–Naturwissenschaftlichen Fakultät der Rheinischen Friedrich–Wilhelms–Universität Bonn

1. Referent: Prof. Dr. Michael Griebel

2. Referent: Prof. Dr. Angela Kunoth

Tag der Promotion:

The lack of information cannot be remedied by any mathematical trickery.

Lanczos, 1961

Inhaltsverzeichnis

Kapitel 1

Einleitung

Wir leben in einer modernen Informationsgesellschaft. Insbesondere computergestützte Technologien erlauben es heutzutage, riesige Datenmengen zu erfassen, zu verarbeiten und zu analysieren. Zugleich verursacht eine solche massive Präsenz von Informationen häufig Probleme dadurch, dass Menschen mit zu vielen Informationen konfrontiert werden, mit denen sie nicht umgehen können. Die besondere Schwierigkeit besteht darin, aus der Unmenge an vorhandenen Rohdaten die jeweils wichtigen Aussagen zu identifizieren. Bereits vor über zwanzig Jahren hat der Zukunftsforscher John Naisbitt zu dieser Problematik den Ausspruch „Wir ersticken in Informationen aber hungern nach Wissen“ geprägt.

Vor diesem Hintergrund wurde im letzten Jahrzehnt das so genannte *Data Mining* entwickelt und hat vielfältige Anwendungen gefunden. Die rechnerbasierte *Wissensentdeckung* soll es ermöglichen, implizit vorhandene und potentiell nützliche, aber bislang verborgene Erkenntnisse in Datenbeständen automatisiert zu finden. Ein Überblick über die verschiedenen Aufgaben, mögliche Anwendungen und Verfahren, die im Data Mining-Prozess auftreten, findet sich beispielsweise in [CPS98, BL00, HTF01, WF01, VR02].

Beispiele der vielfältigen wissenschaftlichen Data Mining-Anwendungen sind die Untersuchung von Genen mittels Mikroarray-Daten [Qua01], die Klassifizierung von Protein-Sequenzen [KKH02], die Auswertung von Beobachtungsdaten der Astrophysik, wie z.B. Bildern von Teleskopen und Observatorien [OSP+92], das Gruppieren seismischer Daten [SU02] oder die Analyse von archäologischen Ausgrabungsergebnissen [LMCP+00].

Ebenso breit wird das Data Mining in der Wirtschaft angewandt. Mit der immer weiter gehenden computergestützten Abwicklung von Geschäftsprozessen, nicht zuletzt gefördert durch die Entwicklung des Internets und des E-Kommerz, werden riesige Datenmengen gesammelt. Durch geeignete Analyse können diese für Geschäftsentscheidungen und strategische Planungen genutzt werden. Die Anwendungen umfassen unter anderem die Untersuchung des Kundenverhaltens bei der Kündigung oder Stornierung von Verträgen, die Beurteilung von Kreditrisiken, die Segmentierung von Kunden für Raten- oder Tarifplanung, die Auswahl von Personen für gezielte Werbemaßnahmen, die Aufdeckung von Betrugsversuchen, die Analyse von Aktienkursen sowie die Umsatzvorhersage, siehe hierzu [BL00] für Fallbeispiele solcher kommerzieller Anwendungen.

Die technische Basis des Auffindens von Mustern, Zusammenhängen und Trends in

großen Datenbeständen ist das *maschinelle Lernen*. Eine Auswahl wichtiger maschineller Lernmethoden umfasst dabei Neuronale Netze [Mit97, HTF01], Entscheidungsbäume [BFOS84, Mit97, CPS98], k-Nächste Nachbarn-Verfahren [Mit97, HTF01], Multivariate Adaptive Regressions Splines (MARS) [Fri91, HTF01], Clusterverfahren [Mit97, CPS98], Assoziationsverfahren [AIS93, HTF01], Support Vektor Maschinen [Vap98, SS02] und Ensemble Methoden [Die00] wie AdaBoost [FS97, ROM01] oder Random Forests [Bre01].

In dieser Arbeit werden wir mit der *Dünngitter-Kombinationstechnik* zur Funktionsrekonstruktion ein neuartiges maschinelles Lernverfahren beschreiben und untersuchen.

Der Prozess des Data Mining wird üblicherweise modellhaft im mehrere Phasen getrennt.

- Im *Planungsschritt* wird grundsätzlich untersucht, was das Ziel des Data Mining-Einsatzes auf dem vorhandenen oder erreichbaren Datenbestand sein soll beziehungsweise sein kann.

- Bei der *Aufbereitung der Daten* werden diese gesammelt und zum Teil neu erhoben. Weiterhin werden die Daten in einheitliche Formate gebracht und von offensichtlichen Fehlern bereinigt. Darüberhinaus findert eine Untersuchtung auf Ausreißer statt, das sind einzelne Datenpunkte, deren Koordinaten sich deutlich von den anderen Datenpunkten unterscheiden.

- In der Phase der *Modellbildung* wird nun das eigentliche maschinelle Lernen durchgeführt. Dabei werden je nach Anwendungsanforderung verschiedene Methoden genutzt. Die so genannte *Klassifikation*, das Sortieren der Daten in zwei oder mehrere Klassen, wird z.B. bei der Beurteilung von Kreditrisiken verwendet. Die Zerlegung von Datenbeständen in kleine, homogene und zweckmäßige Teilmengen, das so genannte *Clustering*, wird beispielsweise für die Sortierung von altertümlichen Keramikscherben nach Produktionsort genutzt. *Assoziationsanalysen* werden zur Bestimmung von Regeln angewandt, die Zusammenhänge oder Muster beschreiben, als Beispiel sei hier die Warenkorbanalyse zur Untersuchung des Einkaufsverhaltens erwähnt. Die Vorhersage von kontinuierlichen Zahlen wie dem Wert einer Immobilie wird mit *Regressionsverfahren* unternommen.

- In der *Auswertungsphase* werden die Ergebnisse des Data Mining-Verfahrens analysiert und bewertet. Bei positiver Begutachtung der erzielten Erkenntnisse werden diese in den Geschäftsprozess beziehungsweise die wissenschaftliche Untersuchung integriert. Die Ergebnisse können aber ebenso einen neuen Zyklus des Data Mining-Prozesses auslösen, da eventuell ungenügende Daten festgestellt werden oder die Ergebnisse nahe legen, bestimmte Aspekte genauer zu untersuchen.

Gewisse Formen des Wissens, das mit Data Mining-Verfahren erworben werden soll, kann durch reellwertige Funktionen von Datenmerkmalen dargestellt werden. Regression und Klassifikation gehören zu dieser Art von Anwendungen und werden in dieser Arbeit genauer untersucht.

Betrachten wir eine Menge S von *Datenpunkten* $\underline{x}$ im d-dimensionalen *Merkmalsraum*, oder *Attributsraum*, zusammen mit einer *Antwortvariable* y, d.h.

$$S = \{(\underline{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}\}_{i=1}^{M}.$$

Aus diesen *Dateninstanzen* soll ein *Vorhersagemodell* konstruiert werden, das es erlaubt, für neu gegebene Datenpunkte $\underline{x}$ geeignet die Antwortvariable y zur weiteren Entscheidung vorherzusagen. Indem wir die Dateninstanzen $(\underline{x}, y)$ als Stichproben einer Funktion auffassen, welche die Beziehung zwischen den Attributen und der Antwortvariable beschreibt, kann die Bestimmung eines Vorhersagemodells als hochdimensionale *Funktionsapproximation* beziehungsweise *Funktionsrekonstruktion* betrachtet werden, d.h.

$$y = f(\underline{x}).$$

Die binäre Klassifikation kann über stochastische Ansätze als Approximation einer Wahrscheinlichkeitsfunktion mit anschließender Fallunterscheidung aufgefasst werden. Dazu wird der kontinuierliche Erwartungswert approximiert, welcher der Klassenverteilung zu Grunde liegt. Der Funktionswert am Punkt $\underline{x}$ entspricht dabei der Wahrscheinlichkeit, dass dort die Klasse 1 vorliegt.

Aus dieser reellwertigen Wahrscheinlichkeitsfunktion wird mittels der so genannten *Bayes'schen Entscheidungsregel* der Klassifikator bestimmt, siehe hierzu beispielsweise [HTF01]. Die Entscheidung, welche Klasse der Klassifikator an einem Punkt $\underline{x}$ vorhersagt, wird mit der Vorschrift

$$f(\underline{x}) \begin{cases} \leq 0.5 & \text{ergibt Klasse 0,} \\ > 0.5 & \text{ergibt Klasse 1,} \end{cases}$$

gefällt. Welcher Klasse eine Dateninstanz $\underline{x}$ mit $f(\underline{x}) = 0.5$ zugeordnet wird, ist dabei willkürlich gewählt.

Darüberhinaus ist zu beachten, dass bei vielen Klassifikationsanwendungen das eigentliche Ziel der Anwendung nicht die strikte Einteilung in zwei Klassen ist, sondern eine geeignete Sortierung der Dateninstanzen, wobei das Sortierkriterium dabei die vorhergesagte Wahrscheinlichkeit ist, der „guten“ Klasse anzugehören.

Ein klassisches Beispiel für eine derartige Anwendung ist die Auswahl von Kunden für Werbemaßnahmen. Sofern die Werbung keinerlei Kosten verursacht, spielt die Anzahl der angeschriebenen Kunden keinerlei Rolle bei der Betrachtung des Kosten-Nutzen-Verhältnisses. Dies ist ein wesentlicher Grund für das Massenaufkommen von unerwünschten Spam-E-Mails. Das Senden einer einzelnen E-Mail verursacht praktisch keine Kosten, sofern die technische Infrastruktur für das Versenden von elektronischen Briefen vorhanden ist. Somit hat der Spammer keinen finanziellen Vorteil davon, nur einen Teil seines E-Mail-Adressenbestandes auszuwählen. Anders ist dies bei der Sendung von herkömmlichen Werbebriefen, hier treten reale Kosten wie das Drucken des Werbematerials oder die Portokosten auf. Deswegen erweist es sich für den Werbetreibenden als hilfreich, eine Auswahl der wahrscheinlichen Reagierer zu bestimmen, um die Kosten der Werbemaßnahme zu begrenzen, aber weiterhin mit Hilfe einer hohen Antwortrate viele Kundenreaktionen zu erhalten.

Nicht nur im Marketing sind solche Sortierungen der Daten nützlich. Immer wenn begrenzte Kapazitäten vorhanden sind, ist es sinnvoll, die zu bearbeitenden Daten vorzusortieren, um die Effizienz zu steigern. So werden in der Astronomie Aufnahmen des Alls automatisiert in interessant und uninteressant eingeteilt. Hier ist es sinnvoll, die Aufnahmen in der Reihenfolge anzuschauen, die das maschinelle Lernverfahren mit Hilfe der Approximation der zu Grunde liegenden Wahrscheinlichkeitsfunktion angibt.

Zur Erstellung eines konkreten Vorhersagemodells wird nun ein Satz von Funktionen, beispielsweise in Form eines Funktionenraums, vorgegeben. Das maschinelle Lernverfahren bestimmt innerhalb dieser Modellfunktionen eine nach geeigneten Kriterien beste Funktion, indem Parameter der Funktionsdarstellung auf Basis der gegebenen Daten berechnet werden.

Bei der Bestimmung einer solchen Funktion aus den gegebenen Daten treten für Data Mining-Anwendungen mehrere Arten von Schwierigkeiten auf.

- Eine Schwierigkeit besteht in der *Schlechtgestelltheit* des Problems. Schlecht gestellte Probleme haben möglicherweise keine Lösung im eigentlichen Sinne, Lösungen müssen nicht eindeutig sein und hängen weiterhin eventuell nicht stetig von den Daten ab. Ernsthafte numerische Schwierigkeiten sind die Folge. Diese Art von Problemen taucht in vielen Bereichen der Mathematik und anderen Naturwissenschaften auf und kann mit dem Konzept der *Regularisierung* [Lou89, EHN96, Kir96] behandelt werden.

- Das Problem der *Überanpassung*, der zu genauen Approximation der gegebenen Daten, die oft zu schlechten Vorhersageergebnissen auf neuen Daten führt, betrifft fast alle maschinellen Lernverfahren. Die Funktion wird aus Daten rekonstruiert, die an mehr oder weniger zufälligen Positionen einen Funktionswert liefern. Die eigentliche Anwendung besteht aber in der Auswertung auf vorher unbekannten Daten, d.h. an Koordinaten, für die oder deren unmittelbare Umgebung in den ursprünglichen Lerndaten möglicherweise keine direkten Informationen vorhanden sind. Die Funktion soll also in erster Linie nicht die gegebenen Daten gut approximieren, sondern eine gute *Verallgemeinerungsqualität* aufweisen. Es tritt hierbei in den Lernverfahren ein gewisser Widerspruch zwischen der genauen Approximation der Lerndaten und einem guten Verallgemeinerungswert der Approximation auf. Bei fast allen Techniken gibt es Parameter im Lernalgorithmus, welche die Genauigkeit der Approximation steuern. Mittels Ansätzen wie Kreuzvalidierung [Sto74, Wah90, HTF01], der L-Kurve [Han92, EHN96, Han98], Methoden aus der strukturellen Risikominimierung [Vap98, Vap00] oder anderen Ansätzen muss ein geeigneter Parametersatz bestimmt werden.

- Bei komplexen und großen Datenmengen ist eine effiziente Darstellung einer hochdimensionalen Funktion unabdingbar. Insbesondere soll die Komplexität des maschinellen Lernverfahrens zur Bestimmung der Modellparameter kontrollierbar sein. Die Komplexität hängt zum einen von der Größe der Datenmenge ab. Oft verdoppelt sich diese bei Anwendungen Jahr für Jahr und erreicht heutzutage den Terabyte-Bereich. Ein Verfahren muss also riesige und weiterhin wachsende Datenbestände

bearbeiten können. Zum anderen wirkt sich die dimensionale Struktur der Daten auf den benötigten algorithmischen Aufwand aus. Verfahren müssen mit hochdimensionalen Daten effizient umgehen können, dieses Problem wird auch als *Fluch der Dimension* bezeichnet, da die Komplexität eines Algorithmus oft exponentiell mit der Dimension der Problemstruktur wächst. Die erwähnten maschinellen Lernverfahren weisen dabei deutliche Unterschiede in ihren Komplexitätseigenschaften auf.

In dieser Arbeit untersuchen wir eine in diesem Zusammenhang neuartige Diskretisierungstechnik, die auf der Methode der dünnen Gitter nach Zenger [Zen91, BG04] basiert. Sie ermöglicht ein maschinelles Lernverfahren, welches linear in der Zahl der Datenpunkte skaliert, aber trotzdem eine nichtlineare Funktion darstellt. Allerdings ist die Zahl der behandelbaren Dimensionen bei diesem Ansatz auf Grund einer exponentiellen Abhängigkeit der Komplexität von der *effektiven Dimension* des Problems beschränkt.

Das den dünnen Gittern zu Grunde liegende Konzept ist dabei nicht neu, sondern bereits in Arbeiten aus den sechziger Jahren des vorigen Jahrhunderts zu finden [Bab60, Smo63]. Definiert werden dünne Gitter mit Hilfe einer Multiskalen-Tensorproduktbasis, die aus der eindimensionalen hierarchischen Basis konstruiert wird. Ausgehend von einem Schema von Teilräumen wird eine Diskretisierung mit einem dünnen Gitter dadurch erhalten, dass lediglich Teilräume berücksichtigt werden, deren Basisfunktionen einen Mindestbeitrag zur Lösung des Problems liefern.

Betrachten wir beispielsweise die Interpolation einer Funktion durch multilineare Basisfunktionen, so wird bei der Nutzung von dünnen Gittern mit $O(N \cdot (\log N)^{d-1})$ Punkten im Diskretisierungsprozess eine Genauigkeit von $O(N^{-2} \cdot (\log N)^{d-1})$ in Bezug auf die L_2- oder L_∞-Norm erzielt, sofern die gemischten zweiten Ableitungen beschränkt sind. Hier bezeichnet N die Anzahl der Gitterpunkte in einer Koordinatenrichtung und d die Dimension. Dieser Aufwand steht im deutlichen Gegensatz zu konventionellen Gittermethoden, die $O(N^d)$ Punkte für eine Genauigkeit von $O(N^{-2})$ benötigen, mit geringeren Voraussetzungen an die Glattheit der darzustellenden Funktion. Somit trifft der Fluch der Dimension bei Nutzung von dünnen Gittern in viel geringerem Maß zu. Die Behandlung von Problemen mit mehr als drei oder vier Dimensionen wird ermöglicht. Allerdings existiert auch beim Dünngitteransatz eine Begrenzung der Zahl der diskretisierbaren Dimensionen, denn selbst das gröbste Gitter, d.h. jenes mit zwei Punkten in jeder Koordinatenrichtung, füllt für $d > 25$ den Hauptspeicher moderner Arbeitsplatzrechner. Zwar können größere Gitter durch eine parallele Behandlung auf mehreren Rechnern behandelt werden, allerdings verdoppelt sich die Zahl der benötigten Rechner in der parallelen Umgebung mit jeder zusätzlichen Dimension.

Realisiert werden dünne Gitter in dieser Arbeit in Form der *Kombinationstechnik*, eingeführt in [GSZ92]. Dabei wird für eine gewisse Sequenz von anisotropen vollen Gittern das jeweilige Teilproblem gelöst und die so erhaltenen Ergebnisse werden zur Lösung auf dem eigentlichen dünnen Gitter geeignet zusammengefasst. Für Aufgabenstellungen, bei denen die Projektionen in die jeweiligen Teilräume vertauschbar sind, wie z.B. bei der Interpolation, kann gezeigt werden, dass die Kombinationslösung der Dünngitterlösung entspricht. Bei der numerischen Behandlung von partiellen Differentialgleichungen ist die Kombinationslösung im Allgemeinen nicht gleich der Lösung auf dem dünnen Gitter, je-

doch ist ihre Genauigkeit von der gleichen Ordnung, sofern gewisse Fehlerentwicklungen für die Teilprobleme existieren [GSZ92, BGRZ94a].

Die Beobachtung, dass der Diskretisierungsaufwand bei dünnen Gittern relativ langsam mit der Zahl der Dimensionen wächst, motiviert die Untersuchung der Möglichkeiten dieser gitterbasierten Technik im Bereich des maschinellen Lernens. Andere gitterorientierte Ansätze sind uns für dieses höherdimensionale Probleme nicht bekannt. Verfahren wie Support Vektor Maschinen oder Neuronale Netze, welche beide auch eine Funktionsdarstellung benutzen, arbeiten mit Basisfunktionen, die auf den Datenpunkten leben. Somit steigt die Anzahl der Basisfunktionen bei diesen Methoden höchstens mit der Anzahl der Dateninstanzen und nicht mit der Anzahl der Dimensionen. Allerdings weisen diese Ansätze einen nichtlinearen Aufwand bezüglich der Zahl der Dateninstanzen auf. Mit diesen Techniken ist es möglich, relativ hochdimensionale Probleme zu behandeln, die Menge von Daten ist allerdings auf Grund der Komplexität beschränkt. In vielen praktischen Anwendungen ist die Situation jedoch gerade umgekehrt. Die Dimension des resultierenden Problems ist, gegebenenfalls nach einigen Vorverarbeitungsschritten, moderat, die Menge der Daten ist aber üblicherweise sehr groß. Hier existiert ein großer Bedarf für Methoden, die auch in solchen Situationen gute Ergebnisse in kurzer Zeit liefern. Dies begründet die Untersuchung der Dünngittermethode im Rahmen des maschinellen Lernens, da sich der Aufwand bezüglich der Datenpunkte als linear erweist.

Ein Verfahren, welches Funktionen verwendet und unserer Methode in gewisser Weise ähnelt, ist der Ansatz der Multivariate Adaptive Regressions Splines (MARS) [Fri91, HTF01]. Hier sind die benutzten stückweise linearen Funktionen über die Datenpunkte definiert. Diese Verankerung geschieht dabei durch eine stückweise lineare Ansatzfunktion für jeden diskret beobachteten Wert einer Dimension, statt für jeden Datenpunkt im d-dimensionalen Raum. Diese eindimensionalen Funktionen werden wiederum per Tensorproduktansatz verknüpft. Dabei wird die Anzahl der Interaktionen, d.h. der für jede d-dimensionale Basisfunktion aktiven Dimensionen, sowohl beschränkt als auch sukzessiv während der Berechnung aufgebaut. Der Ansatz, die Zahl der Interaktionen zwischen den einzelnen Dimensionen in einer Basisfunktionen zu variieren, ist auch als ANOVA-Zerlegung (analysis of variance) [Wah90] bekannt. Hierbei wird eine Funktion allgemein in eine Summe von Funktionen zerlegt, die jeweils von einer Untermenge der Dimensionen abhängen. In [Fri91] wird bemerkt, dass vieles dafür spricht, dass bei realen Anwendungsdaten nur bis zu fünf Interaktionen zwischen den Dimensionen sinnvoll und notwendig sind.

Diese Überlegung führt zu einer Erweiterung des Dünngitteransatzes mit Hilfe von *verallgemeinerten dünnen Gittern*, die durch *dimensionsadaptive Ansätze* bestimmt werden. In der ursprünglichen Definition von dünnen Gittern [Zen91] wird mit stückweise linearen Basisfunktionen in den einzelnen Dimensionen gearbeitet, die daraus mit einem Tensorproduktansatz resultierenden d-dimensionalen multilinearen Basisfunktionen leben in allen Dimensionen. Sofern wir die zu Grunde liegende Gitterhierarchie allerdings mit der konstanten Funktion als gröbster Auflösung beginnen, ergibt sich durch die Tensorierung der Effekt, dass nun die Basisfunktionen nicht mehr in allen Dimensionen aktiv sind. Eine Tensorierung mit einer Konstanten erzeugt keine zusätzlichen Freiheitsgrade. So bleibt eine $(d-1)$-lineare Funktion nach der Verknüpfung mit einer Konstanten in der

d-ten Dimension weiterhin $(d-1)$-linear. Dies ergibt zwar mit der ursprünglichen Definition von dünnen Gittern zunächst keinerlei Vorteile. Es können allerdings verallgemeinerte dünne Gitter verwendet werden, bei denen, unter Beachtung einer Konsistenzbedingung, eine flexiblere Wahl der Teilräume möglich ist. Mit dimensionsadaptiven Methoden können nun auf Basis einer Hierarchie mit konstanten Funktionen verallgemeinerte dünne Gitter erzeugt werden, die Funktionsdarstellungen in deutlich höheren Dimensionen ermöglichen. Das Limit des dimensionsadaptiven Ansatzes stellt nicht mehr die Dimension d des Problems dar, sondern die größte notwendige Zahl von Dimensionsinteraktionen, die *effektive Dimension*. Sofern die oben erwähnte Überlegung aus [Fri91] sich in der Praxis bestätigt, können somit im Vergleich zu normalen dünnen Gittern deutlich höherdimensionale Probleme behandelt werden.

In [Heg03] wird die grundsätzliche Möglichkeit einer dimensionsadaptiven Kombinationstechnik am Beispiel von einfachen synthetischen Interpolationsproblemen vorgestellt. Für die numerischen Integration werden dimensionsadaptive Dünngittermethoden in [GG03] dargestellt. In der vorliegenden Arbeit wird aufbauend auf diesen Arbeiten eine dimensionsadaptive Kombinationstechnik zur Funktionsrekonstruktion für Klassifikations- und Regressionsprobleme im maschinellen Lernen vorgestellt.

Für unseren Ansatz zur Funktionsrekonstruktion ergibt sich durch *Tikhonov-Regularisierung* eines Approximationsproblems als konkrete Aufgabenstellung das Variationsproblem

$$R(f) \underset{f \in V}{\longrightarrow} \min !$$

mit

$$R(f) = \frac{1}{M} \sum_{i=1}^{M} (f(\underline{x}_i) - y_i)^2 + \lambda \|\nabla f\|^2.$$

Der erste Term erzwingt die Nähe der Funktion f zu den Daten, der zweite Term erzeugt eine gewisse Glattheit von f, und der Regularisierungsparameter λ gewichtet diese beiden Terme. Die Darstellung eines maschinellen Lernverfahrens als ein derartiges Regularisierungsproblem ist in der Literatur auch unter dem Begriff *Regularisierungsnetzwerk* bekannt [GJP95, EPP00]. Das Variationsproblem lösen wir in dieser Arbeit unter der Verwendung von dünnen Gittern zur Diskretisierung des Funktionenraumes.

Wie erwähnt ist die Verwendung einer Diskretisierung mit festen Gitterpunkten, wobei diese unabhängig von den Datenpunkten gewählt werden, in diesem Anwendungsgebiet neu und wird erst durch die Nutzung von dünnen Gittern realisierbar. Entscheidend für Anwendungen mit vielen Daten ist die Erkenntnis, dass der Aufwand des Verfahrens nur linear mit der Zahl der Daten skaliert, aber trotzdem eine nichtlineare Funktion berechnet wird. Dies steht im Gegensatz zu vielen anderen maschinellen Lernverfahren, deren Komplexität für eine nichtlineare Klassifikation nichtlinear mit der Zahl der Daten skaliert. Weiterhin erzeugt unser Verfahren in natürlicher Weise eine Sortierung der Dateninstanzen nach ihrer Wahrscheinlichkeit einer der beiden Klassen anzugehören. Die Anzahl der Dimensionen, die mit unserer Methode behandelt werden können, ist allerdings beschränkt, was im Vergleich zu anderen Verfahren von Nachteil ist. In vielen realen Anwendungen

kann die Dimension des Problems aber, sofern sie zu groß ist, durch Vorverarbeitungsschritte substantiell reduziert werden.

Wir wenden in dieser Arbeit das Verfahren auf eine Reihe von Benchmark-Datensätzen an, die dabei erzielten Resultate sind mit denen der besten maschinellen Lernverfahren vergleichbar. Weiterhin führen Experimente mit einer dimensionsadaptiven Kombinationstechnik durch, um deren grundsätzlichen Möglichkeiten im Bereich des maschinellen Lernens zu zeigen. Die erzielten Ergebnisse sind vielversprechend, aber das dimensionsadaptive Verfahren ist für reale Anwendungen noch nicht robust genug.

Im Weiteren gliedert sich die Arbeit in drei Bereiche:

In den folgenden zwei Kapiteln geben wir den für unseren Ansatz im Rahmen des maschinellen Lernens benötigten theoretischen Hintergrund wieder: In Kapitel 2 betrachten wir grundlegende Eigenschaften der Regularisierung von schlecht gestellten Problemen. In Kapitel 3 stellen wir zunächst Ergebnisse einiger Arbeiten vor, die sich mit den mathematischen Grundlagen des Lernens mittels Methoden der Wahrscheinlichkeitstheorie beschäftigen. Anschließend untersuchen wir die Funktionsrekonstruktion im Rahmen der Regularisierungstheorie. Eine Reihe von maschinellen Lernverfahren lassen sich durch Regularisierungsnetzwerke darstellen, diese erlauben eine Darstellung der Lösung des regularisierten Approximationsproblems in einem *Hilbertraum mit reproduzierendem Kern*. In diesem Räumen existiert eine Darstellung des Regularisierungsoperators durch seine orthonormalen Eigenvektoren und dies erlaubt die Lösung durch eine endliche Linearkombination von so genannten Kernfunktionen wiederzugeben, welche auf den Datenpunkten verankert sind. Wir stellen weiterhin eine Möglichkeit vor, diese datenorientierte Darstellung mit der einer unendlichen Funktionenraumbasis zu kombinieren. Darüberhinaus zeigen wir, wie mit geeigneten Waveletbasen, für die gewisse Normäquivalenzen gelten, eine analoge Darstellung des Regularisierungsoperators beziehungsweise der Lösung des regularisierten Problems erhalten werden kann. Für die approximative Darstellung eines Waveletkerns geben wir weiterhin Fehlerabschätzungen an.

Das von uns entwickelte Dünngitter-Lernverfahren stellen wir in den beiden anschließenden Kapiteln im Detail vor: In Kapitel 4 beschreiben wir die Methode der dünnen Gitter, die Kombinationstechnik zur Bestimmung von Funktionen auf dünnen Gittern, und schließlich verallgemeinerte dünne Gitter mit einer geeigneten dimensionsadaptiven Kombinationstechnik. In Kapitel 5 erläutern wir die numerische Realisierung des Verfahrens zur Funktionsrekonstrukion und untersuchen die Komplexität des Verfahrens bezüglich der Zahl der Attribute und der Größe der Datenmenge. Weiterhin erläutern wir numerische Vorteile, die sich durch die Verwendung einer simplizialen Diskretisierung auf den Teilgittern der Kombinationstechnik ergeben. Anschließend beschreiben wir, wie das Verfahren einfach aber effizient parallelisiert werden kann.

In Kapitel 6 schließlich betrachten wir die Qualität der Ergebnisse des vorgestellten Verfahrens anhand einer Reihe von synthetischen und realen Datensätzen. Wir untersuchen die Konvergenzeigenschaften der Dünngitterlösung gegen eine Referenzlösung auf einem hochaufgelösten vollen Gitter bei einem zweidimensionalen Regressionsbeispiel. Weiterhin präsentieren wir Resultate für eine Reihe von Benchmark-Datensätzen, geben dabei einen Überblick über die Anwendungsmöglichkeiten des Data Mining und vergleichen die Qua-

lität der Ergebnisse mit denen anderer maschineller Lernverfahren. Schließlich führen wir Experimente mit der dimensionsadaptiven Kombinationstechnik durch.

Die Arbeit schließt in Kapitel 7 mit einer Zusammenfassung der wichtigsten Ergebnisse und einem Ausblick auf weitergehend zu untersuchende Aspekte und mögliche Erweiterungen des Verfahrens.

Dank zu sagen ist immer das Schönste. An erster Stelle bedanke ich mich bei Prof. Dr. Michael Griebel für die vielen wertvollen Anregungen, Ratschläge und Diskussionen, aber auch für die Bereitstellung eines exzellenten Arbeitsumfeldes. Prof. Dr. Angela Kunoth danke ich für die Übernahme des Zweitgutachten. Auch bei Forschungsarbeiten im wissenschaftlichen Rechnen ist eine direkte Zusammenarbeit mit kommerziellen Partnern bei weitem nicht selbstverständlich. Das in dieser Arbeit entwickelte Verfahren wird allerdings bereits in Softwareprodukten der prudsys AG verwendet. Dank an dieser Stelle an Dr. Michael Theß von der prudsys AG für die anregende Zusammenarbeit. Auch bei Dr. Markus Hegland von der Australien National University in Canberra bedanke ich mich für den regen Gedankenaustausch.

Nicht zuletzt möchte ich mich bei allen Kollegen und Studenten des Bereichs Numerische Simulation in den Natur- und Ingenieurwissenschaften im Institut für Numerische Simulation für das gute Arbeitsklima bedanken, insbesondere bei meinem Zimmerkollegen Dr. Thomas Gerstner für die vielen kurzen oder längeren Diskussionen über den Schreibtisch hinweg. Bei Ralf Wildenhues bedanke ich mich für die große Hilfe beim Korrekturlesen der Arbeit. Und ohne die institutsfremden Mitglieder der Mensarunde wäre meine Arbeit auch nicht so erfolgreich und unterhaltsam gewesen.

Diese Arbeit wurde finanziert im Rahmen des BMBF-Programms „Neue Mathematische Verfahren in Industrie und Dienstleistungen“. Weitere finanzielle Unterstützung kam mir von den Sonderforschungsbereichen 256 „Nichtlineare Partielle Differentialgleichunen“ und 611 „Singuläre Phänomene und Skalierung in mathematischen Modellen“ an der Rheinischen Friedrich–Wilhelms–Universität Bonn zu. Dank auch dafür.

Bonn, im Juni 2004 Jochen Garcke

Kapitel 2

Regularisierung von schlecht gestellten Problemen

Beim maschinellen Lernen durch Funktionsrekonstruktion betrachten wir eine bestimmte Menge von Beobachtungen S bestehend aus Datenpunkten $\underline{x} \in \mathbb{R}^d$ mit zugehörigen Vorhersagevariablen $y \in \mathbb{R}$. Auf Basis dieser Informationen soll eine Funktion bestimmt werden, welche die Beziehung zwischen den Eingabe- und den Ausgabevariablen wiedergibt, d.h. $f(x) = y$. Diese Rekonstruktion einer Funktion erweist sich als *schlecht gestelltes Problem* nach Hadamard. Dabei ist neben dem eher theoretischen Aspekt der Darstellung einer bedingten Wahrscheinlichkeit in einem geeigneten Funktionenraum insbesondere die praktische Bestimmung einer Funktion auf Basis von Stichproben schlecht gestellt.

Um solche Probleme zu behandeln ist zum einen ein verallgemeinerter Lösungsbegriff notwendig um eindeutige Lösungen bestimmen zu können. Dazu werden die Begriffe der *Bestapproximation* und der *Moore-Penrose verallgemeinerten Inverse* eingeführt. Zum anderen sind *Regularisierungstechniken* erforderlich um ein stabiles Verfahren zu erhalten. Wir stellen im Weiteren zwei Regularisierungsverfahren vor, dabei folgen wir [Lou89, EHN96, Kir96], Details, Beweise und weitere Referenzen siehe dort. Zum einen die *Regularisierung durch Diskretisierung*, auch genannt Regularisierung durch Projektion, und zum anderen die *Tikhonov-Regularisierung* [Tik63]. Desweiteren betrachten wir die kombinierte Nutzung beider Regularisierungsansätze.

Es sei an dieser Stelle noch auf eine gänzlich andere Herangehensweise an Regularisierungsverfahren für schlecht gestellte Probleme verwiesen, die in [Neu98] vorgestellt wird. Während in [Tik63, Lou89, Wah90, EHN96, Kir96] Regularisierungsverfahren mit funktionalanalytischen Methoden im Hinblick auf unendlich dimensionale Probleme behandelt werden, betrachtet [Neu98] endlich dimensionale Probleme. Die Aufgabenstellung besteht somit darin, eine gute Näherung $\hat{x}$ zu einem Vektor $x \in \mathbb{R}^n$ zu finden, der eine approximative Gleichung $\mathcal{A}\hat{x} \approx y$ mit schlecht konditionierter oder singulärer Matrix $\mathcal{A} \in \mathbb{R}^{m \times n}$ bei gegebenem $y \in \mathbb{R}^m$ erfüllt. Viele Aussagen über Regularisierungsverfahren lassen sich in dieser Darstellung mit Standardtechniken der Linearen Algebra beweisen. Insbesondere Strategien zur Parameterwahl können in diesem Rahmen mit stochastischen Ansätzen hergeleitet werden.

2.1 Schlecht gestellte Probleme

Betrachten wir die Definition von *gut gestellten* und *schlecht gestellten* Problemen zurückgehend auf Hadamard.

Definition 1. *Ein Problem ist* gut gestellt*, wenn gilt:*

$$(i)\ \textit{Für alle zulässigen Daten existiert eine Lösung.} \tag{2.1}$$

$$(ii)\ \textit{Für alle zulässigen Daten ist die Lösung eindeutig.} \tag{2.2}$$

$$(iii)\ \textit{Die Lösung hängt stetig von den Daten ab.} \tag{2.3}$$

Ist eine der Bedingungen nicht erfüllt, so ist das Problem schlecht gestellt*.*

Wie wir sehen werden, kann bei der Funktionsrekonstruktion im maschinellen Lernen jeder der drei Punkte aus dieser Definition nicht erfüllt sein und somit ein schlecht gestelltes Problem erzeugen. Wir gehen in diesem Kapitel allerdings von folgender allgemeiner Problemstellung aus:

Problem 1. *Gegeben seien Hilberträume X und Y und ein linearer, stetiger Operator $\mathcal{A}: X \to Y$. Gesucht ist die Lösung der linearen Gleichung*

$$\mathcal{A}x = y. \tag{2.4}$$

Mit dieser Aufgabenstellung lassen sich die Bedingungen für ein gut gestelltes Problem formalisieren. Die Bedingung $y \in \mathsf{R}(\mathcal{A})$, wobei $\mathsf{R}(\mathcal{A})$ das Bild des Operators $\mathcal{A}$ bezeichnet, ist offensichtlich äquivalent mit (2.1). Sie wird auch y ist erreichbar genannt. Bei einem endlich dimensionalen Zielraum Y ist $y \in \mathsf{R}(\mathcal{A})$ für einen unendlich dimensionalen Ansatzraum X realisierbar, sofern sowohl der Raum X als auch die Abbildung $\mathcal{A}$ sinnvoll definiert sind. Im Fall von endlich dimensionalen diskreten Darstellungsräumen $X = V_n$ ist die Erreichbarkeit eines beliebigen $y \in Y$ allerdings nicht sichergestellt, insbesondere wenn gilt $\dim(X) \ll \dim(Y)$.

Für die Eindeutigkeit der Lösung (2.2) ist ein trivialer Nullraum der Abbildung $\mathcal{A}$, d.h. $\mathsf{N}(\mathcal{A}) = \{0\}$, eine notwendige und hinreichende Bedingung. Dies ist bei einer Abbildung von einem unendlich dimensionalen Raum in eine endlich dimensionale Menge nicht zu erfüllen. Aber auch bei endlich dimensionalen Darstellungsräumen X ist diese Bedingung im Allgemeinen nicht erfüllt.

Sind aber (2.1) und (2.2) erfüllt und existiert somit die Abbildung $\mathcal{A}^{-1}$, ist die stetige Abhängigkeit der Lösung von den Daten (2.3) äquivalent mit der Stetigkeit bzw. Beschränktheit von $\mathcal{A}^{-1}$.

Betrachten wir nun das Problem der Funktionsrekonstruktion. Gegeben sei eine Menge S von Datenpunkten $\underline{x}$ mit zugehörigen Werten y im Datenraum

$$S = \{(\underline{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}\}_{i=1}^{M}.$$

Ziel ist es, eine Funktion f aus einem über $\mathbb{R}^d$ definierten Funktionenraum V zu bestimmen, deren reellwertige Funktionsauswertungen auf den Datenpunkten $\underline{x}_i$ die zugehörigen

Werte y_i geeignet darstellen. Die Funktionsauswertung kann dabei als Projektion $\mathcal{P}$ in den Datenraum aufgefasst werden

$$\mathcal{P} : V \to \mathbb{R}^M : \quad \mathcal{P}f = \{f(\underline{x}_i)\}_{i=1}^M .$$

Da nur endlich viele Daten vorhanden sind ist der Zielraum $Y = \mathbb{R}^M$ bei unserer Anwendung somit stets endlich dimensional.

Anders verhält es sich, wenn die Elemente von S als Realisierungen einer Wahrscheinlichkeitsverteilung aufgefasst werden und es das Ziel ist, die bedingte Wahrscheinlichkeitsverteilung

$$r(\underline{x}) = \int y \; dF(y|\underline{x})$$

näherungsweise zu bestimmen. Dieser Ansatz wird in mathematischen Betrachtungen der Lerntheorie [Vap98, CS01] verfolgt. In dieser Sichtweise sind beide Hilberträume unendlich dimensional. Der Zielraum besteht nicht aus Werten auf den Punkten $\underline{x}$, sondern aus einer Funktion, welche die bedingte Wahrscheinlichkeitsverteilung beschreibt. Trotzdem entspricht wiederum der Operator $\mathcal{A}$ der Projektion $\mathcal{P}$ vom Darstellungsraum X in den Zielraum Y.

Wir haben es bei unserem Ansatz im maschinellen Lernen mit zwei verschiedenartigen schlecht gestellten Problemen zu tun. Zum einen gibt es den eher theoretischen Aspekt der Darstellung der bedingten Wahrscheinlichkeit in einem geeigneten Funktionenraum. Hierbei verursacht die nicht vorhandene stetige Abhängigkeit der Lösung von den Daten ein schlecht gestelltes Problem [Vap98]. Zum anderen ist die praktische Realisierung der Funktionrekonstrukion auf Basis von Stichproben ein schlecht gestelltes Problem. Mit diesem Aspekt beschäftigen wir uns in dieser Arbeit eingehender.

Betrachten wir nun einige der konkreten Ausprägungen von schlecht gestellten Problemen, die bei der Realisierung einer Funktionsrekonstruktion auftauchen können. Bei den zu behandelnden Daten kann es vorkommen, dass für zwei gegebene Datenpunkte $\underline{x}_i$ und $\underline{x}_j$ zwar die Koordinaten identisch sind, aber unterschiedliche Werte y_i und y_j vorliegen. Dies kann einerseits ein Fehler in den Daten sein. Andererseits kann dies aber auch eine korrekte Information sein, z.B. wenn zwei potentielle Kunden das gleiche Kundenprofil aufweisen, aber nur einer von beiden tatsächlich einkauft. Diese Situation kann durch keine Funktion dargestellt werden. Weiterhin ist offensichtlich, dass die Projektion von einem unendlich dimensionalen Funktionenraum in einen endlich dimensionalen Datenraum nicht injektiv sein kann. Somit ist also weder Existenz noch Eindeutigkeit der Lösung der Aufgabenstellung gegeben.

Den Darstellungsraum X betrachten wir im Weiteren einerseits als unendlich dimensionalen Funktionenraum V, dabei haben wir Sobolevräume H^s, Sobolevräume mit dominierender gemischter Ableitung H^s_{mix} oder auch Besovräume im Sinn. Andererseits erfolgen die letztendlichen Rechnungen natürlich in diskreten und somit endlich dimensionalen Räumen. Wir benutzen verallgemeinerte Dünngitterräume, siehe Kapitel 4, aber auch z.B. Waveletbasen [Dah97] sind denkbar. Diese unterschiedlich dimensionalen Funktionenräume wirken sich offensichtlich in verschiedenen Ausprägungen eines schlecht gestellten Problems aus.

2.1.1 Bestapproximation, Moore-Penrose verallgemeinerte Inverse

Eine eindeutige Lösung eines schlecht gestellten Problems ist wie gesehen nicht immer gegeben. Wir sind aber natürlich trotzdem daran interessiert, eine „vernünftige“ Lösung des behandelten Problems zu erhalten. Sind z.B. die gesuchten Beobachtungen y nicht erreichbar, d.h. $y \notin \mathsf{R}(\mathcal{A})$, sind wir an einem Element aus X interessiert, das die Gleichung (2.4) zumindest näherungsweise löst, d.h. $\mathcal{A}x \approx y$. Entsprechend sind wir bei der Existenz von mehreren Lösungen an einem Lösungsbegriff interessiert, der eine Lösung durch die Erfüllung weiterer Bedingungen eindeutig auszeichnet.

Zur Bestimmung einer Näherungslösung wird häufig die Methode der kleinsten Quadrate benutzt. Eine eindeutige Lösung wird durch die Definition einer Bestapproximierenden erreicht.

Definition 2. *Sei $\mathcal{A} : X \to Y$ ein beschränkter linearer Operator.*

(i) $x \in X$ *heißt* Lösung der Methode der kleinsten Quadrate *oder* Fehlerquadratlösung *von $\mathcal{A}x = y$, falls*

$$\|\mathcal{A}x - y\|_Y = \inf\{\|\mathcal{A}z - y\|_Y \quad | \quad z \in X\}. \tag{2.5}$$

(ii) $x \in X$ *heißt* Bestapproximationslösung *von $\mathcal{A}x = y$, sofern x eine Fehlerquadratlösung von $\mathcal{A}x = y$ ist und weiterhin gilt*

$$\|x\|_X = \inf\{\|z\|_X \quad | \quad z \text{ ist Fehlerquadratlösung von } \mathcal{A}x = y\}. \tag{2.6}$$

Für eine einfachere Notation verzichten wir im Folgenden auf die Kennzeichnung der Normen mit dem zugehörigen Raum X beziehungsweise Y, die verwendete Norm ergibt sich jeweils aus dem Zusammenhang.

Der Begriff der Bestapproximationslösung ist sehr eng verwandt mit der so genannten *Moore-Penrose verallgemeinerten Inversen* $\mathcal{A}^\dagger$ von $\mathcal{A}$. Genauer ist $\mathcal{A}^\dagger$ der Operator, der y auf die Bestapproximationslösung von $\mathcal{A}x = y$ abbildet, sofern diese existiert

$$A^\dagger : \mathsf{D}(A^\dagger) \left(:= \mathsf{R}(\mathcal{A}) \oplus \mathsf{R}(\mathcal{A})^\perp\right) \subset Y \to X.$$

Für eine formale Definition siehe z.B. [EHN96]. Weiterhin kann die Bestapproximationslösung durch die *Gauß-Normalengleichung* charakterisiert werden.

Satz 1. *Sei $y \in \mathsf{D}(\mathcal{A}^\dagger)$. Dann ist $x \in X$ eine Fehlerquadratlösung von $\mathcal{A}x = y$ genau dann, wenn gilt*

$$\mathcal{A}^*\mathcal{A}x = \mathcal{A}^*y. \tag{2.7}$$

Die Bestapproximationslösung $x^\dagger$ ist die Lösung von (2.7) mit minimaler Norm. D.h. es gilt

$$x^\dagger := \mathcal{A}^\dagger y = (\mathcal{A}^*\mathcal{A})^\dagger A^* y. \tag{2.8}$$

Die Menge aller Fehlerquadratlösungen ist charakterisiert durch $x^\dagger + \mathsf{N}(\mathcal{A})$.

Für die Bestapproximationslösung $x^\dagger$ und die verallgemeinerten Inversen $\mathcal{A}^\dagger$ lässt sich einfach zeigen [Lou89, EHN96]:

Bemerkung 1. *Für $\mathcal{A} \in L(X,Y)$ gilt*

- $x^\dagger$ *eindeutig in* $\mathsf{R}(\mathcal{A}^*)$.
- $\mathsf{N}(\mathcal{A}^\dagger) = \mathsf{R}(\mathcal{A})^\perp$.
- $\mathsf{R}(\mathcal{A}^\dagger) = \mathsf{N}(\mathcal{A})^\perp = \overline{\mathsf{R}(\mathcal{A}^*)}$.
- $\mathcal{A}^\dagger$ *ist linear.*

2.1.2 Kompakte lineare Operatoren und deren Singulärwerterweiterung

Betrachten wir nun kompakte lineare Operatoren $\mathcal{A} : X \to Y$ und deren Adjungierte $\mathcal{A}^* : Y \to X$. Probleme mit solch kompakten linearen Operatoren sind von besonderem Interesse, z.B. sind Integraloperatoren unter geeigneten Voraussetzungen kompakt. Aber auch bei dem von uns betrachteten Approximationsproblem ist die Projektion in den Datenraum ein kompakter Operator, da das Bild $\mathsf{R}(\mathcal{A})$ eine beschränkte Untermenge des R^M ist. Für derartige Operatoren führen wir nun den Begriff der *Singulärwerterweiterung* ein, der Aussagen über Eigenschaften der Moore-Penrose-Inversen $\mathcal{A}^\dagger$ ermöglicht.

Seien $\{\sigma_n^2\}_{n\in\mathbb{N}}$ die Eigenwerte des selbstadjungierten Operators $\mathcal{A}^*\mathcal{A}$, wie auch von $\mathcal{A}\mathcal{A}^*$, in absteigender Reihenfolge mit Häufigkeit abgezählt, es gelte $\sigma_n > 0$. Die Vektoren $\{v_n\}_{n\in\mathbb{N}}$ bilden ein dazugehöriges Orthonormalsystem von Eigenvektoren von $\mathcal{A}^*\mathcal{A}$, welches $\overline{\mathsf{R}(\mathcal{A}^*)} = \overline{\mathsf{R}(\mathcal{A}^*\mathcal{A})}$ aufspannt. Wir definieren die Vektoren $\{u_n\}_{n\in\mathbb{N}}$ durch

$$u_n := \frac{\mathcal{A}v_n}{\|\mathcal{A}v_n\|}.$$

Sie bilden ein Orthonormalsystem von Eigenvektoren von $\mathcal{A}\mathcal{A}^*$ und spannen $\overline{\mathsf{R}(\mathcal{A})} = \overline{\mathsf{R}(\mathcal{A}\mathcal{A}^*)}$ auf. Weiterhin gelten die folgenden Aussagen:

$$\begin{aligned}
\mathcal{A}v_n &= \sigma_n u_n, \\
\mathcal{A}^*u_n &= \sigma_n v_n, \\
\mathcal{A}x &= \sum_{n=1}^{\infty} \sigma_n \langle x, v_n\rangle u_n, \quad x \in X, \\
\mathcal{A}^*y &= \sum_{n=1}^{\infty} \sigma_n \langle y, u_n\rangle v_n, \quad y \in Y,
\end{aligned}$$

wobei diese unendlichen Reihen in den Hilbertraumnormen von X bzw. Y konvergieren. Die unendlichen Reihen werden *Singulärwerterweiterung* genannt und sind das unendlich dimensionale Analogon zur wohlbekannten *Singulärwertzerlegung* einer Matrix.

Es zeigt sich, dass $\mathcal{A}$ genau dann ein endlich dimensionales Bild hat, wenn $\mathcal{A}$ endlich viele *Singulärwerte* σ_n besitzt. Somit degenerieren in diesem Fall alle unendlichen Summen

zu endlichen Summen. Weiterhin gilt, dass das Bild $\mathsf{R}(\mathcal{A})$ genau dann abgeschlossen ist, wenn es endlich dimensional ist, d.h. im Allgemeinen ist $\mathsf{R}(\mathcal{A})$ nicht abgeschlossen. Mit anderen Worten:

Proposition 1. *Sei $\mathcal{A} : X \to Y$ kompakt, $\dim \mathsf{R}(\mathcal{A}) = \infty$. Dann ist $\mathcal{A}^\dagger$ ein dicht definierter unbeschränkter linearer Operator mit abgeschlossenem Graphen.*

Somit hängt für einen kompakten linearen Operator mit nicht abgeschlossenem Bild die Bestapproximation von $\mathcal{A}x = y$ nicht stetig von der rechten Seite ab, das Problem ist schlecht gestellt. Dieses gilt z.B. für nicht degenerierte Integraloperatoren. Allerdings tritt diese Schwierigkeit bei der von uns betrachteten Funktionsrekonstruktion auf Grund des endlich dimensionalen Bildraums nicht auf.

Die Moore-Penrose-Inverse eines kompakten Operators kann nun mittels eines singulären Systems durch eine Reihe dargestellt werden.

Satz 2. *Sei $(\sigma_n; v_n, u_n)$ ein singuläres System für den kompakten linearen Operator $\mathcal{A}$ und $y \in Y$. Dann gilt:*

(i)

$$y \in \mathsf{D}(\mathcal{A}^\dagger) \Longleftrightarrow \sum_{n=1}^{\infty} \frac{|\langle y, u_n \rangle|^2}{\sigma_n^2} < \infty. \tag{2.9}$$

(ii) Für $y \in \mathsf{D}(\mathcal{A}^\dagger)$ gilt

$$\mathcal{A}^\dagger y = \sum_{n=1}^{\infty} \frac{\langle y, u_n \rangle}{\sigma_n} v_n. \tag{2.10}$$

Die Bedingung (i) in Satz 2 wird *Picard-Bedingung* genannt. Sie besagt, dass eine Bestapproximationslösung von $\mathcal{A}x = y$ nur existiert, falls die (verallgemeinerten) Fourier-Koeffizienten $\langle y, u_n \rangle$ bezüglich dem *Singulärvektor* u_n schnell genug relativ zum Singulärwert σ_n fallen.

Weiterhin haben lineare Systeme der Form $\mathcal{A}x = y$, auch im Fall von endlich dimensionalen X und Y, folgende Eigenschaften [Han01]:

- Die Singulärwerte von $\mathcal{A}$ häufen sich bei Null. Riesige Konditionszahlen sind die Folge.
- Die exakte rechte Seite y erfüllt die *diskrete Picard-Bedingung*, d.h. in der Singulärwertzerlegung ist die Größe einer Kompenente von y in Richtung eines Singulärvektors von $\mathcal{A}$ verknüpft mit einer Art von Handarbeitszeug, das durch die Verknüpfung verknüpft ist mit der Größe des dazugehörigen Singulärwertes. Dies bedeutet, dass trotz der riesigen Konditionszahlen das lineare System letztendlich *effektiv gut konditioniert* ist [CF88].
- In der Realität hat man allerdings mit Störungen der Daten zu tun die alle Komponenten von y gleich betreffen, siehe auch [BCCI98]. Hier zeigt (2.10), wie Fehler in y

das Ergebnis $\mathcal{A}^\dagger y$ beeinflussen. Fehlerkomponenten (in Bezug auf die Basis $\{u_n\}$), die zu großen Singulärwerten gehören, sind zu vernachlässigen, nicht aber Fehlerkomponenten, die mit kleinen Singulärwerten korrespondieren; deren Fehler wird mit dem nun großen Faktor $1/\sigma_n$ verstärkt.

Im endlich dimensionalen Fall ist der Faktor $1/\sigma_n$ zwar beschränkt, kann aber je nach Anwendung groß werden. Letztendlich kann eine stabile Approximation der Lösung $x^\dagger$ nur mit einer geeigneten Modifizierung des Problems in Form einer so genannten Regularisierung erreicht werden.

2.2 Regularisierung schlecht gestellter Probleme

Unter *Regularisierung* versteht man die Approximation eines schlecht gestellten Problems durch eine parameterabhängige Familie von ähnlichen, gut gestellten Problemen. Wenn wir nun die Bestapproximationslösung $x^\dagger = \mathcal{A}^\dagger y$ für eine gegebene rechte Seite $y \in D(\mathcal{A}^\dagger)$ bestimmen wollen kann in der Praxis nicht davon ausgegangen werden, dass die exakte rechte Seite bekannt ist. Es sind meist nur mit Fehlern behaftete Daten y^δ vorhanden, für die gelte

$$\|y^\delta - y\| \leq \delta, \tag{2.11}$$

wobei δ die Größe der Störung angibt.

Wie im vorherigen Abschnitt erläutert, ist bei schlecht gestellten Problemen im Allgemeinen $\mathcal{A}^\dagger y^\delta$ keine gute Approximation von $\mathcal{A}^\dagger y$. Im Fall $y^\delta \notin D(\mathcal{A}^\dagger)$ ist nicht einmal die Existenz von $x^\dagger$ gegeben. Wir suchen nun nach einer Approximation von $x^\dagger$, bezeichnet mit x_λ^δ, die zum einen stetig von den (gestörten) Daten abhängt und somit stabil berechnet werden kann. Zum anderen soll x_λ^δ mit kleiner werdender Störung δ und geeignet gewähltem *Regularisierungsparameter* λ gegen die gesuchte Lösung $x^\dagger$ streben.

Definition 3. *Sei $\mathcal{A} : X \to Y$ ein beschränkter linearer Operator. Ein* Regularisierungsverfahren *von $\mathcal{A}^\dagger$ ist eine Familie von stetigen Operatoren, genannt* Regularisierungsoperatoren*,*

$$\{\mathcal{A}_\lambda^\dagger\}_{\lambda>0},\ \mathcal{A}_\lambda^\dagger : Y \to X,$$

mit folgender Eigenschaft: Es existiert eine Abbildung $\lambda : \mathbb{R}_+ \times Y \to \mathbb{R}_+$, so dass für alle $y \in \mathsf{D}(\mathcal{A}^\dagger)$ gilt

$$\limsup_{\delta\to 0}\{\|\mathcal{A}_{\lambda(\delta,y^\delta)}^\dagger y^\delta - \mathcal{A}^\dagger y\| \mid y^\delta \in Y,\ \|y - y^\delta\| \leq \delta\} = 0. \tag{2.12}$$

Der Regularisierungsparameter *λ wird so gewählt, dass gilt*

$$\limsup_{\delta\to 0}\{\lambda(\delta, y^\delta) \mid y^\delta \in Y,\ \|y - y^\delta\| \leq \delta\} = 0. \tag{2.13}$$

Hängt λ nicht von y^δ ab, so sprechen wir von einer a-priori Parameterwahl, andernfalls von einer a-posteriori Parameterwahl.

Sofern (2.12) und (2.13) für ein bestimmtes $y \in \mathsf{D}(\mathcal{A}^\dagger)$ gelten, wird ein Paar $(\mathcal{A}_\lambda^\dagger, \lambda)$ Regularisierungsmethode zur Lösung von $\mathcal{A}x = y$ genannt.

Mit der Beschränkung der Größe des Datenfehlers (2.11) ergibt sich nun die folgende grundlegende Fehlerdarstellung [Kir96, EHN96] eines Regularisierungsverfahrens:

$$\|x_\lambda^\delta - x^\dagger\| \;\leq\; \|\mathcal{A}_\lambda^\dagger\|\,\delta + \|\mathcal{A}_\lambda^\dagger y - x^\dagger\|. \tag{2.14}$$

Der Fehler zwischen der Bestapproximierenden $x^\dagger$ und der Lösung x_λ^δ des regularisierten Verfahrens spaltet sich somit in zwei Terme auf. Der Erste bezieht sich auf *Datenfehler*, auch *Messfehler* genannt. Der zweite Term stellt den *Regularisierungs-* oder *Approximationsfehler* für exakte Daten dar. In der Stochastik werden diese beiden Terme mit Varianz und Bias assoziiert [CS01]. Der Datenfehler korrespondiert größtenteils zur Varianz, kann aber auch Bias enthalten. Im Gegensatz dazu enthält der Regularisierungsfehler fast nur Bias [BBDM02].

Es stellt sich somit die Frage, wie die Parameter zu wählen sind, um die Summe aus Daten- und Regularisierungsfehler in (2.14) zu minimieren. Ein optimaler Regularisierungsparameter λ lässt sich bei bekanntem Fehler δ und bekannter *Glattheit* der Lösung bestimmen [Neu98], allerdings sind diese Werte in realen Anwendungen meist nicht bekannt oder nur sehr aufwendig zu schätzen. A-priori Strategien sind somit meist nicht praxisrelevant, spielen aber eine wichtige Rolle in der theoretischen Untersuchung.

Die bekannteste a-posteriori Strategie basiert auf Morozovs *Diskrepanzprinzip* [Mor84, EHN96, Kir96], andere Ansätze sind (generalisierte) *Kreuzvalidierung* [Sto74, Wah90, HTF01] oder die *L-Kurve* [Han92, EHN96, Han98].

2.2.1 Regularisierung durch Diskretisierung

Betrachten wir nun die numerische Realisierung eines Regularisierungsverfahrens. Dabei werden Methoden untersucht, die in endlich dimensionalen Räumen realisiert werden können. Eine Möglichkeit ist hierbei die so genannte Regularisierung durch Projektion, dabei wird eine Regularisierung allein durch die endlich dimensionale Approximation erreicht. Beispiele sind Diskretisierung, Kollokation, Galerkin oder Ritz Approximation. Eine allgemeine mathematische Theorie für diesen Ansatz wurde in [Nat77] vorgestellt, siehe auch [EHN96, Kir96].

Bei der Regularisierung durch Projektion wird das Problem (2.4) in endlich dimensionale Unterräume

$$X_1 \subset X_2 \subset X_3 \subset \cdots \subset X \text{ und } Y_1 \subset Y_2 \subset Y_3 \subset \cdots \subset Y,$$

mit $\dim(X_n) = \dim(Y_n)$ projiziert. Seien $\mathcal{P} : X \to X_n,\ \mathcal{Q}_n : Y \to Y_n$ geeignete Projektionen auf X_n, Y_n. Eine durch die Projektionsmethode bestimmte regularisierte Lösung $x_n^\dagger \in X_n$ von $\mathcal{A}x = y$ ist nun eindeutig charakterisiert durch die Erfüllung von

$$(\mathcal{A}x_n^\dagger, z_n) = (y, z_n) \quad \forall z_n \in Y_n. \tag{2.15}$$

Dabei wird von der Projektionsmethode gesprochen, die durch X_n und $\mathcal{Q}_n$ generiert wird. Dieser Ansatz ist unter den Namen *Petrov-Galerkin-Verfahren* bekannt. Im Falle $X_n = Y_n$ spricht man vom Galerkin-Verfahren. Sofern der Operator $\mathcal{A}$ selbstadjungiert und positiv definit ist, entspricht dieser Ansatz der *Rayleigh-Ritz-Methode*. Der Index n kann hierbei

beispielsweise mit der Maschenweite $h \sim \frac{1}{2^n}$ bei einer uniformen Diskretisierung identifiziert werden.

Bezeichnen wir mit $\mathcal{A}_n^\dagger$ analog wie vorher den Operator, der einem $y \in Y$ die regularisierte Bestapproximationslösung zuordnet. Sofern die Galerkingleichung (2.15) eine eindeutige Lösung hat, kann die folgende Fehlerabschätzung gezeigt werden [EHN96, Kir96]:

$$\|x_n^\delta - x^\dagger\| \leq \|\mathcal{A}_n^\dagger\|\,\delta + \|\mathcal{A}_n^\dagger y - x^\dagger\| \tag{2.16}$$

$$\leq \frac{\delta}{\mu_n} + \|\mathcal{A}_n^\dagger y - x^\dagger\| \tag{2.17}$$

$$= \frac{\delta}{\mu_n} + \|x_n^\dagger - x^\dagger\|, \tag{2.18}$$

wobei in dieser Form für den Fehler der rechten Seite die Voraussetzung $\|\mathcal{Q}_n(y - y^\delta)\| \leq \delta$ gilt. μ_n bezeichnet hierbei den kleinsten Singulärwert von $\mathcal{A}_n := \mathcal{Q}_n\mathcal{A}$. Für eine entsprechende Abschätzung mit $\|y - y^\delta\| \leq \delta$ siehe [Kir96].

Die Fehlerabschätzung (2.17) weist nun die gleiche Struktur auf wie (2.14). Der erste Term bezeichnet den Datenfehler, der zweite Term den Regularisierungsfehler. Bei der Regularisierung durch Diskretisierung hat der Index n die Rolle des Regularisierungsparameters $\lambda = \frac{1}{n}$ übernommen. Da $\mu_n \to 0$ für $n \to \infty$, ist es naheliegend, das Problem eher grob zu diskretisieren, um den Datenfehler zu beschränken. Andererseits bleibt bei einer groben Diskretisierung der Regularisierungsfehler tendenziell groß. Offensichtlich ist wiederum eine Balancierung der beiden Fehlerterme zu erreichen. Eine a-posteriori Regularisierungsstrategie von optimaler Ordnung wird in [Kal00] vorgestellt, siehe auch [BP03].

Es sei bemerkt, dass hier durch die spezielle Wahl $Y_n := \mathcal{A}(X_n)$ die *Fehlerquadratmethode* erhalten wird. Die Lösung $x_n \in X_n$ ist dabei charakterisiert durch

$$(\mathcal{A}x_n, \mathcal{A}z_n) = (y, \mathcal{A}z_n) \quad \forall z_n \in X_n.$$

Allerdings ist ohne zusätzliche Annahmen nicht sichergestellt, dass x_n gegen $x^\dagger$ konvergiert [Sei80, EHN96, Kir96].

Eine andere Projektionsmethode ist die in [Nat77] vorgestellte *duale Fehlerquadratmethode*. Sie ergibt sich durch die spezielle Wahl $X_n := \mathcal{A}^*Y_n$. Bei diesem Verfahren ist die Konvergenz $x_n \to x^\dagger$ für $n \to \infty$ garantiert. Die Lösung ist gegeben durch $x_n := \mathcal{A}^*u_n$, wobei $u_n \in Y_n$ charakterisiert ist über

$$(\mathcal{A}\mathcal{A}^*u_n, z_n) = (y, z_n) \quad \forall z_n \in Y_n.$$

Für festes n stellt sich die Frage, wie Y_n gewählt werden soll, damit der kleinste Singulärwert μ_n maximiert wird. Damit würde nach Abschätzung (2.17) der Datenfehler in diesem Sinne minimiert. Für kompakte Operatoren $\mathcal{A}$ gilt nun [EHN96]

Proposition 2. *Set $\mathcal{A}$ kompakt mit singulärem System $(\sigma_n; v_n, u_n)$ und sei Y_n derart, dass* $\dim(Y_n) = n$. *Dann gilt*

$$\mu_n \leq \sigma_n,$$

wobei μ_n wiederum den kleinsten Singulärwert von $\mathcal{A}_n := \mathcal{Q}_n\mathcal{A}$ bezeichnet.

Gleichheit gilt hier für die Wahl $Y_n = \mathcal{U}_n := \text{span}\{u_1, \ldots, u_n\}$. Die daraus resultierende Methode ist die trunkierte Singulärwertzerlegung. Es lässt sich weiterhin zeigen, dass die Wahl $Y_n = \mathcal{U}_n$ auch bezüglich des Approximationsfehlers optimal ist [EHN96]:

Proposition 3. *Set $\mathcal{A}$ kompakt mit singulärem System $(\sigma_n; v_n, u_n)$ und sei Y_n derart, dass* $\dim(Y_n) = n$. *Dann gilt*

$$\|(\mathcal{I} - \mathcal{P}_n)\mathcal{A}^*\| \geq \sigma_{n+1}.$$

Falls $Y_n = \mathcal{U}_n$, so gilt Gleichheit.

Damit ist die Konvergenzrate

$$\|x_n^\delta - x^\dagger\| \leq O(\frac{\delta}{\sigma_n} + \sigma_{n+1})$$

die bestmögliche Rate, die man für kompakte Operatoren $\mathcal{A}$ und $x^\dagger \in R(\mathcal{A}^*)$ erwarten kann. Sie wird bei der trunkierten Singulärwertzerlegung angenommen.

2.2.2 Tikhonov-Regularisierung

Im Folgenden betrachten wir mit der *Tikhonov-Regularisierung* [Tik63] ein weit verbreitetes Regularisierungsverfahren, welches auch unter dem Namen *Tikhonov-Phillips-Regularisierung* [Phi62] bekannt ist. Wir stellen zunächst eine von Tikhonov vorgeschlagene Verallgemeinerung des Begriffes der Gutgestellheit nach Hadamard vor. Dabei ist die Einschränkung des Definitionsgebietes des betrachteten Operators $\mathcal{A}$ die Grundidee der Gutgestelltheit im Tikhonov-Sinne [Tik43, TA77, Mor84].

Definition 4. *Das Problem (2.4) ist* gut gestellt im Tikhonov-Sinn*, oder auch* bedingt gut gestellt*, wenn gilt:*

(i) Es ist a-priori bekannt, dass eine Lösung x für alle $y \in Y' \subset Y$ existiert und es gilt darüberhinaus $x \in V \subset \mathsf{D}(\mathcal{A})$.

(ii) Die Lösung x ist eindeutig in V.

(iii) Infinitesimal kleine Veränderungen der Lösung x korrespondieren zu infinitesimal kleinen Veränderungen der rechten Seite y, wobei weiterhin $x \in V$ gilt.

Die Menge V wird hierbei Korrektheitsmenge *genannt.*

Der Weg zur Tikhonov-Regularisierung führt nun über eine geeignete Einschränkung des Suchraumes. Sei $V \subset X$, genauer gelte

$$V := f \in X : \Phi(f) \leq \rho^2,$$

wobei $\Phi : X \to \mathbb{R}$ eine nichtnegative Funktion ist, die auf $\mathsf{N}(\mathcal{A})$ strikt konvex ist. Die gesuchte Lösung liege in V, d.h. wir minimieren nun über Elemente aus V

$$\inf_{x \in V} \|Ax - y\|.$$

Aus diesem restringierten Minimierungsproblem erhalten wir nun mit Hilfe eines Lagrangeschen Multiplikators ein unrestringiertes Problem für das *Tikhonov-Funktional*

$$\|\mathcal{A}x - y\|^2 + \lambda\Phi(x).$$

$\Phi(f)$ wird *Strafterm* oder *Regularisierungsoperator* genannt. Tikhonov [Tik63] benutzte ursprünglich $\Phi(x) = \|x\|^2$, d.h. die natürliche Norm des Raumes X, und Phillips [Phi62] verwendete $\Phi(x) = \|x''\|^2$. Sofern Normen wie z.B. Sobolevnormen für $\Phi(x)$ benutzt werden, spricht man auch von *Glättungsoperatoren*, da die Lösungen nun bestimme Glattheitseigenschaften erfüllen müssen.

Im Weiteren betrachten wir nur Operatoren der Form

$$\Phi(x) = \|\mathcal{S}x\|^2, \tag{2.19}$$

wobei $\mathcal{S} : \mathsf{N}(\mathcal{A})^\perp \to X$ mit $\mathsf{D}(\mathcal{S})$ dicht in $\mathsf{N}(\mathcal{A})^\perp$ und $(\mathcal{S}^*\mathcal{S})^{-1} : \mathsf{N}(\mathcal{A})^\perp \to X$ stetig sei, d.h. es existiere $\beta > 0$ mit

$$\|\mathcal{S}x\| \geq \beta\|x\|.$$

Das zu minimierende Tikhonov-Funktional ist somit

$$\|\mathcal{A}x - y\|^2 + \lambda\|\mathcal{S}x\|^2. \tag{2.20}$$

Sofern $\mathcal{S} \neq \mathcal{I}$, also nicht die natürliche Norm des Raums X in (2.20) auftritt, wird auch von einer gewichteten verallgemeinerten Inversen $\mathcal{A}^\dagger_\mathcal{S}$ gesprochen. Eine notwendige Bedingung in diesem Zusammenhang ist die so genannte *Komplementbedingung* nach Morozov [Mor84]

$$\exists\gamma > 0 \quad \text{mit} \quad \|\mathcal{A}x\|^2 + \|\mathcal{S}x\|^2 \geq \gamma\|x\|^2, \quad x \in \mathsf{D}(\mathcal{S}) \subset X, \tag{2.21}$$

sie verallgemeinert die Bedingung (2.19) an die Operatoren $\mathcal{S}$. Notwendig hierfür ist

$$\mathsf{N}(\mathcal{A}) \cap \mathsf{N}(\mathcal{S}) = \{0\},$$

d.h. die Nullräume von $\mathcal{A}$ und $\mathcal{S}$ schneiden sich nur trivial. Andererseits ist eine hinreichende Bedingung gegeben durch

$$\dim \mathsf{N}(\mathcal{S}) < \infty, \quad \mathsf{N}(\mathcal{A}) \cap \mathsf{N}(\mathcal{S}) = \{0\}. \tag{2.22}$$

Diese Bedingung ist für Differentialoperatoren $\mathcal{S}$ in $\mathbb{R}^d$ erfüllt, siehe hierzu auch [LP80, EHN96]. Es lässt sich nun zeigen

Satz 3. *Das minimierende Element x_λ von (2.20) löst die regularisierte Normalengleichung*

$$(\mathcal{A}^*\mathcal{A} + \lambda\mathcal{S}^*\mathcal{S})x_\lambda = \mathcal{A}^*y.$$

Weiterhin ist

$$\mathcal{A}^\dagger_\lambda = (\mathcal{A}^*\mathcal{A} + \lambda\mathcal{S}^*\mathcal{S})^{-1}\mathcal{A}^* \tag{2.23}$$

ein lineares Regularisierungsverfahren.

Während bei schlecht gestellten Problemen die Eigenwerte von $\mathcal{A}$ gegen Null gehen, sind die Eigenwerte von $\mathcal{A}^*\mathcal{A} + \lambda\mathcal{S}^*\mathcal{S}$ für $\lambda > 0$ von Null weg beschränkt. Im Hinblick auf Abschätzung (2.14) kann für kompakte Operatoren gezeigt werden, dass $\|\mathcal{A}_\lambda^\dagger\| \leq \frac{1}{2\sqrt{\lambda}}$ gilt [Lou89, EHN96, Kir96]. Für die Tikhonov-Regularisierung gibt es eine breite Theorie unter anderem zu Bedingungen an ihre Ordnungsoptimalität und über die Saturierung der Konvergenzordnung, siehe wiederum [Lou89, EHN96, Kir96] und die dortigen Referenzen.

2.2.3 Kombinierte Regularisierung

Bei der numerischen Realisierung der Tikhonov-Regularisierung muss der Raum X durch einen endlich dimensionalen Unterraum X_n geeignet approximiert werden. Je größer der diskrete Raum X_n, desto genauer kann ein Element des Raumes X dargestellt werden. Andererseits zeigt für den Fall von fehlerbehafteten Daten und einem sehr schlecht gestellten Problem die Abschätzung (2.17), dass bei der Anwendung einer Regularisierung durch Projektion die Dimension des Unterraums X_n klein gehalten werden muss, damit der Gesamtfehler klein bleibt. Für diese Probleme fällt der kleinste Singulärwert $\sigma_n(\mathcal{A}_n)$ schnell für größer werdendes n. Um größere Approximationsräume zu nutzen, kann die Regularisierung durch Diskretisierung mit einer zusätzlichen Regularisierungsmethode gekoppelt werden.

Betrachten wir also eine diskrete Version der Tikhonov-Regularisierung. Hier muss nun das Funktional

$$\|\mathcal{A}_n x - y\|^2 + \lambda\|\mathcal{S}x\|^2.$$

minimiert werden. Die Lösung dieser Gleichung ist dabei gegeben durch

$$x_{\lambda,n}^\delta = (\mathcal{A}_n^*\mathcal{A}_n + \lambda\mathcal{S}_n^*\mathcal{S}_n)^{-1}\mathcal{A}_n^* y_n^\delta =: \mathcal{A}_{\lambda,n}^\dagger y_n^\delta.$$

$\mathcal{A}_\lambda^\dagger$ aus (2.23) ist beschränkt durch $\frac{1}{2\sqrt{\lambda}}$, was auch für das projizierte System unabhängig von n gelten muss, d.h. es gilt ebenso

$$\|\mathcal{A}_{\lambda,n}^\dagger\| \leq \frac{1}{2\sqrt{\lambda}}. \tag{2.24}$$

Andererseits gilt

$$\|\mathcal{A}_{\lambda,n}^\dagger\| \leq \|\mathcal{A}_{0,n}^\dagger\| \leq \frac{1}{\sigma_n} \quad \forall\lambda \geq 0. \tag{2.25}$$

Da die Abschätzungen (2.24) und (2.25) gleichzeitig gelten, wissen wir somit

$$\|\mathcal{A}_{\lambda,n}^\dagger\| \leq \min\left(\frac{1}{2\sqrt{\lambda}}, \frac{1}{\sigma_n}\right). \tag{2.26}$$

Sofern n so gewählt wird, dass $\sigma_n > 2\sqrt{\lambda}$ gilt, ist die Diskretisierung der bestimmende Regularisierer und die Lösung von λ relativ unabhängig. Andernfalls gilt für $2\sqrt{\lambda} > \sigma_n$, dass die Stabilisierung hauptsächlich durch λ und somit den Glättungsterm erfolgt und die Diskretisierung relativ groß gewählt werden kann [PV90, Rie97, BBDM02].

Ein weiterer Aspekt ist die Beobachtung, dass die Tikhonov-Regularisierung immer auf das ganze Gebiet wirkt, wohingegen eine Diskretisierung für bestimmte Gebiete mit

einer feineren Auflösung arbeiten kann. Der Regularisierungsterm λ kann somit als globaler Parameter angesehen werden. Die Diskretisierungsauflösung n kann sowohl global als auch lokal regularisieren kann, abhängig von der tatsächlichen Wahl der diskreten Räume. Sofern durch ein großes λ die Tikhonov-Regularisierung dominiert, wird die Lösung global verändert obwohl die Diskretisierung das Problem eher lokal statt global auflöst [BBDM02].

Insgesamt haben wir in diesem Kapitel eine kompakte Einführung in die Theorie der Regularisierung von schlecht gestellten Problemen gegeben. Als Verfahren haben wir die Regularisierung durch Diskretisierung und die Tikhonov-Regularisierung vorgestellt. Unter Gesichtspunkten der praktischen Realisierung dieser Regularisierungsmethoden haben wir deren gekoppelte Anwendung betrachtet und eine Wechselwirkung zwischen Regularisierungsparameter und der Auflösung der Diskretisierung festgestellt.

Die bisher vorgestellten Erkenntnisse bilden den theoretischen Hintergrund der im folgenden Kapitel betrachteten Funktionsrekonstruktion und deren Regularisierung.

Kapitel 3

Funktionsrekonstruktion und deren Regularisierung

In diesem Kapitel betrachten wir die Funktionsrekonstruktion als schlecht gestelltes Problem und deren Regularisierung. Dabei sei wie bisher eine Menge S von Datenpunkten $\underline{x}$ im d-dimensionalen Merkmalsraum zusammen mit einer Antwortvariable y gegeben, d.h.

$$S = \{(\underline{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}\}_{i=1}^M.$$

Aus diesen Dateninstanzen soll ein Vorhersagemodell konstruiert werden, das es erlaubt, für neu gegebene Datenpunkte $\underline{x}$ die Antwortvariable y zur weiteren Entscheidung vorherzusagen. Indem wir die Dateninstanzen $(\underline{x}, y)$ als Stichproben einer Funktion auffassen, welche die Beziehung zwischen den Attributen und der Antwortvariable beschreibt, kann die Bestimmung eines Vorhersagemodells als hochdimensionale Funktionsapproximation beziehungsweise Funktionsrekonstruktion betrachtet werden, d.h.

$$y = f(\underline{x}).$$

Zur Erstellung eines Vorhersagemodells wird nun ein Satz von Funktionen vorgegeben, beispielsweise in Form eines Funktionenraums. Das maschinelle Lernverfahren bestimmt innerhalb dieser Modellfunktionen eine nach geeigneten Kriterien beste Funktion, indem Parameter der Funktionsdarstellung auf Basis der gegebenen Daten berechnet werden.

Da die Rekonstruktion einer Funktion aus gegebenen Daten ein schlecht gestelltes Problem ist, wird sie mit dem im letzten Kapitel vorgestellten Konzept der Regularisierung behandelt. Die Regularisierungstheorie nach Tikhonov [Tik63, TA77, Wah90, EHN96, Kir96] fordert dabei zusätzliche Glattheitsbedingungen an die Lösung des Approximationsproblems und betrachtet folglich das Variationsproblem

$$R(f) \underset{f \in V}{\longrightarrow} \min\ ! \tag{3.1}$$

mit

$$R(f) = \frac{1}{M} \sum_{i=1}^{M} C(f(\underline{x}_i), y_i) + \lambda \Phi(f). \tag{3.2}$$

Dabei beschreibt $C(\cdot,\cdot)$ eine Fehlerkostenfunktion, die den Interpolationsfehler misst, und $\Phi(f)$ ist ein Glattheitsfunktional, das für $f \in V$ wohldefiniert ist. Der erste Term erzeugt die Nähe von f zu den Daten, der zweite Term erzwingt die Glattheit der Funktion f, und der Regularisierungsparameter λ balanciert diese beiden Ausdrücke. Ein Term $\Phi(f)$ wird oft auch bei der praktischen Umsetzung einer Regularisierung durch Diskretisierung zur Stabilisierung des numerischen Verfahrens benutzt. Typische Beispiele sind

$$C(x,y) = |x-y| \quad \text{oder} \quad C(x,y) = (x-y)^2,$$

und

$$\Phi(f) = ||\mathcal{S}f||_2^2 \quad \text{mit} \quad \mathcal{S}f = \nabla f \text{ oder } \mathcal{S}f = \Delta f.$$

Der Wert von λ lässt sich gemäß Kreuzvalidierungs-Techniken [Sto74, Wah90, HTF01], der L-Kurve [Han92, EHN96, Han98] oder auch gemäß anderer Prinzipien wie etwa der strukturellen Risikominimierung [Vap82, Vap98] wählen. Diese Art von Formulierung finden wir im Fall $d = 2, 3$ in vielen Scattered Data Approximationsmethoden, siehe auch [HL92, ADT95], wobei der Regularisierungsterm hier üblicherweise physikalisch motiviert ist.

Zunächst betrachten wir allerdings die Ergebnisse einiger Arbeiten, die sich mit den mathematischen Grundlagen des Lernens mittels Methoden der Wahrscheinlichkeitstheorie beschäftigen. Anschließend untersuchen wir eingehend die Funktionsrekonstruktion im Rahmen der Regularisierungstheorie. Diese Darstellung eines maschinellen Lernverfahrens als Regularisierungsproblem ist in der Literatur auch unter dem Begriff *Regularisierungsnetzwerk* bekannt [GJP95, EPP00]. So lassen sich auch Ansätze wie Support Vektor Maschinen oder neuronale Netze in dieser Form wiedergeben. In einem *Hilbertraum mit reproduzierendem Kern* existiert eine Darstellung des Regularisierungsoperators durch dessen orthonormalen Eigenvektoren. Dies erlaubt die Lösung des regularisierten Approximationsproblems in diesem Räumen durch eine endliche Linearkombination von so genannten Kernfunktionen wiederzugeben, welche auf den Datenpunkten verankert sind. Wir stellen darüberhinaus eine Möglichkeit vor, diese datenorientierte Darstellung mit der einer unendlichen Funktionenraumbasis zu kombinieren. Weiterhin zeigen wir, wie mit geeigneten Waveletbasen, für die gewisse Normäquivalenzen gelten, eine Kerndarstellung von bestimmten Regularisierungsoperatoren beziehungsweise den Lösungen des korrespondierenden regularisierten Problems erhalten werden kann. Dazu geben wir Fehlerabschätzungen für Darstellungen von Waveletkernen in einer diskreten Waveletbasis an.

3.1 Die mathematischen Grundlagen des Lernens

Wir unternehmen nun einige Betrachtungen zu den mathematischen Grundlagen des Lernens und folgen hierbei [CS01]. Es soll in diesem Abschnitt exemplarisch gezeigt werden, welche Art von Aussagen in der Theorie des maschinellen Lernens möglich sind. Das wichtige Betrachtungsobjekt ist dabei ein Wahrscheinlichkeitsmaß ρ, welches die Auswahl der Stichproben kontrolliert, aber nicht bekannt ist.

Sei X ein kompaktes Gebiet oder eine Mannigfaltigkeit in einem euklidischen Raum, $Y := \mathbb{R}$, und ρ sei ein Borel-Wahrscheinlichkeitsmaß auf $Z = X \times Y$ mit geeigneten Regularitätsbedingungen. ξ ist eine Zufallsvariable, d.h. eine reellwertige Funktion auf

dem Wahrscheinlichkeitsraum Z, $\mathbf{E}(\xi)$ ist der Erwartungswert (oder Durchschnitt) von ξ und $\sigma^2(\xi)$ die Varianz. Somit gilt

$$\mathbf{E}(\xi) = \int_Z \xi d\rho \quad \text{und} \quad \sigma^2(\xi) = \mathbf{E}((\xi - \mathbf{E}(\xi))^2) = \mathbf{E}(\xi^2) - (\mathbf{E}(\xi))^2.$$

Für jedes $x \in X$ sei $\rho(y|x)$ das bedingte Wahrscheinlichkeitsmaß auf Y in Bezug auf x und ρ_X sei das marginale Wahrscheinlichkeitsmaß auf X. Zwischen $\rho, \rho(y|x)$ und ρ_X besteht folgende Beziehung für jede integrierbare Funktion $\varphi : X \times Y \to \mathbb{R}$

$$\int_{X\times Y} \varphi(x,y) d\rho = \int_X \left(\int_Y \varphi(x,y) d\rho(y|x) \right) d\rho_X.$$

Das grundlegende Konzept der folgenden Betrachtungen ist wiederum der Quadratfehler von f, welcher definiert ist durch:

$$\mathcal{E}(f) = \mathcal{E}_\rho(f) = \int_Z (f(x) - y)^2 d\rho \quad \text{für} \quad f : X \to Y.$$

Für jedes $x \in X$ und Ausgabe $y \in Y$ ist $(f(x) - y)^2$ der Fehler durch die Nutzung von f als Modell des Zusammenhangs zwischen x und y. Durch Integration über $X \times Y$ wird der Fehler über alle Paare (x, y) gemittelt.

Gesucht wird nun das f, welches den Fehler $\mathcal{E}(f)$ minimiert. Dabei lässt sich der Fehler $\mathcal{E}(f)$ in natürlicher Weise in eine Summe aufspalten [CS01]:

$$\mathcal{E}(f) = \int_X (f(x) - f_\rho(x))^2 d\rho_X + \sigma_\rho^2. \tag{3.3}$$

Hierbei ist $f_\rho : X \to Y$ definiert durch

$$f_\rho(x) := \int_Y y d\rho(y|x).$$

Die Funktion f_ρ wird *Regressionsfunktion* genannt, sie ist für jedes $x \in X$ der Durchschnitt der y-Koordinate von $\{x\} \times Y$. Weiterhin betrachten wir für festes x die Funktion von Y nach $\mathbb{R}$, die y nach $(y - f_\rho(x))$ abbildet. Der Erwartungswert dieser Funktion ist 0, somit gilt für die Varianz

$$\sigma^2(x) = \int_Y (y - f_\rho(x))^2 d\rho(y|x).$$

Mit der Mittelung über X definieren wir nun

$$\sigma_\rho^2 := \int_X \sigma^2(x) d\rho_x = \mathcal{E}(f_\rho).$$

Der erste Term der rechten Seite von (3.3) gibt den Durchschnitt des Fehlers an, der durch die Wahl von f als Modell für f_ρ entsteht. Da σ_ρ^2 unabhängig von f ist, impliziert (3.3) weiterhin, dass f_ρ den kleinstmöglichen Fehler unter allen Funktion $f : X \to Y$ hat.

σ_ρ^2 gibt eine untere Schranke für den Fehler $\mathcal{E}$ an und hängt nur vom Wahrscheinlichkeitsmaß ρ ab. Da es somit für festes ρ eine Konstante darstellt [CS01], betrachten wir σ_ρ^2 im Folgenden nicht.

Um Aussagen über den Lernprozess machen zu können, ist eine Struktur notwendig, in deren Rahmen das Lernen stattfindet. Wir gehen im weiteren von einem Funktionenraum H aus und suchen die beste Darstellung von f_ρ in dieser Klasse.

Sei dazu $C(X)$ ein Banachraum von stetigen Funktion auf X mit Norm

$$\|f\|_\infty = \sup_{x \in X} |f(x)|.$$

Wir betrachten einen kompakten Unterraum H von $C(X)$, den *Hypothesenraum*, in dem die Verfahren die beste Approximation von f_ρ finden sollen. Die *Zielfunktion* f_H sei die Funktion $f \in \mathsf{H}$, die den Fehler $\mathcal{E}(f)$ in H minimiert

$$\int_Z (f(x) - y)^2 d\rho \underset{f \in \mathsf{H}}{\longrightarrow} \min !.$$

Mit der Aufspaltung (3.3) ist f_H auch ein Optimierer von

$$\int_X (f(x) - f_\rho(x))^2 d\rho_X \underset{f \in \mathsf{H}}{\longrightarrow} \min !.$$

Sei $\mathbf{z} \in Z^M$ eine Stichprobe. Der *empirische Fehler* von f ist definiert durch

$$\mathcal{E}_\mathbf{z}(f) = \frac{1}{M} \sum_{i=1}^{M} (f(x_i) - y_i)^2.$$

Die *empirische Zielfunktion* $f_{\mathsf{H},\mathbf{z}} = f_\mathbf{z}$ minimiert den empirischen Fehler $\mathcal{E}_\mathbf{z}$ in H. Es ist zu beachten, dass $f_\mathbf{z}$ nicht von ρ abhängt, sondern von der Stichprobe $\mathbf{z}$.

Für einen Ansatzraum H ist der *Fehler in* H einer beliebigen Funktion $f \in \mathsf{H}$ gegeben durch den normalisierten Fehler

$$\mathcal{E}_\mathsf{H}(f) = \mathcal{E}(f) - \mathcal{E}(f_\mathsf{H}) = \int_X (f - f_\rho)^2 - \int_X (f_\mathsf{H} - f_\rho)^2. \tag{3.4}$$

Wir stellen fest, dass $\mathcal{E}_\mathsf{H}(f_\mathsf{H}) = 0$ und $\mathcal{E}_\mathsf{H}(f) \geq 0$ für alle $f \in \mathsf{H}$. Weiterhin ist zu beachten, dass $\mathcal{E}(f_\mathbf{z}) \neq \mathcal{E}_\mathbf{z}(f)$ und $\mathcal{E}(f_\mathcal{H}) \neq \mathcal{E}_\mathcal{H}(f)$, es sind jeweils unterschiedliche Objekte.

Es gilt nun mit (3.3) und (3.4)

$$\int_X (f_\mathbf{z} - f_\rho)^2 = \mathcal{E}(f_\mathbf{z}) = \mathcal{E}_\mathsf{H}(f_\mathbf{z}) + \mathcal{E}(f_\mathsf{H}),$$

wie erwähnt betrachten wir die Konstante σ_ρ^2 aus (3.3) nicht. Der erste Term $\mathcal{E}_\mathsf{H}(f_\mathbf{z})$ ist der *Stichprobenfehler*, auch genannt *Schätzfehler*, und hängt von der Stichprobe ab. Der zweite Term $\mathcal{E}(f_\mathsf{H})$ ist der *Approximationsfehler*, der von der Wahl des Raumes H abhängt, aber nicht von den Daten. Wie wir sehen werden, fällt für festes H der Stichprobenfehler mit der Zahl M der Daten. Andererseits fällt für festes M der Approximationsfehler, wenn H

größer wird, aber der Stichprobenfehler steigt. Diese Eigenschaft ist eng verwandt mit der *Bias-Varianz-Zerlegung* aus der Statistik, wie wir sie auch in Gleichung (2.14) beobachtet haben.

Für den Stichprobenfehler ist es möglich, anzugeben wie viele Daten benötigt werden um mit einer Konfidenz größer als $1-\delta$ sagen zu können, dass $\mathcal{E}_{\mathsf{H}}(f_{\mathbf{z}}) = \int_X (f_{\mathbf{z}} - f_{\mathsf{H}})^2$ nicht größer als ε ist. Hier gibt [CS01] folgendes Resultat:

Satz 4. *H sei eine kompakte und konvexe Teilmenge von $\mathcal{C}(X)$. $\forall f \in \mathsf{H}$ gelte $|f(x)-y| \leq G$ f.ü. Dann gilt $\forall \varepsilon > 0$*

$$\operatorname*{Prob}_{\mathbf{z}\in Z^M} \{\mathcal{E}_{\mathsf{H}}(f_{\mathbf{z}}) \leq \varepsilon\} \geq 1 - \mathcal{N}\left(\mathsf{H}, \frac{\varepsilon}{24G}\right) e^{-\frac{M\varepsilon}{288G^2}}.$$

Hierbei ist M die Größe der Stichprobe, $\mathcal{N}(S,s)$ ist die *Überdeckungszahl*, definiert als die kleinste Zahl $l \in \mathbb{N}$, so dass l Scheiben mit Radius s in S existieren, die S überdecken. Aussagen für nicht konvexes H finden sich ebenso in [CS01]. Dieses Resultat ist eine Version der Abschätzungen aus der statistischen Lerntheorie nach Vapnik [Vap98].

Mit Hilfe eines engen Zusammenhangs zwischen Überdeckungszahlen und Entropiegrößen kann für Sobolevräume H^s gezeigt werden [CS01]:

Korollar 1. *Sei $s > d/2$, B_R abgeschlossener Ball mit Radius R in $H^s(X)$, H ist die Einbettung dieses Balles in $\mathcal{C}(X)$. $\forall f \in \mathsf{H}$ gelte $|f(x)-y| \leq G$ f.ü. Gegeben seien $\varepsilon, \delta > 0$. Dann gilt*

$$\operatorname*{Prob}_{\mathbf{z}\in Z^M}\{\mathcal{E}_{\mathsf{H}}(f_{\mathbf{z}}) \leq \varepsilon\} \geq 1-\delta,$$

falls

$$M \geq \frac{288G^2}{\varepsilon}\left[\left(\frac{24CRG}{\varepsilon}\right)^{d/s} + \ln\left(\frac{1}{\delta}\right)\right].$$

R bestimmt die Größe des Hypothesenraumes und ersetzt als Maß der Raumkomplexität die Vapnik-Chernovenkis (VC)-Dimension aus der statistischen Lerntheorie nach Vapnik [Vap98, HTF01]. Die VC-Dimension ist im Fall von Sobolevräumen unendlich [CS01].

Auch für den Fehler durch die Wahl des Hypothesenraumes, den Approximationsfehler, werden in [CS01] Abschätzungen angegeben. Wir zitieren hier das Resultat für Sobolevräume:

Satz 5. *Sei $s > d/2$ und $0 < r < s$, B_R abgeschlossener Ball mit Radius R in $H^s(X)$, H ist die Einbettung dieses Balles in $\mathcal{C}(X)$. Dann gilt für den Approximationsfehler*

$$\mathcal{E}(f_{\mathcal{H}}) \leq \mathcal{D}_{\mu\rho}^2 C \left(\frac{1}{R}\right)^{\frac{2r}{s-r}} \|f_\rho\|_r^{\frac{2s}{s-r}} + \sigma_\rho^2,$$

wobei die Konstante C von s, r und X abhängt.

$\mathcal{D}_{\mu\rho}$ bezeichnet hierbei die Operatornorm $\|J\|$, wobei J die Identitätsfunktion

$$\mathcal{L}^2_\mu(X) \xrightarrow{J} \mathcal{L}^2_\rho(X)$$

ist, mit $\mathcal{L}^2_\mu(X)$ der Raum der quadratintegrierbaren Funktionen auf X und $\mathcal{L}^2_\rho(X)$ entsprechend für das Wahrscheinlichkeitsmaß ρ. $\mathcal{D}_{\mu\rho}$ wird Verzerrung von ρ (im Verhältnis zu μ) genannt. Sie gibt an, wie sehr ρ das Raummaß μ stört. Der Wert ist Allgemeinen nicht bekannt, da ρ nicht bekannt ist.

Nach [CS01] können also sowohl Ausdrücke für den Fehler durch die Wahl des Ansatzraumes H als auch für den Fehler durch die Stichprobe angeben werden. Unter gewissen Voraussetzungen kann nun weiterhin gezeigt werden, dass eine eindeutige Lösung für bestimmte Formen des Bias-Varianz-Dilemmas existieren, siehe [CS01]. Eine numerische Berechnung ist allerdings nur in speziellen Fällen möglich.

Unbeachtet bleibt bei diesen Betrachtungen der Aspekt der Regularisierung, insbesondere die geeignete Wahl des Regularisierungsparameters im Hinblick auf das Bias-Varianz-Dilemma. Hilberträume mit reproduzierendem Kern [Aro50, Mes62, Wah90, Sai97] spielen, wie wir sehen werden, eine besondere Rolle in der mathematischen Lerntheorie [GJP93, GJP95, Gir98, EPP00, Vap98, CS01, SS02]. Für diese Räume sind in [CS02] Approximationsaussagen und Strategien zur Bestimmung eines optimalen Regularisierungsparameters dargestellt.

3.2 Regularisierung der Datenapproximation

Betrachten wir nun das zu lösende Regularisierungsproblem (3.1) genauer

$$R(f) \underset{f\in V}{\longrightarrow} \min !$$

mit

$$R(f) = \frac{1}{M}\sum_{i=1}^{M} C(f(\underline{x}_i), y_i) + \lambda\Phi(f).$$

Im Folgenden verwenden wir den Quadratfehler als Kostenfunktion

$$C(f(\underline{x}), y_i) = (f(\underline{x}) - y_i)^2$$

und Regularisierungsterme der Form

$$\Phi(f) = ||\mathcal{S}f||^2_{L^2} \tag{3.5}$$

für einen gegebenen linearen Operator $\mathcal{S}$.

Nehmen wir weiterhin an, dass wir eine Basis von V durch $\{\varphi_j(\underline{x})\}_{j=1}^{\infty}$ gegeben haben. Wir können nun jedes $f \in V$ als

$$f(\underline{x}) = \sum_{j=1}^{\infty} \alpha_j \varphi_j(\underline{x}) \tag{3.6}$$

mit zugehörigen Freiheitsgraden α_j darstellen.

Im Funktionenraum V ist somit zu minimieren

$$R(f) = \frac{1}{M}\sum_{i=1}^{M}(f(\underline{x}_i) - y_i)^2 + \lambda\|\mathcal{S}f\|_{L^2}^2. \tag{3.7}$$

Wir setzen die Darstellung (3.6) in (3.7) ein und erhalten

$$R(f) = \frac{1}{M}\sum_{i=1}^{M}\left(\sum_{j=1}^{\infty}\alpha_j\varphi_j(\underline{x}_i) - y_i\right)^2 + \lambda\left\|\mathcal{S}\textstyle\sum_{j=1}^{\infty}\alpha_j\varphi_j\right\|_{L^2}^2 \tag{3.8}$$

$$= \frac{1}{M}\sum_{i=1}^{M}\left(\sum_{j=1}^{\infty}\alpha_j\varphi_j(\underline{x}_i) - y_i\right)^2 + \lambda\sum_{i=1}^{\infty}\sum_{j=1}^{\infty}\alpha_i\alpha_j(\mathcal{S}\varphi_i, \mathcal{S}\varphi_j)_{L^2}. \tag{3.9}$$

Differentiation nach α_k ergibt jeweils

$$0 = \frac{\partial R(f)}{\partial\alpha_k} = \frac{2}{M}\sum_{i=1}^{M}\left(\sum_{j=1}^{\infty}\alpha_j\varphi_j(\underline{x}_i) - y_i\right)\cdot\varphi_k(\underline{x}_i) + 2\lambda\sum_{j=1}^{\infty}\alpha_j(\mathcal{S}\varphi_j, \mathcal{S}\varphi_k)_{L^2}. \tag{3.10}$$

Dies ist äquivalent zu $(k = 1, \ldots, \infty)$

$$\lambda\sum_{j=1}^{\infty}\alpha_j(\mathcal{S}\varphi_j, \mathcal{S}\varphi_k)_{L^2} + \frac{1}{M}\sum_{j=1}^{\infty}\alpha_j\sum_{i=1}^{M}\varphi_j(\underline{x}_i)\cdot\varphi_k(\underline{x}_i) = \frac{1}{M}\sum_{i=1}^{M}y_i\varphi_k(\underline{x}_i), \tag{3.11}$$

und wir erhalten $(k = 1, \ldots, \infty)$

$$\sum_{j=1}^{\infty}\alpha_j\left[M\lambda(\mathcal{S}\varphi_j, \mathcal{S}\varphi_k)_{L^2} + \sum_{i=1}^{M}\varphi_j(\underline{x}_i)\cdot\varphi_k(\underline{x}_i)\right] = \sum_{i=1}^{M}y_i\varphi_k(\underline{x}_i). \tag{3.12}$$

In Matrix-Notation, wobei wir hier unendlich dimensionale Matrizen und Vektoren verwenden, erhalten wir schließlich das lineare System

$$(\mathcal{B}^t\mathcal{B} + \lambda M\cdot\mathcal{C})\alpha = \mathcal{B}^t y. \tag{3.13}$$

Hier ist die *Regularisierungsmatrix* $\mathcal{C}$ eine quadratische positiv semi-definite Matrix mit Einträgen

$$\mathcal{C}_{j,k} = (\mathcal{S}\varphi_j, \mathcal{S}\varphi_k)_{L^2}, \quad j,k = 1, \ldots, \infty, \tag{3.14}$$

und die *Datenmatrix* $\mathcal{B}$ ist eine rechteckige Matrix mit Einträgen

$$\mathcal{B}_{i,j} = \varphi_j(\underline{x}_i), \quad i = 1, \ldots, M,\ j = 1, \ldots, \infty. \tag{3.15}$$

Der Vektor y enthält die Daten y_i und der Vektor α enthält die Freiheitsgrade α_j.

Die Regularisierungsmatrix kann auch dargestellt werden als

$$\mathcal{C} = \mathcal{S}^t\mathcal{S},$$

mit $\mathcal{S}_i = \mathcal{S}\varphi_i$, wobei $\mathcal{S}\varphi_i$ in einem geeigneten Raum mit Skalarprodukt lebt. $\mathcal{S}$ kann im Weiteren ohne Beschränkung der Allgemeinheit als positiv semi-definit angenommen werden, denn von Interesse ist die Matrix $\mathcal{C}$ zugehörig zum Regularisierungsterm Φ. Durch die Definition $\tilde{\mathcal{S}} := \sqrt{(\mathcal{S}^t\mathcal{S})}$ kann stets ein positiv semi-definiter Operator erhalten werden, der $\mathcal{C}$ mittels $\tilde{\mathcal{S}}^t\tilde{\mathcal{S}}$ ergibt.

3.2.1 Der Kern-Ansatz

Da der Regularisierungsterm Φ, und somit auch die Regularisierungsmatrix $\mathcal{C}$, positiv semi-definit ist, kann eine Darstellung in Diagonalform bezüglich einer Basis von Eigenvektoren angegeben werden. Seien $\{\gamma_j\}_{j=1}^{\infty}$ die Eigenwerte von $\mathcal{C}$ und bezeichne $\{\varphi_j(\underline{x})\}_{j=1}^{\infty}$ nun die zugehörigen Eigenvektoren. Bezüglich einer orthogonalen Basis von Eigenvektoren für V hat ein Regularisierungsterm Φ der Form (3.5) somit die einfache Darstellung

$$\Phi(f) = \sum_{j=1}^{\infty} \alpha_j^2 \gamma_j, \tag{3.16}$$

wobei die Freiheitsgrade α_j sich nun auf die Eigenvektorbasis $\{\varphi_j\}_{j=1}^{\infty}$ beziehen. Der Anteil der Funktion, der durch die Eigenvektoren zum Null-Eigenwert dargestellt wird, wird durch einen solchen Regularisierungsterm offensichtlich nicht bestraft. Teilen wir also den Raum V auf in

$$V = V_0 \otimes V_{\mathcal{C}},$$

wobei V_0 der Nullraum von $\Phi(f)$ ist und $V_{\mathcal{C}} = \text{span}\{\varphi_j(\underline{x}) | \gamma_j \neq 0\}_{j=1}^{\infty}$.

Aus Gründen der einfacheren Notation arbeiten wir im Weiteren ohne die Null-Eigenwerte, die bei positiv semi-definiten Operatoren vorhanden sind. Formal gilt das Folgende somit nur für positiv definite Operatoren beziehungsweise auf dem Raum $V_{\mathcal{C}} \subset V$. Die Ergebnisse lassen sich aber entsprechend erweitern, indem Funktionen aus dem Nullraum V_0 des Operators explizit als zusätzliche additive Komponente in der Funktionsdarstellung verwendet werden, siehe z.B. [Wah90, HTF01, SS02].

Betrachten wir also Regularisierungsterme der Form (3.16). Wir haben nun das vereinfachte Funktional $R(f)$ in den Koeffizienten $\{\alpha_j\}_{j=1}^{\infty}$ zu minimieren

$$R(f) = \frac{1}{M} \sum_{i=1}^{M} (f(\underline{x}_i) - y_i)^2 + \lambda \sum_{j=1}^{\infty} \alpha_j^2 \gamma_j \quad \text{mit } f(\underline{x}) = \sum_{j=1}^{\infty} \alpha_j \varphi_j(\underline{x}).$$

Differentiation nach α_j, $j = 1, \ldots, \infty$, ergibt

$$0 = \frac{\partial R(f)}{\partial \alpha_j} = \frac{1}{M} \sum_{i=1}^{M} (f(\underline{x}_i) - y_i) \cdot \varphi_j(\underline{x}_i) + \lambda \alpha_j \gamma_j. \tag{3.17}$$

Wir definieren nun neue punktbezogene Unbekannte β_i

$$\beta_i := y_i - f(\underline{x}_i). \tag{3.18}$$

Einsetzen von (3.18) in (3.17) und Auflösen nach α_j, $j = 1, \ldots, \infty$ ergibt

$$\alpha_j = \frac{1}{\lambda M \gamma_j} \sum_{i=1}^{M} \beta_i \varphi_j(\underline{x}_i). \tag{3.19}$$

Die Lösung von (3.1) für $\Phi(f) = \sum_{j=1}^{\infty} \alpha_j^2 \gamma_j$ ist somit

$$f(\underline{x}) = \sum_{j=1}^{\infty} \alpha_j \varphi_j(\underline{x}) = \frac{1}{\lambda M} \sum_{j=1}^{\infty} \frac{1}{\gamma_j} \sum_{i=1}^{M} \beta_i \varphi_j(\underline{x}_i) \varphi_j(\underline{x}) = \frac{1}{\lambda M} \sum_{i=1}^{M} \beta_i k(\underline{x}_i, \underline{x}), \tag{3.20}$$

wobei die so genannten *Kernfunktionen* k definiert sind mittels

$$k(\underline{x}, \underline{y}) := \sum_{j=1}^{\infty} \frac{1}{\gamma_j} \varphi_j(\underline{x}) \varphi_j(\underline{y}).$$

Die Darstellung (3.20) ist eine Form das Repräsentertheorems nach [KW71]. Werden also bestimmte Funktionen $k(\underline{x}_i, \cdot) \in V$, die an den Positionen der Datenpunkte $\underline{x}_i$ festgemacht sind, zur Darstellung der Lösung der Approximationsaufgabe (3.7) benutzt, erweist sich diese Lösung als eine endliche Reihe von M Termen. Diese Darstellung wird auch lineare Superposition von Kernfunktionen genannt. Im Fall eines positiv semi-definiten Operators wird zu dieser Darstellung wie erwähnt eine Funktion aus dem Nullraum des Operators addiert.

Es sei daran erinnert, dass γ_j und φ_j die Eigenwerte und zugehörigen Eigenfunktionen des Regularisierungsoperators sind. Über die Darstellung der gesuchten Funktion f in der unendlich dimensionalen Eigenvektorbasis erreichen wir also eine äquivalente Repräsentation in Form einer endlichen Summe.

Weiterhin resultiert das Einsetzen der Darstellung (3.20) von f in (3.18) in

$$\beta_i = \left(y_i - \frac{1}{\lambda M} \sum_{k=1}^{M} \beta_i k(\underline{x}_k, \underline{x}_i) \right).$$

In Matrixschreibweise erhalten wir das System

$$(\mathcal{I} + \frac{1}{\lambda M} \mathcal{K}) \beta = y \tag{3.21}$$

mit $\mathcal{K}_{i,j} = k(\underline{x}_i, \underline{x}_j)$.

3.2.2 Hilberträume mit reproduzierendem Kern

Es zeigt sich nun, dass die symmetrische Funktion k

$$k(\underline{x}, \underline{y}) = \sum_{j=1}^{\infty} \frac{1}{\gamma_j} \varphi_j(\underline{x}) \varphi_j(\underline{y})$$

aus Abschnitt 3.2.1 gerade der reproduzierende Kern eines Hilbertraumes mit reproduzierendem Kern (HRK) ist. Wir geben nun in kompakter Form einige der wesentlichen Eigenschaften dieser Funktionenräume wieder. Erstmals vorgestellt wurden diese Räume in [Aro50], als Auswahl von Veröffentlichungen über HRK seien [Aro50, Mes62, Wah90, Sai97, Gir98, SS02] erwähnt. Beweise, weitere Erläuterungen und Referenzen finden sich dort.

Ein Hilbertraum mit reproduzierendem Kern H ist ein Hilbertraum von reellwertigen Funktion, in dem alle Punktauswertungen beschränkte lineare Funktionale sind. Sei nun ein HRK H über einem Gebiet $\mathcal{T}$ gegeben, dann existiert nach dem Rieszschen Darstellungssatz für jedes $x \in \mathcal{T}$ ein Element $\eta_x \in H$ mit der Eigenschaft

$$f(x) = \langle \eta_x, f \rangle \quad \forall f \in H,$$

wobei $\langle \cdot, \cdot \rangle$ das innere Produkt von H ist. η_x wird der Repräsenter der Auswertung an der Stelle x genannt. η_x agiert in ähnlicher Weise wie die Deltafunktion in L^2, wobei nebenbei bemerkt sei, dass L^2 kein HRK ist, da nicht alle Elemente punktweise definiert sind.

Setze nun

$$k(x, y) = \langle \eta_x, \eta_y \rangle.$$

Dann ist $k(x, y)$ positiv semi-definit auf $\mathcal{T} \otimes \mathcal{T}$, d.h. $\sum_{i,j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \geq 0$ und positiv definit, sofern „$>$“ gilt. Es sei an dieser Stelle bemerkt, dass in vielen Arbeiten, insbesondere im Bereich des maschinellen Lernens, auch positiv definit für „$\geq$“ und strikt positiv definit für die zusätzliche Bedingung „$>$“ gebräuchlich ist.

Als eine grundlegende Eigenschaft eines HRK ergibt sich

Satz 6. *Zu jedem HRK gibt es eine eindeutige positiv semi-definite Funktion, den so genannten reproduzierenden Kern. Umgekehrt kann zu jeder positiv semi-definiten Funktion auf $\mathcal{T} \otimes \mathcal{T}$ ein eindeutiger HRK von reellwertigen Funktionen auf $\mathcal{T}$ konstruiert werden.*

Der zu k assoziierte Hilbertraum wird dabei konstruiert, indem er alle endlich dimensionalen Linearkombinationen der Form $\sum \alpha_j k(x_j, \cdot)$ enthält und deren Limiten unter der Norm, welche durch das innere Produkt

$$\langle k(x, \cdot), k(y, \cdot) \rangle = k(x, y)$$

induziert wird. Normkonvergenz impliziert hierbei punktweise Konvergenz in einem HRK, da gilt

$$|f_n(x) - f_m(x)| = |\langle k(x, \cdot), f_n - f_m \rangle| \leq k(x, x) \|f_n - f_m\|.$$

Ein Beispiel für einen HRK sind eindimensionale Sobolevräume $H^s[0, 1]$, z.B. $H^1[0, 1]$ mit der Norm

$$\|f\|^2 = f(0)^2 + \int_0^1 \left(\frac{\partial}{du} f(u) \right)^2 du.$$

Das ist ein Hilbertraum mit reproduzierendem Kern [Wah90]

$$k(x, y) = 1 + \max(x, y).$$

Der folgende Satz über Tensorprodukte von HRK liefert nun eine Möglichkeit zur Konstruktion von mehrdimensionalen HRK:

Satz 7. *Das Tensorprodukt $H = H_1 \otimes H_2$, wobei H_1 und H_2 Hilberträume mit reproduzierendem Kern k_1 beziehungsweise k_2 sind, ist ebenfalls ein HRK. Als reproduzierender Kern von H ergibt sich das Produkt der reproduzierenden Kerne*

$$k((x_1, x_2), (y_1, y_2)) = k_1(x_1, y_1) \cdot k_2(x_2, y_2).$$

Damit ist also $H^1 \otimes \cdots \otimes H^1$ in $[0, 1]^d$ ein Hilbertraum mit dem reproduzierendem Kern

$$k(\underline{x}, \underline{y}) = \prod_{j=1}^{d} (1 + \max(x_j, y_j)).$$

Für Sobolevräume mit dominierender gemischter Ableitung H^s_{mix} gilt die Darstellung [Hoc99, GK00, HKZ00]

$$H^s_{mix} = H^s \otimes \cdots \otimes H^s.$$

Als eine äquivalente Norm ergibt sich, hier für $s = 1$,

$$\|f\|^2_{H^1_{mix}} = \sum_{0 \leq \underline{k} \leq 1} \|\frac{\partial}{\partial \underline{x}_{\underline{k}}} u\|_{L^2}$$

mit einem Multi-Index $\underline{k}$, wobei „$\leq$“ komponentenweise zu verstehen ist. Somit sind also Sobolevräume mit dominierender gemischter Ableitung auch Hilberträume mit reproduzierendem Kern. Der Sobolevraum H^s andererseits ist in höheren Dimensionen im Allgemeinen kein HRK, da die Punktauswertung für $s > d/2$ nach dem Sobolevschen Einbettungssatz nicht mehr beschränkt ist.

Es sei bemerkt, dass in [SW98] sich basierend auf der äquivalenten Norm für H^1

$$\|f\|^2 = f(1)^2 + \int_0^1 \left(\frac{\partial}{du} f(u)\right)^2 du$$

eine leicht anderer Kern für H^1_{mix} ergibt:

$$k(\underline{x}, \underline{y}) = \prod_{j=1}^{d} (1 + \min(1 - x_j, 1 - y_j)).$$

3.2.3 Zusammenhang zwischen Regularisierungsoperatoren und Hilberträumen mit reproduzierendem Kern

Im Folgenden wollen wir die Zusammenhänge zwischen Regularisierungsoperatoren und Hilberträumen mit reproduzierendem Kern genauer betrachten, für den Bereich des maschinellen Lernens sei hier insbesondere auf [GJP95, EPP00, SS02] verwiesen, auf denen dieser Abschnitt basiert.

Wir haben gesehen, dass es zu jedem Regularisierungsoperator der Form (2.19) einen entsprechenden HRK H mit Kern k gibt. Umgekehrt gibt es auch zu jedem HRK H mit Kern k einen Regularisierungsoperator, wie der folgende Satz besagt.

Satz 8 (HRK und Regularisierungsoperatoren). *Für jeden HRK H mit reproduzierendem Kern k existiert ein entsprechender Regularisierungsoperator $\mathcal{S} : H \to D$, wobei D ein Hilbertraum ist, so dass für alle $f \in H$ gilt*

$$\langle \mathcal{S}k(x, \cdot), \mathcal{S}f(\cdot) \rangle_{\mathcal{D}} = f(x). \tag{3.22}$$

Insbesondere gilt

$$\langle \mathcal{S}k(x, \cdot), \mathcal{S}k(y, \cdot) \rangle_{\mathcal{D}} = k(x, y). \tag{3.23}$$

Umgekehrt existiert für jeden Regularisierungsoperator $\mathcal{S} : V \to D$, wobei V ein mit einem Skalarprodukt versehener Funktionenraum ist, ein entsprechender HRK H mit reproduzierendem Kern k, so dass (3.22) und (3.23) gelten.

Dies bedeutet insbesondere, dass D mit dem inneren Produkt $\langle \mathcal{S}\cdot, \mathcal{S}\cdot \rangle$ ein HRK ist.

Es ist wiederum festzustellen, dass allein durch die Wahl des Regularisierungsoperators $\mathcal{S}$, abgesehen von dessen Nullraum, der Funktionenraum H festgelegt wird, in dem die Lösung des Minimierungsproblems (3.1) erhalten wird, unabhängig vom Raum V, in dem die Approximation f ursprünglich gesucht wird. Die Darstellung ist dabei durch das Repräsentertheorem (3.20)

$$f(\cdot) = \sum_{i=1}^{M} \alpha_i k(\underline{x}_i, \cdot)$$

gegeben.

Diese Beziehung zwischen Regularisierungsoperator $\mathcal{S}$ und Raum H mit Kern k ist allerdings nicht eindeutig, zu einem gegebenen Operator $\mathcal{S}$ können verschiedene Kerne gebildet werden, die die Bedingungen (3.22) und (3.23) erfüllen. Allerdings lässt sich k immer durch eine Darstellung in der Eigenvektorzerlegung angeben.

Proposition 4 (Diskretes Gegenstück). *Gegeben sei ein Regularisierungsoperator $\mathcal{S}$ mit zugehöriger Eigenwertzerlegung (γ_j, φ_j) von $\mathcal{S}^*\mathcal{S}$, der Kern K mit*

$$k(\underline{x}, \underline{y}) = \sum_{j=1, \gamma_j \neq 0}^{\infty} \frac{d_j}{\gamma_j} \varphi_j(\underline{x}) \varphi_j(\underline{y}),$$

wobei $d_j \in \{0, 1\}$ und $\sum_j \frac{d_j}{\gamma_j}$ konvergiert, dann erfüllt k (3.23). Der entsprechende HRK ist gegeben durch $\operatorname{span}\{\varphi_i | d_i = 1\}$.

Somit kann eine große Klasse von Kernen einem Regularisierungsoperator zugeordnet werden und umgekehrt. Mit der Wahl eines Kerns wird auf einen Teilraum der Eigenwertzerlegung eingeschränkt.

Ein gängiges Beispiel aus anderen Bereichen sind die in [Duc77] benutzten Sobolev-Semi-Normen als Regularisierungsterme

$$\Phi_m(f) = \sum_{|\alpha|=m} \binom{m}{\alpha} |D^\alpha f|.$$

Der dazugehörige Kern ergibt sich als [GJP95]

$$k(\underline{x}, \underline{y}) = \begin{cases} \|\underline{x} - \underline{y}\|^{2m-d} \ln(\|\underline{x} - \underline{y}\|). & \text{falls } 2m > d \text{ und } d \text{ gerade,} \\ \|\underline{x} - \underline{y}\|^{2m-d}, & \text{sonst.} \end{cases}$$

Für $d = m = 2$ ergibt sich die bekannte *Thin-Plate-Splines*-Anwendung, siehe z.B. [RHA03].

Die Gaußschen radialen Basisfunktionen

$$k(\underline{x}, \underline{y}) = \exp\left(-\frac{\|\underline{x} - \underline{y}\|^2}{2\sigma^2} \right)$$

wiederum sind die Kerne zu [GJP95, SS02]

$$\|\mathcal{S}f\|^2 = \int_X \sum_n \frac{\sigma^{2n}}{n!2^n} (\nabla^n f(x))^2 dx.$$

Aber auch viele andere Approximationsschemata wie etwa additive Modelle, Hyperbasisfunktionen, Ridge-Approximationsmodelle und verschiedene Typen neuronaler Netzwerke lassen sich durch eine spezielle Wahl des Regularisierungsoperators beziehungsweise des Kerns ableiten [GJP95, EPP00, SS02]. Typische Kerne sind neben den bereits erwähnten auch B_n-Splines oder polynomische Kerne $k_c(\underline{x}, \underline{y}) = (\langle \underline{x}, \underline{y} \rangle + c)^d$.

Es sei darauf hingewiesen, dass eine Erweiterung der Theorie der Hilberträume mit reproduzierendem Kern für nicht-Hilbertsche Mengen in [CMR03] vorgestellt wird. In diesem erweiterten Rahmen lässt sich ein verallgemeinertes Repräsentertheorem zeigen, die Lösung des Lernproblems ist dabei weiterhin eine Linearkombination von Kernen.

Support Vektor Maschinen (SVM) nach Vapnik [Vap98, SS02] sind im Bereich des maschinellen Lernens in den letzten Jahren für viele Anwendungen erfolgreich eingesetzt worden. Die Support Vektor Maschine wird dabei für die Klassifikation von Daten motiviert durch eine Hyperebene, welche die zwei Klassen voneinander trennt. Ein Parameter ist die Bandbreite welche den Abstand zwischen der Hyperebene und dem nächstliegenden Datenpunkt angibt. Dieser Abstand soll maximiert werden. Ausgehend von diesem Ansatz wird der nicht trennbare Fall mit Hilfe von Straftermen eingeführt. Mit dem so genannten Kern-Trick wird auf nichtlineare Trennflächen verallgemeinert. Allerdings verliert im nicht-trennbaren Fall die Motivation über die Bandbreite einiges an Bedeutung, wie in [PS03] angemerkt wird.

Es ist bemerkenswert, dass dieser Ansatz gleichwertig in Form des Minimierungsproblems (3.1) ausgedrückt werden kann. Die spezielle Wahl der Kostenfunktion

$$C(x, y) = |y - x|_\varepsilon = \begin{cases} 0 & \text{falls } |y - x| < \varepsilon, \\ |y - x| - \varepsilon & \text{sonst,} \end{cases}$$

in (3.2) statt des L_2-Fehlers resultiert in einer Regularisierungsformulierung, von der gezeigt werden konnte [Gir98, SSM98, EPP00], dass sie zur Support Vektor Maschine äquivalent ist. Durch die Herleitung der Support Vektor Maschine über die Regularisierungstheorie wird eine allgemeinere Theorie entwickelt. Allerdings ist der Begriff der trennenden Hyperebene bei der Klassifikation eine auch dem mathematischen Laien anschauliche Darstellung des Verfahrens.

Auch der Fehler im Quadrat wird im Rahmen der SVM genutzt [SV99, FM01, SGB$^+$02, RYP03]. Dieser Form einer Kerndarstellung einer regularisierten Approximationsproblems lässt sich bis ins Jahr 1989 zurückverfolgen, siehe [PS03] und Referenzen. Empirische Ergebnisse zeigen, dass in den Ergebnissen zwischen beiden Kostenfunktionen keine größeren Unterschiede bestehen [SGB$^+$02, RYP03].

Eine Bemerkung zum Aufwand eines Verfahrens mit Kernansatz ist an dieser Stelle angebracht. So ist nur bei der Wahl des Fehlers im Quadrat im Funktional (3.2) eine lineare Gleichung (3.21) zu lösen. Für die Bestimmung der Lösung bei anderen Fehlerfunktionalen sind Optimierungsverfahren notwendig. Für das Lösen der linearen Gleichung

muss die vollbesetzte $M \times M$-Kernmatrix $\mathcal{K}$ aufgestellt werden, was $O(M^2)$ Speicher und $O(M^2 d)$ Aufwand benötigt. Zur eigentlichen Lösung des Gleichungssystems sind $O(M^3)$ Operationen erforderlich. Die Komplexität eines modernen Optimierungsverfahrens für eine Support Vektor Maschine ist allerdings nicht zugänglich für eine direkte formale Analyse [RYP03]. Der Aufwand hängt ab von einem komplexen Zusammenspiel der Größe der Datenmenge, der Zahl der Support Vektoren und des zur Verfügung stehenden Hauptspeichers. Empirisch wird beispielsweise in [Joa98] eine Skalierung von $O(M^{2.1})$ festgestellt. Somit ist das Verfahren für große Datenmengen nicht geeignet, da die Lösung des Gleichungssystems beziehungsweise des Optimierungsproblems zu viel Rechenzeit benötigt.

3.2.4 Kombination von Funktionenraumdarstellung und Kerndarstellung

Wir untersuchen nun eine gemeinsame Darstellung des Problems durch Funktionen auf den Datenpunkten und einer Basis des Funktionenraumes Daraus ergeben sich einige interessante Aspekte und Bezüge zu gänzlich anderen Anwendungsbereichen.

Betrachten wir nun nochmal (3.17) bis (3.19) Die Matrix $\mathcal{B}$ sei wie in (3.15) bestimmt durch

$$\mathcal{B}_{i,j} := \varphi_j(\underline{x}_i).$$

Damit können wir $f(\underline{x}_i)$ bezüglich der Koeffizienten α_j auch darstellen als

$$f(\underline{x}_i) = \sum_{j=1}^{\infty} \alpha_j \varphi_j(\underline{x}_i) = \sum_{j=1}^{\infty} \mathcal{B}_{i,j}\alpha_j = (\mathcal{B}\alpha)_i.$$

Die Diagonalmatrix $\mathcal{C}$ sei definiert durch

$$\mathcal{C}_{i,i} := \gamma_i. \tag{3.24}$$

Somit können wir (3.19) auch schreiben als

$$\lambda M \cdot \mathcal{C}\alpha = \sum_{i=1}^{M} \beta_i \varphi_j(\underline{x}_i) = (\mathcal{B}^t\beta)_j.$$

Nun stellen wir das System der zu erfüllenden Gleichungen (3.18) und (3.19) in dieser Notation auf

$$\begin{cases} \mathcal{I}\beta = y - \mathcal{B}\alpha & \text{in } \mathbb{R}^M, \\ \lambda M \cdot \mathcal{C}\alpha = \mathcal{B}^t\beta & \text{in } V'. \end{cases}$$

In Matrixschreibweise ergibt sich somit

$$\begin{pmatrix} \mathcal{I} & \mathcal{B} \\ \mathcal{B}^t & -\lambda M \cdot \mathcal{C} \end{pmatrix} \begin{pmatrix} \beta \\ \alpha \end{pmatrix} = \begin{pmatrix} y \\ 0 \end{pmatrix}. \tag{3.25}$$

Es zeigt sich nun, dass das Gleichungssystem (3.13)

$$(\mathcal{B}^t\mathcal{B} + \lambda M \cdot \mathcal{C})\alpha = \mathcal{B}^t y$$

zur Bestimmung einer Lösung des regularisierten Problems im Funktionenraum V sich als Schurkomplement nach α ergibt.

Andererseits erhalten wir das Gleichungssystem (3.21) zur Bestimmung der regularisierten Lösung in Kerndarstellung

$$(\mathcal{I} + \frac{1}{\lambda M}\mathcal{K})\beta = y$$

als Schurkomplement von (3.25) nach β, wobei

$$\mathcal{K} = \mathcal{B}\mathcal{C}^{-1}\mathcal{B}^t.$$

Hier bezeichnet $\mathcal{C}^{-1}$, die Inverse zum Operator $\mathcal{C}$, die Diagonalmatrix mit den Einträgen $\frac{1}{\gamma_i}$. Ein Regularisierungsterm der Form (3.16) kann also auch geschrieben werden als

$$\Phi(f) = \sum_{j=1}^{\infty} \alpha_j^2 \gamma_j = \left\langle \sqrt{\mathcal{C}^{-1}}\alpha, \sqrt{\mathcal{C}^{-1}}\alpha \right\rangle.$$

Eine alternative Möglichkeit zur Herleitung des Systems (3.25) besteht in der Einführung von β als Lagrangeschem Multiplikator. Wir gehen wiederum von (3.7) aus

$$\min_{\alpha}(\mathcal{B}\alpha - y)^2 + \lambda(\mathcal{S}\alpha)^2$$

und führen die Variable w mittels $w := \mathcal{B}\alpha$ ein. Betrachten wir diese als Nebenbedingung für das ursprüngliche Minimierungsproblem, erhalten wir die Lagrangesche L

$$L(\alpha, w, \beta) = (w - y)^2 + \lambda(\mathcal{S}\alpha)^2 + 2(\beta, w - \mathcal{B}\alpha),$$

mit dem Lagrangeschen Multiplikator β. Das dazugehörige $\min - \max$-Problem lautet

$$\min_{\alpha,w} \max_{\beta} (w - y)^2 + \lambda(\mathcal{S}\alpha)^2 + 2(\beta, w - \mathcal{B}\alpha).$$

Die dazugehörigen Eulergleichungen, d.h. die Ableitungen nach α, w und β ergeben sich als

$$\begin{aligned} \nabla_\alpha L &= \lambda\mathcal{S}^*\mathcal{S}\alpha - \mathcal{B}^t\beta &&= 0, \\ \nabla_w L &= w - y + \beta &&= 0, \\ \nabla_\beta L &= w - \mathcal{B}\alpha &&= 0. \end{aligned}$$

Die Elimination von w ergibt nun wiederum das System (3.25)

$$\begin{pmatrix} \mathcal{I} & \mathcal{B} \\ \mathcal{B}^t & -\lambda M \cdot \mathcal{C} \end{pmatrix} \begin{pmatrix} \beta \\ \alpha \end{pmatrix} = \begin{pmatrix} y \\ 0 \end{pmatrix}.$$

Wir untersuchen nun die Frage, wie Funktionale in α und β aussehen, für die dieses System eine notwendige Bedingung darstellt. Da beide Diagonalelemente symmetrisch sind, korrespondiert das System zu dem Sattelpunktproblem

$$\min_{\beta} \max_{\alpha} \frac{1}{2}\|\beta\|^2 + \langle \mathcal{B}^t\beta, \alpha\rangle - \frac{\lambda M}{2}\|\mathcal{S}\alpha\|^2 - \langle y, \beta\rangle.$$

Verallgemeinerte Probleme der Form

$$\begin{cases} \mathcal{A}\beta + \mathcal{B}\alpha = y & \text{in } V_1', \\ \mathcal{B}^t\beta + \mathcal{D}\alpha = z & \text{in } V_2', \end{cases} \tag{3.26}$$

die in ein solches Sattelpunktproblem resultieren, tauchen z.B. bei gemischten Finite Element-Methoden [BF91] auf. Es sei darauf hingewiesen, dass im Vergleich mit der dort üblichen Notation $\mathcal{B}$ und $\mathcal{B}^t$ vertauscht sind. Üblicherweise ist dort allerdings der Operator $\mathcal{D}$ nicht vorhanden. Bei bestimmten Problemen wie z.B. einem fast inkompressiblen Material ergibt sich die allgemeinere Form. Hierbei ist $\mathcal{D}$ ein Penalty-Term zur Stabilisierung des Verfahrens. Anwendungen ergeben sich z.B. bei der numerischen Behandlung von Balken und Bögen, beziehungsweise Schalen mit Finiten Elementen, Referenzen hierzu finden sich in [BF91]. Auch der Begriff des erweiterten Lagrange-Algorithmus für die Lösung von Systemen dieser Art sei in diesem Zusammenhang erwähnt, welcher insbesondere bei der numerischen Behandlung der Stokes-Gleichung eine wichtige Rolle spielt [BF91].

In [BF91] werden Aussagen über die Existenz und Eindeutigkeit von Lösungen (β, α) zu der zu (3.26) gehörenden Variationsformulierung

$$\begin{cases} a(\beta, \zeta) + b(\alpha, \zeta) = \langle y, \zeta \rangle & \forall \zeta \in V_1', \\ b(\eta, \beta) - d(\alpha, \eta) = \langle z, \eta \rangle & \forall \eta \in V_2', \end{cases} \tag{3.27}$$

gemacht.

Satz 9. *Seien $a(\cdot,\cdot)$, $b(\cdot,\cdot)$ und $d(\cdot,\cdot)$ stetige Bilinearformen auf $V_1' \times V_1'$, $V_2' \times V_1'$ beziehungsweise $V_2' \times V_2'$. Weiterhin sei $a(\cdot,\cdot)$ und $d(\cdot,\cdot)$ symmetrisch positiv semi-definit. Gelte weiterhin*

(i) $a(\cdot,\cdot)$ ist invertierbar auf $\mathsf{N}(\mathcal{B}^t)$, d.h. es existiert $c_a > 0$ so dass

$$\inf_{\beta_0 \in \mathsf{N}(\mathcal{B}^t)} \sup_{\zeta_0 \in \mathsf{N}(\mathcal{B}^t)} \frac{a(\beta_0, \zeta_0)}{\|\beta_0\|_{V_1} \|\zeta_0\|_{V_1}} \geq c_a.$$

(ii) Das Bild von $\mathcal{B}$ in V_1' ist abgeschlossen, d.h. es existiert $c_b > 0$ so dass

$$\sup_{\eta \in V_2} \frac{b(\eta, \zeta)}{\|\eta\|_{V_2}} \geq c_b \|\zeta\|_{V_1/\mathsf{N}(\mathcal{B}^t)}.$$

(iii) Es existiert ein $c_d > 0$, so dass für jedes $\overline{\alpha} \in \mathsf{N}(\mathcal{B})^\perp$ und für jedes $\varepsilon > 0$ die Lösung $\alpha_0 \in \mathsf{N}(\mathcal{B})$ der Gleichung

$$\varepsilon((\alpha_0, \eta))_{V_2} + d(\alpha_0, \eta) = -d(\overline{\alpha}, \eta), \quad \forall \eta \in \mathsf{N}(\mathcal{B})$$

beschränkt ist durch $\|\alpha_0\|_{V_2} \leq \frac{1}{c_d} \|\overline{\alpha}\|_{V_2}$.

Dann hat das Problem (3.27) für jedes $y \in V_1'$ und $z \in \mathsf{R}(\mathcal{B}^t)$ eine Lösung (β, α), welche in $V_1' \times V_2'/\mathsf{M}$ eindeutig ist, wobei

$$\mathsf{M} := \mathsf{N}(\mathcal{B}) \cap \mathsf{N}(\mathcal{D}).$$

Weiterhin haben wir die Schranke

$$\|\beta\|_{V_1} + \|\alpha\|_{V_2/\mathsf{N}(B)} \leq K(\|y\|_{V_1'} + \|z\|_{V_2'}) \tag{3.28}$$

mit einer nichtlinearen Funktion K abhängig von $\|a\|, \|d\|, 1/c_a, 1/c_b, 1/c_d$, welche beschränkt auf beschränkten Teilmengen ist.

Beweis siehe [BF91]. Es sei bemerkt, dass es im Allgemeinen ausreicht, wenn einer der beiden Operatoren $a(\cdot,\cdot)$ und $d(\cdot,\cdot)$ symmetrisch ist, zur Vereinfachung der Notation verwenden wir die leicht eingeschränkte Formulierung. Wir verweisen an dieser Stelle darüberhinaus auf den Zusammenhang zwischen Satz 9 und der Morozovschen Komplementbedingung (2.21) aus dem vorherigen Kapitel.

Betrachten wir nun wiederum das System (3.25) im Hinblick auf Satz 9. Sowohl $\mathcal{I}$ als auch $\lambda M \cdot \mathcal{C}$ sind symmetrisch und positiv semidefinit. Die Voraussetzungen (i) für $\mathcal{A} = \mathcal{I}$ und (ii) für $\mathsf{R}(\mathcal{B}) = \mathbb{R}^M$ sind trivialerweise erfüllt.

Für koerzive $\mathcal{D}$, es reicht sogar koerziv auf $\mathsf{N}(\mathcal{B})$, d.h.

$$d(\alpha,\alpha) \geq c_d \|\alpha\|^2_{V_2'},$$

ist Voraussetzung (iii) erfüllt.

Für Regularisierungsoperatoren, die aus der Norm des Funktionenraums bestehen, d.h. für $\Phi(f) = \|f\|_{V_2'}$, gilt

$$d(\alpha,\alpha) = \lambda \|\alpha\|^2_{V_2'}$$

und somit ist $d(\cdot,\cdot)$ koerziv. Allerdings hängt die Konstante K in der Abschätzung (3.28) von λ ab und für verschwindende λs explodieren diese Abschätzungen. Dieses kann praktische Auswirkungen bei der numerische Approximation haben.

Es sei an dieser Stelle bemerkt, dass weitere Funktionale zu Sattelpunktproblemen konstruiert werden können, welches das System (3.25) als notwendige Bedingung aufweisen, wie z.B.

$$\min_\beta \max_\alpha \left(\sqrt{I + \frac{1}{\lambda}\mathcal{K}}\beta - \sqrt{I + \frac{1}{\lambda}\mathcal{K}}^{-1} y \right)^2 - \left(\sqrt{\lambda\mathcal{C}}\alpha - \sqrt{\lambda\mathcal{C}}^{-1}\mathcal{B}^t\beta \right)^2.$$

Betrachten wir die Ableitung dieses Beispiels nach α

$$-\sqrt{\lambda\mathcal{C}}^T \left(\sqrt{\lambda\mathcal{C}}\alpha - \sqrt{\lambda\mathcal{C}}^{-1}\mathcal{B}^t\beta \right) = 0 \quad \Leftrightarrow \quad \mathcal{B}^t\beta - \lambda\mathcal{C}\alpha = 0,$$

so ergibt sich die zweite Gleichung in (3.25). Dessen erste Gleichung ergibt sich mit der Ableitung nach β

$$\begin{aligned} 0 =& \sqrt{I + \frac{1}{\lambda}\mathcal{K}}^T \left(\sqrt{I + \frac{1}{\lambda}\mathcal{K}}\beta - \sqrt{I + \frac{1}{\lambda}\mathcal{K}}^{-1} y \right) \\ &+ \mathcal{B}\sqrt{\lambda\mathcal{C}}^{-T} \left(\sqrt{\lambda\mathcal{C}}\alpha - \sqrt{\lambda\mathcal{C}}^{-1}\mathcal{B}^t\beta \right) \\ =& \left(I + \frac{1}{\lambda}\mathcal{K} \right)\beta - y + \mathcal{B}\alpha - \frac{1}{\lambda}\mathcal{K}\beta \quad \Leftrightarrow \quad \beta + \mathcal{B}\alpha = y. \end{aligned}$$

3.2.5 Darstellung des Regularisierungsoperators in Waveletbasen

Wie wir bisher gesehen haben, kann durch die Darstellung eines Funktionenraumes durch die Eigenvektoren des Regularisierungsoperators eine einfache Diagonalform des Regularisierungsoperators erreicht werden. Die Nutzung von Eigenvektoren ist aber nicht die einzige Möglichkeit um mit orthonormalen Basen zu arbeiten. Wavelets sind eine andere Alternative, und es zeigt sich, dass eine Darstellung in Diagonalform von bestimmten Regularisierungsoperatoren möglich ist. Dies basiert auf Normäquivalenzen zwischen der Norm von bestimmten Funktionenräumen und diskreten Normen der Entwicklung in der Waveletbasis. In [CK04] wird diese Normäquivalenzeigenschaft von Wavelets bei verwandten zweidimensionalen Scattered Data-Problemen zur Regularisierung einer Wavelet-basierten Fehlerquadratmethode verwendet.

Im Folgenden unternehmen wir einige Überlegungen, im welchem Rahmen Wavelets zur Konstruktion von Kernen benutzt werden können und wie diese Kerne darüberhinaus durch eine diskrete Waveletbasis approximativ dargestellt werden können. Für die approximativen Darstellungen können weiterhin Fehlerabschätzungen durchgeführt werden.

Wir zitieren nun [GK00], siehe auch [DK92, Dah97, Sch98], und starten mit einer eindimensionalen Multiskalen-Zerlegung von $L^2 = \bigoplus_{l\geq 0} S_l$. Wir nehmen an, dass die Komplementräume $W_l := S_l \ominus S_{l-1}$ von einer mehrskaligen Waveletbasis ψ_{lj} aufgespannt werden, d.h. $W_l = \operatorname{span}\{\psi_{lj}, j \in \tau_l\}$, wobei τ_l ein Indexset ist, welcher durch eine Verfeinerungsrate des Multiskalenansatzes festgelegt ist. Weiterhin wird angenommen, dass $\{\psi_{lj}, j \in \tau_l\}$ eine Riesz-Basis von W_l ist und dass eine duale Basis $\{\tilde{\psi}_{lj}, j \in \tau_l\}$ von $\tilde{W}_l$ existiert, so dass diese beiden Waveletbasen zueinander biorthogonal sind, d.h. $\langle \psi_{lj}, \tilde{\psi}_{l'j'} \rangle_{L^2} = \delta_{ll'}\delta_{jj'}$ gilt. Somit kann jedes $u \in L^2$ eindeutig dargestellt werden durch

$$u = \sum_{l=0}^{\infty} \sum_{j\in\tau_l} \langle u, \tilde{\psi}_{lj}\rangle \psi_{lj} = \sum_{l=0}^{\infty} \sum_{j\in\tau_l} \langle u, \psi_{lj}\rangle \tilde{\psi}_{lj}.$$

Hierbei geben die Ausdrücke $\langle u, \tilde{\psi}_{lj}\rangle$ die Koeffizienten der Funktion u in der Waveletbasis wieder.

Mittels eines Tensorproduktansatzes werden aus diesen eindimensionalen Basen d-dimensionale Basen erzeugt. Unter gewissen Voraussetzungen [DK92, Dah97, Sch98, GK00] kann nun die folgende Normäquivalenz in Sobolevräumen gezeigt werden:

Satz 10. *Sei $\gamma_* < s < \gamma$ und $u \in H^s(\Omega), u = \sum_{\underline{l}} w_{\underline{l}}, w_{\underline{l}} \in W_{\underline{l}}$, dann gilt die Normäquivalenz*

$$\|u\|_s^2 \approx \sum_{\underline{l}} 2^{2s|\underline{l}|_\infty} \|w_{\underline{l}}\|_{L^2}^2 = \sum_{\underline{l}} 2^{2s|\underline{l}|_\infty} \sum_{\underline{j}\in\tau_{\underline{l}}} |\langle u, \tilde{\psi}_{\underline{l}\underline{j}}\rangle|^2 = \sum_{\underline{l}} 2^{2s|\underline{l}|_\infty} \sum_{\underline{j}\in\tau_{\underline{l}}} |\langle u, \psi_{\underline{l}\underline{j}}\rangle|^2.$$

Somit lässt sich bezüglich einer Waveletbasis die Regularisierungsmatrix $\mathcal{C}$ darstellen als

$$\mathcal{C} = \operatorname{diag}(2^{2s|\underline{l}|_\infty}).$$

Für die Kerne ergibt sich dadurch die Darstellung nach (3.2.2)

$$k(\underline{x}, \underline{y}) = \sum_{\underline{l}} 2^{-2s|\underline{l}|_\infty} \sum_{\underline{j}\in\tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x})\psi_{\underline{l}\underline{j}}(\underline{y}).$$

Eine entsprechende Normäquivalenz für Sobolevräume mit dominierender gemischter Ableitung H^s_{mix} wird in [GK00] gezeigt

$$\|u\|^2_{H^s_{mix}} \approx \sum_{\underline{l}} 2^{2s|\underline{l}|_1} \|w_{\underline{l}}\|^2_{L^2} = \sum_{\underline{l}} 2^{2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} |\langle u, \tilde{\psi}_{\underline{l}\underline{j}}\rangle|^2 = \sum_{\underline{l}} 2^{2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} |\langle u, \psi_{\underline{l}\underline{j}}\rangle|^2.$$

Mit der H^s_{mix}-Norm als Regularisierungsterm ergibt sich somit eine analoge Darstellung der Kerne

$$k(\underline{x},\underline{y}) = \sum_{\underline{l}} 2^{-2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x})\psi_{\underline{l}\underline{j}}(\underline{y}). \tag{3.29}$$

Es sei an dieser Stelle daran erinnert, dass H^s_{mix}-Räume für $s \geq 1$ die stetige Punktauswertung erlauben und damit eine Kerndarstellung ermöglichen. Bei normalen Sobolevräumen H^s ist die Punktauswertung erst für $s > d/2$ beschränkt.

Eine diskrete Berechnung des Kerns beziehungsweise der Kernmatrix kann nun dadurch erreicht werden, dass von der kompletten Waveletbasis des kontinuierlichen Funktionenraums in eine endlich dimensionale Waveletbasis übergegangen wird. Das heißt, es wird nur die Darstellung bis zu einem gewissen Level l_{max} der Waveletauflösung benutzt.

$$k^{l_{max}}(\underline{x},\underline{y}) := \sum_{|\underline{l}|_\infty \leq l_{max}} 2^{-2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x})\psi_{\underline{l}\underline{j}}(\underline{y}). \tag{3.30}$$

Auf eine Erweiterung auf anisotrope Basen durch einen maximalen Level-Multi-Index $\underline{l}_{max}$ verzichten wir an dieser Stelle aus Gründen der Übersichtlichkeit. Wir betrachten hier und im Folgenden die H^s_{mix}-Norm, für die H^s-Norm gelten entsprechende Darstellungen und Aussagen. Es sei an dieser Stelle auch nochmal auf den Effekt der Regularisierung durch Diskretisierung verwiesen, siehe dazu Abschnitt (2.2.1) und weiterhin Abschnitt (2.2.3) für eine gemeinsame Betrachtung von Regularisierung und Diskretisierung.

Betrachten wir den Fehler $k^\varepsilon(\underline{x},\underline{y}) := k(\underline{x},\underline{y}) - k^{l_{max}}(\underline{x},\underline{y})$, der durch die Nutzung einer endlich dimensionalen Basis für die Darstellung der Kernfunktionen entsteht:

$$k^\varepsilon(\underline{x},\underline{y}) = k(\underline{x},\underline{y}) - k^{l_{max}}(\underline{x},\underline{y}) = \sum_{|\underline{l}|_\infty > l_{max}} 2^{-2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x})\psi_{\underline{l}\underline{j}}(\underline{y}).$$

Sofern wir annehmen, dass die Funktionsauswertung eines Wavelets durch $\sqrt{C}$ beschränkt ist, d.h. $\|\psi_{\underline{l}\underline{j}}(\underline{y})\|_\infty \leq \sqrt{C}$, gilt

$$\left|k^\varepsilon(\underline{x},\underline{y})\right| \leq \sum_{|\underline{l}|_\infty > l_{max}} 2^{-2s|\underline{l}|_1} \cdot C \cdot |\tau_{\underline{l}}| \leq C \sum_{|\underline{l}|_\infty > l_{max}} 2^{-2s|\underline{l}|_1} \cdot 2^{|\underline{l}-\underline{1}|_1} = \frac{C}{2^d} \sum_{|\underline{l}|_\infty > l_{max}} 2^{(1-2s)|\underline{l}|_1}, \tag{3.31}$$

wobei $|\tau_{\underline{l}}|$ die Anzahl der Elemente des Indexset $\tau_{\underline{l}}$ und damit des Komplementraums $W_{\underline{l}}$ angibt. Im Weiteren enthält die Konstante C den Faktor 2^{-d}. In einer Dimension lässt sich dieser Ausdruck mit Hilfe der geometrischen Reihe abschätzen:

$$\left|k^\varepsilon(\underline{x},\underline{y})\right| \leq C \sum_{l > l_{max}} 2^{(1-2s)l} \leq C \frac{2^{(1-2s)(l_{max}+1)}}{1-2^{(1-2s)}} = C \cdot 2^{(1-2s)l_{max}} \frac{2^{1-2s}}{1-2^{(1-2s)}}.$$

Für $s = 1$ ergibt sich somit

$$\left|k^{\varepsilon}(\underline{x}, \underline{y})\right| \leq C \cdot 2^{-l_{max}}.$$

Offensichtlich wird der Fehler mit der Größe der benutzten Basis immer kleiner. Für den d-dimensionalen Fall kann die Summe ebenso über l statt über $\underline{l}$ laufen, sofern die Summe in ihre eindimensionalen Komponenten aufgeteilt wird. Es gilt

$$\begin{aligned}
\left|k^{\varepsilon}(\underline{x}, \underline{y})\right| &\leq C \sum_{|\underline{l}|_\infty > l_{max}} 2^{(1-2s)|\underline{l}|_1} \\
&= C \sum_{\underline{l}=\underline{0}}^{\infty} 2^{(1-2s)|\underline{l}|_1} - C \sum_{|\underline{l}|_\infty \leq l_{max}} 2^{(1-2s)|\underline{l}|_1} \\
&= C \left(\sum_{l=0}^{\infty} 2^{(1-2s)l} \right)^d - C \left(\sum_{l=0}^{l_{max}} 2^{(1-2s)l} \right)^d \\
&= C \left(\frac{1}{1 - 2^{(1-2s)}} \right)^d - C \left(\frac{1 - 2^{(1-2s)(l_{max}+1)}}{1 - 2^{(1-2s)}} \right)^d \\
&= C \frac{1}{\left(1 - 2^{(1-2s)}\right)^d} \left(1 - \left(1 - 2^{(1-2s)(l_{max}+1)} \right)^d \right) \\
&\leq C \frac{d}{\left(1 - 2^{(1-2s)}\right)^d} 2^{(1-2s)(l_{max}+1)} \\
&\leq C(|\psi|_\infty, s, d) \cdot 2^{(1-2s) l_{\max}}. \qquad (3.32)
\end{aligned}$$

Für $s = 1$, d.h. für die approximative Kerndarstellung der H^1_{mix}-Norm, ergibt sich somit die Abschätzung

$$\left|k^{\varepsilon}(\underline{x}, \underline{y})\right| \leq C \cdot 2^{-l_{max}}.$$

Betrachten wir nun die Lösung $\tilde{\beta}$ der Gleichung mit approximativem $\mathcal{K}^{l_{max}}$ und stellen die Gleichung so um, dass sie sich Gleichung mit exaktem $\mathcal{K}$ aber mit gestörter rechter Seite ergibt, hierbei bezeichnet $\mathcal{K}^{\varepsilon}\tilde{\beta} := \mathcal{K}\tilde{\beta} - \mathcal{K}^{l_{max}}\tilde{\beta}$

$$\begin{aligned}
& (\mathcal{I} + \frac{1}{\lambda M}\mathcal{K}^{l_{max}})\tilde{\beta} = y \\
\iff \quad & (\mathcal{I} + \frac{1}{\lambda M}\mathcal{K} - \frac{1}{\lambda M}\mathcal{K}^{\varepsilon})\tilde{\beta} = y \\
\iff \quad & (\mathcal{I} + \frac{1}{\lambda M}\mathcal{K})\tilde{\beta} = y + \frac{1}{\lambda M}\mathcal{K}^{\varepsilon}\tilde{\beta} \\
\iff \quad & (\mathcal{I} + \frac{1}{\lambda M}\mathcal{K})(\tilde{\beta} - \beta) = \frac{1}{\lambda M}\mathcal{K}^{\varepsilon}\tilde{\beta}.
\end{aligned}$$

So kann $\mathcal{K}^{\varepsilon}$ nun z.B. mittels der Spaltenbetragssummennorm abgeschätzt werden

$$\|\mathcal{K}^{\varepsilon}\|_1 \leq \max_{1 \leq j \leq M} \sum_{i=1}^{M} |k^{\varepsilon}_{ij}| \leq M \cdot C \cdot 2^{-l_{max}}.$$

Wir betrachten hier auf Grund der einfacheren Darstellung die H^1_{mix}-Norm. Die Abschätzung gilt offensichtlich entsprechend für die Zeilenbetragssummennorm $\| \cdot \|_\infty$, und mittels der Normbeziehung

$$\|\mathcal{A}\|_2^2 \leq \|\mathcal{A}\|_1 \|\mathcal{A}\|_\infty$$

analog auch für die $\| \cdot \|_2$-Norm.

Somit kann der relative Fehler in Teilen abgeschätzt werden,

$$\frac{\|\tilde{\beta} - \beta\|}{\|\tilde{\beta}\|} \leq \|(\mathcal{I} + \frac{1}{\lambda M}\mathcal{K})^{-1}\| \frac{1}{\lambda M} \|\mathcal{K}^\varepsilon\|.$$

Um weitere Abschätzungen unternehmen zu können, fehlen uns an dieser Stelle Aussagen zur Norm von $(\mathcal{I} + \frac{1}{\lambda M}\mathcal{K})^{-1}$.

Betrachten wir nun noch einmal die Darstellung (3.29) der Kerne in der Waveletbasis

$$k(\underline{x}, \underline{y}) = \sum_{\underline{l}} 2^{-2s|\underline{l}|_1} \sum_{\underline{j} \in \tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x}) \psi_{\underline{l}\underline{j}}(\underline{y}).$$

Ab einer gewisser Diskretisierungsstufe m, es könnte auch hier genauer mit einer anisotropen Tiefe $\underline{m}$ gearbeitet werden, wird der Fall eintreten, dass im Träger einer Waveletfunktion $\psi_{\underline{l}\underline{j}}$ mit $\underline{l} > m$ nur noch höchstens ein Datenpunkt liegt. Umgekehrt bedeutet dies auch, dass für jeden Datenpunkt $\underline{x}$ und für jeden Multi-Index $\underline{l} > m$ dann nur noch eine Funktion $\psi_{\underline{l}\underline{j}}$ eine Auswertung ungleich Null hat

$$\forall \underline{x} \; \exists^1 \; \underline{j} \in \tau_{\underline{l}} \text{ mit } \psi_{\underline{l}\underline{j}}(\underline{x}) \neq 0 \quad \text{für } \underline{l} > m. \tag{3.33}$$

Die Diskretisierungsauflösung m ist hierbei von der jeweiligen Datenmenge, d.h. deren Größe und Verteilung, abhängig. Es sei bemerkt, dass diese Beobachtung weiterhin bedeutet, dass die in (3.31) verwendete Relation

$$\sum_{\underline{j} \in \tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x}) \psi_{\underline{l}\underline{j}}(\underline{y}) \leq C \cdot |\tau_{\underline{l}}|$$

für $|\underline{l}|_\infty \to m$ eine eher grobe Abschätzung ist.

Für die Einträge der Kernmatrix bedeutet (3.33) nun, dass, wenn überhaupt, ab dieser diskreten Auflösung höchstens noch Einträge der Form

$$\sum_{\underline{l} > m} 2^{-2s|\underline{l}|_1} \psi_{\underline{l}\underline{j}}(\underline{x})^2$$

auf der Diagonalen der Kernmatrix addiert werden. Ab diesem Level lässt sich also die Differenz zwischen approximativer Kernmatrix und voller Kernmatrix als eine Störung auf

der Diagonalen darstellen. Dann gilt für die Fehlermatrix

$$\begin{aligned}
\|\mathcal{K}^\varepsilon\|_1 = \|\mathcal{K} - \mathcal{K}^{\underline{m}}\|_1 &\leq \max_{1\leq j\leq M} \sum_{i=1}^{M} |k_{ij}^\varepsilon| = \max_{1\leq j\leq M} |k_{jj}^\varepsilon| \\
&\leq C \cdot \sum_{|\underline{l}|_\infty > m} 2^{-2s|\underline{l}|_1} \\
&= C \sum_{\underline{l}=\underline{0}}^{\infty} 2^{(-2s)|\underline{l}|_1} - C \sum_{|\underline{l}|_\infty \leq l_{max}} 2^{(-2s)|\underline{l}|_1} \\
&= C \left(\sum_{l=0}^{\infty} 2^{(-2s)l} \right)^d - C \left(\sum_{l=0}^{l_{max}} 2^{(-2s)l} \right)^d \\
&= C \left(\frac{1}{1-2^{(-2s)}} \right)^d - C \left(\frac{1-2^{(-2s)(l_{max}+1)}}{1-2^{(-2s)}} \right)^d \\
&= C \frac{1}{\left(1-2^{(-2s)}\right)^d} \left(1 - \left(1 - 2^{(-2s)(l_{max}+1)} \right)^d \right) \\
&\leq C \frac{d}{\left(1-2^{(-2s)}\right)^d} 2^{(-2s)(l_{max}+1)} \\
&\leq C(|\psi|_\infty, s, d) \cdot 2^{(-2s)l_{\max}}
\end{aligned}$$

Es treten somit zwei unterschiedliche Fehlerverhalten auf. Eines, das bis zur Auflösung der Punktverteilung durch die diskrete Waveletbasis vorherrscht, und eines, das danach den Fehler bestimmt. Das letztere weist bezüglich der Diskretisierungsauflösung eine um eins höhere Ordnung auf und hat auch eine kleinere Konstanten in d, die wir hier in den Abschätzungen allerdings nicht genauer angegeben haben.

Es sei an dieser Stelle auf [RMC03] hingewiesen, eine Arbeit in der Waveletkerne im Rahmen des maschinellen Lernens exemplarisch in einer Dimension genutzt werden. Ausgehend von Satz 7 über Tensorprodukte von HRK und einem eindimensionalen Waveletkern der allgemeinen Form

$$k(x,y) = \sum_{l,m,n} \alpha_{lm}\alpha_{ln} \cdot \varphi_{lm}(x)\varphi_{ln}(y)$$

kann ein mehrdimensionaler Kern für einen Tensorproduktraum definiert werden. Die Größe von Waveletbasen für Funktionenräume skaliert im Allgemeinen exponentiell in d, der Fluch der Dimension. Durch die Nutzung der Tensorproduktstruktur eines Raumes ist es aber nun möglich, Waveletkerne zu konstruieren, welche nur linear mit der Dimension skalieren. Dazu wird die Produktdarstellung des Kerns nach Satz 7 genutzt, d.h.

$$k_d(\underline{x},\underline{y}) = \prod_{i=1}^{d} k(x_i, y_i). \tag{3.34}$$

Wie mit einfacher Rechnung zu sehen ist, ergibt sich für Kerne dieser Form für $\left|k^\varepsilon(\underline{x},\underline{y})\right|$ eine zu (3.31) analoge Fehlerabschätzung. Das Bemerkenswerte an dieser Darstellung ist der

Aufwand zur Berechnung eines Eintrags $k^{l_{max}}(\underline{x},\underline{y})$. Die Anzahl der Wavelets in der diskreten Basis skaliert exponentiell in d und somit skaliert in der ursprünglichen Form (3.30) auch der Aufwand zur Berechnung von $k^{l_{max}}(\underline{x},\underline{y})$ exponentiell in d. In der Darstellung (3.34) skaliert der Aufwand aber nur noch linear in d. Betrachten wir den eindimensionalen Kern

$$k^{l_{max}}(x,y) = \sum_{l\leq l_{max}} 2^{-2sl} \sum_{j\in\tau_l} \psi_{lj}(x)\psi_{lj}(y). \tag{3.35}$$

Die Berechnung von $\sum_{l\leq l_{max}}\sum_{j\in\tau_l} 1$ Ausdrücken ist hierbei notwendig. Mit $|\tau_l| = O(2^l)$ ergibt sich $\sum_{l=0}^{l_{max}} O(2^l) = O(2^{l_{max}})$ als Komplexität für die Berechnung eines eindimensionalen Kerns. Damit ergibt sich für einen Produktkern (3.34) mit eindimensionalen Kernen (3.35) ein Aufwand von $O(d\cdot 2^{l_{max}})$.

Es ist zu bemerken, dass das in [RMC03] vorgestellte Verfahren dort nur in einer Dimension angewendet wird. Zudem wird die Normäquivalenz der Sobolevnormen H^s beziehungsweise H^s_{mix} in Waveletbasen nicht genutzt, um eine natürliche Wahl der $\alpha_{i,j}$ zu erlangen. Weiterhin erfolgt die Bestimmung der Lösung des Problems (3.1) durch eine Reihe von Problemen in der Multiskalenzerlegung des Raumes, allerdings mit einem für jede Skala variablem λ. Der Ansatz ist in dieser Form durch die Theorie nicht fundiert, da nun auf jeder Skala eine andere Gleichung gelöst wird. Das in [RMC03] präsentierte Verfahren ist so eher als Ensemble Methode [Die00] zu betrachten, bei denen aus verschiedenen Einzellösungen des maschinellen Lernproblems eine Gesamtlösung geeignet zusammengesetzt wird.

Erlauben wir uns an dieser Stelle einen Vorgriff auf das nächste Kapitel, welches sich mit dünnen Gittern beschäftigt. Wenn wir im diskreten Waveletkern (3.30) die $|\cdot|_\infty$-Norm durch die $|\cdot|_1$-Norm ersetzen, erhalten wir mit der Darstellung

$$k_1^{l_{max}}(\underline{x},\underline{y}) := \sum_{|\underline{l}|_1\leq l_{max}} 2^{-2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x})\psi_{\underline{l}\underline{j}}(\underline{y}) \tag{3.36}$$

einen diskreten Waveletkern, den wir Dünngitter-Waveletkern in Anlehnung an die Definition von dünnen Gittern (4.12) nennen. Wie wir im nächsten Kapitel sehen werden skaliert die Anzahl der Funktionen in einer Dünngitter-Waveletbasis mit $O(2^{l_{max}}\cdot l_{max}^{d-1})$. Für den Fehler k_1^ε bei der Verwendung dieser diskreten Basis ergibt sich analog zu (3.32) $O(2^{(1-2s)l_{\max}}\cdot l_{max}^{d-1})$ mit Beweistechniken wie sie in [Bun98, BG99, BG04] verwendet werden. Somit wird zwar im Vergleich zu (3.30), wo wir eine Komplexität von $(O(2^{l_{max}\cdot d})$ feststellen, die Komplexität deutlich verringert, bei etwas schlechteren Approximationsaussagen. Allerdings haben wir durch die Ausnutzung des Produktkerns (3.34) und (3.35) den Aufwand deutlich weiter verringert.

Andererseits können in (3.36) auch Produktkerne verwendet werden. Als weitere Eigenschaft von HRK benötigen wir dazu die Feststellung, dass die Summe von HRK und deren zugehörigen Kernen wiederum ein HRK mit der Summe dieser Kerne als Kern ist, sofern zwei HRK nur die Nullfunktion gemeinsam haben [Aro50]. Diese Bedingung ist bei den hier verwendeten Komplementräumen $W_{\underline{l}}$ gegeben. Somit kann der Kern (3.36) dargestellt werden als

$$k_1^{l_{max}}(\underline{x},\underline{y}) = \sum_{|\underline{l}|_1\leq l_{max}} 2^{-2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x})\psi_{\underline{l}\underline{j}}(\underline{y}) = \sum_{|\underline{l}|_1\leq l_{max}} k^{\underline{l}}(\underline{x},\underline{y})$$

mit

$$k^{\underline{l}}(\underline{x},\underline{y}) := 2^{-2s|\underline{l}|_1} \sum_{\underline{j}\in\tau_{\underline{l}}} \psi_{\underline{l}\underline{j}}(\underline{x})\psi_{\underline{l}\underline{j}}(\underline{y}) = \prod_{i=1}^{d} k^{l_i}(x_i,y_i)$$

und

$$k^{l}(x,y) := 2^{-2sl} \sum_{j\in\tau_l} \psi_{lj}(x)\psi_{lj}(y).$$

Der Aufwand zur Berechnung des Dünngitter-Waveletkerns in dieser Form ergibt sich mit $|\tau_l| \leq C \cdot 2^l$ als

$$\sum_{|\underline{l}|_1\leq l_{max}} \sum_{i=1}^{d} 2^{l_i} = \sum_{k=0}^{l_{max}} \sum_{|\underline{l}|_1=k} \sum_{i=1}^{d} 2^{l_i} \leq \sum_{k=0}^{l_{max}} d\cdot 2^k = O(d\cdot 2^{l_{max}}),$$

wobei $\sum_{|\underline{l}|_1=k} \sum_{i=1}^{d} 2^{l_i} \leq d\cdot 2^k$ per Induktion über d folgt. Die Verwendung von dünnen Gittern, die wie wir sehen werden bei der Diskretisierung eines Funktionenraums mit einem Finite Element-Ansatz deutliche Vorteile aufweisen, bringt somit bei der diskreten Waveletkern-Darstellung keine Vorteile.

Im Kernansatz besteht eine Möglichkeit, eine Funktion im unendlich dimensionalen Funktionenraum durch eine endliche Linearkombination darzustellen. Allerdings können resultierende Verfahren sehr aufwendig sein, je nach Ansatz zwischen empirisch gemessenen $O(M^{2.1})$ und $O(M^3)$ [RYP03]. Der Aufwand zum Lösen des System beträgt somit mindestens $O(M^2)$, bei großen Datenmengen ein nicht zu unterschätzender Aufwand. Ansätze, dieses Problem zu beheben, bestehen in der Auswahl nur einiger Datenpunkte und deren zugehörigen Kernfunktionen, was zu einer Approximation des ursprünglichen Problems führt.

Wir haben mit Wavelets eine Möglichkeit vorgestellt, den eigentlichen Funktionenraum zu approximieren. Sofern eine Sobolevnorm zur Regularisierung verwendet wird, kann mit geeigneten Wavelets und Normäquivalenzeigenschaften die Sobolevnorm als Regularisierungsoperator in der Form (3.16)

$$\Phi(f) = \sum_{j=1}^{\infty} \alpha_j^2 \gamma_j$$

dargestellt werden und dadurch mit Hilfe der Kernmatrix eine einfache Darstellung des Problems erhalten werden. Die Lösungsfunktion kann durch Kerne dargestellt werden, wobei diese durch eine diskrete Waveletbasis dargestellt werden. Dies resultiert aber weiterhin in ein Verfahren, das nichtlinear mit der Zahl der Datenpunkte skaliert.

Allerdings kann eine diskrete Waveletbasis auch dazu benutzt werden, das Problem im Funktionenraum mittels Gleichung (3.7) zu lösen, d.h. zu behandeln ist nun das lineare Gleichungssystem (3.13)

$$(\mathcal{B}^t\mathcal{B} + \lambda M\cdot\mathcal{C})\alpha = \mathcal{B}^t y.$$

Andere Ansätze wie z.B. Finite Elemente ermöglichen ebenfalls eine diskrete Darstellung eines Funktionenraums. Insbesondere für große Datenbestände können sich dadurch Vorteile ergeben, da im eigentlichen Lösungsprozess dann nicht mit einer $M\times M$-Matrix

gearbeitet werden muss, sondern mit einer $N \times N$-Matrix, wobei N hier die Größe des diskreten Funktionenraums angibt. Allerdings gilt sowohl für die normale Waveletbasis als auch für normale Finite Element-Ansätze der Fluch der Dimension, der Aufwand des Verfahrens steigt exponentiell mit der Zahl der Dimensionen. Eine Diskretisierungstechnik, die dem Fluch der Dimension in weit geringeren Maße unterliegt, sind die so genannten *dünnen Gitter* [Zen91]. Diese betrachten wir im folgenden Kapitel eingehend.

Kapitel 4

Diskretisierung mit dünnen Gittern

Für die numerischen Behandlung von Funktionen aus einem unendlich dimensionalen Funktionenraum V ist eine diskrete Darstellung im Rechner erforderlich. Diskret bedeutet hierbei, das eine Funktion f in einem nun endlich dimensionalen linearen Funktionenraum V_n durch einen endlichen Satz von Funktionen, der Basis des diskreten Raumes, dargestellt wird. Das heißt, es existiert eine Darstellung der Form

$$f(\underline{x}) = \sum_{i=1}^{n} \alpha_i \varphi_i(\underline{x}),$$

wobei die Funktionen φ_i eine geeignete Basis des diskreten Funktionenraumes V_n bilden.

In den bisherigen Kapiteln haben wir nicht spezifiziert, welchen endlich dimensionalen Teilraum V_n und welchen Typ von Basisfunktionen $\{\varphi_i\}_{i=1}^n$ wir benutzen wollen. Im Bereich der numerischen Behandlung von partiellen Differentialgleichungen spielen Finite Element-Räume eine sehr wichtige Rolle, hier kann z.B. die Konvergenz der diskreten Lösung zur kontinuierlichen Lösung aus Sobolev-Räumen gezeigt werden. Andere Möglichkeiten sind Wavelet-Basen mit gewissen Bestapproximationseigenschaften in Besov-Räumen oder Spline-Funktionen.

All diese gitterorientierten Diskretisierungstechniken zeigen eine exponentiell mit der Zahl der Dimensionen wachsende Komplexität, was sich in einer exponentiell wachsenden Anzahl von Basisfunktionen ausdrückt. Dieser Effekt wird häufig als Fluch der Dimension bezeichnet. Alternative Ansätze bestehen in gitterlosen Diskretisierungstechniken wie z.B. radialen Basisfunktionen, bei denen zwar die Anzahl der Basisfunktionen nicht, aber die Komplexität des Gesamtverfahrens weiterhin exponentiell mit der Zahl der Dimensionen wächst.

Der Ansatz der dünnen Gitter [Zen91] zur Funktionsapproximation weist hingegen einen deutlich langsameren Anstieg der Komplexität in der Anzahl der Basisfunktionen im Vergleich zu den erwähnten gitterorientierten Verfahren auf. Wir realisieren in dieser Arbeit ein dünnes Gitter durch die Kombinationstechnik, es wird dabei eine Sequenz von konventionellen Gittern mit uniformen Maschenweiten in jeder Koordinatenrichtung benötigt. Aus deren Kombination ergibt sich dann eine Funktion aus einem Dünngitterraum.

Wir werden im Folgenden neben der Methode der dünnen Gitter und der Kombinationstechnik auch eine Erweiterung des Ansatz mittels verallgemeinerten dünnen Gittern und einer dimensionsadaptiven Kombinationstechnik beschreiben. Darüberhinaus stellen wir dar, wie mit einer Hierarchie, welcher mit der konstanten Funktion startet, Funktionen in noch größeren Dimensionen dargestellt werden können.

4.1 Dünne Gitter und die Kombinationstechnik

Die Methode der dünnen Gitter ist eine spezielle Diskretisierungstechnik, die es erlaubt, zu einem gewissen Grad die Komplexität des Problems in Bezug auf die Zahl der Dimensionen zu beherrschen. Die Idee basiert auf der hierarchischen Basis [Fab09, Yse86, Yse92], eine zur üblichen nodalen Basis äquivalente Darstellungsmöglichkeit eines diskreten Funktionenraumes. Die Methode wurde ursprüngliche für die Lösung elliptischer partieller Differentialgleichungen [Zen91, Bun92, GSZ92, Bal94, Ach03] entwickelt und wird nun auch erfolgreich für Integralgleichungen [FHP96, GOS99], Interpolation und Approximation [Bas85, Tem89, SS99, GK00, Kna00], Eigenwertprobleme [GG00] und Integrationsprobleme [GG98, GG03] eingesetzt. Weiterhin werden in neueren Arbeiten auch stochastische Differentialgleichungen [ST03a, ST03b], Differentialformen im Rahmen der Maxwell-Gleichung [GH03] und parabolische Gleichungen für Optionspreisaufgaben [Rei04] mit Dünngitteransätzen behandelt. Mit einer Wavelet-basierten Dünngitterdiskretisierung werden in [vPS04] parabolische Probleme behandelt. Neben Galerkin Finite Element-Ansätzen gibt es auch Arbeiten mit Finiten Differenzen auf dünnen Gittern [Gri98, Sch99, Kos02, GK03] und Finite Volumen-Ansätze [Hem95].

In [BG04] ist ein Übersichtsartikel zum Thema dünne Gitter zu finden.

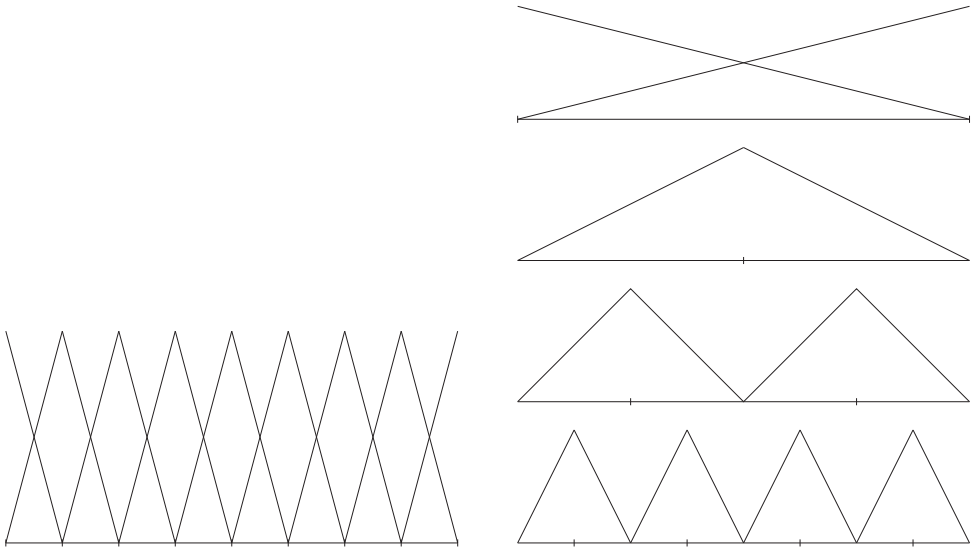
Die Idee kann bis zum russischen Mathematiker Smolyak [Smo63] zurückverfolgt werden, der das Prinzip bei der numerischen Integration nutzt. Bekannt ist das den dünnen Gittern zu Grunde liegende Konzept auch unter den Namen Hyperbolische Kreuze [Bab60, Tem89, Tem93a, Tem93b], boolean methods [Del82, DS89], discrete blending methods [BDJ92] und splitting extrapolation methods [LLS95].

Für ein d-dimensionales Problem benötigt der Dünngitteransatz $O(h_n^{-1} \cdot \log(h_n^{-1})^{d-1})$ Gitterpunkte im Diskretisierungsprozess, wobei h_n die Maschenweite angibt. Es lässt sich zeigen, dass eine Genauigkeit von $O(h_n^2 \cdot \log(h_n^{-1})^{d-1})$ punktweise oder im Bezug auf die L^2- oder L^∞-Norm erzielt werden kann, vorausgesetzt, dass die Lösung genügend glatt ist. Dies steht im Gegensatz zu konventionellen Vollgittermethoden, die $O(h_n^{-d})$ Punkte für eine Genauigkeit von $O(h_n^2)$ benötigen. Somit lässt sich die Dünngittermethode für höherdimensionale Probleme einsetzen. Der Fluch der Dimension, der die Vollgittermethoden betrifft, tritt bei der Dünngittertechnik in einem viel geringerem Ausmaß auf.

Zur Vereinfachung der Darstellung beschränken wir uns im Folgenden auf den Fall $\Omega = [0,1]^d$. Diese Situation lässt sich bei beschränkten Rechteckgebieten immer durch eine geeignete Reskalierung des Raumes erreichen.

4.1.1 Nodale Basis

Eine konventionelle Finite Element-Diskretisierung verwendet ein Gitter $\Omega_{\underline{l}}$ auf $\bar{\Omega}$, mit $\underline{l} = (l_1, \ldots, l_d) \in \mathbb{N}^d$ einem Multi-Index und Gitterweiten $h_{\underline{l}} := (h_{l_1}, \ldots, h_{l_d}) := (2^{-l_1}, \ldots, 2^{-l_d})$.



(a) Nodale Basis für V_3 (b) Hierarchische Basis für V_3

Abbildung 4.1. *Nodale und hierarchische Basis der Stufe 3*

Hier ist $\Omega_{\underline{l}}$ äquidistant in jeder Koordinatenrichtung, hat aber im Allgemeinen verschiedene Maschenweiten in den Koordinatenrichtungen. Im Gitter $\Omega_{\underline{l}}$ sind die Punkte

$$x_{\underline{l},\underline{j}} \;:=\; (x_{l_1,j_1},\ldots,x_{l_d,j_d}) \tag{4.1}$$

enthalten, mit $x_{l_t,j_t} := j_t \cdot h_{l_t} = j_t \cdot 2^{-l_t}$ und $j_t = 0,\ldots,2^{l_t}$. Auf jedem Gitter $\Omega_{\underline{l}}$ definieren wir den Raum $V_{\underline{l}}$ der stückweise d-linearen Funktionen:

$$V_{\underline{l}} \;:=\; \operatorname{span}\{\phi_{\underline{l},\underline{j}} \mid j_t = 0,\ldots,2^{l_t},\ t = 1,\ldots,d\}, \tag{4.2}$$

der durch die übliche Basis der d-dimensionalen stückweise d-linearen Hut-Funktionen

$$\phi_{\underline{l},\underline{j}}(\underline{x}) \;:=\; \prod_{t=1}^{d} \phi_{l_t,j_t}(x_t) \tag{4.3}$$

aufgespannt wird. Hierbei sind die eindimensionalen Funktionen $\phi_{l,j}(x)$ mit Träger $[x_{l,j} - h_l, x_{l,j} + h_l] \cap [0,1] = [(j-1)h_l, (j+1)h_l] \cap [0,1]$ gegeben durch:

$$\phi_{l,j}(x) = \begin{cases} 1 - |x/h_l - j|, & x \in [(j-1)h_l, (j+1)h_l] \cap [0,1]; \\ 0, & \text{sonst.} \end{cases} \tag{4.4}$$

Siehe auch Abbildung 4.1(a) für ein eindimensionales Beispiel.

4.1.2 Hierarchische Teilraumzerlegung

In den bisherigen Definitionen und auch im Folgenden bezeichnet der Multi-Index $\underline{l} \in \mathbb{N}^d$ den Level beziehungsweise die Diskretisierungsstufe eines Gitters, Raums oder einer Funktion, wohingegen der Multi-Index $\underline{j} \in \mathbb{N}^d$ die Lage eines gegebenen Gitterpunktes $x_{\underline{l},\underline{j}}$ oder der entsprechenden Basisfunktion $\phi_{\underline{l},\underline{j}}(\underline{x})$ bezeichnet. Nun definieren wir die Differenzräume

$$W_{\underline{l}} \;:=\; V_{\underline{l}} \;\setminus\; \bigcup_{t=1}^{d} V_{\underline{l}-\underline{e}_t}, \tag{4.5}$$

wobei $\underline{e}_t$ den t-ten Einheitsvektor bezeichnet. Um die Definition zu vervollständigen, setzen wir formal $V_{\underline{l}} := \emptyset$, wenn $l_t = -1$ für mindestens ein $t \in \{1, \ldots, d\}$ gilt. Diese hierarchischen Differenzräume erlauben uns die Definition einer Multilevel-Teilraumzerlegung, d.h. die Definition des Raums V_n als die direkte Summe der Teilräume

$$V_n \;:=\; \bigoplus_{l_1=0}^{n} \cdots \bigoplus_{l_d=0}^{n} W_{\underline{l}} \;=\; \bigoplus_{|\underline{l}|_\infty \leq n} W_{\underline{l}}. \tag{4.6}$$

Hier und im Folgenden bezeichnet „ $\leq$ “ die entsprechende elementweise Relation. $|\underline{l}|_\infty := \max_{1 \leq t \leq d} l_t$ und $|\underline{l}|_1 := \sum_{t=1}^{d} l_t$ sind die diskrete L_∞- beziehungsweise die diskrete L_1-Norm von $\underline{l}$. Beachte, dass für die Räume $V_{\underline{l}}$ gilt

$$V_{\underline{l}} \;:=\; \bigoplus_{k_1=0}^{l_1} \cdots \bigoplus_{k_d=0}^{l_d} W_{\underline{k}} = \bigoplus_{\underline{k} \leq \underline{l}} W_{\underline{k}}.$$

Wie man leicht aus (4.2) und (4.5) sehen kann, führt die Einführung der Indexmengen

$$\mathsf{B}_{\underline{l}} \;:=\; \left\{ \underline{j} \in \mathbb{N}^d \;\middle|\; \begin{array}{lll} j_t = 1, \ldots, 2^{l_t} - 1, \quad j_t \text{ ungerade}, & t = 1, \ldots, d, & \text{falls } l_t > 0, \\ j_t = 0, 2^{l_t}, & t = 1, \ldots, d, & \text{falls } l_t = 0 \end{array} \right\} \tag{4.7}$$

zu

$$W_{\underline{l}} \;=\; \operatorname{span}\{\phi_{\underline{l},\underline{j}}, \underline{j} \in \mathsf{B}_{\underline{l}}\}. \tag{4.8}$$

Deswegen ist die Familie von Funktionen

$$\{\phi_{\underline{l},\underline{j}}, \underline{j} \in \mathsf{B}_{\underline{l}}\}_{\underline{l}=\underline{0}}^{\underline{n}} \tag{4.9}$$

gerade eine hierarchische Basis [Fab09, Yse86, Yse92] von V_n, die die eindimensionale hierarchische Basis nach [Fab09], siehe auch Abbildung 4.1(b), auf den d-dimensionalen Fall mit Hilfe eines Tensorproduktansatzes verallgemeinert. Es sei bemerkt, dass die Träger der Basisfunktionen $\phi_{\underline{l},\underline{j}}(\underline{x})$, die $W_{\underline{l}}$ nach (4.8) aufspannen, disjunkt sind. Siehe Abbildung 4.2 für eine Darstellung der Träger der Basisfunktionen aus den Teilräumen W_{l_1,l_2} von $V_{3,3}$.

Nun lässt sich jede Funktion $f \in V_n$ entsprechend mittels

$$f(\underline{x}) \;=\; \sum_{|\underline{l}|_\infty \leq n} \sum_{\underline{j} \in \mathsf{B}_{\underline{l}}} \alpha_{\underline{l},\underline{j}} \cdot \phi_{\underline{l},\underline{j}}(\underline{x}) \tag{4.10}$$

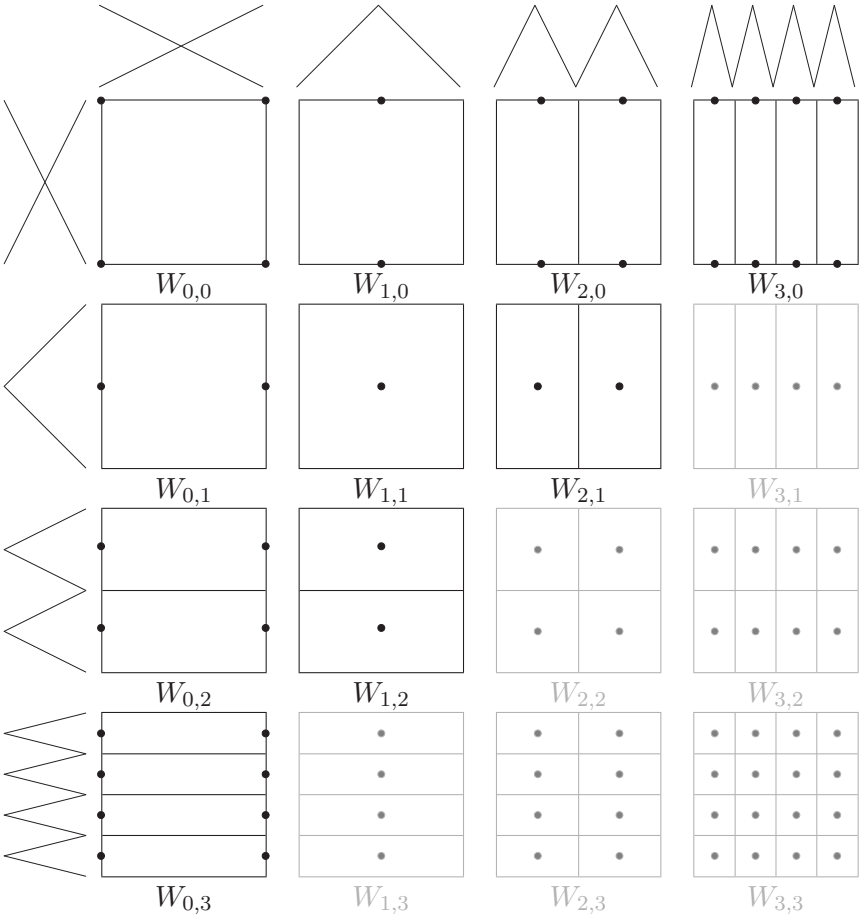


Abbildung 4.2. *Träger der hierarchischen Basisfunktionen des Raumes V_3*

zerlegen, wobei $\alpha_{\underline{l},\underline{j}} \in \mathbb{R}$ die Koeffizienten der Darstellung in der hierarchischen Tensorproduktbasis sind.

Die Zahl der Basisfunktionen, die nötig ist, um ein $f \in V_n$ in nodaler oder hierarchischer Basis darzustellen, beträgt hierbei $(2^n + 1)^d$. Mit einer Auflösung von z.B. 17 Punkten in jeder Dimension, d.h. $n = 4$, würde ein zehndimensionales Problem die Berechnung und Speicherung von ungefähr $2 \cdot 10^{12}$ Koeffizienten benötigen. Dies übersteigt die Möglichkeiten von heutigen Computern.

4.1.3 Dünne Gitter

Zenger zeigt in [Zen91] den Zusammenhang zwischen der Größe des Trägers einer hierarchischen Basisfunktion und dessen Anteil an der Interpolation einer Funktion zunächst für den zweidimensionalen Fall. Für den mehrdimensionalen hierarchischen Interpolanten

$$f_n = \sum_{|\underline{l}|_\infty \leq n} f_{\underline{l}} \quad \text{mit } f_{\underline{l}} \in W_{\underline{l}}$$

einer Funktion f wird in [Bun92] gezeigt, dass unter gewissen Glattheitsvoraussetzungen an die darzustellende Funktion, der Anteil einer hierarchischen Basisfunktion an der Approximation durch die Größe des Trägers dieser Basisfunktion nach oben beschränkt ist, es gilt

$$\|f_{\underline{l}}\|_2 \leq C(|f|_2, d) h_{l_1}^2 \cdot \ldots \cdot h_{l_d}^2 \tag{4.11}$$

mit

$$|f|_2 := \left\| \frac{\partial^{2d} f}{\partial x_1^2 \cdots \partial x_d^2} \right\|_2 .$$

Entsprechende Aussagen gelten für die $\|\cdot\|_\infty$ Norm mit analog definierter $|\cdot|_\infty$ Seminorm.

Die Glattheitsvoraussetzung besteht gerade in der Beschränktheit von $|f|_2$, der gemischten zweiten Ableitungen der Funktion f. Solche Funktionen leben im so genannten *Sobolevraum mit dominierender gemischter Ableitung* H^t_{mix} für $t = 2$, auch *Sobolevraum mit dominierender gemischter Glattheit* genannt, dessen Norm wie folgt definiert ist

$$\|f\|^2_{H^t_{mix}} = \sum_{0 \leq \underline{k} \leq t} \left| \frac{\partial^{|\underline{k}|_1}}{\partial x_{\underline{k}}} u \right|_2^2 .$$

Es sei darauf hingewiesen, dass die Räume H^t_{mix} ebenso wie diskreten Räume $V_{\underline{l}}$ eine Tensorproduktstruktur aufweisen und sich als

$$H^t_{mix} = H^t \otimes \cdots \otimes H^t$$

darstellen lassen [Hoc99, GK00, HKZ00].

Ausgehend von der Fehlerabschätzung (4.11) werden in [Zen91] die so genannten dünnen Gitter definiert, in denen die hierarchischen Basisfunktionen mit kleinem Träger, und somit kleinem Beitrag zur Funktionsdarstellung, nicht mehr zum diskreten Ansatzraum gehören. In der hierarchischen Darstellung wird der Dünngitterraum V_n^s über

$$V_n^s := \bigoplus_{|\underline{l}|_1 \leq n} W_{\underline{l}} \tag{4.12}$$

als Teilraum von V_n definiert. Er wird dadurch erhalten, dass wir $|\underline{l}|_\infty$ durch $|\underline{l}|_1$ in der Teilraumzerlegung (4.6) von V_n ersetzen. In der Abbildung 4.2 sind die im Gegensatz zu (4.6) nun nicht mehr benutzten Teilräume grau dargestellt. Da jedes $f \in V_n$ sich nach (4.10) zerlegen lässt, gilt entsprechend für jede Funktion $f \in V_n^s$

$$f_n^s(\underline{x}) = \sum_{|\underline{l}|_1 \leq n} \sum_{\underline{j} \in \mathsf{B}_{\underline{l}}} \alpha_{\underline{l},\underline{j}} \phi_{\underline{l},\underline{j}}(\underline{x}). \tag{4.13}$$

Die Gitter Ω_n^s, die zu den Approximationsräumen V_n^s gehören, werden dünne Gitter genannt und werden im Detail unter anderem in [Gri91, Zen91, Bun92, GSZ92, Bun98, BG99, BG04] studiert. Beispiele dünner Gitter für den zwei- und dreidimensionalen Fall sind in Abbildung 4.3 gegeben. Dargestellt sind die nach (4.1) enthaltenen Gitterpunkte im Falle der Verwendung einer linearen hierarchischen Basis.

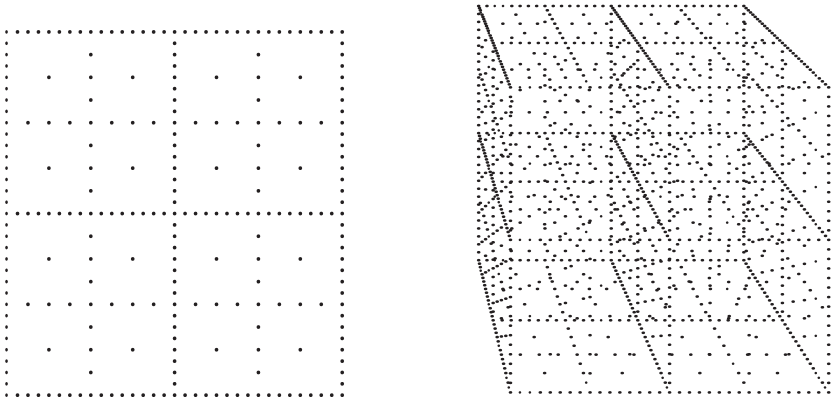


Abbildung 4.3. *Zweidimensionales dünnes Gitter (links) und ein dreidimensionales dünnes Gitter (rechts), $n = 5$.*

Eine einfache Rechnung [BG99] zeigt nun, dass die Dimension des Dünngitter-Raums V_n^s von der Ordnung $O(h_n^{-1} \cdot \log(h_n^{-1})^{d-1})$ ist, beziehungsweise $O(n^{d-1}2^n)$ mit $h_n = 2^{-n}$. Sowohl für Interpolationsprobleme als auch für Approximationsprobleme, die von elliptischen PDEs zweiter Ordnung herrühren, wird gezeigt [GSZ92, BGRZ94a, BGRZ94b], dass die Dünngitter-Lösung f_n^s einer Verfeinerungstiefe n beinahe so genau ist wie die Vollgitterlösung f_n dieses Levels, d.h. der Diskretisierungsfehler bei der Nutzung von dünnen Gittern erfüllt

$$||f - f_n^s||_2 = O(h_n^2 \log(h_n^{-1})^{d-1})$$

im Gegensatz zu

$$||f - f_n||_2 = O(h_n^2)$$

bei einer Vollgitterlösung. Hierbei wird für die dünnen Gitter mit $f \in H_{mix}^2$ eine stärkere Glattheitsbedingung der Lösung vorausgesetzt als beim Vollgitteransatz, wo nur die zweiten Ableitungen für jede Dimension beschränkt sein müssen, d.h. es gilt $f \in H^2$, um Approximationsaussagen der angegebenen Form zu erhalten. Entsprechende Approximationsaussagen für dünne Gitter gelten auch in der Maximumsnorm.

4.1.4 Die Dünngitterkombinationstechnik

Mit der so genannten Kombinationstechnik [GSZ92], die auf multivariater Extrapolation [BGR94] basiert, existiert ein weiteres Verfahren, das mit Funktionen auf dünnen Gittern arbeitet.

Dazu betrachten wir das Problem, sei es eine Interpolationsaufgabe oder die Approximation einer Funktion, auf einer bestimmten Folge von Gittern $\Omega_{\underline{l}}$ mit uniformer Maschenweite $h_t = 2^{-l_t}$ für die t-te Koordinatenrichtung. Diese Gitter besitzen verschiedene Maschenweiten für die verschiedenen Koordinatenrichtungen. Genauer betrachten wir die

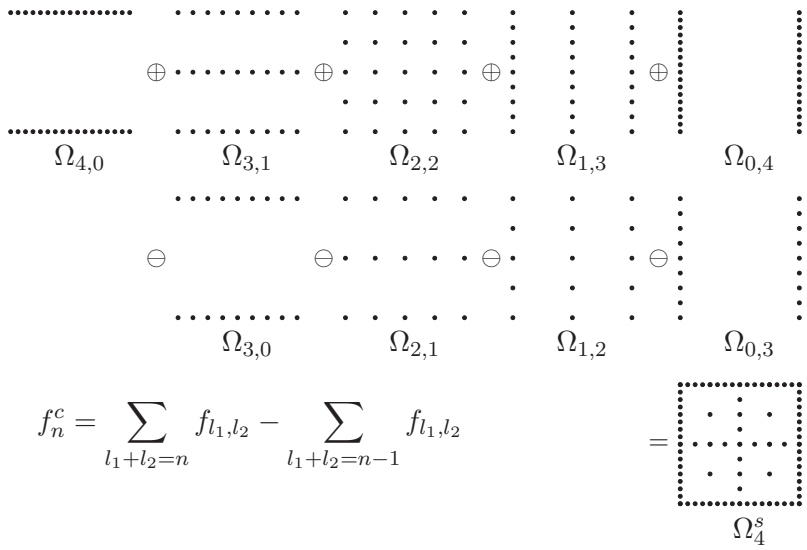


Abbildung 4.4. *Kombinationstechnik der Stufe vier,* $d = 2, n = 4$.

Sequenz von $\Omega_{\underline{l}}$ mit

$$l_1 + \ldots + l_d = n - q, \quad q = 0, \ldots, d-1, \quad l_t \geq 0. \tag{4.14}$$

Abbildung 4.4 zeigt die Gitter Ω_{l_1,l_2} und das resultierende dünne Gitter Ω_4^s, welche für die Kombinationsformel des Levels vier in zwei Dimensionen benutzt werden.

Ein Finite Elemente-Ansatz mit stückweise d-linearen Ansatz- und Testfunktionen $\phi_{\underline{l},\underline{j}}(\underline{x})$ auf dem Gitter $\Omega_{\underline{l}}$ ergibt die Darstellung

$$f_{\underline{l}}(\underline{x}) = \sum_{j_1=0}^{2^{l_1}} \cdots \sum_{j_d=0}^{2^{l_d}} \alpha_{\underline{l},\underline{j}} \phi_{\underline{l},\underline{j}}(\underline{x})$$

auf den lokalen Gittern. Die Funktionen $f_{\underline{l}}$, diskrete lokale Lösungen des zu Grunde liegenden Problems, sind in den Räumen $V_{\underline{l}}$ der stückweise d-linearen Funktionen auf dem Gitter $\Omega_{\underline{l}}$ enthalten, siehe (4.2). Wir kombinieren diese Teilergebnisse $f_{\underline{l}}(\underline{x})$ der verschiedenen Gitter $\Omega_{\underline{l}}$ nun wie folgt zur Dünngitterkombinationslösung f_n^c:

$$f_n^c(\underline{x}) := \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\underline{l}|_1 = n-q} f_{\underline{l}}(\underline{x}). \tag{4.15}$$

Siehe wiederum Abbildung 4.4 für die zweidimensionale Form dieses Ausdrucks. Die resultierende Funktion f_n^c lebt im oben definierten Dünngitter-Raum V_n^s.

Die Kombinationstechnik lässt sich als eine bestimmte multivariate Extrapolationsmethode interpretieren, die auf dem dünnen Gitter arbeitet, siehe [GSZ92, BGR94, PZ99]. In [GSZ92] wird gezeigt, dass der mit der Kombinationstechnik erhaltene Interpolant f_n^c

gleich dem Interpolanten f_n^s im Dünngitterraum V_n^s ist. Allerdings entspricht die Kombinationslösung f_n^c bei der numerischen Behandlung von partiellen Differentialgleichungen im Allgemeinen nicht der Galerkin-Lösung f_n^s, jedoch ist ihre Genauigkeit üblicherweise von der gleichen Ordnung, siehe [GSZ92]. Hierzu ist eine Reihenentwicklung des Fehlers

$$f - f_{\underline{l}} = \sum_{i=1}^{d} \sum_{j_1,\ldots,j_m \subset 1,\ldots,d} c_{j_1,\ldots,j_m}(h_{j_1},\ldots,h_{j_m}) h_{j_1}^p \cdot \ldots \cdot h_{j_m}^p,$$

mit beschränkten $c_{j_1,\ldots,j_m}(h_{j_1},\ldots,h_{j_m})$ notwendig, deren Existenz für PDE-Modellprobleme in [BGRZ94a] gezeigt wird, siehe auch [GSZ92, PZ99, Rei04]. Zu beachten ist, dass all diese Teilprobleme im Vergleich zu herkömmlichen Finite Element-Ansätzen substantiell in ihrer Größe reduziert sind. An Stelle eines Problems der Größe $\dim(V_n) = O(h_n^{-d}) = O(2^{nd})$ müssen wir nun $O(dn^{d-1})$ Probleme der Größe $\dim(V_{\underline{l}}) = O(h_n^{-1}) = O(2^n)$ behandeln. Weiterhin können all diese Probleme unabhängig voneinander betrachtet werden, was eine einfache Parallelisierung ermöglicht, siehe [Gri92]. Hierzu existiert auch eine simple, aber effektive statische Lastbalancierungsstrategie [GHSZ93].

Weiterhin sei bemerkt, dass die Summation über die diskreten Funktionen aus den verschiedenen Räumen $V_{\underline{l}}$ in (4.15) die d-lineare Interpolation benötigt, die gerade der Transformation auf die Darstellung in hierarchischer Basis (4.9) entspricht [GT95, Gri98]. Oft wird jedoch die Funktion f_n^c nicht explizit aufgestellt, sondern stattdessen werden die Lösungen $f_{\underline{l}}$ auf den verschiedenen Gittern $\Omega_{\underline{l}}$, die in der Kombinationsformel auftreten, bereit gehalten. Jede lineare Operation F über f_n^c kann leicht mit Hilfe der Kombinationsformel (4.15) ausgedrückt werden

$$\mathrm{F}(f_n^c) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\underline{l}|_1 = n-q} \mathrm{F}(f_{\underline{l}}), \tag{4.16}$$

wobei direkt auf den Funktionen $f_{\underline{l}}$ gearbeitet wird.

Bei der numerischen Behandlung von partiellen Differentialgleichungen mit Dirichlet-Randwerten existieren auf dem Rand des behandelten Gebietes keine Freiheitsgrade. Gitter $\Omega_{\underline{l}}$ mit $\exists l_i = 0$ haben somit keinen Freiheitsgrad und auch keinen Beitrag bei der Darstellung der Gesamtfunktion, welche im Raum $H^k_{mix,0}$ enthalten ist. Es kann auch bei anderen Anwendungen sinnvoll sein, die Randdominanz des bisher vorgestellten dünnen Gitters zu reduzieren. In der ursprünglichen Arbeit über die Dünngitterkombinationstechnik [GSZ92] werden partielle Differentialgleichungen behandelt. Somit wird dort nicht mit einer Kombinationstechnik der Form (4.14) gearbeitet, stattdessen wird die Sequenz der Gitter bestimmt über

$$l_1 + \ldots + l_d = n + (d-1) - q, \quad q = 0,\ldots,d-1, \quad l_t > 0. \tag{4.17}$$

Dieser Ansatz unterscheidet sich von (4.14) in den Summenergebnissen $n+(d-1)-q$ statt $n-q$, sowie der Einschränkung $l_t > 0$ statt $l_t \geq 0$. Für die Dünngitterfunktion $f_n^{\hat{c}} \in V_n^{\hat{s}}$ ergibt sich somit die Darstellung

$$f_n^{\hat{c}}(\underline{x}) := \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\underline{l}|_1 = n+(d-1)-q} f_{\underline{l}}(\underline{x}). \tag{4.18}$$

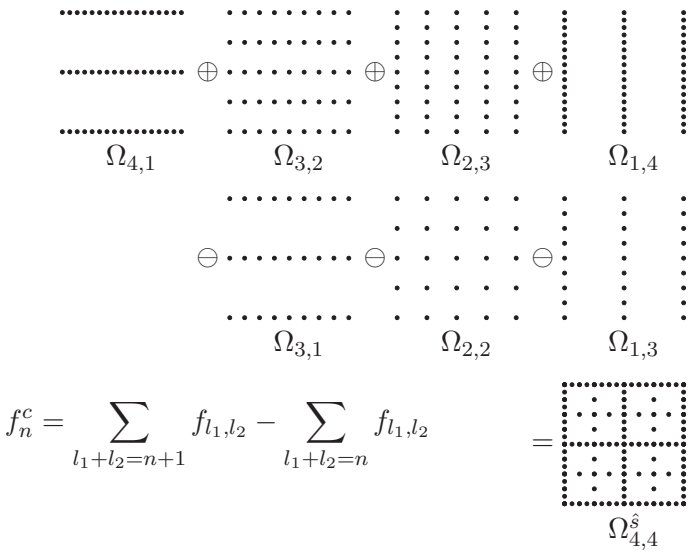


Abbildung 4.5. *Kombinationstechnik der Stufe 4 nach (4.17), $d = 2, n = 4$.*

In Abbildung 4.5 ist ein wiederum zweidimensionales Beispiel dargestellt. Trotz dieser Veränderung in der Auswahl der Gittersequenz ist der minimale Punktabstand weiterhin $h_n = 2^{-n}$. Die erwähnten Komplexitäts- und Approximationsaussagen gelten entsprechend für diese Variante der Kombinationstechnik, wobei sie in der ersten Arbeit zur Kombinationstechnik [GSZ92] für diese Form bewiesen werden.

Anwendungsbeispiele für die Kombinationstechnik bestehen in der Strömungssimulation [GHSZ93, GH95, GT95, Hub96, Kra02], der Lamé-Gleichung [Heu97], bei Eigenwertproblemen [Gar98, GG00], singulär gestörten Problemen [NH00] und parabolischen Gleichungen für Optionspreisaufgaben [Rei04].

Weiterhin existieren enge Bezüge zu den bereits erwähnten Boolean und blending-Methoden [Del82, DS89, BDJ92] so wie der splitting extrapolation-Methode [LLS95]. Desweiteren bestehen Ähnlichkeiten zwischen der Dünngitterkombinationstechnik und anderen additiven Ansätzen zur Funktionsdarstellung wie ANOVA [Wah90] oder Ensemble Methoden [Die00], bei denen ebenfalls Summen von Teilfunktionen gebildet werden.

4.2 Verallgemeinerte dünne Gitter mit der Kombinationstechnik

Eine erste einfache Verallgemeinerung der Kombinationstechnik besteht aus einer Gradierung der Gitter [GT95, GG00], um durch diese nun ortsadaptiven Gitter unter anderem Singuläritäten besser aufzulösen. Ein allgemeines Konzept für eine ortsadaptive Kombinationstechnik wird in [NH00] vorgestellt. In den von uns betrachteten Anwendungen spielt eine Ortsadaptivität allerdings keine größere Bedeutung, wichtiger ist hierbei eine unterschiedliche Auflösung der einzelnen Dimensionen. Dazu existieren Erweiterungen mittels anisotropen dünnen Gittern [GH95, Hub96, GG98, BM03] und komplexere Ansätze basie-

rend auf *verallgemeinerten dünnen Gittern* [GG03, Heg03], wobei das Gitter dimensionsadaptiv bestimmt wird.

Eine erweiterte Definition im Rahmen des ursprünglichen Dünngitteransatzes wird bereits in [Bun92] durch die so genannten *dünnen Gitter auf Basis der Energienorm* eingeführt. Der Ansatz einer veränderten Ausdünnung des Gitters wird in [Kna00] weiter generalisiert und ein Zusammenhang zwischen dem Ausdünnungsgrad des Gitters und der Glattheit der darzustellen Funktion, gemessen in geeigneten verallgemeinerten Sobolevnormen, gezeigt. Für diese *dünnen Gitter mit variablem Ausdünnungsgrad* wird in [Kna00] auch eine Kombinationstechnik eingeführt. Ein anderes Konzept eines Ausdünnungsgrad für die Kombinationstechnik wird in [Kra02] vorgestellt.

4.2.1 Anisotrope dünne Gitter

In [GH95, Hub96] werden erstmals anisotrope dünne Gitter durch zusätzliche Gewichtungsfaktoren κ_t für jede Koordinatenrichtung t definiert. Die neue Maschenweite $\hat{\underline{h}}$ wird dabei bestimmt durch $\hat{h}_t := \kappa_t^{-1} \cdot h_t$ mit $\kappa_t \in \mathbb{N}, \kappa_t \geq 2$. Die grundsätzliche Form der Kombinationstechnik wird dabei aber nicht verändert. In ähnlicher Weise werden in [BM03] anisotrope dünne Gitter genutzt.

In [GG98] werden anisotrope dünne Gitter mittels einem auf der Gewichtung von einzelnen Dimensionen durch unterschiedliche maximale Verfeinerungslevel aufbauendem Konzept eingeführt. Weiterhin wird eine verallgemeinerte Kombinationsformel definiert.

Definieren wir nun für einen gegebenen Levelindex $\underline{n} = (n_1, \ldots, n_d)$ eines anisotropen dünnen Gitters die zugehörige Indexmenge $\mathsf{I}_{\underline{n}}$ durch all jene Indizes, die auf oder unter der durch die Indizes $(n_1, 0, \ldots, 0), \ldots, (0, \ldots, 0, n_t, 0, \ldots, 0), \ldots, (0, \ldots, 0, n_d)$ definierten Hyperebene liegen. Damit gehören zu $\mathsf{I}_{\underline{n}}$ all die Indizes $\underline{k}$ mit

$$\sum_{t=1}^{d} \frac{k_t}{n_t} \leq 1, \quad k_t \geq 0, \tag{4.19}$$

wobei $k_t/n_t := 0$ für $n_t = 0$ gesetzt wird. Die Indexmenge bildet also einen verzerrten Simplex. Die Vereinigung aller Gitter $\Omega_{\underline{k}}$ mit $\underline{k} \in \mathsf{I}_{\underline{n}}$ ergibt nun ein anisotropes dünnes Gitter $\Omega_{\mathsf{I}_{\underline{n}}}$ vom Level $\underline{n}$. Den dazugehörigen Dünngitterraum bezeichnen wir mit $V_{\mathsf{I}_{\underline{n}}}$.

Weiterhin definieren wir die charakteristische Funktion $\chi^{\mathsf{I}_{\underline{n}}}$ von $\mathsf{I}_{\underline{n}}$ durch

$$\chi^{\mathsf{I}_{\underline{n}}}(\underline{k}) := \begin{cases} 1 & \text{falls } \underline{k} \in \mathsf{I}_{\underline{n}}, \\ 0 & \text{sonst.} \end{cases}$$

Damit kann nun die Formel für eine verallgemeinerte Kombinationstechnik

$$f^c_{\mathsf{I}_{\underline{n}}}(\underline{x}) := \sum_{\underline{k} \in \mathsf{I}_{\underline{n}}} \left(\sum_{\underline{z}=\underline{0}}^{\underline{1}} (-1)^{|\underline{z}|_1} \cdot \chi^{\mathsf{I}_{\underline{n}}}(\underline{k} + \underline{z}) \right) f_{\underline{k}}(\underline{x}) \tag{4.20}$$

für anisotrope dünne Gitter definiert werden, siehe hierzu auch [GG98, HKZ00, Kna00]. Natürlich werden bei der numerischen Umsetzung nur die Gitter $\Omega_{\underline{k}}$ behandelt, für die die Summe über die charakteristische Funktion $\chi^{\mathsf{I}_{\underline{n}}}$ der Nachbarn $\underline{k} + \underline{z}$ ungleich null ist.

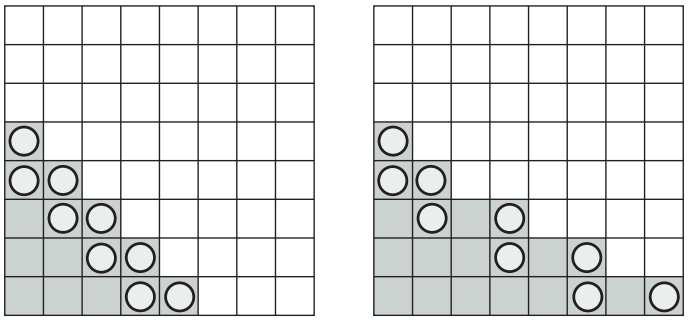


Abbildung 4.6. *Normale Kombinationstechnik für den Index* $\mathsf{I}_{4,4}$ *und anisotrope Kombinationstechnik für den Index* $\mathsf{I}_{7,4}$*. Die Indizes* (l_1, l_2)*, die nach (4.19) zum Indexset gehören sind grau unterlegt. Zusätzlich sind die Indizes mit einem Kreis gekennzeichnet, sofern das dazugehörige Gitter nach Kombinationsformel (4.20) einen Faktor ungleich Null aufweist. Die Achsen sind jeweils von Null bis Sieben durchnummeriert.*

In Abbildung 4.6 sind die Indexmengen zweier Kombinationstechniken schematisch dargestellt, einmal $\mathsf{I}_{4,4}$ zur regulären Kombinationstechnik, und einmal $\mathsf{I}_{7,4}$ zur anisotropen. Sofern die charakteristischen Funktion $\chi^{\mathsf{I}_{\underline{n}}}$ eines Index ungleich Null ist, d.h. wenn das dazugehörige Gitter in der Kombinationstechnik benutzt wird, ist dieser Index mit einem Kreis gekennzeichnet.

Zu solchen anisotropen Gittern verwandte Funktionenräume werden in [SW98, HW00] durch *gewichtete Sobolovräume* für Integrationsprobleme eingeführt. Dort wird mit Hilfe dieser Räume das Fehlerverhalten bei der numerischen Integration von Quasi-Monte-Carlo-Verfahren in Abhängigkeit von der Gewichtung der einzelnen Dimensionen untersucht.

Definieren wir dazu das innere Produkt $\langle f, g\rangle_{\underline{\gamma}}$ durch

$$\langle f, g\rangle_{\underline{\gamma}} := \sum_{\mathsf{u}\subset\{1,2,\ldots,d\}} \gamma_{\mathsf{u}}^{-1} \int_{[0,1]^{\mathsf{u}}} \frac{\partial^{|\mathsf{u}|}}{\partial \underline{x}_{\mathsf{u}}} f(\underline{x}_{\mathsf{u}}, \underline{1}) \frac{\partial^{|\mathsf{u}|}}{\partial \underline{x}_{\mathsf{u}}} g(\underline{x}_{\mathsf{u}}, \underline{1}) d\underline{x}_{\mathsf{u}}, \tag{4.21}$$

wobei $|\mathsf{u}|$ die Kardinalität von u ist. $(\underline{x}_{\mathsf{u}}, \underline{1})$ ist der d-dimensionale Vektor deren j-te Komponente definiert ist durch

$$(\underline{x}_{\mathsf{u}}, \underline{1})_j := \begin{cases} x_j, & j \in \mathsf{u}, \\ 1, & \text{sonst.} \end{cases}$$

$[0,1]^{\mathsf{u}}$ ist der $|\mathsf{u}|$-dimensionale Einheitswürfel passend zu den Variablen der Untermenge u. Weiterhin definieren wir

$$\gamma_{\mathsf{u}} := \prod_{j\in\mathsf{u}} \gamma_j, \quad d\underline{x}_{\mathsf{u}} := \prod_{j\in\mathsf{u}} dx_j.$$

Es zeigt sich, dass der Raum $F_{\underline{\gamma}}$ mit innerem Produkt $\langle f, g\rangle_{\underline{\gamma}}$ und Norm $\|f\|_{\underline{\gamma}}^2 := \langle f, f\rangle_{\underline{\gamma}}$ ein Hilbertraum ist. Die Rolle der Gewichte zeigt sich dadurch, dass für kleines γ_{u} die quadrierten partiellen Ableitungen $|\partial^{|\mathsf{u}|} f/\partial \underline{x}_{\mathsf{u}}|^2$ ähnlich klein werden müssen, um zur Einheitskugel in $F_{\underline{\gamma}}$ zu gehören.

Weiterhin ist der Raum $F_{\underline{\gamma}}$ auch ein Tensorproduktraum, er kann geschrieben werden als

$$F_{\underline{\gamma}} = F_{1,\gamma_1} \otimes \cdots \otimes F_{d,\gamma_d}.$$

Somit ist er auch ein Hilbertraum mit reproduzierendem Kern mit dem einfachen Kern

$$k_{\underline{\gamma}}(\underline{x}, \underline{y}) = \prod_{j=1}^{d} \left(1 + \gamma_j \min(1 - x_j, 1 - y_j)\right).$$

$F_{\underline{\gamma}}$ kann auch als Sobolevraum $H^1_{mix,\underline{\gamma}}$ mit dominierender gemischter und dimensionsgewichteter Ableitung bezeichnet werden, analog zu den bei dünnen Gittern gebräuchlichen Sobolevräumen H^1_{mix} mit dominierender gemischter Ableitung. Hierzu definieren wir $H^1_{mix,\underline{\gamma}}$ als

$$H^1_{mix,\underline{\gamma}}([0,1]^d) := H^1_{\gamma_1}([0,1]) \otimes \cdots \otimes H^1_{\gamma_d}([0,1]),$$

wobei gilt

$$H^1_{\gamma_j}([0,1]) = F_{j,\gamma_j}([0,1]).$$

Es seien an dieser Stelle noch die so genannten *klassischen anisotropen Sobolevräume* $H^{\underline{l}}_{\underline{p}}$ mit der Norm

$$\|f\|_{H^{\underline{l}}_{\underline{p}}} := \|f\|_{L_p} + \sum_{k=1}^{d} \left\| \frac{\partial^{l_k} f}{\partial x_k^{l_k}} \right\|_{L_p}$$

erwähnt, siehe [Dac03] und Referenzen dort. Für diese Räume wird weiterhin eine gemittelte Glattheit s definiert durch

$$\frac{1}{s} = \frac{1}{d}\left(\frac{1}{l_1} + \cdots + \frac{1}{l_d}\right).$$

Bei diesen Räumen sind also die einzelnen Dimensionen unterschiedlich glatt, im Gegensatz zur Definition 4.21, wo die einzelnen Dimensionen unterschiedlich gewichtet werden, aber die gleiche Glattheit mit dominierender gemischter Ableitung vorliegt. Weiterhin liegt bei den anisotropen Sobolevräumen auch keine Tensorproduktstruktur vor. Für Aussagen über das Verhalten von diskreten Approximationen, insbesondere beim Vorliegen einer Tensorproduktstruktur im Problem oder beim Verfahren, bringt die Nutzung dieser Räume somit nicht den Vorteil, den Sobolevräume mit dimensionsgewichteter und dominierender gemischter Ableitung in sich tragen.

4.2.2 Dünne Gitter mit variablem Ausdünnungsgrad

Erste Ansätze zur Veränderung des Ausdünnungsgrades eines dünnen Gitters werden in [Bun92] dargestellt. Ausgehend von der Beobachtung, dass in der H^1-Seminorm $|f|_{H^1}$, oft auch als Energienorm $\|f\|_E$ bezeichnet, die im Vergleich zu (4.11) schärfere Abschätzung

$$\|f_{\underline{l}}\|_E \leq C(|f|_\infty, d) h_{l_1}^2 \cdot \ldots \cdot h_{l_d}^2 \cdot \max_{1 \leq j \leq d} \left(\frac{h_{l_1}^2 \cdot \ldots \cdot h_{l_d}^2}{h_{l_j}^2} \right)$$

gilt, werden im Vergleich zur Standardformulierung stärker ausgedünnte Gitter definiert. In der Energienorm liefern nach dieser Abschätzung Teilräume mit $|l|_1 = c$ und $l_1 \approx \ldots \approx l_d$ deutlich kleinere Beiträge als solche mit $|l|_1 = c$ und sehr unterschiedlichen Werten l_j, d.h. $\min(l_j) \ll \max(l_j)$. Deswegen werden Teilräume $W_{\underline{l}}$ mit vielen Basiselementen, aber einem geringen Anteil an der Problemlösung, nach einem bestimmtem Kosten-Nutzen-Verhältnis vernachlässigt.

Dünne Gitter auf Grundlage der Energienorm werden mit diesen Überlegungen in [Bun92] über die folgende Indexmenge I_n^E definiert

$$\mathsf{I}_n^E := \left\{ \underline{l} \in \mathbb{N}^d \,\middle|\, |\underline{l}|_1 + \frac{1}{2} \min_{1 \le j \le d} \sum_{k \neq j} l_k \le n + \frac{3}{2}d - 1 \right\}. \tag{4.22}$$

Die Dimension des dazugehörigen Dünngitterraums $V_{\mathsf{I}_n^E}$ weist die Ordnung $O(h_n^{-1})$ auf, wobei die Konstante wie bisher exponentiell von der Dimension d abhängt. Der Approximationsfehler für Funktionen aus H^2_{mix} in der Energienorm beträgt

$$\|f - f_{\mathsf{I}_n^E}\|_E = O(h_n),$$

was dem Interpolationsfehler in der Energienorm bei der Verwendung von vollen Gittern entspricht. Allerdings wird der Dünngitter-Ansatz auf der Basis der Energienorm bisher nicht für eine verallgemeinerte Kombinationstechnik genutzt.

In [BG99] wird über eine optimale binäre Lösung eines Rucksack-Problems eine weiter verkleinerte Indexmenge für ein dünnes Gitter auf Basis der Energienorm wie folgt definiert

$$\mathsf{I}_n^E := \left\{ \underline{l} \in \mathbb{N}^d \,\middle|\, |\underline{l}|_1 - \frac{1}{5} \cdot \log_2 \left(\sum_{j=1}^{d} 4^{l_j} \right) \le (n + d - 1) - \frac{1}{5} \cdot \log_2(4^n + 4d - 4) \right\}. \tag{4.23}$$

Es ergeben sich die gleichen Approximations- und Dimensionsordnungen wie für das dünne Gitter auf Basis der Energienorm nach [Bun92]. Allerdings ist bei der Abschätzung der Dimension des diskreten Raumes die Konstante für Definition 4.23 kleiner.

In [GK00, HKZ00, Kna00] wird das Konzept des Ausdünnungsgrades eines dünnen Gitters weiter verallgemeinert und mit Approximationsaussagen abhängig von Glattheitsbedingungen so verknüpft, das Aussagen über ein optimales Gitter zur Approximation einer Funktion, abhängig von deren Glattheit, möglich sind. Optimal versteht sich hierbei als größtmögliche Reduktion der Anzahl der Freiheitsgrade im Vergleich zum vollen Gitter bei gleichbleibender Approximationsordnung.

Die dazu benötigten Funktionenräume sind Verallgemeinerungen der Sobolevräume mit dominierender gemischter Ableitung H^t_{mix} und wie folgt definiert:

$$H^{t,l}_{mix} := H^{t\underline{1}+l\underline{e}_1}_{mix} \cap \ldots \cap H^{t\underline{1}+l\underline{e}_d}_{mix},$$

wobei

$$H^{\underline{k}}_{mix} := H^{k_1} \otimes \ldots \otimes H^{k_d}.$$

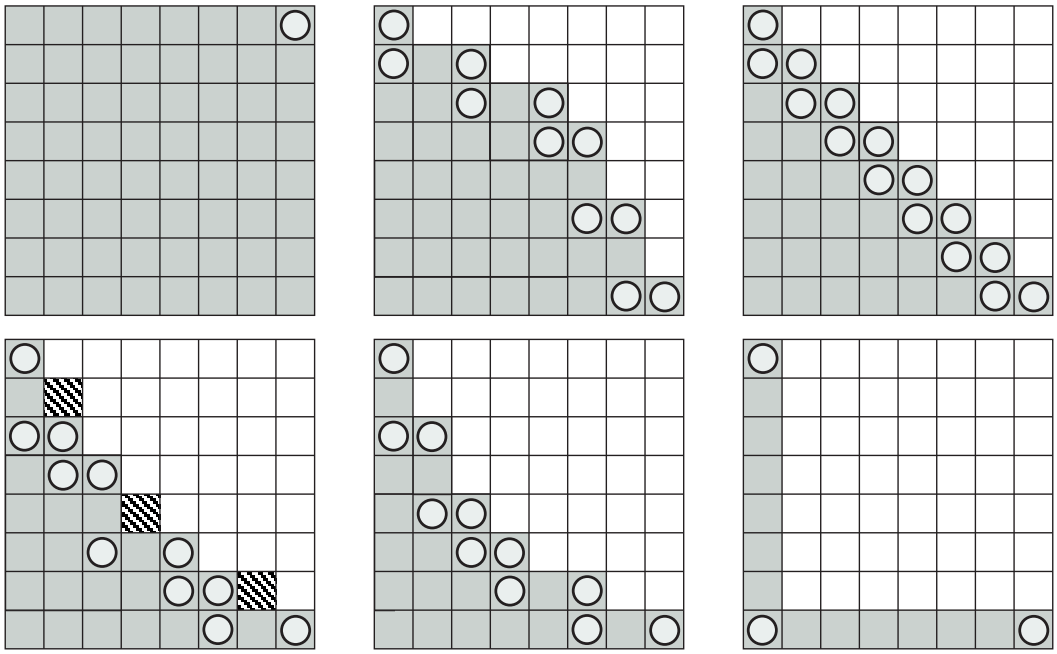


Abbildung 4.7. *Kombinationstechniken zu den Indexmengen* I_8^{∞}, I_8^{-1}, I_8^{0}, $\mathsf{I}_8^{\frac{1}{3}}$, $\mathsf{I}_8^{0.5}$ *und* I_8^{1} *(zeilenweise von links oben nach rechts unten), wobei die Gitter, die einen Faktor ungleich Null nach Formal (4.20) aufweisen, mit einem Kreis versehen sind. Die Indexmengen* $\hat{\mathsf{I}}_8^{\infty}$,$\hat{\mathsf{I}}_8^{-1}$, $\hat{\mathsf{I}}_8^{0}$, $\hat{\mathsf{I}}_8^{\frac{1}{3}}$, $\hat{\mathsf{I}}_8^{0.5}$ *und* $\hat{\mathsf{I}}_8^{1}$ *ergeben sich wenn die Achsen mit der Nummerierung von Eins bis Acht statt wie bisher Null bis Sieben versehen werden. Im Bild zur Indexmenge* $\hat{\mathsf{I}}_8^{\frac{1}{3}}$ *sind die in* I_8^E *zusätzlich enthaltenen Indizes gestreift dargestellt. Die Indexmenge zur Definition (4.23) entspricht für diesen Level der Indexmenge* $\hat{\mathsf{I}}_8^{0.5}$.

Die Räume $H^{t,l}_{mix}$ sind Mischungen aus den Sobolevräumen $H^t_{mix} = H^{t,0}_{mix}$ mit dominierender gemischter Ableitung und den üblichen Sobolevräumen $H^l = H^{0,l}_{mix}$. In [GK00, HKZ00, Kna00] werden Approximationsaussagen in diesen Funktionenräumen mit diskreten Räumen V_n^T, $n \in \mathbb{N},\ T \in (-\infty, 1]$, dargestellt. V_n^T ist hier definiert über

$$V_n^T := \bigoplus_{\underline{l} \in \mathsf{I}_n^T} W_{\underline{l}},$$

mit der Indexmenge

$$\mathsf{I}_n^T := \{\underline{l} \in \mathbb{N}^d \mid |\underline{l}|_1 - T|\underline{l}|_\infty \leq n - Tn\}, \tag{4.24}$$

wobei für $T = -\infty$ mit der natürlichen Erweiterung

$$\mathsf{I}_n^\infty := \{\underline{l} \in \mathbb{N}^d \mid |\underline{l}|_\infty \leq n\}$$

das bekannte volle Gitter dargestellt wird. Mit $T = 0$ ergibt sich der Dünngitterraum V_n^s, für $T = 1$ der Extremraum mit Teilgittern, die in einer Dimension den Index n besitzen und in den anderen 0. Siehe Abbildung 4.7 für zweidimensionale Beispielmengen.

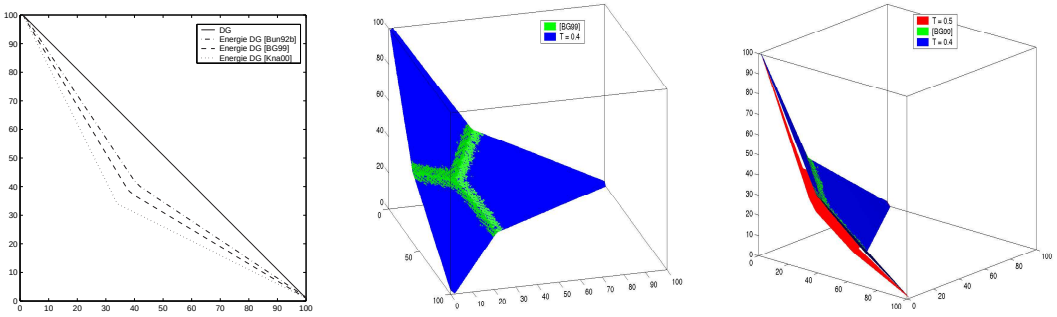


Abbildung 4.8. *Die Grenzflächen der Indexmengen für die verschiedenen dünnen Gitter auf Basis der Energienorm in zwei Dimensionen (links) und drei Dimensionen (mitte und rechts). Die Indexmengen nach [BG99] und für $T = 0.4$ nach [Kna00] sind nahezu identisch.*

Mit diesen Indexmengen lässt sich mittels der Formel (4.20) auch eine Kombinationstechnik definieren [HKZ00, Kna00]. Es ergeben sich dabei Kombinationstechniken des Typs (4.15). Die für gewisse Probleme, wie der numerischen Behandlung von partiellen Differentialgleichungen mit Dirichlet-Randwerten, geeignetere Kombinationstechnik der Art (4.18), d.h. mit immer vorhanderen inneren Punkten in den Teilgittern, lässt sich mit der folgenden Indexmenge erreichen:

$$\hat{\mathrm{I}}_n^T := \{\underline{l} \in \mathbb{N}^d \mid |\underline{l}|_1 - T|\underline{l}|_\infty \leq n + d - 1 - Tn\}. \tag{4.25}$$

In dieser Form werden diese Indexmengen in [GK00, HKZ00, Kna00] auch ursprünglich eingeführt und untersucht.

Die dünnen Gitter auf Basis der Energienorm können näherungsweise in Form dieses Ansatzes geschrieben werden. Aus der Definition (4.22) der Indexmenge I_n^E

$$\frac{3}{2}|\underline{l}|_1 - \frac{1}{2}|\underline{l}|_\infty = |\underline{l}|_1 + \frac{1}{2}\min_{1\leq j\leq d}\sum_{k\neq j} l_k \leq n + \frac{3}{2}d - 1$$

ergibt sich die Darstellung

$$\mathrm{I}_n^E := \left\{\underline{l} \in \mathbb{N}^d \,\middle|\, |\underline{l}|_1 - \frac{1}{3}|\underline{l}|_\infty \leq n + d - 1 - \frac{1}{3}n + \frac{1}{3}\right\}.$$

Bis auf die Konstante $1/3$ entspricht dieses der Indexmenge $\hat{\mathrm{I}}_n^{\frac{1}{3}}$. Allerdings zeigt sich, dass die Konstante eine im Vergleich zu $\hat{\mathrm{I}}_n^{\frac{1}{3}}$ etwas größere Indexmenge I_n^E erzeugt, siehe auch Abbildung 4.7.

Die nach (4.23) definierte Indexmenge entspricht annähernd der Menge $\hat{\mathrm{I}}_n^{0.4}$, sie ist geringfügig größer. Siehe Abbildung 4.8 für Indexmengen in zwei und drei Dimensionen. Im dreidimensionalen Fall ist zu erkennen, dass die beiden Indexmenge nahezu identisch sind, nur bei Indizes mit ähnlich großen Einzelkomponenten ergeben sich kleine Unterschiede zwischen den Mengen.

Wir zitieren nun ein Approximationsresultat aus [GK00, Kna00]. Für diesen Satz werden Normäquivalenzen in einer diskreten Multiskalenbasis der Form

$$\|f\|^2_{H^{t,l}_{mix}} \simeq \sum_{\underline{j}} \left(\sum_{i=1}^{n} 2^{2t|\underline{j}|_1 + 2lj_i} \right) \|w_{\underline{j}}\|^2_{L^2} \simeq \sum_{\underline{j}} 2^{2t|\underline{j}|_1 + 2l|\underline{j}|_\infty} \|w_{\underline{j}}\|^2_{L^2} \tag{4.26}$$

benötigt, wobei $0 \leq t + l < r,\ 0 \leq t < r$ und r abhängig von der Approximationsordnung der zu Grunde liegenden eindimensionalen Multiskalenzerlegung, für Details siehe [GK00, Kna00].

Satz 11. *Sei eine eindimensionale Multiskalenbasis gegeben, so dass (4.26) bezüglich der Tensorprodukt-Basen erfüllt ist. Die Parameter aus (4.26) seien so, dass die Beziehungen $s < t + l < r$ und $0 \leq t \leq r$ gelten. Dann gilt für $f \in H^{t,l}_{mix}$*

$$\inf_{v \in V_n^T} \|f - v\|_{H^s} \leq \begin{cases} C \cdot 2^{(s-l-t+(Tt-s+l)\frac{d-1}{d-T})n} \|u\|_{H^{t,l}_{mix}} & \text{für } T \geq \frac{s-l}{t}, \\ C \cdot 2^{(s-l-t)n} \|f\|_{H^{t,l}_{mix}} & \text{für } T \leq \frac{s-l}{t}. \end{cases} \tag{4.27}$$

Ab $T > 0.5$ gilt somit eine Fehlerabschätzung mit Ordnung $O(h_n)$ in der Energienorm für den betrachten Fall von Funktionen aus H^2_{mix} nicht mehr. Mit anderen Worten, das dünnste Gitter der Form (4.25) mit der gewünschten Approximationsordnung wird für $T = 0.5$ erreicht.

Desweiteren liefert [GK00, Kna00] auch Aussagen über die Dimension der diskreten Räume V_n^T.

Satz 12. *Es gilt*

$$dim\left(V_n^T\right) \leq \begin{cases} d \cdot 2^n & \text{für } T = 1, \\ \frac{d}{2} \left(\frac{1}{1 - 2^{-\frac{1}{1/T-1}}} \right)^d \cdot 2^n = O(2^n) & \text{für } 1/n < T < 1, \\ O(2^{\frac{T-1}{T/d-1} n}) & \text{für } T < 0, \\ O(2^{dn}) & \text{für } T = -\infty. \end{cases} \tag{4.28}$$

Außerdem gilt die Abschätzung

$$dim\left(V_n^T\right) \leq \left(\frac{n^{d-1}}{(d-1)!} + O(n^{d-2}) \right) \cdot 2^n = O(2^n n^{d-1}) \text{ für } 0 \leq T \leq 1/n. \tag{4.29}$$

Mit diesen beiden Sätzen ist es somit prinzipiell möglich, für Approximationsprobleme mit $f \in H^{t,l}_{mix}$ die Wahl eines Approximationsraumes V_n^T abhängig von der Glattheit so zu treffen, dass die Anzahl der Freiheitsgrade möglichst klein ist, aber trotzdem eine mit dem vollen Gitter vergleichbare Approximationsordnung erreicht wird. Allerdings ist in einer konkreten Anwendung normalerweise nicht bekannt, welcher Glattheitsklasse die darzustellende Funktion angehört.

4.2.3 Dimensionsadaptive Kombinationstechnik

Die im vorherigen Abschnitt vorgestellten dünnen Gitter mit variablem Ausdünnungsgrad weisen in allen Dimensionen gleiche Glattheitseigenschaften auf und sind in den diskreten Räumen jeweils gleich fein aufgelöst. Naheliegend ist eine Erweiterung mit anisotropen Ansätzen wie in Abschnitt 4.2.1 beschrieben. Dieses Konzept kann noch weiter verallgemeinert werden, indem statt einer Hyperebene, wie bei anisotropen dünnen Gittern, oder den Flächen, die bei einem dünnen Gitter mit variablem Ausdünnungsgrad entstehen, noch weniger Voraussetzungen an die Grenzfläche der Indexmenge gestellt werden.

Dafür betrachten wir nun verallgemeinerte Indexmengen I welche die so genannte *Zulässigkeitsbedingung* [GG03], oder *Konsistenzbedingung* [Pla00],

$$\underline{k} \in \mathsf{I} \text{ und } \underline{j} \leq \underline{k} \quad \Rightarrow \quad \underline{j} \in \mathsf{I} \tag{4.30}$$

erfüllen. Mit diesen Indexmengen definieren wir nun ein *verallgemeinertes dünnes Gitter* V_{I} mittels

$$V_{\mathsf{I}} := \bigoplus_{\underline{l} \in \mathsf{I}} W_{\underline{l}},$$

analog zur Definition der dünnen Gittern mit variablem Ausdünnungsgrad. Diese erfüllen ebenso wie die Indexmengen für anisotrope dünne Gitter offensichtlich (4.30).

Mit der Summationsformel (4.20) wird zu diesen verallgemeinerten dünnen Gittern eine Kombinationstechnik definiert. Theoretische Betrachtungen der Eigenschaften von vorgeschlagenen numerischen Integrationsverfahren basierend auf solchen Indexmengen sind in [WW99, Pla00] dargestellt.

In Abbildung 4.9 sind zweidimensionale Beispiele für solche Indexmengen und der dazugehörigen Kombinationstechnik dargestellt, von der normalen Kombinationstechnik über gewisse Anisotropien bis zu extremen Gittern, bei denen keine Kopplung zwischen den einzelnen Dimensionen mehr auftritt. In höherdimensionalen Anwendungen treten solche Effekte verstärkt auf, sie ermöglichen und erfordern eine große Flexibilität in der Gitterwahl.

Entscheidend bei der Nutzung solcher verallgemeinerter dünner Gitter ist die geeignete Wahl der Indexmenge I. Es kann externes Wissen über die Eigenschaften und das Zusammenspiel von einzelnen Dimensionen vorhanden sein und so a priori eine Indexmenge ausgewählt werden. Im Allgemeinen muss aber das numerische Verfahren die Wahl der benutzten Gitter automatisch und *dimensionsadaptiv* im Laufe der Berechnung selbst tätigen. In [Heg03] wird die grundsätzliche Anwendungsmöglichkeit eines dimensionsadaptiven Ansatzes für die Kombinationstechnik am Beispiel von Interpolationsproblemen vorgestellt. In [GG03] wird dieser Ansatz für die numerische Quadratur in einem dimensionsadaptiven Verfahren praktisch umgesetzt, wobei für die numerische Integration zwar dieselben Indexmengen benutzt werden, die Berechnung der Problemlösung aber nicht mit Formel (4.20) erfolgt, sondern die Ergebnisse der Teilprobleme einfach addiert werden. Darüberhinaus werden effiziente Datenstrukturen zur Verwaltung der Indexmengen beschrieben.

In einem dimensionsadaptiven Verfahren wird nun mit dem kleinsten Gitter begonnen, die Indexmenge besteht somit nur aus dem Gitter zum Index $\underline{0} = (0, \ldots, 0)$, d.h. $\mathsf{I} = \{\underline{0}\}$, und nach und nach werden weitere Indizes hinzugefügt so dass:

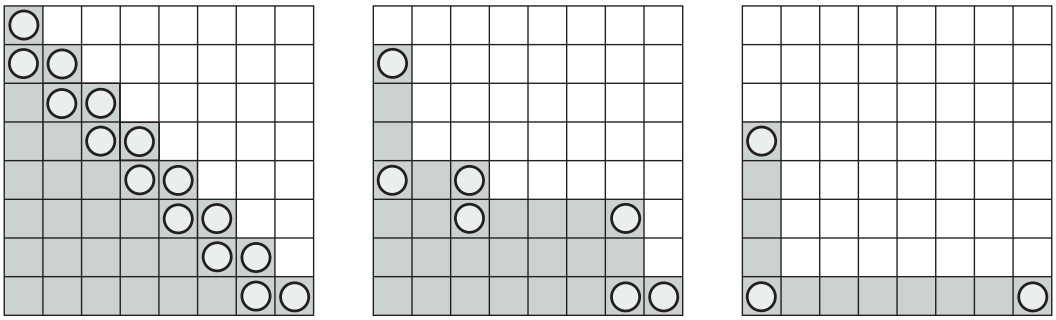


Abbildung 4.9. *Kombinationstechnik für das normale und zwei verallgemeinerte dünne Gitter.*

(i) die neue Indexmenge weiterhin zulässig ist und

(ii) das zugehörige Teilproblem einen möglichst großen Beitrag zur Lösung des Problems liefert.

Für den zweiten Punkt wird eine Methodik benötigt, die für jedes neue Gitter den potentiellen Beitrag zur Problemlösung schätzt. Bei der numerischen Integration ist dies zum Beispiel einfach das Teilintegral auf dem jeweiligen Gitter, bei Approximationsproblemen kann untersucht werden, um wie viel ein Gitter das der Approximation zu Grunde liegende Fehlerfunktional verkleinert. Wir unterscheiden hierbei zwischen einem *Fehlerindikator* und einem *Fehlerschätzer*. Einem Fehlerschätzer liegt dabei eine Theorie zu Grunde, die Aussagen über die Qualität der Schätzung ermöglicht, für einen Fehlerindikator existiert eine solche Theorie im Allgemeinen nicht, er ist meist heuristisch oder anschaulich begründet. Wir benutzen im Weiteren den Begriff des Fehlerindikators.

Das *Adaptionskriterium* entscheidet dann abhängig von diesem Fehlerindikator und eventuell weiteren Werten, wie dem Aufwand für die Berechnung einer Teillösung, welches Gitter den größten Nutzen bei der Problemlösung liefert. Dieses wird zur Indexmenge hinzugefügt und in dessen *vorderer Nachbarschaft* werden weitere Gitterkandidaten gesucht. Dabei bezeichnen wir als vordere Nachbarschaft eines Index $\underline{k}$ die Menge $\{\underline{k} + \underline{e}_t,\ 1 \leq t \leq d\}$, die Menge $\{\underline{k} - \underline{e}_t,\ 1 \leq t \leq d\}$ nennen wir die *hintere Nachbarschaft*. Diese Prozedur läuft solange bis ein geeignet zu definierendes globales Stoppkriterium für einen globalen Fehlerindikator E erfüllt ist, d.h. weitere neue Gitter nur noch einen sehr kleinen Beitrag zur Problemlösung liefern würden.

Zur Prüfung der Zulässigkeit eines neuen Index ist es notwendig, die äußere Schicht der bisher betrachteten Indizes zu betrachten. Die Menge A dieser so genannten *aktiven Indizes* enthält die Elemente aus I, deren vordere Nachbarn noch nicht betrachtet wurden. Die Menge O, genannt der *alte Indexset* (O für engl. old), enthält alle anderen Indizes von I, d.h. $\mathsf{O} := \mathsf{I} \backslash \mathsf{A}$. Ein Index kann nur dann zu der aktiven Indexmenge neu hinzugefügt werden, wenn alle hinteren Nachbarn in der alten Indexmenge sind.

Sofern sichergestellt ist, dass jedes weitere Gitter einen positiven Beitrag zur Problemlösung liefert, wie das z.B. bei der numerischen Integration der Fall ist [GG03], kann

jedes neu untersuchte Gitter sofort zur Gesamtlösung addiert werden. In diesem Fall besteht die Kombinationslösung aus allen Gittern mit Indizes in I. Es kann aber auch die Situation eintreten, dass eine Teillösung das Gesamtergebnis verschlechtert, hierbei besteht die Kombinationslösung dann nur aus Gitterindizes, die in O enthalten sind. Desweiteren kann es auch numerisch stabiler sein, mit der zweiten Variante zu arbeiten, obwohl die Problemstellung auch die erste Variante erlauben würde.

```
Symbole und Initialisierung:
  l := 0
  A := {l}      aktive Indexmenge
  O := ∅        alte Indexmenge
  ε_l           lokale Fehlerindikatoren
  E             globaler Fehlerindikator

Algorithmus
while (E > TOL) do
    wähle l ∈ A mit größtem ε_l
    O := O ∪ {l}
    A := A\{l}
    for t = 1, ..., d do
        j := l + e_t
        if j − e_l ∈ O für alle l = 1, ..., d then
            A := A ∪ {j}
            berechne Teilproblem für Index j
        endif
    endfor
    sofern notwendig (abhängig von Problem und Fehlerindikator)
        berechne lokale Fehlerindikatoren ε_k für alle k ∈ A
    sonst
        berechne lokalen Fehlerindikator ε_j für alle j := l + e_t mit j ∈ A
    berechne Gesamtfehlerindikator E
endwhile
Berechne Kombinationstechnik für Elemente aus I beziehungsweise O
```

Abbildung 4.10. *Der dimensionsadaptive Algorithmus.*

Aus diesen Überlegungen ergibt sich damit der in Abbildung 4.10 dargestellte Algorithmus. In jedem Schritt des dimensionsadaptiven Verfahrens wird das Element der aktiven Indexmenge A mit dem größtem lokalen Fehlerindikator ausgewählt und der alten Indexmenge O hinzugefügt. In der vorderen Nachbarschaft dieses Index werden dann alle zulässigen Teilprobleme berechnet und der aktiven Indexmenge hinzugefügt. Abhängig von Problem und Fehlerindikator kann es dabei notwendig sein, die lokalen Fehlerindikatoren der anderen aktiven Gitter neu zu berechnen. Anschließend wird der globale Fehlerindika-

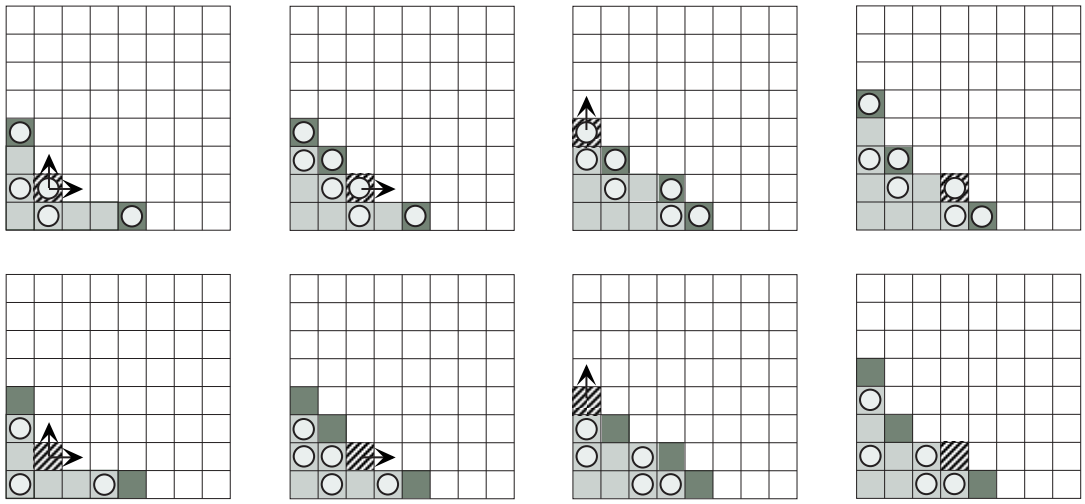


Abbildung 4.11. *Einige Schritte eines dimensionsadaptiven Verfahrens. Dargestellt sind die Indizes der beteiligten Gitter, wobei aktive Indizes* $\underline{i} \in \mathsf{A}$ *dunkelgrau dargestellt sind und alte Indizes* $\underline{i} \in \mathsf{O}$ *hellgrau. Der jeweils ausgewählte aktive Index mit dem größten Fehlerindikator ist gestreift dargestellt und Pfeile in die zulässigen vorderen Nachbarn sind angetragen. Die Dünngitterlösung besteht aus den Elementen der Indexmenge* I *(oben) beziehungsweise* O *(unten), wobei die Gitter die einen Faktor ungleich Null nach der Kombinationsformel (4.20) aufweisen mit einem Kreis versehen sind.*

tor E neu berechnet.

Diese Prozedur wird solange wiederholt, bis der globale Fehlerindikator eine gegebene Toleranzschwelle unterschreitet oder eine gegebene maximale Rechenzeit überschritten ist. Anschließend wird die durch die dimensionsadaptive Kombinationstechnik ermittelte Dünngitterlösung aus den Elementen aus I beziehungsweise O berechnet.

In der Abbildung 4.11 sind einige Adaptionsschritte eines zweidimensionales Beispiels dargestellt. Anhand ihres Fehlerindikators werden nacheinander die Indizes (1,1), (2,1), (0,3) und (3,1) ausgewählt und jeweils ihre vorderen Nachbarn auf Zulässigkeit untersucht. Nur im ersten Fall werden beide Nachbarn in die aktive Indexmenge A aufgenommen. Im letzten Fall ist sogar keiner der vorderen Nachbarn zulässig, so dass im nächsten Schritt nur mit den verbleibenden Indizes der aktiven Menge weitergearbeitet wird.

Es besteht die Hoffnung, dass effiziente dimensionsadaptive Verfahren heuristisch eine optimale Indexmenge im Sinne von [GK00, Kna00] aufbauen, was eng verwandt mit der N-Term-Bestapproximation [DeV98] ist.

Die gewichteten Sobolevräumen (4.21) stellen einen anderen Funktionenraum dar, der Indexmengen definiert, die in den einzelnen Dimension unterschiedlich ausgeprägt sind. Die Definition des inneren Produktes (4.21) für gewichtete Sobolevräume kann nun dahingehend verallgemeinert werden, dass die einzelnen Gewichte γ_{u} nicht mehr als Produkt von Dimensionsgewichten definiert werden, d.h. $\gamma_{\mathsf{u}} = \prod_{j \in \mathsf{u}} \gamma_j$, sondern für jede Indexmenge u einzeln. In [SWW04] wird dieser Ansatz für die numerische Integration untersucht.

Dazu wird dort der Begriff des *gewichteten Sobolevraums mit Gewichten von beschränkter Ordnung* $H(k_{\underline{a}})$ eingeführt. Definieren wir dazu das innere Produkt im Raum $H(k_{\underline{a}})$ durch

$$\langle f, g \rangle := \sum_{\mathsf{u} \subseteq \{1,\ldots,d\}} \gamma_{\mathsf{u}}^{-1} \int_{[0,1]^{|\mathsf{u}|}} \frac{\partial^{|\mathsf{u}|} f(\underline{x}_{\mathsf{u}}, \underline{a}_{-\mathsf{u}})}{\partial \underline{x}_{\mathsf{u}}} \, \frac{\partial^{|\mathsf{u}|} g(\underline{x}_{\mathsf{u}}, \underline{a}_{-\mathsf{u}})}{\partial \underline{x}_{\mathsf{u}}} \, d\underline{x}_{\mathsf{u}}, \tag{4.31}$$

wobei $|\mathsf{u}|$ die Kardinalität von u bezeichnet. Die γ_{u} sind beliebige nichtnegative Zahlen, von denen einige gleich Null sein dürfen, in diesem Fall werden Grenzwerte von positiven γ_{u} betrachtet. $\underline{x}_{\mathsf{u}}$ bezeichnet den $|\mathsf{u}|$-dimensionalen Vektor der Komponenten x_j mit $j \in \mathsf{u}$, und $\underline{x}_{-\mathsf{u}}$ bezeichnet den Vektor $\underline{x}_{\{1,\ldots,d\}\setminus\mathsf{u}}$. Weiterhin bezeichnet $(\underline{x}_{\mathsf{u}}, \underline{a}_{-\mathsf{u}})$ einen d-dimensionalen Vektor dessen j-te Komponente definiert ist durch

$$(\underline{x}_{\mathsf{u}}, \underline{a}_{-\mathsf{u}})_j := \begin{cases} x_j, & j \in \mathsf{u}, \\ a_j, & \text{sonst.} \end{cases}$$

Für $\mathsf{u} = \emptyset$ wird die Konvention benutzt, dass $\int_{[0,1]^0} f(\underline{x}_\emptyset, \underline{a}_{-\emptyset}) \, d\underline{x}_\emptyset = f(\underline{a})$, und ohne Beschränkung der Allgemeinheit wird $\gamma_\emptyset = 1$ angenommen. Der Punkt $\underline{a} = (a_1, \ldots, a_d)$ wird *Anker* genannt, der Raum wird auch als *verankerter Sobolevraum* bezeichnet [SWW04]. Mit $\underline{a} = \underline{1}$ und Produktgewichten γ_{u} erhalten wir die Darstellung (4.21) aus Abschnitt 4.2.1.

Der Raum $H(k_{\underline{a}})$ ist wiederum ein Hilbertraum mit reproduzierendem Kern, dieser ergibt sich als [SWW04]

$$k_{\underline{a}}(\underline{x}, \underline{y}) = \sum_{\mathsf{u} \subseteq \{1,\ldots,d\}} \gamma_{\mathsf{u}} \prod_{j \in \mathsf{u}} \eta_j(x_j, y_j),$$

wobei

$$\eta_j(x, y) = \begin{cases} \min(|x - a_j|, |y - a_j|), & \text{falls } (x - a_j)(y - a_j) > 0, \\ 0, & \text{sonst.} \end{cases}$$

Hier wird die Konvention benutzt, dass für $\mathsf{u} = \emptyset$ das Produkt in der Summationsformel für den Kern gleich 1 ist, und ohne Beschränkung der Allgemeinheit wird $\gamma_\emptyset = 1$ angenommen.

Mit diesen Definitionen wird in [SWW04] untersucht, wie hoch der Aufwand zur Reduktion des Fehlers um einen Faktor ε für Quasi-Monte-Carlo-Verfahren ist, sofern die Funktion aus einem solchen gewichteten Sobolevraum stammt. Falls die Gewichte nun von beschränkter Ordnung q sind, d.h. wenn ein q existiert mit

$$\gamma_{\mathsf{u}} = 0 \quad \forall \mathsf{u} \text{ mit } |\mathsf{u}| > q, \tag{4.32}$$

wird gezeigt, dass der Aufwand eines effizienten QMC-Verfahrens die Ordnung $O(\varepsilon^{-2} \cdot 3^q)$ aufweist und somit unabhängig von der Dimension d des Problems ist.

Das Integrationsproblem und das Approximationsproblem sind eng verwandt [HW00], deswegen besteht die Hoffnung in weiteren Untersuchungen diese Resultate in geeigneter Weise auf ein verallgemeinertes dünnes Gitter, welches mit einem dimensionsadaptiven Verfahren erzeugt wird, übertragen zu können.

Ein effizientes dimensionsadaptives Verfahren sollte nun passend zu den Gewichten die diskreten Teilräume bestimmen. Jedes Gewicht γ_{u} trägt hierbei zwei Informationen, zum

einen den Wert des Gewichtes, d.h. die Bedeutung des Indizes u, was eine gewisse diskrete Auflösung des Teilgitters impliziert, zum anderen die im Index enthaltenen und somit zu nutzenden Dimensionen. Bisher haben wir in der verallgemeinerten Kombinationstechnik Teilgitter betrachtet, die in allen Dimensionen leben. Das Konzept von gewichteten Sobolevräumen mit Gewichten von beschränkter Ordnung legt es nun aber nahe, Gitter zu nutzen, die nicht in allen Dimensionen leben, jeweils passend zum Index γ_{u} mit $|\mathsf{u}| \leq q$. Im folgenden Abschnitt betrachten wir dafür eine etwas andere Hierarchie mit zugehörigem Teilraumschema.

4.2.4 Hierarchie mit konstanten Funktionen

Eine andere hierarchische Darstellung einer Funktion aus V_n mit vielen Konsequenzen für das resultierende Verfahren wollen wir nun genauer vorstellen. Dazu definieren wir nun geringfügig andere hierarchische Differenzräume $\widetilde{W}_{\underline{l}}$, analog zur Definition der Differenzräume $W_{\underline{l}}$ aus (4.5), und Basisfunktionen $\widetilde{\phi}_{\underline{l},\underline{j}}(\underline{x})$ basierend auf den Basisfunktionen $\phi_{\underline{l},\underline{j}}(\underline{x})$. Der Unterschied besteht in der Einführung einer Stufe unter der bisherig kleinsten Stufe 0. Definieren wir dazu die Indexmenge $\widetilde{\mathsf{B}}_{\underline{l}}$ analog zu (4.7)

$$\widetilde{\mathsf{B}}_{\underline{l}} := \left\{ \underline{j} \in \mathbb{N}^d \;\middle|\; \begin{array}{lll} j_t = 1, \ldots, 2^{l_t} - 1,\ j_t \text{ ungerade}, & t = 1, \ldots, d, & \text{falls } l_t > 0, \\ j_t = 0, & t = 1, \ldots, d, & \text{falls } l_t \in \{0, -1\} \end{array} \right\}. \tag{4.33}$$

Desweiteren definieren wir die eindimensionalen Basisfunktionen $\widetilde{\phi}_{l,j}(x)$ durch

$$\begin{aligned} \widetilde{\phi}_{-1,0} &:= 1, \\ \widetilde{\phi}_{0,0} &:= \phi_{0,1}, \\ \widetilde{\phi}_{l,j} &:= \phi_{l,j} \quad \text{für } l \geq 1, \end{aligned}$$

mit $\phi_{l,j}$ definiert nach (4.4). Es gilt offensichtlich $\phi_{0,0} = \widetilde{\phi}_{-1,0} - \widetilde{\phi}_{0,0}$. Die d-dimensionalen Basisfunktionen werden wie üblich per Tensorproduktansatz gebildet

$$\widetilde{\phi}_{\underline{l},\underline{j}}(\underline{x}) := \prod_{t=1}^{d} \widetilde{\phi}_{l_t,j_t}(x_t). \tag{4.34}$$

Der Wechsel von Stufe -1 zu Stufe 0 kann als Übergang von konstanten zu linearen Basisfunktionen aufgefasst werden. Weiterhin definieren wir neue hierarchische Teilräume $\widetilde{W}_{\underline{l}}$ analog zu (4.8)

$$\widetilde{W}_{\underline{l}} = \operatorname{span}\{\widetilde{\phi}_{\underline{l},\underline{j}}, \underline{j} \in \widetilde{\mathsf{B}}_{\underline{l}}\}, \tag{4.35}$$

siehe auch Abbildung 4.12.

Es gilt offensichtlich $\widetilde{W}_{\underline{l}} = W_{\underline{l}}$ für $\underline{l} \geq \underline{0}$. Hiermit kann nun ein Vollgitterraum $\widetilde{V}_n$ wie folgt definiert werden

$$\widetilde{V}_n := \bigoplus_{l_1=-1}^{n} \cdots \bigoplus_{l_d=-1}^{n} \widetilde{W}_{\underline{l}} = \bigoplus_{|\underline{l}|_\infty \leq n} \widetilde{W}_{\underline{l}}. \tag{4.36}$$

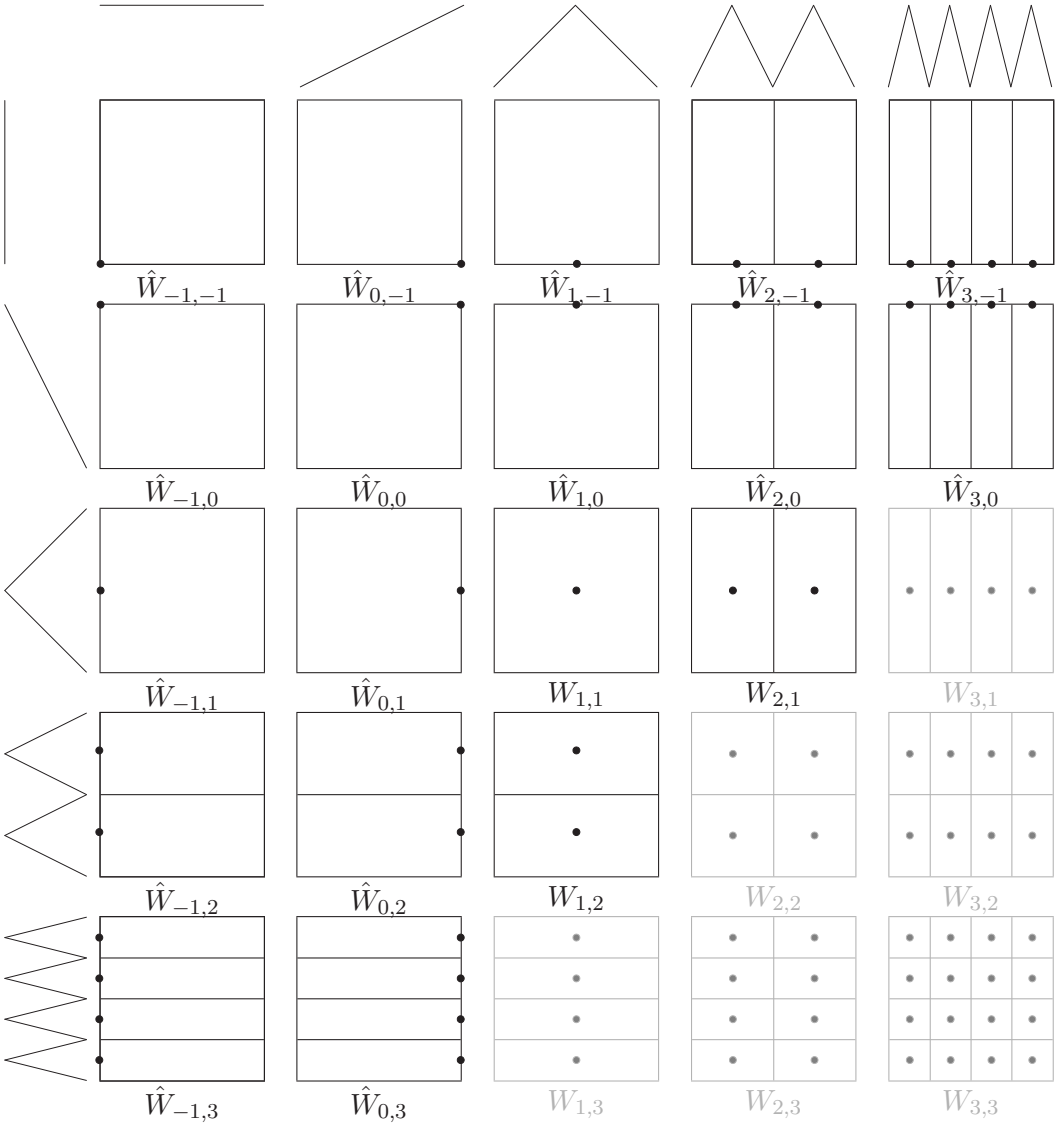


Abbildung 4.12. *Träger der hierarchischen Basisfunktionen des Raumes V_3. Zum Raum $\widetilde{V}_3^s$ gehören zusätzlich die nicht dargestellten Teilräume $\widetilde{W}_{4,-1}$ und $\widetilde{W}_{-1,4}$.*

Auch hier gilt wiederum $\widetilde{V}_n = V_n$ für $n \geq 0$. Die entsprechende Definition von Dünngitterräumen $\widetilde{V}_n^s$ mittels

$$\widetilde{V}_n^s := \bigoplus_{|\underline{l}|_1 \leq n} \widetilde{W}_{\underline{l}} \tag{4.37}$$

hat jedoch keine analoge Beziehung zu den ursprünglichen Räumen V_n^s. Allerdings gilt

$$V_n^s = \widetilde{V}_n^s \setminus \bigoplus_{\substack{|\underline{l}|_1 = n \text{ und} \\ \exists l_t = -1}} \widetilde{W}_{\underline{l}} \quad \text{für } n \geq 0,$$

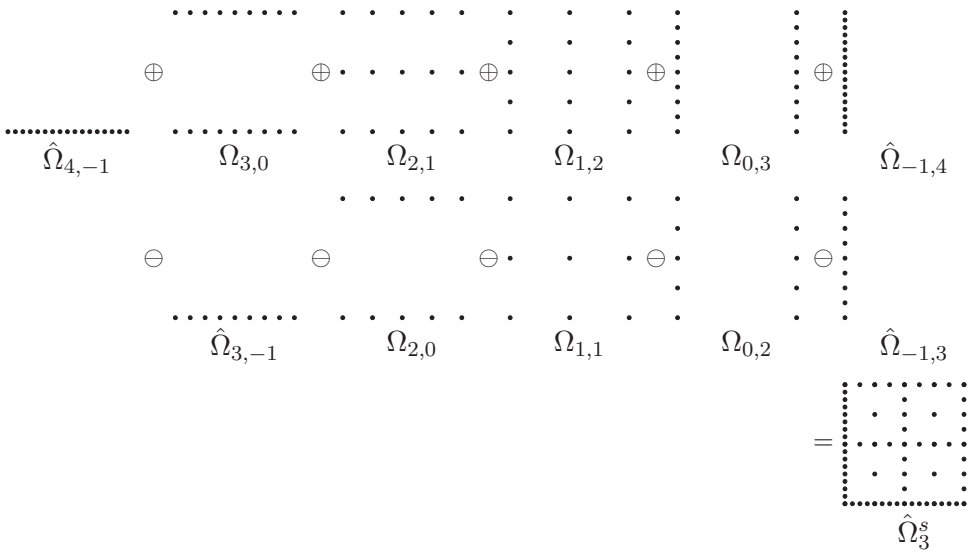


Abbildung 4.13. *Kombinationstechnik der Stufe 3 basierend auf veränderter Hierarchie, $d = 2, n = 3$.*

siehe wiederum Abbildung 4.12.

Die natürliche Fortsetzung des Übergangs von konstanten zu linearen Basisfunktionen wäre die Nutzung quadratischer, kubischer, usw. Basisfunktionen für die nächsten Stufen. Dieser Ansatz kann grundsätzlich auch verfolgt werden.

In analoger Weise zu (4.14) und (4.15) kann nun für die Räume $\widetilde{V}_n^s$ eine Kombinationstechnik definiert werden durch

$$l_1 + \ldots + l_d = n - q, \quad q = 0, \ldots, d-1, \quad l_t \geq -1, \tag{4.38}$$

und

$$\widetilde{f_n^c}(\underline{x}) := \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\underline{l}|_1 = n-q} f_{\underline{l}}(\underline{x}), \tag{4.39}$$

siehe auch Abbildung 4.13.

Der entscheidende Vorteil der Verwendung von konstanten Basisfunktionen in der Hierarchie zeigt sich im höherdimensionalen Fall. Eine Tensorierung mit einer Konstanten erzeugt keine zusätzlichen Freiheitsgrade. So bleibt eine $(d-1)$-lineare Funktion nach der Verknüpfung mit einer Konstanten in der d-ten Dimension weiterhin $(d-1)$-linear, durch das Produkt wird somit kein zusätzlicher Freiheitsgrad hinzugewonnen. Es mag auf den ersten Blick widersinnig sein, diese Eigenschaft als Vorteil zu sehen, in der numerischen Realisierung des Verfahrens zeigen sich allerdings Vorteile. Da nun die Dimensionen mit konstanten Basisfunktionen nicht benutzt werden, verändert sich die Komplexität der Teilprobleme. In der ursprünglichen Hierarchie hat das kleinste mögliche Gitter $\Omega_{\underline{0}}$ bereits 2^d Punkte, nun hat das kleinste Gitter $\Omega_{\underline{-1}}$ genau einen Punkt. Somit sind für deutlich höherdimensionale Probleme Teilgitter berechenbar. Für die Kombinationstechnik nach (4.39)

ergeben sich allerdings dadurch keine Vorteile, da auch alle bisherigen Gitter berechnet werden müssen.

Anders ist dies bei einer dimensionsadaptiven Kombinationstechnik, hier werden wie in Abschnitt 4.2.3 beschrieben die Gitter adaptiv ausgewählt. Insbesondere kann mit einem verallgemeinerten dünnen Gitter, das eine Hierarchie mit konstanten Basisfunktionen nutzt, nun ein gewichteter Sobolevraum (4.31) mit beschränkter Ordnung der Gewichte geeignet diskretisiert werden. Somit ist die Größe der Teilgitter in der Kombinationstechnik für die Darstellung einer Funktion aus einem d-dimensionalen gewichteten Sobolevraum nicht mehr abhängig von der Dimension d, sondern dementsprechend von der Ordnung q der Gewichte.

Entsprechend der Definition (4.32) der Ordnung eines gewichteten Sobolevraums werden wir im Weiteren auch von der *Ordnung* q, oder auch der *effektiven Dimension*, eines verallgemeinerten dünnen Gitters auf Basis der Hierarchie mit konstanten Basisfunktionen sprechen. Die Ordnung q eines verallgemeinerten Dünngitteraums $\widetilde{V}_{\underline{l}}$ definieren wir somit als

$$q(\widetilde{V}_{\underline{l}}) := \max_{\underline{l} \in \mathbb{I}} \sum_{i=1}^{d} \left\{ \begin{array}{ll} 1 & \text{falls } l_i > -1 \\ 0 & \text{falls } l_i = -1 \end{array} \right\}.$$

Es sei abschließend bemerkt, dass wir uns bisher nur mit einer Sorte von Hierarchie beschäftigt haben, nämlich der Verfeinerung eines Gitters durch Hinzunahme von weiteren Punkten, gesteuert durch die Halbierung des Abstandes der beteiligten Gitterpunkte, die so genannte h-Verfeinerung. Es lassen sich aber auch andere Formen einer Hierarchie nutzen. Als mögliche Beispiele seien erwähnt

- Funktionen ϕ_j, die konstant auf dem Teilintervall $[j/2^n, (j+1)/2^n]$ sind. Dies führt zu Haar-Wavelets.
- Stetige Funktionen ϕ_j, die linear auf dem Intervall $[j/2^n, (j+1)/2^n]$ sind.
- Funktionen $\phi : C \to \mathbb{R}$, die konstant auf A_j sind, wobei $\bigcup A_j = C$ eine Aufteilung der Menge C ist.
- Regressionsbäume und Multivariate Splines.

Auch mit diesen Hierarchien lassen sich Differenzräume und Teilräume definieren und das Prinzip des systematischen Ausdünnens eines Funktionenraumes lässt sich entsprechend anwenden [Heg03].

In diesem Kapitel haben wir die Diskretisierung mit dünnen Gittern nach [Zen91] und die Kombinationstechnik zur Bestimmung von Funktionen auf dünnen Gittern eingehend betrachtet. Wir haben Erweiterungen in Form von anisotropen dünnen Gittern, solchen mit variablem Ausdünnungsgrad und schließlich verallgemeinerten dünnen Gittern beschrieben. Um verallgemeinerte dünne Gitter angepasst an das Problem bestimmen zu können haben wir eine dimensionsadaptive Kombinationstechnik vorgestellt. Mit Hilfe von verallgemeinerten Dünnen Gittern auf Basis einer Hierarchie mit konstanten Funktion können weiterhin Funktionen mit einer effektiven Dimension $q < d$ geeignet dargestellt werden. Im folgenden Kapitel beschäftigen wir uns nun mit der Anwendung von dünnen Gittern bei der Funktionsrekonstruktion.

Kapitel 5

Implementierungsaspekte und Komplexitätsdiskussion

Wir betrachten nun die Anwendung der Dünngitter-Kombinationstechnik bei der numerischen Realisierung der Funktionsrekonstruktion nach Kapitel 3. Insbesondere untersuchen wir dabei die Komplexität des Verfahrens in Bezug auf die Menge der Datenpunkte beziehungsweise die Zahl der Dimensionen. Weiterhin erläutern wir numerische Vorteile, die sich durch die Verwendung einer simplizialen Diskretisierung auf den Teilgittern der Kombinationstechnik ergeben. Anschließend beschreiben wir, wie das Verfahren einfach aber effizient parallelisiert werden kann.

Dünne Gitter benötigen, wie in Kapitel 4 dargestellt, zur Approximation von Funktionen gewisser Glattheitsklassen $O(h_n^{-1} \cdot \log(h_n^{-1})^{d-1})$ Gitterpunkte, im Gegensatz zu $O(h_n^{-d})$ bei vollen Gittern, d.h. sie weisen eine deutlich geringere Steigerung mit der Zahl der Dimensionen auf. Allerdings ist diese in der Dünngitter-Literatur übliche Ordnungsabschätzung eine Darstellung bezüglich der Verfeinerung h, so werden von der Dimension d abhängige Konstanten nicht explizit aufgeführt. Wie in Kapitel 4 gesehen hat ein dünnes Gitter mindestens die Größe 2^d, d.h. die Komplexität des Verfahrens hängt exponentiell von der Zahl der Dimensionen ab. Sofern mit einem dimensionsadaptiven Verfahren verallgemeinerte dünne Gitter bestimmt werden und deren effektive Dimension $q < d$ ist, hängen die Komplexitäten des Verfahrens entsprechend von q ab.

Andere funktionsorientierte Ansätze, die ursprünglich zur Darstellung von diskreten Daten entwickelt wurden, nutzen Basisfunktionen auf den Datenpunkten, wie z.B. Ansätze mit radialen Basisfunktionen oder Support Vektor Maschinen. Die Komplexität dieser Verfahren skaliert zwar nicht exponentiell mit der Zahl der Dimension, sie weisen allerdings nichtlinearen Aufwand bezüglich der Menge der Datenpunkte auf und sind für große Datenmengen in der Praxis nicht geeignet.

Der Ansatz mit festen Gitterpunkten, auf denen Basisfunktionen definiert sind, wobei die Gitterpunkte unabhängig von den Datenpunkten gewählt werden, ist in diesem Anwendungsgebiet neu und wird erst durch die Nutzung von dünnen Gittern realisierbar. Der entscheidende Vorteil im Vergleich zu den erwähnten maschinellen Lernverfahren, die nichtlinear bezüglich der Zahl der Dateninstanzen skalieren, ist die Beobachtung, dass die Verwendung von dünnen Gittern ein maschinelles Lernverfahren ermöglicht, welches linear in der Zahl der Datenpunkte skaliert, aber trotzdem eine nichtlineare Funktion darstellt.

5.1 Funktionsrekonstruktion mit dünnen Gittern

Beim Ansatz der der Dünngitter-Kombinationstechnik zur Funktionsrekonstruktion gehen wir wie folgt vor: Wir diskretisieren und lösen für eine Datenmenge

$$S = \{(\underline{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}\}_{i=1}^M$$

das Variationsproblem (3.1) mit Funktional (3.7) auf einer bestimmten Sequenz von anisotropen Gittern $\Omega_{\underline{l}}$ mit äquidistanten Maschenweiten $h_t = 2^{-l_t}$ in der t-ten Koordinatenrichtung. Genauer betrachten wir

$$\Omega_{\underline{l}} \text{ mit } \underline{l} \in \mathsf{I},$$

mit einer nach (4.30) zulässigen Indexmenge I. Im Folgenden werden wir für das numerische Verfahren immer den Gradienten $\mathcal{S} = \nabla$ im Regularisierungsausdruck (3.5) benutzen. Wir erhalten somit als zu minimierendes Funktional

$$R(f) = \frac{1}{M} \sum_{i=1}^{M} (f(\underline{x}_i) - y_i)^2 + \lambda \|\nabla f\|_{L^2}^2, \quad f \in V_{\mathsf{I}}. \tag{5.1}$$

Es sei darauf hingewiesen, dass der hier benutzte Regularisierungsoperator ∇ in einem unendlich dimensionalen Funktionenraum die Morozovsche Komplementbedingung (2.21) beziehungsweise die Voraussetzungen von Satz 9 nicht erfüllt und somit keine eindeutige Lösung liefert. In der Kombination mit einer vorherigen Diskretisierung des Raumes und der damit verbundenen Projektion in einen endlich dimensionalen Raum ergibt das Verfahren mit $\mathcal{S} = \nabla$ hingegen eine eindeutige Lösung. Der Regularisierungsterm kann dabei als Stabilisierung des numerischen Approximationsverfahrens betrachtet werden. Eine andere Sichtweise besteht darin, den nun benutzten einfachen Operator als Approximation des eigentlichen Regularisierungsoperators aufzufassen, in diesem Fall z.B. die H^1_{mix}-Norm. Entscheidender Grund für die Nutzung von $\mathcal{S} = \nabla$ trotz dieser aus der Regularisierungstheorie begründeten Bedenken sind die dadurch entstehenden algorithmischen Vorteile. Zum einen ist der so entstehende Regularisierungsoperator mit stückweise linearen Funktionen diskret behandelbar, anders wie z.B. höhere Sobolevnormen, die eine Diskretisierung höherer Ordnung erfordern würde. Auch gegenüber der H^1_{mix}-Norm ergeben sich algorithmische Vorteile, diese Norm ist zwar mit stückweise linearen Funktionen realisierbar, allerdings benötigt deren Berechnung im Verfahren im Vergleich zu $\mathcal{S} = \nabla$ deutlich mehr Aufwand.

Eine weitere Überlegung zur dimensionsadaptiven Variante des Verfahrens betrifft den Zusammenhang zwischen Regularisierungsoperator und dem Funktionenraum, in dem die Lösung erreicht wird. Dieser ergibt sich nach Kapitel 3 bekanntlich als Hilbertraum mit reproduzierendem Kern, wobei ein enger Zusammenhang zwischen Kern und Regularisierungsoperator besteht, siehe Satz 8. Im Hinblick auf die Betrachtungen aus Kapitel 4 zur Approximationseigenschaft der dimensionsadaptiven Kombinationstechnik, insbesondere bezüglich gewichteter Sobolevräume mit Gewichten von beschränkter Ordnung (4.31), sollte in zukünftigen Arbeiten untersucht werden, inwiefern die feste Wahl eines Regularisierungsoperators im Verfahren angebracht ist. Da der Regularisierungsoperator in gewissem Sinne den Lösungsraum festlegt, könnte sich ein Konzept mit adaptiven Regularisierungsoperatoren als sinnvoll erweisen. Denn eine zu starke Einschränkung durch den

Regularisierungsoperator könnte sich im Verfahren als nachteilig erweisen. Somit kann die Nutzung eines einfachen Regularisierungsterms vorteilhaft sein, der dabei hauptsächlich zur Stabilisierung des Verfahrens dient. Im Rahmen von Support Vektor Maschinen gibt es erste Untersuchungen in diesem Sinne mit der adaptiven Bestimmung von so genannten *Hyperkernen* [OSW03], die in den Experimenten als Summe von gewissen anisotropen Kernen dargestellt werden.

Ein Finite Elemente-Ansatz mit stückweise d-linearen Ansatz- und Testfunktionen $\phi_{\underline{l},\underline{j}}(\boldsymbol{x})$ auf dem Gitter $\Omega_{\underline{l}}$ verwendet nun

$$f_{\underline{l}}(\underline{x}) = \sum_{j_1=0}^{2^{l_1}} \cdots \sum_{j_d=0}^{2^{l_d}} \alpha_{\underline{l},\underline{j}} \phi_{\underline{l},\underline{j}}(\underline{x})$$

und der Variationsansatz resultiert nach (3.7) – (3.12) mit dem Funktional (5.1) in dem diskreten Gleichungssystem

$$(B_{\underline{l}}^t \cdot B_{\underline{l}} + \lambda M \cdot C_{\underline{l}})\alpha_{\underline{l}} = B_{\underline{l}}^t y, \tag{5.2}$$

mit den Matrizen

$$(C_{\underline{l}})_{\underline{j},\underline{k}} := (\nabla\phi_{\underline{l},\underline{j}}, \nabla\phi_{\underline{l},\underline{k}}) \tag{5.3}$$

und

$$(B_{\underline{l}})_{i,\underline{j}} := \phi_{\underline{l},\underline{j}}(\underline{x}_i), \tag{5.4}$$

und dem unbekannten Vektor $(\alpha_{\underline{l}})_{\underline{j}}$, wobei $j_t, k_t = 0, \ldots, 2^{l_t},\ t = 1, \ldots, d$ und $i = 1, \ldots, M$. Diese Probleme lösen wir dann mit einer geeigneten Methode. Dazu benutzen wir das konjugierte Gradientenverfahren zusammen mit einem diagonalen Vorkonditionierer. Die diskreten Lösungen $f_{\underline{l}} = \sum_{\underline{j}} \alpha_{\underline{l},\underline{j}} \phi_{\underline{l},\underline{j}}(\underline{x})$ sind in den Räumen $V_{\underline{l}}$ der stückweise d-linearen Funktionen auf dem Gitter $\Omega_{\underline{l}}$ enthalten. Schließlich kombinieren wir diese Ergebnisse auf den verschiedenen Gitter $\Omega_{\underline{l}}$ nach (4.20), d.h.

$$f_{\mathsf{l}}(\underline{x}) := \sum_{\underline{l} \in \mathsf{l}} \left(\sum_{\underline{z}=\underline{0}}^{\underline{1}} (-1)^{|\underline{z}|_1} \cdot \chi^{\mathsf{l}}(\underline{l} + \underline{z}) \right) f_{\underline{l}}(\underline{x}).$$

Wenn wir nun eine neu gegebene Menge von Datenpunkten $\{\underline{\tilde{x}}_i\}_{i=1}^{\tilde{M}}$ (die Testdaten) mit

$$\tilde{y}_i := f_{\mathsf{l}}(\underline{\tilde{x}}_i), \quad i = 1, \ldots, \tilde{M}$$

auswerten wollen, müssen wir nur die Kombination der zugehörigen Werte für f_{l} wiederum nach Formel (4.20) bilden. Insgesamt erhalten wir den Algorithmus aus Abbildung 5.1.

5.1.1 Komplexität des Verfahrens

Wie bei der numerischen Behandlung von partiellen Differentialgleichungen sind also letztendlich Lösungen von linearen Gleichungssystemen mit bestimmten Matrizen zu berechnen. Sofern deren Aktion auf einen Vektor, d.h. das Ergebnis der Matrix-Vektor-Multiplikation, mit wenig Rechenaufwand zu berechnen ist, braucht die Matrix nicht explizit aufgestellt und gespeichert zu werden, und das lineare Gleichungssystem kann durch

Gegeben sind die Datenpunkte $\{(\underline{x}_i, y_i)\}_{i=1}^{M}$ (die Trainingsdaten)
Berechnung des Dünngitterklassifikators:
Schleife über alle $\underline{l} \in \mathsf{I}$
  sofern 0 ungleich $\sum_{\underline{z}=\underline{0}}^{\underline{1}} (-1)^{|\underline{z}|_1} \cdot \chi^{\mathsf{I}_{\underline{n}}}(\underline{l} + \underline{z})$
    löse das lineare Gleichungssystem $(\lambda C_{\underline{l}} + B_{\underline{l}}^t \cdot B_{\underline{l}})\alpha_{\underline{l}} = B_{\underline{l}}^t y$
    speichere die Lösung $\alpha_{\underline{l}}$ geeignet

Evaluierung von neu gegebenen Datenpunkten $\{\tilde{\underline{x}}_i\}_{i=1}^{\tilde{M}}$ (die Testdaten):
$\tilde{y}_i := 0, \quad i = 1, \ldots, \tilde{M}$
Schleife über alle $\underline{l} \in \mathsf{I}$
  sofern 0 ungleich $\sum_{\underline{z}=\underline{0}}^{\underline{1}} (-1)^{|\underline{z}|_1} \cdot \chi^{\mathsf{I}_{\underline{n}}}(\underline{l} + \underline{z})$
    $i = 1, \ldots, \tilde{M}$
      evaluiere die Lösung $f_{\underline{l}}$ in $\tilde{\underline{x}}_i$ auf dem Gitter $\Omega_{\underline{l}}$
      setze $\tilde{y}_i := \tilde{y}_i + (\sum_{\underline{z}=\underline{0}}^{\underline{1}} (-1)^{|\underline{z}|_1} \cdot \chi^{\mathsf{I}_{\underline{n}}}(\underline{l} + \underline{z})) f_{\underline{l}}(\tilde{\underline{x}}_i)$

Abbildung 5.1. *Der Algorithmus.*

einen iterativen Löser wie z.B. cg-Verfahren oder GMRES mit so genannter on-the-fly-Berechnung der Matrix-Vektor-Multiplikation gelöst werden.

Wenn allerdings die Berechnung der Matrizen zu kostspielig ist, müssen diese explizit gespeichert werden. Der benötigte Speicherplatz ist dabei abhängig von der Zahl der Gitterpunkte und der Anzahl der Nachbarn eines Gitterpunktes, beides d-abhängige Größen. Bei Diskretisierungen mit einem Tensorproduktansatz hat ein innerer Gitterpunkt 3^d-Nachbarn, wenn er auf dem Rand liegt, reduziert sich dies, ein Eckpunkt hat mit 2^d-Nachbarn die wenigsten. Bei der Speicherung der Matrizen ist durch den Hauptspeicher des Computers eine Limitierung der Größe der zu behandelnden Probleme gegeben. Das gröbste Gitter mit 2^d-Punkten, bei dem jeder Gitterpunkt 2^d-Nachbarn hat, benötigt somit 2^{2d}-Matrixeinträge, eine Matrix dieser Größe kann bei mehr als 13 Dimensionen nicht mehr auf aktuellen Arbeitsplatzrechnern im Hauptspeicher behandelt werden. Aus dieser Sichtweise ist somit eine Aufstellung der Matrizen eher zu vermeiden.

Allerdings ist eine on-the-fly-Berechnung der in Gleichung (5.2) auftretenden Matrizen für sehr große Datenmengen nicht angebracht. Die Berechnung der Datenmatrix ist äußerst teuer, da dabei pro Datenpunkt mindestens 2^d Operationen benötigt werden. Dies begründet sich darin, dass jeder Datenpunkt im Träger von genau 2^d Basisfunktionen lebt und alle Basisfunktionen auf diesem Datenpunkt ausgewertet werden müssen. Dabei können zwar Teilergebnisse mehrfach verwendet werden, aber dadurch verbessert sich die Komplexität nicht. Eine wiederholte on-the-fly-Berechnung der Matrix, wie es bei iterativen Lösern erforderlich ist, ist somit aus Gründen der Rechenzeit bei großen Datenmengen zu vermeiden.

Weiterhin sind bei der Berechnung der Matrizen zwei verschiedene Zugänge möglich,

	$C_{\underline{l}}$	$G_{\underline{l}}$	$B_{\underline{l}}$	$r_{\underline{l}}$	$\alpha_{\underline{l}}$
Speicher	$O(3^d \cdot N)$	$O(3^d \cdot N)$	$O(2^d \cdot M)$	$O(N)$	$O(N)$
Aufstellung	$O(3^d \cdot N)$	$O(2^{2d} \cdot M)$	$O(d \cdot 2^d \cdot M)$	$O(d \cdot 2^d \cdot M)$	-
MV-Multiplikation	$O(3^d \cdot N)$	$O(3^d \cdot N)$	$O(2^d \cdot M)$	-	-

Tabelle 5.1. *Komplexitäten für das Speichern, das Aufstellen und die Matrix-Vektor-Multiplikation für verschiedene Matrizen und Vektoren, die bei der Kombinationstechnik auf dem Gitter $\Omega_{\underline{l}}$ entstehen.*

wir können die Matrizen

$$C_{\underline{l}} \quad \text{und} \quad G_{\underline{l}} := B_{\underline{l}}^t \cdot B_{\underline{l}} \tag{5.5}$$

aufstellen und nutzen, wobei sie beide in einer gemeinsamen Matrixstruktur gespeichert werden sollten, oder wir können

$$C_{\underline{l}} \text{ und } B_{\underline{l}} \tag{5.6}$$

aufstellen, einzeln speichern und einsetzen. Natürlich sind all diese Matrizen dünn besiedelt, wir stellen nur die nicht-Null Einträge auf. Die zugehörigen Komplexitäten sind in Tabelle 5.1 für die Diskretisierung mit Tensorproduktfunktionen angegeben. Hier bezeichnet $N = \prod_{t=1}^{d} 2^{l_t} + 1$ die Zahl der Unbekannten auf dem Gitter $\Omega_{\underline{l}}$. Für $G_{\underline{l}}$ ergibt sich die Darstellung

$$(G_{\underline{l}})_{\underline{j},\underline{k}} = (B_{\underline{l}}^t \cdot B_{\underline{l}})_{\underline{j},\underline{k}} = \sum_{i \leq M} \varphi_{\underline{l},\underline{j}}(\underline{x}_i) \cdot \varphi_{\underline{l},\underline{k}}(\underline{x}_i),$$

d.h. für jeden Datenpunkt $\underline{x}_i$ muss nun das Produkt der Auswertung aller Basisfunktionen berechnet werden, in dessen Träger der Punkt liegt; dies erfordert 2^{2d} Operationen.

Interessanterweise gibt es somit einen Unterschied in den Komplexitäten für diese beiden Zugänge: Die Speicherkosten und die Kosten der Matrix-Vektor-Multiplikation des ersten Zugangs skalieren mit N, wohingegen sie für den zweiten Zugang mit M skalieren. Insbesondere für den Fall $M \gg N$, d.h. für eine moderate Zahl von Leveln n im dünnen Gitter, aber für eine sehr große Menge von Daten, die in der Trainingsphase im Klassifikationsproblem verarbeitet werden müssen, gibt es somit einen deutlichen Unterschied in den resultierenden Speicherkosten und Laufzeiten.

Die Matrix $G_{\underline{l}}$ kann zusammen mit der Matrix $C_{\underline{l}}$ in einer gemeinsamen Matrixstruktur gespeichert werden. Somit ist der Speicherbedarf der Matrix $B_{\underline{l}}$ immer zusätzlich zur Matrix $C_{\underline{l}}$, was sich insbesondere bei sehr vielen Daten deutlich bemerkbar macht, da er mit der Zahl der Datenpunkte M skaliert. In Bezug auf die Gesamtkosten für das Berechnen der Matrizen, $O(3^d \cdot N + 2^{2d} \cdot M)$ für $C_{\underline{l}}$ und $G_{\underline{l}}$ gegen $O(3^d \cdot N + d \cdot 2^d \cdot M)$ für $C_{\underline{l}}$ und $B_{\underline{l}}$, wirkt sich bei kleiner Dimensionszahl der Faktor 2^d noch nicht stark aus. Aber ab Dimension 6 macht sich der Unterschied zwischen den Faktoren 2^d, hier $2^6 = 64$, und 2^{2d}, hier $2^{12} = 4096$, immer deutlicher bemerkbar. Für höhere Dimensionen hat somit die zweite Variante (5.6) einen Vorteil bei der Rechenzeit für das Aufstellen der Matrizen.

Die Kosten einer Matrix-Vektor-Multiplikation bei der Behandlung eines Teilproblems weisen eine Komplexität der Ordnung $O(3^d \cdot N)$ für den ersten Ansatz (5.5) auf, da beide Matrizen $C_{\underline{l}}$ und $G_{\underline{l}}$ in einer Matrixstruktur gespeichert werden können. Eine Komplexität

der Ordnung $O(3^d \cdot N + 2^d \cdot M)$ wird mittels des zweiten Ansatzes (5.6) erhalten. Im Gegensatz zu den Komplexitäten der Berechnung der Matrizen verhält sich somit bei der Matrix-Vektor-Multiplikation der erste Ansatz immer vorteilhafter als der zweite Ansatz.

Es ist jedoch zu beachten, dass die Speicher- und Laufzeitkomplexitäten exponentiell von der Dimension d abhängen. Auf Grund der Beschränkungen des Speichers moderner Arbeitsplatzrechner (512 MByte – 2 GByte) können wir deswegen in dieser Form, d.h. wenn wir die Matrizen aufstellen und sie im Speicher des Rechners halten, nur Fälle bis $d \approx 13$ behandeln. Wie erwähnt können wir es vermeiden, diese Matrizen explizit aufzustellen, und stattdessen die notwendigen Einträge on-the-fly berechnen, wenn sie in der CG-Iteration gebraucht werden. Auf diese Weise tauschen wir Speicher gegen Rechenzeit ein. Bis zur Dimension $d = 26$ wäre eine Rechnung mit iterativem Gleichungslöser und on-the-fly-Berechnung der Matrix-Vektor-Multiplikation auf dem gröbsten Gitter mit 2^d-Punkten im Speicher moderner Arbeitsplatzrechner noch möglich. Ein Vektor des Datentyps *double* der Länge 2^{26} benötigt genau 512 Megabyte, bei iterativer Rechnung sind davon mindestens 3 in Gebrauch. Die Kosten des Aufstellens der Matrizen fallen nun in jeder Matrix-Vektor-Multiplikation im CG-Verfahren an, somit gibt es jetzt keinen Vorteil mehr für den ersten Ansatz (5.5). Im Gegenteil, auf Grund des Unterschieds in der d-Abhängigkeit (2^{2d} gegen 2^d) verhält sich der zweite Ansatz (5.6) besser. Da nun aber für jede Matrix-Vektor-Operation die Datenmatrix on-the-fly berechnet werden muss, benötigt diese Form des Verfahrens bei der Behandlung von großen Datenmengen äußerst viel Rechenzeit.

Betrachten wir nun den Aufwand der Auswertung des mit der Kombinationstechnik berechneten Klassifikators an neu gegebenen Punkten $\{\tilde{\underline{x}}_i\}_{i=1}^{\tilde{M}}$. Aus (4.14) sehen wir direkt, dass dies $O(d \cdot n^{d-1} \cdot 2^d \cdot \tilde{M})$ Operationen bei der normalen Kombinationstechnik nach sich zieht, bei der verallgemeinerten Kombinationstechnik ersetzt die Anzahl der beitragenden Gitter den Faktor $d \cdot n^{d-1}$. Der Faktor 2^d steht für den Aufwand der jeweils notwendigen bilinearen Interpolation.

Es sei darauf hingewiesen, dass die Lösung der verschiedenen in der Kombinationstechnik auftretenden Probleme durch eine iterative Methode wie das konjugierte Gradientenverfahren sicherlich nicht optimal ist. Hier hängt die Zahl der Iterationen, die notwendig ist, um eine vorgegebene Genauigkeit zu erzielen, von den Konditionszahlen der entsprechenden Systemmatrizen ab. Es wird zu untersuchen sein, ob mit einem geeigneten iterativen Mehrgitterverfahren die linearen Gleichungssysteme mit einer Konvergenzrate unabhängig von der Auflösung des Gitters gelöst werden können. Weiterhin lässt sich auch Robustheit, d.h. eine Konvergenzrate unabhängig von der Anisotropie, die durch Unterschiede in den Maschenweiten bzgl. verschiedener Koordinatenrichtungen entsteht, erhalten. Dies lässt sich erzielen, indem Semi-Vergröberungstechniken [GO95] eingesetzt werden. Genauer zu untersuchen ist dabei der Einfluss der diskreten Datenmenge auf das Mehrgitterverfahren und die dafür benötigten Prolongations- und Restriktionsoperatoren.

Schließlich ist die Abhängigkeit der Konvergenzrate vom Regularisierungsparameter eine weitere Schwierigkeit. Da $B_{\underline{l}}^t B_{\underline{l}}$ eine in gewisser Weise verschmierte lokale Mischung zwischen der Identität und der Null-Matrix darstellt, besteht Hoffnung, eine insgesamt robuste Methode durch die Ansätze von [Osw91, Stü99, OR00] zu erhalten.

All diese Techniken sind mittlerweile im zwei- und drei-dimensionalen Fall, bis auf den

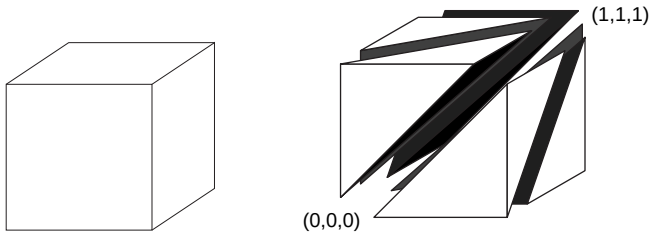


Abbildung 5.2. *Freudenthal-Kuhn-Triangulierung eines dreidimensionalen Einheitswürfels.*

Einfluss der diskreten Datenverteilung, wohl verstanden. Sie sollten analog auch im höherdimensionalen Fall funktionieren. Jedoch gibt es bisher unseres Wissens noch keine Implementierung einer hochdimensionalen robusten Mehrgittermethode mit Semi-Vergröberung. Momentan verwenden wir deswegen nur die Methode der konjugierten Gradienten mit diagonaler Vorkonditionierung.

5.2 Simpliziale Basisfunktionen

Bisher haben wir d-lineare Basisfunktionen basierend auf einem Tensorproduktansatz verwendet. Wie beschrieben verursachen diese im Algorithmus pro Datenpunkt einen Aufwand von mindestens 2^d Operationen, für höhere Dimensionen wird dementsprechend sehr viel Rechenzeit für die Bearbeitung der Datenpunkte benötigt. Andererseits wird bei vielen Finite Element-Rechnungen auf den Gittern mit einer simplizialen Diskretisierung gearbeitet, in zwei Dimensionen sind dies Dreiecke, in drei Dimensionen Tetraeder. Diese können nun auch auf den Gittern benutzt werden, die in der Sequenz von Gittern der Kombinationstechnik auftreten. Wir benutzen dabei die so genannte *Freudenthal-Kuhn-Triangulierung* [Fre42, Kuh60], um jeden rechteckigen Block von Gitterpunkten aufzulösen, siehe Abbildung 5.2.

Auf dieser Diskretisierung werden nun lineare Basisfunktionen definiert, dabei ist wie üblich jedem Gitterpunkt $\underline{x}$ eindeutig eine Basisfunktion $\varphi_{\underline{x}}$ zugeordnet. Jeder Gitterpunkt ist Eckpunkt einer gewissen Anzahl von Simplexen, der Träger der zugehörigen Basisfunktion besteht aus deren Vereinigung. Auf jedem dieser Simplexe wird nun die lineare Basisfunktion $\varphi_{\underline{x}}$ so definiert, dass sie auf dem zugehörigen Gitterpunkt $\underline{x}$ Eins ist und auf den anderen d Punkten des Simplex Null, d.h. die Basisfunktion entspricht den baryzentrischen Koordinaten innerhalb eines Simplex. Bei der Nutzung dieser Diskretisierung und den zugehörigen linearen Basisfunktionen sind nun deutlich weniger Operationen pro Datenpunkt notwendig als bei dem d-linearen Ansatz. Denn die Überlappungen der Träger der verschiedenen Basisfunktionen sind deutlich reduziert und somit sind pro Datenpunkt nur noch $d+1$ Basisfunktionen ungleich null. Damit wächst die Zahl der Operationen, die pro Instanz für die Berechnung der von ihm abhängigen Einträge in den Datenmatrizen $G_{\underline{l}}$ oder $B_{\underline{l}}$ benötigt werden, nicht mehr exponentiell in d, sondern nur noch quadratisch beziehungsweise linear.

	d-lineare Basisfunktionen		lineare Basisfunktionen		
	$C_{\underline{l}}$	$G_{\underline{l}} := B_{\underline{l}}^t \cdot B_{\underline{l}}$	$C_{\underline{l}}$	$G_{\underline{l}} := B_{\underline{l}}^t \cdot B_{\underline{l}}$	$B_{\underline{l}}$
Speicher	$O(3^d \cdot N)$	$O(3^d \cdot N)$	$O(2d \cdot N)$	$O(2^d \cdot N)$	$O(d \cdot M)$
Aufstellung	$O(3^d \cdot N)$	$O(2^{2d} \cdot M)$	$O(2d \cdot N)$	$O((d+1)^2 \cdot M)$	$O(d \cdot M)$
MV-Multiplikation	$O(3^d \cdot N)$	$O(3^d \cdot N)$	$O(2d \cdot N)$	$O(2^d \cdot N)$	$O(d \cdot M)$

Tabelle 5.2. *Komplexitäten für das Speichern, das Aufstellen und die Matrix-Vektor-Multiplikation für verschiedene Matrizen, die bei der Kombinationstechnik auf dem Gitter $\Omega_{\underline{l}}$ für d-lineare (links) und lineare Basisfunktionen (rechts) auftreten.*

Für die Auswertung einer Funktion an einem Punkt $\underline{x}$ bei der Nutzung der Freudenthal-Kuhn-Triangulierung ist es notwendig, den Simplex zu finden, in dem der Datenpunkt liegt, und die Werte der Basisfunktionen zu bestimmen, die in diesem Simplex ungleich Null sind. Im ersten Schritt ist dabei das Rechteck im Gitter $\Omega_{\underline{l}}$ zu finden, in dem der Punkt $\underline{x}$ liegt. Zur einfacheren Notation transformieren wir im Weiteren dieses Rechteck und entsprechend den Punkt $\underline{x}$ auf $[0,1]^d$. π sei nun eine Permutation für die $1 \geq x_{\pi(1)} \geq \ldots \geq x_{\pi(d)} \geq 0$ gilt. Damit ist der Simplex, in dem der Datenpunkt $\underline{x}$ liegt, durch die Eckpunkte $(v_0, \ldots, v_d)$ eindeutig festgelegt, wobei $v_o := \underline{0}$ und $v_i := v_{i-1} + \underline{e}_{\pi(i)}$ für $i = 1, \ldots, d$. Die baryzentrischen Koordinaten ergeben sich als $\lambda_m := x_{\pi(d)}$, $\lambda_i := x_{\pi(i)} - x_{\pi(i-1)} \geq 0$ und $\lambda_0 := 1 - \sum_{i=1}^{d} \lambda_i$. Damit ist die Auswertung der Basisfunktionen am Datenpunkt $\underline{x}$ gegeben.

Weiterhin ist auch die Zahl der Matrixeinträge bei der Speicherung der Matrix $B_{\underline{l}}$ entsprechend geringer. Die Speicherung der Matrix $G_{\underline{l}}$ benötigt ebenfalls weniger Matrixeinträge, denn bei simplizialer Diskretisierung hat ein innerer Gitterpunkt nun mit höchstens $2^{d+1} - 2$ Gitterpunkten einen gemeinsamen Simplex und damit einen potentiellen Matrixeintrag, im Gegensatz zu $3^d - 1$ beim Tensorproduktansatz. Die Speicherkomplexität der Matrix $C_{\underline{l}} + G_{\underline{l}}$ beträgt somit $O(2^d \cdot N)$. In Tabelle 5.2 ist ein Vergleich beider Diskretisierungsarten dargestellt. Es ist dabei zu beachten, dass für allgemeine Regularisierungsoperatoren $\mathcal{S}$ die Speicherkomplexität für $C_{\underline{l}}$ mit $O(2^d \cdot N)$, der Anzahl der Nachbarn, skaliert. Für unsere spezielle Wahl von $\mathcal{S} = \nabla$ treten allerdings strukturelle Nulleinträge auf, da nur die $d+1$ Nachbarn über die Kanten einen Beitrag für die Matrix $C_{\underline{l}}$ liefern. Somit hat die Gesamtmatrix $C_{\underline{l}} + G_{\underline{l}}$ mindestens $(d+1) \cdot N$ Einträge. Je nach Menge und Verteilung der Datenpunkte werden aber weiterhin für jeden Gitterpunkt bis zu $2^{d+1} - 2$ Matrixeinträge benötigt.

Allerdings hängt auch bei der simplizialen Diskretisierungsvariante die Komplexität des benötigten Speichers und der Rechenzeit auf jeden Fall exponentiell von der Dimension d ab, bedingt durch die Mindestgröße 2^d eines Teilgitters. Bei der expliziten Speicherung der Matrizen können aber, im Vergleich zu den d-linearen Funktionen, einige Dimensionen mehr betrachtet werden. Der entscheidende Vorteil bleibt aber die deutlich geringere Komplexität bei der Verarbeitung der Datenpunkte. Hier erreichen wir durch die Verwendung linearer Basisfunktionen auf Simplexen eine in d quadratische beziehungsweise lineare Komplexität, im Gegensatz zum in d exponentiellen Aufwand bei Tensorprodukt-

funktionen. Somit können in höheren Dimensionen große Datenmengen in sinnvoller Zeit verarbeitet werden.

Die im vorherigen Abschnitt gemachten Erläuterungen zu einer on-the-fly Berechnung der Matrixeinträge gelten analog für die simpliziale Diskretisierungsvariante. Eine weitere Option für den Ansatz (5.6) besteht nun darin, bei der on-the-fly-Rechnung nicht beide Matrizen $C_{\underline{l}}$ und $B_{\underline{l}}$ jeweils neu zu berechnen, da die Datenmatrix $B_{\underline{l}}$ bei effizienter Speicherung höchstens doppelt soviel Speicherplatz benötigt wie die Daten alleine. Dies gilt, da für jeden Datenpunkt die $d+1$-Einträge des Simplex, in denen sich der Datenpunkt befindet, gespeichert werden müssen. Für die Speicherung einer dünnbesetzten Matrix muss für jeden Matrixeintrag zusätzlich dessen Index gespeichert werden. So kann die Matrix $B_{\underline{l}}$ in (sizeof(`double`)·(d+1)+sizeof(`index_t`))·M Speicher abgelegt werden, die Datenmenge alleine benötigt sizeof(`double`)$\cdot d \cdot M$. Damit kann ein Kompromiss zwischen Speicherplatz- und Rechenzeitverbrauch erreicht werden.

Die Approximationseigenschaften dieser simplizialen Variante der Kombinationstechnik sind allerdings in der Theorie bisher nicht behandelt. Die Beweise der in Kapitel 4 vorgestellten Approximationsaussagen basieren auf der Auslöschung von Termen in der punktweisen Fehlerentwicklung auf jedem Gitter. Zum einen werden dazu Aussagen der Form

$$f - f_{\underline{l}} = \sum_{i=1}^{d} \sum_{j_1,\ldots,j_m \subset 1,\ldots,d} c_{j_1,\ldots,j_m}(h_{j_1},\ldots,h_{j_m}) h_{j_1}^p \cdot \ldots \cdot h_{j_m}^p,$$

mit beschränkten $c_{j_1,\ldots,j_m}(h_{j_1},\ldots,h_{j_m})$ benötigt. Dies gilt bei einem Tensorproduktansatz für die zu Grunde liegenden Basisfunktionen. Für jedes Gitter weisen simpliziale Basisfunktionen ähnliche Approximationsordnungen auf, allerdings beziehen sich diese Fehlerabschätzungen auf die maximale Kantenlänge eines Simplex und liefern somit keine Fehlerentwicklung der obigen Form. Darüberhinaus ist die für diese Abschätzung benötige Quasi-Uniformität der Diskretisierung nicht gegeben. Zum anderen muss für die erwähnte Auslöschung eine Schachtelung der diskreten Funktionenräume vorliegen, was bei einer simplizialen Diskretisierung nicht der Fall ist.

Die in Kapitel 6 präsentierten numerischen Ergebnisse rechtfertigen allerdings diesen Ansatz. Die Ergebnisse mit linearen Basisfunktionen sind, wenn überhaupt, nur geringfügig schlechter als die mit d-linearen Basisfunktionen, und wir können somit auf ähnliche Approximationsaussagen hoffen.

5.2.1 Berechnung der Steifigkeitsmatrix

Ein wichtiger Aspekt bei der algorithmischen Realisierung der simplizialen Diskretisierung ist die Aufstellung der Matrix $C_{\underline{l}}$, oft *Steifigkeitsmatrix* genannt. In herkömmlichen Anwendungen bei der numerischen Behandlung von partiellen Differentialgleichungen mit Finiten Elementen in zwei oder drei Dimensionen spielt dessen Komplexität keine große Rolle. Dort wird auf den einzelnen Simplexen eine lokale Steifigkeitsmatrix berechnet und aus diesen die globale Matrix berechnet. Dieses ist insbesondere bei adaptiven Finite Element-Methoden notwendig.

Mit steigender Zahl der Dimensionen wirkt sich aber merklich aus, dass $d!$ Simplexe benötigt werden, um einen Würfel aufzuteilen. Es müssten somit auch $d!$ lokale Element-

steifigkeitsmatrizen berechnet werden. Ab $d \gtrsim 10$ macht sich dieser Faktor in der Rechenzeit bemerkbar und bremst den Algorithmus entscheidend. Da die vorliegenden Gitter aber eine äquidistante Struktur in den einzelnen Dimensionen aufweisen, kann die Berechnung der Steifigkeitsmatrix trotzdem auf effiziente Weise erfolgen. Wegen dieser besonderen Gitterstruktur werden nur über die Kanten eines Würfels Nicht-Null-Einträge in der Steifigkeitsmatrix erzeugt. Die Werte einer lokalen Steifigkeitsmatrix hängen von der Anzahl der Dimensionen, dem Abstand der einzelnen Gitterpunkte und dem Ort im Würfel ab. Wie in Abbildung 5.2 zu sehen, ist die Anzahl von Simplexen, die den Träger eines Elements bilden, von der Lage der zugehörigen Ecke im Würfel abhängig. Daher bekommen die Eckpunkte eine Würfels jeweils unterschiedliche Einträge in der lokalen Steifigkeitsmatrix des Würfels.

Wenn wir nun die Elementsteifigkeitsmatrizen der $d!$ Simplexe eines Würfels und deren Summierung zur Steifigkeitsmatrix des Würfels analysieren, erkennen wir gewisse wiederkehrende Strukturen, welche einfach zu berechnen sind.[1] Auf Grund der Würfelstruktur bestehen die Einträge der Steifigkeitsmatrix eines Simplex nur aus den Werten $+1$ und -1 gewichtet mit der Inversen der Maschenweite und dem Volumen des Simplex. Diese Gewichte sind für alle in einem Würfel vorhandenen Simplexe gleich, somit ergeben sich mit kombinatorischen Überlegungen, bei denen untersucht wird in wie vielen Simplexen ein Eckpunkt des Würfels enthalten ist, die folgenden Formeln zur Berechnung der lokalen Steifigkeitsmatrix C_d des Würfels. Aus diesen kann wiederum die Steifigkeitsmatrix des gesamten Gebietes zusammengesetzt werden.

Wir geben hier die lokale Steifigkeitsmatrix für ein Gitter mit Maschenweite $\underline{h}$ an. Die 2^d Eckpunkte $\underline{x}$ eines Würfels werden bezüglich des Einheitswürfels, d.h. per Transformation auf $\hat{\underline{x}} \in \{0,1\}^d$, mit $\sum_{i=1}^{d} \hat{x}_i \cdot 2^{i-1}$ durchnummeriert, d.h. lexikographisch. Für die Einträge der lokalen Steifigkeitsmatrix C_d eines Würfels ergibt sich

$$(C_d)_{j,j} = \frac{1}{d!} \prod_{i=1}^{d} h_i \sum_{i=1}^{d} \frac{M_d(j, i-1)}{h_i^2}$$

und

$$(C_d)_{j,k} = (C_d)_{k,j} = -\frac{1}{d!} \prod_{i=1}^{d} h_i \sum_{i=1}^{d} \frac{M_d(j, i-1)}{h_i^2},$$

für $k = j + 2^i,\ i = 1, \ldots, d$. Der Ausdruck vor der Summe gibt hierbei das Volumen eines Simplex an. Weiterhin ist

$$M_d := \begin{pmatrix} a_d(0,0) & \ldots & a_d(0, d-1) \\ \vdots & \ddots & \vdots \\ a_d(2^d - 1, 0) & \ldots & a_d(2^d - 1, d-1) \end{pmatrix},$$

mit

$$a_d(i,j) := ZZ_d(bit(i,j), \sum_{k=0}^{d-1} bit(i,k)).$$

[1] An dieser Stelle möchte ich meinen Dank an Rolf Rossius von der prudsys AG ausrichten, der mir mit seiner Analyse dieser Struktur zuvorgekommen ist und mir die Ergebnisse zur Verfügung gestellt hat.

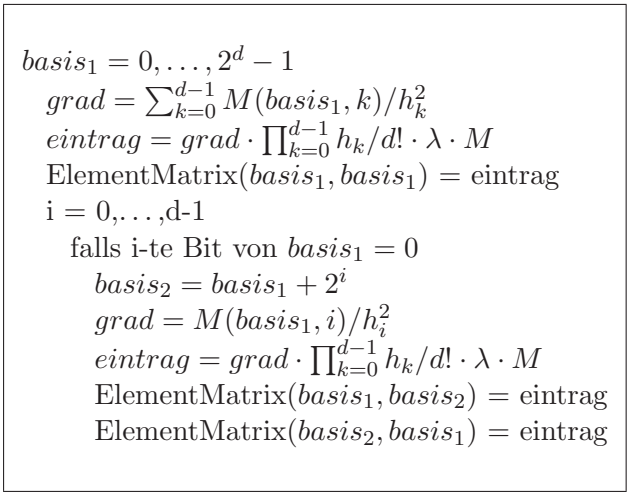


Abbildung 5.3. *Algorithmus zur Berechnung der lokalen Steifigkeitsmatrix eines Würfels bei der Verwendung einer simplizialen Diskretisierung.*

Dabei ist

$$ZZ_d(s,b) := \begin{cases} Z_d(1) & \text{falls } b = 0, \text{ oder } b = d, \\ Z_d(b) & \text{falls } 2b = d, \\ Z_d(b + (s \text{ xor } 1)) & \text{falls } 2b < d, \\ Z_d(d - b + s) & \text{falls } 2b > d \end{cases}$$

und

$$Z_d(i) := (i-1)!\ (d-i)!.$$

$$bit(i,j) := (i/2^j) \mod 2 \quad (\text{entspricht } \texttt{(i >> j) \& 1})$$

gibt an, ob das j-te Bit der Ganzzahl i gesetzt ist. Der Algorithmus zur Berechnung der lokalen Steifigkeitsmatrix eines d-dimensionalen Würfels ist in Abbildung 5.3 dargestellt.

Dieser „Trick“ ist allerdings nur möglich, wenn eine äquidistante Punktverteilung vorliegt. Bei ortsadaptiven Gittern, wie sie bei der numerischen Behandlung von partiellen Differentialgleichungen heutzutage üblich sind, würde dies in höheren Dimensionen nicht funktionieren.

5.3 Parallelisierung

Bei der Parallelisierung von Algorithmen wird zwischen verschiedenen Ebenen der Parallelisierung unterschieden. Meist wird dabei von grobkörniger Granularität, bei der große Aufgaben unabhängig voneinander parallel bearbeitet werden können, und feinkörniger Granularität gesprochen, hierbei können entsprechend kleine Teilaufgaben parallel ausgeführt werden [GO93]. Manchmal wird zwischen diesen Definitionen noch der Begriff der mittelkörnigen Parallelität verwendet, die zwischen längeren Sequenzen von Instruktionen auftritt.

Bei der Umsetzung ist das Ahmdahlsche Gesetz [GO93] zu beachten, das unter Anderem impliziert, dass die Kosten der parallel auszuführenden Teiloperationen wesentlich größer sein sollten als die Kosten der Kommunikation und Synchronisation. Je feiner die Granularität des parallelen Ansatzes, desto schwieriger wird im Allgemeinen die Parallelisierung sein, eine grobkörnige Parallelisierung ist somit zu bevorzugen. Welche Art der Parallelisierung sinnvoll genutzt werden kann hängt auch von der verwendeten Hardware ab. Clustersysteme mit verteiltem Speicher, bei dem Rechner über ein Netz verknüpft sind, die mittels Message-Passing kommunizieren, haben einen höheren Kommunikationsaufwand als Systeme, bei denen mehrere Prozessoren einen gemeinsamen Speicher nutzen.

Allerdings sind in vielen Fällen die Ansätze für eine grobkörnige Parallelisierung eines Verfahrens limitiert. Sofern eine grobkörnige Parallelität die vorhandenen Ressourcen nicht genügend ausnützt, kann eine zusätzliche feinkörnige Parallelisierung von Nutzen sein.

Die Kombinationstechnik ist einfach auf einer grobkörnigen Granulierung parallelisierbar [Gri92]. Zusätzlich kann eine feinkörnige Parallelisierung für jedes Teilproblem der Kombinationstechnik auf Systemen mit gemeinsamen Speicher erreicht werden. Beide Ansätze können kombiniert und gleichzeitig genutzt werden, was zu einem parallelen Verfahren führt, das für Cluster von Mehrprozessormaschinen gut geeignet ist.

Eine weitere Anwendungsmöglichkeit entsteht, wenn die einzelnen Teilprobleme der Kombinationstechnik bereits soviel Arbeitsspeicher benötigen, dass auf einer Mehrprozessormaschine nur ein Teilproblem im Hauptspeicher bearbeitet werden kann. Durch die zusätzliche feinkörnige Parallelisierung können auf solchen Systemen die parallelen Ressourcen weiterhin effizient genutzt werden.

5.3.1 Parallelisierung über der Gittersequenz

Die linearen Gleichungssysteme (5.2) können für die verschiedenen Teilprobleme der Gittersequenz unabhängig voneinander berechnet werden. Somit kann die Berechnung ohne weiteres parallel erfolgen, indem jeder Prozess die Lösung einer gewissen Anzahl von Gittern berechnet. Im Extremfall stehen genau so viele Prozessoren zur Verfügung wie Gitter in der Sequenz enthalten sind. Jeder Prozessor berechnet dann nur die Lösung eines Teilproblems. Ein Kontrollprozess steuert hierbei die Verteilung der Teilprobleme und sammelt die Ergebnisse für die Berechnung der Dünngitterfunktion wieder ein. Die hierfür benötigte Rechenzeit trägt im Verhältnis zur Berechnung eines Teilproblems nur äußerst geringfügig zur Gesamtzeit bei.

Da die Kosten der Berechnung eines Teilproblems ungefähr a-priori bekannt sind, ist eine einfache aber effektive statische Lastbalancierung möglich, siehe [GHSZ93]. Dabei erfolgt die Sortierung der Teilprobleme über die absteigende Größe des Gitters. Bei großen Datenmengen dominiert wie erwähnt die Datenverarbeitung die Rechenzeit, so dass in diesem Fall die verschiedenen Gitter annähernd die gleiche Rechenzeit benötigen.

5.3.2 Parallelisierung über Daten

Um die Matrixeinträge für $B^t \cdot B$ in (5.2) für jede Dateninstanz zu berechnen, ist das Produkt der Werte aller Basisfunktionen φ_j, φ_k zu bilden, welche ungleich 0 am Punkt $\underline{x}_i$ sind. Die Ergebnisse werden in die Matrix jeweils an die Position (j, k) geschrieben, d.h.

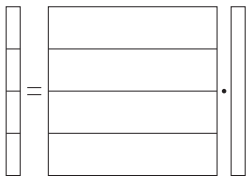


Abbildung 5.4. *Aufteilen von Matrix und Vektor in p Teile (hier $p = 4$) für die parallele Matrix-Vektor-Multiplikation auf einem Mehrprozessorsystem mit gemeinsamen Speicher.*

$(B^t \cdot B)_{j,k} = \sum_{i \leq M} \varphi_j(\underline{x}_i) \cdot \varphi_k(\underline{x}_i)$.

Die innere Berechnung $\varphi_j(\underline{x}_i) \cdot \varphi_k(\underline{x}_i)$ hängt jeweils nur von einem Datenpunkt ab, daher kann jeder Datenpunkt unabhängig von den anderen bearbeitet werden. Die $M \times d$ große Datenmenge wird dazu in p Abschnitte der Größe $M/p \times d$ aufgeteilt, wobei p die Anzahl der Prozessoren des Systems mit gemeinsamen Speicher ist. Jeder Prozessor berechnet nun die Matrixeinträge, die aus einer dieser Teilmengen entstehen. Für die notwendige Synchronisation des seriellen Schreibens der Teilergebnisse in die gemeinsame Matrixstruktur treten zusätzliche Kosten auf.

Analog kann die Berechnung der Klassifikatorwerte auf einer Datenmenge während der Auswertungsphase parallelisiert werden.

5.3.3 Parallelisierung des Gleichungslösers

Nachdem die Matrix aufgebaut ist, kann auf einem Mehrprozessorsystem mit gemeinsamen Speicher auch die Lösung des linearen Gleichungssystems (5.2) mit feinkörniger Parallelisierung versehen werden. Wir nutzen einen iterativen Löser mit Jacobi-Vorkonditionierung, dabei wird der größte Teil der Rechenzeit für Matrix-Vektor-Multiplikationen der Form $\alpha = A\beta$ benötigt. Der Vektor α mit Länge N kann nun virtuell in p Teile zerlegt werden. Jeder Prozessor berechnet nun das Ergebnis der Matrix-Vektor-Multiplikation für einen Vektor der Größe N/p, siehe Abbildung 5.4. In dieser Weise wird jede Vektorkomponente nur von einem Prozessor verändert, eine Synchronisation des Schreibens der Ergebnisse in die Matrixstruktur ist somit nicht notwendig.

In Abbildung 5.5 ist der gemessene Speedup der feinkörnigen Parallelisierung für eine Beispielrechnung aus [GG01] dargestellt.

5.3.4 Kombination der grob- und feinkörnigen Parallelität

In den meisten Fällen erzielt eine grobkörnige Parallelisierung die höchsten Geschwindigkeitsvorteile und ist die bevorzugte Parallelisierungsstrategie. Die Rechnungen in [GG01] bestätigen dieses Verhalten der Parallelisierung der Kombinationstechnik für Klassifikationsprobleme anhand von Beispielrechnungen.

Wie bereits erwähnt kann eine Situation entstehen, in der einzelne Teilprobleme der Kombinationstechnik bereits soviel Arbeitsspeicher benötigen, dass auf einer Mehrprozessormaschine nur ein Teilproblem im Hauptspeicher bearbeitet werden kann. Hier ist die gleichzeitige Nutzung der feinkörnigen Parallelisierung von großem Nutzen.

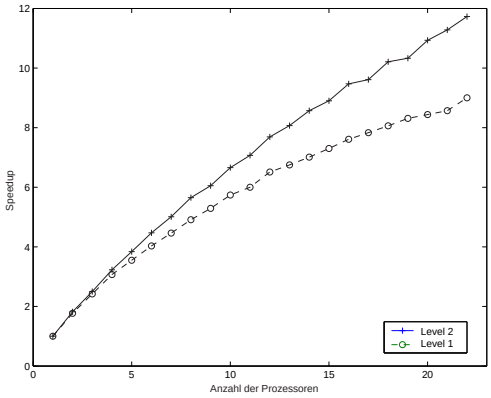


Abbildung 5.5. *Gemessener Speedup für einen zehndimensionalen Datensatz aus [GG01], die Rechnungen haben für Level 1 und 2 stattgefunden.*

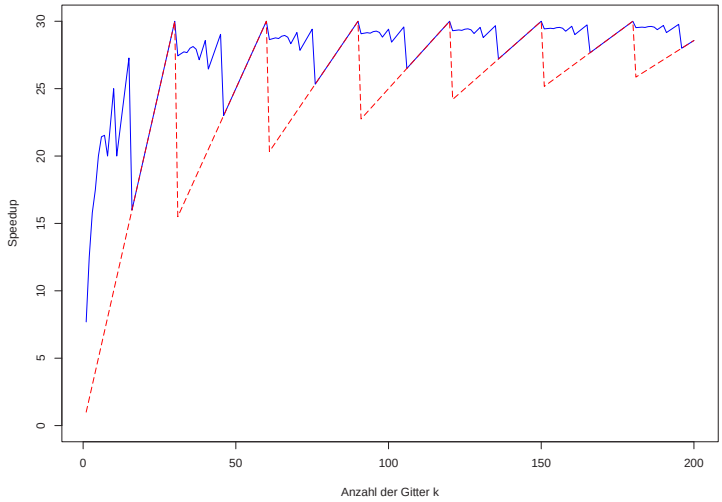


Abbildung 5.6. *Theoretischer Speedup der grobkörnigen Parallelisierung (gestrichelte Linie) und der Kombination der grobkörnigen und feinkörnigen Parallelisierung (durchgezogene Linie).*

Weiterhin kann der Fall eintreten, dass die Anzahl der Gitter ungünstig ist und dadurch die vorhandenen parallelen Ressourcen nicht vollständig genutzt werden. Diese tritt dann ein, wenn die Anzahl der zu berechnenden Gitter nicht der Anzahl der zur Verfügung stehenden Prozessoren entspricht. Durch die zusätzliche Anwendung der feinkörnigen Parallelisierung können nun die Ressourcen effizienter genutzt werden.

Nehmen wir nun an, dass p Prozessoren bei einer parallelen Berechnung mit k Gittern benutzt werden. Der maximal mögliche Speedup der grobkörnigen Parallelisierung beträgt $k/\lceil k/p \rceil$, sofern wir davon ausgehen, dass alle Gitter ähnlich viel Rechenzeit benötigen.

Dies ist bei der Behandlung von großen Datenmengen der Fall, da hier der größte Teil der Rechenzeit bei der Bearbeitung der Daten anfällt. Anders wäre dies bei der numerischen Behandlung von partiellen Differentialgleichungen, dort benötigen kleine Gitter deutlich weniger Rechenzeit als große. Der Fall $p = 30$ und $k = 1, \ldots, 200$ ist in Abbildung 5.6 dargestellt.

Um die feinkörnige Parallelität zu nutzen, wird erst der grobkörnige Ansatz für $p\lfloor k/p \rfloor$ Teilprobleme angewandt. In einem zweiten Schritt werden die Prozessoren gleichmäßig auf die übrigen Gitter verteilt. Nach dem ersten Schritt sind somit

$$p_x = k - p\lfloor k/p \rfloor < p$$

Teilaufgaben übrig. Damit ergeben sich $p_l = \lfloor p/p_x \rfloor$ Prozessoren für jedes verbleibendes Gitter. Diese werden mit der feinkörnigen Parallelisierung berechnet, es ergibt sich somit für diesen Ansatz unter Beachtung des Ahmdahlschen Gesetz ein Speedup von

$$S_{p,k} = \frac{k}{\lfloor k/p \rfloor + \mathrm{sign}(p_x) * (0.1 + 0.9/p_l)},$$

wiederum dargestellt in Abbildung 5.6. Wir gehen hier modellhaft von konstant zehn Prozent des seriellen Rechenaufwandes aus, der nicht parallelisiert werden kann. Es sei darauf hingewiesen, dass in [GG01] ungefähr fünf Prozent beobachtet werden.

Dieser Ansatz kann noch weiter verfeinert werden durch eine weitergehende Verknüpfung beider parallelen Strategien. Dabei werden $p/2$ Gitter mit der grobkörnigen Parallelität behandelt und für jedes Teilproblem werden zwei Prozessoren eines Mehrprozessorsystems mit gemeinsamen Speicher genutzt. Solche Verfeinerungen des Ansatzes würden die in Abbildung 5.6 noch vorhandenen größeren Sprünge im Speedup verkleinern.

Wir haben in diesem Kapitel Aspekte der praktischen Realsierung der Dünngitter-Kombinationstechnik zur Funktionsrekonstruktion untersucht. Es stellt sich heraus, dass die Komplexität des Verfahrens linear bezüglich der Zahl der Datenpunkte ist. Allerdings tritt im ursprünglichen Ansatz ein in der Dimension d exponentieller Aufwand pro Instanz auf. Mit Hilfe einer simplizialen Diskretisierung kann dieser Faktor auf quadratisch in d substantiell reduziert werden. Somit ist die Komplexität des Verfahrens bezüglich der Zahl der Datenpunkte deutlich geringer als bei anderen maschinellen Lernverfahren, die einen nichtlinearen Klassifikator mit nichtlinearen Aufwand in der Zahl der Instanzen bestimmen. Darüberhinaus haben wir betrachtet, wie das Verfahren grobkörnig und feinkörnig parallelisiert werden kann und wie beide Parallelisierungsstrategien zur Effizienzsteigerung kombiniert werden können.

Im folgenden Kapitel werden wir dann zum einen empirisch die Komplexitätseigenschaften des Verfahrens an einigen Beispielen untersuchen. Zum anderen werden wir die Qualität der Ergebnisse im Vergleich zu anderen maschinellen Lernverfahren betrachten.

Kapitel 6

Numerische Ergebnisse

In diesem Kapitel betrachten wir die Qualität der Klassifikationsergebnisse des vorgestellten Verfahrens zur Funktionsrekonstruktion mit dünnen Gittern anhand einer Reihe von synthetischen und realen Datensätzen. Zunächst untersuchen wir die Konvergenzeigenschaft der diskreten Lösungen am Beispiel eines zweidimensionalen Regressionsbeispiels unter anderem mit Fehlernormen, wie sie bei bei der numerischen Behandlung von partiellen Differentialgleichungen üblich sind. Wir vergleichen dazu die verschiedenen Ergebnisse auf vollen und dünnen Gittern mit einer Referenzlösung auf einem hochaufgelösten vollen Gitter.

Anschließend werden Messmethoden und graphische Analysemethoden aus dem Bereich des maschinellen Lernens und des Data Mining beschrieben, um die Ergebnisse des Klassifikationsverfahrens beurteilen und vergleichen zu können. Weiterhin stellen wir Methoden zur Bestimmung der Parameter des Verfahrens vor. Bei realen Anwendungen spielt dies eine wichtige Rolle, da es primär nicht darum geht, die gegebenen Daten gut zu approximieren, sondern vor allem auf neuen Daten gute Klassifikationsergebnisse zu erzielen.

Diese Methoden wenden wir ausführlich für einen einfachen synthetischen zweidimensionalen Datensatz an. Weiterhin präsentieren wir die Resultate für eine Reihe von Benchmark-Datensätzen, geben dabei einen Überblick über die Anwendungsmöglichkeiten des Data Mining und vergleichen die Qualität der Ergebnisse mit denen andere Verfahren. Schließlich führen wir einige Experimente mit einer dimensionsadaptiven Kombinationstechnik durch.

6.1 Konvergenz der diskreten Lösung

Wir betrachten im Folgenden das Verhalten der numerischen Lösung auf verschiedenen Diskretisierungsstufen im Vergleich zu einer hoch aufgelösten diskreten Lösung. Desweiteren werden Betrachtungen zur Abhängigkeit des Konvergenzverhaltens von der Größe der Datenmengen durchgeführt.

Hierzu benutzen wir im Folgenden einen synthetisch erzeugten zweidimensionalen Datensatz mit kontinuierlichen Werten, d.h. wir betrachten ein Regressionsproblem. Die Datenkoordinaten sind zufällige Positionen im Gebiet $[0,1]^2$, die zu approximierenden Funk-

tion ist

$$f(x,y) = e^{-(x^2+y^2)} + x \cdot y.$$

Da das grundsätzliche Verhalten des Verfahrens dargestellt werden soll, wird kein künstlicher Fehler eingeführt. Wir betrachten jeweils das Verhalten bei hundert, tausend, zehntausend, hunderttausend und einer Million Datenpunkte.

Dabei nutzen wir als Normen der Differenzfunktionen $f_{12} - f_n$ beziehungsweise $f_{12} - f_n^c$

- den l_2-Fehler auf den Gitterpunkten

$$|\alpha|_{l_2}^2 = \sum_{i=1}^{N} |\alpha_i|^2, \quad \text{wobei } f = \sum_{i=1}^{N} \alpha_i \varphi_i,$$

- den Fehler in der H^1-Seminorm

$$|f|_{H^1}^2 = \int (\nabla f, \nabla f) = \alpha^T \mathcal{C} \alpha, \quad \text{mit } \mathcal{C} \text{ nach (5.3)},$$

- den l_2-Fehler auf den Datenpunkten

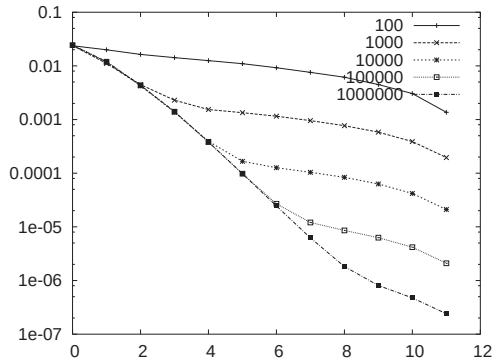
$$|f|_{l_2}^2 = \sum_{i=1}^{M} f(\underline{x}_i)^2 \qquad \text{und}$$

- den Fehler im Funktional R (5.1)

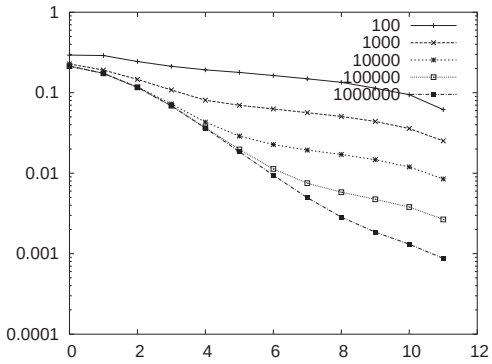
$$|f|_R = |f|_{l_2}^2 + \lambda |f|_{H_1}^2.$$

In Abbildung 6.1 ist das Konvergenzverhalten auf vollen Gittern für diesen Datensatz mit $\lambda = 0.01$ dargestellt, Rechnungen mit anderen Regularisierungsparametern liefern ähnliche Ergebnisse. Auf der horizontalen Achse ist die Verfeinerungstiefe durch ihren Level n angetragen, auf der vertikalen Achse der Fehler zwischen der Lösung des jeweiligen Levels und der Lösung auf dem vollen Gitter Ω_{12} mit 4097×4097 Punkten. In Tabelle 6.1 ist der levelweise Abfall der verschiedenen Fehler für das Beispiel mit einer Million Datenpunkten aufgetragen.

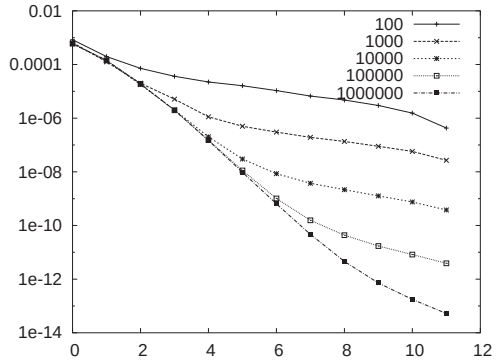
Bei der Betrachtung des Konvergenzverhaltens der numerischen Lösungen von partiellen Differentialgleichungen wird bei einer Diskretisierung mit linearen Basisfunktionen und unter der Voraussetzung von bestimmten Glattheitseigenschaften eine h^2-Konvergenz in der L_2-Norm erwartet, wobei h wie üblich die Maschenweite bezeichnet. Wie zu sehen ist, trifft dies bei der vorliegenden Art von Problemen nicht durchgehend zu. Solange ausreichend Datenpunkte vorhanden sind, ist die erwartete h^2-Konvergenz zu beobachten. Wenn aber nur wenige Dateninstanzen im Vergleich zur Gitterauflösung benutzt werden, bricht die Konvergenz ein. Die Gitterauflösung, bei der dieser Wechsel im Konvergenzverhalten stattfindet, steigt mit der Zahl der Datenpunkte. Bei einhundert Punkten ist durchweg keine h^2-Konvergenz feststellbar.



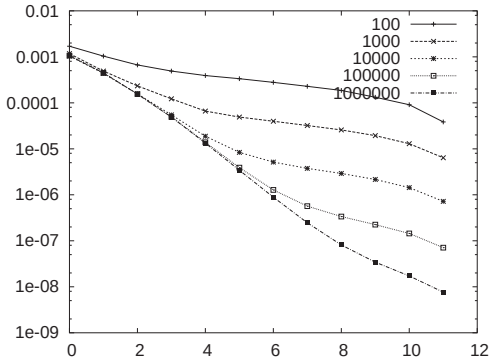
(a) l_2-Fehler auf den Gitterpunkten



(b) H^1-Seminorm-Fehler



(c) l_2-Datenfehler



(d) Fehler des Funktionals R

Abbildung 6.1. *Konvergenzverhalten der Lösung auf vollen Gittern im Vergleich zur Lösung des vollen Gitters der Stufe $n = 12$ für $\lambda = 0.01$. Dargestellt ist auf der horizontalen Achse die Diskretisierungsstufe n und auf der vertikalen der Fehler in logarithmischer Skala.*

Anhand von Tabelle 6.1 wird sichtbar, dass sich die Konvergenzrate für alle gemessenen Fehlernormen der zu erwartenden h^2 bzw. h-Ordnung nähert, um dann wieder abzufallen. Tabelle 6.2 enthält die entsprechenden Zahlen für eintausend Datenpunkte. Hier ist nur am Anfang eine Näherung der gemessenen Konvergenzordnung an die erwartete zu beobachten, die dann wiederum abfällt.

Die Datenmenge enthält somit nicht genügend Information, um eine durchgehende Konvergenzrate zu ermöglichen, ab einer bestimmten Verfeinerungstiefe tritt eine Art Sättigung ein. Einen ähnlichen Zusammenhang zwischen Gitterauflösung und Datenauflösung haben wir in anderer Form in Abschnitt 3.2.5 bereits bei der Analyse des Fehlers von approximativen Wavelet-Kernen festgestellt.

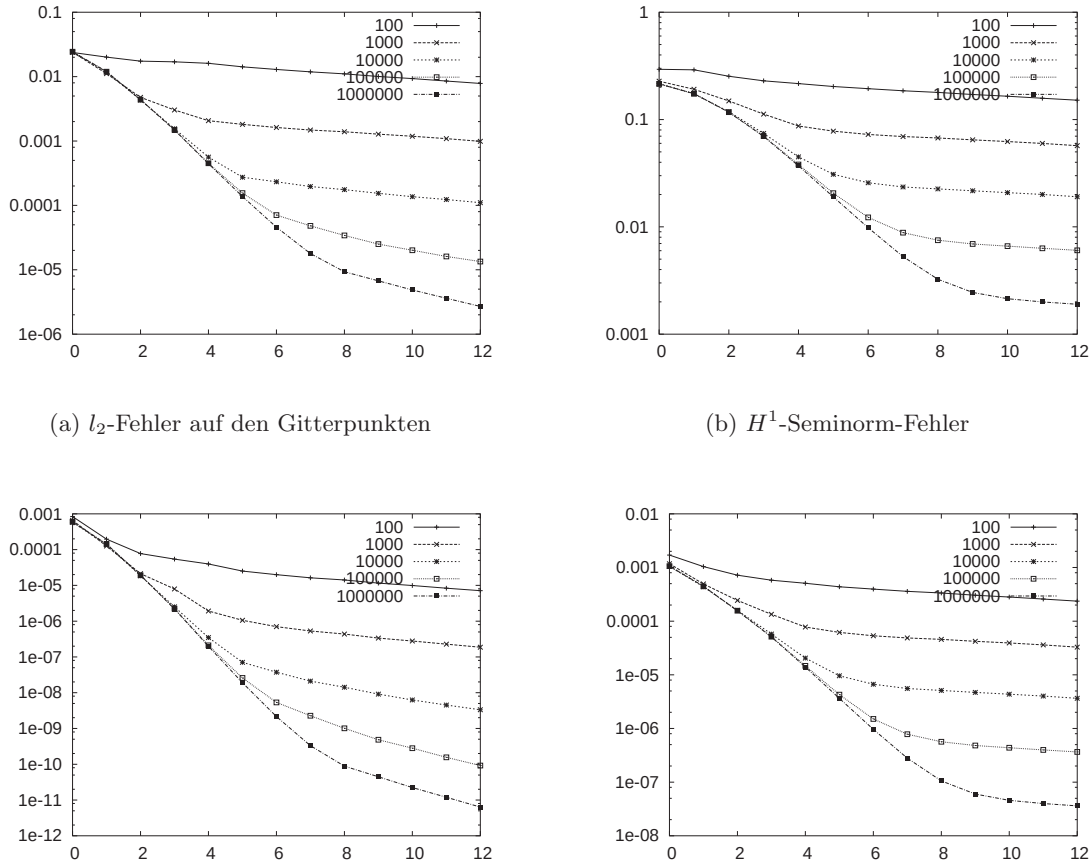
n	$\frac{\lvert\alpha_{n-1}-\alpha_{12}\rvert_{l_2}}{\lvert\alpha_n-\alpha_{12}\rvert_{l_2}}$	$\frac{\lvert f_{n-1}-f_{12}\rvert_{H^1}}{\lvert f_n-f_{12}\rvert_{H^1}}$	$\frac{\lvert f_{n-1}(\underline{x}_i)-f_{12}(\underline{x}_i)\rvert_{l_2}}{\lvert f_n(\underline{x}_i)-f_{12}(\underline{x}_i)\rvert_{l_2}}$	$\frac{R(f_{n-1}-f_{12})}{R(f_n-f_{12})}$
1	2.0350	1.2315	2.0366	2.3588
2	2.7533	1.4981	2.7543	2.8973
3	3.0895	1.6977	3.0887	3.1509
4	3.6798	1.8919	3.6680	3.6884
5	3.9237	1.9650	3.9054	3.8934
6	3.9776	1.9584	3.8864	3.8435
7	3.9322	1.8974	3.7052	3.6021
8	3.4283	1.7429	3.1889	3.0382
9	2.2500	1.5399	2.5191	2.3714
10	1.7022	1.4110	2.0301	1.9909
11	1.9755	1.5007	1.8283	2.2520

Tabelle 6.1. *Konvergenzverhalten auf vollen Gittern bei einer Million Datenpunkten für $\lambda = 0.01$. Dargestellt werden der l_2-Fehler auf den Gitterpunkten, der H^1-Seminorm-Fehler, der l_2-Datenfehler und der Fehler im Funktional R.*

n	$\frac{\lvert\alpha_{n-1}-\alpha_{12}\rvert_{l_2}}{\lvert\alpha_n-\alpha_{12}\rvert_{l_2}}$	$\frac{\lvert f_{n-1}-f_{12}\rvert_{H^1}}{\lvert f_n-f_{12}\rvert_{H^1}}$	$\frac{\lvert f_{n-1}(\underline{x}_i)-f_{12}(\underline{x}_i)\rvert_{l_2}}{\lvert f_n(\underline{x}_i)-f_{12}(\underline{x}_i)\rvert_{l_2}}$	$\frac{R(f_{n-1}-f_{12})}{R(f_n-f_{12})}$
1	2.1610	1.1922	2.2859	2.3950
2	2.4831	1.3068	2.5068	2.0989
3	1.9486	1.3505	1.9744	1.9108
4	1.4947	1.3424	2.1419	1.8492
5	1.1427	1.1567	1.4986	1.3472
6	1.1625	1.1099	1.2827	1.2351
7	1.2084	1.1116	1.2541	1.2377
8	1.2441	1.1166	1.1912	1.2477
9	1.3184	1.1517	1.2321	1.3273
10	1.5012	1.2260	1.2455	1.5033
11	1.9870	1.4213	1.4636	2.0205

Tabelle 6.2. *Konvergenzverhalten auf vollen Gittern bei eintausend Datenpunkten für $\lambda = 0.01$.*

In Tabelle 6.3 und Abbildung 6.2 ist das Fehlerverhalten der Dünngitterlösungen, berechnet mit der Kombinationstechnik, gegen die Lösung auf dem vollen Gitter des Levels 12 dargestellt. Das grundsätzliche Verhalten des Fehlerabfalls ist ähnlich wie auf vollen Gittern, d.h. sofern viele Datenpunkte vorhanden sind, steigt die Konvergenzordnung an, um dann in eine schlechtere Abnahmerate zu wechseln. Die Auflösung des dünnen Gitters, bei dem dieser Wechsel passiert, ist auch hier abhängig von der Anzahl der Datenpunkte. Allerdings ist das Konvergenzverhalten der Dünngitterlösungen bezüglich der Maschen-



(a) l_2-Fehler auf den Gitterpunkten

(b) H^1-Seminorm-Fehler

(c) l_2-Datenfehler

(d) Fehler des Funktionals R

Abbildung 6.2. *Konvergenzverhalten der Lösung auf dünnen Gittern im Vergleich zur Lösung des vollen Gitters der Stufe $n = 12$ für $\lambda = 0.01$. Dargestellt ist auf der horizontalen Achse die Diskretisierungsstufe n und auf der vertikalen der Fehler in logarithmischer Skala.*

weite h quantitativ schlechter als das der Vollgitterlösungen, wie es auch auf Grund der $h^2 \log(h)^{d-1}$-Konvergenz der dünnen Gitter zu erwarten ist.

Diese Untersuchung des Konvergenzverhaltens kann in höheren Dimensionen nicht fortgesetzt werden, da hochaufgelöste volle Gitter dann wegen ihres zu hohen Berechnungs- und Speicheraufwands nicht mehr betrachtet werden können. Die Erkenntnis, dass bei einer geringen Anzahl von Datenpunkten feinere Diskretisierungen keinen entscheidenden Vorteil liefern, kann aber mit Sicherheit auf höhere Dimensionen übertragen werden. Es kann weiterhin davon ausgegangen werden, dass für die Notwendigkeit einer feineren Diskretisierungsstufe in höheren Dimensionen mehr Datenpunkte vorliegen müssen als in niedrigen.

n	$\frac{\lvert\alpha_{n-1}-\alpha_{12}\rvert_{l_2}}{\lvert\alpha_n-\alpha_{12}\rvert_{l_2}}$	$\frac{\lvert f_{n-1}-f_{12}\rvert_{H^1}}{\lvert f_n-f_{12}\rvert_{H^1}}$	$\frac{\lvert f_{n-1}(\underline{x}_i)-f_{12}(\underline{x}_i)\rvert_{l_2}}{\lvert f_n(\underline{x}_i)-f_{12}(\underline{x}_i)\rvert_{l_2}}$	$\frac{R(f_{n-1}-f_{12})}{R(f_n-f_{12})}$
1	2.0338	1.2312	2.0355	2.3571
2	2.7340	1.4917	2.7347	2.8709
3	2.9637	1.6801	2.9632	3.0794
4	3.3284	1.8778	3.3235	3.6327
5	3.2361	1.9502	3.2308	3.8378
6	2.9948	1.9391	2.9679	3.7716
7	2.5672	1.8558	2.5497	3.4476
8	1.9192	1.6273	1.9356	2.6491
9	1.3804	1.3270	1.4075	1.7612
10	1.3892	1.1431	1.4125	1.3070
11	1.3433	1.0723	1.3693	1.1501
12	1.3391	1.0509	1.3703	1.1045

Tabelle 6.3. *Konvergenzverhalten der Lösungen auf dünnen Gittern im Vergleich zur Lösung auf dem vollen Gitter Ω_{12} bei einer Million Datenpunkte für $\lambda = 0.01$. Dargestellt werden der l_2-Fehler auf den Gitterpunkten, der H^1-Seminorm-Fehler, der l_2-Datenfehler und der Fehler im Funktional R.*

6.2 Messmethoden und Parameterbestimmung im maschinellen Lernen

Bei der Beurteilung von Klassifikationsverfahren ist das Grundmerkmal die *Erkennungsrate* oder *Genauigkeit*, in vielen Arbeiten auch in inverser Form als *Fehlerrate*. Die Erkennungsrate ist dabei die Anzahl der korrekt vorhergesagten Instanzen im Verhältnis zu der Zahl aller Datenpunkte. Die Fehlerrate ist dementsprechend die Anzahl der falsch vorhergesagten Datenpunkte im Verhältnis zur Gesamtzahl. Offensichtlich gilt, dass Erkennungsrate und Fehlerrate zusammen 100% der Datenmenge ergeben.

Beim maschinellen Lernen besteht grundsätzlich das Problem des so genannten *Overfitting*, des *Überanpassens* der Klassifikatorfunktion an die zum Lernen benutzten Daten. Es ist keine große Kunst, auf gegebenen Daten eine Erkennungsrate von 100% zu erreichen, im Extremfall wird für jeden Datenpunkt explizit der Wert vorgegeben. Das Ziel ist es vielmehr, auf neuen Daten gute Erkennungsraten zu erzielen. Um dieses Ziel zu erreichen und faire Vergleiche der Qualität der Lösung mit anderen maschinellen Lernverfahren zu ermöglichen, gibt es verschiedene Methoden.

Sofern sehr viele Daten vorhanden sind, bietet es sich an, den Datensatz aufzuteilen in eine Trainingsmenge, eine Testmenge und eine Evaluierungsmenge. Es sind auch andere Begriffe üblich wie Generalisierungsmenge für Evaluierungsmenge, oder in-sample für die Trainings- und Testmenge, und out-of-sample für die Evaluierungsmenge. Je nach Anwendungsbereich gibt es eine Konvention der dort üblichen Bezeichnungen.

Auf der Trainingsmenge wird das Lernverfahren mit verschiedenen Parameterkombinationen durchgeführt. Mittels der Testmenge wird der beste Parametersatz bestimmt,

welcher auch das beste Modell genannt wird. Die Vorhersagewerte des mit diesen Parametern erzielten Klassifikators auf der Evaluierungsmenge können dann als Ergebnisse auf neuen Daten aufgefasst werden. Diese Daten wurden im eigentlichen Lernprozess nicht benutzt und sind somit für den fairen Vergleich mit anderen Ergebnissen beziehungsweise für die korrekte Beschreibung der Qualität des Klassifikators geeignet.

Oft wird auch nur mit Trainings- und Evaluierungsmenge gearbeitet. Dann wird nur mit der Trainingsmenge durch geeignete Ansätze der beste Parametersatz bestimmt. Hierbei wird die Evaluierungsmenge in vielen Arbeiten auch Testmenge genannt. In diesen Fällen wird meist mit n-fach Kreuzvalidierung [Sto74] gearbeitet, gängig im maschinellen Lernen ist dabei die 10-fach Kreuzvalidierung. Hierzu wird die Datenmenge in n Teile zerlegt, jeweils $n-1$ Teilmengen werden zum Lernen benutzt, und auf der übrigen Teilmenge wird evaluiert. Dieses wird für jede Teilmenge vorgenommen. Nachdem alle n Teilmengen einmal Evaluierungsmenge gewesen sind, wird der Durchschnitt der Erkennungsraten auf den Trainingsmengen und den Evaluierungsmengen gebildet. Die Varianz der Erkennungsraten wird in diesem Fall oft mit angegeben. Ähnlich zur Kreuzvalidierung ist die Methode, n zufällige Aufteilungen in Trainingsmenge und Testmenge durchzuführen und über deren Erkennungsraten zu mitteln.

Neben der Möglichkeit, über eine experimentelle Kreuzvalidierung den besten Parametersatz zu bestimmen, gibt es den Ansatz, die generalisierte Kreuzvalidierung (GCV) zu berechnen [Wah90, HTF01] oder die Methode der L-Kurve [Han92, EHN96, Han98] zu nutzen, siehe [HTF01] für eine allgemeine Diskussion der Problematik der Modellbestimmung.

Bei einigen Benchmark-Datensätzen ist nur eine Datenmenge vorhanden, hierbei werden meist die Kreuzvalidierungsergebnisse angegeben, wobei die Parameter des Lernverfahrens auf den Teiltrainingsmengen mit geeigneten Methoden, oft wiederum der Kreuzvalidierung, bestimmt werden.

Wir werden uns im Folgenden auch an die im maschinellen Lernen übliche Konvention halten und den Begriff Evaluierungsmenge nur benutzen, wenn wir den Datensatz explizit in drei separate Teilmengen aufteilen.

Insbesondere für Datenmengen, bei denen die beiden Klassen nicht gleichhäufig auftreten, ist das Angeben der Erkennungsrate oft nicht ausreichend, um die Qualität des Klassifikators zu beurteilen. Gehören z.B. nur fünf Prozent der Daten zur Klasse A, wird mit der einfachen Regel, dass alle Dateninstanzen zur Klasse B gehören, eine Erkennungsrate von 95% erreicht, was aber keine sinnvolle Regel ist. Die *Verteilungsmatrix*, auch *Konfusionsmatrix* in direkter Übersetzung aus dem Englischen genannt, ist die gängigste Methode, sich einen Überblick über die genauere Klassenzuordnung zu verschaffen. Dabei werden in einer Tabelle

		reale Klasse	
		gut	schlecht
vorhergesagte Klasse:	gut	Anzahl (Anteil) Instanzen	Anzahl (Anteil) Instanzen
vorhergesagte Klasse:	schlecht	Anzahl (Anteil) Instanzen	Anzahl (Anteil) Instanzen

für jede Klasse die Anteile der richtig und falsch vorhergesagten Instanzen aufgetragen. In

der Hauptdiagonalen stehen offensichtlich die jeweils korrekt klassifizierten Daten, in den beiden anderen Zellen die falsch klassifizierten.

In Wirtschaftsanwendungen können den einzelnen Zellen der Verteilungsmatrix zusätzlich geeignet Kosten und Gewinne zugeordnet werden. In diesen Fällen wird dann nicht nach der besten Erkennungsrate gesucht, sondern nach dem größten Gewinn. Dargestellt wird das durch die *Kostenmatrix*

		reale Klasse	
		gut	schlecht
vorhergesagte Klasse:	gut	Gewinn/Kosten	Gewinn/Kosten
vorhergesagte Klasse:	schlecht	Gewinn/Kosten	Gewinn/Kosten

Auf der Hauptdiagonalen werden die im Allgemeinen positiven Zahlen für den Gewinn eingetragen, auf der Nebendiagonale die durch die falsche Klassifizierung verursachten Kosten. Zur Berechnung des potentiellen Gewinns werden diese Werte zellenweise mit den Einträgen der Verteilungsmatrix multipliziert, die Ergebnisse addiert und entweder als absoluter Wert angegeben oder aber durch die Gesamtzahl der Instanzen geteilt und somit als relative Größe präsentiert.

In Anwendungen gibt es für die Kostenmatrix zwei Anwendungsmöglichkeiten, zum einen kann die Parameterkombination gesucht werden, die den größten Gewinn erbringt. Bei Klassifikationsverfahren, die eine Sortierung der Datensätze liefern, kann zum anderen für jede Parameterkombination die Trennposition in den sortierten Daten bestimmt werden, die den größten Gewinn erzielt. Sofern nur eine Datenmenge vorliegt, wird auch hier mit Kreuzvalidierung ein durchschnittlicher Gewinn bestimmt.

Die Parameter unseres Verfahrens sind der Regularisierungsparameter λ und die Verfeinerungstiefe n. Wir geben in den Ergebnistabellen das mittels Trainings- und Testmenge, oder sofern nicht genügend Daten vorhanden sind per Kreuzvalidierung, bestimmte beste λ für jeden berechneten Level n an.

6.2.1 Graphische Darstellung der Ergebnisse

Da wir jedem Datenpunkt einen kontinuierlichen Wert zuordnen, der, wie im Einleitungskapitel erwähnt, der Wahrscheinlichkeit der Klassenzugehörigkeit zur gesuchten positiven Klasse entspricht, können die Datenpunkte in Reihenfolge ihres Wahrscheinlichkeitswertes absteigend sortiert werden. Somit ist es z.B. einfach möglich, eine Stichprobe S_p bestehend aus den 10% der Instanzen einer Datenmenge S auszuwählen, die das Verfahren als am wahrscheinlichsten zur gesuchten Klasse gehörend vorhersagt. Wichtig ist hierbei nur die Reihenfolge der Wahrscheinlichkeitswerte und nicht der tatsächliche numerische Wert. Mit anderen Worten, wir haben eine Durchnummerierung der Datenmenge, so dass

$$f(\underline{x}_i) \leq f(\underline{x}_{i+1}) \quad \forall i$$

gilt. Eine solche Sortierung kann nun zur graphischen Darstellung der Effizienz eines Lernverfahrens benutzt werden.

Da bei den Daten bekannt ist, ob die Vorhersage richtig oder falsch ist, kann somit bei einer Stichprobe S_p mit den vorhergesagten 10% Instanzen, die die wahrscheinlichsten positiven Instanzen sind, untersucht werden, wie viele davon richtig vorhersagt sind. Im Data Mining sind drei Methoden üblich, um diese Untersuchung graphisch darzustellen.

Es gibt so genannte *Steigerungsdiagramme*, oft werden auch die englischen Bezeichnungen *gains chart* oder *cumulative gains chart* benutzt. Gehen wir zum Beispiel von einer Werbebriefsendung an eine Million Haushalte aus. Wir nehmen an, dass die Erfahrung zeigt, dass 0,1% der Empfänger antworten, was in diesem Beispiel 1000 Kundenreaktionen entspricht. Nun wird mit einem Data-Mining-Werkzeug eine Teilmenge der Größe 100.000 des vorhandenen Datenbestandes auf Basis der bekannten Informationen über die Haushalte ausgewählt. Für diese wird eine Antwortrate von 0,4% ermittelt, d.h. 400 Antworten. Es könnte sich nun auszahlen nur an diese 100.000 Haushalte Werbung zu versenden, je nach Kosten der Werbesendung und der Einnahmen durch die Kundenreaktionen. Im Marketing wird bei einer solchen Erhöhung der Antwortrate, hier um dem Faktor 4, vom *Steigerungsfaktor* oder auch vom *Lift* gesprochen. Sofern die Kosten bekannt sind kann zusätzlich die Rendite einer Auswahl berechnet werden. So besteht bei Marketinguntersuchungen oft das Ziel, nicht einfach die wahrscheinlichsten Reagierer auszuwählen, sondern darüber hinaus die besonders gewinnträchtigen Reagierer unter den wahrscheinlichsten zu ermitteln.

Beim Steigerungsdiagramm wird nun auf der horizontalen Achse die Anzahl der ausgewählten vorhergesagten Instanzen, prozentual zur Anzahl der Daten,

$$\frac{|S_p|}{|S|} = \frac{|S_p|}{M}$$

aufgetragen. Auf der vertikalen Achse wird die Anzahl der korrekt vorhergesagten Instanzen der gesuchten Klasse, im Verhältnis zur Gesamtzahl der Instanzen dieser positiven Klasse,

$$\frac{|\{\underline{x}_i | \underline{x}_i \in S_p \text{ mit } f(\underline{x}_i) = 1 = y_i\}|}{|\{\underline{x}_i | \underline{x}_i \in S \text{ mit } y_i = 1\}|}$$

aufgetragen. Als Vergleich wird die diagonale Linie eingezeichnet, was dem Ergebnis für eine zufällig ausgewählte Stichprobe entspricht. Weiterhin wird die optimale Steigerungskurve aufgetragen, hierfür wird davon ausgegangen, dass alle Instanzen der ausgewählten Stichprobe zur gesuchten positiven Klasse gehören. Die Ergebniskurve kann sich somit nur zwischen diesen beiden befinden, siehe Abbildung 6.4(a) für ein Beispiel. Ziel ist es offensichtlich, mit der Vorhersage eine Kurve möglichst nah an der optimalen Kurve zu erreichen, was einer möglichst großen Fläche unter der Kurve entspricht. Die Mindestvoraussetzung an ein Data Mining-Verfahren ist, dass die Kurve sich oberhalb der Diagonalen befindet.

Im so genannten *Lift-Chart* werden die bereits erwähnten Steigerungsfaktoren im Vergleich zu einer zufälligen Teilmengenauswahl über die Größe der Stichprobe angetragen, d.h. die vertikale Achse hat nun die Werte

$$\frac{|\{\underline{x}_i | \underline{x}_i \in S_p \text{ mit } f(\underline{x}_i) = 1 = y_i\}|}{\frac{|S_p|}{|S|} \cdot |\{\underline{x}_i | \underline{x}_i \in S \text{ mit } y_i = 1\}|}.$$

Mit anderen Worten, die vertikale Achse gibt für die Menge S_p das Verhältnis zwischen der Zahl der durch das Verfahren korrekt vorhergesagten positiven Instanzen und der zu erwartenden Zahl positiver Instanzen bei zufälliger Auswahl einer Menge der Größe $|S_p|$ wieder. Siehe Abbildung 6.4(c) für ein Beispiel.

Eine mit Steigerungsdiagrammen verwandte graphische Darstellung der Güte einer Vorhersage ist die so genannte *ROC-Kurve*, wobei die Abkürzung für „Receiver Operating Characteristic“ steht. Dieser Begriff kommt aus der Signalerkennung und beschreibt die Abwägung zwischen Trefferrate und Fehleralarmrate über einem verrauschten Kanal. Auf der vertikalen Achse wird wiederum die Anzahl der korrekt vorhergesagten Instanzen der gesuchten Klasse im Verhältnis zur Gesamtzahl der Instanzen dieser Klasse aufgetragen. Auf der horizontalen Achse wird nun allerdings die Anzahl der vorhergesagten Instanzen der nicht gesuchten Klasse im Verhältnis zur Gesamtzahl der Instanzen dieser negativen Klasse aufgetragen, d.h.

$$\frac{|\{\underline{x}_i | \underline{x}_i \in S_p \text{ mit } f(\underline{x}_i) = -1 = y_i\}|}{|\{\underline{x}_i | \underline{x}_i \in S \text{ mit } y_i = -1\}|}.$$

Für diese Darstellung ist nun das vollständige obere Dreieck der mögliche Bereich für die Ergebniskurve. Ziel ist wiederum, eine möglichst große Fläche unter der Kurve zu erhalten, d.h. eine Kurve weit links oben zu erhalten, siehe Abbildung 6.4(b). Die ROC-Kurve kann auch als eine Transformation des Steigerungsdiagramms auf das ganze obere Dreieck aufgefasst werden.

6.3 Zweidimensionale synthetische Klassifizierungsprobleme

6.3.1 Ripley-Datensatz

Der *Ripley*-Datensatz stammt aus [Rip94] und besteht aus 250 Trainingsdaten und 1000 Testdaten. Die Datenmenge wurde synthetisch generiert, wobei die beiden Klassen gleichverteilt sind, und enthält 8% Fehler. Somit können keine besseren Korrektheitsraten als 92% erwartet werden.

Da Training- und Testdaten vorliegen, verfahren wir wie folgt: Zuerst benutzen wir die Trainingsmenge, um den besten Regularisierungsparameter λ mittels zehnfacher Kreuzvalidierung für jeden Level n zu bestimmen. Für verschiedene Level n geben wir in der zweiten Spalte der Tabelle 6.4 die jeweils größte erreichte Erkennungsrate mit dazugehöriger Standardabweichung auf den Testmengen an. Die dritte Spalte enthält das dazugehörige λ. Mit diesem berechnen wir dann den Dünngitter-Klassifikator bezüglich aller 250 Trainingsdaten und wenden ihn auf die Testmenge an. Diese Erkennungsrate geben wir in Spalte vier der Tabelle 6.4 an.

Auf Basis der Kreuzvalidierung auf der Trainingsmenge wird Level 4 als Klassifikator ausgewählt, für Level 10 liefert die Kreuzvalidierung ebenso eine durchschnittliche Erkennungsrate von 88,4%, im Zweifelsfall wird im maschinellen Lernen allerdings das einfachere Modell gewählt, nachdem unter dem Namen *Ockhams Rasiermesser* bekannten Grundprinzip der Sparsamkeit bezwiehungsweise Einfachheit in der wissenschaftlichen Methodik. Weiterhin ist auch die Standardabweichung für Level 4 etwas geringer als für Level 10, das Ergebnis somit stabiler. Allerdings liefert der so bestimmte Parametersatz

	10-fache Kreuzvalidierung			Beste Ergebnisse	
Level	Kreuzvalidierung %	λ	Testdaten %	λ	Testdaten %
0	87.3 ± 6.2	0.01	89.4	0.015	89.5
1	85.7 ± 7.2	0.01	89.6	0.015	89.7
2	85.7 ± 5.7	0.02	90.1	0.00075	90.8
3	88.0 ± 6.8	0.0006	90.7	0.0055	91.1
4	88.4 ± 6.1	0.001	90.2	0.0055	91.1
5	87.6 ± 6.6	0.0055	90.9	0.125	91.1
6	86.8 ± 6.3	0.003	91.0	0.007	91.2
7	86.8 ± 5.6	0.0009	88.4	0.007	90.9
8	87.2 ± 5.5	0.0006	88.2	0.0055	91.0
9	88.0 ± 6.6	0.001	89.4	0.007	91.0
10	88.4 ± 6.9	0.0025	90.8	0.007	91.0

Tabelle 6.4. *Ergebnisse für den Ripley-Datensatz [Rip94] mit der Kombinationstechnik auf Basis des Tensorproduktansatzes. Angegeben sind die Diskretisierungstiefe, die mittels zehnfacher Kreuzvalidierung bestimmte beste Erkennungsrate mit Standardabweichung auf den Trainingsdaten, das dazugehörige λ und die mit diesem λ erzielte Rate auf den Testdaten. Die bestmöglichen Ergebnisse für jeden Level sind in den letzten beiden Spalten dargestellt.*

bestehend aus $\lambda = 0.001$ und Level $n = 4$ mit 90,2% nicht das optimale Ergebnis. Dies zeigt sowohl der Vergleich mit den Ergebnisse der anderen λ-Level Kombinationen auf den Testdaten, hier erzielt Level 6 mit 91% leicht bessere Ergebnisse, als auch der Vergleich mit den bestmöglichen Ergebnissen. Diese geben wir in Tabelle 6.4 in den letzten beiden Spalten an, um zum einen die grundsätzlichen Möglichkeiten des Dünngitteransatzes und zum anderen die Auswahlqualität des Kreuzvalidierungsansatzes zu untersuchen. Dazu berechnen wir für eine große Auswahl von Werten für λ den Dünngitter-Klassifikator aus den 250 Datenpunkten und evaluieren sie. Wir wählen dann das bestmögliche Ergebnis für jeden Level aus. Dieses wird bei Level 6 mit 91,4% erreicht.

Es ist weiterhin zu beobachten, dass bei diesem Datensatz für jede einzelne Diskretisierungstiefe meist kein deutlicher Unterschied zwischen den Erkennungsraten zum bestmöglichen und zu dem über Kreuzvalidierung bestimmten λ besteht. Bei höheren Diskretisierungsstufen, bei denen Überanpassungs-Effekte stärker auftreten können, werden die Unterschiede allerdings größer. Grundsätzlich funktioniert aber der Zugang für die Bestimmung des Wertes von λ aus der Trainingsmenge mittels Kreuzvalidierung in geeigneter Art und Weise.

Es sei erwähnt, dass mit Neuronalen Netzen 90.6% [Rip94] beziehungsweise 91.1% [PR99] Erkennungsraten auf der Testmenge berichtet werden, unsere Ergebnisse sind somit von vergleichbarer Güte.

In Tabelle 6.5 geben wir entsprechende Ergebnisse bei der Nutzung von linearen Ansatzfunktionen auf Basis einer simplizialen Diskretisierung an. Mittels Kreuzvalidierung wird hier Level 6 ausgewählt, in diesem Fall treffen wir mit dieser Wahl sogar die bestmögli-

	10-fache Kreuzvalidierung			Beste Ergebnisse	
Level	Kreuzvalidierung %	λ	Testdaten %	λ	Testdaten %
0	85.7 ± 6.5	0.013	89.9	0.0006	90.5
1	84.0 ± 7.0	0.00275	88.9	0.4	90.1
2	87.2 ± 7.8	0.0015	90.8	0.003	91.2
3	84.9 ± 5.8	0.125	90.1	0.00015	91.3
4	88.4 ± 6.6	0.006	90.4	0.0045	90.6
5	87.7 ± 7.1	0.003	91.2	0.003	91.2
6	88.5 ± 6.6	0.0035	91.4	0.0035	91.4
7	88.4 ± 2.7	0.000125	85.5	0.08	90.8
8	87.6 ± 4.8	0.00075	88.9	0.0045	91.1
9	88.0 ± 3.9	0.0004	87.9	0.01	91.2
10	88.4 ± 6.5	0.001	89.6	0.007	91.1

Tabelle 6.5. *Ergebnisse für den Ripley-Datensatz [Rip94] mit der Kombinationstechnik auf Basis der simplizialen Diskretisierung. Angegeben sind die Diskretisierungstiefe, die mittels zehnfacher Kreuzvalidierung bestimmte beste Erkennungsrate mit Standardabweichung auf den Trainingsdaten, das dazugehörige λ und die mit diesem λ erzielte Rate auf den Testdaten. Die bestmöglichen Ergebnisse für jeden Level sind in den letzten beiden Spalten dargestellt.*

che Wahl für diesen Datensatz und erreichen eine Erkennungsrate von 91.4% auf den Testdaten. Weiterhin ist hier wiederum zu beobachten, dass bei niedrigen Verfeinerungsstufen die erzielten Erkennungsraten näher bei den für den jeweiligen Level möglichen besten Ergebnissen liegen als bei den feineren Auflösungen. Das spricht wiederum dafür, in der Tendenz niedrige Auflösungen und somit einfachere Modelle zu bevorzugen.

Wir sehen bei beiden Diskretisierungsvarianten, dass es nicht notwendig ist, höhere Level zu verwenden. Grund dafür ist sicherlich die relative Einfachheit der Daten, siehe Abbildung 6.3. Einige wenige Hyperebenen sollten genug sein, um die Klassen relativ sauber zu trennen. Dies lässt sich mit dünnen Gittern bereits mit einem niedrigen Level n erzielen. In den Bildern lassen sich auch Überanpassungs-Effekte beobachten. So sind im oberen rechten Viertel dunkelgraue Punkte zu finden, die beim niedrigen Level der hellgrauen Menge zugeordnet werden, beim höheren Level allerdings der dunkelgrauen Menge. Wie erwähnt sind in dieser synthetischen Datenmenge Fehler bezüglich der Trennfläche eingebaut, bei einem höheren Level werden diese eher aufgelöst und nicht mehr durch Punkte in der Nachbarschaft weggeglättet. Bei realen Daten ist es im Allgemeinen nicht klar, ob das nun ein erwünschter Effekt ist, d.h. in diesem Fall die dunkelgrauen Insel besondere Datenpunkte sind, oder ein unerwünschter Effekt, d.h. diese Instanzen fehlerbehaftete Daten sind.

Weiterhin sind in Abbildung 6.4 das Steigerungsdiagramm, der ROC-Chart und der Lift-Chart für die erzielten Ergebnisse dargestellt. Wie zu sehen ist, unterscheiden sich die Kurven für beide Diskretisierungsvarianten kaum, wobei die Kurven für den Klassifikator, der mit der simplizialen Variante berechnet wird, leicht bessere Eigenschaften aufweisen.

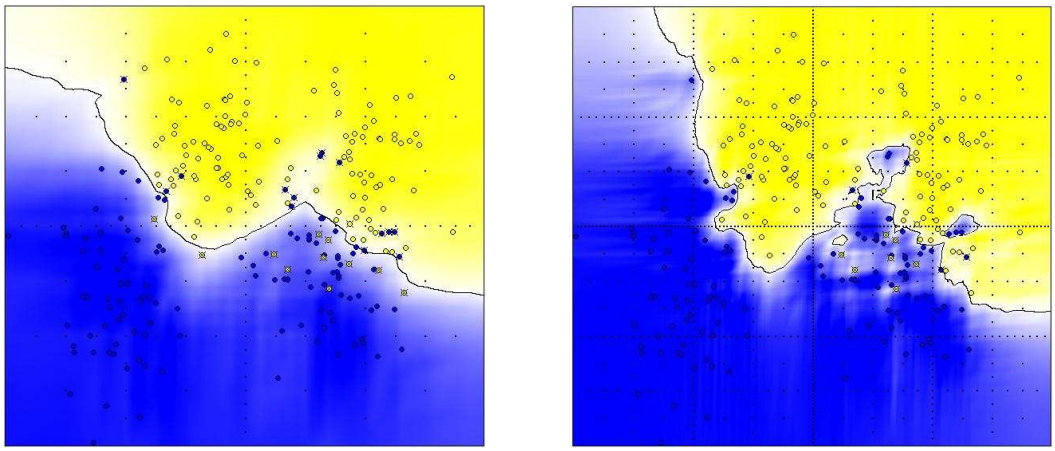


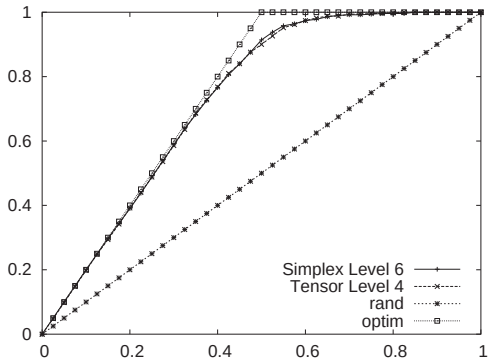
Abbildung 6.3. *Bilder zum Ripley-Datensatz. Kombinationstechnik mit simplizialer Diskretisierung für Level 6, $\lambda = 0.0035$ (links) und Level 8, $\lambda = 0.00075$ (rechts), siehe auch Tabelle 6.5. Die Klassenzugehörigkeit, d.h. $f(\underline{x}) = \pm 1$ ist farbkodiert, weiterhin wird die Trennlinie für $f(\underline{x}) = 0$ dargestellt. Je heller die Farbe desto näher ist der kontinuierliche Wert des Klassifikators bei 0.*

Über 80% der gesuchten Instanzen sind in den ersten 42,5% der sortierten Daten enthalten, was im Steigerungsdiagramm 6.4(a) dargestellt ist. Dementsprechend ist dem ROC-Chart 6.4(b) zu entnehmen, dass in dem Abschnitt der Sortierung, der 80% der gesuchten Instanzen enthält, weniger als 5% der falschen Instanzen vorkommen. Im Lift-Chart 6.4(c) wiederum ist zu beobachten, dass bei der simplizialen Variante für die ersten zehn Prozent der sortierten Daten ein perfekter Lift von zwei erreicht wird.

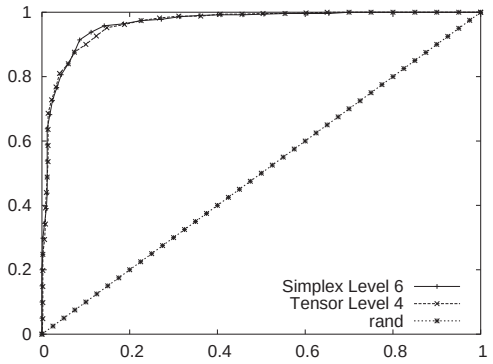
6.3.2 Schachbrett-Datensatz

Der *Schachbrett*-Datensatz ist aus [Kau99] übernommen. Hier sind 1000 Trainings-Datenpunkte gegeben, die mehr oder weniger gleichförmig aber zufällig verteilt sind. Die zugehörigen Datenwerte sind abhängig von der Position, so dass eine 4×4 Schachbrettstruktur entsteht, siehe Abbildung 6.5 (links). Hierbei gehören 514 Punkte zur einen und 486 Datenpunkte zur anderen Klasse, die Verteilung ist somit nicht gleichmäßig für beide Klassen. Wir berechnen die Trainings- und Testkorrektheit für die zehnfache Kreuzvalidierung mit der Dünngitter-Kombinationstechnik für verschiedene Werte des Regularisierungsparameters λ und für verschiedene Level n, wobei wir für diesen Datensatz hier nur die Ergebnisse für d-lineare Basisfunktionen angegeben. Die Erkennungsraten sind in Abbildung 6.5 (rechts) gezeigt.

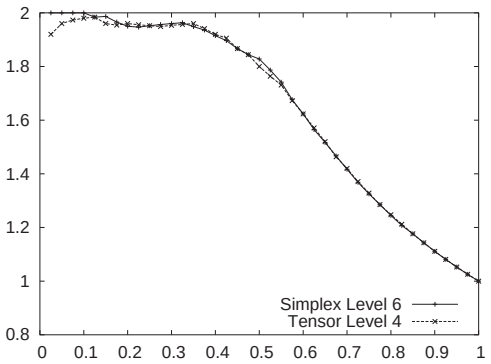
Wir sehen, dass die zehnfache Testgenauigkeit für Werte von λ zwischen $3 \cdot 10^{-5}$ und $5 \cdot 10^{-3}$ um 95% liegt. Für Regularisierungsparameter außerhalb dieses Bereichs fällt die Erkennungsrate deutlich ab, für kleine λ auf Grund von Überanpassungs-Effekten, bei großen Werten von λ wird zu stark geglättet. Die beste erreichte zehnfache Testkorrektheit liegt bei 96.2% mit einer Verfeinerungstiefe 10 und $\lambda = 4.5 \cdot 10^{-5}$.



(a) Steigerungsdiagramm



(b) ROC-Chart



(c) Lift-Chart

Abbildung 6.4. *Steigerungsdiagramm, ROC-Chart und Lift-Chart für den Ripley-Datensatz, Kombinationstechnik mit Tensorproduktansatz für Level* $n = 4$ *und* $\lambda = 0.,001$ *sowie mit simplizialer Diskretisierung für Level* $n = 6$ *und* $\lambda = 0.0035$.

Gemessen auf einem Testdatensatz, der die Schachbrettstruktur auf einem regulären Gitter der Größe 200×200 wiedergibt, wird eine Erkennungsrate von 95.2% erreicht. Eine perfekte Erkennung der Schachbrettstruktur ist auf Basis des gegebenen Testdatensatzes nicht möglich, da zum einen die beiden Klassen unterschiedlich viele Instanzen aufweisen, zum anderen Bereiche gerade an den jeweiligen Grenzen der Muster nicht genügend Punkte aufweisen, um ohne weitere äußere Annahmen die regelmäßige Struktur vollständig wiederzugeben. Siehe auch Abbildung 6.5 (links).

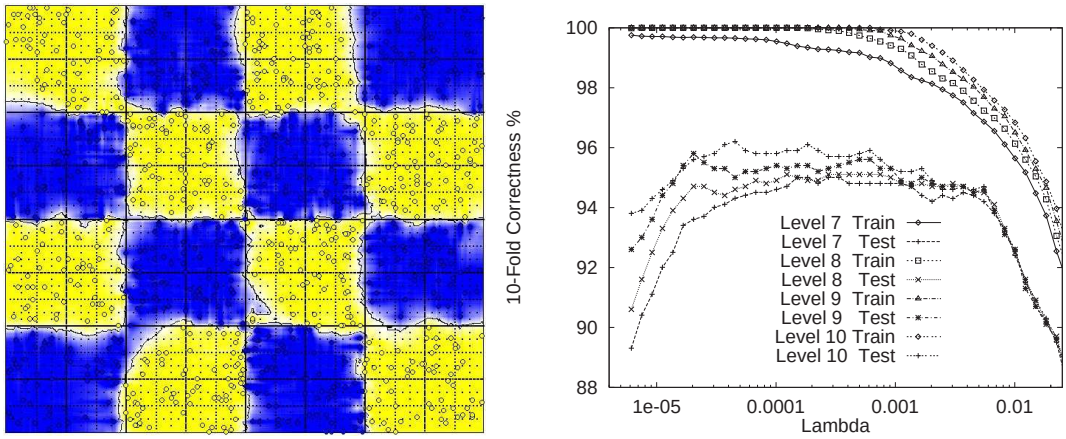


Abbildung 6.5. *Checkerboard-Datensatz [Kau99]. Links dargestellt ist der Klassifikator der Kombinationstechnik mit Level $n = 10$ und $\lambda = 4.5e\text{-}05$. Rechts ist ein Graph zur Abhängigkeit von λ (in logarithmischer Skalierung) und Level n wiedergegeben.*

6.4 Höherdimensionale synthetische Datensätze

Die im folgenden Abschnitt gemessenen Rechenzeiten sind bei der Nutzung eines Intel Xeon 3GHz-Prozessors gemessen worden.

6.4.1 Sehr große synthetische Datenmenge in sechs Dimensionen

Um das Skalierungsverhalten des Verfahrens bei großen Datenmengen zu untersuchen, erzeugen wir mit DatGen [Mel99] eine sechsdimensionale Datenmenge mit fünf Millionen Trainingspunkten und 20000 Testpunkten. Dazu verwenden wir den Aufruf `datgen -r1 -X0/100,R,0:0/100,R,0:0/100,R,0:0/100,R,0:0/200,R,0:0/200,R,0 -R2 -C2/4 -p -D2/5 -T10/60 -O502 0000 -e0.15`. Da wir bei diesem Beispiel hauptsächlich das Laufzeitverhalten untersuchen wollen, sind in Tabelle 6.6 nur die Ergebnisse für $\lambda = 0.01$ dargestellt. Es sei bemerkt, dass bereits auf Level 1 eine Testgenauigkeit von über 90% erzielt wird. Neben der Gesamtlaufzeit geben wir die CPU-Zeit für das Aufstellen der Datenmatrizen $B^t \cdot B$ und die Anzahl der Iterationen an, die das cg-Verfahren zur Lösung des Gleichungssystems benötigt.

Wir sehen, dass auch große Datenmengen mit fünf Millionen Punkten verarbeitet werden können. Die Ausführungszeiten skalieren linear in der Zahl der Datenpunkte. An den Zeiten, die für die Berechnung der Matrizen $B^t \cdot B$ benötigt werden, ist zu sehen, dass bei Verwendung von d-linearen Basisfunktionen mehr als 97% der Rechenzeit für das Aufstellen der Datenmatrizen benötigt wird. Die Bestimmung der Lösung auf Level 2 benötigt annähernd vier Stunden, eine für die Praxis nicht akzeptable Zeit, welche die Nutzung von schnelleren Methoden nahelegt.

Wie in Abschnitt 5.2 dargestellt ist die Verwendung von linearen Basisfunktionen auf Basis einer simplizialen Diskretisierung solch eine schnellere Methode. Bei diesem Beispiel benötigt das Verfahren mit linearen Basisfunktion für Level 2 ungefähr fünf Minuten und

Level	Punkte	%-Trainings-genauigkeit	%-Test-genauigkeit	Gesamtzeit (Sek.)	Datenmatrix Zeit (Sek.)	cg-Iterationen
			lineare Basisfunktionen			
1	$5 \cdot 10^4$	90.4	90.5	0.4	0.2	34
	$5 \cdot 10^5$	90.5	90.5	3.9	2.4	37
	$5 \cdot 10^6$	90.5	90.6	39.9	25.1	39
2	$5 \cdot 10^4$	91.4	91.0	3.4	2.0	274
	$5 \cdot 10^5$	91.2	91.1	30.8	19.8	295
	$5 \cdot 10^6$	91.1	91.1	301.8	195.8	313
3	$5 \cdot 10^4$	92.2	91.4	15.1	8.8	1302
	$5 \cdot 10^5$	91.7	91.7	132.3	88.0	1401
	$5 \cdot 10^6$	91.6	91.6	1303.2	879.4	1486
			d-lineare Basisfunktionen			
1	$5 \cdot 10^4$	90.8	90.8	18.6	18.2	127
	$5 \cdot 10^5$	90.7	90.8	179.3	177.2	138
	$5 \cdot 10^6$	90.7	90.7	1800.2	1781.0	147
2	$5 \cdot 10^4$	91.9	91.5	134.9	131.6	934
	$5 \cdot 10^5$	91.5	91.6	1306.1	1290.8	999
	$5 \cdot 10^6$	91.4	91.4	13299.0	13164.3	1064

Tabelle 6.6. *Ergebnisse für einen synthetischen, großen, sechsdimensionalen Datensatz mit* $\lambda = 0.01$.

ist somit um den Faktor 40 schneller als die Verwendung von d-linearen Ansatzfunktionen, wobei geringfügig schlechtere Genauigkeitsraten erzielt werden. Bessere Erkennungsraten mit linearen Funktionen liefert Level 3, wobei diese Rechnung für ein feineres dünnes Gitter immer noch 10-mal schneller ist als Level 2 der d-linearen Variante. Bei der Nutzung der simplizialen Diskretisierung wird bis zu 65% der Gesamtrechenzeit für die Aufstellung der Datenmatrix benötigt. Wiederum skaliert die Laufzeit linear mit der Anzahl der Datenpunkte.

In höheren Dimensionen ist der Faktor des Rechenzeitgewinns der linearen im Vergleich zur d-linearen Diskretisierungsvariante deutlich größer.

6.4.2 ndcHard-Datensatz in zehn Dimensionen

Der zehndimensionale, synthetisch generierte Datensatz *ndcHard* mit zwei Millionen Instanzen stammt aus [MM01]. In [FM01] wird er als ndcHard bezeichnet, da er nur schwer linear separierbar ist, im Gegensatz zu einem zweiten mit ndc [Mus98] generierten Datensatz namens *ndcEasy*.

Wie im vorherigen Beispiel 6.4.1 ist das Rechenzeitverhalten ein wichtiger Gesichtspunkt. Wiederum geben wir in Tabelle 6.7 neben der Gesamtlaufzeit die Rechenzeit für das Aufstellen der Datenmatrix $B^t \cdot B$ und die Anzahl der Iterationen an, die das cg-Verfahren zur Lösung des Gleichungssystems benötigt. Mehr als 50% der Rechenzeit wird

Level	Punkte	%-Trainings-genauigkeit	%-Test-genauigkeit	Gesamtzeit (Sek.)	Datenmatrix Zeit (Sek.)	cg-Iterationen
1	$2 \cdot 10^4$	86.2	84.2	6.3	0.9	45
	$2 \cdot 10^5$	85.1	84.8	16.2	8.7	51
	$2 \cdot 10^6$	84.9	84.9	114.9	84.9	53
2	$2 \cdot 10^4$	85.1	83.8	134.6	10.3	566
	$2 \cdot 10^5$	84.5	84.2	252.3	98.2	625
	$2 \cdot 10^6$	84.3	84.2	1332.2	966.6	668

Tabelle 6.7. *Ergebnisse für den ndcHard Datensatz.*

für die Aufstellung der Datenmatrix benötigt, dieser Teil skaliert linear mit der Anzahl der Datenpunkte. Die Gesamtzeit skaliert sogar etwas besser als linear, da der Aufwand für die Lösung des linearen Gleichungssystems nicht von der Größe der Datenmenge abhängt. Wir benutzen hier Gitter mit mindestens drei Punkten je Raumdimension, d.h. die Kombinationstechnik nach Formel (4.17). Der Speicherbedarf beträgt bei Level 2 ungefähr 290 MByte, circa 120 MByte für die Matrix und ungefähr 170 MByte, um die Datenpunkte im Speicher zu behalten, wobei die Datenpunkte nach ihrer Verarbeitung in die Datenmatrix nicht mehr unbedingt im Speicher benötigt werden und dieser Speicher wieder freigegeben werden könnte.

Bereits mit dem groben Gitter mit drei Punkten in jeder Dimension erreichen wir eine Testerkennungsrate von 84.9%, mit höheren Leveln wird keine Verbesserung gemessen. Mit linearen Support Vektor Maschinen (SVM) wird in [FM01, MM01] eine Erkennungsrate von 69.5% berichtet. Nichtlineare Support Vektor Maschinen können diesen großen Datensatz aus Komplexitätsgründen nicht bearbeiten. Es sei bemerkt, dass wir für den einfacher linear separierbaren ndcEasy Datensatz eine Erkennungsrate von über 98% erreichen, im Vergleich zu 91% in [FM01] für lineare Support Vektor Maschinen.

6.5 Höherdimensionale reale Datensätze

Im Folgenden präsentieren wir Ergebnisse für eine Reihe von Benchmark-Datenmengen, die auf realen Erhebungen basieren. Wir haben die Daten jeweils auf $[0,1]^d$ transformiert. Dabei haben wir keine besondere Rücksicht auf Ausreißer genommen, d.h. einzelne Datenpunkte, deren Koordinaten sich deutlich von den anderen Datenpunkten unterschieden. Wir haben aber unterschiedliche Transformationen durchgeführt. Die einfachste ist dabei die lineare Transformation des Intervalls $[\min, \max]$ auf $[0,1]$ durch die Vorschrift $\hat{x} = (x - \min)/(\max - \min)$. Eine weitere Methode besteht in der so genannten Z-Normierung, hier wird die Transformationsvorschrift $\hat{x} = (x - m)/s$ genutzt, wobei m der Mittelwert des Attributs und s die Standardabweichung ist. Das normierte Attribut hat nun Mittelwert 0 und Standardabweichung 1. Für unsere Zwecke muss dieser Transformation allerdings noch eine Normierung auf $[0,1]$ folgen. Als dritte Methode wird von uns die Transformation mit Hilfe des Median $x_{0.5}$, d.h. des mittleren Wert des Attributs, genutzt. Mittlerer Wert bedeutet hier, dass die eine Hälfte der Daten kleinere Werte als

		d-linear		linear		Dim. 3 verfeinert	
Level	10-fold	Rate %	Sek.	Rate %	Sek.	Rate	Sek.
1	train	77.7 ± 1.3	1.0	82.3 ± 1.1	0.1		
	test	71.8 ± 6.6		72.4 ± 8.3			
2	train	84.3 ± 1.0	12.0	80.0 ± 1.0	0.6	78.7 ± 1.1	0.3
	test	70.4 ± 8.6		72.5 ± 6.1		73.9 ± 6.9	
3	train	91.4 ± 0.8	89.1	86.6 ± 1.2	8.0	79.8 ± 1.2	1.2
	test	70.8 ± 7.7		69.9 ± 7.1		73.0 ± 7.0	
4	train	92.6 ± 0.9	546.3	94.2 ± 0.7	54.1	86.0 ± 1.5	3.9
	test	68.8 ± 6.5		71.4 ± 7.8		72.4 ± 8.0	

Tabelle 6.8. *Ergebnisse für den BUPA Lebererkrankung-Datensatz. Dargestellt sind die Erkennungsraten und die benötigte Rechenzeit für den d-linearen und linearen Ansatz. Weiterhin werden die Ergebnisse bei einem anisotropen Gitter mit zusätzlicher Verfeinerung von Dimension 3 angegeben.*

$x_{0.5}$ aufweist und die andere Hälfte größere Werte. Wir transformieren dann das Intervall $[\min, x_{0.5}]$ auf $[0, 0.5]$ und das Intervall $[x_{0.5}, \max]$ auf $[0.5, 1]$. Diese Transformation ist robuster gegen Ausreißer [Han02]. Wir gehen im Weiteren aus Gründen der Kompaktheit der Darstellung nicht darauf ein, welche Transformation wir für einen Datensatz jeweils genutzt haben.

Es gibt eine Reihe von Arbeiten, die sich unter den Namen *Robuste Statistik* weitergehend mit der Thematik von Ausreißern und geeigneten robusten Transformationsmethoden beschäftigen, siehe z.B. [RL87]. Allgemeine Informationen zur Datenvorverarbeitung finden sich [CPS98, HTF01, WF01].

Die im Folgenden angegebenen Rechenzeiten sind wiederum auf einem Rechner mit Intel Xeon 3GHz-Prozessor gemessen worden. Wir verwenden, sofern nicht anders angegeben, die simpliziale Diskretisierung auf den Gittern der Kombinationstechnik.

6.5.1 BUPA Lebererkrankung-Datensatz

Die BUPA Lebererkrankungs-Datenmenge aus der UC Irvine Machine Learning Datenbank [BM98] besteht aus 345 Datenpunkten mit sechs Merkmalen. Fünf Attribute sind Bluttests, von denen angenommen wird, dass sie sensitiv auf Lebererkrankungen reagieren, welche durch exzessiven Alkoholkonsum verursacht werden. Das sechste Attribut gibt die Menge des täglich getrunkenen Alkohols an, normiert auf die entsprechende Anzahl von halben Pints. Jede Instanz in dieser Datenmenge repräsentiert die Werte eines ledigen Mannes. Das Selektorfeld trennt den Datensatz in zwei Mengen mit je 145 Punkten und 200 Punkten, eine besteht aus den Personen mit einer Lebererkrankung, die andere Menge enthält die Personen ohne festgestellte Lebererkrankung.

Hier liegen uns nur Trainingsdaten vor und wir können deswegen in Tabelle 6.8 nur über unsere zehnfachen Kreuzvalidierungsergebnisse berichten. Der Ansatz mit linearen Basisfunktionen erreicht hier auf Level 2 mit 72.5% Testgenauigkeit leicht bessere Ergeb-

				Dim 5 verfeinert	
Level	10-fold	Rate %	Zeit (Sek.)	Rate %	Zeit (Sek.)
0	train	97.1 (± 0.1)	6.2		
	test	95.6 (± 1.0)			
1	train	93.1 (± 0.2)	108.1	97.7 (± 0.1)	10.4
	test	91.7 (± 1.8)		95.9 (± 1.0)	
2	train			97.8 (± 0.1)	17.9
	test			96.1 (± 0.9)	

Tabelle 6.9. *Ergebnisse für den Galaxy Dim-Datensatz für die normale Kombinationstechnik und für anisotrope Gitter mit zusätzlicher Verfeinerung in Dimension 5.*

nisse als der mit d-linearen Funktionen, die als bestes Ergebnis 71.8% erreichen. Bei diesem kleinen Datensatz macht sich die Berechnung der Datenmatrix noch nicht so deutlich in der Laufzeit bemerkbar, die Kombinationstechnik mit der simplizialen Diskretisierung ist trotzdem um den Faktor 10 schneller. Unser Verfahren liefert etwas bessere Resultate als eine lineare SVM, die 70.0% erreicht, aber etwas schlechtere Resultate im Vergleich zu einer nichtlinearen SVM, welche 73.7% erzielt, siehe [FM01].

Desweiteren unternehmen wir einfache Experimente mit anisotropen dünnen Gittern, um festzustellen, ob die Verfeinerung einzelner Dimensionen von Vorteil ist. Durch Kreuzvalidierung haben wir die dritte Dimension als geeignetste bestimmt. Verfeinern wir diese einmal, d.h. nutzen wir das anisotrope Gitter $\Omega^s_{1,1,2,1,1,1}$, erhalten wir die bisher beste Erkennungsrate von 73.9%. Weitere Verfeinerungen dieser Dimension resultieren nicht in besseren Ergebnissen.

6.5.2 Galaxy Dim-Datensatz

Der Galaxy Dim-Datensatz ist eine oft benutzte Teilmenge der in [OSP+92] präsentierten Daten. Er enthält 4192 Instanzen mit 14 Attributen, die beiden Klassen treten annähernd gleich häufig auf. Die Attribute sind Metainformationen von astronomischen Aufnahmen. Das Ziel besteht darin, auf deren Basis die Bilder in Aufnahmen von Sternen und Galaxien zu klassifizieren. Da in 14 Dimensionen gerechnet wird, kann nur die simpliziale Diskretisierung mit der Kombinationsformel (4.14) verwendet werden, da sonst die Matrizen und Vektoren nicht mehr in den Hauptspeicher passen.

Mit isotropen dünnen Gittern erreichen wir eine Testerkennungsrate von 95.6% für Level 0 und bei Level 1 werden 91.7% erzielt, siehe Tabelle 6.9. Wenn wir die Dimension 5 jeweils einmal zusätzlich verfeinern. erzielen wir 95.9% Genauigkeit, was eine geringfügige Verbesserung darstellt. Auch bei diesem Datensatz ist es somit sinnvoll, mit leicht anisotropen Gittern zu rechnen. Weiterhin zeigt sich, dass bei diesem Datensatz größere Verfeinerungsstufen keine Verbesserung ergeben. Mit etwas mehr als 4000 Punkten sind für 14 Dimension zu wenig Dateninstanzen vorhanden, um eine Funktion zu beschreiben, die eine höhere Auflösung benötigen würde. Mit linearen Support Vektor Maschinen werden in [FM01] 95.0% erreicht, die Rechenzeit beträgt wenige Sekunden.

6.5.3 Mushroom-Datensatz

Der Mushroom-Datensatz aus der UCI Machine Learning Sammlung [BM98] besteht aus 8124 hypothetischen Stichproben, die 23 verschiedene Spezies von Lamellenpilzen der Gattungen Agaricus und Lepiota beschreiben. Jede Spezies wird entweder der Klasse definitiv essbar oder der Klasse giftig zugeordnet, wobei letztere auch die Fälle nicht bekannter Genießbarkeit und nicht empfohlen beinhaltet.

Der originale Datensatz enthält 22 nominale Attribute wie z.B. den Geruch, die Form des Pilzhuts oder die Farbe. Wir benutzen eine Variante aus [MM01]. Darin werden acht Merkmale ausgewählt und in 22 binäre Indikatoren umgewandelt. Das Attribut "Oberfläche des Pilzhuts" kann z.B. die vier Werte "faserig", "schuppig", "glatt" und "gerillt" annehmen, daraus werden vier binäre Eigenschaften, von denen jeweils eine zutrifft und die anderen drei nicht, was mit Eins beziehungsweise Null gekennzeichnet wird.

Da alle 22 Attribute nur Null oder Eins sein können, reicht offensichtlich das Gitter mit zwei Punkten in jeder Dimension aus. Die benötigte Matrix für diese 22 Dimensionen kann auf 32-bit Workstations mit 2 GByte Hauptspeicher nicht gespeichert werden, da allein durch die Steifigkeitsmatrix bereits $23 \cdot 2^{22} = 96.468.992$ Einträge erzeugt werden, und über die Datenpunkte weitere Einträge hinzukommen. Die Matrix B der Daten hat allerdings nur $8124 \cdot 23$ Einträge und kann somit im Hauptspeicher bereit gehalten werden, was gewisse Geschwindigkeitsvorteile mit sich bringt.

In [MM01] werden bei Berechnungen mit SVMs aus Komplexitätsgründen nur 500 Instanzen benutzt, insbesondere bei nichtlinearen Kernen muss bei einer korrekten SVM-Behandlung eine $8124 \cdot 8124$ Matrix benutzt werden. Mit einem linearen Kern werden bei 10-facher Kreuzvalidierung Trainings- und Testerkennungsraten von 78,0% beziehungsweise 76,8% gemessen, mit einem kubischen Kern Raten von 90,9% beim Trainieren beziehungsweise 88,2% beim Testen. In [FM01] werden mit Raten von 89,0% beziehungsweise 88,9% leicht bessere Ergebnisse erzielt.

Wir beobachten bei 10-facher Kreuzvalidierung eine Trainingserkennungsrate von 89.5% mit 0,06% Standardabweichung, und eine Testerkennungsrate von 89.6% bei einer Standardabweichung von 0.52%, und erreichen damit leicht bessere und stabilere Ergebnisse. Die Rechenzeit für diese 10-fache Rechnung beträgt 16,000 Sekunden und ist damit deutlich höher als die Zeiten der SVM-Rechnungen im Bereich von wenigen hundert Sekunden, bei denen allerdings aus Komplexitätsgründen nur ein Teilproblem behandelt wird.

6.5.4 Data-Mining-Cup 2000

Wir betrachten nun den Datensatz, der beim Data-Mining-Cup 2000 [DMC00] verwendet wurde. Die Aufgabenstellung besteht im Auswählen von geeigneten Adressaten für Werbebriefe, wobei diese noch nicht Kunde beim betreffenden Versandhändler sind, aber deren Adressen bekannt sind. Um die Kosten des Mailings zu minimieren, soll nur an eine Auswahl aller Adressen der Werbebrief versendet werden. Diese Auswahl ist zu optimieren.

Dazu wurde ein Testmailing mit 10.000 zufällig ausgewählten Adressen durchgeführt und für jede angeschriebene Person gespeichert, ob diese reagiert hat oder nicht. Ausgehend davon wird ein Klassifikator generiert, der die Adressen in Bezug auf die Reagiererwahrscheinlichkeit bewertet. Dieser wird auf die insgesamt 34.820 weiteren Adressen in

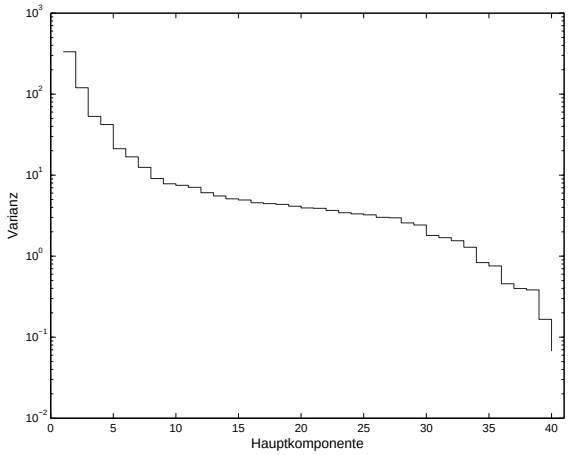


Abbildung 6.6. *Die Varianz der Hauptkomponenten des Data-Mining-Cup 2000-Datensatzes in logarithmischer Skalierung.*

der Datenbank angewendet.

Um den Gewinn durch das Mailing zu maximieren, werden die bekannten Kosten und Gewinne des Mailings benutzt. Die durchschnittlichen Kosten eines Anschreiben werden in der Wettbewerbsbeschreibung mit 12,– DM angegeben. Reagiert die angeschriebene Person, wird ein durchschnittlicher Gewinn von 185,– DM erzielt.

Die 40 Merkmale bestehen aus mikrogeographischen Daten wie z.B. Kaufkraft, Altersstruktur, Bebauungstyp, Straßentyp, Kundentyp. Alle diese Merkmale sind keine personengebundenen Informationen, sondern stellen durchschnittliche Informationen eines Gebietes dar. Die in Deutschland feinste datenschutzrechtlich freigegebene Informations-Ebene besteht aus (mindestens) fünf Haushalten. Professionelle Datenanbieter behaupten, dass sie auf dieser Ebene mit ca. fünf Millionen kleinster Straßenabschnitte eine Fülle von Informationen bereitstellen, die durch die Verdichtung von über 150 Millionen Adressdaten aus den verschiedensten Quellen generiert werden.

Um die Anzahl der Merkmale zu reduzieren, wenden wir auf die gegebenen Daten eine Hauptkomponentenanalyse (principal component analysis (PCA)) [CPS98, HTF01, Han02] auf die Trainingsdaten an. Es werden dann nur die wichtigsten Komponenten, bestimmt durch die Varianz, während des eigentlichen Lernens benutzt. Die entstehende Transformation und die Auswahl der wichtigsten Komponenten wird entsprechend auf die Testdaten angewandt. Einige Instanzen des Datensatzes weisen fehlende Werte auf, diese werden hier durch den Durchschnitt des jeweiligen Attributs ersetzt. Bei der Hauptkomponentenanalyse werden die Trainingsinstanzen mit fehlenden Werten allerdings nicht benutzt.

In Abbildung 6.6 ist die Varianz der Hauptkomponenten wiedergegeben. Wir sehen, dass die ersten zehn Komponenten einen großen Teil der Varianz wiedergeben, sie enthalten 87.6% der Gesamtvarianz. Im Folgenden verwenden wir die ersten 18 Hauptkomponenten, sie umfassen 93.5% der Varianz.

Die so transformierten Trainingsdaten werden zufällig in zwei Mengen zu 6000 und 4000

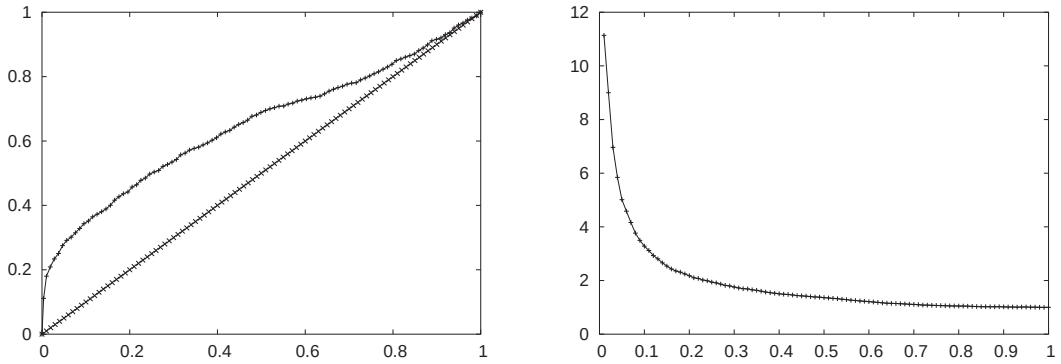
λ	GewinnTest in DM	Anteil %	real. Gewinn in DM
0.0001	9337	0.22	84986
0.00025	9645	0.23	85735
0.0005	10028	0.30	88296
0.00075	10014	0.21	87586
0.001	10039	0.23	87705
0.0025	10000	0.12	86779
0.005	8954	0.15	87241

Tabelle 6.10. *Ergebnisse des Dünngitter-Klassifikators für den Data-Mining-Cup 2000-Datensatz. Angegeben sind die mit verschiedenen Regularisierungsparametern erzielen Ergebnisse bei Level 0, deren maximaler Gewinn auf der Testmenge mit dazugehörigem Optimalwert und das mit diesen Parametern erzielte Ergebnis auf der Evaluierungsmenge. Der faire Vergleichswert ist das Ergebnis von DM 87705.*

Instanzen aufgeteilt. Auf der größeren wird jeweils gelernt, auf der kleinen werden die Ergebnisse gemessen und der beste Parametersatz bestimmt. Da das Ziel der Anwendung nicht in der Erkennungsrate liegt, sondern im maximalen Gewinn unter der Berücksichtigung der Kosten, bestehen die Parameter nicht nur aus dem Level und dem Regularisierungsparameter. Als weiteren Parameter gibt es nun den Anteil der zu benutzenden Instanzen, in diesem Falle also welcher Teil der Personen einen Werbebrief erhalten soll. Dazu werden die Instanzen in abfallender Reihenfolge ihrer vorhergesagten Reagierwahrscheinlichkeit sortiert. Anschließend wird die Position in der entstehenden Liste bestimmt, die den maximalen Gewinn erzielt, der so genannte *Optimalwert*. Dies bedeutet, dass alle Personen oberhalb des Optimalwertes angeschrieben werden und alle unterhalb nicht. Der Profit für einen Abschneidewert berechnet sich einfach dadurch, dass die Anzahl der tatsächlichen Reagierer in der genutzten Teilmenge mit 185 multipliziert wird, davon wird entsprechend das Produkt der Anzahl der Nichtreagierer mit 12 abgezogen. Für jeden Level und Regularisierungsparameter wird so der maximale Profit und der dazugehörige Abschneidepunkt bestimmt.

Mit Hilfe des Parametersatzes, der den größtem Profit auf der Testmenge erzielt hat, wird anschließend auf der ganzen Trainingsmenge der Klassifikator bestimmt. Diese Funktion wird dann auf dem Evaluierungsdatensatz angewandt und dessen Instanzen werden nach der vorhergesagten Zugehörigkeit zur Reagiererklasse sortiert. Die oberhalb des Abschneidewertes liegenden Instanzen werden nun ausgewertet und der realisierte Gewinn auf der Testmenge berechnet. Im Rahmen des Data-Mining-Cups 2000 [DMC00] reichte ein Gewinn von 67038 DM zum Erreichen des ersten Platzes. Data-Mining Spezialisten erreichten im Weiteren ein Ergebnis von 84995 DM.

In Tabelle 6.10 sind Ergebnisse für Level 0 und verschiedene Regularisierungsparameter dargestellt, höhere Level haben keine Verbesserung erzielt. Die vier Regularisierungsparameter mit den besten Ergebnissen auf der Trainingsmenge liegen mit ihren Ergebnissen nah beieinander, und auch auf der Testmenge erzielen sie ähnliche Gewinne. Mit $\lambda = 0.001$ wird der größte Profit auf der Trainingsmenge erzielt, somit ist das mit diesem Parameter



(a) ROC-Chart

(b) Lift-Chart

Abbildung 6.7. *ROC-Chart und Lift-Chart für den DMC2000-Datensatz zur Kombinationstechnik mit* $\lambda = 0.001$.

erzielte Ergebnis von 87705 DM der faire Vergleichswert mit anderen Ergebnissen. Wir liegen damit deutlich über den Ergebnissen des Wettbewerbs und auch über dem Ergebnis der Data-Mining Spezialisten. In Abbildung haben wir die ROC-Chart und den Lift-Chart für $\lambda = 0.001$ dargestellt. Es sei noch bemerkt, dass das Verhältnis von ca. 87000 zu ca. 10000 dabei ungefähr dem Verhältnis zwischen den Größen der Evaluierungsmenge (34820) und der Testmenge (4000) entspricht.

An dieser Stelle seien sozio-ökonomische Bemerkungen zur kommerziellen Nutzung von personenbezogen Daten und Wohnumfeldinformationen erlaubt. So gibt es Untersuchungen, siehe z.B. [Pro04], die vor einer zunehmend pauschalen Beurteilung von Verbrauchergruppen warnen. Es besteht die Gefahr, dass viele Menschen gewisse Dienstleistungen oder Produkte nicht in Anspruch nehmen können, weil sie von Unternehmen, die ein Data Mining-Modell in ihren Geschäftsprozess integriert haben, auf Grund der Datenlage schlicht davon ausgesperrt werden. Zu den besonders gravierenden Fällen zählen Kunden, die abhängig von ihrem Einkommen entweder sieben oder zwölf Prozent Kreditzinsen zahlen müssen, Versicherte, die pauschal auf Grund von Erkrankungen keine Berufsunfähigkeitsversicherung erhalten oder Personen, die wegen verspäteter Bezahlung der Handyrechnung und resultierendem negativen SCHUFA-Eintrag keine Wohnung bekommen.

Banken, Versicherungen, zunehmend aber auch der Einzelhandel arbeiten in immer stärkerem Maße mit Kundenprofilen und Risikogruppen – gute Konditionen gibt es dabei nur für „gute“ Kunden. Verstärkt wird dieser Trend durch immer umfassendere Datenbanken, durch Kundenkarten und einen schwunghaften Handel mit Verbraucherdaten. Personen, die in einer Risikogruppe landen, in der falschen Gegend wohnen oder über die schlicht falsche oder nicht ausreichend positive Daten gespeichert sind, stoßen bei der Versicherung, beim Kredit oder bei Handy-Verträgen zunehmend auf Schwierigkeiten. Im Extremfall kann es bedeuten, dass jemand eine Kreditabsage erhält oder von einer Versi-

Datensatz	Anz. Dim.	dünne Gitter %	Bestes anderes %	Platz	# Verfahren
breast cancer	9	97.66 ± 1.4	97.72	2	25
pima diabetis	8	76.80 ± 1.9	77.63	5	28
heart	13	80.85 ± 2.4	86.90	18	25
thyroid	5	96.07 ± 2.7	95.60	1	11
titanic	3	77.94 ± 0.7	78.96	8	19
flare-solar	9	66.85 ± 1.8	67.57	2	7

Tabelle 6.11. *Vergleich der mit der Dünngitter-Kombinationstechnik erzielten Resultate mit Ergebnissen anderer Verfahren [ROM01, MLH03, OSW03]. Angegeben sind jeweils das Ergebnis des Dünngitter-Verfahrens, das beste Ergebnis aus den Referenzen und die Position des Dünngitter-Klassifikators innerhalb einer Liste aller Verfahren.*

cherung abgewiesen wird, nur weil die Adresse auf eine schlechtere Wohngegend hindeutet. Hier wird ein individueller Kunde gewissermaßen abgestraft für früheres Fehlverhalten in seiner Umgebung, auf das er keinen Einfluss hatte.

6.5.5 Vergleich der Ergebnisse mit anderen Verfahren

Betrachten wir nun die Ergebnisse des Dünngitter-Klassifikationsverfahrens mit einer Reihe von anderen Verfahren. Wir verwenden Datensätze, die in [ROM01, MLH03, OSW03] zum Vergleich von Data Mining-Verfahren genutzt werden, wobei wir höherdimensionale Datenmengen nicht betrachten, die eine Vorverarbeitung mit einer Hauptkomponentenanalyse erfordern würden.

Die Verfahren die in den Referenzen [ROM01, MLH03, OSW03] benutzt werden, sind unter Anderem Support Vektor Maschinen, Boosting-Verfahren wie AdaBoost, Linear discriminant analysis, neuronale Netze, nächste Nachbar-Verfahren, Random forests und Multiple additive regression splines.

In Tabelle 6.11 sind die mit der Dünngitter-Kombinationstechnik erzielten Resultate im Vergleich zu den anderen Ergebnissen aufgeführt. Zu beachten ist, dass nicht jeder Datensatz in allen Referenzen beziehungsweise mit allen Verfahren verwendet wird. Wie zu sehen ist, liegt das Ergebnis der Dünngitter-Kombinationstechnik nur bei einem Datensatz in der zweiten Hälfte der verwendeten Verfahren. Bei einem Datensatz liefert sie das beste Ergebnis, bei zwei Benchmarks das zweitbeste Ergebnis.

Da in den Referenzen keine Rechenzeiten angegeben sind, ist ein Vergleich der Laufzeiten der Verfahren an dieser Stelle leider nicht möglich. Die verwendeten Benchmark-Datensätze weisen aber eher wenige Datenpunkte auf, der Größte hatte gerade mal 2200 Instanzen. Somit dürfte sich der Unterschied zwischen der linearen Laufzeit unseres Ansatzes und der nichtlinearen Laufzeit vieler anderer Verfahren für diese Datensätze nicht besonders bemerkbar machen.

6.6 Experimente mit der dimensionsadaptiven Kombinationstechnik

In diesem Abschnitt untersuchen wir die grundsätzlichen Möglichkeiten einer dimensionsadaptiven Kombinationstechnik bei der Funktionsrekonstruktion und dem maschinellen Lernen.

Es gibt bei der Umsetzung des dimensionsadaptiven Algorithmus 4.10 wie erwähnt zwei Möglichkeiten, aus den Indexmengen O und A eine Lösung auf einem verallgemeinerten dünnen Gitter mit der Kombinationstechnik zu erhalten. Bei der einen Variante werden nur die Indizes aus der alten Indexmenge O betrachtet, bei der anderen die Indizes aus der Vereinigung der aktiven und der alten Indexmenge $\mathsf{O} \cup \mathsf{A}$. Im Weiteren verwenden wir die erste Variante.

Zur Berechnung eines Adaptionskriteriums für ein Gitter $\underline{i}$ der aktiven Indexmenge A betrachten wir den Unterschied zwischen der Kombinationstechnik zur Indexmenge O und der Kombinationstechnik zu der um $\underline{i} \in \mathsf{A}$ erweiterten Indexmenge $\mathsf{I}_{\mathsf{O}+\underline{i}} := \mathsf{O} \cup \{\underline{i}\}$.

Einen einfachen und direkt von der Funktion abhängigen Fehlerindikator berechnen wir über die Darstellung des Klassifikators in der hierarchischen Basis

$$f^c_{\mathsf{I}_{\mathsf{O}+\underline{i}}} = \sum_{\phi_{\underline{l},\underline{j}} \in V_{\mathsf{I}_{\mathsf{O}+\underline{i}}}} \alpha_{\underline{l},\underline{j}} \phi_{\underline{l},\underline{j}}.$$

Für den Fehlerindikator $\varepsilon_h(\underline{i}, \mathsf{O})$ addieren wir die Betragswerte $|\alpha_{\underline{l},\underline{j}}|$ der durch das Gitter $\Omega_{\underline{i}}$ neu dazugekommenen Basisfunktionen des verallgemeinerten dünnen Gitters $\Omega_{\mathsf{I}_{\mathsf{O}+\underline{i}}}$ und teilen diese Summe durch die Anzahl der neu dazugekommenen Basisfunktionen

$$\varepsilon_h(\underline{i}, \mathsf{O}) := \frac{1}{|\{\phi_{\underline{l},\underline{j}} \mid \phi_{\underline{l},\underline{j}} \in V_{\mathsf{I}_{\mathsf{O}+\underline{i}}} \setminus V_{\mathsf{O}}\}|} \sum_{\phi_{\underline{l},\underline{j}} \in V_{\mathsf{I}_{\mathsf{O}+\underline{i}}} \setminus V_{\mathsf{O}}} |\alpha_{\underline{l},\underline{j}}|. \tag{6.1}$$

Bei der Verwendung der Hierarchie mit konstanten Funktionen nach Abschnitt 4.2.4 liefert dieser Fehlerindikator beim Übergang von den konstanten zu den linearen Funktionen allerdings keine geeigneten Ergebnisse und kann in diesem Fall nicht genutzt werden.

Andere Fehlerindikatoren beziehen die Fehler auf den Datenpunkten mit ein, d.h. wir betrachten den Fehler des Klassifikators zu den y-Werten der Datenpunkte

$$|f|_{l_2} = \frac{1}{M} \sum_{i=1}^{M} (f(\underline{x}_i) - y_i)^2 \tag{6.2}$$

und bilden die Differenz zwischen der aktuellen Funktion f_{O} und der Funktion $f_{\mathsf{I}_{\mathsf{O}+\underline{i}}}$

$$\varepsilon_{l_2} := |f_{\mathsf{O}}|_{l_2} - |f_{\mathsf{I}_{\mathsf{O}+\underline{i}}}|_{l_2}. \tag{6.3}$$

Diese Differenz ist allerdings nicht immer positiv, da die Funktion $f_{\mathsf{I}_{\mathsf{O}+\underline{i}}}$ die Werte auf den Datenpunkten unter Umständen schlechter darstellt als die Funktion f_{O}. Aus diesem Grund verwenden wir den Betrag dieses Wertes als einen alternativen Fehlerindikator

$$\varepsilon_{|l_2|} := \left| |f_{\mathsf{O}}|_{l_2} - |f_{\mathsf{I}_{\mathsf{O}+\underline{i}}}|_{l_2} \right|. \tag{6.4}$$

Level	λ	Train	Test	Gitter	Punkte
6	$1 \cdot 10^{-4}$	$1.86 \cdot 10^{-3}$	$1.84 \cdot 10^{-3}$	13	1033
7	$7.5 \cdot 10^{-5}$	$1.42 \cdot 10^{-3}$	$1.41 \cdot 10^{-3}$	15	2251
8	$5 \cdot 10^{-5}$	$1.08 \cdot 10^{-3}$	$1.15 \cdot 10^{-3}$	17	4877
9	$2.5 \cdot 10^{-5}$	$7.87 \cdot 10^{-4}$	$9.30 \cdot 10^{-4}$	19	10511
10	$2.5 \cdot 10^{-5}$	$5.45 \cdot 10^{-4}$	$8.84 \cdot 10^{-4}$	21	22545
11	$1 \cdot 10^{-5}$	$4.13 \cdot 10^{-4}$	$9.70 \cdot 10^{-4}$	23	48147

Tabelle 6.12. *Regressionsergebnisse mit der normalen Kombinationstechnik für die Funktion $e^{-x_1^2} + x_2$, angegeben ist der l_2-Quadratfehler auf den Datenpunkten der Trainings- beziehungsweise Testmenge. Weiterhin sind die Anzahl der benutzten Teilgitter und die Summe der Punkte der Teilgitter aufgeführt.*

Als weiteres Adaptionskriterium nutzen wir die Summe der Differenzen zum Quadrat der Funktionen f_{O} und $f_{\mathsf{I}_{\mathsf{O}+\underline{i}}}$ auf den Datenpunkten $\underline{x}_i$

$$\varepsilon_{\underline{x}_i} := \frac{1}{M} \sum_{i=1}^{M} (f_{\mathsf{I}_{\mathsf{O}+\underline{i}}}(\underline{x}_i) - f_{\mathsf{O}}(\underline{x}_i))^2. \tag{6.5}$$

Das Funktional (5.1) verwenden wir nicht als Fehlerindikator, da die dazu benötigte Berechnung des Regularisierungsoperators auf einem verallgemeinerten dünnen Gitter mit den bisher implementierten Datenstrukturen nicht möglich ist. Darüberhinaus ist in Abschnitt 6.1 kein wesentlicher Unterschied zwischen dem Konvergenzverhalten bezüglich des Fehlers im Funktional und bezüglich des l_2-Fehlers auf den Datenpunkten gemessen worden. Aus diesem Grund ist mit dem Funktional (5.1) auch kein grundätzlich anderes Adaptionsverhalten zu erwarten.

6.6.1 Regressionsbeispiel in zwei Dimensionen

Betrachten wir nun ein einfaches zweidimensionales Regressionsbeispiel. Wir erzeugen jeweils 5000 Trainings- und Testdaten für die Funktion $f(x_1, x_2) = e^{-x_1^2} + x_2$. Bei diesen Rechnungen ergibt das λ mit dem besten Trainingsergebnis auch jeweils die besten Ergebnisse auf den Testdaten, so dass wir bei diesen Experimenten auf eine Evaluierungsmenge verzichten und das λ über das beste Trainingsergebnis bestimmt wird. Als Approximationsfehler der Regression wird der l_2-Quadratfehler auf den Datenpunkten (6.2) angegeben.

In Tabelle 6.12 sind die Ergebnisse der normalen Kombinationstechnik wiedergegeben. Dabei ist zu beobachten, dass bis Level 10 die Ergebnisse besser werden und dann Überanpassungseffekte auftreten.

In Tabelle 6.13 sind die Ergebnisse der Experimente mit der dimensionsadaptiven Kombinationstechnik angegeben. Der Algorithmus 4.10 bricht ab, wenn die Elemente der aktiven Indexmenge A alle einen lokalen Fehlerindikator ε größer als die vorgegebene Toleranz θ aufweisen. Somit sind die Parameter bei der dimensionsadaptiven Kombinationstechnik der Regularisierungsparameter und die Toleranzschranke.

Indikator	Toleranz	λ	Train	Test	Gitter	Punkte
ε_h	0.005	$1 \cdot 10^{-5}$	$6.89 \cdot 10^{-4}$	$6.96 \cdot 10^{-4}$	6	78
	0.0025	$1 \cdot 10^{-5}$	$6.89 \cdot 10^{-4}$	$6.96 \cdot 10^{-4}$	6	78
	0.001	$2.5 \cdot 10^{-5}$	$2.41 \cdot 10^{-4}$	$2.40 \cdot 10^{-4}$	7	144
	0.00075	$5 \cdot 10^{-5}$	$3.72 \cdot 10^{-4}$	$3.62 \cdot 10^{-4}$	7	144
	0.0005	$1 \cdot 10^{-4}$	$6.58 \cdot 10^{-4}$	$6.32 \cdot 10^{-4}$	8	274
	0.0001	$2.5 \cdot 10^{-5}$	$5.91 \cdot 10^{-4}$	$6.26 \cdot 10^{-4}$	47	24772
ε_{l_2}	0.00005	$5 \cdot 10^{-7}$	$4.30 \cdot 10^{-5}$	$4.44 \cdot 10^{-5}$	8	274
	0.00001	$1 \cdot 10^{-7}$	$1.07 \cdot 10^{-5}$	$1.09 \cdot 10^{-5}$	9	532
	0.000005	$1 \cdot 10^{-8}$	$2.67 \cdot 10^{-6}$	$2.87 \cdot 10^{-6}$	10	1046
	0.000001	$1 \cdot 10^{-9}$	$7.59 \cdot 10^{-7}$	$8.74 \cdot 10^{-7}$	11	2072
$\varepsilon_{\lvert l_2 \rvert}$	0.0001	$2.5 \cdot 10^{-5}$	$9.32 \cdot 10^{-4}$	$9.73 \cdot 10^{-4}$	32	6371
	0.00005	$2.5 \cdot 10^{-5}$	$8.56 \cdot 10^{-4}$	$8.75 \cdot 10^{-4}$	37	10137
	0.00001	$5 \cdot 10^{-6}$	$3.44 \cdot 10^{-4}$	$4.19 \cdot 10^{-4}$	55	135540
	0.000005	$7.5 \cdot 10^{-6}$	$3.45 \cdot 10^{-4}$	$4.19 \cdot 10^{-4}$	57	192889
$\varepsilon_{\underline{x}}$	0.0005	$5 \cdot 10^{-5}$	$1.03 \cdot 10^{-3}$	$1.04 \cdot 10^{-3}$	29	3289
	0.00025	$2.5 \cdot 10^{-5}$	$8.13 \cdot 10^{-4}$	$8.23 \cdot 10^{-4}$	38	12702
	0.0001	$1 \cdot 10^{-5}$	$3.52 \cdot 10^{-4}$	$3.99 \cdot 10^{-4}$	52	68067
	0.00075	$7.5 \cdot 10^{-6}$	$3.47 \cdot 10^{-4}$	$3.93 \cdot 10^{-4}$	56	135670

Tabelle 6.13. *Regressionsergebnisse mit der dimensionsadaptiven Kombinationstechnik für die Funktion $e^{-x_1^2}+x_2$, angegeben ist der l_2-Quadratfehler auf den Datenpunkten der Trainings- beziehungsweise Testmenge. Weiterhin sind die Anzahl der Teilgitter in der Indexmenge $\mathsf{O} \cup \mathsf{A}$ und die Summe der Punkte dieser Teilgitter aufgeführt.*

Mit dem Fehlerindikator ε_h wird das beste Ergebnis mit der Toleranz $\theta = 0.001$ erreicht, berechnet werden dabei die Gitter mit den Indizes $(i, 0), i = 0, \ldots, 5$ und $(0, 1)$. In der verallgemeinerten Kombinationsformel für den Indexset O ergibt sich nur beim Gitter $\Omega_{4,0}$ ein Faktor ungleich Null. Die zu rekonstruierende Funktion ist in der zweiten Koordinate linear, es ist also nicht notwendig, diese Dimension mit mehr als zwei Gitterpunkten aufzulösen. Auch bei dem besten Ergebnis mit dem Fehlerindikator ε_{l_2} wird die zweite Dimension nicht weiter aufgelöst. Es werden die Gitter $(i, 0), i = 0, \ldots, 9$ und $(0, 1)$ berechnet, in der Kombinationsformel ist allein das Gitter $\Omega_{8,0}$ aktiv. Damit wird auf der Testmenge das beste Ergebnis in diesem Experiment mit einem Fehler von $8.74 \cdot 10^{-7}$ erreicht.

Die Rechnungen mit den Fehlerindikatoren $\varepsilon_{|l_2|}$ und $\varepsilon_{\underline{x}}$ liefern bei diesem Datensatz die schlechtesten adaptiven Ergebnisse, außerdem verwenden sie deutlich mehr Gitter als die anderen Ansätze. Aber diese beiden Fehlerindikatoren erzielen immer noch bessere Ergebnisse als Rechnungen auf einem normalen dünnen Gitter, allerdings mit sechsmal so vielen Gitterpunkten.

Level	λ	Train	Test	Gitter	Punkte
3	0.00074505	0.015970	0.016447	35	1712
4	0.00093132	0.013651	0.014634	69	5297
5	0.00145519	0.012821	0.014285	121	15149
6	0.00181899	0.012475	0.015169	195	41103
7	0.00181899	0.012218	0.015458	295	107427
8	0.00181899	0.011785	0.017320	425	273013

Tabelle 6.14. *Regressionsergebnisse mit der normalen Kombinationstechnik für das vierdimensionale Beispiel, angegeben ist der l_2-Quadratfehler auf den Datenpunkten sowie die Anzahl der benutzten Teilgitter und die Summe der Punkte der Teilgitter.*

6.6.2 Regressionsbeispiel in vier Dimensionen

Wir unternehmen nun Rechnungen für ein vierdimensionales Regressionsbeispiel, die darzustellende Funktion ist $f(x_1, x_2, x_3, x_4) = e^{-x_1^2} + e^{-x_2^2 - x_3^3}$. Die vierte Variable hat keinen Einfluss auf den Funktionswert. In Tabelle 6.14 sind die Ergebnisse der normalen Kombinationstechnik wiedergegeben, das beste Ergebnis wird für Level 5 mit einem Fehler von 0.0143 erreicht.

Der Fehlerindikator ε_{l_2}, der beim zweidimensionalen Beispiel deutlich am besten war, bricht bei diesem Beispiel zu früh mit der Adaption ab, da er bereits nach wenigen Adaptionsschritten negative Werte als Fehlerindikatoren der Gitter in der aktiven Indexmenge A liefert.

Die dimensionsadaptive Kombinationstechnik mit den anderen drei Fehlerindikatoren ε_h, $\varepsilon_{|l_2|}$ und $\varepsilon_{\underline{x}}$ liefert jeweils bessere Ergebnisse als ein normales dünnes Gitter, siehe Tabelle 6.15. Dabei erzielt der Fehlerindikator $\varepsilon_{|l_2|}$, der den Absolutwert der Veränderung der Approximation der Datenwerte $\{y_j\}_1^M$ durch die Funktion $f_{\mathsf{I}_{\mathsf{O}+\underline{i}}}$ misst, die besten Ergebnisse mit einem Fehler von 0.0109. Weiterhin sind hierbei die Ergebnisse auf der Trainingsmenge und der Testmenge näher beieinander als bei den beiden anderen Fehlerindikatoren, somit ist der Fehlerindikator in einem gewissen Sinne stabiler.

Allerdings wird bei den Rechnungen mit guten Gesamtergebnissen auch die vierte Dimension verfeinert, obwohl dies für die Darstellung der Funktion nicht notwendig ist. Da bei kleinen Regularisierungsparametern dieser Effekt nicht auftritt, ist davon auszugehen, dass dies durch den Regularisierungsoperator verursacht wird, der alle Dimensionen gleich behandelt. Andererseits erzielen Rechnungen mit derart kleinen Regularisierungsparametern keine brauchbaren Fehlerraten.

Das beste Ergebnis mit einem Fehler von 0.0109 wird bei einer Toleranz von $\theta = 0.0001$ mit dem Fehlerindikator $\varepsilon_{|l_2|}$ erreicht, es werden dabei 81 Gitter mit insgesamt 12812 Gitterpunkten betrachtet. Das sind weniger als Level 5 der normalen Kombinationstechnik verwendet, welcher das beste Ergebnis unter allen Verfeinerungsstufen erzielt hat. Die Anzahl der benutzten Gitter bei einer effizienten dimensionsadaptiven Kombinationstechnik kann aber sicherlich noch weiter reduziert werden.

Indikator	Toleranz	λ	Train	Test	Gitter	Punkte
ε_h	0.025	0.00025	0.026329	0.025285	12	412
	0.01	0.00025	0.013672	0.013616	28	1756
	0.0075	0.00025	0.012673	0.013539	64	5954
	0.005	0.001	0.010664	0.011586	74	6418
	0.0025	0.001	0.011963	0.015132	262	137995
$\varepsilon_{\|l_2\|}$	0.001	1e-05	0.020213	0.019963	7	224
	0.0005	0.00075	0.014079	0.014730	39	3194
	0.00025	0.001	0.011646	0.012139	48	4240
	0.0001	0.00075	0.010253	0.010896	81	12812
	0.000075	0.001	0.012882	0.013950	123	22710
$\varepsilon_{\underline{x}}$	0.01	0.00025	0.014121	0.014346	17	776
	0.0075	0.00075	0.014329	0.014626	17	776
	0.005	0.0005	0.014278	0.014721	23	1624
	0.0025	0.001	0.012081	0.013096	63	6416
	0.001	0.0025	0.011671	0.015488	239	65156

Tabelle 6.15. *Regressionsergebnisse mit einer dimensionsadaptiven Kombinationstechnik für das vierdimensionale Beispiel, angegeben ist der l_2-Quadratfehler auf den Datenpunkten sowie die Anzahl der Teilgitter in der Indexmenge $\mathsf{O} \cup \mathsf{A}$ und die Summe der Punkte dieser Teilgitter.*

6.6.3 Deutscher Kreditdatensatz

Der *Deutsche Kreditdatensatz* wird in [MST94] mit einer Reihe von Data Mining Verfahren behandelt. Die Anwendung besteht in der Kreditvergabe, die zwanzig Attribute des originalen Datensatzes enthalten, unter anderem

- Status des existierenden Girokontos,
- Dauer des Girokontos,
- Kreditgeschichte,
- Grund des Kredites,
- Betrag des Kredites,
- Status des Sparkontos,
- Dauer der Beschäftigung,
- Rate in Prozent des zur Verfügung stehenden Einkommens,
- Geschlecht und Ehestatus,
- Weitere Kredite,

- Weiteres Eigentum,
- Art und Dauer der Wohnung,
- Art der Arbeit.

Für die Vergleiche in [MST94] wurden die kategorischen Attribute in numerische umgewandelt. Dadurch wurde die Anzahl der Attribute auf 24 erhöht. Wir benutzen diese Variante des Datensatzes, der aus 1000 Instanzen besteht. Desweiteren wird zu diesem Datensatz die Kostenmatrix

		reale Klasse	
		gut	schlecht
vorhergesagte Klasse:	gut	0	5
vorhergesagte Klasse:	schlecht	1	0

angegeben.

Es ist schlimmer, einen schlechten Kunden als gut zu klassifizieren, als einen guten Kunden als schlecht einzuordnen. In diese modellhafte Kostenmatrix aus [MST94] gehen nur die falschen Vorhersagen ein. Bei realen Anwendungen dieser Art wird aber meist dem richtig vorhergesagten guten Kunden ebenso ein Wert zugewiesen, um den potentiellen Gesamtgewinn zu berechnen. Der vorliegende Datensatz besteht aus 700 Instanzen der Klasse gut ($= 1$) und 300 Instanzen der Klasse schlecht ($= -1$). Um dieses Ungleichgewicht der beiden Datenklassen zu beheben, wird die kleine Klasse stärker gewichtet, dies geschieht z.B. durch die Verschiebung der Trennpositon zwischen den beiden Klassen, numerisch 1 bzw. -1 von der Stelle 0 auf die Position 0.7. Alternativ kann auch für die zweite Klasse statt -1 ein größerer Wert gewählt werden und die Trennposition an der Stelle 0 beibehalten werden. Hierbei entspräche die erwähnte Trennposition 0.6 für das Intervall [-1;1] ungefähr dem Wert -5. Die Position der Trennstelle ist somit ein weiterer Parameter in der Untersuchung des Verhaltens eines Data Mining Verfahrens. Sofern wir nur an der Sortierung der Instanzen nach ihrer wahrscheinlichen Zugehörigkeit zu einer Klasse interessiert sind, spielt die Trennposition keine Rolle. In diesem Benchmark wird das Ergebnis aber über die Klasseneinteilungen bestimmt, so dass diese unterschiedliche Gewichtung notwendig ist.

In [MST94] werden verschiedene Data Mining Verfahren wie z.B. Nächste Nachbarn, Naive Bayes, Entscheidungsbäume oder Neuronale Netze auf diesen Datensatz angewandt und die Ergebnisse verglichen. Interessanterweise erreichen nur 10 der 23 Verfahren bessere Ergebnisse als die naive Vorgabe, bei der alle Daten der Klasse -1 zugeordnet werden und somit durchschnittliche Kosten von 0.7 erreicht werden. Natürlich ist diese Regel für die Praxis nicht zu gebrauchen, da so keine Kredite mehr vergeben würden. Die besten Ergebnisse liefert Fishers lineare bzw. logistische Diskriminanzanalyse mit Testkosten von 0.535, alle anderen Verfahren erzielen Testkosten von 0.583 und schlechter.

Der Datensatz hat 24 Attribute, ein Gittervektor dieser Größe kann zwar grundsätzlich im Speicher bereit gehalten werden, allerdings ist die benötigte Rechenzeit für die on-the-fly Berechnung der Matrixoperation in der Praxis zu lang. Wir benutzen stattdessen den dimensionsadaptiven Algorithmus, wobei wir in jeder Dimension mit der konstanten

Toleranz	λ	Train	Test
0.001	0.1	94.24 ($\pm$ 0.5)	93.12 ($\pm$ 4.7)
0.0005	0.1	94.59 ($\pm$ 0.4)	93.15 ($\pm$ 3.7)
0.0001	0.1	95.03 ($\pm$ 0.6)	93.73 ($\pm$ 3.9)
0.00005	0.25	95.35 ($\pm$ 0.6)	93.71 ($\pm$ 3.6)
0.00001	0.25	95.56 ($\pm$ 0.6)	93.72 ($\pm$ 3.7)

Tabelle 6.16. *Ergebnisse der dimensionsadaptiven Kombinationstechnik für den Ionosphere-Datensatz. Angegeben sind die Erkennungsraten die per 10-facher Kreuzvalidierung erreicht werden.*

Funktion anfangen, d.h. die einzelnen Dimensionen werden erst während der laufenden Rechnung aktiviert.

Wie erwähnt zeigen wir in diesem Abschnitt die grundsätzlichen Möglichkeiten des dimensionsadaptiven Verfahrens, somit geben wir nur die jeweils besten Erkennungsraten an, ohne eine im Data-Mining-Sinne saubere Bestimmung der Parameter des Verfahrens unternommen zu haben. Sofern wir als Adaptionskriterium ε_{l_2} nutzen, d.h. die Differenz des globalen l_2-Fehlers an den Datenpunkten, und das Abbruchkriterium bei 0.0005 setzen, erhalten wir bei der zehnfachen Kreuzvalidierung Testkosten von 0.516, erreichen damit etwas bessere Ergebnisse als die beiden besten Verfahren aus [MST94] und deutlich bessere Ergebnisse als die restlichen dort untersuchten Ansätze.

Weiterhin untersuchen wir die Möglichkeit, anhand der Auswahl der dimensionsadaptiven Rechnungen eine Dimensionsreduktion durchzuführen. Dazu wählen wir zuerst 11 Attribute aus, die so gut wie immer genutzt werden. In einem zweiten Test werden nur die Attribute nicht gewählt, die in den meisten Fällen nicht genutzt werden, dieser Ansatz ergibt eine Reduktion auf 18 Dimensionen. Die für diese reduzierten Dimensionszahlen mit der normalen Kombinationstechnik durchgeführten Rechnungen führen allerdings nicht zu verbesserten Ergebnissen, deutliche Überanpassungseffekte sind festzustellen.

6.6.4 Ionosphere-Datensatz

Der Ionosphere-Datensatz [SWHB89, BM98] besteht aus Radardaten, aufgenommen von einem System von 16 Hochfrequenzantennen mit einer Gesamtleistung von 6,4 Kilowatt. Das Ziel sind freie Elektronen in der Ionosphäre. Die empfangenen elektromagnetischen Signale wurden mit einer Autokorrelationsfunktion verarbeitet, deren Resultate bilden die 34 kontinuierlichen Attribute dieses Datensatzes. Die 351 Messungen sind eingeteilt in zwei Sorten von Radarechos. Zum einen Signale, die Hinweise auf gewisse Strukturen in der Ionosphäre aufweisen, und zum anderen solche, die es nicht tun, diese Signale passieren die Ionosphäre.

Da der Datensatz 34 Dimensionen aufweist, benutzen wir wiederum den dimensionsadaptiven Algorithmus mit konstanten Funktionen, d.h. die einzelnen Dimensionen werden erst während der laufenden Rechnung aktiviert. Wir betrachten Ergebnisse mit 10-fach Kreuzvalidierung, in einigen anderen Arbeiten wird der Datensatz in Training- und Test-

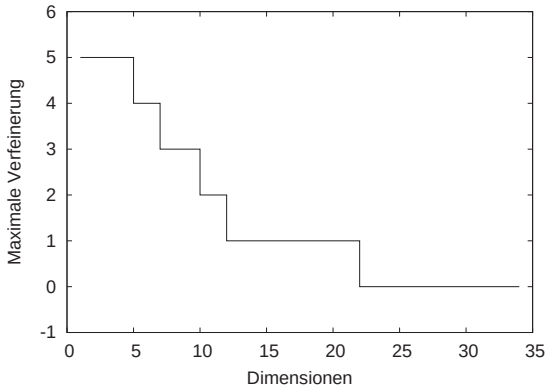


Abbildung 6.8. *Die maximalen Verfeinerungen der (sortierten) Dimensionen in der Indexmenge* O *beim Ionosphere-Datensatz.*

mengen aufgeteilt und dort werden Erkennungsraten zwischen 90% und 96% auf der Testmenge gemessen. In [FM01] wird mit 10-facher Kreuzvalidierung für diesen Datensatz eine Testgenauigkeit von 88.3% mit einer linearen und eine von 95.5% mit einer nichtlinearen Support Vektor Maschine erreicht.

In Tabelle 6.6.4 sind die Ergebnisse für verschiedene Toleranzen und den Fehlerindikator ε_{l_2} dargestellt. Wir erreichen bei 10-facher Kreuzvalidierung ein durchschnittliches Ergebnis von 93.7% auf der Testmenge. In Abbildung 6.8 sind die maximalen Verfeinerungen der Dimensionen in der Indexmenge O für dieses Ergebnis dargestellt. Wie zu sehen, werden viele Dimension gar nicht oder höchstens einmal verfeinert. Nur vier Dimensionen werden bis zu fünfmal verfeinert.

Kapitel 7

Zusammenfassung und Ausblick

In dieser Arbeit haben wir die Funktionsrekonstruktion mit der Dünngitter-Kombinationstechnik für Anwendungen im maschinellen Lernen betrachtet. Konkret zu lösen ist dabei ein regularisiertes Approximationsproblem in einem Funktionenraum. Die Verwendung einer Diskretisierung mit festen Gitterpunkten, wobei die Gitterpunkte unabhängig von den Datenpunkten gewählt werden, ist in diesem Anwendungsgebiet neu und wird erst durch die Nutzung von dünnen Gittern realisierbar. Wichtig für viele praktische Anwendungen ist die Erkenntnis, dass der Aufwand des Verfahrens nur linear mit der Zahl der Daten skaliert, aber trotzdem eine nichtlineare Funktion berechnet wird. Dies gilt insbesondere im Vergleich zu vielen anderen maschinellen Lernverfahren, die nichtlinear mit der Zahl der Daten skalieren. Allerdings ist die Anzahl der Dimensionen beschränkt, die mit unserer Methode behandelt werden können. In vielen realen Anwendungen kann aber die Dimension des Problems durch Vorverarbeitungsschritte substantiell reduziert werden.

Zur Untersuchung der numerischen Eigenschaften des Verfahrens haben wir das Konvergenzverhalten der diskreten Lösung im Vergleich zur Lösung auf einem hochaufgelösten vollen Gitter untersucht. Dabei sind zwei unterschiedliche Ordnungen des Approximationsverhaltens festgestellt worden. Die Diskretisierungsauflösung, bei der dieser Übergang zwischen den beiden Phasen stattfindet ist dabei abhängig von der Anzahl und der Verteilung der zu approximierenden Daten. Sowohl die Vollgitterlösungen als auch die Dünngitterlösungen weisen dieses Verhalten auf, wobei die Konvergenzordnung des dünnen Gitters wie zu erwarten etwas schlechter ist. Bei der theoretischen Betrachtung eines verwandten Wavelet-Kern-Ansatzes haben wir in ähnlicher Form zwei Konvergenzphasen festgestellt

Weiterhin haben wir das Dünngitter-Verfahren auf eine Reihe von Benchmark-Datensätzen angewandt. Dabei erzielen wir Resultate, deren Qualität mit denen der besten maschinellen Lernverfahren vergleichbar ist. Bei einigen großen Beispielen messen wir darüberhinaus die Laufzeit des Verfahrens. Die vorhergesagte lineare Skalierung des Aufwands mit der Zahl der Dateninstanzen wird bestätigt.

Als eine Erweiterung haben wir verallgemeinerte dünne Gitter auf Basis einer Hierarchie definiert, die auch konstante Basisfunktionen enthält. Für diese Gitter führen wir eine dimensionsadaptive Kombinationstechnik ein, welche die Behandlung von Problemen mit höheren Dimensionen ermöglicht. Um ein effizientes dimensionsadaptives Verfahren zu

erhalten, ist ein geeigneter Fehlerindikator und eine darauf aufbauende Adaptionsstrategie notwendig. Wir haben hierzu Experimente mit verschiedenen Fehlertermen durchgeführt und die grundsätzlichen Möglichkeiten einer dimensionsadaptiven Kombinationstechnik im Bereich des maschinellen Lernens gezeigt. Die erzielten Ergebnisse sind vielversprechend, aber das jetzige Verfahren ist für Praxisanwendungen noch nicht robust genug. Das Zusammenspiel zwischen Fehlerindikator, Adaptionsstrategie und Abbruchkriterium muss in der Zukunft eingehend untersucht werden. Perspektivisch besteht die Hoffnung, dass eine Theorie für robuste und effiziente Fehlerschätzung entwickelt werden kann, wie sie bei der numerischen Behandlung von partiellen Differentialgleichungen für lokale Adaptionsmethoden existiert [BR78, EEHJ95, Ver96, BR01].

Interessante Aspekte für die Untersuchung in weiteren Arbeiten ergeben sich bei einer Parallelisierung der dimensionsadaptiven Kombinationstechnik. Bei der Nutzung der grobkörnigen Parallelität werden dabei mehrere Teilprobleme aus der aktiven Indexmenge A gleichzeitig berechnet. Wenn nun ein Teilergebnis beim Kontrollprozess eintrifft, werden die Fehlerindikatoren bestimmt und die Berechnung eines weiteren Teilproblems wird gestartet, wobei dieses anhand der Adaptionsstrategie ausgewählt wird. Der Unterschied zwischen sequentiellen und parallelen Verfahren zeigt sich nun bei dieser Auswahl des nächsten Teilproblems. Beim sequentiellen Ablauf sind bei der adaptiven Auswahl alle Teilergebnisse für die aktive Indexmenge bekannt. Für die parallele Variante trifft dies aber nicht zu, da Teilgitter noch berechnet werden müssen. Das bedeutet, dass bei einer parallelen dimensionsadaptiven Kombinationstechnik eine andere Reihenfolge in der Berechnung der Teilgitter stattfinden kann. Somit wird das parallele Verfahren andere Ergebnisse liefern als die sequentielle Version. Es ist zu untersuchen, welche Auswirkungen eine derart veränderte Adaption auf das Verfahren hat. Dies betrifft sowohl die Qualität der Ergebnisse als auch die Zahl der verwendeten Teilprobleme.

Eine mögliche Erweiterung unseres Ansatzes für das maschinelle Lernen besteht in der Integration von vorhandenem Wissen in die Problemstellung. Naheliegend ist die Nutzung von anderen Randbedingungen, es wird sicherlich oft bekannt sein, welche Klassen beziehungsweise Funktionswerte an einigen Rändern des Gebietes vorliegen. Diese Information kann in die Formulierung durch Dirichlet-Randwerte eingebracht werden, was zu gemischten Randbedingungen führen wird. Es ist ebenso zu untersuchen, ob und wie weiteres vorhandenes Wissen in die Problemformulierung eingebracht werden kann. In der Regularisierungstheorie gibt es Aussagen, wie konvexe Nebenbedingungen geeignet in die Problemformulierung integriert werden können [Mor84, EHN96].

Darüberhinaus stellt sich die verwandte Frage, wie mit einem Klassifikator bei einer Erweiterung des Datenbestandes umgegangen wird. Es kann zwar mit allen alten und neuen Daten der Klassifikator komplett neu bestimmt werden. Aber es ist eventuell möglich, den mit den älteren Daten bestimmten Klassifikator geeignet in ein Problem bezüglich der neuen Daten einzubringen.

Aus Gründen der Einfachheit haben wir bisher nur den Operator $\mathcal{S} = \nabla$ zur Regularisierung benutzt, aber andere Differential- oder Pseudo-Differentialoperatoren lassen sich ebenfalls einsetzen. Für die Regularisierung durch Diskretisierung kann darüberhinaus untersucht werden, inwiefern Operatoren, die Glattheitseigenschaften fordern, zur Stabilisierung des numerischen Verfahrens notwendig sind, oder ob eine einfache Stabilisierung wie

z.B. durch die Addition der gewichteten Diagonalen der Datenmatrix $B^t \cdot B$ ausreichend ist.

Mit wenigen Modifikationen kann unser Ansatz auch zur Dichteschätzung eingesetzt werden. Dabei ist nur eine Punktverteilung S gegeben

$$S = \{\underline{x}_i \in \mathbb{R}^d\}_{i=1}^M.$$

Die Aufgabe ist es nun, die dieser Punktverteilung zu Grunde liegende Wahrscheinlichkeitsdichte zu bestimmen. Mit dem Ansatz als Variationsproblem [HHR00]

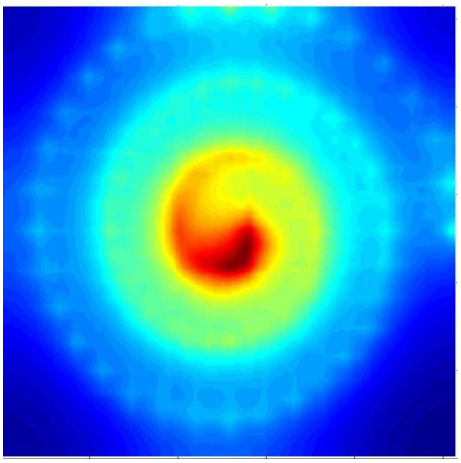
$$R(f) \underset{f \in V}{\longrightarrow} \min !$$

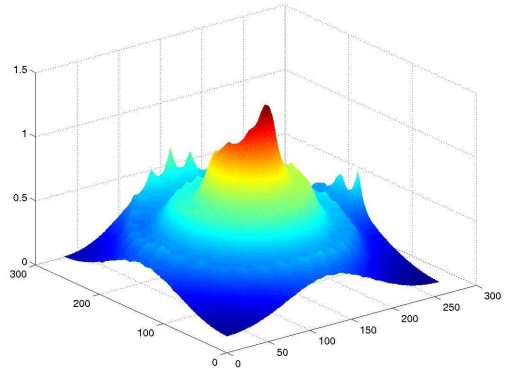
wobei

$$R(f) = \|f\|_{L_2}^2 + \lambda \|\mathcal{S} f\|^2 - \frac{1}{M} \sum_{i=1}^{M} f(\underline{x}_i),$$

kann die Dichteschätzung mit einem Dünngitter-Ansatz behandelt werden, analog zum vorgestellten Verfahren zur Funktionsrekonstruktion. So können nun d-dimensionale Klassifikationsprobleme auch als $(d+1)$-dimensionale Dichteschätzungsaufgaben formuliert werden. Dadurch ist in dieser Form die Behandlung von mehreren Klassen bei der Klassifikation einfach möglich.

Wir beschließen nun den Ausblick und diese Arbeit mit Bildern eines zweidimensionalen Experiments zur Dichteschätzung mit der Dünngitter-Kombinationstechnik.





Literaturverzeichnis

[Ach03] Achatz, S. *Higher Order Sparse Grid Methods for Elliptic Partial Differential Equations with Variable Coefficients.* Computing, 71(1):1–15, 2003.

[ADT95] Arge, E., M. Dæhlen und A. Tveito. *Approximation of scattered data using smooth grid functions.* J. Comput. Appl. Math, 59:191–205, 1995.

[AIS93] Agrawal, R., T. Imielinski und A. Swami. *Mining Association Rules between Sets of Items in Large Databases.* In *Proc. of the ACM SIGMOD Int'l Conference on Management of Data*, Seiten 207–216. Washington D.C., 1993.

[Aro50] Aronszajn, N. *Theory of reproducing kernels.* Trans. Amer. Math. Soc., 68:337–404, 1950.

[Bab60] Babenko, K. I. *Approximation of periodic functions of many variables by trigonometric polynomials.* Dokl. Akad. Nauk SSSR, 132:247–250, 1960. Russian, Engl. Transl.: Soviet Math. Dokl. 1:513–516, 1960.

[Bal94] Balder, R. *Adaptive Verfahren für elliptische und parabolische Differentialgleichungen auf dünnen Gittern.* Doktorarbeit, Technische Universität München, 1994.

[Bas85] Baszenski, G. *N–th order polynomial spline blending.* In W. Schempp und K. Zeller (Herausgeber), *Multivariate Approximation Theory III*, ISNM 75, Seiten 35–46. Birkhäuser, Basel, 1985.

[BBDM02] Binder, T., L. Blank, W. Dahmen und W. Marquardt. *On the Regularization of Dynamic Data Reconciliation Problems.* J. Proc. Cont., 12(4):557–567, 2002.

[BCCI98] Banoczi, J. M., N.-C. Chiu, G. E. Cho und I. C. F. Ipsen. *The Lack of Influence of the Right-Hand Side on the Accuracy of Linear System Solution.* SIAM J. Sci. Comput, 20(1):203–227, 1998.

[BDJ92] Baszenski, G., F.-J. Delvos und S. Jester. *Blending approximations with sine functions.* In D. Braess (Herausgeber), *Numerical Methods in Approximation Theory IX*, ISNM 105, Seiten 1–19. Birkhäuser, Basel, 1992.

[BF91] Brezzi, F. und M. Fortin. *Mixed and hybrid finite element methods*, Band 15 von *Springer Series in Computational Mathematics*. Springer-Verlag, New York, 1991.

[BFOS84] Breiman, L., J. H. Friedman, R. A. Olshen und C. J. Stone. *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth Publishing Company, Belmont, California, U.S.A., 1984.

[BG99] Bungartz, H.-J. und M. Griebel. *A Note on the Complexity of Solving Poisson's Equation for Spaces of Bounded Mixed Derivatives*. J. Complexity, 15:167–199, 1999.

[BG04] Bungartz, H.-J. und M. Griebel. *Sparse Grids*. In *Acta Numerica, 2004*, Band 13, Seiten 147–269. Cambridge Univ. Press, Cambridge, 2004.

[BGR94] Bungartz, H.-J., M. Griebel und U. Rüde. *Extrapolation, combination, and sparse grid techniques for elliptic boundary value problems*. Comput. Methods Appl. Mech. Eng., 116:243–252, 1994.

[BGRZ94a] Bungartz, H.-J., M. Griebel, D. Röschke und C. Zenger. *Pointwise convergence of the combination technique for the Laplace equation*. East-West J. Numer. Math., 2:21–45, 1994.

[BGRZ94b] Bungartz, H.-J., M. Griebel, D. Röschke und C. Zenger. *Two proofs of convergence for the combination technique for the efficient solution of sparse grid problems*. In D. E. Keyes und J. Xu (Herausgeber), *Domain Decomposition Methods in Scientific and Engineering Computing, DDM7*, Contemp. Math. 180, Seiten 15–20. American Mathematical Society, Providence, 1994.

[BL00] Berry, M. J. A. und G. S. Linoff. *Mastering Data Mining*. Wiley, 2000.

[BM98] Blake, C. L. und C. J. Merz. *UCI Repository of machine learning databases*. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 1998.

[BM03] Brendel, M. und W. Marquardt. *Stepwise Refinement of Sparse Grids in Data Mining Applications*. In *Proc. of the 13th IFAC Symposium on System Identification, SYSID 2003*, Seiten 1667–1672. 2003.

[BP03] Bruckner, G. und S. Pereverzev. *Self-regularization of projection methods with a posteriori discretization level choice for severely ill-posed problems*. Inverse Problems, 19(1):147–156, 2003.

[BR78] Babuska, I. und W. Rheinboldt. *Error estimates for adaptive finite element computations*. SIAM J. Numer. Anal., 15:736–754, 1978.

[BR01] Becker, R. und R. Rannacher. *An optimal control approach to a posteriori error estimation in finite element methods*. In *Acta Numerica, 2001*, Seiten 1–102. Cambridge Univ. Press, Cambridge, 2001.

[Bre01] Breiman, L. *Random forests.* Machine Learning, 45(1):5–32, 2001.

[Bun92] Bungartz, H.-J. *Dünne Gitter und deren Anwendung bei der adaptiven Lösung der dreidimensionalen Poisson-Gleichung.* Doktorarbeit, Institut für Informatik, Technische Universität München, 1992.

[Bun98] Bungartz, H.-J. *Finite Elements of Higher Order on Sparse Grids.* Habilitation, Institut für Informatik, Technische Universität München and Shaker Verlag, Aachen, 1998.

[CF88] Chan, T. F. und D. E. Foulser. *Effectively well-conditioned linear systems.* SIAM J. Sci. Stat. Comput., 9:963–969, 1988.

[CK04] Castano, D. und A. Kunoth. *Multilevel Regularization of Wavelet Based Fitting of Scattered Data - Some Experiments.* Numer. Algor., 2004. To appear.

[CMR03] Canu, S., X. Mary und A. Rakotomamonjy. *Functional Learning through Kernels.* In J. Suykens, G. Horvath, S. Basu, C. Micchelli und J. Vandewalle (Herausgeber), *Advances in Learning Theory: Methods, Models and Applications*, Band 190 von *NATO Science Series: Computer & Systems Sciences*, Seiten 89–110. IOS Press Amsterdam, 2003.

[CPS98] Cios, K., W. Pedrycz und R. Swiniarski. *Data Mining Methods for Knowledge Discovery.* Kluwer, 1998.

[CS01] Cucker, F. und S. Smale. *On the mathematical foundations of learning.* Bulletin of the AMS, 39(1):1–49, 2001.

[CS02] Cucker, F. und S. Smale. *Best choices for regularization parameters in learning theory: on the bias-variance problem.* Found. Comput. Math., 2(4):413–428, 2002.

[Dac03] Dachkovski, S. *Anisotropic function spaces and related semi-linear hypoelliptic equations.* Math. Nachr., 248/249:40–61, 2003.

[Dah97] Dahmen, W. *Wavelet and multiscale methods for operator equations.* In *Acta Numerica, 1997*, Seiten 55–228. Cambridge Univ. Press, Cambridge, 1997.

[Del82] Delvos, F.-J. *d-Variate Boolean Interpolation.* J. Approx. Theory, 34:99–114, 1982.

[DeV98] DeVore, R. *Nonlinear approximation.* In *Acta Numerica, 1998*, Seiten 51–150. Cambridge Univ. Press, Cambridge, 1998.

[Die00] Dietterich, T. G. *Ensemble Methods in Machine Learning.* Lecture Notes in Computer Science, 1857:1–15, 2000.

[DK92] Dahmen, W. und A. Kunoth. *Multilevel preconditioning.* Numer. Math., 63(3):315–344, 1992.

[DMC00] *Data Mining Cup 2000*. `http://www.data-mining-cup.de/2000/`, 2000.

[DS89] Delvos, F.-J. und W. Schempp. *Boolean Methods in Interpolation and Approximation*. Pitman Research Notes in Mathematics Series 230. Longman Scientific & Technical, Harlow, 1989.

[Duc77] Duchon, J. *Splines Minimizing Rotation-Invariant Semi-Norms in Sobolev Spaces*. In W. Schempp und K. Zeller (Herausgeber), *Constructive Theory of Functions of Several Variables*, Nummer 571 in Lecture Notes in Mathematics, Seiten 85–100. Springer, 1977.

[EEHJ95] Eriksson, K., D. Estep, P. Hansbo und C. Johnson. *Introduction to adaptive methods for differential equations*. In *Acta Numerica, 1995*, Seiten 105–158. Cambridge Univ. Press, Cambridge, 1995.

[EHN96] Engl, H., M. Hanke und A. Neubauer. *Regularization of Inverse Problems*. Kluwer, 1996.

[EPP00] Evgeniou, T., M. Pontil und T. Poggio. *Regularization networks and support vector machines*. Advances in Computational Mathematics, 13:1–50, 2000.

[Fab09] Faber, G. *Über stetige Funktionen*. Mathematische Annalen, 66:81–94, 1909.

[FHP96] Frank, K., S. Heinrich und S. Pereverzev. *Information Complexity of Multivariate Fredholm Integral Equations in Sobolev Classes.* J. of Complexity, 12:17–34, 1996.

[FM01] Fung, G. und O. Mangasarian. *Proximal Support Vector Machine Classifiers*. In F. Provost und R. Srikant (Herausgeber), *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seiten 77–86. 2001.

[Fre42] Freudenthal, H. *Simplizialzerlegungen von beschränkter Flachheit*. Annals of Mathematics, 43:580–582, 1942.

[Fri91] Friedman, J. H. *Multivariate adaptive regression splines*. Ann. Statist., 19(1):1–141, 1991.

[FS97] Freund, Y. und R. Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting*. Journal of Computer and System Sciences, 55(1):119–139, 1997.

[Gar98] Garcke, J. *Berechnung von Eigenwerten der stationären Schrödingergleichung mit der Kombinationstechnik*. Diplomarbeit, Institut für Angewandte Mathematik, Rheinische Friedrich-Wilhelms-Universität Bonn, 1998.

[GG98] Gerstner, T. und M. Griebel. *Numerical Integration using Sparse Grids*. Numer. Algorithms, 18:209–232, 1998.

[GG00] Garcke, J. und M. Griebel. *On the computation of the eigenproblems of hydrogen and helium in strong magnetic and electric fields with the sparse grid combination technique*. Journal of Computational Physics, 165(2):694–716, 2000.

[GG01] Garcke, J. und M. Griebel. *On the parallelization of the sparse grid approach for data mining*. In S. Margenov, J. Wasniewski und P. Yalamov (Herausgeber), *Large-Scale Scientific Computations, Third International Conference, LSSC 2001, Sozopol, Bulgaria*, Band 2179 von *Lecture Notes in Computer Science*, Seiten 22–32. Springer, 2001.

[GG03] Gerstner, T. und M. Griebel. *Dimension–Adaptive Tensor–Product Quadrature*. Computing, 71(1):65–87, 2003.

[GH95] Griebel, M. und W. Huber. *Turbulence simulation on sparse grids using the combination method*. In N. Satofuka, J. Periaux und A. Ecer (Herausgeber), *Parallel Computational Fluid Dynamics, New Algorithms and Applications*, Seiten 75–84. North-Holland, Elsevier, 1995.

[GH03] Gradinaru, V. und R. Hiptmair. *Multigrid for discrete differential forms on sparse grids*. Computing, 71(1):17–42, 2003.

[GHSZ93] Griebel, M., W. Huber, T. Störtkuhl und C. Zenger. *On the parallel solution of 3D PDEs on a network of workstations and on vector computers*. In A. Bode und M. D. Cin (Herausgeber), *Parallel Computer Architectures: Theory, Hardware, Software, Applications*, Band 732 von *Lecture Notes in Computer Science*, Seiten 276–291. Springer Verlag, 1993.

[Gir98] Girosi, F. *An Equivalence Between Sparse Approximation and Support Vector Machines*. Neural Computation, 10(6):1455–1480, 1998.

[GJP93] Girosi, F., M. Jones und T. Poggio. *Priors, Stabilizers and Basis Functions: from regularization to radial, tensor and additive splines. A.I. Memo 1430*, Artificial Intelligence Laboratory, MIT, 1993.

[GJP95] Girosi, F., M. Jones und T. Poggio. *Regularization Theory and Neural Networks Architectures*. Neural Computation, 7:219–265, 1995.

[GK00] Griebel, M. und S. Knapek. *Optimized tensor-product approximation spaces*. Constructive Approximation, 16(4):525–540, 2000.

[GK03] Griebel, M. und F. Koster. *Multiscale Methods for the Simulation of Turbulent Flows*. In E. Hirschel (Herausgeber), *Numerical Flow Simulation III*, Band 82 von *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Seiten 203–214. Springer-Verlag, 2003.

[GO93] Golub, G. und M. Ortega. *Scientific Computing*. Academic Press, 1993.

[GO95] Griebel, M. und P. Oswald. *Tensor Product Type Subspace Splitting and Multilevel Iterative Methods for Anisotropic Problems*. Adv. Comput. Math., 4:171–206, 1995.

[GOS99] Griebel, M., P. Oswald und T. Schiekofer. *Sparse Grids for Boundary Integral Equations*. Numer. Mathematik, 83(2):279–312, 1999.

[Gri91] Griebel, M. *A parallelizable and vectorizable multi-level algorithm on sparse grids*. In W. Hackbusch (Herausgeber), *Parallel Algorithms for partial differential equations, Notes on Numerical Fluid Mechanics*, Band 31, Seiten 94–100. Vieweg Verlag, Braunschweig, 1991.

[Gri92] Griebel, M. *The combination technique for the sparse grid solution of PDEs on multiprocessor machines*. Parallel Processing Letters, 2(1):61–70, 1992.

[Gri98] Griebel, M. *Adaptive Sparse Grid Multilevel Methods for elliptic PDEs based on finite differences*. Computing, 61(2):151–179, 1998.

[GSZ92] Griebel, M., M. Schneider und C. Zenger. *A combination technique for the solution of sparse grid problems*. In P. de Groen und R. Beauwens (Herausgeber), *Iterative Methods in Linear Algebra*, Seiten 263–281. IMACS, Elsevier, North Holland, 1992.

[GT95] Griebel, M. und V. Thurner. *The efficient solution of fluid dynamics problems by the combination technique*. Int. J. Num. Meth. for Heat and Fluid Flow, 5(3):251–269, 1995.

[Han92] Hansen, P. C. *Analysis of discrete ill-posed problems by means of the L-curve*. SIAM Rev., 34(4):561–580, 1992.

[Han98] Hansen, P. *Rank-Deficient and Discrete Ill-Posed Problems*. SIAM, 1998.

[Han01] Hanke, M. *On Lanczos based methods for the regularization of discrete ill-posed problems*. BIT, 41:1008–1018, 2001.

[Han02] Handl, A. *Multivariate Analysemethoden*. Springer, 2002.

[Heg03] Hegland, M. *Adaptive sparse grids*. In K. Burrage und R. B. Sidje (Herausgeber), *Proc. of 10th Computational Techniques and Applications Conference CTAC-2001*, Band 44 von *ANZIAM J.*, Seiten C335–C353. 2003.

[Hem95] Hemker, P. *Sparse-Grid finite-volume multigrid for 3D-problems*. Advances in Computational Mathematics, 4:83–110, 1995.

[Heu97] Heusser, N. *Berechnung einer Dünngitterlösung der dreidimensionalen Lamé-Gleichung mit der Kombinationsmethode*. Diplomarbeit, Institut für Angewandte Mathematik, Rheinische Friedrich-Wilhelms-Universität Bonn, 1997.

[HHR00] Hegland, M., G. Hooker und S. Roberts. *Finite element thin plate splines in density estimation.* In *Proceedings of the 1999 International Conference on Computational Techniques and Applications (Canberra)*, Band 42, Seiten C712–C734. 2000.

[HKZ00] Hochmuth, R., S. Knapek und G. Zumbusch. *Tensor products of Sobolev spaces and applications. SFB Bericht 685*, `http://wissrech.ins.uni-bonn.de/research/pub/zumbusch/tensor.pdf`, SFB 256, Universität Bonn, 2000.

[HL92] Hoschek, J. und D. Lasser. *Grundlagen der goemetrischen Datenverarbeitung*, Kapitel 9. Teubner, 1992.

[Hoc99] Hochmuth, R. *Wavelet Bases in numerical Analysis and Restrictal Nonlinear Approximation.* Habilitation, Freie Universität Berlin, 1999.

[HTF01] Hastie, T., R. Tibshirani und J. Friedman. *The Elements of Statistical Learning.* Springer, 2001.

[Hub96] Huber, W. *Turbelenzsimulation mit der Kombinationsmethode auf Workstation-Netzen und Parallelrechnern.* Doktorarbeit, Institut für Informatik, Technische Universität München, 1996.

[HW00] Hickernell, F. J. und H. Woźniakowski. *Integration and approximation in arbitrary dimensions.* Adv. Comput. Math., 12(1):25–58, 2000.

[Joa98] Joachims, T. *Making Large-Scale SVM Learning Practical.* In B. Schölkopf, C. J. C. Burges und A. J. Smola (Herausgeber), *Advances in Kernel Methods – Support Vector Learning*, Seiten 169–184. MIT Press, Cambridge, USA, 1998.

[Kal00] Kaltenbacher, B. *Regularization by projection with a posteriori discretization level choice for linear and nonlinear ill-posed problems.* Inverse Problems, 16(5):1523–1539, 2000.

[Kau99] Kaufman, L. *Solving the quadratic programming problem arising in support vector classification.* In B. Schölkopf, C. J. C. Burges und A. J. Smola (Herausgeber), *Advances in Kernel Methods - Support Vector Learning*, Seiten 146–167. MIT Press, 1999.

[Kir96] Kirsch, A. *An Introduction to the Mathematical Theory of Inverse Problems.* Springer, 1996.

[KKH02] Karchin, R., K. Karplus und D. Haussler. *Classifying G-protein coupled receptors with support vector machines.* Bioinformatics, 18(1):147–159, 2002.

[Kna00] Knapek, S. *Approximation und Kompression mit Tensorprodukt-Multiskalenräumen.* Doktorarbeit, Rheinische Friedrich-Wilhelms-Universität Bonn, 2000.

[Kos02] Koster, F. *Multiskalen-basierte Finite Differenzen Verfahren auf adaptiven dünnen Gittern.* Doktorarbeit, Rheinische Friedrich-Wilhelms-Universität Bonn, 2002.

[Kra02] Kranz, C. J. *Untersuchungen zur Kombinationstechnik bei der numerischen Strömungssimulation auf versetzten Gittern.* Doktorarbeit, Institut für Informatik, Technische Universität München, 2002.

[Kuh60] Kuhn, H. W. *Some combinatorial lemmas in topology.* IBM J. Res. Develop., 4:518–524, 1960.

[KW71] Kimeldorf, G. und G. Wahba. *Some results on Tchebycheffian spline functions.* J. Math. Anal. Appl., 33:82–95, 1971.

[Lan61] Lanczos, C. *Linear differential operators.* D. Van Nostrand, Co., London, Seite 132, 1961.

[LLS95] Liem, C. B., T. Lü und T. M. Shih. *The Splitting Extrapolation Method.* World Scientific, Singapore, 1995.

[LMCP+00] Lopez-Molinero, A., A. Castro, J. Pino, J. Perez-Arantegui und J. R. Castillo. *Classification of ancient Roman glazed ceramics using the neural network of Self-Organizing Maps.* Fresenius J. Anal. Chem., 367(8):586–589, 2000.

[Lou89] Louis, A. K. *Inverse und schlecht gestellte Probleme.* Teubner, 1989.

[LP80] Locker, J. und P. M. Prenter. *Regularization with Differential Operators. I. General Theory.* J. Math. Anal. Appl., 75:504–529, 1980.

[Mel99] Melli, G. *Datgen: A program that creates structured data.* `http://www.datasetgenerator.com`, 1999.

[Mes62] Meschkowski, H. *Hilbertsche Räume mit Kernfunktion.* Die Grundlehren der mathematischen Wissenschaften, Bd. 113. Springer-Verlag, Berlin, 1962.

[Mit97] Mitchell, T. M. *Machine Learning.* McGraw-Hill, New York, 1997.

[MLH03] Meyer, D., F. Leisch und K. Hornik. *The support vector machine under test.* Neurocomputing, 55:169–186, 2003.

[MM01] Mangasarian, O. L. und D. R. Musicant. *Lagrangian support vector machines.* Journal of Machine Learning Research, 1:161–177, 2001.

[Mor84] Morozov, V. A. *Methods for solving incorrectly posed problems.* Springer-Verlag, New York, 1984.

[MST94] Michie, D., D. J. Spiegelhalter und C. C. Taylor (Herausgeber). *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994.

[Mus98] Musicant, D. R. *NDC: Normally Distributed Clustered Datasets*. `www.cs.wisc.edu/dmi/svm/ndc/`, 1998.

[Nat77] Natterer, F. *Regularisierung schlecht gestellter Probleme durch Projektionsverfahren*. Numer. Math., 28:329–341, 1977.

[Neu98] Neumaier, A. *Solving Ill-Conditioned and Singular Linear Systems: A Tutorial on Regularization*. SIAM Review, 40(3):636–666, 1998.

[NH00] Noordmans, J. und P. Hemker. *Application of an Adaptive Sparse Grid Technique to a Model Singular Perturbation Problem*. Computing, 65:357–378, 2000.

[OR00] Olshanskii, M. A. und A. Reusken. *On the Convergence of a Multigrid Method for Linear Reaction-Diffusion Problems. Bericht 192*, Institut für Geometrie und Praktische Mathematik, RWTH Aachen, 2000.

[OSP+92] Odewahn, S., E. Stockwell, R. Pennington, R. Humphreys und W. Zumach. *Automated star/galaxy discrimination with neural networks*. Astronomical Journal, 103(1):318–331, 1992.

[Osw91] Oswald, P. *Two remarks on multilevel preconditioners. Forschungsergebnisse Math-1-91*, FSU Jena, 1991.

[OSW03] Ong, C. S., A. Smola und R. Williamson. *Learning with Hyperkernels*. `http://mlg.anu.edu.au/~smola/papers/unpubOngSmoWil03.pdf`, 2003. Preprint.

[Phi62] Phillips, D. L. *A technique for the numerical solution of certain integral equations of the first kind*. J. Assoc. Comput. Mach., 9:84–97, 1962.

[Pla00] Plaskota, L. *The exponent of discrepancy of sparse grids is at least 2.1933*. Adv. Comput. Math., 12(1):3–24, 2000.

[PR99] Penny, W. D. und S. J. Roberts. *Bayesian neural networks for classification: how useful is the evidence framework ?* Neural Networks, 12:877–892, 1999.

[Pro04] Prothmann, H. *Verbraucher, die außen vor bleiben...*. Verbraucherzentrale Bundesverband, `www.die-kraft-der-verbraucher.de`, 2004.

[PS03] Poggio, T. und S. Smale. *The mathematics of learning: dealing with data*. Notices Amer. Math. Soc., 50(5):537–544, 2003.

[PV90] Plato, R. und G. Vainikko. *On the regularization of projection methods for solving ill-posed problems*. Numer. Math., 57(1):63–79, 1990.

[PZ99] Pflaum, C. und A. Zhou. *Error analysis of the combination technique*. Numer. Math., 84(2):327–350, 1999.

[Qua01] Quackenbush, J. *Computational analysis of microarray data.* Nat. Rev. Genet., 2(6):418–427, 2001.

[Rei04] Reisinger, C. *Numerische Methoden für hochdimensionale parabolische Gleichungen am Beispiel von Optionspreisaufgaben.* Doktorarbeit, Ruprecht-Karls-Universität Heidelberg, 2004. In Vorbereitung.

[RHA03] Roberts, S., M. Hegland und I. Altas. *Approximation of a thin plate spline smoother using continuous piecewise polynomial functions.* SIAM J. Numer. Anal., 41(1):208–234 (electronic), 2003.

[Rie97] Rieder, A. *A wavelet multilevel method for ill-posed problems stabilized by Tikhonov regularization.* Numer. Math., 75(4):501–522, 1997.

[Rip94] Ripley, B. D. *Neural networks and related methods for classification.* Journal of the Royal Statistical Society B, 56(3):409–456, 1994.

[RL87] Rousseeuw, P. J. und A. M. Leroy. *Robust regression and outlier detection.* Wiley, N.Y., 1987.

[RMC03] Rakotomamonjy, A., X. Mary und S. Canu. *Non Parametric regression with wavelet kernels.* `http://asi.insa-rouen.fr/~arakotom/publi/asmbi2003.pdf`, 2003. Preprint.

[ROM01] Rätsch, G., T. Onoda und K. R. Müller. *Soft margins for AdaBoost.* Machine Learning, 42(3):287–320, 2001.

[RYP03] Rifkin, R., G. Yeo und T. Poggio. *Regularized Least-Squares Classification.* In J. Suykens, G. Horvath, S. Basu, C. Micchelli und J. Vandewalle (Herausgeber), *Advances in Learning Theory: Methods, Models and Applications*, Band 190 von *NATO Science Series: Computer & Systems Sciences*, Seiten 131–153. IOS Press Amsterdam, 2003.

[Sai97] Saitoh, S. *Integral transforms, reproducing kernels and their applications.* Nummer 369 in Pitman research notes in mathematics series. Longman, 1997.

[Sch98] Schneider, R. *Multiskalen- und Wavelet-Matrixkompression.* Advances in Numerical Mathematics. Teubner, 1998.

[Sch99] Schiekofer, T. *Die Methode der Finiten Differenzen auf dünnen Gittern zur Lösung elliptischer und parabolischer partieller Differentialgleichungen.* Doktorarbeit, Rheinische Friedrich-Wilhelms-Universität Bonn, 1999.

[Sei80] Seidman, T. I. *Nonconvergence results for the application of least-squares estimation to ill-posed problems.* J. Optim. Theory Appl., 30:535–547, 1980.

[SGB+02] Suykens, J., T. V. Gestel, J. D. Brabanter, B. D. Moor und J. Vandewall. *Least Squares Support Vector Machines.* World Scientific Pub. Co., Singapore, 2002.

[Smo63] Smolyak, S. A. *Quadrature and interpolation formulas for Tensor Products of Certain Classes of Functions.* Dokl. Akad. Nauk SSSR, 148:1042–1043, 1963. Russian, Engl. Transl.: Soviet Math. Dokl. 4:240–243, 1963.

[SS99] Sickel, W. und F. Sprengel. *Interpolation on Sparse Grids and Nikol'skij–Besov spaces of dominating mixed smoothness.* J. Comput. Anal. Appl., 1:263–288, 1999.

[SS02] Schölkopf, B. und A. Smola. *Learning with Kernels.* MIT Press, 2002.

[SSM98] Smola, A., B. Schölkopf und K. Müller. *The connection between regularization operators and support vector kernels.* Neural Networks, 11:637–649, 1998.

[Stü99] Stüben, K. *Algebraic Multigrid (AMG): An Introduction with Applications. GMD Report 53*, GMD, 1999.

[ST03a] Schwab, C. und R. Todor. *Sparse finite elements for stochastic elliptic problems — higher order moments.* Computing, 71(1):43–63, 2003.

[ST03b] Schwab, C. und R.-A. Todor. *Sparse finite elements for elliptic problems with stochastic loading.* Numer. Math., 95(4):707–734, 2003.

[Sto74] Stone, M. *Cross-validatory choice and assessment of statistical predictions.* Journal of the Royal Statistical Society, 36:111–147, 1974.

[SU02] Strecker, U. und R. Uden. *Data mining of 3D poststack seismisc attribute volumes using Kohonen self-organizing maps.* The Leading Edge, 21(10):1032–1037, 2002.

[SV99] Suykens, J. A. K. und J. Vandewalle. *Least squares support vector machine classifiers.* Neural Processing Letters, 9(3):293–300, 1999.

[SW98] Sloan, I. H. und H. Woźniakowski. *When are quasi-Monte Carlo algorithms efficient for high-dimensional integrals?* J. Complexity, 14(1):1–33, 1998.

[SWHB89] Sigillito, V. G., S. P. Wing, L. V. Hutton und K. B. Baker. *Classification of radar returns from the ionosphere using neural networks. Technischer Bericht 10*, Johns Hopkins APL Technical Digest, 1989.

[SWW04] Sloan, I. H., X. Wang und H. Woźniakowski. *Finite-order weights imply tractability of multivariate integratin.* Journal of Complexity, 20:46–74, 2004.

[TA77] Tikhonov, A. N. und V. A. Arsenin. *Solutions of ill-posed problems.* W.H. Winston, Washington D.C., 1977.

[Tem89] Temlyakov, V. N. *Approximation of functions with bounded mixed derivative.* Proc. Steklov Inst. Math., 1, 1989.

[Tem93a] Temlyakov, V. N. *Approximation of Periodic Functions.* Nova Science, New York, 1993.

[Tem93b] Temlyakov, V. N. *On approximate recovery of functions with bounded mixed derivative.* J. Complexity, 9:41–59, 1993.

[Tik43] Tikhonov, A. N. *On the stability of inverse problems.* C. R. (Doklady) Acad. Sci. URSS, 39(5):176–179, 1943.

[Tik63] Tikhonov, A. N. *Solution of Incorrectly Formulated Problems and the Regularization Method.* Soviet Math. Dokl., 4:1035–1038, 1963.

[Vap82] Vapnik, V. N. *Estimation of dependences based on empirical data.* Springer-Verlag, Berlin, 1982.

[Vap98] Vapnik, V. N. *Statistical Learning Theory.* Wiley, 1998.

[Vap00] Vapnik, V. N. *The Nature of Statistical Learning Theory.* Springer, zweite Auflage, 2000.

[Ver96] Verfürth, R. *A review of a posteriori error estimation and adaptive mesh-refinement techniques.* Wiley/Teubner, Chichester/Stuttgart, 1996.

[vPS04] von Petersdorff, T. und C. Schwab. *Numerical Solution of Parabolic Equations in High Dimensions.* Mathematical Modeling and Numerical Analysis, 2004. To appear.

[VR02] Venables, W. N. und B. D. Ripley. *Modern applied statistics with S-Plus.* Statistics and Computing. Springer-Verlag, New York, 2002.

[Wah90] Wahba, G. *Spline models for observational data*, Band 59 von *Series in Applied Mathematics.* SIAM, Philadelphia, 1990.

[WF01] Witten, I. H. und E. Frank. *Data Mining - Praktische Werkzeuge und Techniken für das maschinelle Lernen.* Hanser Verlag, 2001.

[WW99] Wasilkowski, G. W. und H. Woźniakowski. *Weighted tensor product algorithms for linear multivariate problems.* J. Complexity, 15:402–447, 1999.

[Yse86] Yserentant, H. *On the Multi-Level Splitting of Finite Element Spaces.* Numerische Mathematik, 49:379–412, 1986.

[Yse92] Yserentant, H. *Hierarchical bases.* In J. R. E. O'Malley et al. (Herausgeber), *Proc. ICIAM'91.* SIAM, Philadelphia, 1992.

[Zen91] Zenger, C. *Sparse Grids.* In W. Hackbusch (Herausgeber), *Parallel Algorithms for Partial Differential Equations, Proceedings of the Sixth GAMM-Seminar, Kiel, 1990*, Band 31 von *Notes on Num. Fluid Mech.*, Seiten 241–251. Vieweg-Verlag, 1991.