Diplomarbeit

Waveletbasierte LU-Faktorisierung für elliptische Probleme

angefertigt am Institut für Angewandte Mathematik
vorgelegt der Math.-Nat. Fakultät der
Rheinischen Friedrich-Wilhelms-Universität Bonn

November 2000

von Daniel Schittko aus Bonn

Ich versichere, daß ich diese Arbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Bonn, den 27. November 2000

Inhaltsverzeichnis

1. Einleitung

Die Naturwissenschaften haben das Ziel, Vorgänge und Gesetzmäßigkeiten der Natur zu beschreiben und erforschen. Der traditionelle Ansatz ist, entweder aus Beobachtungen oder aus theoretischen Überlegungen heraus, ein Modell zu entwickeln, welches die natürlichen Prozesse möglichst genau darstellt. Solche Modelle lassen sich dann wiederum gezielt in praktischen Experimenten überprüfen und gegebenenfalls verbessern. Dies ist jedoch nicht immer möglich (zum Beispiel weil es zu teuer oder aufwendig wäre oder zu lange dauern würde) oder erwünscht (zum Beispiel weil es zu gefährlich wäre).

Einen Ausweg bietet hier seit einigen Jahren die numerische Simulation der Experimente auf dem Computer. Dabei wird ein im jeweiligen Fachgebiet entwickeltes Modell zur formalen mathematischen Beschreibung der natürlichen Vorgänge auf den Rechner übertragen, um dort getestet werden zu können. Auf diese Weise lassen sich Meßreihen schnell mit leicht veränderten Parametern oder Geometrien wiederholen, sodaß diese auf einfachem Wege optimierbar sind. Es ist außerdem im Gegensatz zu traditionellen Experimenten bei numerischen Simulationen möglich, eine sehr große Menge von Daten in kürzester Zeit zu erfassen. Weiterhin eröffnet die numerische Simulation einen zumindest näherungsweisen Zugang zu den Lösungen und zur Verifikation der zugrundeliegenden mathematischen Modelle.

Da viele Prozesse in der Natur durch Differentialgleichungen beschrieben werden, ist es innerhalb einer numerischen Simulation meist notwendig, Differentialgleichungen mit Hilfe eines Rechners zu lösen. Dazu müssen die kontinuierlichen Gleichungen zunächst einmal diskretisiert werden, zum Beispiel indem man sie auf den Punkten eines geeigneten endlichen Gitters betrachtet.

Auf diese Weise entsteht, eventuell nach einem weiteren Schritt, einer Linearisierung, als Kern der Simulationsschleife ein lineares Gleichungssystem, dessen Größe der Zahl der Diskretisierungspunkte entspricht. Die für realistische Simulationen benötigte hohe Auflösung des Raumes mit diskreten Punkten führt zu extrem hohen Anforderungen an Rechenzeit und Speicherplatz, insbesondere wenn zeitabhängige Vorgänge in drei Raumdimensionen simuliert werden sollen. Es ist meistens keine triviale Aufgabe, das entstehende Gleichungssystem effizient zu lösen. Die direkte Lösung mittels Gaußelimination, die zum Beispiel über eine vollständige Faktorisierung der Systemmatrix sowie Vorwärts- und Rückwärtssubstitutionen implementiert werden kann, benötigt im allgemeinen für ein Gleichungssystem mit n Unbekannten einen Aufwand, der kubisch mit n steigt, man spricht von einem Aufwand von

$\mathcal{O}(n^3)$ [23]. Leider überträgt sich dieses Verhalten auch auf die Lösungszeit.

Daß die Leistungssteigerung der zur Verfügung stehenden Hardware nicht in der Lage ist, dieses Problem auszugleichen, wird klar, wenn man bedenkt, daß die Taktrate der Prozessoren in den vergangenen 14 Jahren etwa um den Faktor 100 bis 150 gestiegen ist. Eine Leistungssteigerung um den Faktor 1000 würde es gerade einmal ermöglichen, ein Problem mit zehnmal mehr Unbekannten in der gleichen Zeit zu lösen. Betrachtet man zusätzlich zur Prozessorgeschwindigkeit auch die Verbesserung der unterstützenden Hardware angefangen von prozessornahem „Cache-“Speicher über die Busverarbeitungsbreite bis zur inzwischen relativ preiswerten Verkettung mehrerer Prozessoren zu einem leistungsfähigen Parallelrechner, erhält man natürlich eine wesentlich höhere Leistungssteigerung.

Da die wachsende Rechnerleistung stets von dem Wunsch, noch größere Probleme zu rechnen, begleitet wird, wird man auch durch den Einsatz von Parallelrechnern nicht davon entbunden, ein möglichst gutes Verfahren anzuwenden.

Diese Arbeit beschäftigt sich mit Verfahren zur Lösung großer schwachbesetzter linearer Gleichungssyteme, die von elliptischen partiellen Differentialgleichungen stammen. Hier wurden bislang verschiedene Ansätze verfolgt. Zunächst wurden anstelle der direkten Löser iterative Löser wie das Gauß-Seidel- oder das Jacobiverfahren entwickelt, die sich ausgehend von einer Startapproximation schrittweise der tatsächlichen Lösung nähern. Sie haben den Vorteil, daß sie einen geringen Rechenaufwand pro Iteration benötigen. Außerdem lassen sie sich ohne zusätzlichen Speicheraufwand implementieren. Leider steigt die Zahl der Iterationen, die sie zur Bestimmung einer Näherungslösung mit festem Fehler ε benötigen, bei Verfeinerung des zugrundeliegenden Gitters, so daß sie bei hinreichend feinen Gittern äußerst langsam sind. Dieser Nachteil konnte auch durch die Einführung von Varianten mittels Relaxationsparametern, wie etwa dem SOR-Verfahren, nicht entscheidend verbessert werden [55].

Zwar dämpfen die klassischen iterativen Verfahren hochfrequente Anteile des Fehlers zwischen Lösung und Iterierter stark, dafür aber niederfequente Anteile nur schwach. Das führt dazu, daß die Fehler nach einigen wenigen Iterationsschritten im wesentlichen nur noch niederfrequente Anteile besitzen, für deren Dämpfung dann viele weitere Iterationsschritte nötig sind. Die Überlegung, diese niederfrequenten Fehleranteile auf ein gröberes Gitter zu transportieren, um sie dort mittels eines Grobgitterproblems schneller reduzieren zu können, führte zur Entwicklung von Mehrgitterverfahren [12, 32, 33]. Mit ihnen entstanden erstmals Iterationsverfahren linearen Aufwandes, für die die Zahl der notwendigen Iterationsschritte zum Erreichen der Lösung bis auf Diskretisierungsgenauigkeit unabhängig von der Gitterweite ist. Solche Löser heißen auch optimal.

Leider sind Mehrgitterverfahren im allgemeinen schwer zu handhaben, weshalb sie sich bislang noch nicht in der Praxis auf breiter Ebene durchsetzen konnten.

Eine andere Alternative bieten moderne Krylovraum-Löser, wie beispielsweise das CG-, das GMRES- oder das BICGSTAB-Verfahren, zusammen mit geeigneten Vorkonditionierern [43].

Krylovraum-Methoden sind Projektions-Verfahren, die die Lösung des linearen Gleichungssystems iterativ dadurch bestimmen, daß das Residuum zur neuen Iterierten senkrecht zu einem Unterraum, dem sogenannten Krylovraum, ist, der durch die Folge der bisherigen Residuen definiert wird. Unterschiedliche (verallgemeinerte) Skalarprodukte führen dabei zu unterschiedlichen Verfahren [13, 43, 53]. Wesentlich für die Konvergenzeigenschaften von Krylovraum-Methoden ist – zumindest bei symmetrisch positiv definiten Problemen – die Gestalt des Spektrums der Systemmatrix A. So konvergiert zum Beispiel das CG-Verfahren schnell, wenn die Eigenwerte von A um eine Zahl herum konzentriert sind [26]. Für Gleichungssysteme mit nichtnormalen Matrizen spielen der numerische Wertebereich oder Pseudospektren eine entscheidende Rolle [26, 45, 46, 49].

Vorkonditionierung bedeutet eine Transformation des ursprünglichen Problems auf neue Koordinaten mittels einer Matrix M, des Vorkonditionierers. Anstelle der oft betrachteten Vorkonditionierung von links ist auch die Vorkonditionierung von rechts oder eine symmetrische Vorkonditionierung möglich. Für die Konvergenz des (links) vorkonditionierten Problems ist das (Pseudo-)Spektrum des entsprechend transformierten Operators MA verantwortlich. An Vorkonditionierer wird daher generell die Forderung gestellt, die Inverse der Operatormatrix A des zu lösenden Gleichungssystems $Ax = b$ möglichst genau zu approximieren. Eine wichtige Nebenbedingung ist, daß die Multiplikation $M \cdot b$ nicht zu aufwendig sein darf. Dies ist insofern kritisch, als daß die Inverse von A selbst bei dünnbesetztem A im allgemeinen nicht mehr schwachbesetzt ist.

Es gibt verschiedenste Vorkonditionierungstechniken, die oft wiederum auf klassischen Iterationsverfahren beruhen. Zu den gebräuchlichsten zählen die Vorkonditionierung mittels einiger Schritte des Jacobi- oder des SSOR-Verfahrens sowie mittels einer unvollständigen Faktorisierung (ILU) von A [31, 43]. In jüngster Zeit werden auch wieder dünnbesetzte approximative Inverse (SPAI) vorgeschlagen [29], da sie unter speziellen Nebenbedingungen hervorragende Parallelisierungseigenschaften aufweisen. Will man zur Lösung elliptischer Probleme mittels Krylovraum-Methoden ähnlich wie bei Mehrgitterverfahren gitterweitenunabhängige Konvergenzraten erreichen, so muß man dazu Multilevelvorkonditionierer [28, 38] einsetzen. Einer der bekanntesten Multilevelvorkonditionierer ist der BPX-Vorkonditionierer von Bramble, Pasciak und Xu [11].

Relativ neu ist die Idee, Wavelets für die Lösung von Gleichungssystemen zu benutzen. Die meisten Arbeiten beschäftigen sich hier mit der Lösung vollbesetzter Systeme, die durch die Diskretisierung von Integralgleichungen entstehen. Verschiedene Ansätze stammen von Beylkin [8, 9], Beylkin, Coifman und Rohklin [10], Coult [17], Dahmen [19], Gines [24], Gines, Beylkin und Dunn [25] sowie Schneider [44]. Im Unterschied zu Differentialgleichungen hat man bei Integralgleichungen als Hauptschwierigkeit das in der Regel volle Besetzungsmuster der nach Diskretisierung entstehenden Matrizen. Somit entsteht bereits für die gewöhnliche Matrix-Vektor-Multiplikation in diesem Falle ein Aufwand von $\mathcal{O}(n^2)$, wenn n die Anzahl der Freiheitsgrade bezeichnet, und ein optimaler Löser erscheint damit außer Reichweite.

Ausgehend von der Überlegung, daß die Matrix des zu lösenden Gleichungssystems eine Abbildung zwischen zwei Funktionenräumen repräsentiert, nutzen Waveletverfahren eine geschickte Zerlegung dieser Räume in Unterräume, die von den Dilaten und Translaten einer einzigen Funktion, nämlich dem Wavelet, aufgespannt werden. Dies führt zu einer einfachen und kostengünstigen Basistransformation ähnlich der schnellen Fouriertransformation [16]. Die schnelle Wavelettransformation allein ist jedoch für die Konstruktion eines optimalen Lösers noch nicht ausreichend. Darüberhinaus nutzen Waveletmethoden zur Lösung von Integralgleichungen die Kompressionseigenschaften von Waveletbasen. Genügt die Kernfunktion zum kontinuierlichen Integraloperator gewissen Glattheitsbedingungen, so fallen die Einträge der Darstellung der diskreten Operatormatrix in der Waveletbasis mit zunehmendem Abstand von der Diagonalen stark ab. Man kann sich daher beim Lösungsprozeß auf die Betrachtung gewisser Bänder innerhalb der Matrix beschränken, ohne wesentlich an Lösungsgenauigkeit zu verlieren. Zusammen mit der kostengünstigen Transformation führt dies zu optimalen ($\mathcal{O}(n)$ Aufwand) oder fast optimalen ($\mathcal{O}(n \log n)$ Aufwand) Lösern.

Es gibt aber auch Arbeiten, die sich mit der waveletbasierten Lösung von Differentialgleichungen beschäftigen, beispielsweise von Rieder [40], Rieder, Wells und Zhou [41], Rieder, Glowinski, Wells und Zhou [42], Vassilevski und Wang [50, 51], Xu [54] sowie Chan, Tang und Wan [14]. Letztere berechnen einen SPAI-Vorkonditionierer ausgehend von der Darstellung der Matrix A bezüglich einer gewählten Waveletbasis. Diese Darstellung bezeichnet man auch als *Standardform* von A [8]. Da die Inversen von Diskretisierungen elliptischer Differentialoperatoren auch als diskretisierte Integraloperatoren interpretiert werden können [34], sollte deren Approximation in Waveletkoordinaten und damit (modulo Transformation) die Konstruktion effektiver waveletbasierter Multilevelvorkonditionierer besser möglich sein als in der ursprünglichen Basis.

Die vorliegende Arbeit fußt im wesentlichen auf dem Ansatz von Gines, Beylkin und Dunn, die in [24, 25] einen direkten Wavelet-Multiskalen-Löser einführen. Obwohl sie hauptsächlich Integralgleichungen betrachten, kann ihre Methode ebenfalls zur Lösung von Differentialgleichungen eingesetzt werden. Ausgehend von einer orthogonalen Waveletbasis des $L^2(\mathbb{R})$ nutzt das vorgeschlagene Verfahren die Tatsache, daß Integraloperatoren nach Kompression eine schwachbesetzte Darstellung über die sogenannte *Nichtstandardform* besitzen [10].

Die Nichtstandardform ist im Gegensatz zur Standardform eine Erzeugendensystemdarstellung von A. Während das Aufstellen der Standardform selbst für dünnbesetzte Operatoren einen Aufwand von $\mathcal{O}(n \log n)$ erfordert, läßt sich die Nichtstandardform für solche Operatoren in $\mathcal{O}(n)$ berechnen, sodaß sie für die Konstruktion eines optimalen Lösers die geeignetere Darstellung ist.

Der in [24, 25] vorgestellte direkte Löser beruht auf einer (im Falle von Integraloperatoren komprimierten) Darstellung der Operatoren über ihre Nichtstandardform sowie Multiskalen-Vorwärts- und Rückwärtssubstitutionen. Im Verlauf einer rekursiven Faktorisierung werden dabei Bandapproximationen der entstehenden Schurkomplementoperatoren verwendet.

Die Breite der Bänder, innerhalb derer gerechnet wird, ist für alle Level gleich und wird zu Beginn anhand der beabsichtigten Lösungsgenauigkeit festgelegt. Da in den numerischen Beispielen in [24, 25] die Bandweiten stets sehr groß gewählt und zudem zur Validierung lediglich Rechenzeiten herangezogen werden, ist das tatsächliche Verhalten des Lösers nur schwer zu erkennen.

Ein zentrales Anliegen dieser Arbeit, in der es um waveletbasierte Löser für elliptische Differentialgleichungen geht, ist daher, das Konvergenzverhalten der Methode von Gines, Beylkin und Dunn sowie möglicher Varianten für den Fall eines Modellproblems genauer zu untersuchen. Dies geschieht sowohl für die (trivial zu lösende) eindimensionale als auch für die zweidimensionale Poissongleichnung. Die in dieser Arbeit gewählte Bandstrategie erweitert etwas abweichend von dem Originalverfahren auf jedem Level die betrachteten Bänder um eine feste Anzahl von Nachbarbändern. Alternativen, wie zum Beispiel eine levelabhängige Wahl, sind ebenfalls vorstellbar. In numerischen Experimenten wird durch ein systematisches Verbreitern der Rechenbänder der Übergang des Verfahrens von einem approximativen hin zu einem direkten Löser nachvollzogen. Daneben werden Varianten getestet, bei denen die Schurkomplementapproximation lediglich durch die im Verlauf der Konstruktion der Nichtstandardform entstehenden Grobgitteroperatoren erfolgt. Sie sind ähnlich zur Hierarchischen Basis Mehrgittermethode [6].

Ferner wird die Frage behandelt, ob auf diese Weise ebenfalls ein verläßlicher Multiskalen-Löser für singulär gestörte Probleme, wie zum Beispiel eine Konvektionsdiffusionsgleichung, möglich ist. Es zeigt sich, daß man mittels starr gewählter Waveletzerlegungen in den meisten betrachteten Fällen keine zufriedenstellenden Ergebnisse erhält. Verwendet man stattdessen problemangepaßte Zerlegungen, die ähnlich wie bei operator- und matrixabhängigen Mehrgitterverfahren konstruiert werden [1, 22, 27], so ergeben sich robustere Verfahren. Hier wird ein kurzer Ausblick gegeben.

Die Arbeit gliedert sich wie folgt: In Kapitel 2 werden die Grundbegriffe der Wavelettheorie erklärt, soweit sie für diese Arbeit erforderlich sind. Der Schwerpunkt liegt dabei auf der Nichtstandardform, aber auch auf die Standardform wird eingegangen. Außerdem werden die verwendeten Varianten des Verfahrens dargestellt. Kapitel 3 beschreibt das Konzept des für die erforderlichen Testreihen entwickelten C-Programmes, welche in Kapitel 4 ausgewertet werden. Abschließend erfolgt in Kapitel 5 die Zusammenfassung.

Folgenden Personen, die mich bei der Erstellung dieser Arbeit unterstützt haben, möchte ich danken: Herrn Prof. Dr. M. Griebel für Aufgabenstellung und Betreuung, Herrn Dipl.-Math. F. Kiefer, der für meine Anliegen immer ein offenes Ohr hatte, meinen Freunden, die Probleme beim Umgang mit der Programmiersprache C schnell lösen konnten, meinen Eltern.

2. Operatoren in Waveletbasen

In diesem Kapitel werden die Grundbegriffe der Wavelettheorie, wie sie in der vorliegenden Arbeit verwendet wird, eingeführt. Abschnitt 2.1 erläutert die Multi-Resolution-Analysis [18, 36, 37], anschließend wird die Besonderheit einer Waveletbasis dargestellt. In Abschnitt 2.2 werden darauf aufbauend die Zerlegungen eines Operators in Standard- und Nichtstandardform [10, 25] erklärt. Verschiedene Ansätze zur Lösung von Gleichungen im Nichtstandardsystem für ein- und zweidimensionale Probleme werden im Abschnitt 2.3 erläutert. Wie man diese Verfahren effizient einsetzen kann und warum, wird in Abschnitt 2.4 beschrieben [17]. In Abschnitt 2.5 wird bewiesen, warum diese Verfahren zumindest für eine bestimmte Klasse von Operatoren konvergent sein sollten. Der Beweis stützt sich auf [50].

2.1. Multi-Resolution-Analysis

Die Darstellung von Elementen eines Raumes ist abhängig vom gewählten Erzeugendensystem oder der verwendeten Basis. Eine solche Basis oder ein Erzeugendensystem sollte gewisse ideale Eigenschaften besitzen, die die resultierende Darstellung der betrachteten Elemente für die jeweilige Situation optimiert. Im vorliegenden Fall geht es um Funktionen des $L_2(\mathbb{R}^d)$, $d \in \mathbb{N}$, beziehungsweise des $L_2(\Omega)$, mit dem Gebiet $\Omega \subset \mathbb{R}^d$. Die ideale Eigenschaft der Basis soll die sein, daß sie sich aus den Dilaten und Translaten einer einzigen Funktion zusammensetzt, da sich dadurch der betrachtete $L_2(\mathbb{R}^d)$ in eine Folge von geschachtelten Unterräumen aufteilen läßt.

Bevor die Konstruktion einer solchen Basis näher beschrieben wird, ist es notwendig, den Begriff der Riesz-Basis durch die folgende Definition gegen den der Orthonormalbasis abzugrenzen.

Definition 2.1 (Riesz-Basis)

1. *Für zwei Zahlen $j, k \in \mathbb{R}$ ist definiert:*
$$j \sim k \Leftrightarrow \exists \text{ konst. } 0 < c_1, c_2 < \infty \text{ mit } c_1 j \leq k \leq c_2 j.$$
2. $\{\varphi(\cdot - k)_{k\in\mathbb{Z}^d}\}$ *bilden eine* Riesz-Basis *des Funktionenraumes V, wenn*
$$V = \overline{\text{span}}\,\{\varphi(\cdot - k)_{k\in\mathbb{Z}^d}\}$$

und

$$\|\sum_{k\in\mathbb{Z}^d} c_k\varphi(\cdot - k)\|_2 \sim \|c_k\|_{l^2} \quad \forall\{c_k\} \in l^2(\mathbb{Z}^d)$$

gelten. Die zweite Bedingung bezeichnet man als l^2-Stabilität.

Um die gewünschte ideale Basis zu erhalten, wird zunächst der zugrundeliegende Funktionenraum in eine Kette aus geschachtelten abgeschlossenen Unterräumen zerlegt. Besitzt einer der Unterräume eine Basis aus Translaten einer Funktion und ist mit einer Funktion deren Dilatation im nächstgröberen Unterraum enthalten, so bezeichnet man die Schachtelung als Multi-Resolution-Analysis [18, 36, 37]:

Definition 2.2 (Multi-Resolution-Analysis)
Eine Multi-Resolution-Analysis *des* $L_2(\mathbb{R}^d)$, $d \geq 1$, *ist eine Folge von geschachtelten abgeschlossenen Unterräumen*

$$\cdots \subset V_1 \subset V_0 \subset V_{-1} \subset \cdots \subset L_2(\mathbb{R}^d)$$

mit folgenden Eigenschaften:

1. *Die Schnittmenge aller dieser Unterräume enthält nur die Null:*

$$\bigcap_{j\in\mathbb{Z}} V_j = \{0\}.$$

2. *Die abgeschlossene Vereinigung dieser Unterräume ist der* $L_2(\mathbb{R}^d)$:

$$\overline{\bigcup_{j\in\mathbb{Z}}} V_j = L_2(\mathbb{R}^d).$$

3. *Es gilt*

$$f(\cdot) \in V_j \Longleftrightarrow f(2\cdot) \in V_{j-1}.$$

4. *Es gibt ein* $\varphi \in V_0$, *für das gilt:* $\{\varphi(\cdot - k)\}_{k\in\mathbb{Z}^d}$ *ist Riesz-/orthonormale Basis von* V_0. φ *heißt* Skalierungsfunktion.

Bemerkung 2.3 *Die Schachtelung in Unterräume ist nur unter gewissen Voraussetzungen möglich. Da die in dieser Arbeit betrachteten Fälle in diesem Zusammenhang unproblematisch sind, wird auf diese Thematik hier nicht weiter eingegangen.*

Die Idee besteht nun darin, diese Schachtelung der Räume V_j zur Konstruktion einer Riesz-/Orthonormalbasis für den $L_2(\mathbb{R}^d)$ zu verwenden, indem man die Basis von V_j zu einer von V_{j-1} ergänzt. Es ist also

$$V_{j-1} = W_j \oplus V_j,$$

wobei $\oplus$ die L_2-orthogonale Summe bezeichnet. Rekursives Fortsetzen führt zu der Darstellung

$$V_{j-1} = \bigoplus_{k=j}^{m} W_k \oplus V_m.$$

Für $k > m,\ (k, m \in \mathbb{Z})$, gilt $V_k \subset V_m$. Somit ist es erlaubt (und in der Praxis unausweichlich), einen Raum als gröbsten der betrachteten Räume festzulegen, da dieser alle gröberen Räume enthält. Dieser gröbste betrachtete Raum wird im folgenden als V_m bezeichnet. Dann läßt sich $L_2(\mathbb{R}^d)$ auch als

$$L_2(\mathbb{R}^d) = \bigoplus_{k=-\infty}^{m} W_k \oplus V_m$$

schreiben. Da jede Rechnung mit dem Computer in ihrer Speichernutzung beschränkt ist, wird ohne Einschränkung ein feinster Raum V_0 ausgewählt, so daß man eine Näherung

$$L_2(\mathbb{R}^d) \approx \bigoplus_{k=1}^{m} W_k \oplus V_m = V_0$$

erhält.

In der Anwendung wird oft statt $\mathbb{R}^d$ ein Gebiet $\Omega \subset \mathbb{R}^d$ betrachtet. Durch die Festlegung einer Anzahl von Gitterpunkten auf Ω bei der Diskretisierung wird zunächst der feinste Raum V_0 bestimmt. Die Wahl der Basis von V_0 legt auch den gröbstmöglichen Raum V_m fest.

Die Behandlung des Randes von Ω läßt sich auf zwei Weisen realisieren: entweder werden die Funktionen periodisch fortgesetzt oder durch Modifikationen angepaßt.

Nun ist es besonders wünschenswert, wenn nicht nur die Räume V_j, sondern auch die Räume W_j von den Dilaten und Translaten einer einzigen Basisfunktion aufgespannt werden, die kompakten Träger besitzt. Eine solche Basisfunktion von W_j nennt man, je nach Art der von ihr erzeugten Basis, Prewavelet oder Wavelet:

Definition 2.4 (Prewavelet/Wavelet)

- $\psi \in W_j$ *heißt* Prewavelet $\Longleftrightarrow \{\psi(\cdot - k)\}_{k\in\mathbb{Z}^d}$ *bilden eine Rieszbasis von* W_j.
- $\psi \in W_j$ *heißt* Wavelet $\Longleftrightarrow \{\psi(\cdot - k)\}_{k\in\mathbb{Z}^d}$ *bilden eine Orthonormalbasis von* W_j.

Das wohl einfachste Beispiel für eine Waveletbasis wird durch die Haar-Funktion [8, 30] definiert. Sie ist gegeben als

$$h(x) := \begin{cases} 1 & \text{für } 0 \le x < \frac{1}{2}, \\ -1 & \text{für } \frac{1}{2} \le x < 1, \\ 0 & \text{sonst.} \end{cases}$$

Die Funktionen

$$h_{j,k}(x) := \frac{1}{2^{\frac{j}{2}}} h\left(\frac{1}{2^j}x - k\right), \; j,k \in \mathbb{Z},$$

bilden die Haar-Basis. Als zugehörige Skalierungsfunktion wird die charakteristische Funktion auf dem Intervall $[0,1)$ verwendet.

Eine wichtige Eigenschaft von Wavelets ist die Zahl ihrer verschwindenden Momente. Sie gibt an, bis zu welchem Grad Polynome orthogonal zur Waveletfunktion sind.

Definition 2.5 (verschwindende Momente)
Eine Waveletfunktion $\psi(x)$ hat M verschwindende Momente, *wenn für $m = 0, \ldots, M-1$*

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

gilt.

Die Haar-Wavelets haben also genau ein verschwindendes Moment.

Die Wichtigkeit der Zahl der verschwindenden Momente beruht auf der Überlegung, daß es zur Kompression und Datenreduktion nützlich ist, wenn glatte Funktionen unter Wavelettransformation annähernd auf Null abgebildet werden.

Die Daubechies-Wavelets [18] wurden so konstruiert, daß sie mit wachsendem Index eine wachsende Anzahl von verschwindenden Momenten haben. So haben Daubechies2-Wavelets – die mit den Haar-Wavelets identisch sind – ein verschwindendes Moment, Daubechies4-Wavelets zwei, usw. Die Bezeichnung der Wavelets bezieht sich dabei auf die Zahl ihrer Filterkoeffizienten, die doppelt so hoch ist wie die Zahl der verschwindenden Momente. Man findet in der Literatur auch die andere Variante, bei der die Daubechies-Wavelets nach der Zahl ihrer verschwindenden Momente bezeichnet werden.

2.2. Darstellung von Operatoren in Waveletbasen

Gegeben sei ein Operator

$$T : L_2(\Omega) \to L_2(\Omega), \qquad \Omega \subset \mathbb{R}^d$$

und eine Multi-Resolution-Analysis mit einem feinsten Unterraum V_0 und einem gröbsten Unterraum V_m. Des weiteren seien für $j,k \in \{0,1,\ldots,m\}$ die Abbildungen

$$P_j^k : V_j \to V_k$$

und

$$Q_j^k : \begin{cases} V_j & \to & W_k & \text{für } k > j \\ W_j & \to & V_k & \text{für } k < j \end{cases}$$

gegeben, die im folgenden mit ihrer Matrixdarstellung identifiziert werden. Anhand dieser Restriktionen beziehungsweise Prolongationen werden in Abschnitt 2.2.1 die Standardform und in Abschnitt 2.2.2 die Nichtstandardform des Operators T erläutert [10, 25]. Da man zur Berechnung beider im allgemeinen einen diskretisierten Operator benötigt, wird T zunächst zu T_0 diskretisiert:

$$T_0 : V_0 \to V_0.$$

2.2.1. Standardform

Die Standardform ist die Darstellung des Operators T_0 bezüglich der Zerlegung des Raumes V_0 in

$$V_0 = \bigoplus_{j=1}^{m} W_j \oplus V_m.$$

Sie wird wie folgt aus T_0 berechnet: Zunächst wird T_0 in vier Teiloperatoren gespalten:

$$\begin{aligned} A_1 &:= Q_1^{0\,T} T_0 Q_1^0 : & W_1 \to W_1, \\ B_1 &:= Q_1^{0\,T} T_0 P_1^0 : & V_1 \to W_1, \\ C_1 &:= P_1^{0\,T} T_0 Q_1^0 : & W_1 \to V_1, \\ T_1 &:= P_1^{0\,T} T_0 P_1^0 : & V_1 \to V_1. \end{aligned}$$

Mittels der Zweiskalentransformation

$$\mathcal{W}_1^0 := (Q_1^0 | P_1^0)$$

und deren Transponierter läßt sich dies auch als

$$\mathcal{W}_1^{0\,T} T_0 \mathcal{W}_1^0 = \begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 \end{pmatrix}$$

ausdrücken. Es mag verwundern, daß von links nicht mit der Inversen, sondern mit der Transponierten multipliziert wird. Betrachtet man das sich über die schwache Form ergebende lineare Gleichungssystem $T_0 x_0 = f_0$, so bildet die Steifigkeitsmatrix T_0 als Darstellung des diskreten Operators nach Darstellungen in Momentendarstellung ab. Die transponierte Transformationsmatrix ist also notwendig, um von der Momentendarstellung in eine ebensolche bezüglich der Waveletbasis zu gelangen.

Man bezeichnet die Anwendung von $\mathcal{W}_j^{j-1}$ oder $\mathcal{W}_j^{j-1^T}$ als Zweiskalentransformation, da sie von Skala $j-1$ nach Skala j abbilden. Genauso kann man auch von einer ein Level tiefen *(Pre-)Wavelettransformation* sprechen. Auf die Vorsilbe „Pre-“ wird aus Gründen der Lesbarkeit im folgenden verzichtet, soweit dies der jeweilige Zusammenhang erlaubt. Die Transformationsmatrizen $\mathcal{W}_j^{j-1}$ und $\mathcal{W}_j^{j-1^T}$ sind für jedes Level unterschiedlich dimensioniert, daher werden sie hier indiziert. Es gibt (mindestens) drei verschiedene Möglichkeiten, eine Wavelettransformation durchzuführen: die algebraische Transformation, die direkte algebraische Transformation und die Transformationen über einen Finite-Elemente-Ansatz. Selbstverständlich lassen sich alle drei Methoden sowohl bei der Berechnung der Standard- als auch der Nichtstandardform (unabhängig von der Anzahl der Dimensionen) einsetzen.

Die algebraische Transformation entspricht genau dem oben angedeuteten Vorgehen: man stellt Transformationsmatrizen auf und multipliziert diese mit dem diskretisierten Operator. Das Abspeichern der Transformationsmatrizen führt zwar zu einem unnötig hohen Speicherverbrauch, dafür ist die Transformation über dünne Matrix-Matrix-Multiplikation möglich, sodaß diese Variante effizient implementiert werden kann. Benutzt man randangepaßte Wavelets, erfordert die Randanpassung nur beim Aufstellen der Transformationsmatrizen Beachtung.

Bei der direkten algebraischen Transformation verzichtet man auf das Aufstellen und Speichern von Transformationsmatrizen und führt stattdessen die Transformation direkt auf der Operatormatrix durch. Ein Nachteil besteht darin, daß bei randangepaßten Wavelets eine Sonderbearbeitung der Ränder notwendig wird.

Beylkin [9, 10] verwendet die dritte Variante, indem er direkt die transformierten Teilmatrizen A_1, B_1, C_1 und T_1 durch Verwendung der Wavelet- und Skalierungsfunktionenbasen als Ansatz- und Testfunktionen in einer Finite-Elemente-Diskretisierung aufstellt.

Wegen $V_1 = V_2 \oplus W_2$ lassen sich B_1, C_1 und T_1 weiter zerlegen. Für T_1 verläuft die weitere Zerlegung genauso wie die von T_0. B_1 wird aufgespalten in einen Anteil

$$B_1^2 := B_1 Q_2^1 : W_2 \to W_1$$

und einen

$$\tilde{B}_1^2 := B_1 P_2^1 : V_2 \to W_1.$$

Letzterer wird solange weiterzerlegt, bis man zu einem Anteil

$$B_1^{m+1} : V_m \to W_1$$

gelangt. Entsprechend lassen sich Anteile C_2^1 bis C_{m+1}^1 erzeugen.

Auch die bei der weiteren Zerlegung von T_1 entstehenden Anteile B_2 und C_2 werden weiter aufgespalten. Das Ergebnis dieser Prozedur bezeichnet man als *Standardform.* Abbildung 2.1 zeigt anhand eines drei Level tief zerlegten Operators T_0, wie eine eindimensionale Standardform als Matrix aufgebaut ist. Für die zweidimensionale Standardform gibt es zwei mögliche

$$
\left[\begin{array}{c|c|c|c}
A_1 & B_1^2 & B_1^3 & B_1^4 \\ \hline
C_2^1 & A_2 & B_2^3 & B_2^4 \\ \hline
C_3^1 & C_3^2 & A_3 & B_3^4 \\ \hline
C_4^1 & C_4^2 & C_4^3 & T_3
\end{array}\right]
\bullet
\left[\begin{array}{c}
w_1 \\ \hline w_2 \\ \hline w_3 \\ \hline v_3
\end{array}\right]
=
\left[\begin{array}{c}
\hat{w}_1 \\ \hline \hat{w}_2 \\ \hline \hat{w}_3 \\ \hline \hat{v}_3
\end{array}\right]
$$

Abbildung 2.1.: Skizze einer Standardformmultiplikation eines eindimensional zerlegten Operators.

Extreme. Ausgehend von der Zerlegung des zweidimensionalen Raumes V_j in $V_j^x \otimes V_j^y$ kann man entweder nur dann B- oder C-Teile weiterzerlegen, wenn sie von einem oder in einen Raum der Form $V_j^x \otimes V_j^y$ abbilden, oder zusätzlich, sobald auch nur einer der Faktoren aus diesem Produkt ein Raum $V_j^{(x \text{ oder } y)}$ ist. Auch Zwischenformen (zum Beispiel nur in x-Richtung zerlegen) sind denkbar.

Abbildung 2.3 zeigt ein Beispiel für eine nach der zweiten Methode berechnete zweidimensionale Standardform. Sie besitzt die gleiche Grobstruktur wie die eindimensionale Standardform, und ihre einzelnen Blöcke weisen ebenfalls eine eindimensionale Standardformstruktur auf.

Die eindimensionale Standardform einer Diagonalbandmatrix besitzt eine Fingerbandstruktur, wie in Abbildung 2.2 anhand des Laplaceoperators dargestellt wird. Entsprechend hat die zweidimensionale Standardform eine von der gewählten Zerlegungsvariante abhängige Multifingerbandstruktur (Abbildung 2.3).

Es soll nun überlegt werden, in welcher Komplexität sich eine eindimensionale Standardform einer Matrix in Bandgestalt mit $\mathcal{O}(n),\ n = 2^{lz}$ Einträgen aufstellen läßt. Hierbei steht die „Levelzahl" lz für die maximal mögliche Zerlegungstiefe. Es sei weiter vorausgesetzt, daß so weit wie möglich zerlegt wird. Dann gibt es genau lz Teilmatrizen $B_1^k,\ k = 2, \ldots lz+1$. Mit den Teilmatrizen $C_k^1,\ k = 2, \ldots lz + 1$ und A_1 erhält man $2lz + 1$ Teilmatrizen mit jeweils $\mathcal{O}(n)$ Einträgen(, da die Bandgestalt sich im allgemeinen auf die Teilmatrizen überträgt). Zusammen sind das $\mathcal{O}((2lz+1)n)$ Einträge. Wegen $n = 2^{lz}$ gilt $lz = \log_2(n)$, sodaß man auf $\mathcal{O}(n \log_2(n))$ Einträge alleine für diese Teilmatrizen kommt. Für das zweite Level kommen folglich $\mathcal{O}(\frac{n}{2} \log_2(\frac{n}{2}))$ Einträge hinzu, für das dritte $\mathcal{O}(\frac{n}{4} \log_2(\frac{n}{4}))$, usw. Daraus resultiert die Erkenntnis, daß man zum Aufstellen der vollen Standardform selbst für eine Matrix in

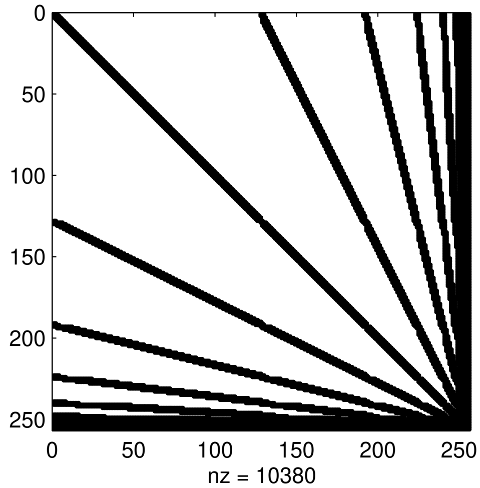

Abbildung 2.2.: Besetzungsmuster einer eindimensionalen Standardform am Beispiel eines mit Daubechies4-Wavelets transformierten Laplaceoperators, gezeigt sind alle Einträge vom Betrag größer 10^{-7}

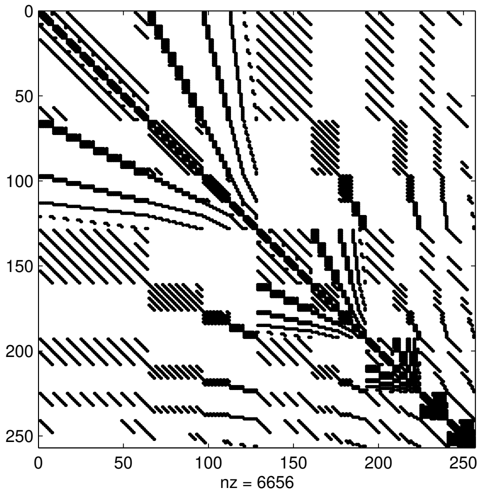

Abbildung 2.3.: Besetzungsmuster einer zweidimensionalen Standardform am Beispiel eines mit Daubechies4-Wavelets transformierten Laplaceoperators, gezeigt sind alle Einträge vom Betrag größer 10^{-7}

Bandgestalt bereits einen Rechenaufwand von $\mathcal{O}(n \log_2(n))$ benötigt. Für das gesuchte Verfahren der Komplexität $\mathcal{O}(n)$ ist die Standardform folglich nicht geeignet. Besser ist es mit der Nichtstandardform, die sich in $\mathcal{O}(n)$ Operationen berechnen läßt.

2.2.2. Nichtstandardform

Die Nichtstandardform ist die Darstellung des Operators T_0 bezüglich der Zerlegung jedes einzelnen Teilraumes V_j in

$$V_j = W_{j+1} \oplus V_{j+1}.$$

Im Gegensatz zur Standardform ist die Nichtstandardform keine Basisdarstellung, sondern eine Erzeugendensystemdarstellung. Dies wird insbesondere an Abbildung 2.4 deutlich. Die Darstellung eines zur Nichtstandardform gehörenden Vektors enthält mehr Information, als zur Rekonstruktion des ursprünglichen Vektors notwendig ist. Dies erklärt auch die Namen „Standardform" und „Nichtstandardform".

Der Unterschied zwischen der Berechnung von Standardform und Nichtstandardform ist der, daß bei der Nichtstandardform die Teilmatrizen B_j und C_j nicht weiter zerlegt werden. Daher ist es im Eindimensionalen auch relativ einfach, die eine Form aus der anderen zu berechnen. Für die zweidimensionalen Formen ist dies nur dann kostengünstig möglich, wenn man bei der Standardform nur nach ganzen Räumen $V_j = V_j^x \otimes V_j^y$ zerlegt. Zerlegt man nach einzelnen Richtungen, müßte man zuerst bis zum ersten Level rekonstruieren und anschließend in die andere Form zerlegen.

Wie bei der Standardform wird der Operator T_0 zunächst in vier Teile zerlegt:

$$\mathcal{W}_1^{0\,T} T_0 \mathcal{W}_1^0 = \begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 \end{pmatrix}.$$

Anschließend geschieht das gleiche mit T_1, dann mit T_2, usw., bis die gewünschte oder die maximale Zerlegungstiefe erreicht ist. Das Ergebnbis dieser Prozedur ist dann die Nichtstandardform.

Eine Wavelet- beziehungsweise Zweiskalenrekonstruktion von T_{j-1} läßt sich entsprechend der Zweiskalentransformation als

$$T_{j-1} = \mathcal{W}_j^{j-1} \begin{pmatrix} A_j & B_j \\ C_j & T_j \end{pmatrix} \mathcal{W}_j^{j-1\,T}$$

schreiben. Mit Prewavelets ist diese Rekonstruktion aufgrund der fehlenden Orthogonalität nicht durchführbar. Während für Wavelets

$${\mathcal{W}_{j+1}^{j}}^T \cdot \mathcal{W}_{j+1}^{j} = Id$$

gilt, ist dies bei Prewavelets nicht der Fall. Hier müßte man statt der Transponierten die Inverse verwenden. Im Gegensatz zur Transponierten ist die Inverse bei Prewavelets jedoch im

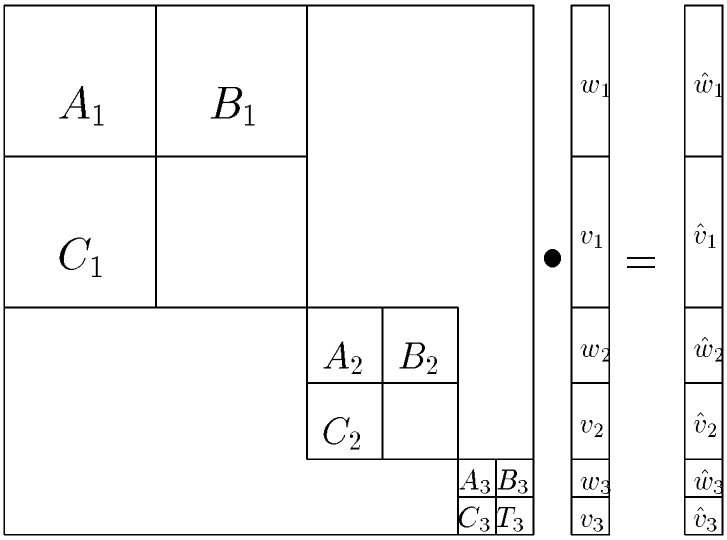

Abbildung 2.4.: Skizze einer drei Level tiefen Nichtstandardformmultiplikation. Da T_1 weiter zerlegt wird, ist der Block, in dem sonst T_1 stehen würde, leer.

allgemeinen vollbesetzt, daher erforderte auch ihre Anwendung auf die Nichtstandardform zwecks Rekonstruktion einen zu hohen Aufwand. Da die in den folgenden Kapiteln beschriebenen Algorithmen keine solche Rekonstruktion benötigen, ist dies jedoch nicht weiter problematisch.

2.2.3. Zweidimensionale Nichtstandardform

Eine Einführung in die zweidimensionale Nichtstandardform ausgehend von einem Tensorproduktansatz gibt Gines in [24]. Wesentlich ist, daß der Raum V_{j-1} jetzt in $V_{j-1}^x \otimes V_{j-1}^y$ mit

$$\begin{aligned} V_{j-1}^x &= V_j^x \oplus W_j^x \qquad \text{und} \\ V_{j-1}^y &= V_j^y \oplus W_j^y \end{aligned}$$

aufgespalten werden kann. Damit ergibt sich

$$\begin{aligned} V_{j-1} &= V_{j-1}^x \otimes V_{j-1}^y \\ &= (V_j^x \otimes V_j^y) \oplus (V_j^x \otimes W_j^y) \oplus (W_j^x \otimes V_j^y) \oplus (W_j^x \otimes W_j^y). \end{aligned}$$

Also läßt sich V_{j-1} in vier Teilräume zerlegen, sodaß T_{j-1} in die folgenden 16 Abbildungen zerfällt:

$$\begin{array}{rclcrcl}
Aa_j & := & (Q_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(Q_j^{j-1})_y & : & W_j^x \otimes W_j^y & \to & W_j^x \otimes W_j^y, \\
Ab_j & := & (Q_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(Q_j^{j-1})_y & : & V_j^x \otimes W_j^y & \to & W_j^x \otimes W_j^y, \\
Ba_j & := & (Q_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(P_j^{j-1})_y & : & W_j^x \otimes V_j^y & \to & W_j^x \otimes W_j^y, \\
Bb_j & := & (Q_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(P_j^{j-1})_y & : & V_j^x \otimes V_j^y & \to & W_j^x \otimes W_j^y, \\
Ac_j & := & (Q_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(Q_j^{j-1})_y & : & W_j^x \otimes W_j^y & \to & V_j^x \otimes W_j^y, \\
At_j & := & (Q_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(Q_j^{j-1})_y & : & V_j^x \otimes W_j^y & \to & V_j^x \otimes W_j^y, \\
Bc_j & := & (Q_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(P_j^{j-1})_y & : & W_j^x \otimes V_j^y & \to & V_j^x \otimes W_j^y, \\
Bt_j & := & (Q_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(P_j^{j-1})_y & : & V_j^x \otimes V_j^y & \to & V_j^x \otimes W_j^y, \\
Ca_j & := & (P_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(Q_j^{j-1})_y & : & W_j^x \otimes W_j^y & \to & W_j^x \otimes V_j^y, \\
Cb_j & := & (P_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(Q_j^{j-1})_y & : & V_j^x \otimes W_j^y & \to & W_j^x \otimes V_j^y, \\
Ta_j & := & (P_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(P_j^{j-1})_y & : & W_j^x \otimes V_j^y & \to & W_j^x \otimes V_j^y, \\
Tb_j & := & (P_j^{j-1})_y^T(Q_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(P_j^{j-1})_y & : & W_j^x \otimes V_j^y & \to & V_j^x \otimes V_j^y, \\
Cc_j & := & (P_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(Q_j^{j-1})_y & : & W_j^x \otimes W_j^y & \to & V_j^x \otimes V_j^y, \\
Ct_j & := & (P_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(Q_j^{j-1})_y & : & V_j^x \otimes W_j^y & \to & V_j^x \otimes V_j^y, \\
Tc_j & := & (P_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(Q_j^{j-1})_x(P_j^{j-1})_y & : & W_j^x \otimes V_j^y & \to & V_j^x \otimes V_j^y, \\
Tt_j & := & (P_j^{j-1})_y^T(P_j^{j-1})_x^T T_{j-1}(P_j^{j-1})_x(P_j^{j-1})_y & : & V_j^x \otimes V_j^y & \to & V_j^x \otimes V_j^y.
\end{array} \tag{2.1}$$

Eine davon, nämlich $Tt_j : V_j \to V_j$, ist die, welche nun weiter zerlegt wird. Entsprechend der zweidimensionalen Standardform besitzt auch die zweidimensionale Nichtstandardform eine Grob- und eine Fein-Nichtstandardform-Struktur (siehe Abbildung 2.5).

Die Notwendigkeit einer zweidimensionalen Nichtstandardform entsteht aus der Überlegung, daß die physikalischen Eigenschaften einer zweidimensionalen Gleichung von einem Löser, der die Zweidimensionalität berücksichtigt, besser erhalten werden können als von einem Löser für eindimensionale Probleme. Hinzu kommt, daß der Versuch, eine zweidimensionale Gleichung mit n Unbekannten mittels des eindimensionalen Verfahrens zu lösen, zwangsläufig zu einem $\mathcal{O}(n^{\frac{3}{2}})$-Verfahren führt, wie unten näher erläutert wird.

Zum Aufstellen eines zweidimensionalen Operators ist zunächst die physikalische Gleichung zu diskretisieren. Dazu wird das zweidimensionale Gebiet in ein Gitter mit l Punkten in jeder Richtung unterteilt, sodaß es insgesamt $l^2 = n$ Punkte aufweist. Auf dieses Gitter wird ein Diskretisierungsstern aufgesetzt, dessen Auswertung zu einer Matrix führt. Da hierbei jeder Gitterpunkt mit jedem verknüpft wird, besitzt die sich ergebende Matrix a priori $n^2 = l^4$ Einträge. (Bei partiellen Differentialgleichungen sind oft viele dieser Einträge null, weil weit auseinanderliegende Gitterpunkte nicht miteinander koppeln.)

Während bei der Berechnung der eindimensionalen Nichtstandardform eine Ausgangsmatrix

Aa	Ab	Ba	Bb
Ac	At	Bc	Bt
Ca	Cb	Ta	Tb
Cc	Ct	Tc	Tt

Abbildung 2.5.: Aufbau einer ein Level tiefen 2D-Nichtstandardform.

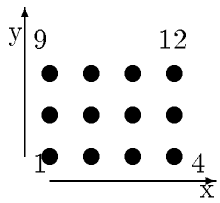

Abbildung 2.6.: Zweidimensionales Gitter, lexikographisch numeriert.

der Größe $2^k \times 2^k$ sinnvoll ist, sollte die Ausgangsmatrix für die zweidimensionale Nichtstandardform eine Größe von $4^k \times 4^k$ besitzen. Das entspricht einem Gitter der Größe $2^k \times 2^k$.

Die Numerierung der Gitterpunkte spielt für das Besetzungsmuster des diskretisierten Operators eine wesentliche Rolle. Eine Möglichkeit ist, sie aufsteigend zuerst in x- und dann in y-Richtung (lexikographisch) zu numerieren, wie es in Abbildung 2.6 dargestellt ist. Als Alternative kann man auch zuerst in y-Richtung sortieren. Darüberhinaus gibt es weitere Möglichkeiten, die jedoch im Rahmen dieser Arbeit nicht weiter betrachtet wurden.

Wählt man die lexikographische Numerierung (Abbildung 2.6), so zerfällt die sich ergebende Matrix in Blöcke, von denen der erste (oben links in der Matrix) die Kopplungen der ersten Zeile des Gitters mit der ersten Zeile enthält, der zweite (rechts daneben) die der ersten Zeile mit der zweiten, und so weiter (vgl. Abbildung 2.7). Numeriert man die Gitterpunkte zuerst in y-Richtung, so ergeben sich entsprechend Blöcke mit Spaltenkopplungen.

Die entstandene Matrix besteht also aus Blöcken, die (zum Beispiel) Kopplungen in x-Richtung beinhalten. Die zweidimensionale Wavelettransformation der Matrix, welche im

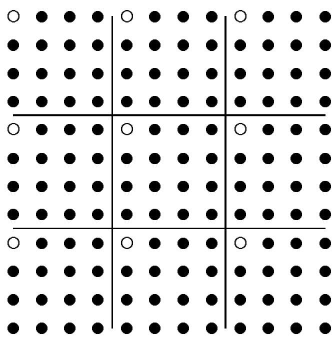

Abbildung 2.7.: Die sich ergebende Matrix zerfällt in Blöcke mit Zeilen-/Spaltenkopplungen. Nicht ausgefüllte Kreise verdeutlichen Kopplungen zwischen den Gitterpunkten 1, 5 und 9.

folgenden anhand der direkten algebraischen Transformation näher erläutert wird, erfordert allerdings sowohl eine eindimensionale Transformation in x- als auch in y-Richtung. Daher wird zuerst eine Transformation der einzelnen Blöcke durchgeführt. Anschließend ist die Transformation in die andere Richtung durchzuführen. Hierfür kann die Matrix so umsortiert werden, daß Blöcke entstehen, die Kopplungen in y-Richtung enthalten. Stattdessen könnte man auch die jeweils benötigten Elemente heraussuchen, in einem externen Block speichern und diesen transformieren, bevor man die Elemente anschließend zurückschreibt. Da man beim Zurückschreiben jedes Element einzeln bearbeitet, kann man die Umsortierung gleichzeitig durchführen, indem man die Elemente in einen zusätzlich angelegten Speicher bereits umsortiert einträgt und sie anschließend für die zweite Transformation blockweise wieder ausliest. Die Umsortierung läßt sich alternativ dazu auch in den Ausleseprozeß einbinden. Wichtig ist, daß sie auf diese Weise keinen zusätzlichen Rechenaufwand verursacht.

Es ist keineswegs offensichtlich, daß sich die zweidimensionale Nichtstandardform beispielsweise eines Differentialoperators mit einem Aufwand von $\mathcal{O}(n)$ berechnen läßt, wenn n die Anzahl der Freiheitsgrade beziehungsweise Gitterpunkte ist. Geht man also davon aus, daß die Ausgangsmatrix $n \times n$ Einträge besitzt, so besteht jeder Block aus $\sqrt{n} \times \sqrt{n}$ Einträgen. (Es wird vorausgesetzt, daß ein Gitter mit gleich vielen Punkten in x- und in y-Richtung gewählt wurde. Die nachfolgenden Betrachtungen sind allerdings auch für den Fall möglich, daß das Gitter in einer Richtung mehr Punkte aufweist als in der anderen.) Da die Blöcke (in vielen Fällen, insbesondere bei Differentialoperatoren) ein Diagonalbandbesetzungsmuster aufweisen, lassen sie sich jeweils mit $\mathcal{O}(\sqrt{n})$ Operationen transformieren. Von diesen Blöcken besitzt die Matrix zwar n Stück, aber nicht alle müssen transformiert werden. Nur die Blöcke um die Diagonalblöcke herum sind zu bearbeiten. Von den eigentlich n Blöcken sind also nur auf $\mathcal{O}(\sqrt{n})$ Transformationen auszuführen. Zusammen sind das $\mathcal{O}(n)$ Operationen für die

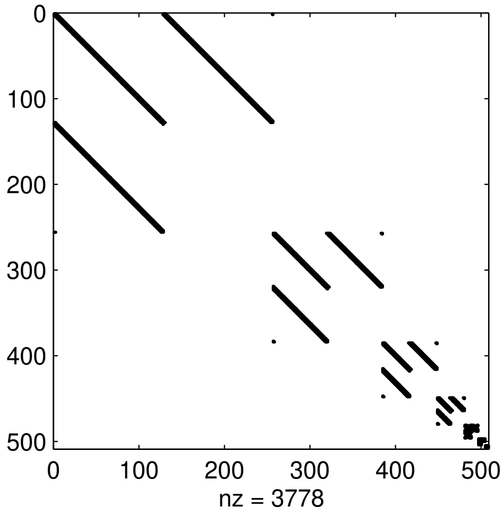

Abbildung 2.8.: Besetzungsmuster einer eindimensionalen Nichtstandardform am Beispiel eines mit Daubechies4-Wavelets transformierten Laplaceoperators, gezeigt sind alle Einträge vom Betrag größer 10^{-7}

Transformation der gesamten Matrix in die eine Richtung. Da die Umsortierung bei geschickter Programmierung keine zusätzliche Rechenzeit in Anspruch nimmt, die Transformation in die andere Richtung folglich ebenfalls in $\mathcal{O}(n)$ durchführbar ist und die anschließend notwendige Einsortierung ebenfalls in $\mathcal{O}(n)$ funktioniert, läßt sich die gesamte Transformation der Ausgangsmatrix in $\mathcal{O}(n)$ ausführen.

Es wurde bereits oben angedeutet, daß ein eindimensionales Verfahren bei zweidimensionalen Nichtstandardformmatrizen zu einem Aufwand von (mindestens) $\mathcal{O}(n^{\frac{3}{2}})$ führen würde. Eine eindimensionale Behandlung der Matrix bedeutet nämlich, daß man zwar ein Band von Blöcken betrachtet, jedoch jeden dieser Blöcke fälschlicherweise als vollbesetzt annimmt. Auf dem Band befinden sich also $\mathcal{O}(\sqrt{n})$ zu bearbeitende (transformierende) Blöcke, die jeweils n Elemente beinhalten. Zusammen ergibt sich für das gesamte Band aus „vollständig besetzten" Blöcken ein Aufwand von $\mathcal{O}(n^{\frac{3}{2}})$ bereits für die Transformation der Matrix in die Nichtstandardform. Somit kommt diese sicherlich in manchen Punkten einfacher zu handhabende Variante für praktische Zwecke nicht in Frage.

Wie bei der Standardform weist auch die zweidimensionale Nichtstandardform das entsprechende Metabesetzungsmuster der eindimensionalen Form auf. Die Abbildungen 2.8 und 2.9 verdeutlichen dies.

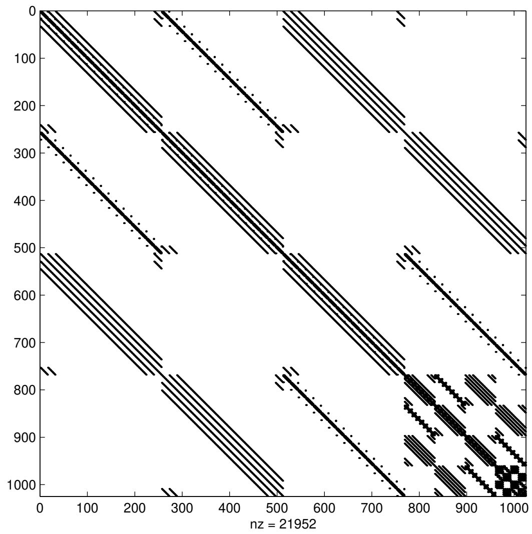

Abbildung 2.9.: Besetzungsmuster einer zweidimensionalen Nichtstandardform am Beispiel eines mit Daubechies4-Wavelets transformierten Laplaceoperators, gezeigt sind alle Einträge vom Betrag größer 10^{-7}

2.3. Lösung von Gleichungen im Nichtstandardsystem

Beylkin, Gines und Dunn schlagen in [25] vor, Differential- und Integralgleichungen durch LU-Faktorisierung der Nichtstandardform zu lösen. Sie kommen zu dem Ergebnis, daß dies bei Pseudodifferentialoperatoren mit einem Aufwand von $\mathcal{O}(n)$ möglich ist, wenn n die Zahl der Unbekannten ist. Dies läßt sich mit dem bei diesen Operatoren gegebenen polynomialen Abfall der Einträge in Abhängigkeit der Zahl der verschwindenden Momente der verwendeten Waveletbasis mit zunehmender Entfernung von der Diagonalen begründen. Voraussetzung für die Anwendbarkeit des Verfahrens ist, daß die zugrundeliegende Gleichung auf einem rechteckigem Gebiet betrachtet wird. Ferner wird eine Waveletbasis mit einer ausreichenden Anzahl von verschwindenden Momenten benötigt. Abschnitt 2.3.1 beschreibt das entsprechende Verfahren.

Als alternative Methode kann man, ausgehend von der Überlegung, daß der Grobgitteroperator T_j auf jedem Level j den Schurkomplementoperator approximiert [50], anstatt den Schurkomplementoperator, wie in 2.3.1 geschildert, auf einem Band zu approximieren, ihn durch den Grobgitteroperator ersetzen (siehe Abschnitt 2.3.2) oder gar eine andere Faktorisierung wählen (Abschnitt 2.3.5), sodaß die (I)LU-Zerlegung auch durch ein anderes Verfahren zur (approximativen) Invertierung ersetzt werden kann. Insbesondere wird auf diese Weise die Nähe zu anderen Lösungsmethoden deutlich.

In zwei Dimensionen ist die reine (inkomplette) LU-Zerlegung aufgrund der Struktur der Nichtstandardform problematisch. Die Bearbeitung in $\mathcal{O}(n)$ bei n Unbekannten verlangt einen geschickten Umgang mit der Dicke des bearbeiteten Bandes (siehe 2.3.3). Der komplette Verzicht auf die approximative Berechnung des Schurkomplementes (2.3.4) befreit von diesen Schwierigkeiten, besitzt dafür aber andere Nachteile. So ist eine Banderweiterung zusammen mit dieser Methode nicht möglich, wodurch ein wichtiger Parameter zur Steuerung des Konvergenzverhaltens des Verfahrens nicht zur Verfügung steht. Verbreitert man das Band nämlich immer mehr, so wird aus dem Verfahren im Grenzfall ein direkter Löser, der dann allerdings mit hohen Kosten verbunden ist. Die Schwierigkeit besteht nun darin, mittels einer möglichst geringen Banderweiterung bei fixen und möglichst geringen Kosten einen zufriedenstellenden Löser zu konstruieren.

2.3.1. LU-Faktorisierung im Nichtstandardsystem

Im folgenden wird zunächst ein direkter Löser vorgestellt, um einen Einblick in die Funktionsweise des Verfahrens zu geben. Ausgehend von diesem für die praktische Anwendung zu teuren Löser wird anschließend die kostengünstigere Variante erläutert, bei der sich die Rechnungen auf ein Diagonalband der jeweiligen Matrix beschränken.

Es soll die diskrete Gleichung

$$T_0 x = f$$

gelöst werden. Dafür wird zunächst T_0 in eine Nichtstandardform T_{ns} zerlegt. Zur Veranschaulichung wird das Gleichungssystem erst für die Zerlegungstiefe 1 betrachtet: der Vektor f wird ebenfalls in das Nichtstandardsystem transformiert, so daß man

$$\begin{array}{ccc} \mathcal{W}_1^{0\,T} T_0 \mathcal{W}_1^0 (\mathcal{W}_1^0)^{-1} x & = & \mathcal{W}_1^{0\,T} f \\ \| & & \| \\ T_{ns} x_{ns} & = & f_{ns} \end{array}$$

erhält. Nun erzeugt man zwei Faktoren L_{ns} (untere Dreiecksmatrix) und U_{ns} (obere Dreiecksmatrix) mit

$$L_{ns} U_{ns} = T_{ns}$$

und löst

$$L_{ns} y_{ns} = f_{ns}$$

und anschließend

$$U_{ns} x_{ns} = y_{ns}.$$

Daraus läßt sich dann x rekonstruieren.

Durch die Transformation erhält man also das System

$$\underbrace{\begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 \end{pmatrix}}_{=:\tilde{T}_0} \begin{pmatrix} w_1 \\ v_1 \end{pmatrix} = \begin{pmatrix} \hat{w}_1 \\ \hat{v}_1 \end{pmatrix}.$$

Die Matrix $\tilde{T}_0$ soll nun mittels LU-Zerlegung in zwei Faktoren

$$\begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 \end{pmatrix} = \begin{pmatrix} L_1 & 0 \\ \hat{C}_1 & L_{\bar{T}_1} \end{pmatrix} \begin{pmatrix} U_1 & \hat{B}_1 \\ 0 & U_{\bar{T}_1} \end{pmatrix}$$

zerlegt werden. Man beginnt mit der LU-Zerlegung der Teilmatrix A_1 in $A_1 = L_1 U_1$. Dann müssen erfüllt sein:

$$\begin{aligned} B_1 &= L_1 \hat{B}_1, \\ C_1 &= \hat{C}_1 U_1. \end{aligned}$$

Die Teilmatrizen $\hat{B}_1$ und $\hat{C}_1$ lassen sich mittels einer Vorwärtssubstitution aus L_1 und B_1 beziehungsweise einer Rückwärtssubstitution aus U_1 und C_1 konstruieren. Weiterhin muß

$$\begin{aligned} T_1 &= \hat{C}_1 \hat{B}_1 + L_{\bar{T}_1} U_{\bar{T}_1} \\ &= C_1 U_1^{-1} L_1^{-1} B_1 + L_{\bar{T}_1} U_{\bar{T}_1} \\ &= C_1 A_1^{-1} B_1 + L_{\bar{T}_1} U_{\bar{T}_1} \end{aligned}$$

gelten. Um also $L_{\bar{T}_1}$ und $U_{\bar{T}_1}$ berechnen zu können, ist es nötig, vorher $\hat{C}_1 \hat{B}_1$ von T_1 abzuziehen:

$$\bar{T}_1 := T_1 - \hat{C}_1 \hat{B}_1.$$

$L_{\bar{T}_1}$ und $U_{\bar{T}_1}$ erhält man dann mittels einer LU-Zerlegung des Schurkomplementoperators $\bar{T}_1$. Die Subtraktion des *Korrekturterms* $\hat{C}_j \hat{B}_j$ von $T_j,\ j \in \{1, 2, \ldots, lt\}$, wird im folgenden als *Schurkomplementkorrektur* bezeichnet. Algorithmus 2.1 faßt das Vorgehen zusammen.

Nach diesen Vorbereitungen ist es nur noch ein kleiner Schritt bis zur Lösung des Gleichungssystems. Der wavelettransformierte Vektor f zerfällt in die Bestandteile $\hat{w}_1$ und $\hat{v}_1$. Zunächst wird aus $\hat{w}_1$ mit einer Vorwärtssubstitution mit L_1 der Feinanteil $\tilde{w}_1$ berechnet. Anschließend korrigiert man $\hat{v}_1$, indem man das Produkt $\hat{C}_1 \tilde{w}_1$ subtrahiert. Es folgt die Grobgitterkorrektur: $\tilde{v}_1$ wird aus einer Vorwärtssubstitution mit $L_{\bar{T}_1}$ und $\hat{v}_1$ berechnet, danach v_1 aus einer Rückwärtssubstitution mit $U_{\bar{T}_1}$ und $\tilde{v}_1$. Nachdem $\tilde{w}_1$ über die Subtraktion von $\hat{B}_1 v_1$ korrigiert wurde, wird mittels einer weiteren Rückwärtssubstitution mit U_1 und $\tilde{w}_1$ der Feinanteil w_1 der Lösung berechnet. Nun muß nur noch der Lösungsvektor zurücktransformiert werden. Algorithmus 2.2 gibt den Lösungsprozeß wieder.

Jetzt ist es ist nicht mehr schwierig, dieses Verfahren auf eine Leveltiefe größer 1 zu übertragen. Anstatt den Schurkomplementoperator $\bar{T}_1$ in $L_{\bar{T}_1}$ und $U_{\bar{T}_1}$ zu zerlegen, transformiert

Eingabedaten: diskretisierter Operator T_0
Ausgabedaten: L_{ns} und U_{ns}

transformiere T_0 in $A_1,\ B_1,\ C_1$ und T_1
berechne die LU-Zerlegung von A_1 als L_1 und U_1
berechne $\hat{B}_1$ mit einer Vorwärtssubstitution aus L_1 und B_1
berechne $\hat{C}_1$ mit einer Rückwärtssubstitution aus U_1 und C_1
berechne $\bar{T}_1 := T_1 - \hat{C}_1 \hat{B}_1$
berechne die LU-Zerlegung von $\bar{T}_1$ als $L_{\bar{T}_1}$ und $U_{\bar{T}_1}$

Algorithmus 2.1: Wavelettransformation und LU-Zerlegung von T_0 für Leveltiefe 1

Eingabedaten: Vektor $f,\ L_{ns},\ U_{ns}$
Ausgabedaten: Lösung des Gleichungssystems $T_0 x = f$

$\hat{v}_0 := f$
zerlege $\hat{v}_0$ in $\hat{w}_1$ und $\hat{v}_1$
berechne $\tilde{w}_1$ mit einer Vorwärtssubstitution aus $\hat{w}_1$ und L_1
berechne $\hat{v}_1 := \hat{v}_1 - \hat{C}_1 \cdot \tilde{w}_1$
berechne $\tilde{v}_1$ mit einer Vorwärtssubstitution aus $\hat{v}_1$ und $L_{\bar{T}_1}$
berechne v_1 mit einer Rückwärtssubstitution aus $\tilde{v}_1$ und $U_{\bar{T}_1}$
berechne $\tilde{w}_1 := \tilde{w}_1 - \hat{B}_1 \cdot v_1$
berechne w_1 mit einer Rückwärtssubstitution aus $\tilde{w}_1$ und U_1
rekonstruiere v_0 aus w_1 und v_1
setze $x := v_0$

Algorithmus 2.2: direkter Löser im Einlevelfall

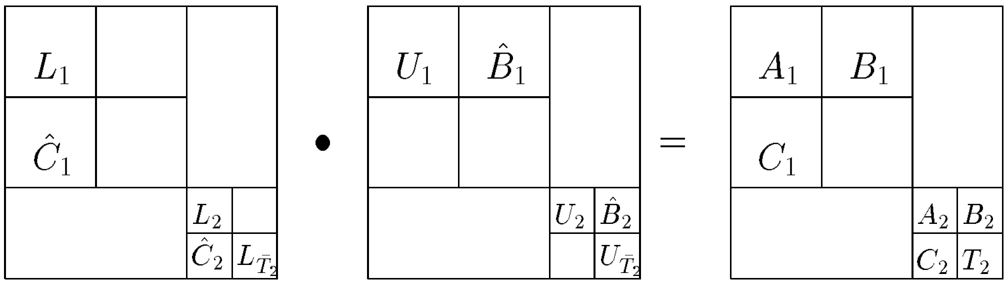

Abbildung 2.10.: Die Faktorisierung der Matrix T_{ns} bei Leveltiefe 2.

man ihn in A_2 bis T_2. Mit diesen Teilmatrizen verfährt man dann wie im Einlevelfall, es sei denn, man möchte noch weiter zerlegen. Erst auf dem letzten Level lt wird die LU-Zerlegung von $\bar{T}_{lt}$ durchgeführt. Abbildung 2.10 stellt die Faktorisierung für Leveltiefe 2 dar.

Algorithmus 2.3 beschreibt den Ablauf der Zerlegung und Faktorisierung im Mehrlevelfall.

Eingabedaten: diskretisierter Operator T_0, Leveltiefe lt, bis zu der T_0 zerlegt werden soll
Ausgabedaten: L_{ns} und U_{ns}

`for` $i = 1$ bis lt
  zerlege T_{i-1} in A_i, B_i, C_i und T_i
  berechne die LU-Zerlegung von A_i als L_i und U_i
  berechne $\hat{B}_i$ mit einer Vorwärtssubstitution aus L_i und B_i
  berechne $\hat{C}_i$ mit einer Rückwärtssubstitution aus U_i und C_i
  berechne $\bar{T}_i := T_i - \hat{C}_i \cdot \hat{B}_i$
`endfor`
berechne die LU-Zerlegung von $\bar{T}_{lt}$ als $L_{\bar{T}_{lt}}$ und $U_{\bar{T}_{lt}}$

Algorithmus 2.3: Transformation in die Nichtstandardform und gleichzeitige LU-Zerlegung von T_0

Die Übertragung des Lösungsprozesses auf den Mehrlevelfall verläuft analog. Algorithmus 2.4 gibt das Ergebnis unter Voraussetzung einer bereits gemäß Algorithmus 2.3 faktorisierten Matrix T_0 wieder.

Der Lösungsalgorithmus 2.4 kann auch rekursiv formuliert werden, wie in Algorithmus 2.5 dargestellt wird. Auf diese Weise wird die Verwandtschaft zum aus den Mehrgitterverfahren bekannten V-Zyklus [31] deutlich sichtbar, denn auch dieser glättet zunächst auf einem feineren Gitter, geht dann auf das nächstgröbere, glättet dort wieder, erreicht auf diese Weise

Eingabedaten: Vektor f, Leveltiefe lt, L_{ns}, U_{ns}
Ausgabedaten: Lösung des Gleichungssystems $T_0 x = f$

$\hat{v}_0 := f$
`for` $i = 1$ bis lt
  zerlege $\hat{v}_{i-1}$ in $\hat{w}_i$ und $\hat{v}_i$
  berechne $\tilde{w}_i$ mit einer Vorwärtssubstitution aus $\hat{w}_i$ und L_i
  berechne $\hat{v}_i := \hat{v}_i - \hat{C}_i \cdot \tilde{w}_i$
`endfor`
berechne $\tilde{v}_{lt}$ mit einer Vorwärtssubstitution aus $\hat{v}_{lt}$ und $L_{\bar{T}_{lt}}$
berechne v_{lt} mit einer Rückwärtssubstitution aus $\tilde{v}_{lt}$ und $U_{\bar{T}_{lt}}$
`for` $i = lt$ bis 1 (abwärts zählend)
  berechne $\tilde{w}_i := \tilde{w}_i - \hat{B}_i \cdot v_i$
  berechne w_i mit einer Rückwärtssubstitution aus $\tilde{w}_i$ und U_i
  rekonstruiere v_{i-1} aus w_i und v_i
`endfor`
setze $x := v_0$

Algorithmus 2.4: direkter Löser im Mehrlevelfall, ausgehend von einer mit Algorithmus 2.3 zerlegten und faktorisierten Matrix T_0

das gröbste Gitter und läuft anschließend wiederum glättend über sämtliche Zwischengitter zum feinsten Giter zurück. Insofern entspricht der obere Teil von Algorithmus 2.5 dem Vorglätter und der untere Teil von Algorithms 2.5 dem Nachglätter eines V-Zyklusses. Die verwendeten Zweiskalentransformationen zur Zerlegung und Rekonstruktion des Vektors f in Algorithmus 2.5 entsprechen Prolongation und Restriktion des Mehrgitteralgorithmus.

Eingabedaten: Vektor f, Leveltiefe lt und ein mit Algorithmus 2.3 LU-zerlegter Operator
Ausgabedaten: Lösung des Gleichungssystems $T_0 x = f$

$i := 1$
rufe `löse` $\{i, f\}$ auf
`löse` $\{i, f\}$
  zerlege f in $\hat{w}$ und $\hat{v}$
  berechne $\tilde{w}$ mit einer Vorwärtssubstitution aus $\hat{w}$ und L_i
  berechne $\hat{v} := \hat{v} - \hat{C}_i \cdot \tilde{w}$
  `if` i=lt
    berechne $\tilde{v}$ mit einer Vorwärtssubstitution aus $\hat{v}$ und $L_{\bar{T}_{lt}}$
    berechne v mit einer Rückwärtssubstitution aus $\tilde{v}$ und $U_{\bar{T}_{lt}}$
  `else`
    rufe `löse`$\{i+1, \hat{v}\}$ auf
  `endif`
  berechne $\tilde{w} := \tilde{w} - \hat{B}_i \cdot v$
  berechne w mit einer Rückwärtssubstitution aus $\tilde{w}$ und U_i
  rekonstruiere f aus w und v
`end löse`
setze $x := f$

Algorithmus 2.5: Rekursive Formulierung des direkten Lösers im Mehrlevelfall, ausgehend von einer mit Algorithmus 2.3 zerlegten und faktorisierten Matrix T_0

Dieses Verfahren ist bis auf Rechenungenauigkeiten exakt, wenn man T_{ns} und seine Faktorisierung exakt berechnet. T_0 besitzt n^2 Einträge, also ist auch die Summe der Anzahl der Einträge von A_1 bis T_1 n^2. Daraus folgt, daß eine exakte Berechnung von A_1 bis T_1 mindestens einen Rechenaufwand von $\mathcal{O}(n^2)$ verursacht, was zu teuer ist. Beylkin, Gines und Dunn reduzieren die Kosten auf $\mathcal{O}(n)$, indem sie von vorneherein sämtliche Rechnungen nur innerhalb eines Diagonalbandes durchführen [25]. So wird es ihnen möglich, mit eigentlich vollbesetzten Integraloperatoren in einer dünnbesetzten Nichtstandardformdarstellung zu arbeiten. Für Pseudodifferentialoperatoren ist dies aufgrund eines polynomialen Abfalls der Einträge mit zunehmendem Abstand von der Diagonalen zunächst nicht erstaunlich. Also müßte auch für diese eine Beschränkung der Rechnungen auf ein Diagonalband möglich sein. Dafür läuft man also zum Beispiel zwar noch über sämtliche Zeilen von T_0, aber nur noch über eine begrenzte Anzahl von Spalten in jeder Zeile. Auf diese Weise läßt sich das hier betrachtete Verfahren in $\mathcal{O}(n)$ durchführen, allerdings auf Kosten von Genauigkeit. Man erhält dann zwei Faktoren $\tilde{L}_{ns}$ und $\tilde{U}_{ns}$ mit

$$\tilde{L}_{ns} \cdot \tilde{U}_{ns} \approx T_{ns}.$$

Laut [25] läßt sich der Fehler determinieren, der bei dieser Approximation in Abhängigkeit einer bestimmten Bandbreite ω des Bandes um die Diagonale entsteht. Kritisch ist dabei der Schurkomplementoperator, der auch bei Pseudodifferentialoperatoren vollbesetzt ist. Abschnitt 2.4 geht näher auf diese Thematik ein.

In der Praxis ergeben sich in diesem Zusammenhang einige Schwierigkeiten. So wurde, um die vorgegebene Bandbreite nicht zu vergrößern, eine ILU-Zerlegung (incomplete LU) für die Matrizen $A_j,\ j \in \{1, \ldots, lt\}$ und T_{lt} verwendet, die nur innerhalb des betrachteten Bandbereiches Einträge berechnet. Eine Alternative wäre gewesen, nur dort Einträge zu berechnen, wo vorher keine Null stand. Die Bandfaktorisierung liefert im allgemeinen eine bessere Approximation an die LU-Zerlegung. Weitere Varianten wie beispielsweise ein MILU [3], bei dem Einträge, die außerhalb des betrachteten Bandes liegen, zur Diagonalen hinzuaddiert werden, sind möglich. Gerade für letzteren war jedoch der Aufwand höher als der Nutzen, sodaß die Bandfaktorisierung verwendet wurde. Dies führt jedoch – zumindest bei periodisierten Operatoren oder nicht randangepaßten Wavelets – zu einem Fehler im Randbereich der Matrix.

Anschließend ist $\hat{B}_j$ mit einer Vorwärtssubstitution zu berechnen. Während B_j Bandgestalt besitzt, ist das bei einem korrekt berechneten $\hat{B}_j$ nicht so. Nur die Einträge oberhalb des Bandes um die Diagonale bleiben leer. Da man es sich jedoch nicht leisten kann, die linke untere Hälfte von $\hat{B}_j$ komplett zu füllen, wurden auch hier nur die Elemente berechnet, die innerhalb des in B_j besetzten Bandes lagen. Dieses Band konnte zusätzlich um eine vorher festgelegte Zahl von Spalten erweitert werden, um die Genauigkeit der Rechnung zu erhöhen. Dasselbe gilt für die Berechnung von $\hat{C}_j$, nur daß sich hier bei einer exakten Rechnung die rechte obere Hälfte der Matrix komplett füllen würde. Auch die folgende Berechnung der Schurkomplementkorrektur ist in diesem Zusammenhang problematisch. Bei einer gegebenen Bandbreite von $\hat{B}_j$ und $\hat{C}_j$ von k hat ihr Produkt eine Bandbreite von etwa $2k$. An dieser Stelle wurde jedoch auf eine erneute Verbreiterung des Bandes verzichtet. Das Schurkomplement wurde in einem Band derselben Breite approximiert wie vorher $\hat{B}_j$ und $\hat{C}_j$. Dabei spielte die Breite des Bandes eine wesentliche Rolle für das Konvergenzverhalten des iterierten Verfahrens. Es stellt sich die Frage, wie breit man das Band machen muß, um das Verhältnis zwischen Rechenaufwand auf der einen Seite und Genauigkeit der Lösung auf der anderen Seite zu optimieren.

Da man mit einem direkten Waveletbandlöser keine hinreichend genauen Ergebnisse erhält, wird ein iteratives Verfahren verwendet. Die neue Iterierte wird über

$$x^i = x^{i-1} - M \cdot (A \cdot x^{i-1} - b)$$

ermittelt. Zunächst wird während der Erzeugung des Waveletvorkonditionierers M

$$x^0 = M \cdot b$$

berechnet. Diese Vorgehensweise bietet sich an, da es für die Konstruktion von M sowieso notwendig ist, einmal über sämtliche Level zu laufen. Sie ist äquivalent zur Verwendung des Nullvektors als Startvektor der Iteration. So wird x^0 als Startvektor in die Iterationsschleife übergeben.

Dieses Verfahren, welches aufgrund der Beschränkung auf das (erweiterte) Band, den Schurkomplementoperator approximiert, soll im folgenden als *Verfahren mit approximativer Schurkomplementberechnung* oder schlicht als *Verfahren 1* bezeichnet werden.

2.3.2. LU-Faktorisierung ohne Schurkomplementkorrektur

Um die oben geschilderten Schwierigkeiten bei der Berechnung des Schurkomplementoperators

$$\bar{T}_j = T_j - \hat{C}_j \hat{B}_j$$

zu umgehen, wurde diese Variante eingeführt. Sie verzichtet auf die Schurkomplementkorrektur, $\bar{T}_j$ wird also durch T_j approximiert:

$$\bar{T}_j \approx T_j.$$

Berechtigung für dieses Verfahren gibt die Überlegung, daß das Produkt $\hat{C}_j \hat{B}_j$ das gleiche (starke) Abfallverhalten der Einträge mit zunehmendem Abstand von der Diagonalen aufweist wie die Teilmatrizen A_j, B_j und C_j (siehe Abschnitt 2.4) und insofern der Operator T_j auf jedem Level j den wesentlicheren Anteil des Schurkomplementoperators beinhaltet.

Dadurch wird die Berechnung des Korrekturterms $\hat{C}_j \hat{B}_j$ unnötig. Es stellt sich die Frage, ob es möglich ist, ganz auf die Berechnung von $\hat{C}_j$ und $\hat{B}_j$ zu verzichten, wodurch die Banderweiterung ebenfalls ihren Sinn verlieren würde. Die einzige übrige Stelle im Algorithmus, an der diese beiden Teilmatrizen bislang benötigt wurden, sind die Multiplikationen $\hat{B}_j \cdot v_j$ und $\hat{C}_j \cdot \tilde{w}_j$. Wegen

$$L_j \hat{B}_j = B_j \Longleftrightarrow \hat{B}_j = L_j^{-1} B_j$$

läßt sich die Multiplikation von $\hat{B}_j$ mit v_j als Multiplikation von B_j mit v_j mit anschließender Vorwärtssubstitution mit L_j und v_j realisieren. Ebenso gilt für $\hat{C}_j$:

$$\hat{C}_j = C_j U_j^{-1}.$$

Aa	Ab	Ba	Bb
Ac	At	Bc	Bt
Ca	Cb	Ta	Tb
Cc	Ct	Tc	Tt

Abbildung 2.11.: Aufteilung der 2D-Nichtstandardform nach Teilmatrizen.

Das Produkt $\hat{C}_j\tilde{w}_j$ erhält man folglich genauso durch eine Rückwärtssubstitution mit U_j und $\tilde{w}_j$ und anschließender Multiplikation von C_j mit dem Ergebnis der Substitution.

Somit kann man gänzlich auf eine Verbreiterung des Bandes verzichten und erhält dennoch das exakte Produkt $\hat{B}_jv_j$ beziehungsweise $\hat{C}_j\tilde{w}_j$. Andererseits verliert man durch diesen Verzicht einen wichtigen Parameter zur Steuerung der Konvergenzrate des Verfahrens. Der durch die Weglassung der Schurkomplementkorrektur entstehende Fehler wird von Level zu Level übertragen und sukzessive vergrößert. Es fragt sich also, ob diese Variante besser oder schlechter ist als die Variante mit Banderweiterung.

Wie bereits durch die Überschrift angedeutet, wird dieses Verfahren im folgenden als *Verfahren ohne Schurkomplementkorrektur* oder schlicht als *Verfahren 2* bezeichnet.

2.3.3. LU-Faktorisierung in zwei Dimensionen

Analog zum Löser für eindimensionale Probleme läßt sich auch ein direkter Löser für zweidimensionale Probleme aufbauen. Dieser wäre allerdings ebenfalls mit Kosten in Höhe von $\mathcal{O}(n^3)$ zu teuer. Daher wird im folgenden direkt der Bandlöser betrachtet.

Wie bereits in Abschnitt 2.2.3 erläutert wurde, besteht die zweidimensionale Nichtstandardform auf jedem Level j aus 16 Teilmatrizen Aa_j bis Tt_j (Abbildung 2.11). Diese natürliche Aufteilung der zweidimensionalen Nichtstandardform legt es nahe, auch das Lösungsverfahren entsprechend zu konstruieren.

Dies würde bedeuten, zunächst eine (I)LU-Zerlegung von Aa_j durchzuführen und die zu-

Aa	Ab	Ba	Bb
Ac	At	Bc	Bt
Ca	Cb	Ta	Tb
Cc	Ct	Tc	Tt

Abbildung 2.12.: Aufteilung der 2D-Nichtstandardform nach der Grobstruktur.

gehörigen Teilmatrizen $\hat{A}b_j$, $\hat{B}a_j$, $\hat{B}b_j$, $\hat{A}c_j$, $\hat{C}a_j$ und $\hat{C}c_j$ zu berechnen, bevor man die neun Teilmatrizen unten rechts (At_j bis Ta_j) korrigiert. Anschließend müßte man At_j (I)LU-zerlegen, um nach einer Berechnung von $\hat{B}c_j$, $\hat{B}t_j$, $\hat{C}b_j$ und $\hat{C}t_j$ die Teilmatrizen Ta_j bis Tt_j korrigieren zu können. Erst danach käme die (I)LU-Zerlegung von Ta_j an die Reihe, usw.

Der Nachteil dieser Methode ist, daß man, wenn man mit Bandmatrizen rechnet, mit jeder Multiplikation zweier Teilmatrizen die Bandbreite verdoppeln müßte, um eine gute Approximation berechnen zu können. Auf jedem Level verachtfachte sich somit die erforderliche Bandbreite. Folglich ist diese Methode unbrauchbar.

Die nächste Variante wird durch Abbildung 2.12 motiviert. Man behandelt die Grobstrukturblöcke entsprechend dem eindimensionalen Verfahren. Da man dabei das Schurkomplement auf ganz T (also Ta_j bis Tt_j) berechnen würde, es aber nur auf Tt_j berechnet werden darf, ist auch diese Idee nicht verwendbar.

Der Ausweg ist, die (I)LU-Zerlegung auf den neun Teilblöcken oben links als Ganzes durchzuführen. Die drei Blöcke Bb_j, Bt_j und Tb_j werden dann so behandelt wie B im eindimensionalen Fall, entsprechend Cc_j, Ct_j und Tc_j wie C und Tt_j wie T. Abbildung 2.13 verdeutlicht die dafür verwendete Aufteilung der zweidimensionalen Nichtstandardform. Man erhält diese Aufteilung, wenn man die einzelnen Teiloperatoren nach ihren Ursprungs- und Bildräumen sortiert (vergleiche Formel (2.1)): Tt_j bildet von V_j nach V_j ab, der Block mit den Matrizen Cc_j, Ct_j und Tc_j von W_j, $V_j^x \otimes W_j^y$ und $W_j^x \otimes V_j^y$ nach V_j, enthält also sämtliche Abbildungen nach V_j (außer Tt_j). Ensprechend werden in Bb_j, Bt_j und Tb_j die Abbildungen aus V_j nach W_j, $V_j^x \otimes W_j^y$ und $W_j^x \otimes V_j^y$ zu einem B-Block zusammenge-

Aa	Ab	Ba	Bb
Ac	At	Bc	Bt
Ca	Cb	Ta	Tb
Cc	Ct	Tc	Tt

Abbildung 2.13.: Aufteilung der 2D-Nichtstandardform nach Räumen.

faßt. Der große A-Block beinhaltet die restlichen Abbildungen, welche sämtlich weder aus V_j heraus noch nach V_j führen.

Da das zweidimensionale Verfahren auf diese Weise auf das eindimensionale Verfahren zurückgeführt wird, verdoppelt sich die Breite des zu betrachtenden Bandes nur einmal pro Level. Man hat somit das gleiche Problem wie in der eindimensionalen Variante, nämlich wie breit das Band zu wählen ist.

Allerdings gibt es jetzt nicht mehr nur ein Band, sondern ein Band in jedem Block und ein Band aus Blöcken. Da – je nach Sortierung – das eine die Spalten-, das andere die Zeilenkopplungen beinhaltet, müssen beide verbreitert werden. Problematisch ist die Auswirkung der zweidimensionalen Struktur auf die Effizienz des Verfahrens, wenn man die Rechnung auf das Band beschränken möchte. So ist die Größe der Ausgangsmatrix bereits 64×64, wenn jeder Block gerade einmal 8×8 Elemente beinhaltet. Nach Transformation hat jeder Block dann nur noch 4×4 Elemente, sodaß es nicht mehr möglich ist, die Rechnung auf ein Band zu beschränken. Erst bei größeren Ausgangsmatrizen läßt sich der effiziente Bandalgorithmus einsetzen.

2.3.4. LU-Faktorisierung ohne Schurkomplementkorrektur in zwei Dimensionen

Wie in der eindimensionalen Fassung ist die Approximation des Schurkomplementoperators durch den Grobgitteroperator eine Alternative, die ohne Verbreitern der Bänder auskommt. Hierbei wird die gleiche (I)LU-Zerlegung wie in der Fassung mit Approximation des Schurkomplementes verwendet.

Wie bereits oben beschrieben wurde, wirkt sich der Verzicht auf die Banderweiterung, der mit dieser Variante einhergeht, gerade im Zweidimensionalen nachteilig aus. Ansonsten ist das Vorgehen prinzipiell das gleiche wie im Eindimensionalen.

2.3.5. Gemeinsamkeiten mit bekannten Verfahren

Die folgenden Betrachtungen beschränken sich aufgrund der besseren Verständlichkeit auf den Einlevelfall. Der Mehrlevelfall ist völlig analog. Bislang wurde $\tilde{T}_0$ wie in (2.2) faktorisiert:

$$\begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 \end{pmatrix} = \begin{pmatrix} L_1 & 0 \\ C_1 U_1^{-1} & L_{\bar{T}_1} \end{pmatrix} \begin{pmatrix} U_1 & L_1^{-1} B_1 \\ 0 & U_{\bar{T}_1} \end{pmatrix}. \tag{2.2}$$

Dabei galt $L_{\bar{T}_1} U_{\bar{T}_1} = \bar{T}_1 = T_1 - C_1 U_1^{-1} L_1^{-1} B_1$. Eine äquivalente Faktorisierung ist diese:

$$\begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 \end{pmatrix} = \begin{pmatrix} A_1 & 0 \\ C_1 & T_1 - C_1 A_1^{-1} B_1 \end{pmatrix} \begin{pmatrix} Id & A_1^{-1} B_1 \\ 0 & Id \end{pmatrix}. \tag{2.3}$$

Wählt man zur Invertierung von A_1 die (I)LU-Zerlegung, so sind die Faktorisierungen (2.2) und (2.3) offensichtlich äquivalent. Approximiert man weiter das Schurkomplement $\bar{T}_1$ durch den Grobgitteranteil T_1, wie in Abschnitt 2.3.2 beschrieben wurde, so wird daraus

$$\begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 + C_1 A_1^{-1} B_1 \end{pmatrix} = \begin{pmatrix} A_1 & 0 \\ C_1 & T_1 \end{pmatrix} \begin{pmatrix} Id & A_1^{-1} B_1 \\ 0 & Id \end{pmatrix}, \tag{2.4}$$

und dies läßt sich wie folgt umschreiben:

$$\begin{pmatrix} A_1 & B_1 \\ C_1 & T_1 + C_1 A_1^{-1} B_1 \end{pmatrix} = \begin{pmatrix} A_1 & 0 \\ C_1 & T_1 \end{pmatrix} \begin{pmatrix} A_1^{-1} & 0 \\ 0 & T_1^{-1} \end{pmatrix} \begin{pmatrix} A_1 & B_1 \\ 0 & T_1 \end{pmatrix}. \tag{2.5}$$

Die approximative Faktorisierung (2.5) entspricht also genau einem symmetrischen Block-Gauß-Seidel-Verfahren [5].

Die Verwandtschaft mit dem V-Zyklus wurde bereits in Abschnitt 2.3.1 erwähnt [31]. Man korrigiert in beiden Fällen zunächst auf dem feinsten Gitter, übernimmt die Korrektur sukzessive auf die gröberen Gitter, korrigiert dort und übernimmt anschließend diese Korrektur wieder sukzessive bis zum feinsten Gitter, um hier einen letzten Korrekturschritt auszuführen. Betrachtet man, wie durch die Multi-Resolution-Analysis vorgegeben, die gröberen Räume als Unterräume der feineren, so kann man das Verfahren auch als multiplikatives Unterraumkorrekturverfahren [54] auffassen.

Die in Gleichung (2.3) dargestellte Faktorisierung läßt sich für weitere Varianten des Verfahrens nutzen. So kann man statt mit einer (I)LU-Zerlegung die Submatrizen A_j mit einem anderen Verfahren (approximativ) invertieren. Der Frage, inwieweit und in welchen Fällen dieses Vorgehen sinnvoll ist, wurde im Rahmen dieser Arbeit nicht weiter nachgegangen.

2.4. Schnelle LU-Faktorisierung

In Abschnitt 2.3.1 wurde bereits darauf hingewiesen, daß sich nach [25] eine approximative LU-Faktorisierung für gewisse nichttriviale Operatoren mit einem Aufwand von $\mathcal{O}(n)$ durchführen läßt, wobei n die Zahl der Freiheitsgrade ist.

Während die Feinstlevelmatrixdarstellung von Differentialoperatoren (Block-)Bandgestalt besitzt, ist dies für Integraloperatoren im allgemeinen nicht der Fall. Um so interessanter ist es, daß die in [25] gemachten Aussagen auch für Integraloperatoren gelten.

Beylkin, Gines und Dunn stellen die zum diskretisierten Operator T_0 gehörende Matrix innerhalb eines (Block-)Diagonalbandes der Breite ω auf. Da sich ω mit jeder Transformation auf das nächste Level halbiert, erweitern sie vor der Zerlegung von $T_k, k \in \{2, 3, \ldots, lt-1\}$ das aufgrund der vorangegangen Zerlegung schmaler gewordene (Block-)Band auf die Breite ω. Hierfür werden die Einträge der zu transformierenden Matrix mittels einer Quadraturformel approximiert. Die neuen Einträge werden so berechnet, als wären sie wie die anderen durch Transformation des ursprünglichen Operators entstanden. Auf diese Weise bleibt einerseits die (Block-)Bandbreite auf jedem Level konstant, andererseits wird nach und nach die komplette zugehörige Operatormatrix bearbeitet. Es ist somit möglich, eine eigentlich vollbesetzte $n \times n$-Matrix mit einem Aufwand von nur $\mathcal{O}(n)$ in eine approximative Nichtstandardform zu überführen. Der Fehler dieser Näherung an die Nichtstandarform hängt dabei in seiner Höhe direkt von der Breite des betrachteten (Block-)Bandes ab.

In der vorliegenden Arbeit wurden statt der im allgemeinen vollbesetzten Integraloperatoren Differentialoperatoren untersucht, die nach einer geeigneten Diskretisierung bereits eine (Block-)Diagonalbandgestalt aufweisen. Insofern verwundert es nicht, wenn sich die erste Zweiskalentransformation dieser Matrizen in $\mathcal{O}(n)$ durchführen läßt. Wird vor der Transformation auf das jeweils nächste Level der Schurkomplementoperator um den entsprechenden Term korrigiert, so ist nicht offensichtlich, daß die sich ergebenden Teilmatrizen nach Transformation noch (Block-)Bandgestalt besitzen. Daß dies unter bestimmten Voraussetzungen so ist, beweist Coult in [17].

Im folgenden sollen zunächst eindimensionale Operatoren betrachtet werden, bevor kurz auf zweidimensionale Operatoren eingegangen wird. Zunächst besagt Satz 2.6, daß die Einträge der Nichtstandardform eines Operators unter bestimmten Voraussetzungen mit zunehmender Entfernung von der Diagonalen betragsmäßig abnehmen. Er bezieht sich auf eindimensionale Operatoren, obwohl entsprechende Sätze auch in höheren Dimensionen gelten.

Satz 2.6 *Gegeben sei ein Integraloperator T der Gestalt*

$$(Tf)(x) = \int_{\mathbb{R}} K(x,y) f(y) dy,$$

der mit Wavelets mit M verschwindenden Momenten in seine Nichtstandardform zerlegt *wird und die folgenden Eigenschaften besitzt:*

$$\begin{aligned} |K(x,y)| &\le |x-y|^{-1}, \\ |\partial_x^M K(x,y)| + |\partial_y^M K(x,y)| &\le C|x-y|^{-M-1} \text{ und} \\ \left| \int_{I\times I} K(x,y)dxdy \right| &\le C|I|. \end{aligned}$$

Hierbei bezeichnet $I := \left[k \cdot 2^{-j}, (k+1)\cdot 2^{-j}\right[\subset \mathbb{R},\ k,\ j \in \mathbb{Z}$ *ein beliebiges dyadisches Intervall.*

Dann gilt für die Einträge $\alpha_{k,l}^j$, $\beta_{k,l}^j$ *und* $\gamma_{k,l}^j$ *der Blöcke* A_j, B_j *und* C_j

$$|\alpha_{k,l}^j| + |\beta_{k,l}^j| + |\gamma_{k,l}^j| \le C(1+|k-l|)^{-M-1}, \tag{2.6}$$

wobei C *vom Level* j, *von* M *und von* $K(x,y)$ *abhängig ist.*

Beweis: siehe [10]. □

Abschätzung (2.6) besagt, daß die Einträge in A_j, B_j und C_j mit zunehmendem Abstand von der Diagonalen betragsmäßig abnehmen, sodaß es zur Approximation der Nichtstandardform mit einer bestimmten Genauigkeit ε ausreichen sollte, ein Band um die Diagonale zu bearbeiten. Es stellt sich nun die Frage, ob auch der Schurkomplementoperator das Abfallverhalten (2.6) aufweist. Zu seiner Berechnung benötigt man das Produkt $C_j A_j^{-1} B_j$. Angenommen, C_j, A_j, B_j und T_j würden zu einer Klasse von Operatoren gehören, die unter Addition und Multiplikation abgeschlossen wäre und mit einer Matrix auch deren Inverse beinhalten würde, dann würde auch der Schurkomplementoperator $\bar{T}_j$ in dieser Klasse enthalten sein. Man definiert also für $\alpha > 0$ folgende Klasse bi-unendlicher Matrizen G mit polynomialem Abfall:

$$\mathcal{T}_\alpha = \{\{g_{k,l}\}_{k,l\in\mathbb{Z}} : \exists C(G) > 0 : |g_{k,l}| \le C(G)(1+|k-l|)^{-\alpha-1}\}.$$

Für diese gilt

Satz 2.7

1. *Die Klasse* $\mathcal{T}_\alpha$ *ist bezüglich Addition und Multiplikation abgeschlossen.*

2. *Wenn die Matrix* $G \in \mathcal{T}_\alpha$ *invertierbar bezüglich* l^2 *ist, dann ist auch ihre Inverse Element von* $\mathcal{T}_\alpha$.

Beweis: siehe [24]. □

In der Praxis werden diese idealen bi-unendlichen Matrizen nicht vorkommen, da den Rechnungen immer ein begrenztes Gitter zugrundeliegt. Insofern sind die hier angestellten Betrachtungen als Grenzwertbetrachtungen aufzufassen. Ebenso liegen bei bi-unendlichen Matrizen aufgrund der Abwesenheit von Rändern keine Randbedingungen vor. Insofern könnten in der Praxis periodische Randbedingungen die Eigenschaften von bi-unendlichen Matrizen besser widerspiegeln als Dirichlet-Randbedingungen. Da periodisierte Operatoren jedoch mitunter nicht vollen Rang besitzen, erfordern sie in numerischen Anwendugen eine gewisse Vorsicht. Die im Rahmen dieser Arbeit eingesetzten Verfahren hätten ohne weitere Modifikationen keine periodischen Matrizen verarbeiten können. Daher wurden ausschließlich nicht periodisierte Operatoren mit Nullrandbedingungen behandelt.

Aus den obigen Überlegungen folgt, daß das für die Berechnung des Schurkomplementoperators $\bar{T}_j,\ j \in \{1, \dots, lt\}$ benötigte Produkt $C_j A_j^{-1} B_j$ in $\mathcal{T}_\alpha$ enthalten ist, wenn dies auf die zu seiner Berechnung verwendeten Matrizen $A_j,\ B_j$ und C_j zutrifft. Der Schurkomplementoperator besteht jedoch zusätzlich aus T_j, und es wurde bislang nicht behauptet, daß T_j in $\mathcal{T}_\alpha$ enthalten wäre. Da T_j allerdings erst auf dem tiefsten Level direkt in die Faktorisierung von $\bar{T}_{lt}$ eingeht, genügt es, wenn jeweils die Teilmatrizen A_{j+1}, B_{j+1} und $C_{j+1},\ j \in \{1, \dots, lt-1\}$ des transformierten Schurkomplementoperators $\bar{T}_j$ wiederum in $\mathcal{T}_\alpha$ enthalten sind. Satz 2.8 gewährleistet dies für endlich viele Level j.

Satz 2.8 *Unter den Voraussetzungen von Satz 2.6 gilt für die Einträge der durch eine Waveletzerlegung des Schurkomplementoperators $\bar{T}_j$ gewonnenen Matrizen $A_{j+1},\ B_{j+1}$ und C_{j+1}*

$$|\alpha_{k,l}^{j+1}| + |\beta_{k,l}^{j+1}| + |\gamma_{k,l}^{j+1}| \leq C(1 + |k-l|)^{-M-1}. \tag{2.7}$$

Außerdem läßt sich $\bar{T}_{j+1}$ zerlegen in

$$\bar{T}_{j+1} = T_{j+1} - K_{j+1}, \tag{2.8}$$

wobei K_{j+1} in $\mathcal{T}_\alpha$ enthalten ist.

Beweis:
Der Beweis erfolgt induktiv. Zunächst gilt Gleichung (2.7) nach Satz 2.6 für $j = 0$, wie direkt aus der Voraussetzung folgt. Das bedeutet insbesondere, daß A_1, B_1 und C_1 in $\mathcal{T}_\alpha$ enthalten sind. Wegen Satz 2.7 ist somit auch die Inverse von A_1 und folglich das Produkt $C_1 A_1^{-1} B_1$ Element von $\mathcal{T}_\alpha$. Über die Definition $K_1 := C_1 A_1^{-1} B_1$ folgt Gleichung (2.8).

Nun sei vorausgesetzt, daß die Gleichungen (2.7) und (2.8) für ein $j = k$ gelten. Dann ist zu zeigen, daß beide auch für $j = k+1$ erfüllt sind. Dazu betrachtet man zunächst $\bar{T}_k$. Nach Voraussetzung gilt $\bar{T}_k = T_k - K_k$. Die Matrizen A_{k+1}, B_{k+1} und C_{k+1} lassen sich also aus den Zweiskalentransformierten von T_k und K_k zusammensetzen. Für die bei der Zerlegung von T_k entstehenden Anteile $A_{T,k+1}, B_{T,k+1}$ und $C_{T,k+1}$ gilt Satz 2.6, und da K_k gemäß Induktionsannahme in $\mathcal{T}_\alpha$ enthalten ist, überträgt sich dies auch auf die bei der Transformation entstehenden Anteile $A_{K,k+1}, B_{K,k+1}$ und $C_{K,k+1}$. Daraus folgt, daß die Differenzen $A_{k+1} = A_{T,k+1} - A_{K,k+1}, B_{k+1} = B_{T,k+1} - B_{K,k+1}$ und $C_{k+1} = C_{T,k+1} - C_{K,k+1}$ Gleichung (2.7) erfüllen, was zu zeigen war. Wegen $T_{K,k+1} \in \mathcal{T}_\alpha$ und

$$\bar{T}_{k+1} = T_{T,k+1} - (T_{K,k+1} - C_{k+1}A_{k+1}^{-1}B_{k+1})$$

gilt Gleichung (2.8) für $k+1$. □

Entsprechend läßt sich für zweidimensionale Operatoren die Klasse

$$\mathcal{U}_\alpha = \{\{g_{k,k',l,l'}\}_{k,k',l,l' \in \mathbb{Z}} : \exists C(G) > 0 : \ |g_{k,k',l,l'}| \leq C(G)(1 + |k-k'| + |l-l'|)^{-\alpha-2}\}$$

definieren. Auch diese Klasse ist abgeschlossen unter Addition und Multiplikation und beinhaltet mit jeder Matrix auch deren Inverse, wie Coult in [17] ausführlich beweist. Also gilt Satz 2.8 entsprechend auch für zweidimensionale Operatoren.

Somit lassen sich sämtliche Rechnungen zur Invertierung des Operators T in seiner Nichtstandardform auf ein (Block-)Diagonalband beschränken, wenn T die Voraussetzungen von Satz 2.6 erfüllt. Daraus resultiert ein Gesamtaufwand von $\mathcal{O}(n)$ zur (approximativen) Berechnung der Inversen in der Nichtstandardform, wenn $n \times n$ die Matrixgröße von T_0 ist.

2.5. Konvergenz

Ziel ist es, die spektrale Äquivalenz zwischen einem mittels Wavelettransformationen gewonnenen Vorkonditionierer $\tilde{M}_k$ und dem zugehörigen waveletzerlegten Operator $\tilde{T}_k$ des ursprünglichen Operators T_0 unabhängig vom Level k zu zeigen [50], da daraus die gitterweitenunabhängige Konvergenz des entsprechenden iterativen Verfahrens folgt [54]. Der Vorkonditionierer $\tilde{M}_k$ ist dabei wie folgt erklärt: zunächst ist $M_{lt} := T_{lt}$. Anschließend definiert man rekursiv für $k = lt-1, lt-2, \ldots, 0$

$$\begin{aligned} \tilde{M}_k &:= \begin{pmatrix} \breve{A}_{k+1} & \\ C_{k+1} & M_{k+1} \end{pmatrix} \begin{pmatrix} Id & \breve{A}_{k+1}^{-1}B_{k+1} \\ & Id \end{pmatrix}, \\ M_k^{-1} &:= \mathcal{W}_{k+1}^k \tilde{M}_k^{-1} {\mathcal{W}_{k+1}^k}^T. \end{aligned}$$

Ferner bezeichnet $\tilde{T}_k$ die Zweiskalentransformierte des Operators T_{k-1}, sodaß das hier betrachtete Verfahren gemäß Abschnitt 2.3.5 äquivalent zum Verfahren ohne Schurkomplementkorrektur ist.

Außerdem sei h_j die Gitterweite auf dem Level j sowie $a(\cdot,\cdot)$ die dem Operator T zugeordnete Bilinearform. Weitere Voraussetzungen sind, daß T_0 die Steifigkeitsmatrix einer elliptischen Randwertaufgabe zweiter Ordnung mit positiv definitem Operator ist, sowie, daß die Teilmatrizen A_k der Nichtstandardform T_{ns} von T_0 im Verlauf einer approximativen Block-Faktorisierung durch spektral äquivalente symmetrische positiv definite Matrizen $\breve{A}_k$ approximiert werden und wohlkonditioniert sind, das heißt:

$$c(\breve{A}_k w_k, w_k) \leq \quad (A_k w_k, w_k) \quad \leq C(\breve{A}_k w_k, w_k) \qquad \forall w_k \in W_k, k = 1, \ldots, lt, \quad (2.9)$$

$$\frac{c}{h_k^2}(w_k, w_k) \leq \quad (A_k w_k, w_k) \quad \leq \frac{C}{h_k^2}(w_k, w_k) \qquad \forall w_k \in W_k, k = 1, \ldots, lt. \quad (2.10)$$

Ferner werden die beiden Ungleichungen

$$\|P_k^{lt} v_k\|_0^2 + \sum_{s=k+1}^{lt} \frac{\|(P_k^{s-1} - P_k^s) v_k\|_0^2}{h_s^2} \leq C \|v_k\|_1^2 \quad \forall v_k \in V_k \qquad (2.11)$$

und

$$|a(v_j, v_s)|^2 \leq \frac{C}{\delta^{2(s-j)} h_s^2} a(v_j, v_j) \|v_s\|_0^2 \quad \forall v_j \in V_j, \quad \forall v_s \in V_s, \quad j \geq s \qquad (2.12)$$

benötigt, wobei $\delta = \frac{h_{k-1}}{h_k}$ das Verhältnis zweier aufeinanderfolgender Gitterweiten beschreibt.

Nun sei zunächst

$$E_k := \tilde{M}_k - \tilde{T}_k.$$

Dann gilt für $v_k \in V_k$ unter Benutzung der Notation $\tilde{v}_k := [w_{k+1}; v_{k+1}]$ für die Zerlegung von v_k in seine Grob- und Feinanteile

$$\begin{aligned} (E_k \tilde{v}_k, \tilde{v}_k) \quad = \quad & \Big((\breve{A}_{k+1} - A_{k+1}) w_{k+1}, w_{k+1} \Big) \\ & + (\breve{A}_{k+1}^{-1} B_{k+1} v_{k+1}, B_{k+1} v_{k+1}) + (E_{k+1} \tilde{v}_{k+1}, \tilde{v}_{k+1}). \end{aligned} \qquad (2.13)$$

Wie sich induktiv zeigen läßt, folgt $E_k \geq 0$. Denn für $k = lt$ gilt $\tilde{M}_{lt} = \tilde{T}_{lt}$ und somit $E_{lt} = 0$, und unter der Annahme, daß $E_{k+1} \geq 0$ gilt, folgert man unter der Ausnutzung von (2.9) und der Tatsache, daß $\breve{A}_{k+1}$ symmetrisch positiv definit ist, direkt mit (2.13) $E_k \geq 0$.

Setzt man (2.13) rekursiv in sich selbst ein, erhält man

$$\begin{aligned} (E_k \tilde{v}_k, \tilde{v}_k) \quad &= \quad \Big((\breve{A}_{k+1} - A_{k+1}) w_{k+1}, w_{k+1} \Big) \\ & \quad + (\breve{A}_{k+1}^{-1} B_{k+1} v_{k+1}, B_{k+1} v_{k+1}) + (E_{k+1} \tilde{v}_{k+1}, \tilde{v}_{k+1}) \\ &= \quad \sum_{s=k+1}^{k+2} \Big((\breve{A}_s - A_s) w_s, w_s \Big) \\ & \quad + \sum_{s=k+1}^{k+2} (\breve{A}_s^{-1} B_s v_s, B_s v_s) + (E_{k+2} \tilde{v}_{k+2}, \tilde{v}_{k+2}) \\ &= \quad \ldots, \end{aligned}$$

und mit $E_{lt} = 0$ folgt

$$(E_k \tilde{v}_k, \tilde{v}_k) = \sum_{s=k+1}^{lt} \Big((\breve{A}_s - A_s) w_s, w_s \Big) + \sum_{s=k+1}^{lt} (\breve{A}_s^{-1} B_s v_s, B_s v_s).$$

Daraus wird mittels der aus (2.9) folgenden Abschätzung

$$\Big((\breve{A}_s - A_s) w_s, w_s \Big) \leq C (A_s w_s, w_s)$$

$$(E_k \tilde{v}_k, \tilde{v}_k) \leq \sum_{s=k+1}^{lt} C (A_s w_s, w_s) + \sum_{s=k+1}^{lt} (\breve{A}_s^{-1} B_s v_s, B_s v_s). \tag{2.14}$$

Nun gilt es, die beiden Terme auf der rechten Seite von (2.14) abzuschätzen:

Satz 2.9 *Sind die Blöcke A_k für $k = 1, \ldots, lt$ wohlkonditioniert ((2.10) gilt), und ist die Abschätzung (2.11) erfüllt, so gilt:*

$$\sum_{s=k+1}^{lt} (A_s w_s, w_s) \leq C_1 (T_k v_k, v_k). \tag{2.15}$$

Beweis:
Aufgrund der Multiskalenzerlegung ist $w_s = (P_k^{s-1} - P_k^s) v_k$. Weiter gilt:

$$\begin{aligned} \textstyle\sum_{s=k+1}^{lt} (A_s w_s, w_s) &\leq C \textstyle\sum_{s=k+1}^{lt} \frac{\|w_s\|_0^2}{h_s^2} && \text{(wegen (2.10))} \\ &\leq C \textstyle\sum_{s=k+1}^{lt} \frac{\|(P_k^{s-1} - P_k^s) v_k\|_0^2}{h_s^2} \\ &\leq C (T_k v_k, v_k) && \text{(wegen (2.11))} \end{aligned}$$

□

Satz 2.10 *Gelten die gleichen Voraussetzungen wie in Satz 2.9, und ist zusätzlich die verschärfte Cauchy-Schwarzsche Ungleichung (2.12) erfüllt, so gilt:*

$$\sum_{s=k+1}^{lt} (\breve{A}_s^{-1} B_s v_s, B_s v_s) \leq C_2 (T_k v_k, v_k). \tag{2.16}$$

Beweis:

1. Nach der einfachen Cauchy-Schwarzschen Ungleichung gilt

$$\begin{aligned}
\|B_s v_s\|_0^2 &= (B_s v_s, B_s v_s) \\
&= \left(\tilde{T}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} B_s v_s \\ 0 \end{pmatrix}\right) \\
&\leq \left\|\tilde{T}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}\right\|_0 \left\|\begin{pmatrix} B_s v_s \\ 0 \end{pmatrix}\right\|_0,
\end{aligned}$$

und daher folgt

$$\|B_s v_s\|_0^2 \leq \left\|\tilde{T}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}\right\|_0^2.$$

2. Es sei $\tilde{G}_j := h_{j+1}^2 \tilde{T}_j$. Dann gilt:

$$\begin{aligned}
h_s^2 \|B_s v_s\|_0^s &\overset{1.}{\leq} h_s^2 \left\|\tilde{T}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}\right\|_0^2 \\
&= a\left(h_s^2 \tilde{T}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right) \\
&= a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right).
\end{aligned}$$

3. Wegen $v_s = \sum\limits_{j=s+1}^{lt+1} w_j$ mit $w_{lt+1} := v_{lt}$ gilt die Gleichheit

$$a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right) = \sum_{j=s+1}^{lt+1} a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, w_j\right).$$

4. Weiter läßt sich wegen $w_j \in V_{s-1} \quad \forall j > s$ jeder einzelne Summand mit der verschärften Cauchy-Schwarzschen Ungleichung (2.12) wie folgt abschätzen:

$$\begin{aligned}
\left|a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, w_j\right)\right|^2 &\overset{(2.12)}{\leq} \frac{C}{\delta^{2(s-j-1)} h_{s-1}^2} a(w_j, w_j) \left\|\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}\right\|_0^2 \\
&= \frac{C h_{s-1}^2}{\delta^{2(s-j)} h_{s-1}^2 h_s^2} a(w_j, w_j) \left\|\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}\right\|_0^2 \\
&\leq \frac{C h_s^2}{\delta^{2(s-j)}} a(w_j, w_j) \left\|\tilde{T}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}\right\|_0^2 \\
&= C \delta^{2(j-s)} a(w_j, w_j) a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right).
\end{aligned}$$

5. Das Ergebnis von 4. wird nun in 3. eingesetzt:

$$\begin{aligned} a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right) &\leq \sum_{j=s+1}^{lt+1} \left| a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, w_j\right)\right| \\ &\overset{4.}{\leq} C \sum_{j=s+1}^{lt+1} \delta^{j-s}[a(w_j, w_j)]^{\frac{1}{2}} \left[a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right)\right]^{\frac{1}{2}} \end{aligned}$$

6. Es folgt direkt

$$\left[a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right)\right]^{\frac{1}{2}} \leq C \sum_{j=s+1}^{lt+1} \delta^{j-s}[a(w_j, w_j)]^{\frac{1}{2}}$$

und damit

$$\begin{aligned} a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right) &\leq C \left(\sum_{j=s+1}^{lt+1} \left(\delta^{\frac{j-s}{2}} \delta^{\frac{j-s}{2}} a(w_j, w_j)^{\frac{1}{2}}\right)\right)^2 \\ &\leq C \frac{1}{1-\delta} \sum_{j=s+1}^{lt+1} \delta^{j-s} a(w_j, w_j), \end{aligned}$$

wobei im letzten Schritt die Konvergenz der geometrischen Reihe ausgenutzt wurde.

7. Nun wird über s aufsummiert:

$$\begin{aligned} \sum_{s=k+1}^{lt} a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right) &\leq C \frac{1}{1-\delta} \sum_{s=k+1}^{lt} \sum_{j=s+1}^{lt+1} \delta^{j-s} a(w_j, w_j) \\ &\leq C \left(\frac{1}{1-\delta}\right)^2 \sum_{j=k+1}^{lt+1} a(w_j, w_j) \\ &= C \left(\frac{1}{1-\delta}\right)^2 \left(\sum_{j=k+1}^{lt} a(w_j, w_j) + a(v_{lt}, v_{lt})\right) \\ &\overset{\text{Satz } 2.9}{\leq} C (T_k v_k, v_k). \end{aligned}$$

8. Die Behauptung folgt dann aus

$$\begin{aligned}
\sum_{s=k+1}^{lt} (\breve{A}_s^{-1} B_s v_s, B_s v_s) &\overset{(2.9),\,(2.10)}{\leq} \sum_{s=k+1}^{lt} h_s^2 \|B_s v_s\|_0^2 \\
&\overset{2.}{\leq} \sum_{s=k+1}^{lt} a\left(\tilde{G}_{s-1}\begin{pmatrix} 0 \\ v_s \end{pmatrix}, \begin{pmatrix} 0 \\ v_s \end{pmatrix}\right) \\
&\overset{7.}{\leq} C(T_k v_k, v_k).
\end{aligned}$$

Satz 2.11 *Gelten (2.9) bis (2.12), so folgt die spektrale Äquivalenz:*

$$(T_k v_k, v_k) \leq (\tilde{M}_k \tilde{v}_k, \tilde{v}_k) \leq (1 + C_1 + C_2)(T_k v_k, v_k) \quad \forall v_k \in V_k. \tag{2.17}$$

Beweis:
Die linke Abschätzung folgt aus $E_k = \tilde{M}_k - \tilde{T}_k \geq 0$.

Die rechte Abschätzung ist das Ergebnis der in den Sätzen 2.9 und 2.10 bewiesenen Relationen für (2.14). □

3. Programmkonzeption

Die in Kapitel 2.3 beschriebenen Verfahren wurden in einem Computerprogramm in der Sprache „C“ umgesetzt. Zur Erzielung eines tatsächlichen $\mathcal{O}(n)$-Verhaltens wurde eine geeignete Datenstruktur gewählt. Ein Vorteil dieser Datenstruktur ist, daß sie die bei den Wavelets verwendete Randperiodisierung unterstützt.

In den folgenden Abschnitten wird das Konzept des Programmes erläutert. Dabei wird zunächst unter 3.1 auf die Behandlung von eindimensionalen Operatoren eingegangen, bevor in 3.2 die darauf aufbauende Behandlung von zweidimensionalen Operatoren dargestellt wird. Ebenfalls wird in diesen beiden Unterkapiteln die Frage aufgegriffen, mit welchen Kosten eine sinnvolle Lösungsstrategie verbunden ist, das heißt, ob die Kosten für einen guten Löser bei $\mathcal{O}(n)$ liegen oder höher sind. Mit einem „guten“ Löser ist hier ein solcher gemeint, der mit wenigen Iterationsschritten oder gleichbedeutend einer geringen Fehlerreduktionsrate die geforderte Lösungsgenauigkeit unterschreitet. Die Vor- und Nachteile verschiedener Konzepte zur Beurteilung der Qualität des Verfahrens anhand der gemessenen Ergebnisse wird in Abschnitt 3.3 erläutert.

3.1. Eindimensionale Operatoren

3.1.1. Datenstruktur

Für ein effizientes Programm ist eine sinnvolle Datenstruktur wesentlich. Da im vorliegenden Fall sämtliche behandelte Operatoren eine Bandstruktur aufweisen, sprich, alle Nichtnulleinträge der zugehörigen Matrix sich auf einem Band um die Diagonale herum befinden, genügt es, dieses Band abzuspeichern.

Gegeben ist also eine $n \times n$-Matrix T mit Diagonaleinträgen $d_1, \ldots, d_n$. Rechts von jedem Diagonaleintrag befinden sich (bis zu) p weitere Nichtnulleinträge, links (bis zu) q, sodaß die

Matrix (für $k \in \{q+1, \ldots, n-p\}$) so aussieht:

$$\begin{pmatrix}
d_1 & r_{1,1} & r_{1,2} & \ldots & r_{1,p} & 0 & \ldots & \ldots & 0 & l_{1,q} & l_{1,q-1} & \ldots & l_{1,1} \\
l_{2,1} & d_2 & r_{2,1} & \ldots & \ldots & r_{2,p} & 0 & \ldots & \ldots & 0 & l_{2,q} & \ldots & l_{2,2} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \ldots & 0 & l_{k,q} & \ldots & l_{k,1} & d_k & r_{k,1} & \ldots & r_{k,p} & 0 & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
r_{n-1,2} & \ldots & r_{n-1,p} & 0 & \ldots & \ldots & 0 & l_{n-1,q} & \ldots & l_{n-1,2} & l_{n-1,1} & d_{n-1} & r_{n-1,1} \\
r_{n,1} & r_{n,2} & \ldots & r_{n,p} & 0 & \ldots & \ldots & 0 & l_{n,q} & \ldots & l_{n,2} & l_{n,1} & d_n
\end{pmatrix}.$$

Die Matrix wird also in jedem Fall periodisch fortgesetzt betrachtet, auch wenn der diskretisierte Operator in der oberen rechten und der unteren linken Ecke Nulleinträge aufweist. Bei der Verwendung von periodisierten Wavelets werden diese beiden Ecken bereits durch die erste Zweiskalentransformation aufgefüllt, sodaß sie bei der Speicherung berücksichtigt werden müssen. Somit bleiben p und q für die gesamte Matrix konstant. Verwendet man hingegen randangepaßte Wavelets, brauchen diese Ecken (im Falle einer nicht periodischen Matrix) nicht mitgespeichert zu werden. Andererseits würde es – gerade bei großen Matrizen – einen höheren Aufwand erfordern, diese Nullen wegzulassen als sie mitzuspeichern.

Es liegt nahe, T in einer $n \times (q+1+p)$-Matrix $\tilde{T}$ mit der folgenden Gestalt abzuspeichern:

$$\begin{pmatrix}
l_{1,q} & \ldots & l_{1,1} & d_1 & r_{1,1} & \ldots & r_{1,p} \\
l_{2,q} & \ldots & l_{2,1} & d_2 & r_{2,1} & \ldots & r_{2,p} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
l_{n,q} & \ldots & l_{n,1} & d_n & r_{n,1} & \ldots & r_{n,p}
\end{pmatrix}.$$

Zur Rekonstruktion der Originalmatrix T aus $\tilde{T}$ benötigt man lediglich die Kenntnis von q, n und $q+1+p$.

Dieses Format wurde außerdem verwendet, weil sich einerseits aus p und q die Breite des Bandes der Wavelettransformierten von T berechnen läßt und andererseits die Zeilenindizes von T und $\tilde{T}$ übereinstimmen. Man hätte genausogut ein Format wählen können, bei dem die Spaltenindizes gleich geblieben wären.

Eine andere Möglichkeit wäre die bekannte Struktur gewesen, bei der zu jedem Nichtnullelement einer Matrix auch seine Koordinaten („Zeile, Spalte, Wert") abgespeichert werden. Aufgrund der im vorliegenden Fall stets gleichen Struktur der behandelten Matrizen hätte dies jedoch überflüssigen Speicherverbrauch bedeutet, da sich die Koordinaten der Elemente einfach herleiten lassen. Ein Nachteil der gewählten Methode ist, daß die Matrixstruktur starr bleiben muß. Man kann also bei der Verwendung einer ILU-Zerlegung nicht ohne weiteres eine Zeilenvertauschung (Pivotisierung) vornehmen. Dies wäre mit dem Format („Zeile, Spalte, Wert") einfacher [23].

3.1.2. Programmablauf

Mittels eines beim Aufruf übergebenen Parameters läßt sich festlegen, welche Aufgabe das Programm ausführt. Zur Auswahl stehen die in 2.3 beschriebenen Verfahren.

Sodann werden die Problemparameter eingelesen. Dazu gehören

- die Leveltiefe, das heißt die Tiefe der Waveletzerlegung,
- die Operatormatrix,
- die rechte Seite beziehungsweise der Lösungsvektor,
- die Anzahl der Bänder links und rechts der Diagonalen, um die (in der Banderweiterungsversion) auf jedem Level erweitert werden soll,
- die gewünschte Genauigkeit ε der Lösung,
- sowie die maximale Iterationszahl.

Ferner lassen sich verschiedene Wavelets (Daubechies 2 bis 8) sowie Prewavelets einstellen.

Wie bereits in Abschnitt 2.3.1 beschrieben wurde, wird ein iteratives Verfahren eingesetzt. In jeder Iteration werden der Fehler, die Fehlerreduktionsrate des Verfahrens und das Residuum in der L_2-Norm gemessen. Fehler und Reduktionsrate werden zusammen mit der aktuellen Iterationszahl ausgegeben.

Außerdem gibt das Programm die Laufzeit aus.

Algorithmus 3.1 gibt den Programmablauf wieder. Wird mit vollbesetzten Matrizen gerechnet, so wird auf diesen keine ILU- sondern eine LU-Zerlegung verwendet.

3.1.3. Effizienz

In diesem Abschnitt wird erklärt, warum das Band bei Erweiterung um eine konstante Anzahl von Spalten auf jedem Level beidseitig der Diagonalen eine bestimmte Dicke nie übersteigt, sodaß sich eine obere Schranke für den Aufwand ergibt. Außerdem wird erläutert, warum ab einer gewissen Problemgröße trotz eines sehr breiten Bandes noch ein Rechenaufwand von $\mathcal{O}(n)$ möglich ist.

Im Ausgangszustand hat das Band eine Breite von

$$q + 1 + p.$$

In der Banderweiterungsversion wird es noch auf diesem Level um e_l Elemente links der Diagonalen und um e_r Elemente rechts der Diagonalen auf

$$q + e_l + 1 + p + e_r$$

Eingabedaten: Operator, rechte Seite b, Parameter
Ausgabedaten: Laufzeit, Konvergenzrate

1. Initialisieren der verwendeten Wavelets
   Daten einlesen und Speicher reservieren
   speichere die Ausgangsmatrix in T_0
   speichere die rechte Seite in v_0
2. `for` $i = 1$ bis lt
   transformiere T_{i-1} zu A_i, B_i, C_i und T_i
   führe eine ILU-Zerlegung von A_i zu L_i und U_i durch
   falls Verfahren 1 gewählt wurde: berechne $\hat{C}_i$, $\hat{B}_i$ und $T_i = T_i - \hat{C}_i \cdot \hat{B}_i$
   3. transformiere v_{i-1} zu w_i und v_i
   glätte w_i mittels einer Vorwärtssubstitution mit L_i
   korrigiere v_i durch $v_i = v_i - C_i \cdot w_i$ (bzw. $\hat{C}_i$)
   4. falls das Band auf T_{i+1} zu breit würde:
   konvertiere T_i in eine volle Matrix und rechne ab jetzt
   bis $i = lt$ mit vollen Matrizen
   `endfor`
5. führe eine ILU-Zerlegung von T_{lt} zu $L_{T_{lt}}$ und $U_{T_{lt}}$ durch
6. glätte v_{lt} mittels einer Vorwärtssubstitution mit $L_{T_{lt}}$
   glätte v_{lt} mittels einer Rückwärtssubstitution mit $U_{T_{lt}}$
7. `for` $i = lt$ bis 1 (rückwärts laufend)
   korrigiere w_i durch $w_i = w_i - B_i \cdot v_i$ (bzw. $\hat{B}_i$)
   glätte w_i mittels einer Rückwärtssubstitution mit U_i
   transformiere w_i und v_i zu v_{i-1}
   `endfor`
8. $v_0^0 := v_0$
9. `for` $j = 1$ bis maximale Iterationszahl $maxit$
   berechne das Residuum als $res = T_0 \cdot v_0^{j-1} - b$
   $s := v_0^{j-1}$; $v_0^{j-1} := res$
   `for` $i = 1$ bis lt
   führe Schritt 3. mit v_i^{j-1} durch
   `endfor`
   führe Schritt 6. mit v_{lt}^{j-1} durch
   führe Schritt 7. mit v_i^{j-1} und w_i durch
   berechne die neue Iterierte $v_0^j = s - v_0^{j-1}$
   berechne den Fehler und die Reduktionsrate und gib beide aus
   falls der Fehler kleiner als ε ist: $j := maxit$
   `endfor`
10. Speicher freigeben und Laufzeit ausgeben

Algorithmus 3.1: Programmablauf des eindimensionalen iterativen Lösers

Elemente erweitert.

Lemma 3.1 *Ist q die Breite eines Halbbandes rechts oder links der Diagonalen einer Diagonalbandmatrix, so hat dieses Halbband nach einer Zweiskalentransformation eine Breite von*

$$\frac{q + (q \bmod 2) + fl - 2}{2} = \frac{q + (q \bmod 2) + fl}{2} - 1, \tag{3.1}$$

wobei fl die Länge des verwendeten Waveletfilters bezeichnet. Es wird davon ausgegangen, daß fl gerade ist.

Beweis:
Betrachtet wird die erste Zeile der zu transformierenden Matrix im vollen Format:

$$[\ d_1 \quad r_{1,1} \quad \ldots \quad r_{1,p} \quad 0 \quad \ldots \quad 0 \quad l_{1,q} \quad \ldots \quad l_{1,1}\ ].$$

Der Beweis wird anhand der q Elemente $l_{1,q}$ bis $l_{1,1}$ geführt. Die Idee dieses Beweises ist es, sich zu fragen, wie viele Elemente nach der Zweiskalentransformation am rechten Rand der ersten Zeile stehen, wenn dort vor der Transformation q Stück standen. Die Zweiskalentransformation setzt sich zusammen aus einer vertikalen und einer horizontalen Filterung, deren Reihenfolge unerheblich ist, da der Filterungsprozeß eine lineare Transformation ist. Betrachtet man zuerst den vertikalen Filterungsprozeß, so stellt man fest, daß er an der Zahl der am rechten Rand der ersten Zeile stehenden Elemente nichts ändert, denn

- entweder setzt der Filter in den Zeilen am unteren Rand der Matrix auf und trifft wegen der Periodisierung auf Elemente aus der ersten Zeile, dann werden die Filterungsergebnisse auch in den unteren Zeilen gespeichert,
- oder der Filter setzt in der ersten Zeile auf, trifft dort aber nur auf eine Null, dann ist das Filterungsergebnis 0, da auch in den $fl - 1$ Zeilen unterhalb der ersten Zeile Nullen stehen, (unter der Prämisse, daß der Filter nicht auf das Hauptdiagonalband trifft, was jedoch zur Folge hätte, daß man ohnehin eine volle Matrixtransformation mit einer (nahezu) vollbesetzten Ergebnismatrix durchführen müßte, sodaß die Bandbetrachtung dann gar keinen Sinn mehr hätte),
- oder der Filter trifft am rechten Rand der ersten Zeile auf eine Zahl ungleich Null, dann ist auch im allgemeinen das Filterungsergebnis ungleich Null.

Somit bleibt das Besetzungsmuster des rechten Randes der ersten Zeile bei der vertikalen Filterung erhalten. Es bleibt zu klären, wie es nach der horizontalen Filterung aussieht. Man unterteilt nach der Parität von q in zwei Fälle. Ist q gerade, so trifft der Filter in der ersten Zeile zum ersten Mal auf ein Element ungleich Null, wenn er $fl - 2$ Elemente vor diesem

Element aufsetzt. Die Länge des gefilterten Bereiches mit Ergebnissen (im allgemeinen) ungleich Null am rechten Rand der ersten Zeile ist dann also $q + fl - 2$. Ist q ungerade, so trifft der Filter schon auf ein Nichtnullelement, wenn er $fl - 3$ Elemente vor diesem aufsetzt, sodaß die Länge des gefilterten Bereiches dann $q + 1 + fl - 2$ beträgt. Man faßt beide Fälle in der Formel $q + (q \bmod 2) + fl - 2$ zusammen. Da der Filter mit jedem Aufsetzen um zwei Elemente weitergerückt wird, erzeugt er halb so viele neue Elemente, nämlich

$$\frac{q + (q \bmod 2) + fl - 2}{2}.$$

□

Nach der Erweiterung des Bandes auf $q + e_l + 1 + p + e_r$ Elemente erfolgt eine Zweiskalentransformation, nach der das Band also eine Breite von

$$\frac{q + e_1 + (q + e_1 \bmod 2) + fl - 2}{2} + 1 + \frac{p + e_2 + (p + e_2 \bmod 2) + fl - 2}{2}$$

aufweist.

Da die Dicke der rechten und der linken Seite des Bandes nach der selben Formel berechnet wird, reicht es, eine Seite zu betrachten: ist die Dicke der linken Seite auf dem Level i vor Transformation

$$x_i := q + e_1,$$

so berechnet sich x_{i+1} als

$$\begin{aligned} x_{i+1} &= e_1 + \frac{x_i + (x_i \bmod 2) + fl - 2}{2} \\ &= \underbrace{e_1 - 1 + \frac{fl}{2}}_{=:k} + \frac{x_i + (x_i \bmod 2)}{2} \\ &= k + \frac{x_i + (x_i \bmod 2)}{2}. \end{aligned}$$

Diese Folge konvergiert gegen $2k = 2e_1 + fl - 2$. Für Haarwavelets ist dies beispielsweise gerade $2e_1$. Geht man o.B.d.A. davon aus, daß

$$p \leq q \quad \text{und} \quad e_2 \leq e_1$$

gelten, so folgt daraus, daß die Gesamtbanddicke gegen $4k + 1$ (oder einen kleineren Wert) konvergiert.

Das bedeutet, daß der Rechenaufwand nicht unbegrenzt levelabhängig steigt. Andererseits verdoppelt sich mit jeder Verfeinerung des Ausgangslevels die Zahl der Gitterpunkte und somit die Größe der zugehörigen vollen Matrix. Bei 16 Gitterpunkten muß das Band (deutlich) weniger als 16 Elemente aufweisen, damit fehlerfrei auf diesem und dem nächstgröberen Level mit der Bandversion gerechnet werden kann. Besitzt das feinste Level hingegen 256 Gitterpunkte, so kann man auch mit einer Bandbreite von 32 noch (mindestens zwei Level weiter) auf dem Band rechnen.

3.1.4. Kosten und Nutzen des Verfahrens

Anhand der in 3.1.3 ermittelten Bandbreiten werden hier zunächst die durch die Bandverbreiterung entstehenden Kosten im allgemeinen berechnet, bevor mittels der im experimentellen Teil festgestellten optimalen Bandverbreiterung die für eine gute Lösung tatsächlich anfallenden Kosten hergeleitet werden. Das Verfahren ohne Schurkomplementkorrektur verursacht zwar geringere Kosten, liefert dafür aber auch ein nur bedingt nutzbares Resultat (siehe Abschnitt 4.1). Daher wird hier ausschließlich das Banderweiterungsverfahren behandelt.

Die Kosten des Programmteils, in dem auf der vollen (Unter-) Matrix gerechnet wird, betragen $\mathcal{O}(n^3)$, wenn n die Zahl der Unbekannten auf dem aktuellen Level darstellt. Auf Level l ist die Ausgangsmatrix jedoch nicht von der Größe $n \times n$, sondern von der Größe $\frac{n}{2^l} \times \frac{n}{2^l}$, sodaß die Kosten für die Behandlung der vollen Matrix auf Level l bezogen auf die Gesamtproblemgröße n im Bereich $\mathcal{O}(\frac{n^3}{2^{3l}})$ liegen.

Da die Kosten für die Berechnung des Lösungsvektors beziehungsweise der neuen Iterierten bei $\mathcal{O}(n \cdot b)$ liegen, wobei b die Breite des Bandes bezeichnet, sind sie im Vergleich zur Berechnung der Bandapproximation des Schurkomplementoperators, die mit $\mathcal{O}(n \cdot b^2)$ zu Buche schlägt, zu vernachlässigen. Jene wird zwar im Programmverlauf nur genau einmal auf jedem Level berechnet, verursacht aber die höchsten Kosten, zumal sie nach Erweiterung des Bandes durch eine „dünne" Matrix-Matrix-Multiplikation ermittelt wird. Das bedeutet, daß man sich die Bandbreite b näher ansehen muß, um Aufschluß darüber zu erhalten, welche Kosten hier entstehen.

Die Ausgangsbandweite vor der ersten Zerlegung liegt bei $b_0 = q_0 + 1 + p_0$. Es wird davon ausgegangen, daß $q_0 = p_0$ gilt und daß ein Waveletfilter mit gerader Filterlänge verwendet wird. Da vor der ersten Zerlegung keine Banderweiterung stattfindet, ist die Bandbreite vor Zerlegung $2q_0 + 1$. Gemäß der oben angegebenen Formel (3.1) für die Transformation $q_{i+1} = \frac{q_i + (q_i \bmod 2) + fl - 2}{2}$ gilt für die Bandbreite

$$\begin{aligned} b_1 &= 2 \cdot \frac{q_0 + (q_0 \bmod 2) + fl - 2}{2} + 1 \\ &= q_0 + fl - 1 + (q_0 \bmod 2) \\ &\leq q_0 + fl. \end{aligned} \tag{3.2}$$

Im folgenden wird $q_0 \bmod 2$ aus Rechnung (3.2) zu 1 abgeschätzt, sodaß sich $b_1 = q_0 + fl$

ergibt. Durch die Banderweiterung um $2e$ Elemente wird aus b_1

$$b_{1,erw} = q_0 + fl + 2e.$$

Nun erfolgt die Transformation auf das zweite Level. Für die Ermittlung der resultierenden Bandbreite wird zunächst die Diagonale berücksichtigt: $b_{1,erw} = q_0 + fl + 2e - 1 + 1$. Das Band enthält also $q_0 + fl + 2e - 1$ Elemente rechts und links der Diagonalen, davon jeweils die Hälfte auf einer der beiden Seiten, also

$$2 \underbrace{\left(\frac{q_0}{2} + \frac{fl}{2} + e - \frac{1}{2} \right)}_{=:q_1},$$

die Gesamtbreite $b_{1,erw}$ vor Transformation läßt sich also als $2q_1 + 1$ darstellen, sodaß man für b_2 gemäß Rechnung (3.2) erhält:

$$\begin{aligned} b_2 &\leq q_1 + fl \\ &= \frac{q_0}{2} + \frac{3}{2} fl + e - \frac{1}{2}. \end{aligned} \tag{3.3}$$

Abermals wird $q_1 \bmod 2$ durch 1 abgeschätzt, daher das „$\leq$" in der ersten Zeile von Rechnung (3.3).

Durch die Erweiterung auf dem zweiten Level wird daraus $b_{2,erw} = \frac{q_0}{2} + \frac{3}{2} fl + 3e - \frac{1}{2}$, bevor wieder zerlegt wird, usw.

Wie bereits oben bemerkt wurde, ist der Aufwand zur approximativen Berechnung des Schurkomplementoperators aufgrund der dabei nötigen Matrix-Matrix-Multiplikation $\mathcal{O}(n \cdot b^2)$. Insbesondere ist darin ein Term der Form $\frac{1}{2} ne^2$ enthalten, da die Berechnung der Approximation an den Schurkomplementoperator bei bereits erweitertem Band erfolgt. Der Faktor $\frac{1}{2}$ erklärt sich dadurch, daß der Schurkomplementoperator halb so viele Zeilen aufweist wie die Ausgangsmatrix.

Aufgrund der Tatsache, daß ab dem zweiten Level die Bandbreite b_i bereits vor der auf dem aktuellen Level i stattfindenden Erweiterung einen Ausdruck der Form $j \cdot e,\ j \in \mathbb{Q}$, enthält, geht die Banderweiterung nun nicht mehr nur bei der Berechnung des Schurkomplementoperators quadratisch in den Gesamtaufwand ein. So enthalten die Kostenausdrücke für die Berechnung der ILU-Zerlegung und der Matrizen $\hat{B}$ und $\hat{C}$ für $i > 1$ ebenfalls Terme der Form $j \cdot ne^2,\ j \in \mathbb{Q}$, sodaß auf jedem Level i die Kosten bezüglich der Banderweiterung durch Terme $j_i \cdot ne^2,\ j_i \in \mathbb{Q}$ beschrieben werden. Es gilt

$$\begin{aligned} j_1 &= \frac{1}{2}, \\ j_2 &= \frac{25}{16} = 1\frac{9}{16} \text{ und} \\ j_3 &= \frac{121}{128}. \end{aligned}$$

Folglich handelt es sich hierbei nicht um Kosten, die durch die Vorfaktoren j_i geschluckt würden. Zwar ist zu erwarten, daß für $i > 1$ die Beziehung $j_i > j_{i+1}$ gilt, jedoch ist durch den hohen Faktor j_2 eine Berücksichtigung dieser Kosten für den Gesamtaufwand des Verfahrens unausweichlich.

Aus diesen Betrachtungen folgt, daß ein $\mathcal{O}(n)$-Löser mit der Banderweiterunsstrategie nur möglich ist, wenn die Banderweiterung e unabhängig von der Problemgröße $n = 2^l$ gewählt werden kann. Wie in den numerischen Experimenten festgestellt wurde, erhält man ab einer Banderweiterung von $e = l-2$ stets ein konvergentes Verfahren, wobei $l = \log_2(n)$ gilt. Setzt man dies in die Kostenberechnung ein, so erhält man einen führenden Term der Ordnung $n(\log_2(n))^2$.

Somit ist die Banderweiterungsmethode kein $\mathcal{O}(n)$-Löser, sondern ein $\mathcal{O}(n(\log_2(n))^2)$-Löser.

3.2. Zweidimensionale Operatoren

3.2.1. Datenstrukturen

Ein zweidimensionaler Vektor kann als Matrix dargestellt werden. Da dies auch für Vektoren in der Nichtstandardform gilt, werden die zweidimensionalen Vektoren als eindimensionale vollbesetzte Matrizen gespeichert und behandelt. Somit wird statt mit einer eigenen Transformationsroutine der Vektor jetzt mittels der Routine für eindimensionale volle Matrizen transformiert.

Für die Operatormatrizen wird das dünne Datenformat aus dem Eindimensionalen zu einem entsprechenden Metaformat erweitert. Da Matrizen von zweidimensionalen Operatoren eine Blockstruktur aufweisen, muß diese entsprechend aufgelöst werden. Zusätzlich zu den beiden aus dem Eindimensionalen bekannten Variablen p und q werden zwei Variablen qb und pb definiert. (Das „b“ steht für „Block“.)

Als Beispiel wird der mittels finiter Differenzen auf $[0,1]^2$ diskretisierte zweidimensionale Laplaceoperator (bei lexikographischer Gitteranordnung) betrachtet. Es ergibt sich eine Block-Tridiagonal-Matrix T

$$T = \frac{1}{h^2} \begin{pmatrix} G & -Id & & & & \\ -Id & G & -Id & & & \\ & -Id & G & -Id & & \\ & & \ddots & \ddots & \ddots & \\ & & & -Id & G & -Id \\ & & & & -Id & G \end{pmatrix}$$

mit den tridiagonalen Blöcken

$$G = \begin{pmatrix} 4 & -1 & & & & \\ -1 & 4 & -1 & & & \\ & -1 & 4 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 4 & -1 \\ & & & & -1 & 4 \end{pmatrix}$$

und der negativen Einheitsmatrix. Die Blöcke der Matrix T werden nun gemäß dem Vorgehen beim eindimensionalen Datenformat als Elemente betrachtet und in eine Reihe „geschoben“:

$$T_{sparse} = \frac{1}{h^2} \begin{pmatrix} & G & -Id \\ -Id & G & -Id \\ \vdots & \vdots & \vdots \\ -Id & G & -Id \\ -Id & G & \end{pmatrix}.$$

Ebenso werden die einzelnen Blöcke wie eindimensionale Matrizen behandelt:

$$\tilde{G} := G_{sparse} = \begin{pmatrix} & 4 & -1 \\ -1 & 4 & -1 \\ \vdots & \vdots & \vdots \\ -1 & 4 & -1 \\ -1 & 4 & \end{pmatrix}, \tilde{I} := -Id_{sparse} = \begin{pmatrix} & -1 & 0 \\ 0 & -1 & 0 \\ \vdots & \vdots & \vdots \\ 0 & -1 & 0 \\ 0 & -1 & \end{pmatrix},$$

sodaß die Matrix T als

$$\frac{1}{h^2} \begin{pmatrix} & \tilde{G} & \tilde{I} \\ \tilde{I} & \tilde{G} & \tilde{I} \\ \vdots & \vdots & \vdots \\ \tilde{I} & \tilde{G} & \tilde{I} \\ \tilde{I} & \tilde{G} & \end{pmatrix}$$

abgespeichert wird. Die Nullen der Matrizen $\tilde{I}$ werden explizit mitgespeichert, da man auf diese Weise die gleiche Struktur speichert wie bei $\tilde{G}$, was den Vorteil hat, daß zwei Parameter q und p zur Beschreibung der Struktur auf den Blöcken sowohl von $\tilde{G}$ als auch von $\tilde{I}$ ausreichen. Zur Beschreibung der Blockstruktur werden die Parameter qb und pb verwendet. Für das oben behandelte Beispiel des Laplaceoperators erhält man

$$qb = pb = q = p = 1.$$

Würde man anstelle des Fünf-Punkte-Sternes

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix} \tag{3.4}$$

einen Neun-Punkte-Stern verwenden, so bekäme man denselben Speicheraufwand.

3.2.2. Programmablauf

Der Programmablauf ist der gleiche wie im eindimensionalen Fall, nur daß jetzt die Faktorisierung auf jedem Level nach der unter 2.3.3 beschriebenen Methode abläuft beziehungsweise die entsprechenden Routinen für den eindimensionalen Fall durch solche für die zweidimensionale Version ersetzt wurden.

3.2.3. Effizienz

Gemäß den oben gemachten Beobachtungen steigt auch bei zweidimensionalen Operatoren der Rechenaufwand nicht unbegrenzt levelabhängig an. Allerdings verursacht eine Banderweiterung um e Elemente in allen vier Teilen des (Block-)Bandes jetzt einen erheblichen Mehraufwand, denn die Dicke des gesamten berechneten Bandes ergibt sich aus

$$(q+1+p)\cdot(qb+1+pb),$$

beziehungsweise nach erfolgter Banderweiterung um e :

$$(q+1+p+2e)\cdot(qb+1+pb+2e).$$

Es ergibt sich also gegenüber $e=0$ eine Verbreiterung um

$$2e\cdot(qb+1+pb+q+1+p)+4e^2,$$

die sich im Rechenaufwand und somit auch in der Laufzeit deutlich bemerkbar macht. Daraus läßt sich direkt die Forderung ableiten, mit der Banderweiterung sparsam umzugehen, und zwar unabhängig von der Ausgangsgitterweite. Da sich bei zweidimensionalen Operatoren mit Halbierung der Gitterweite die Punktezahl vervierfacht, gilt auch hier, daß man bei Verwendung eines feineren Diskretisierungsgitters das Band zunächst stärker verbreitern kann als bei Verwendung eines gröberen.

3.2.4. Kosten und Nutzen des Verfahrens

Ebenso wie bei der eindimensionalen Banderweiterungsvariante treten für die zweidimensionale Banderweiterungsvariante in Abhängigkeit der Verbreiterung zusätzliche Kosten auf, sodaß die Gesamtkosten über $\mathcal{O}(n)$ hinausgehen. Setzt man wie für eindimensionale Operatoren die Anzahl der Elemente, um die erweitert wird, auf $l-2$, wobei die Problemgröße 4^l beträgt, so gelangt man wegen $l = \log_2(\sqrt{n})$ zu einem Aufwand, der von

$$\mathcal{O}\left(n(\log_2(\sqrt{n}))^4\right)$$

dominiert wird. Der Wert $l-2$ wird hierbei durch die numerischen Ergebnisse motiviert.

Verallgemeinert man diese Kostenabschätzung bezüglich der Dimension d, gelangt man zu Kosten in Höhe von

$$\mathcal{O}\left(n(\log_2(n^{\frac{1}{d}}))^{2d}\right).$$

Diese Abschätzung ist insofern hypothetisch, als daß sie sich nicht durch alle numerischen Ergebnisse bestätigen ließ. Für den Konvektionsdiffusionsoperator konnte bei Konvektion in x-Richtung genauso wie bei Konvektion in y-Richtung mit dem Banderweiterungsverfahren in der getesteten Version kein zufriedenstellendes Konvergenzverhalten erzielt werden. Dennoch läßt das am Ende von 4.2.2 beschriebene Resultat die Möglichkeit offen, daß sich auch ein schlecht konditioniertes Problem zu höchstens den oben beschriebenen Kosten lösen läßt.

In [17] kommt Coult zu dem Ergebnis, daß sich die Bandapproximation des Schurkomplementoperators in $\mathcal{O}(n^{1+\frac{4}{M}})$ berechnen läßt, wobei M die Anzahl der verschwindenden Momente der verwendeten Wavelets ist. Für $M = 2$ erhält man auf diese Weise einen Aufwand von $\mathcal{O}(n^3)$, sodaß diese Abschätzung beispielsweise für den Laplaceoperator etwas zu grob ist.

3.3. Zur Bewertung der Ergebnisse

Zunächst ist zwischen zwei verschiedenen Verfahrenseigenschaften zu unterscheiden: als *effizientes Iterationsverfahren* soll ein Verfahren bezeichnet werden, das für jede Iteration einen Aufwand von $\mathcal{O}(n)$ benötigt. Ein *optimaler Löser* ist in der Lage, für jede Problemgröße n mit Gesamtkosten in Höhe von $\mathcal{O}(n)$ zu lösen.

Zur Beurteilung der Qualität eines iterativen Lösungsverfahrens gibt es sodann mehrere Kriterien [31]:

1. Die benötigte Zeit:
   Mit Hilfe der Zeitmessung läßt sich einerseits feststellen, ob das programmierte Verfahren ein effizientes Iterationsverfahren ist. Voraussetzung dafür ist, daß man es auf einem Rechner gleicher Architektur (Prozessorleistung, Speicher, ...) und bei gleicher sonstiger Rechnerauslastung (Speicherverbrauch) (möglichst mehrmals) über eine feste Anzahl von Iterationen bei verschiedenen Problemgrößen startet.

   Die zweite Frage, für deren Beantwortung die Zeitmessung hilfreich sein kann, ist die, ob das programmierte Verfahren tatsächlich ein optimaler Löser ist. Die Rechner-

Voraussetzungen sind die gleichen wie oben, allerdings kann das programmierte Verfahren zum Lösen nur dann lineare Komplexität besitzen, wenn es ein effizientes Iterationsverfahren ist.

2. Die Zahl der benötigten Iterationen zur Erreichung einer vorgegebenen Lösungsgenauigkeit:
   Unter Kenntnis der Zahl der benötigten Iterationen ist es möglich, mit Hilfe des im letzten Schritt gemessenen Fehlers und des Anfangsfehlers die durchschnittliche Fehlerreduktionsrate zu bestimmen. Voraussetzung für die Messung des Fehlers ist die Vorgabe der Lösung x. Über eine Multiplikation mit der Operatormatrix A erhält man die zugehörige rechte Seite b.

   Außerdem kann die Iterationszahl für die Ermittlung einer Lösungsstrategie, mit der man in $\mathcal{O}(n)$ lösen möchte, eine Rolle spielen.

3. Die Fehlerreduktionsrate:
   Die (Fehler-)Reduktionsrate von Schritt $i-1$ zum Schritt i ist gemäß [31] definiert als

$$\varrho_{i,i-1} := \frac{\|x_i - x\|}{\|x_{i-1} - x\|},$$

   wobei x_i die Iterierte im Schritt i und x die Lösung ist. Das geometrische Mittel der Reduktionsraten ergibt sich für k Iterationsschritte als

$$\varrho_{i+k,i} := \left(\frac{\|x_{i+k} - x\|}{\|x_i - x\|}\right)^{\frac{1}{k}}.$$

   Als Norm wurde die problemunabhängige l_2-Norm verwendet. Laut [31] konvergiert das geometrische Mittel der Reduktionsraten für $k \to \infty$ gegen die Konvergenzrate des Verfahrens, sodaß das Mittel eine zuverlässige Aussage über die Konvergenzrate ermöglichen sollte.

   Gemessen wurde die Reduktionsrate im letzten Schritt. Dies hat gegenüber der Messung des geometrischen Mittels (der durchschnittlichen Reduktionsrate) den Vorteil, daß man ungefähr erkennen kann, wohin man mit weiteren Iterationsschritten kommen kann. Es bestehen Abhängigkeiten von der maximalen Zahl der Iterationsschritte und von der gewünschten Lösungsgenauigkeit. Gerade in den getesteten Fällen kommt es oft zu der Situation, daß das Verfahren in den ersten l Iterationsschritten den Fehler nicht reduziert, bevor sich die Reduktionsrate dann immer weiter verbessert. Auf diese Weise kann es geschehen, daß man eine letzte Reduktionsrate von 0.05 mißt, obwohl man 24 Iterationen benötigt hat, um die geforderte Genauigkeit zu erreichen. In so einem Fall liegt das geometrische Mittel der Reduktionsraten natürlich deutlich über 0.05 und ist somit aussagefähiger.

Bei der später eingesetzten Messung des geometrischen Mittels spielt es eine wesentliche Rolle, über wie viele Iterationen gemittelt wird. Setzt man die maximale Iterationszahl auf 50, so wird man im oben geschilderten Fall bessere Reduktionsraten bekommen, als wenn man sie auf 20 gesetzt hätte. Insofern sollte man zusätzlich die Iterationszahl messen, um feststellen zu können, ob das Verfahren innerhalb der 50 maximalen Iterationen die Grenzgenauigkeit erreicht hat oder nicht.

Andererseits läßt sich die durchschnittliche Reduktionsrate bei Kenntnis des Endfehlers und der Iterationszahl bestimmen, da man den Anfangsfehler in Form von „Lösung minus Startvektor" kennt, wobei man die Lösung vorgegeben hat, um den Fehler in jedem Schritt i bestimmen zu können. Folglich ist es am besten, sowohl die durchschnittliche Reduktionsrate als auch die Reduktionsrate im letzten Schritt zu kennen, um das Verfahren beurteilen zu können.

Für die Darstellung der numerischen Ergebnisse in Kapitel 4 wurde das geometrische Mittel der Fehlerreduktionsraten mittels des Endfehlers und der Zahl der Iterationen berechnet, da sich die Verfahrensvarianten so besser miteinander vergleichen lassen.

4. Numerische Beispiele

Es wurden zwei der in 2.3 beschriebenen Methoden für den eindimensionalen Laplace- und den eindimensionalen Konvektionsdiffusionsoperator mit Konvektion $b_x \in \mathbb{R}$ getestet. Die Ergebnisse der Verfahren 1 und 2 werden unter 4.1 dargestellt.

Es stellte sich die Frage, ob sich der Algorithmus für zweidimensionale Operatoren genauso verhalten würde. Abschnitt 4.2 beschäftigt sich mit diesem Thema. Dabei wurde zunächst ebenfalls der Laplaceoperator untersucht (Abschnitt 4.2.1), bevor der Konvektionsdiffusionsoperator mit verschiedenen Konvektionsrichtungen und -stärken $b_x, b_y \in \mathbb{R}$ betrachtet wurde (Abschnitt 4.2.2). In Abschnitt 4.3 erfolgt schließlich eine zusammenfassende Bewertung der Resultate.

Die untersuchten Operatoren, der Laplaceoperator

$$\Delta = \frac{\partial^2}{\partial x^2} \left(+ \frac{\partial^2}{\partial y^2} \right)$$

sowie der Konvektionsdiffusionsoperator

$$\Delta + \{b_x(, b_y)\} \cdot \nabla, \ \nabla = \left\{ \begin{array}{c} \frac{\partial}{\partial x} \\ \left(\frac{\partial}{\partial y} \right) \end{array} \right\},$$

wurden jeweils mittels finiter Differenzen diskretisiert. Für den konvektiven Anteil $b\nabla$ des Konvektionsdiffusionsoperators wurde eine stabile Upwind-Diskretisierung verwendet. (Die mit runden Klammern bezeichneten Ausdrücke gehören nur zu den zweidimensionalen Operatoren.)

Die Operatormatrizen wurden stets in der nichtperiodisierten Form aufgestellt. Die periodisierte Form wäre, da ihr Rang $n-1$ ist, mit den gegebenen Verfahren nicht lösbar gewesen. Andererseits ist die Wavelettransformation für periodische Operatoren ausgelegt. Ein randangepaßter Filter, wie er bei der Prewaveletvariante verwendet wurde, ist bei nicht periodisierten Operatoren vorteilhafter, da er keinen Rechenaufwand für die oberen rechten und unteren linken Ecken der Matrizen verursacht. Außerdem bewirkt eine solche Randanpassung schon vom Besetzungsmuster her eine bessere Approximation an den Feingitteroperator. Ein Nachteil ist allerdings, daß durch die Randanpassung der Algorithmus komplizierter wird, falls direkt algebraisch transformiert wird. Realisiert man die Transformation über Matrizenmultiplikation, verlangt die Randanpassung nur beim Aufstellen der Transformationsmatrizen Beachtung.

Die Prewavelettransformation wurde mittels des starren randangepaßten Feinfilters

$$\begin{bmatrix} \frac{9}{10} & -\frac{3}{5} & \frac{1}{10} \end{bmatrix}, \begin{bmatrix} \frac{1}{10} & -\frac{3}{5} & 1 & -\frac{3}{5} & \frac{1}{10} \end{bmatrix}, \begin{bmatrix} \frac{1}{10} & -\frac{3}{5} & \frac{9}{10} \end{bmatrix}$$

umgesetzt. Die Ausdrücke links und rechts sind die Filter für den linken und rechten Rand der Matrix. Ihr Aufsatzpunkt ist jeweils beim Wert $\frac{9}{10}$, also am linken beziehungsweise rechten Rand des Filters und somit auch der Matrix. Der in der Mitte dargestellte Filter für das Innere der Matrix hat seinen Aufsatzpunkt in der Mitte beim Wert 1. Für den Grobfilter wurde der zur skalierten „Hutfunktion" gehörende Filter verwendet:

$$\begin{bmatrix} \frac{1}{2\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{2\sqrt{2}} \end{bmatrix}.$$

Gegeben war jeweils der entsprechende diskretisierte Operator und eine konstante Lösung. Nach der Berechnung der zugehörigen rechten Seite wurde mittels eines iterativen Verfahrens der Form

$$x_{i+1} = x_i - M(Ax_i - b)$$

mit Prewavelets sowie verschiedenen Wavelets (Daubechies2 bis Daubechies8) versucht, in möglichst kurzer Zeit die Lösung bis auf einen maximalen Fehler

$$\|x_i - x\|_2 < \varepsilon$$

zu bestimmen. Die maximale Iterationszahl betrug 50, Startvektor war der Nullvektor. Zusätzlich wurden einige Läufe mit einer zufälligen nichtkonstanten Lösung durchgeführt, wodurch sich am qualitativen Verhalten der Verfahren jedoch nichts änderte.

Zur Ermittlung der approximativen Inversen M wurden

1. das unter 2.3.1/2.3.3 beschriebene Verfahren als Bandversion mit möglicher Bandverbreiterung um eine feste Spaltenzahl auf jedem Level mit Wavelets sowie

2.a) das unter 2.3.2/2.3.4 beschriebene Verfahren ohne Schurkomplementkorrektur mit Wavelets und

b) mit Prewavelets eingesetzt.

4.1. Eindimensionale Operatoren

4.1.1. Laplaceoperator

Für den Laplaceoperator wurde

$$\varepsilon = 10^{-11}$$

verwendet.

Im einzelnen konnten folgende Resultate beobachtet werden:

Beide Verfahren sind effiziente Iterationsverfahren, das heißt die benötigte Zeit steigt bei gleicher Iterationszahl linear mit der Zahl der Unbekannten. Aus der Tatsache, daß zusätzliche Iterationen zusätzliche Zeit verbrauchen, folgt die Forderung nach einem Verfahren, das mit möglichst wenigen Iterationen zur Lösung mit hinreichender Genauigkeit auskommt. Andererseits entsteht bei Benutzung der Banderweiterung mit breiter werdendem Band auch ein höherer Rechenaufwand. Diese beiden Punkte sind gegeneinander abzuwägen.

Tabelle 4.1 gibt das Konvergenzverhalten des Verfahrens mit Banderweiterung wieder. Dargestellt sind die benötigten Iterationen und das geometrische Mittel der Fehlerreduktionsraten für verschiedene Wavelets, Problemgrößen und Bandweiten, jeweils bei einer ein Level tiefen Zerlegung. Das Band wurde symmetrisch rechts und links der Diagonalen um die angegebene Zahl von Elementen verbreitert. Mit „-“ sind die Fälle gekennzeichnet, für die keine Ergebnisse vorliegen.

Nach einer Phase, in der sich sehr langsame Konvergenz und Divergenz abwechseln, verbessert sich das Verfahren ab einer gewissen Bandbreite. Dabei hängt es von den verwendeten Wavelets ab, ob die Divergenz bei Erweiterung um eine gerade oder ungerade Zahl von Elementen (auf beiden Seiten) auftritt. Die Ursache für die Divergenz liegt in der Instabilität der ILU-Zerlegung für diese Fälle.

Für kleinere Probleme kommt es schon bei geringerer Bandbreite zur schnellen Konvergenz. Die Bandbreite, bei der dieser „Phasenwechsel“ stattfindet, steigt mit der Verdoppelung der Problemgröße um je 1, wie durch die fettgedruckten Zahlen verdeutlicht wird. Ab hier sollte man die Tabelle entsprechend schräg lesen, um die Zahlen für die verschiedenen Problemgrößen besser miteinander vergleichen zu können. Ist die Problemgröße 2^l, so liegt der Schwellenwert bei einer Erweiterung um $l-2$ Elemente auf beiden Seiten der Diagonalen. Hat das Band mindestens diese Breite, verhält sich das Verfahren fast unabhängig von l. Besonders deutlich wird dies bei den Daubechies4-Wavelets. Hier sind die Iterationszahlen und Reduktionsraten bei Vervierfachung der Problemgröße zum Teil sogar identisch.

Es ist zu beachten, daß die tatsächliche Bandbreite, ausgehend von einer Breite b_{vor} vor Banderweiterung, nach selbiger bei $b_{nach} = b_{vor} + 2b_{zusatz}$ liegt.

Da man für einen guten Löser um $l-2$ Elemente auf beiden Seiten erweitern muß, folgt, – wie bereits in Abschnitt 3.1.4 geschildert wurde – daß das Verfahren Kosten in Höhe von

Daubechies2-Wavelets										
Band-erwei-terung	Zahl der Unbekannten n									
	256		512		1024		2048		4096	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
0	50	div	50	div	50	div	50	div	50	div
1	50	0.9927	50	0.9970	50	0.9986	50	0.9993	50	0.9997
2	50	div	50	div	50	div	50	div	50	div
3	50	0.8438	50	0.9544	50	0.9867	50	0.9952	50	0.9979
4	50	div	50	div	50	div	50	div	50	div
5	14	0.1213	30	0.3835	50	0.7186	50	0.9097	50	0.9743
6	**7**	**0.0171**	13	0.1076	50	0.8188	50	div	50	div
7	5	0.0016	**7**	**0.0106**	10	0.0547	20	0.2190	49	0.5456
8	3	$7 \cdot 10^{-5}$	4	0.0008	**6**	**0.0065**	10	0.0417	22	0.2533
9	3	$7 \cdot 10^{-6}$	3	$4.5 \cdot 10^{-5}$	4	0.0005	**6**	**0.0041**	8	0.0229
10	-	-	-	-	-	-	-	-	**5**	**0.0022**
Daubechies4-Wavelets										
0	50	0.9946	50	0.9977	50	0.9989	50	0.9995	50	0.9997
1	50	div	50	div	50	div	50	div	50	div
2	50	0.9444	50	0.9840	50	0.9945	50	0.9976	50	0.9989
3	50	div	50	div	50	div	50	div	50	div
4	43	0.5184	50	0.8129	50	0.9443	50	0.9839	50	0.9944
5	29	0.3729	50	div	50	div	50	div	50	div
6	**10**	**0.0470**	18	0.1994	45	0.5206	50	0.8132	50	0.9444
7	7	0.0114	**11**	**0.0608**	30	0.3796	50	div	50	div
8	6	0.0044	6	0.0076	**10**	**0.0476**	19	0.2010	46	0.5213
9	6	0.0047	6	0.0048	7	0.0114	**11**	**0.0611**	31	0.3811
10	-	-	-	-	6	0.0052	6	0.0076	**10**	**0.0477**
11	-	-	-	-	6	0.0044	6	0.0045	7	0.0114
Daubechies6-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	50	0.9615	50	0.9886	50	0.9957	50	0.9981	50	0.9991
2	50	div	50	div	50	div	50	div	50	div
3	50	0.7106	50	0.9037	50	0.9725	50	0.9914	50	0.9966
4	50	div	50	div	50	div	50	div	50	div
5	31	0.3920	40	0.4907	50	0.7739	50	0.9301	50	0.9800
6	**27**	**0.3494**	35	0.4362	50	div	50	div	50	div
7	27	0.3340	**27**	**0.3473**	29	0.3671	50	0.5559	50	0.8299
8	25	0.3110	25	0.3131	**27**	**0.3281**	50	0.5344	50	div
9	25	0.3012	25	0.3103	26	0.3292	**27**	**0.3260**	29	0.3544
10	23	0.2871	-	-	25	0.2995	25	0.2969	**27**	**0.3126**
Daubechies8-Wavelets										
0	50	0.9697	50	0.9907	50	0.9964	50	0.9983	50	0.9992
1	50	div	50	div	50	div	50	div	50	div
2	50	0.7910	50	0.9349	50	0.9813	50	0.9937	50	0.9973
3	50	div	50	div	50	div	50	div	50	div
4	39	0.4822	50	0.6409	50	0.8692	50	0.9621	50	0.9887
5	36	0.4518	50	div	50	div	50	div	50	div
6	**32**	**0.4143**	34	0.4300	40	0.4815	50	0.7522	50	0.9220
7	31	0.3806	**31**	**0.3843**	37	0.4543	50	div	50	div
8	29	0.3710	31	0.3844	**31**	**0.3936**	33	0.4084	50	0.5820
9	29	0.3559	29	0.3654	29	0.3670	**31**	**0.3702**	50	0.8299
10	27	0.3433	-	-	-	-	30	0.3770	**31**	**0.3743**

Tabelle 4.1.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Laplaceoperator, verschiedene Wavelets, $lt = 1$, Verfahren 1

$\mathcal{O}(n(\log_2(n))^2)$ verursacht.

Es stellt sich die Frage, ob sich das bei der ein Level tiefen Zerlegung beobachtete Verhalten auf den Fall der tiefstmöglichen Zerlegung überträgt. Der Begriff „tiefstmöglich" wurde als Abgrenzung zu „vollständig" gewählt. Da ab einem gewissen Level das Band so breit wird, daß durch Banderweiterung und zerlegungsbedingte Verkleinerung der Matrizen die volle Matrix betrachtet werden muß, macht es keinen Sinn, von diesem Level an noch einmal zu zerlegen. Dieses Level soll also bei „tiefstmöglicher" Zerlegung erreicht werden. Das Interesse an einer möglichst tiefen Zerlegung liegt unter anderem darin begründet, daß die Güte der ILU-Zerlegung nicht von der Problemgröße respektive Matrixgröße unabhängig ist [31]. Da jedoch bei Betrachtung der vollen Teilmatrix auf dem jeweiligen Level exakt gelöst wird, würde jede weitere Zerlegung den Rechenaufwand erhöhen, ohne das Ergebnis zu verbessern.

Tabelle 4.2 gibt die entsprechenden Ergebnisse wieder. Es fällt auf, daß die Resultate für Haarwavelets nahezu identisch mit denen bei Leveltiefe 1 sind. Für diese lohnt es sich also nicht, tiefer als ein Level zu zerlegen. Andererseits steigen die Iterationszahlen (und damit auch die durchschnittlichen Fehlerreduktionsraten) bei tiefstmöglicher Zerlegung im Gegensatz zur ein Level tiefen Zerlegung mit steigender Länge der Waveletfilter kaum an, sodaß es von diesen Kriterien her sehr wohl sinnvoll ist, bei Verwendung von Daubechies8-Wavelets so tief wie möglich zu zerlegen. Nicht berücksichtigt wird bei dieser Betrachtung der Rechenaufwand, der in die zum Lösen benötigte Zeit eingeht. So kann es durchaus sein, daß man im letztgenannten Fall trotz höherer Iterationszahl bei Leveltiefe eins schneller die gewünschte Lösungsgenauigkeit erreicht als bei tiefstmöglicher Zerlegung. Auch hier gilt, daß man erst ab einem Aufwand von $\mathcal{O}(n(\log_2(n))^2)$ einen zuverlässigen Löser erhält.

Einige Ergebnisse des Verfahrens ohne Schurkomplementkorrektur sind in den Tabellen 4.3 und 4.4 dargestellt.

Bei der Verwendung von Wavelets zeigt sich die Abhängigkeit des Ergebnisses von der Leveltiefe deutlich. Durch das nicht korrigierte Schurkomplement ist das Ergebnis auf jedem folgenden Level bereits a priori falsch. Dieser Effekt wird dadurch verstärkt, daß statt randangepaßten Wavelets periodisierte Wavelets benutzt wurden, sodaß die Approximation an den Feingitteroperator bereits durch die zusätzlichen Einträge in der oberen rechten und unteren linken Ecke der Matrix, die durch die Periodisierung entstehen, fehlerhaft ist. Bei den mit randangepaßten Prewavelets erzeugten Ergebnissen wird das Verhalten des Verfahrens bei tieferer Zerlegung nur leicht schlechter und bleibt schließlich konstant gut.

Bei Leveltiefe 1 erreicht man sowohl für Waveletes als auch für Prewavelets ein recht gutes Verhalten. Für Daubechies-4-Wavelets bekommt man eine bessere Reduktionsrate als für die anderen getesten Wavelets, was sich auch in der Iterationszahl zeigt. Sogar wesentlich besser sind die Raten und Iterationszahlen für Prewavelets, da sie für den Laplaceopertaor problemangepaßt sind.

Daubechies2-Wavelets										
Band-	Zahl der Unbekannten n									
erwei-	256		512		1024		2048		4096	
terung	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
0	50	div	50	div	50	div	50	div	50	div
1	50	0.9957	50	0.9981	50	0.9991	50	0.9996	50	0.9998
2	50	div	50	div	50	div	50	div	50	div
3	50	0.8036	50	0.9413	50	0.9832	50	0.9942	50	0.9975
4	50	div	50	div	50	div	50	div	50	div
5	15	0.1362	33	0.4136	50	0.7423	50	0.9190	50	0.9769
6	**7**	**0.0163**	13	0.1023	50	0.7578	50	div	50	div
7	5	0.0015	**7**	**0.0098**	10	0.0508	19	0.2055	47	0.5270
8	3	$7 \cdot 10^{-5}$	4	0.0008	**6**	**0.0066**	10	0.0428	22	0.2614
9	3	$7 \cdot 10^{-6}$	3	$5 \cdot 10^{-5}$	4	0.0005	**6**	**0.0042**	8	0.0237
10	-	-	3	$5 \cdot 10^{-6}$	3	$3 \cdot 10^{-5}$	4	0.0003	**5**	**0.0022**
Daubechies4-Wavelets										
0	50	0.9963	50	0.9983	50	0.9992	50	0.9996	50	0.9998
1	50	div	50	div	50	div	50	div	50	div
2	50	0.9359	50	0.9817	50	0.9939	50	0.9974	50	0.9988
3	50	div	50	div	50	div	50	div	50	div
4	46	0.5388	50	0.8250	50	0.9483	50	0.9850	50	0.9947
5	28	0.3569	50	div	50	div	50	div	50	div
6	**10**	**0.0471**	18	0.1937	43	0.5101	50	0.8069	50	0.9422
7	6	0.0090	**11**	**0.0601**	31	0.3873	50	div	50	div
8	5	0.0019	7	0.0097	**10**	**0.0506**	19	0.2050	46	0.5262
9	4	0.0009	5	0.0013	7	0.0099	**11**	**0.0590**	31	0.3763
10	5	0.0012	5	0.0013	5	0.0018	7	0.0095	**10**	**0.0493**
11	5	0.0012	5	0.0013	5	0.0014	5	0.0015	7	0.0100
Daubechies6-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	50	0.9615	50	0.9874	50	0.9954	50	0.9980	50	0.9990
2	50	div	50	div	50	div	50	div	50	div
3	50	0.7122	50	0.9078	50	0.9738	50	0.9918	50	0.9967
4	50	div	50	div	50	div	50	div	50	div
5	15	0.1516	36	0.4443	50	0.7645	50	0.9273	50	0.9793
6	**10**	**0.0520**	27	0.3390	50	div	50	div	50	div
7	7	0.0105	**10**	**0.0554**	20	0.2219	49	0.5499	50	0.8298
8	5	0.0021	7	0.0136	**12**	**0.0836**	50	0.5819	50	div
9	4	0.0008	5	0.0025	7	0.0159	**12**	**0.0819**	24	0.2918
10	4	0.0008	4	0.0008	6	0.0037	8	0.0225	**15**	**0.1362**
11	4	0.0008	4	0.0007	4	0.0007	6	0.0046	9	0.0267
Daubechies8-Wavelets										
0	50	0.9667	50	0.9899	50	0.9961	50	0.9982	50	0.9992
1	50	div	50	div	50	div	50	div	50	div
2	50	0.7914	50	0.9373	50	0.9821	50	0.9939	50	0.9974
3	50	div	50	div	50	div	50	div	50	div
4	22	0.2714	50	0.6133	50	0.8632	50	0.9605	50	0.9883
5	15	0.1441	50	div	50	div	50	div	50	div
6	**9**	**0.0316**	15	0.1417	34	0.4237	50	0.7496	50	0.9217
7	6	0.0078	**10**	**0.0512**	26	0.3263	50	div	50	div
8	5	0.0018	7	0.0118	**11**	**0.0620**	21	0.2379	50	0.5707
9	4	0.0006	5	0.0027	8	0.0190	**14**	**0.1129**	50	0.8641
10	4	0.0006	4	0.0007	6	0.0044	8	0.0246	**14**	**0.1163**
11	4	0.0006	4	0.0006	4	0.0006	6	0.0063	10	0.0410

Tabelle 4.2.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Laplaceoperator, verschiedene Wavelets, tiefstmögliche Zerlegung, Verfahren 1

Daubechies2-Wavelets										
Level-tiefe	Zahl der Unbekannten n									
	256		512		1024		2048		4096	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
1	37	0.4637	38	0.4647	38	0.4648	39	0.4656	39	0.4656
2	50	0.7160	50	0.7164	50	0.7166	50	0.7167	50	0.7168
3	50	0.8419	50	0.8426	50	0.8429	50	0.8430	50	0.8431
4	50	0.9074	50	0.9084	50	0.9089	50	0.9091	50	0.9092
5	50	0.9443	50	0.9436	50	0.9443	50	0.9446	50	0.9447
Daubechies4-Wavelets										
1	22	0.2675	22	0.2679	22	0.2681	23	0.2694	23	0.2694
2	36	0.4504	36	0.4506	37	0.4513	37	0.4513	38	0.4519
3	50	0.6290	50	0.6276	50	0.6268	50	0.6363	50	0.6260
4	50	0.7816	50	0.7777	50	0.7745	50	0.7721	50	0.7706
5	50	0.8844	50	0.8800	50	0.8763	50	0.8737	50	0.8719
Daubechies6-Wavelets										
1	50	0.5641	50	0.5682	50	0.5702	50	0.5712	50	0.5717
2	50	0.5661	50	0.5778	50	0.5834	50	0.5861	50	0.5875
3	50	0.6904	50	0.6952	50	0.6980	50	0.6994	50	0.7001
4	50	0.7724	50	0.7748	50	0.7763	50	0.7771	50	0.7775
5	50	0.7643	50	0.8580	50	0.8589	50	0.8594	50	0.8597
Daubechies8-Wavelets										
1	50	0.5743	50	0.5803	50	0.5833	50	0.5847	50	0.5855
2	50	0.6122	50	0.6266	50	0.6334	50	0.6367	50	0.6384
3	49	0.5629	50	0.6020	50	0.6195	50	0.6278	50	0.6318
4	50	0.7982	50	0.8008	50	0.8024	50	0.8033	50	0.8038
5	50	0.7940	50	0.9097	50	0.9105	50	0.9109	50	0.9111

Tabelle 4.3.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Laplaceoperator, verschiedene Wavelets, verschiedene Leveltiefen, Verfahren 2

Level-tiefe	Zahl der Unbekannten n									
	255		511		1023		2047		4095	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
1	13	0.1025	13	0.0997	13	0.0974	13	0.0947	13	0.0922
2	15	0.1531	15	0.1496	15	0.1462	15	0.1429	15	0.1400
3	16	0.1695	16	0.1658	16	0.1623	16	0.1590	16	0.1555
4	17	0.1803	17	0.1772	17	0.1749	17	0.1721	17	0.1707
5	16	0.1726	17	0.1821	17	0.1788	17	0.1770	17	0.1715
6	-	-	17	0.1778	17	0.1830	17	0.1797	18	0.1846
7	-	-	-	-	17	0.1793	18	0.1916	18	0.1918
8	-	-	-	-	-	-	18	0.1862	18	0.1880
9	-	-	-	-	-	-	-	-	18	0.1928

Tabelle 4.4.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Laplaceoperator, Prewavelets, verschiedene Leveltiefen, Verfahren 2

Eine Abhängigkeit von der Problemgröße ist bei Wavelets höchstens in langsam steigenden Iterationszahlen zu erkennen. Für Prewavelets hat man konstante Iterationszahlen bei leicht fallenden Fehlerreduktionsraten. Es ist anzunehmen, daß letztere ab einer bestimmten Problemgröße konstant bleiben. Die Reduktionsrate des Verfahrens ist also offenbar in beiden Fällen von der Matrixgröße unabhängig.

Vergleicht man die Varianten miteinander, so ergibt sich, daß die Approximation des Schurkomplementes mit Banderweiterung dem Verfahren ohne Schurkomplementkorrektur mit Wavelets klar überlegen ist. Die benötigte Zahl von Elementen, um die beidseitig der Diagonale zu erweitern ist, läßt sich bei Kenntnis der Problemgröße a priori optimal festlegen. Verfahren 2 ist zwar problemgrößenunabhängig, aber höchstens bei Leveltiefe eins sinnvoll einzusetzen. Die sich ergebenden Reduktionsraten lassen sich ohne weiteres mit der Banderweiterungsvariante unterbieten. Andererseits kostet dies mehr Rechenaufwand, was sich auch in den Lösungszeiten niederschlägt. Hier zeigt sich ebenfalls, daß Verfahren 2 mit Prewavelets ab einer gewissen Problemgröße überlegen ist, da es aufgrund der linearen Komplexität trotz höherer Fehlerreduktionsraten Verfahren 1 schlägt. Tabelle 4.5 enthält einen entsprechenden Vergleich. Für das Banderweiterungsverfahren werden jeweils die Zeiten für die in den Tabellen 4.1 und 4.2 fett gedruckten Fälle angegeben. Die Zeiten von Verfahren 2 entstanden bei Leveltiefe eins.

Deutlich erkennbar ist das $\mathcal{O}(n)$-Verhalten des Verfahrens ohne Schurkomplementkorrektur. Die Zahlen verdoppeln sich nahezu exakt von Zeile zu Zeile. Außerdem steigen die Werte mit der Länge der verwendeten Waveletfilter, was trivialerweise an den durch eben diese Länge verursachten breiteren Bändern nach der Zerlegung liegt. Ebenso zu sehen ist, daß die Banderweiterung einen höheren Rechenaufwand verursacht, auch wenn der zusätzliche Kostenfaktor von $(\log_2(n))^2$ besonders für Leveltiefe 1 aus verschiedenen Gründen nur schwer zu erkennen ist. Dies liegt zum einen daran, daß die Zeitmessung an sich leichte Ungenauigkeiten beinhaltet, die sich gerade bei geringen Zeiten stärker bemerkbar machen. Hinzu kommt, daß, wie in Abschnitt 3.1.4 festgestellt wurde, der Faktor $(\log_2(n))^2$ auf dem zweiten und dem dritten Level wesentlich höhere Koeffizienten aufweist als auf dem ersten. Da die Banderweiterung auch mit längeren Filtern einen höheren Aufwand verursacht, fällt es mit steigender Länge der Filter leichter, ein $\mathcal{O}\left(n(\log_2(n))^2\right)$-Verhalten aus der Tabelle abzulesen.

Beurteilt man die Verfahrensvarianten nach der benötigten Zeit, so schneidet Verfahren 2 mit Prewavelets am besten ab. Dicht dahinter folgt die Banderweiterung bei Leveltiefe eins zusammen mit Haarwavelets vor der Variante ohne Schurkomplementkorrektur ebenfalls mit Daubechies2-Wavelets gleichauf mit derselben Variante mit Daubechies4-Wavelets. Diese Platzierung ergibt sich aus den Iterationszahlen. Die Banderweiterung zieht zwar einen höheren Rechenaufwand pro Iteration nach sich, dieser wird aber durch die geringe Anzahl von notwendigen Iterationen ausgeglichen. Der Mehraufwand durch die längeren Daubechies4-Waveletfilter gegenüber den Haarwavelets macht sich in Form einer geringeren Iterationszahl

n	Verfahren 1 $lt = 1$				Verfahren 1 tiefstmögl.				Verfahren 2				
	D2	D4	D6	D8	D2	D4	D6	D8	D2	D4	D6	D8	Pre
256	**0.03**	0.05	0.09	0.11	0.09	0.11	0.14	0.16	0.05	0.05	0.1	0.15	0.06
512	**0.09**	0.12	0.21	0.26	0.18	0.24	0.28	0.36	0.12	0.11	0.21	0.27	**0.09**
1024	0.16	0.22	0.42	0.52	0.36	0.52	0.67	0.77	0.2	0.19	0.41	0.5	**0.14**
2048	0.31	0.46	0.85	1.06	0.8	1.15	1.43	1.82	0.39	0.35	0.83	1.02	**0.25**
4096	0.64	0.94	1.83	2.45	1.64	2.45	3.56	3.93	0.76	0.72	1.6	2.04	**0.49**

Tabelle 4.5.: Zeitvergleich für 1D-Laplaceoperator, jeweils bis zur Lösung mit Genauigkeit ε. Die Spalten „D2“ bis „D8“ beinhalten die Zeiten für die verwendeten Daubechies-Wavelets in Sekunden.

bezahlt, sodaß für beide Wavelets in etwa die gleichen Zeiten benötigt werden. Aufgrund des durch die größere Zerlegungstiefe höheren Rechenaufwandes ist die Banderweiterung mit tiefstmöglicher Zerlegung die langsamste Variante.

Insgesamt erreicht man zwar mit der Banderweiterungsvariante stets bei genügend breitem Band die besten Fehlerreduktionsraten, durch die konstanten Raten von Verfahren 2 mit den problemangepaßten Prewavelets ist dieses Verfahren aufgrund des geringeren Aufwandes von $\mathcal{O}(n)$ jedoch ab einer gewissen Problemgröße (hier $n = 1024/1023$) der schnellere Löser.

4.1.2. Konvektionsdiffusionsoperator

Für den Konvektionsdiffusionsoperator wurden bis auf die Fehlergrenze, die auf

$$\varepsilon = 10^{-10}$$

gesetzt wurde, die gleichen Einstellungen verwendet wie für den Laplaceoperator. Es wurde sowohl für Konvektionsstärke 10 als auch für Konvektionsstärke 1000 getestet.

Folgende Resultate konnten beobachtet werden: Für Konvektionsstärke 10 schneiden alle drei Verfahrensvarianten im allgemeinen besser ab als für den Laplaceoperator (Tabellen 4.6, 4.7, 4.8 und 4.9). Gute Konvergenz bei Verfahren 1 erhält man jetzt schon bei einer Banderweiterung um $l - 1$ Elemente auf beiden Seiten der Diagonalen. Wie beim Laplaceoperator bekommt man mit Ausnahme vom Haarwaveletfall bei tiefstmöglicher Zerlegung bessere Fehlerreduktionsraten als bei Leveltiefe eins, sodaß auch hier Verfahren 1 mit Haarwavelets und Leveltiefe eins dicht hinter dem Verfahren ohne Schurkomplementkorrektur mit Prewavelets den zweitbesten Löser bildet. Etwas unglaubwürdig erscheint, daß die Raten bei Problemgröße 4096 mit zunehmender Bandbreite zum Teil ansteigen. Dies läßt sich auf die ILU-Zerlegung zurückführen.

Für das Verfahren ohne Schurkomplementkorrektur (Tabellen 4.8 und 4.9) gilt das gleiche wie für den Laplaceoperator. Aufgrund der geringen Konvektionsstärke machen sich hier sowohl für Wavelets als auch für Prewavelets kaum Unterschiede bemerkbar.

Band-erwei-terung	256		512		1024		2048		4096	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
Daubechies2-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	50	0.9910	50	0.9968	50	0.9986	50	0.9993	50	0.9997
2	50	div	50	div	50	div	50	div	50	div
3	50	0.6263	50	0.8824	50	0.9781	50	0.9947	50	0.9978
4	28	0.3866	50	div	50	div	50	div	50	div
5	**8**	**0.0353**	14	0.1508	32	0.4320	50	0.7625	50	0.9418
6	-	-	**8**	**0.0228**	14	0.1421	50	div	50	div
7	-	-	-	-	**7**	**0.0151**	11	0.0740	37	0.4793
8	-	-	-	-	-	-	**6**	**0.0079**	10	0.0655
9	-	-	-	-	-	-	-	-	**9**	**0.0450**
Daubechies4-Wavelets										
0	50	0.9939	50	0.9976	50	0.9989	50	0.9995	50	0.9997
1	50	div	50	div	50	div	50	div	50	div
2	50	0.8549	50	0.9710	50	0.9937	50	0.9975	50	0.9989
3	50	div	50	div	50	div	50	div	50	div
4	19	0.2403	47	0.5693	50	0.8530	50	0.9707	50	0.9937
5	**11**	**0.0805**	35	0.4728	50	div	50	div	50	div
6	6	0.0087	**10**	**0.0619**	19	0.2380	48	0.5716	50	0.8526
7	-	-	7	0.0161	**11**	**0.0788**	36	0.4699	50	div
8	-	-	-	-	6	0.0115	**10**	**0.0606**	19	0.2289
9	-	-	-	-	-	-	7	0.0155	**20**	**0.2543**
10	-	-	-	-	-	-	-	-	6	0.0115
Daubechies6-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	50	0.9056	50	0.9830	50	0.9954	50	0.9981	50	0.9991
2	50	div	50	div	50	div	50	div	50	div
3	35	0.4735	50	0.7542	50	0.9353	50	0.9889	50	0.9964
4	29	0.4032	50	div	50	div	50	div	50	div
5	**24**	**0.3341**	27	0.3788	41	0.5218	50	0.8170	50	0.9600
6	-	-	**25**	**0.3385**	35	0.4642	50	div	50	div
7	-	-	23	0.3146	**25**	**0.3423**	28	0.3737	50	0.6023
8	-	-	-	-	23	0.3014	**25**	**0.3383**	50	0.7496
9	-	-	-	-	-	-	25	0.3169	**26**	**0.3510**
10	-	-	-	-	-	-	-	-	31	0.4128
Daubechies8-Wavelets										
0	50	0.9289	50	0.9875	50	0.9961	50	0.9983	50	0.9992
1	50	div	50	div	50	div	50	div	50	div
2	47	0.5717	50	0.8319	50	0.9640	50	0.9927	50	0.9973
3	48	0.5837	50	div	50	div	50	div	50	div
4	29	0.3999	35	0.4655	50	0.6807	50	0.9065	50	0.9833
5	**25**	**0.3467**	31	0.4295	50	div	50	div	50	div
6	-	-	**29**	**0.3886**	31	0.4224	40	0.4957	50	0.7966
7	-	-	-	-	**29**	**0.3792**	36	0.4735	50	div
8	-	-	-	-	27	0.3739	**30**	**0.4006**	32	0.4205
9	-	-	-	-	-	-	27	0.3552	**44**	**0.5360**
10	-	-	-	-	-	-	-	-	39	0.4878

Tabelle 4.6.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 10, verschiedene Wavelets, $lt = 1$, Verfahren 1

Die Situation verändert sich deutlich bei Konvektionsstärke 1000 (Tabellen 4.10, 4.11, 4.12 und 4.13). Für Leveltiefe eins ist Verfahren 1 jetzt sogar fast ein $\mathcal{O}(n)$-Löser. Die Iterationszahlen steigen bei gleicher Banderweiterung mit steigender Problemgröße nur sehr langsam an. Besonders gut verhalten sich Daubechies2-Wavelets, die als einzige auch bei tiefstmöglicher Zerlegung ähnlich gut abschneiden, wohingegen die anderen Wavelets hier nur bei sehr breitem Band zuverlässig konvergieren. Der Grund für dieses Verhalten ist der, daß die Haarwavelets mit steigender Konvektion besser problemangepaßt sind.

Für Leveltiefe 2 ließen sich oft bessere Resultate erzielen als für Leveltiefe 1. Ab Leveltiefe 3 werden die Reduktionsraten dann meist wieder schlechter. Dieses Verhalten läßt sich dadurch begründen, daß der sich aus der verwendeten Upwinddiskretisierung ergebende Stern mit jeder Zweiskalentransformation an den entsprechenden zentrale-Differenzen-Stern angenähert wird [27]. Da dies ein Prozeß ist, der sich erst nach und nach auswirkt, erhält man für Leveltiefe 2 bessere Reduktionsraten.

Das Verfahren ohne Schurkomplementkorrektur liefert auf dem ersten Level ebenfalls bessere Fehlerreduktionsraten als bisher, dafür kommt es nun (außer bei den in diesem Fall problemangepaßteren Haarwavelets) aufgrund der ILU-Zerlegung und dem Übergehen des Upwindsternes in einen zentrale-Differenzen-Stern mit steigender Zerlegungstiefe zur Divergenz. Ansonsten gilt wie bereits für den Laplaceoperator, daß man bei Leveltiefe 1 mit Daubechies4-Wavelets die besten Reduktionsraten erzielt.

Die Prewaveletvariante verschlechtert sich gegenüber geringerer Konvergenz bei Leveltiefe eins im wesentlichen nur für Problemgröße 255. Bei tieferer Zerlegung wirkt sich die Konvektion jedoch genauso aus wie bei den Wavelets.

Durch das jetzt bessere Abschneiden der Haarwavelets bei beiden Verfahren sind die drei Varianten Haarwavelets mit Verfahren 1 und Leveltiefe eins, Haarwavelets mit Verfahren 2 und Prewavelets mit Verfahren 2 (jeweils bei Leveltiefe eins) für Konvektionsstärke 1000 in etwa gleich schnell. Es ist zu erwarten, daß das Verhalten der Haarwavelets bei weiter steigender Konvektion besser wird, da sie dem Problem immer besser angepaßt erscheinen.

4.2. Zweidimensionale Operatoren

4.2.1. Laplaceoperator

Für die Diskretisierung des zweidimensionalen Laplaceoperators wurde der gewöhnliche Finite-Differenzen-5-Punktestern (3.4) verwendet [31]. Die Fehlertoleranzgrenze lag bei $\varepsilon = 10^{-11}$, die übrigen Parameter blieben unverändert.

Daubechies2-Wavelets										
Band-erwei-terung	Zahl der Unbekannten n									
	256		512		1024		2048		4096	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
0	50	div	50	div	50	div	50	div	50	div
1	50	0.9954	50	0.9980	50	0.9991	50	0.9996	50	0.9998
2	50	div	50	div	50	div	50	div	50	div
3	45	0.5570	50	0.8453	50	0.9686	50	0.9934	50	0.9974
4	32	0.4368	50	div	50	div	50	div	50	div
5	**9**	**0.0412**	15	0.1684	35	0.4627	50	0.7854	50	0.9499
6	-	-	**7**	**0.0229**	14	0.1268	50	0.9881	50	div
7	-	-	-	-	**7**	**0.0171**	10	0.0666	27	0.3640
8	-	-	-	-	-	-	**6**	**0.0101**	17	0.2019
9	-	-	-	-	-	-	-	-	**8**	**0.0324**
Daubechies4-Wavelets										
0	50	0.9961	50	0.9983	50	0.9992	50	0.9996	50	0.9998
1	50	div	50	div	50	div	50	div	50	div
2	50	0.8333	50	0.9644	50	0.9928	50	0.9973	50	0.9988
3	50	div	50	div	50	div	50	div	50	div
4	20	0.2585	50	0.5905	50	0.8644	50	0.9737	50	0.9941
5	**10**	**0.0751**	34	0.4605	50	div	50	div	50	div
6	6	0.0119	**10**	**0.0620**	19	0.2303	47	0.5649	50	0.8467
7	-	-	7	0.0128	**11**	**0.0783**	37	0.4797	50	div
8	-	-	-	-	7	0.0125	**10**	**0.0629**	31	0.4102
9	-	-	-	-	-	-	7	0.0163	**13**	**0.1175**
10	-	-	-	-	-	-	-	-	14	0.1390
Daubechies6-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	50	0.8915	50	0.9800	50	0.9949	50	0.9979	50	0.9990
2	50	div	50	div	50	div	50	div	50	div
3	32	0.4401	50	0.7603	50	0.9404	50	0.9894	50	0.9965
4	20	0.2588	50	div	50	div	50	div	50	div
5	**9**	**0.0473**	16	0.1878	38	0.4936	50	0.8066	50	0.9570
6	-	-	**10**	**0.0717**	32	0.4282	50	div	50	div
7	-	-	7	0.0151	**11**	**0.0773**	20	0.2600	50	0.5966
8	-	-	-	-	7	0.0205	**12**	**0.1068**	50	0.7378
9	-	-	-	-	-	-	8	0.0271	**12**	**0.0991**
10	-	-	-	-	-	-	-	-	9	0.0487
Daubechies8-Wavelets										
0	50	0.9203	50	0.9858	50	0.9959	50	0.9982	50	0.9992
1	50	div	50	div	50	div	50	div	50	div
2	43	0.5465	50	0.8350	50	0.9654	50	0.9930	50	0.9973
3	50	0.6118	50	div	50	div	50	div	50	div
4	12	0.0993	24	0.3209	50	0.6606	50	0.8996	50	0.9820
5	**8**	**0.0355**	16	0.1938	50	div	50	div	50	div
6	-	-	**9**	**0.0437**	16	0.1732	36	0.4678	50	0.7921
7	-	-	-	-	**10**	**0.0686**	31	0.4127	50	div
8	-	-	-	-	7	0.0138	**11**	**0.0689**	23	0.3066
9	-	-	-	-	-	-	8	0.0290	**14**	**0.1405**
10	-	-	-	-	-	-	-	-	9	0.0417

Tabelle 4.7.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 10, verschiedene Wavelets, tiefstmögliche Zerlegung, Verfahren 1

Level-tiefe	256 k	256 $\varrho_{k,0}$	512 k	512 $\varrho_{k,0}$	1024 k	1024 $\varrho_{k,0}$	2048 k	2048 $\varrho_{k,0}$	4096 k	4096 $\varrho_{k,0}$
Daubechies2-Wavelets										
1	30	0.4181	32	0.4409	34	0.4503	35	0.4638	42	0.5213
2	50	0.6784	50	0.6913	50	0.7053	50	0.7113	50	0.7141
3	50	0.8022	50	0.8220	50	0.8335	50	0.8385	50	0.8409
4	50	0.8793	50	0.8946	50	0.9023	50	0.9059	50	0.9076
5	50	0.9111	50	0.9298	50	0.9382	50	0.9417	50	0.9433
Daubechies4-Wavelets										
1	19	0.2474	19	0.2493	20	0.2542	20	0.2551	23	0.2995
2	31	0.4282	32	0.4327	32	0.4347	33	0.4350	33	0.4381
3	50	0.6157	50	0.6210	50	0.6223	50	0.6226	50	0.6222
4	50	0.7648	50	0.7747	50	0.7771	50	0.7746	50	0.7777
5	50	0.8633	50	0.8730	50	0.8764	50	0.8774	50	0.8776
Daubechies6-Wavelets										
1	42	0.5369	45	0.5572	47	0.5666	49	0.5742	50	0.5818
2	36	0.4838	43	0.5384	47	0.5687	50	0.5836	50	0.5847
3	50	0.6765	50	0.6860	50	0.6936	50	0.6989	50	0.7022
4	50	0.7666	50	0.7706	50	0.7746	50	0.7775	50	0.7795
5	50	0.7573	50	0.8526	50	0.8561	50	0.8586	50	0.8602
Daubechies8-Wavelets										
1	41	0.5277	45	0.5564	49	0.5712	50	0.5870	50	0.5881
2	41	0.5269	49	0.5822	50	0.6137	50	0.6297	50	0.6380
3	36	0.4877	40	0.5164	47	0.5673	50	0.6034	50	0.6213
4	50	0.7948	50	0.7991	50	0.8018	50	0.8041	50	0.8058
5	50	0.7960	50	0.9082	50	0.9101	50	0.9112	50	0.9120

Tabelle 4.8.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 10, verschiedene Wavelets, verschiedene Leveltiefen, Verfahren 2

Level-tiefe	255 k	255 $\varrho_{k,0}$	511 k	511 $\varrho_{k,0}$	1023 k	1023 $\varrho_{k,0}$	2047 k	2047 $\varrho_{k,0}$	4095 k	4095 $\varrho_{k,0}$
	Zahl der Unbekannten n									
1	11	0.0947	11	0.0921	11	0.0893	12	0.1014	12	0.0971
2	14	0.1467	14	0.1453	14	0.1422	14	0.1410	21	0.2720
3	14	0.1563	15	0.1592	15	0.1595	15	0.1606	15	0.1567
4	15	0.1644	15	0.1647	15	0.1614	15	0.1614	15	0.1618
5	15	0.1666	15	0.1670	15	0.1668	17	0.2024	17	0.2011
6	-	-	15	0.1686	15	0.1685	16	0.1841	16	0.1786
7	-	-	-	-	15	0.1697	16	0.1748	25	0.3284
8	-	-	-	-	-	-	16	0.1772	21	0.2721
9	-	-	-	-	-	-	-	-	30	0.4032

Tabelle 4.9.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 10, Prewavelets, verschiedene Leveltiefen, Verfahren 2

Verfahren mit Schurkomplementapproximation und Banderweiterung mit Wavelets

Wie im eindimensionalen Fall sind die Ergebnisse für Leveltiefe eins (Tabelle 4.14) und bei tiefstmöglicher Zerlegung (Tabelle 4.15) wiedergegeben. Für die Banderweiterung gilt wie vormals, daß sie stets symmetrisch durchgeführt wurde, das heißt sowohl die Blockbänder als auch die Bänder auf den Blöcken wurden um die gleiche Anzahl Elemente auf beiden Seiten der Diagonalen erweitert.

Leider überträgt sich das Konvergenzverhalten bei Leveltiefe 1 nicht aus dem Eindimensionalen auf das Zweidimensionale. Bei Vervierfachung der Problemgröße muß man die Zahl der Elemente, um die das Band erweitert wird, um fünf (bei Daubechies4-Wavelets sogar sechs) anstatt um eins erhöhen. Ist die Problemgröße 4^l, so bekommt man, setzt man beispielsweise bei Haarwavelets bei Problemgröße 1024 und Erweiterung um 4 an, für die Banderweiterung die Wachstumsformel $5l - 21$.

Wie sich das Verfahren bei tiefstmöglicher Zerlegung verhält, geht aus Tabelle 4.15 hervor. Hier gilt offensichtlich ein besseres Wachstumsgesetz, um über die verschiedenen Problemgrößen konstante Reduktionsraten zu erhalten. Es ist dies – ebenso wie auch im Eindimensionalen – $l-2$. Gemäß Abschnitt 3.2.4 entspricht dies einem Aufwand von $\mathcal{O}\left(n(\log_2(\sqrt{n}))^4\right)$.

Die angegebenen Zeiten sind nicht auf den ersten Blick verständlich. Zunächst ist bei 1024 Unbekannten der Sprung in der Lösungszeit zwischen Berechnung auf einem Band über mehr als ein Level und Berechnung auf der vollen 1024×1024-Matrix deutlich zu erkennen. Dieser Sprung taucht auch bei 4096 und bei 16384 Unbekannten auf, da die einzelnen Blöcke der 1024×1024-Matrix die Größe 32×32 besitzen, wohingegen ein Level tiefer die Blockgröße nur noch 16×16 beträgt. Somit wird auf der vollen 1024×1024-Matrix gelöst, sobald das Band die Breite von 16 annähernd erreicht. Für Haarwavelets beispielsweise ist der Sprung

Band-erwei-terung	256		512		1024		2048		4096	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
Daubechies2-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	32	0.4272	50	0.6900	50	0.9814	50	div	50	div
2	18	0.2349	43	0.5374	50	0.7628	50	div	50	div
3	11	0.0928	15	0.1510	19	0.2410	29	0.3743	50	0.6099
4	8	0.0288	9	0.0540	11	0.0849	15	0.1498	24	0.3177
5	6	0.0073	6	0.0128	7	0.0187	8	0.0332	11	0.0750
6	**4**	**0.0015**	**5**	**0.0025**	**5**	**0.0034**	**6**	**0.0056**	**7**	**0.0149**
7	-	-	4	0.0004	4	0.0005	4	0.0007	5	0.0020
Daubechies4-Wavelets										
0	50	0.7711	50	0.8827	50	0.9917	50	0.9976	50	0.9993
1	50	div	50	div	50	div	50	div	50	div
2	20	0.2623	28	0.3856	41	0.5137	50	0.7050	50	0.9723
3	15	0.1607	21	0.2708	32	0.4267	50	0.8310	50	div
4	9	0.0545	11	0.0819	13	0.1119	15	0.1537	25	0.3341
5	7	0.0173	8	0.0301	9	0.0424	10	0.0626	15	0.1501
6	**6**	**0.0072**	**6**	**0.0098**	**6**	**0.0115**	**7**	**0.0167**	**9**	**0.0362**
7	5	0.0034	5	0.0040	5	0.0038	6	0.0065	7	0.0134
8	-	-	-	-	-	-	5	0.0029	5	0.0022
Daubechies6-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	41	0.5327	50	0.6271	50	0.7375	50	0.8730	50	0.9888
2	48	0.5796	50	0.9376	50	div	50	div	50	div
3	17	0.2178	19	0.2509	23	0.3028	31	0.4140	50	0.6104
4	14	0.1352	14	0.1530	16	0.1903	23	0.2981	50	0.6890
5	10	0.0682	10	0.0603	10	0.0667	12	0.0939	16	0.1739
6	**9**	**0.0437**	**9**	**0.0414**	**9**	**0.0460**	**10**	**0.0554**	**12**	**0.0937**
7	8	0.0265	7	0.0210	7	0.0185	8	0.0273	11	0.0674
8	-	-	-	-	-	-	8	0.0246	9	0.0459
Daubechies8-Wavelets										
0	39	0.5097	50	0.6333	50	0.7724	50	0.8800	50	0.9925
1	45	0.5634	50	div	50	div	50	div	50	div
2	17	0.2150	20	0.2692	26	0.3535	39	0.4994	50	0.7123
3	14	0.1570	18	0.2078	21	0.2805	40	0.5075	50	div
4	10	0.0730	11	0.0809	12	0.1000	13	0.1259	20	0.2519
5	10	0.0586	10	0.0659	11	0.0682	12	0.0884	16	0.1633
6	**7**	**0.0230**	**8**	**0.0271**	**8**	**0.0333**	**9**	**0.0392**	**11**	**0.0739**

(Columns: Zahl der Unbekannten n)

Tabelle 4.10.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 1000, verschiedene Wavelets, $lt = 1$, Verfahren 1

Daubechies2-Wavelets										
Band-	Zahl der Unbekannten n									
erwei-	256		512		1024		2048		4096	
terung	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
0	50	div	50	div	50	div	50	div	50	div
1	50	0.6766	50	div	50	div	50	div	50	div
2	16	0.1939	26	0.3618	50	0.6835	50	div	50	div
3	11	0.0813	14	0.1311	19	0.2412	46	0.5573	50	0.6052
4	9	0.0394	11	0.0777	13	0.1196	16	0.1664	22	0.2836
5	7	0.0149	8	0.0370	9	0.0469	11	0.0865	15	0.1557
6	**5**	**0.0024**	**6**	**0.0064**	**7**	**0.0137**	**8**	**0.0185**	**9**	**0.0315**
7	-	-	5	00020	6	0.0050	6	0.0061	7	0.0171
Daubechies4-Wavelets										
0	50	div	50	div	50	div	50	div	50	0.9997
1	50	div	50	div	50	div	50	div	50	div
2	50	div	50	div	50	div	50	div	50	div
3	50	0.9533	50	div	50	div	50	div	50	div
4	21	0.2803	37	0.4921	50	div	50	div	50	div
5	50	div	50	0.6419	50	div	50	div	50	div
6	46	0.5644	19	0.2463	50	div	50	div	50	div
7	17	0.2183	15	0.1668	17	0.2102	30	0.3979	50	0.5912
8	-	-	-	-	-	-	20	0.2476	32	0.4223
Daubechies6-Wavelets										
0	50	div	50	div	50	div	50	div	50	div
1	50	div	50	div	50	div	50	div	50	0.9917
2	50	div	43	0.5415	47	0.5652	50	div	50	div
3	21	0.2805	50	0.5949	50	div	50	div	50	div
4	50	0.5991	34	0.4583	50	0.6804	50	div	50	div
5	50	0.5977	23	0.3134	50	0.7152	50	0.8164	50	div
6	22	0.2978	16	0.1925	16	0.1890	31	0.4151	40	0.5056
7	8	0.0330	11	0.0866	20	0.2639	23	0.3017	36	0.4687
8	-	-	-	-	-	-	19	0.2362	27	0.3578
Daubechies8-Wavelets										
0	50	div	50	div	50	div	50	0.9626	50	0.9931
1	49	0.5906	50	0.8842	50	div	50	div	50	div
2	31	0.4219	23	0.3093	50	0.9509	50	div	50	0.9205
3	50	div	19	0.2526	39	0.4989	50	0.7337	50	0.9174
4	50	div	14	0.1373	50	0.5995	23	0.2982	50	0.9921
5	50	div	43	0.5436	12	0,1088	23	0.2982	27	0.3645
6	8	0.0379	16	0.1820	14	0.1449	19	0.2367	31	0.4085
7	-	-	-	-	-	-	-	-	22	0.2805

Tabelle 4.11.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 1000, verschiedene Wavelets, tiefstmögliche Zerlegung, Verfahren 1

Daubechies2-Wavelets										
	Zahl der Unbekannten n									
Level-	256		512		1024		2048		4096	
tiefe	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
1	27	0.3731	21	0.2812	22	0.2847	24	0.3232	27	0.3498
2	50	0.6842	49	0.5843	46	0.5619	50	0.5947	50	0.6221
3	50	0.8626	50	0.7591	50	0.7187	50	0.7311	50	0.7484
4	50	0.9379	50	0.8628	50	0.8158	50	0.8154	50	0.8281
5	50	0.9497	50	0.9220	50	0.8724	50	0.8622	50	0.8740
Daubechies4-Wavelets										
1	26	0.3649	16	0.1821	13	0.1218	14	0.1462	17	0.1938
2	50	div	50	0.7115	25	0.3348	21	0.2745	25	0.3253
3	50	div	50	div	50	0.7469	33	0.4400	36	0.4634
4	50	div	50	div	50	div	50	0.7306	50	0.6124
5	50	div	50	div	50	div	50	div	50	0.7295
Daubechies6-Wavelets										
1	41	0.5172	31	0.4166	26	0.3372	20	0.2606	21	0.2426
2	50	div	50	div	28	0.2806	22	0.2843	21	0.2700
3	50	div	50	div	50	div	36	0.4742	50	0.5826
4	50	div	50	div	50	div	50	div	50	0.6713
5	50	div	50	div	50	div	50	div	50	div
Daubechies8-Wavelets										
1	29	0.2812	25	0.3305	21	0.2788	17	0.1669	20	0.2271
2	50	div	50	div	35	0.2806	24	0.3225	25	0.3322
3	50	div	50	div	50	div	50	0.7127	34	0.4404
4	50	div	50	div	50	div	50	div	50	0.8714
5	50	div	50	div	50	div	50	div	50	div

Tabelle 4.12.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 1000, verschiedene Wavelets, verschiedene Leveltiefen, Verfahren 2

	Zahl der Unbekannten n									
Level-	255		511		1023		2047		4095	
tiefe	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
1	25	0.3554	15	0.1673	12	0.0982	12	0.0905	12	0.0910
2	50	div	50	div	23	0.3041	14	0.1430	14	0.1257
3	50	div	50	div	50	div	41	0.5178	15	0.1595
4	50	div	50	div	50	div	50	div	50	0.6888
5	50	div	50	div	50	div	50	div	50	div

Tabelle 4.13.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 1D-Konvektionsdiffusionsoperator, Konvektionsstärke 1000, Prewavelets, verschiedene Leveltiefen, Verfahren 2

Daubechies2-Wavelets								
Band-	Zahl der Unbekannten n							
erwei-	256		1024		4096		16384	
terung	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
0	50	div	50	div	50	div	50	div
1	26	0.3356	50	0.7427	50	0.9232	50	0.9769
2	1	$1 \cdot 10^{-15}$	22	0.2582	50	0.6539	50	0.8883
3	-	-	20	0.2293	50	0.6531	50	0.8895
4	-	-	13	0.1067	45	0.5145	50	0.8303
5	-	-	10	0.0543	34	0.4183	50	0.7825
6	-	-	1	$1 \cdot 10^{-14}$	27	0.3245	50	0.7260
7	-	-	-	-	22	0.2473	50	0.6694
8	-	-	-	-	18	0.1833	50	0.6122
Daubechies4-Wavelets								
0	34	0.4319	50	0.7991	50	0.9415	50	0.9819
1	1	$1 \cdot 10^{-15}$	17	0.1822	41	0.4856	50	0.8120
2	-	-	23	0.2842	50	0.6948	50	0.9050
3	-	-	13	0.1055	43	0.4988	50	0.8211
4	-	-	11	0.0693	36	0.4343	50	0.7893
5	-	-	1	$1 \cdot 10^{-15}$	27	0.3313	50	0.7273
6	-	-	-	-	22	0.2583	50	0.6740
7	-	-	-	-	18	0.1942	50	0.6167
8	-	-	-	-	16	0.1455	50	0.5617
9	-	-	-	-	14	0.1069	45	0.5071
10	-	-	-	-	12	0.0771	39	0.4547
Daubechies6-Wavelets								
0	-	-	29	0.3663	50	0.6049	50	0.7524
1	-	-	32	0.4000	50	0.7265	50	0.9142
2	-	-	21	0.2467	44	0.5107	50	0.8102
3	-	-	19	0.2030	41	0.4830	50	0.7982
4	-	-	1	$1 \cdot 10^{-15}$	31	0.3854	50	0.7284
5	-	-	-	-	27	0.3301	50	0.6818
6	-	-	-	-	23	0.2760	50	0.6244
7	-	-	-	-	21	0.2348	50	0.5728
8	-	-	-	-	19	0.1998	47	0.5213
9	-	-	-	-	17	0.1705	41	0.4725
Daubechies8-Wavelets								
0	-	-	36	0.4433	50	0.7489	50	0.9213
1	-	-	23	0.1747	45	0.5173	50	0.7990
2	-	-	20	0.2322	44	0.5106	50	0.8054
3	-	-	1	$1 \cdot 10^{-14}$	34	0.4097	50	0.7269
4	-	-	-	-	29	0.3603	50	0.6875
5	-	-	-	-	25	0.3054	50	0.6284
6	-	-	-	-	23	0.2645	50	0.5798

Tabelle 4.14.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Laplaceoperator, verschiedene Wavelets, $lt = 1$, Verfahren 1

deutlich sichtbar bei der Banderweiterung um 4 Elemente, wo die Leveltiefe zum ersten Mal 2 beträgt. Für die anderen Wavelets und bei 16384 Unbekannten verlagert sich der Übergang zumeist in den vorkonvergenten Bereich.

Es ist also nicht erstaunlich, daß sich die Zeiten bei Problemgröße 1024 und die bei Problemgröße 4096 um einen Faktor unterscheiden, der wesentlich höher ist als 4. Etwas besser ist es zwischen den Problemgrößen 4096 und 16384. Hierbei ist zu berücksichtigen, daß die Banderweiterung e nach Abschnitt 3.2.4 einen quadratischen Einfluß auf den Rechenaufwand hat. Weiterhin bewirkt auch die zunehmende Breite des Bandes wie im Eindimensionalen mit längeren Waveletfiltern steigende Kosten.

Außerdem treten für Daubechies6- und Daubechies8-Wavelets im unteren Tabellenbereich bei 16384 Unbekannten Effekte auf, die durch die Rechnerarchitektur zu erklären sind. Ab einer bestimmten Problemgröße müssen Daten auf Festplatte ausgelagert werden, was zu einer Programmverlangsamung führt, da nicht gerechnet werden kann, solange der Prozessor auf die Daten wartet.

Verfahren ohne Schurkomplementkorrektur mit Wavelets

Tabelle 4.16 zeigt die Ergebnisse der Testreihe. Im Gegensatz zu den Experimenten mit dem eindimensionalen Laplaceoperator ist die Reduktionsrate des Verfahrens jetzt nicht mehr von der Problemgröße unabhängig. Von daher liegt hier kein $\mathcal{O}(n)$-Löser vor. Der Anstieg der Fehlerreduktionsraten mit wachsender Problemgröße läßt sich mit der Abhängigkeit der ILU-Zerlegung von n erklären. Diese Tatsache bewirkt auch, daß das Konvergenzverhalten besser wird, wenn man tiefer zerlegt. Der resultierende Mehraufwand und die durch die bessere Reduktionsrate gegebenenfalls kürzere Lösungszeit sind dabei gegeneinander abzuwägen. Auch im zweidimensionalen Fall gibt es eine maximale Leveltiefe, ab der man keine Verbesserungen in der Reduktionsrate mehr erzielen kann. Diese liegt da, wo das Programm aufgrund der Breite des benötigten Bandes in Relation zur Matrixgröße anfängt, auf der vollen Matrix zu rechnen. Ab hier löst das Programm exakt (bis auf Rundungsfehler). Das erklärt, warum für Daubechies6- und Daubechies8-Wavelets für $n = 256$ nur eine Iteration benötigt wird.

Die Länge der Waveletfilter hat offenbar ebenfalls einen Einfluß auf die Reduktionsrate. Dies läßt sich nicht nur mit der Zahl der verschwindenden Momente, sondern auch mit der Breite des betrachteten Bandes erklären. Je länger der Filter, desto breiter das betrachtete Band, desto besser die Reduktionsrate.

Verfahren ohne Schurkomplementkorrektur mit Prewavelets

Tabelle 4.17 gibt die Ergebnisse der zugehörigen Testreihe wieder. Zusätzlich zu den Problemgrößen in den beiden vorangegangen Testreihen wurden hier noch der entsprechende Fall von $n = 65536$ und $n = 262144$ untersucht. Da bei der verwendeten Prewaveletzer-

Daubechies2-Wavelets									
Band-erweiterung	Zahl der Unbekannten n								
	1024			4096			16384		
	k	$\varrho_{k,0}$	t[s]	k	$\varrho_{k,0}$	t[s]	k	$\varrho_{k,0}$	t[s]
0	50	div	1.62	50	div	6.18	50	div	-
1	50	0.8134	2.32	50	0.9558	9.12	50	0.9864	36.64
2	50	0.8171	4.7	50	div	17.73	50	div	70.88
3	**9**	**0.0343**	**3.17**	18	0.1857	17.05	46	0.5118	106.96
4	8	0.0254	4.5	**11**	**0.0641**	**52.71**	30	0.3584	216.92
5	7	0.0156	6.47	8	0.0212	63.02	**9**	**0.0255**	**225.77**
6	1	$1 \cdot 10^{-16}$	31.8	8	0.0206	74.43	9	0.0324	372.37
7	-	-	-	8	0.0208	94.89	8	0.0220	526.25
Daubechies4-Wavelets									
0	50	0.8562	3.59	50	0.9635	14.37	50	0.9881	58.25
1	50	div	5.96	50	div	24.75	50	div	103.67
2	16	0.1573	4.78	39	0.4613	30.46	50	0.7800	150.08
3	**9**	**0.0318**	**5.94**	24	0.2908	69.58	50	div	300.28
4	7	0.0154	8.19	**9**	**0.0368**	**71.93**	19	0.1903	300.96
5	1	$1 \cdot 10^{-15}$	33.54	6	0.0071	85.08	**10**	**0.0432**	**454.96**
6	-	-	-	6	0.0053	101.35	7	0.0131	612.61
Daubechies6-Wavelets									
0	50	div	7.77	50	div	35.57	50	div	144.79
1	21	0.2499	6.94	50	0.5811	45.75	50	0.8512	203.33
2	13	0.0935	8.35	50	0.8617	112.08	50	div	359.98
3	**8**	**0.0207**	**10.84**	13	0.1014	86.2	31	0.3714	407.44
4	1	$1 \cdot 10^{-15}$	35.37	**9**	**0.0292**	**97.64**	18	0.1816	579.49
5	-	-	-	6	0.0052	127.03	**9**	**0.0322**	**719.52**
6	-	-	-	5	0.0024	146.72	7	0.0083	1160.67
7	-	-	-	5	0.0019	199.5	6	0.0045	1490.84
Daubechies8-Wavelets									
0	26	0.3173	10.01	50	0.6513	62.05	50	0.8827	283.34
1	17	0.1747	11.77	50	div	126.46	50	div	460.78
2	**9**	**0.0329**	**14.33**	16	0.1556	109.85	42	0.4831	550.7
3	1	$1 \cdot 10^{-15}$	40.06	11	0.0638	124.55	35	0.4205	783.08
4	-	-	-	**7**	**0.0123**	**143.19**	12	0.0724	887.8
5	-	-	-	6	0.0043	176.04	**8**	**0.0216**	**1352.89**
6	-	-	-	5	0.0023	218.63	7	0.0085	1700.65

Tabelle 4.15.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, benötigte Zeit in s, 2D-Laplaceoperator, verschiedene Wavelets, tiefstmögliche Zerlegung, Verfahren 2

Daubechies2-Wavelets								
Level-tiefe	Zahl der Unbekannten n							
	256		1024		4096		16384	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
1	50	0.6503	50	0.8603	50	0.9557	50	0.9853
2	33	0.4256	50	0.8108	50	0.9249	50	0.9751
3	-	-	50	0.6988	50	0.8974	50	0.9586
4	-	-	-	-	50	0.8328	50	0.9422
5	-	-	-	-	-	-	50	0.9025
Daubechies4-Wavelets								
1	22	0.2755	50	0.5734	50	0.8489	50	0.9551
2	21	0.2595	37	0.4541	50	0.5926	50	0.8533
3	-	-	37	0.4517	50	0.6317	50	0.6521
4	-	-	-	-	50	0.6314	50	0.7851
5	-	-	-	-	-	-	50	0.7848
Daubechies6-Wavelets								
1	1	$1 \cdot 10^{-15}$	40	0.4797	50	0.7548	50	0.9228
2	-	-	19	0.2080	32	0.3979	50	0.6658
3	-	-	-	-	23	0.2761	50	0.6743
4	-	-	-	-	-	-	50	0.6593
Daubechies8-Wavelets								
1	1	$1 \cdot 10^{-15}$	31	0.3867	50	0.6643	50	0.8842
2	-	-	14	0.1165	36	0.4383	50	0.6081
3	-	-	-	-	32	0.3956	41	0.4775
4	-	-	-	-	32	0.3956	40	0.4697

Tabelle 4.16.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Laplace-operator, verschiedene Wavelets, verschiedene Leveltiefen, Verfahren 2

legung die beiden entstehenden Anteile unterschiedlich groß sind $(2^k + 2^{k+1})$, stimmen die Matrixgrößen nicht mit den bisher benutzten überein. So ist die frühere 64×64-Matrix jetzt eine 49×49-Matrix. Die Problemgrößen sind hier folglich nicht $2^l 2^l \times 2^l 2^l$, sondern $(2^l - 1)(2^l - 1) \times (2^l - 1)(2^l - 1)$.

Um einen Vergleich zwischen den Laufzeiten der Varianten mit Wavelets und der Prewaveletvariante durchführen zu können, müßte man also die Laufzeiten entsprechend umrechnen.

Die Prewaveletvariante besitzt auf jedem Level nahezu konstante Reduktionsraten. Zwar erreicht man mit der Banderweiterungsmethode stets eine bessere Reduktionsrate, aber durch den höheren Aufwand aufgrund des breiten Bandes ist dies erheblich teurer. Die Prewaveletstrategie dagegen führt bei tiefstmöglicher Zerlegung definitiv zu einem $\mathcal{O}(n)$-Löser (für den Laplaceoperator). Das gute Abschneiden dieser Variante im Vergleich zur Schurkomplementapproximation mit Wavelets ist darin begründet, daß mit Prewavelets die Randbedingungen Beachtung finden. Außerdem sind Prewavelets im Gegensatz zu den verwendeten Wavelets für den Laplaceoperator problemangepaßt.

Vergleichend läßt sich also festhalten:

- Mit der Banderweiterungsvariante lassen sich gute Reduktionsraten erzielen, dies geht allerdings nur bei Kosten der Ordnung $\mathcal{O}\left(n(\log_2(\sqrt{n}))^4\right)$. Zum Erreichen dieser guten Raten ist eine tiefstmögliche Zerlegung notwendig. Der Schwellenwert für zuverlässige Konvergenz liegt wie im Eindimensionalen bei $l - 2$.

- Die Waveletvariante ohne Schurkomplementkorrektur ist die schwächste der eingesetzten Varianten. Sie konvergiert zwar, aber die Reduktionsraten steigen mit der Problemgröße an.

- Auch bei der Prewaveletvariante ist die tiefstmögliche Zerlegung Voraussetzung für einen guten Löser, der dann in $\mathcal{O}(n)$ ein Resultat liefert. Insofern schlägt sie die anderen beiden Varianten für den Laplaceoperator, zumal der Aufwand geringer ist als der für die Banderweiterungsvariante und die Prewavelets für den Laplaceoperator problemangepaßt sind.

4.2.2. Konvektionsdiffusionsoperator

Die Fehlertoleranzgrenze lag beim Konvektionsdiffusionsoperator wiederum bei

$$\varepsilon = 10^{-10}.$$

Zunächst wurde mit Konvektion in x-Richtung, dann mit Konvektion in y-Richtung und schließlich mit Konvektion in Diagonalrichtung getestet.

lt	Zahl der Unbekannten n								
	225			961			3969		
	k	$\varrho_{k,0}$	t[s]	k	$\varrho_{k,0}$	t[s]	k	$\varrho_{k,0}$	t[s]
1	25	0.3552	0.41	50	0.7461	2.51	50	0.9236	10.23
2	**13**	**0.1319**	**0.36**	27	0.3695	2.03	50	0.7536	12.44
3	-	-	-	**17**	**0.2002**	**1.65**	28	0.3734	8.79
4	-	-	-	-	-	-	**17**	**0.2019**	**6.83**
	16129			65025			261121		
1	50	0.9771	41.68	50	0.9914	173.79	50	0.9962	835.29
2	50	0.9260	50.57	50	0.9777	210.15	50	0.9962	989.27
3	50	0.7555	52.84	50	0.9266	221.01	50	0.9778	1021.1
4	29	0.3747	37.19	50	0.7559	221.66	50	0.9267	1091.6
5	**18**	**0.2037**	**28.75**	30	0.3754	161.03	50	0.7560	1032.6
6	-	-	-	**18**	**0.2030**	**122.31**	30	0.3753	779.45
7	-	-	-	-	-	-	**19**	**0.2054**	**639.43**

Tabelle 4.17.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, benötigte Zeiten in s, 2D-Laplaceoperator, Prewavelets, Verfahren 2

Konvektion in x-Richtung

Es wurden zwei Testreihen durchgeführt, die erste mit einer Konvektionsstärke von 10, die zweite mit einer von 1000. Die Tabellen 4.18 und 4.19 enthalten die wesentlichen Testergebnisse der Banderweiterungsvariante. Für Leveltiefe eins gilt, daß es eine optimale Bandweite gibt, für die die Reduktionsrate am niedrigsten ist, zuvor fällt sie, und hernach steigt sie an, gelegentlich tritt sogar Divergenz auf. Meist ist die Reduktionsrate für Leveltiefe 1 bei schmalen Banderweiterungen besser als die für Leveltiefe 2, bevor sich dieses Verhältnis bei stärker erweiterten Bändern umkehrt. Für Leveltiefe 2 erreicht man ab einer gewissen Bandbreite sehr gute Reduktionsraten, die besten Resultate werden mit Daubechies4-Wavelets erzielt. Auf die Verwendung der Leveltiefe 3 sollte man verzichten, da es mit breiter werdendem Band ebenfalls eher zu Divergenz kommen kann. Es liegen zu wenig Testergebnisse vor, um Leveltiefe 4 beurteilen zu können. Dafür wären weitere Tests mit mindestens zwei größeren Problemen notwendig gewesen, was eine zu lange Rechenzeit benötigt hätte. Aufgrund des Übergehens des Upwindsternes in einen zentralen-Differenzen-Stern ist jedoch zu erwarten, daß die Ergebnisse mit steigender Zerlegungstiefe stets schlechter werden.

Tabelle 4.20 gibt die Ergebnisse der Variante ohne Schurkomplementkorrektur für verschiedene Wavelets und Konvektionsstärke 10 wieder. Auffallend ist, daß man im wesentlichen die besten Reduktionsraten für gerade Leveltiefen erzielt. Auch hier werden die Reduktionsraten im allgemeinen mit steigender Problemgröße n schlechter.

Für Prewavelets sehen die Ergebnisse etwas besser aus. Abgesehen davon, daß man hier ebenfalls nur für gerade Leveltiefen gute Reduktionsraten bekommt, ist das Verhalten im Ansatz dem für den Laplaceoperator ähnlich. Die Resultate dieser Testreihe finden sich in

Band-erweiterung	256 k	256 $\varrho_{k,0}$	1024 k	1024 $\varrho_{k,0}$	4096 k	4096 $\varrho_{k,0}$	16384 k	16384 $\varrho_{k,0}$
Daubechies2-Wavelets								
0	50	div	50	div	50	div	50	div
1	50	0.8074	50	0.5846	50	0.8541	50	0.9709
2	50	0.8863	50	0.9222	50	0.6956	50	0.7485
3	-	-	50	0.8141	50	0.6974	50	0.7545
4	-	-	50	0.9441	50	0.4350	41	0.5035
5	-	-	50	0.9494	50	0.8683	50	0.5728
6	-	-	50	0.9491	50	0.9120	50	0.6257
7	-	-	-	-	50	0.9385	50	0.6851
Daubechies4-Wavelets								
0	50	0.8062	50	0.5920	50	0.8976	50	0.9789
1	50	0.8863	50	0.9870	50	0.8839	50	0.7014
2	-	-	50	0.9021	50	0.6512	50	0.8004
3	-	-	50	0.9427	50	0.8099	42	0.5124
4	-	-	50	0.9452	50	0.8579	45	0.5367
5	-	-	50	0.9491	50	0.9064	50	0.6132
6	-	-	-	-	50	0.9321	50	0.6648
7	-	-	-	-	50	0.9493	50	0.7216
8	-	-	-	-	50	0.9602	50	0.7728
Daubechies6-Wavelets								
0	50	0.8863	50	0.9476	50	div	50	div
1	-	-	50	0.9158	50	0.6676	50	0.8257
2	-	-	50	0.9284	50	0.8382	47	0.5469
3	-	-	50	0.9406	50	0.8643	46	0.5441
4	-	-	50	0.9491	50	0.9212	50	0.6236
5	-	-	-	-	50	0.9485	50	0.6744
7	-	-	-	-	50	0.9785	50	0.7848
Daubechies8-Wavelets								
0	50	0.8863	50	0.9067	50	0.6572	50	0.8445
1	-	-	50	0.9158	50	0.8533	50	0.5758
2	-	-	50	0.9383	50	0.8574	46	0.5432
3	-	-	50	0.9491	50	0.9197	50	0.6299
4	-	-	-	-	50	0.9452	50	0.6729
5	-	-	-	-	50	0.9660	50	0.7366

(Column groups 256, 1024, 4096, 16384: Zahl der Unbekannten n)

Tabelle 4.18.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in x-Richtung, verschiedene Wavelets, $lt = 1$, Verfahren 1

Band-erweiterung	1024 k	1024 $\varrho_{k,0}$	1024 t[s]	4096 k	4096 $\varrho_{k,0}$	4096 t[s]	16384 k	16384 $\varrho_{k,0}$	16384 t[s]
Daubechies2-Wavelets									
0	50	div	1.51	50	div	5.84	50	div	-
1	50	0.6802	2.19	50	0.9173	8.92	50	0.9823	34.88
2	21	0.2694	2.9	50	div	16.04	50	div	64.24
3	**14**	**0.1338**	**3.41**	13	0.1057	14.88	37	0.4680	83.49
4	14	0.1347	4.9	**11**	**0.0764**	**56.42**	13	0.1060	122.91
5	14	0.1353	7.13	11	0.0736	65.39	**12**	**0.0780**	**179.42**
6	-	-	-	11	0.0739	79.75	10	0.0506	325.8
7	-	-	-	11	0.0738	95.94	10	0.0501	456.01
Daubechies4-Wavelets									
0	50	0.7588	3.44	50	0.9391	13.69	50	0.9859	55.2
1	50	div	5.96	50	div	22.92	50	div	-
2	12	0.1093	4.43	24	0.3196	24.58	50	0.6714	131.13
3	**13**	**0.1144**	**6.35**	13	0.1119	64.79	26	0.3391	182.47
4	12	0.1099	8.79	**9**	**0.0472**	**73.1**	17	0.1819	240.8
5	-	-	-	9	0.0458	88.18	**9**	**0.0365**	**392.89**
6	-	-	-	9	0.0460	106.25	8	0.0289	542.36
Daubechies6-Wavelets									
0	50	div	8.03	50	div	32.6	50	div	132.66
1	16	0.1790	6.56	35	0.4559	38.26	50	0.7630	178.85
2	**13**	**0.1223**	**8.35**	26	0.3471	84.38	50	div	289.04
3	13	0.1215	11.45	**10**	**0.0616**	**83.35**	24	0.3123	324.09
4	-	-	-	10	0.0527	100.65	**12**	**0.0973**	**496.62**
5	-	-	-	10	0.0493	122.92	13	0.1073	667.33
Daubechies8-Wavelets									
0	19	0.2398	9.18	43	0.5273	56.29	50	0.8080	244.49
1	**14**	**0.1302**	**11.31**	50	0.6269	118.44	50	div	377.53
2	13	0.1244	14.9	**12**	**0.0917**	**101.8**	30	0.3878	435.02
3	-	-	-	10	0.0567	121.22	**13**	**0.1017**	**626.47**
4	-	-	-	10	0.0567	147.86	14	0.1324	820.58
5	-	-	-	10	0.0573	183.5	10	0.0549	1278.29

(Column groups 1024, 4096, 16384: Zahl der Unbekannten n)

Tabelle 4.19.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, benötigte Zeit in s, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in x-Richtung, verschiedene Wavelets, $lt = 2$, Verfahren 1

Daubechies2-Wavelets								
	Zahl der Unbekannten n							
Level-	256		1024		4096		16384	
tiefe	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
1	50	0.5917	50	0.6850	50	0.9280	50	0.9835
2	24	0.3411	50	0.6629	50	0.8650	50	0.9689
3	-	-	50	0.7263	50	0.8224	50	0.9382
4	-	-	-	-	50	0.7913	50	0.9124
5	-	-	-	-	-	-	50	0.8821
Daubechies4-Wavelets								
1	50	0.8158	50	0.7802	50	0.5982	50	0.9249
2	20	0.2661	29	0.3890	34	0.4435	50	0.7264
3	-	-	50	0.8692	50	0.8531	50	0.7958
4	-	-	-	-	50	0.6098	50	0.7757
5	-	-	-	-	-	-	50	0.8609
Daubechies6-Wavelets								
1	50	0.8863	50	0.8746	50	0.6392	50	0.8458
2	-	-	16	0.1836	21	0.2655	34	0.4378
3	-	-	-	-	50	0.9610	50	0.9440
4	-	-	-	-	-	-	50	0.6495
Daubechies8-Wavelets								
1	50	0.8863	50	0.9067	50	0.7357	50	0.7330
2	-	-	13	0.1259	29	0.3807	40	0.4911
3	-	-	-	-	50	0.9763	50	0.9861
4	-	-	-	-	28	0.3695	37	0.4651

Tabelle 4.20.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in x-Richtung, verschiedene Wavelets, verschiedene Leveltiefen, Verfahren 2

Tabelle 4.21.

Während also die Variante mit Wavelets ohne Schurkomplementkorrektur für dieses Problem ungeeignet erscheint, sind die anderen beiden Verfahren hier offensichtlich besser. Es fragt sich nun, ob dies auch bei einer Konvektionsstärke von 1000 noch so bleibt.

Tabelle 4.22 enthält sämtliche Fälle, für die bei Konvektionsstärke 1000 Konvergenz erzielt werden konnte. In allen anderen Fällen kam es zur Divergenz. Selbst eine Verbreiterung des Bandes um 12 Elemente (zusammen mit Haarwavelets) brachte bei Problemgröße 16384 keine Verbesserung. Interessant ist, daß das Verfahren in sämtlichen in der Tabelle aufgeführten Fällen zunächst nicht konvergierte, bevor es dann in einigen wenigen Schritten zur Unterschreitung der Fehlertoleranzgrenze kam. Dies erklärt die große Differenz zwischen der Fehlerreduktionsrate im letzten Schritt und dem geometrischen Mittel der Fehlerreduktionsraten über den gesamten Lauf.

Mit diesen Ergebnissen stand fest, daß die getesteten Verfahren ohne zusätzliche Modifikatio-

Level-	Zahl der Unbekannten n									
	49		225		961		3969		16129	
tiefe	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
1	50	0.6699	50	0.7798	50	0.5882	50	0.8543	50	0.9709
2	30	0.4328	**21**	**0.2835**	**16**	**0.1758**	50	0.5822	50	0.8728
3	-	-	50	0.8442	50	0.9317	50	0.9119	50	0.6280
4	-	-	-	-	44	0.5451	**23**	**0.2982**	**18**	**0.2089**
5	-	-	-	-	-	-	50	0.9556	50	0.9852
6	-	-	-	-	-	-	-	-	50	0.5694

Tabelle 4.21.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in x-Richtung, Prewavelets, verschiedene Leveltiefen, Verfahren 2

n	Wavelets	Banderw.	k	$\varrho_{k,k-1}$	$\varrho_{k,0}$
49	Prewavelets	-	23	0.158	0.3149
64	Daubechies2	-	24	0.040	0.3275
64	Daubechies4	-	28	0.054	0.3908
64	Daubechies6	-	29	0.217	0.4174
64	Daubechies8	-	28	0.014	0.4078
256	Daubechies2	2	50	0.052	0.5649

Tabelle 4.22.: Konvergente Fälle für 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 1000 in x-Richtung, $lt = 1$. Die Ergebnisse für Problemgröße $n = 64$ gelten sowohl für Verfahren 1 als auch für Verfahren 2, da in beiden Fällen auf der vollen Matrix gerechnet wird.

nen keine zufriedenstellenden Löser mehr sein konnten. Das schlechte Verhalten sämtlicher Varianten liegt daran, daß für achsenparallele Konvektion weder Wavelets noch Prewavelets problemangepaßt sind. Während die verwendeten Daubechies-Wavelets insbesondere für starke Konvektion gut geeignet sind, ist dies bei Prewavelets nur für geringe oder gar keine Konvektion der Fall. Bei achsenparalleler Konvektion müßte man folglich mit einem Tensorprodukt aus Wavelets und Prewavelets gute Ergebnisse erzielen können. Die weiter unten in diesem Abschnitt beschriebenen operatorabhängigen Prewaveletfilter setzen diese Idee um. Ebenso müßten die Wavelets bei Diagonalkonvektion im Gegensatz zu den Prewavelets gut funktionieren.

Konvektion in y-Richtung

Es wurde eine Versuchsreihe mit Konvektionsstärke 10 durchgeführt. Auf die tabellarische Darstellung von Ergebnissen dieser Testreihe wird hier verzichtet, da sie nahezu identisch mit denen bei Konvektionsstärke 10 in x-Richtung sind. Dies liegt daran, daß man den einen Operator aus dem anderen herstellen kann, indem man die Zeilenkopplungen und die Spaltenkopplungen miteinander vertauscht, oder algebraisch ausgedrückt, die Koordinaten auf den Blöcken mit den Blockkoordinaten vertauscht. Aufgrund dieser Symmetrie der beiden Operatoren wurde auf eine Testreihe mit Konvektionsstärke 1000 in y-Richtung ganz verzichtet, da auch ihre Ergebnisse nahezu identisch mit denen der entsprechenden Versuche in x-Richtung gewesen wären.

Konvektion in Diagonalrichtung

Für diese Konvektionsrichtung wurde wieder sowohl mit Stärke 10 als auch 1000 getestet. Tabellen 4.23 und 4.24 geben die Resultate für die Banderweiterungsvariante mit Wavelets wieder, und Tabellen 4.25 und 4.26 zeigen die Ergebnisse für die Variante ohne Schurkomplementkorrektur mit Wavelets und Prewavelets, jeweils für Konvektionsstärke 10.

Das Verfahren mit Schurkomplementapproximation zeigt bei Leveltiefe 1 zwar bei einer hinreichend großen Bandverbreiterung ein gutes Konvergenzverhalten, dieses wird jedoch mit steigender Problemgröße zunehmend schlechter. Je breiter das Band gemacht wird, desto geringer wird die Fehlerreduktionsrate. Allerdings verlangsamt sich dieses Absinken ebenfalls mit steigender Problemgröße. Insgesamt erscheint die ein Level tiefe Zerlegung in diesem Fall nur bedingt sinnvoll.

Verwendet man stattdessen die tiefstmögliche Zerlegung, so bekommt man das vom Laplaceoperator her bekannte gute Verhalten des Verfahrens. Der Schwellenwert für gute Konvergenz liegt im Bereich $l-2$ oder $l-3$, bei Annahme von $l-2$ erhält man in den getesten Fällen stets einstellige Iterationszahlen. Wie für den Laplaceoperator führt dies zu einem $\mathcal{O}\left(n(\log_2(\sqrt{n}))^4\right)$-Löser. Die gemessenen Zeiten ergeben aus den bekannten Gründen auch hier kein einheitliches Bild.

Für das Verfahren ohne Schurkomplementkorrektur zeigt sich, daß die Reduktionsrate wieder mit steigender Länge der verwendeten Waveletfilter sinkt. Außerdem steigt sie mit der Problemgröße. Auch bei der Verwendung von Prewavelets entstehen ähnliche Resultate wie für den Laplaceoperator. Je tiefer man zerlegt, desto besser das Verfahren. Allerdings steigen die Reduktionsraten jetzt mit der Problemgröße leicht an.

Für Konvektionsstärke 1000 zeigt sich ein leicht verändertes Bild: Das Verhalten der Banderweiterungsvariante bei ein Level tiefer Zerlegung ist wesentlich besser als das bei Konvektionsstärke 10 (Tabelle 4.27). Der Schwellenwert für gute Konvergenz läßt sich jetzt sogar waveletunabhängig bei $l-4$ ansetzen. Bei tiefstmöglicher Zerlegung (Tabelle 4.28) zeigt

Daubechies2-Wavelets								
Band-	Zahl der Unbekannten n							
erwei-	256		1024		4096		16384	
terung	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
0	50	div	50	div	50	div	50	div
1	14	0.1473	38	0.4971	50	0.8238	50	0.9655
2	1	$1 \cdot 10^{-15}$	10	0.0656	24	0.3163	50	0.7205
3	-	-	10	0.0544	26	0.3406	50	0.7287
4	-	-	7	0.0217	17	0.1968	50	0.5949
5	-	-	-	-	14	0.1301	42	0.5083
6	-	-	-	-	11	0.0783	33	0.4203
7	-	-	-	-	-	-	27	0.3450
Daubechies4-Wavelets								
0	17	0.2190	50	0.5864	50	0.8724	50	0.9759
1	1	$1 \cdot 10^{-15}$	13	0.1126	22	0.2827	44	0.5290
2	-	-	11	0.0867	30	0.3993	50	0.7686
3	-	-	6	0.0110	16	0.1775	50	0.5739
4	-	-	6	0.0087	14	0.1424	43	0.5199
5	-	-	-	-	11	0.0823	33	0.4207
6	-	-	-	-	-	-	27	0.3498
Daubechies6-Wavelets								
0	1	$1 \cdot 10^{-15}$	17	0.2053	40	0.5049	50	0.7098
1	-	-	17	0.2027	37	0.4747	50	0.7962
2	-	-	9	0.0487	19	0.2294	47	0.5504
3	-	-	10	0.0593	18	0.2173	46	0.5432
4	-	-	1	$1 \cdot 10^{-15}$	15	0.1491	34	0.4307
5	-	-	-	-	13	0.1178	29	0.3736
Daubechies8-Wavelets								
0	1	$1 \cdot 10^{-15}$	19	0.2342	41	0.5140	50	0.8160
1	-	-	9	0.0505	20	0.2446	43	0.5216
2	-	-	10	0.0681	20	0.2430	49	0.5599
3	-	-	1	$1 \cdot 10^{-15}$	15	0.1620	34	0.4319
4	-	-	-	-	14	0.1320	30	0.3875

Tabelle 4.23.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in Diagonalrichtung, verschiedene Wavelets, $lt = 1$, Verfahren 1

Daubechies2-Wavelets									
Band-erwei-terung	Zahl der Unbekannten n								
	1024			4096			16384		
	k	$\varrho_{k,0}$	t[s]	k	$\varrho_{k,0}$	t[s]	k	$\varrho_{k,0}$	t[s]
0	50	div	1.54	50	div	6.02	50	div	-
1	50	0.6151	2.17	50	0.9085	8.73	50	0.9843	35.75
2	16	0.1781	2.25	50	div	16.2	50	div	67.15
3	**7**	**0.0182**	**2.61**	10	0.0574	13.43	19	0.2203	71.15
4	7	0.0173	3.83	**8**	**0.0275**	**48.56**	13	0.1002	155.48
5	7	0.0157	5.73	7	0.0179	54.55	**8**	**0.0238**	**207.02**
6	-	-	-	7	0.0173	65.84	8	0.0303	343.23
7	-	-	-	7	0.0174	82.2	7	0.0185	463.84
Daubechies4-Wavelets									
0	50	0.7251	3.42	50	0.9310	13.84	50	0.9864	56.44
1	50	0.6754	5.12	50	div	22.89	50	div	96.68
2	11	0.0796	3.84	18	0.2103	21.63	46	0.5406	135.29
3	**7**	**0.0148**	**5.14**	11	0.0833	54.54	47	0.5519	263.82
4	6	0.0083	7.29	**7**	**0.0132**	**62.23**	11	0.0700	257.79
5	-	-	-	6	0.0067	74.98	**7**	**0.0153**	**411.58**
6	-	-	-	5	0.0039	92.5	7	0.0111	558.92
Daubechies6-Wavelets									
0	50	div	6.81	50	div	32.26	50	div	137.39
1	15	0.1651	5.72	25	0.3229	33.31	50	0.6628	192.22
2	**10**	**0.0588**	**7.26**	20	0.2435	71.68	50	div	332.81
3	7	0.0142	9.7	**9**	**0.0469**	**75.62**	17	0.1805	346.45
4	-	-	-	7	0.0125	90.13	**10**	**0.0531**	**518.62**
5	-	-	-	5	0.0027	108.34	7	0.0118	663.91
Daubechies8-Wavelets									
0	18	0.2161	8.22	33	0.4348	49.79	50	0.7427	260.58
1	**12**	**0.1070**	**10.06**	32	0.4233	93.52	50	div	437.25
2	8	0.0253	12.94	**11**	**0.0789**	**90.78**	21	0.2579	451.09
3	-	-	-	8	0.0309	111.44	**14**	**0.1187**	**653**
4	-	-	-	6	0.0073	130.62	8	0.0268	815.23
5	-	-	-	5	0.0023	164.3	6	0.0079	1250.93

Tabelle 4.24.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in Diagonalrichtung, verschiedene Wavelets, tiefstmögliche Zerlegung, Verfahren 1

Level-tiefe	256 k	256 $\varrho_{k,0}$	1024 k	1024 $\varrho_{k,0}$	4096 k	4096 $\varrho_{k,0}$	16384 k	16384 $\varrho_{k,0}$
Daubechies2-Wavelets								
1	30	0.4177	50	0.6295	50	0.9169	50	0.9819
2	22	0.2985	50	0.6858	50	0.8212	50	0.9624
3	-	-	50	0.6177	50	0.8315	50	0.9228
4	-	-	-	-	50	0.7816	50	0.9045
5	-	-	-	-	-	-	50	0.8652
Daubechies4-Wavelets								
1	15	0.1617	21	0.2781	50	0.6318	50	0.9055
2	14	0.1451	24	0.3185	30	0.3975	50	0.6192
3	-	-	23	0.3151	45	0.5455	50	0.6021
4	-	-	-	-	46	0.5488	50	0.7447
5	-	-	-	-	-	-	50	0.7457
Daubechies6-Wavelets								
1	1	$1 \cdot 10^{-15}$	20	0.2565	36	0.4655	50	0.8154
2	-	-	15	0.1589	20	0.2474	29	0.3694
3	-	-	-	-	19	0.2377	50	0.6521
4	-	-	-	-	-	-	50	0.6407
Daubechies8-Wavelets								
1	1	$1 \cdot 10^{-15}$	16	0.1746	26	0.3463	50	0.7149
2	-	-	12	0.1013	28	0.3743	39	0.4849
3	-	-	-	-	27	0.3579	37	0.4664
4	-	-	-	-	28	0.3713	36	0.4609

Tabelle 4.25.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in Diagonalrichtung, verschiedene Wavelets, verschiedene Leveltiefen, Verfahren 2

Level-tiefe	49 k	49 $\varrho_{k,0}$	225 k	225 $\varrho_{k,0}$	961 k	961 $\varrho_{k,0}$	3969 k	3969 $\varrho_{k,0}$	16129 k	16129 $\varrho_{k,0}$
1	1	$1 \cdot 10^{-15}$	**13**	**0.1211**	37	0.4818	50	0.8236	50	0.9657
2	-	-	-	-	**14**	**0.1482**	36	0.4656	50	0.8229
3	-	-	-	-	-	-	**16**	**0.1758**	36	0.4523
4	-	-	-	-	-	-	-	-	**17**	**0.1819**

Tabelle 4.26.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 10 in Diagonalrichtung, Prewavelets, verschiedene Leveltiefen, Verfahren 2

Daubechies2-Wavelets								
Band- erwei- terung	Zahl der Unbekannten n							
	256		1024		4096		16384	
	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$	k	$\varrho_{k,0}$
0	**12**	**0.1043**	23	0.3153	50	0.6947	50	div
1	7	0.0176	**11**	**0.0820**	19	0.2065	32	0.4065
2	1	$1 \cdot 10^{-15}$	8	0.0311	**12**	**0.1019**	19	0.2114
3	-	-	7	0.0156	8	0.0267	**12**	**0.0738**
4	-	-	6	0.0117	8	0.0228	9	0.0394
5	-	-	-	-	7	0.0149	7	0.0157
Daubechies4-Wavelets								
0	**13**	**0.1166**	20	0.2463	38	0.4806	50	0.6745
1	1	$1 \cdot 10^{-15}$	**16**	**0.1869**	47	0.5585	50	div
2	-	-	9	0.0395	**13**	**0.1227**	20	0.2316
3	-	-	7	0.0211	10	0.0640	**15**	**0.1471**
4	-	-	6	0.0117	8	0.0256	10	0.0489
5	-	-	-	-	-	-	6	0.0058
Daubechies6-Wavelets								
0	1	$1 \cdot 10^{-15}$	44	0.5416	50	div	50	div
1	-	-	**17**	**0.2080**	24	0.3184	33	0.4279
2	-	-	13	0.1161	**19**	**0.2303**	31	0.4050
3	-	-	11	0.0759	13	0.1174	**16**	**0.1635**
4	-	-	1	$1 \cdot 10^{-15}$	10	0.0543	12	0.0783
5	-	-	-	-	-	-	10	0.0587
Daubechies8-Wavelets								
0	1	$1 \cdot 10^{-15}$	14	0.1384	20	0.2563	30	0.3850
1	-	-	**12**	**0.0954**	17	0.1919	29	0.3789
2	-	-	9	0.0473	**12**	**0.0896**	15	0.1449
3	-	-	1	$1 \cdot 10^{-15}$	10	0.0634	**12**	**0.0971**

Tabelle 4.27.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 1000 in Diagonalrichtung, verschiedene Wavelets, $lt = 1$, Verfahren 1

Band-erweiterung	1024 k	1024 $\varrho_{k,0}$	1024 t[s]	4096 k	4096 $\varrho_{k,0}$	4096 t[s]	16384 k	16384 $\varrho_{k,0}$	16384 t[s]
Daubechies2-Wavelets									
0	50	div	1.57	50	div	6.05	50	div	24.4
1	15	0.1623	1.12	50	0.6892	8.73	50	div	35.12
2	8	0.0310	1.88	13	0.1216	9.12	39	0.4813	58.85
3	**7**	**0.0168**	**2.58**	9	0.0430	13.18	14	0.1208	66.36
4	7	0.0158	3.85	**8**	**0.0234**	**46.58**	11	0.0619	153.56
5	7	0.0157	5.76	7	0.0186	55.58	**9**	**0.0428**	**211.57**
6	-	-	-	7	0.0185	69.02	9	0.0366	348.24
7	-	-	-	7	0.0184	83.4	9	0.0339	471.13
Daubechies4-Wavelets									
0	50	div	3.41	50	div	13.87	50	div	56.08
1	50	0.5824	5.16	50	div	22.88	50	div	96.08
2	13	0.1130	3.98	24	0.3088	23.48	50	div	140.57
3	**9**	**0.0458**	**5.29**	13	0.1229	55.94	50	div	263.21
4	7	0.0138	7.4	**8**	**0.0275**	**65.38**	21	0.2566	279.36
5	-	-	-	7	0.0131	75.81	**11**	**0.0657**	**422.76**
6	-	-	-	6	0.0081	94.52	10	0.0482	569.55
Daubechies6-Wavelets									
0	50	div	6.88	50	div	32.24	50	div	136.67
1	19	0.2316	6.04	50	0.7114	43.26	50	0.5814	192.06
2	**15**	**0.1647**	**7.74**	25	0.3317	75.46	50	div	333.14
3	10	0.0546	10.06	**12**	**0.0943**	**79.38**	29	0.3824	371.46
4	-	-	-	9	0.0409	93.54	**16**	**0.1591**	**533.5**
5	-	-	-	7	0.0131	115.69	11	0.0704	676.63
Daubechies8-Wavelets									
0	17	0.2019	8.13	24	0.3201	45.28	50	0.7716	261.67
1	**16**	**0.1800**	**10.49**	27	0.3645	90.74	50	div	423.74
2	10	0.0648	13.2	**13**	**0.1108**	**96.69**	26	0.3316	461.94
3	-	-	-	11	0.0693	113.74	**12**	**0.0974**	**641.82**
4	-	-	-	7	0.0193	132.02	9	0.0409	814.28
5	-	-	-	7	0.0120	170.34	11	0.0647	1274.03

(Spaltengruppen: Zahl der Unbekannten n = 1024, 4096, 16384)

Tabelle 4.28.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, benötigte Zeit in s, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 1000 in Diagonalrichtung, verschiedene Wavelets, tiefstmögliche Zerlegung, Verfahren 1

Level-tiefe	256 k	256 $\varrho_{k,0}$	1024 k	1024 $\varrho_{k,0}$	4096 k	4096 $\varrho_{k,0}$	16384 k	16384 $\varrho_{k,0}$
Daubechies2-Wavelets								
1	26	0.3513	34	0.4495	46	0.5446	50	0.5817
2	26	0.3484	39	0.5053	50	0.5979	50	0.7578
3	-	-	39	0.5058	50	0.6432	50	0.7669
4	-	-	-	-	50	0.6430	50	0.7706
5	-	-	-	-	-	-	50	0.7707
Daubechies4-Wavelets								
1	50	0.6834	50	0.7287	50	0.6911	50	0.5785
2	50	0.6768	50	div	50	div	50	div
3	-	-	50	div	50	div	50	div
4	-	-	-	-	50	div	50	div
5	-	-	-	-	-	-	50	div
Daubechies6-Wavelets								
1	1	$1 \cdot 10^{-15}$	50	0.7027	50	0.6837	50	0.5804
2	-	-	50	0.8389	50	div	50	div
3	-	-	-	-	50	div	50	div
4	-	-	-	-	-	-	50	div
Daubechies8-Wavelets								
1	1	$1 \cdot 10^{-15}$	50	0.7517	50	0.7106	50	0.5786
2	-	-	50	0.9595	50	div	50	div
3	-	-	-	-	50	div	50	div
4	-	-	-	-	50	div	50	div

(Zahl der Unbekannten n)

Tabelle 4.29.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 1000 in Diagonalrichtung, verschiedene Wavelets, verschiedene Leveltiefen, Verfahren 2

Level-tiefe	49 k	49 $\varrho_{k,0}$	225 k	225 $\varrho_{k,0}$	961 k	961 $\varrho_{k,0}$	3969 k	3969 $\varrho_{k,0}$	16129 k	16129 $\varrho_{k,0}$
1	1	$1 \cdot 10^{-15}$	21	0.2914	28	0.3793	37	0.4657	41	0.4994
2	-	-	21	0.2830	50	div	50	div	50	div
3	-	-	-	-	50	div	50	div	50	div
4	-	-	-	-	-	-	50	div	50	div

(Zahl der Unbekannten n)

Tabelle 4.30.: Iterationszahlen k, gemittelte Fehlerreduktionsraten $\varrho_{k,0}$, 2D-Konvektionsdiffusionsoperator, Konvektion der Stärke 1000 in Diagonalrichtung, Prewavelets, verschiedene Leveltiefen, Verfahren 2 sich eine ähnliche Situation wie bei Konvektionsstärke 10: es fällt schwer, einen einheitlichen waveletunabhängigen Schwellenwert zu finden. Für Haar- und Daubechies8-Wavelets könnte man $l-4$ verwenden, für Daubechies6-Wavelets noch $l-3$, für Daubechies4-Wavelets jedoch höchstens $l-2$.

Insofern ist für Konvektion in Diagonalrichtung der Stärke 1000 die ein Level tiefe Zerlegung der tiefstmöglichen Zerlegung überlegen, auch wenn letztere bei genügend großer Banderweiterung ebenfalls gute Fehlerreduktionsraten aufweist. Es stellt sich damit genau das vorhergesagte Verhalten ein, daß die problemangepaßten Daubechies- und insbesondere Haarwavelets bei Diagonalkonvektion wieder gute Ergebnisse liefern. Zur Überprüfung dieses Verhaltens wurde eine weitere Testreihe mit Konvektionsstärke 100000 durchgeführt, auf deren tabellarische Darstellung hier verzichtet wird, da sie die guten Resultate bestätigen konnte.

Das Verfahren ohne Schurkomplementkorrektur konvergiert jetzt – mit Ausnahme des Falls von Haarwavelets – nur noch für Leveltiefe 1 (Tabelle 4.29). Für Haarwavelets liefert es die besten Ergebnisse, allerdings steigt auch hier die Reduktionsrate mit der Problemgröße an, ab Leveltiefe 2 bleibt die Reduktionsrate zwar stehen, ist aber höher als die bei Leveltiefe 1. Dieses Verhalten weisen auch Prewavelets auf (Tabelle 4.30). Für Leveltiefe 1 erhält man ein relativ gut konvergentes Verfahren, welches ab Leveltiefe 2 und Problemgröße 961 divergiert. Interessant ist dieses Verhalten besonders deswegen, weil es dem bisherigen Verhalten der Prewaveletvariante genau entgegengesetzt ist. Bisher lautete die Strategie für den besten Löser immer „zerlege so tief wie möglich“, jetzt lautet sie „zerlege genau ein Level tief“. Das schlechte Abschneiden der Prewavelets bei Diagonalkonvektion läßt sich – wie vorhergesagt – auf die Tatsache zurückführen, daß sie hiefür nicht problemangepaßt sind.

Betrachtet man die Ergebisse unter dem Aspekt der Abhängigkeit von den verwendeten Wavelets, so gelangt man zu der Erkenntnis, daß es am besten sein müßte, für jeden Operator stets problemangepaßte Wavelets zu benutzen. Diese Problemanpassung erreicht man, indem man gemäß [27] den Grobfilter so aufbaut, daß er die Lösungsfunktion des Operators gut approximiert. Für Konvektionsdiffusionsoperatoren mit Konvektion b_x in x-Richtung und b_y in y-Richtung erhält man als Filterkoeffizienten für den Anteil in x-Richtung

$$\left[\begin{array}{ccc} \frac{1}{2}(1-\tanh(\frac{b_x h_j}{2})) & 1 & \frac{1}{2}(1+\tanh(\frac{b_x h_j}{2})) \end{array}\right]$$

und entsprechend für den Anteil in y-Richtung

$$\left[\begin{array}{c} \frac{1}{2}(1+\tanh(\frac{b_y h_j}{2})) \\ 1 \\ \frac{1}{2}(1-\tanh(\frac{b_y h_j}{2})) \end{array}\right].$$

Hierbei steht h_j für die Gitterweite auf Level j. Die beiden richtungsabhängigen Filter werden über ein Tensorprodukt miteinander oder mit dem Feinfilter verknüpft. Wegen $\tanh(0)=0$

und $\tanh(x) = 1$ für große x erhält man auf diese Weise je nach Konvektionsstärke b eine Mischung aus Prewavelet- und Haarwaveletfilter.

Dieser operatorabhängige Grobfilter wurde bei einem weiteren Versuch eingesetzt. Als Feinfilter wurden die frei gewählten Filterkoeffizienten

$$\left[\begin{array}{ccc} 1 & -\frac{1}{2} & 0 \end{array}\right], \left[\begin{array}{ccccc} 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \end{array}\right], \left[\begin{array}{ccc} 0 & -\frac{1}{2} & 1 \end{array}\right]$$

verwendet.

Zusätzlich wurde das Verfahren modifiziert. So wurde das Schurkomplement zwar nicht explizit von Level zu Level mitberechnet, aber dennoch wurde zur Berechnung der Matrizen B_i und C_i jeweils auf dem vorangegangen Level die Matrix T_{i-1} durch $T_{i-1} - C_{i-1} \cdot A_{i-1}^{-1} \cdot B_{i-1}$ ersetzt. Auf diese Weise entstand eine Mischung aus den beiden bisher benutzten Verfahren 1 und 2.

Mit diesen Einstellungen wurden die in Tabelle 4.31 dargestellten durchschnittlichen Reduktionsraten gemessen, jeweils nach dem Minimum aus der maximalen Iterationszahl von 50 und der zur Unterschreitung der Fehlertoleranzgrenze $\varepsilon = 10^{-10}$ benötigten Anzahl von Iterationen. (Zur Diskussion über die Messung von Reduktionsraten siehe Abschnitt 3.3.) Von den drei Testvektoren wird der Durchschnitt der für sie gemessenen Reduktionsraten wiedergegeben. Es wurde jeweils so tief wie möglich zerlegt.

Zur Erzielung dieser Ergebnisse wurde ein Programm verwendet, welches Herr Kiefer freundlicherweise zur Verfügung stellte. Das Verfahren änderte sich dadurch im wesentlichen nicht. Unterschiede bestehen beispielsweise in der Datenverwaltung (Matrizen, Vektoren) oder der (I)LU-Zerlegung, die hier in Form eines „Black-Box“-Unterprogrammes mit Hilfe der bestehenden Programmbibliothek [39] realisiert wurde.

Die Reduktionsraten in der Spalte „0“ sind unabhängig vom Winkel, da der Operator hier mit dem Laplaceoperator identisch ist. Erfreulicherweise konvergierte das Verfahren in allen Fällen, und zwar nahezu winkelunabhängig. Kritisch zu sehen ist die Abhängigkeit der Reduktionsrate von Problemgröße und Konvektionsstärke. Leider wurde bei dieser Meßreihe der Aufwand weder in der Form der benötigten Zeit noch der Iterationen gemessen, was die Aussagefähigkeit der Ergebnisse relativiert. Insofern ist es wahrscheinlich, daß auch dieses Verfahren, will man eine gänzlich problemgrößenunanbängige Fehlerreduktionsrate erreichen, höhere Kosten als $\mathcal{O}(n)$ mit sich bringt.

Dennoch gibt dieses Ergebnis Grund zur Annahme, daß es eine robuste Variante des Verfahrens geben könnte. Für die Erforschung der hierfür notwendigen Wavelets/Prewavelets beziehungsweise Filterkoeffizienten ist noch einiges an Arbeit zu investieren, was den Rahmen dieser Diplomarbeit übersteigen würde.

Konvektionsrichtung	n	Reduktionsraten bei Konvektionsstärke					
		0	10	100	1000	10^4	10^5
0° (x)	49	0.119	0.082	0.031	0.066	0.072	0.072
	225	0.144	0.104	0.071	0.128	0.143	0.145
	961	0.154	0.127	0.138	0.166	0.205	0.210
	3969	0.165	0.141	0.136	0.194	0.250	0.266
	16129	0.172	0.150	0.152	0.216	0.283	0.309
	65025	0.177	0.157	0.176	0.216	0.302	0.339
22.5°	49	0.119	0.071	0.062	0.106	0.113	0.113
	225	0.144	0.101	0.086	0.153	0.175	0.169
	961	0.154	0.123	0.127	0.201	0.216	0.224
	3969	0.165	0.138	0.144	0.223	0.258	0.265
	16129	0.172	0.148	0.162	0.261	0.342	0.357
	65025	0.177	0.157	0.176	0.304	0.436	0.444
45° (Diagonal)	49	0.119	0.067	0.083	0.124	0.130	0.132
	225	0.144	0.097	0.124	0.200	0.220	0.146
	961	0.154	0.122	0.146	0.262	0.288	0.291
	3969	0.165	0.136	0.162	0.283	0.311	0.308
	16129	0.172	0.147	0.180	0.316	0.354	0.359
	65025	0.177	0.155	0.194	0.358	0.418	0.427
67.5°	49	0.119	0.071	0.062	0.104	0.109	0.114
	225	0.144	0.099	0.088	0.153	0.176	0.172
	961	0.154	0.123	0.123	0.212	0.235	0.239
	3969	0.165	0.136	0.149	0.221	0.262	0.271
	16129	0.172	0.148	0.175	0.265	0.341	0.357
	65025	0.177	0.157	0.190	0.316	0.452	0.460
90° (y)	49	0.119	0.082	0.030	0.066	0.072	0.072
	225	0.144	0.101	0.067	0.128	0.143	0.145
	961	0.154	0.130	0.127	0.166	0.204	0.210
	3969	0.165	0.142	0.133	0.194	0.250	0.266
	16129	0.172	0.152	0.156	0.214	0.284	0.310
	65025	0.177	0.157	0.173	0.214	0.302	0.339
112.5°	49	0.119	0.071	0.065	0.105	0.111	0.117
	225	0.144	0.098	0.095	0.159	0.174	0.182
	961	0.154	0.131	0.134	0.211	0.229	0.236
	3969	0.165	0.144	0.152	0.225	0.268	0.269
	16129	0.172	0.154	0.168	0.270	0.342	0.357
	65025	0.177	0.158	0.181	0.321	0.440	0.453
135°	49	0.119	0.066	0.089	0.123	0.133	0.133
	225	0.144	0.095	0.127	0.207	0.217	0.231
	961	0.154	0.132	0.149	0.265	0.290	0.297
	3969	0.165	0.145	0.169	0.284	0.309	0.310
	16129	0.172	0.154	0.181	0.321	0.354	0.357
	65025	0.177	0.160	0.189	0.351	0.404	0.409
157.5°	49	0.119	0.071	0.063	0.100	0.108	0.111
	225	0.144	0.098	0.096	0.165	0.172	0.185
	961	0.154	0.133	0.112	0.209	0.229	0.235
	3969	0.165	0.148	0.132	0.244	0.279	0.288
	16129	0.172	0.156	0.171	0.276	0.378	0.390
	65025	0.177	0.161	0.187	0.337	0.450	0.459
180°	49	0.119	0.082	0.024	0.055	0.061	0.062
	225	0.144	0.101	0.065	0.117	0.131	0.130
	961	0.154	0.133	0.115	0.153	0.191	0.195
	3969	0.165	0.149	0.134	0.180	0.235	0.250
	16129	0.172	0.157	0.155	0.197	0.267	0.294
	65025	0.177	0.162	0.168	0.218	0.287	0.323

Tabelle 4.31.: Ergebnisse des Verfahrens mit modifizierter Schurkomplementberechnung für den 2D-Konvektionsdiffusionsoperator mit verschiedenen Konvektionsstärken und -richtungen

4.3. Bewertung

Die zunächst bei Tests mit dem Laplaceoperator aufkeimende Hoffnung, eines oder gar mehrere der eingesetzten Lösungsverfahren könnte robust genug sein, um auch den Konvektionsdiffusionsoperator unabhängig von der Konvektionsrichtung und -stärke in linearer Zeit zu lösen, erfüllte sich nicht. Stattdessen kam es bei Verfahren 1 und 2 für Konvektion in x- und in y-Richtung oft sogar zur Divergenz.

Bereits beim ein- und auch beim zweidimensionalen Laplaceoperator zeigte sich aufgrund des den eingesetzten Waveletfiltern überlegenen Verhaltens der Prewavelets, daß man mit problemangepaßten Filtern arbeiten sollte. Dieses Ergebnis wurde vom Verhalten beider zunächst benutzter Verfahren auch für den Konvektionsdiffusionsoperator bestätigt. Im eindimensionalen Fall zeigte sich ebenso wie bei Diagonalkonvektion im zweidimensionalen Fall, daß man mit den dann problemangepaßten Haarwavelets die besten Resultate erreicht. Das schlechte Abschneiden bei Konvektion in x- und in y-Richtung im zweidimensionalen Fall läßt sich darauf zurückführen, daß hier weder die Wavelets noch die Prewavelets problemangepaßt sind. Um für achsenparallele Konvektion einen problemangepaßten Filter zu bekommen, müßte man in Konvektionsrichtung Wavelets und in die andere Richtung Prewavelets verwenden, wie es in dem zuletzt eingesetzten modifizierten Verfahren implizit geschah, wodurch sich die nahezu winkelunabhängigen Fehlerreduktionsraten erklären lassen.

Von den drei Verfahrensvarianten

1. Schurkomplementapproximation mit Banderweiterung,
2. Approximation des Schurkomplementoperators durch den Grobgitteroperator (ohne Schurkomplementkorrektur) und
3. Mischung aus den beiden erstgenannten

zeigte zwar die zweite stets aufgrund des damit verbundenen geringen Aufwandes das beste Laufzeitverhalten und damit auch einen nahezu lienaren Aufwand, dafür erwies sie sich als extrem anfällig gegenüber nicht problemangepaßten Waveletfiltern. Etwas robuster präsentierte sich die Variante mit Banderweiterung. Mit breiter werdendem Band nähert sie sich mehr und mehr einem exakten Löser der Komplexität $\mathcal{O}(n^3)$. Obwohl für eine zufriedenstellende Fehlerreduktionsrate oft eine gewisse Mindestbandbreite erforderlich ist, erhält man in besonders günstigen Fällen (von problemangepaßten Waveletfiltern) auch hier eine nahezu lineare Komplexität. Ansonsten liegt der Aufwand für diese Variante meist bei immerhin noch recht passablen $\mathcal{O}\left(n(\log_2(n^{\frac{1}{d}}))^{2d}\right)$. Auch die dritte Variante ist ein effizientes Iterationsverfahren. Bei der Approximation des Schurkomplementoperators des jeweils vorangegangenen Levels ließe sich auch hier eine Banderweiterung einsetzen, sodaß gegebenenfalls ein wichtiger Parameter zur Steuerung des Konvergenzverhaltens zur Verfügung stünde. Insofern würden hier die Vorteile der beiden anderen Varianten, nämlich ein geringer Aufwand bei Verfahren 2 und eine frei bestimmbare Banderweiterung wie bei Verfahren 1, miteinander verknüpft. Leider war es aus zeitlichen Gründen nicht möglich, eine entsprechende Versuchsreihe durchzuführen.

Zusammenfassend läßt sich festhalten, daß man auf jeden Fall problemangepaßte Waveletfilter verwenden sollte. Außerdem ist die Benutzung einer Banderweiterung von Vorteil, um die Konvergenzrate des Verfahrens steuern zu können, auch wenn sich dadurch der Aufwand zur Lösung des gegebenen Gleichungssystems von $\mathcal{O}(n)$ auf $\mathcal{O}\left(n(\log_2(n^{\frac{1}{d}}))^{2d}\right)$ erhöht.

5. Zusammenfassung

In mathematischen Anwendungen kommt es häufig zu linearen Gleichungssystemen mit einer hohen Anzahl n von Unbekannten, die in möglichst kurzer Zeit zu lösen sind. Gines, Beylkin und Dunn schlagen in [25] eine Methode vor, mit der dies zumindest für bestimmte Operatoren in $\mathcal{O}(n)$ realisierbar sein soll. Sie betrachten dabei in erster Linie Integraloperatoren, aber auch für gewisse Klassen von Differentialoperatoren soll ihr Verfahren anwendbar sein. Voraussetzung dafür ist ein starkes (polynomiales) Abfallverhalten der Matrixeinträge der zugehörigen Operatormatrix in der Waveletbasis mit zunehmender Entfernung von der Diagonalen.

Wesentlich für diese Methode ist es, sämtliche Rechnungen des Lösungsprozesses auf ein Band um die Diagonale der Operatormatrix herum zu beschränken. Daraus ergeben sich zwei Fragen: einmal, wie breit das in der Rechnung berücksichtigte Band sein muß, um eine Lösung mit hinreichender Genauigkeit erzielen zu können, und zweitens, mit welchen (Pre-)Wavelets man für den jeweiligen Operator einen schnellen Löser bekommt. Die starke Abhängigkeit der Ergebnisse vom verwendeten Operator bei starren Waveletfiltern führt zur Verwendung problemabhängiger (Pre-)Wavelets sowie zu leichten Modifikationen am Verfahren selbst.

Das von Gines, Beylkin und Dunn vorgeschlagene Verfahren läßt sich in mehreren Varianten durchführen, von denen zwei in dieser Arbeit näher untersucht wurden. Die eine basiert auf der Annahme, daß die Korrektur des Schurkomplementoperators nicht nötig ist, da der Grobgitteroperator auf jedem Level die wesentlichen Anteile des Schurkomplementoperators beinhaltet. Das in Abschnitt 2.4 geschilderte starke Abfallverhalten der Einträge des Korrekturterms mit zunehmendem Abstand von der Diagonalen unter bestimmten Voraussetzungen unterstützt diese Annahme. Auf diese Weise verringert sich der Gesamtaufwand gegenüber der Variante mit Bandapproximation des Schurkomplementes deutlich, das Verfahren ist auch im Prinzip gitterweitenunabhängig konvergent, fraglich ist jedoch, ob die Reduktionsrate zufriedenstellend ist. Die andere Variante geht von der Überlegung aus, daß man, anstatt die Berechnungen strikt auf ein Band der ursprünglichen Breite zu beschränken, durch eine Verbreiterung desselben eine größere Genauigkeit gewinnen kann. Die entstehenden Mehrkosten sind gegen das bessere Ergebnis abzuwägen.

Weitere Varianten sind denkbar, zum Beispiel Mischformen, bei denen das Schurkomplement bei der Zerlegung des Operators zunächst nicht berechnet, beim Lösen aber levelweise approximiert wird. Die Tiefe dieser Approximation läßt sich variieren. Ferner kann man verschiedene Banderweiterungsmethoden verwenden. In dieser Arbeit wurde eine variable Bandweite gewählt, indem auf jedem Level um eine konstante Anzahl von Spalten erweitert wurde. Genauso denkbar ist eine variable Erweiterung um einen konstanten Faktor an Spalten auf jedem Level. Eine weitere Methode ist die, eine konstante Bandweite zu wählen und diese über sämtliche Level beizubehalten. Ebenso möglich wäre es, auf jedem Level um eine variable Anzahl von Spalten zu erweitern, jedoch erscheint diese Variante ohne eine sinnvole Strategie etwas unplausibel.

Die oben genannten Verfahrensvarianten wurden am Laplaceoperator und am Konvektionsdiffusionsoperator sowohl in einer als auch in zwei Dimensionen getestet. Für letzteren wurden dabei verschiedene Konvektionsstärken und in zwei Dimensionen auch verschiedene Konvektionsrichtungen in Betracht gezogen. Für den Laplaceoperator erwies sich das Banderweiterungsverfahren als ein zuverlässiger Löser der Komplexität $\mathcal{O}\left(n(\log_2(n^{\frac{1}{d}}))^{2d}\right)$. Das gleiche Lösungsverhalten konnte für den eindimensionalen Konvektionsdiffusionsoperator und in zwei Dimensionen bei Konvektion in Diagonalrichtung beobachtet werden. Sowohl bei Konvektion in x-Richtung als auch bei Konvektion in y-Richtung zeigte sich die Schwäche des Verfahrens bei nicht problemangepaßten (Pre-)Wavelets.

Die Variante ohne Schurkomplementkorrektur brachte ebenfalls für den Laplaceoperator gute Ergebnisse, mit Prewavelets konnte hier auch in zwei Dimensionen nahezu ein $\mathcal{O}(n)$-Lösungsverhalten beobachtet werden. Für den Konvektionsdiffusionsoperator konnte das Verfahren selbst bei Konvektion in Diagonalrichtung keine zufriedenstellenden Ergebnisse liefern, was für Standardprewavelets ebenfalls daran liegt, daß sie für Probleme mit großer Konvektionsstärke nicht geeignet sind.

Obwohl für den Konvektionsdiffusionsoperator nicht immer Konvergenz erzielt werden konnte, zeigen die konvergenten Fälle der Banderweiterungsvariante, daß dieses Verfahren für gut konditionierte Probleme ein $\mathcal{O}\left(n(\log_2(n^{\frac{1}{d}}))^{2d}\right)$-Löser ist. Zwar ist dieser Aufwand höher als der bei den für den Laplaceoperator gemessenen Ergebnissen ohne Schurkomplementkorrektur mit Prewavelets, welcher nahezu bei $\mathcal{O}(n)$ lag, aber der logarithmische Term läßt sich asymptotisch fast vernachlässigen. Das Verfahren ohne Schurkomplementkorrektur mit Wavelets erwies sich als den anderen beiden Varianten unterlegen.

Die mit operatorabhängigen Prewavelets und einer Mischform aus Schurkomplementapproximation und Verzicht auf die -korrektur gemachten Experimente zeigen, daß sich das Verfahren grundsätzlich auch für schlecht konditionierte Probleme einsetzen läßt, wenn man problemangepaßte (Pre-)Wavelets verwendet. Die Fülle der sich bietenden Alternativen für sinnvolle Modifikationen des Verfahrens begründet die Hoffnung, daß es – wenn auch nach längerer Forschungstätigkeit – möglich sein dürfte, einen zuverlässigen Löser für weite Klassen von Operatoren zu konstruieren, der in $\mathcal{O}(n)$ oder zumindest in $\mathcal{O}\left(n(\log_2(n^{\frac{1}{d}}))^{2d}\right)$ funktioniert.

A. Funktionsweise von Wavelettransformationen

In diesem Kapitel wird erklärt, wie eigentlich eine Wavelettransformation funktioniert. Es wird dabei in erster Linie auf die direkte algebraische Transformation eingegangen. In der Literatur findet man zu diesem Thema nur wenig Informationen. Zum Nachschlagen eignen sich [47] und [52], wobei diese beiden keine einheitliche Notation verwenden.

Zunächst wird in Abschnitt A.1 die Waveletzerlegung eines Vektors beschrieben, bevor in den Abschnitten A.2 und A.3 die ein- und die zweidimensionale Zerlegung von Matrizen behandelt wird.

A.1. Waveletzerlegung eines Vektors

Gegeben sei ein reeller Vektor v mit acht Elementen. Wichtig ist dabei, daß die Länge des Vektors eine Potenz von zwei sein sollte, da bei der Waveletzerlegung jeweils zwei Vektoren von der halben Länge des Ursprungsvektors entstehen. Der eine von beiden sollte sich abermals zerlegen lassen. Verwendet man Prewavelets, so ist die ideale Länge $2^k - 1$.

Außerdem benötigt man Wavelets, im folgenden werden Daubechies4-Wavelets als Beispiel verwendet. Diese werden repräsentiert durch einen Vektor g, den sogenannten Filterkoeffizienten. Für Daubechies4-Wavelets gilt:

$$\begin{aligned} g[1] &= \frac{1+\sqrt{3}}{4\sqrt{2}} \\ g[2] &= \frac{3+\sqrt{3}}{4\sqrt{2}} \\ g[3] &= \frac{3-\sqrt{3}}{4\sqrt{2}} \\ g[4] &= \frac{1-\sqrt{3}}{4\sqrt{2}}. \end{aligned}$$

Damit lassen sich die Grobanteile von v herausfiltern. Die Feinanteile des Vektors werden mit

$$\begin{aligned} h[1] &= g[4] \\ h[2] &= -g[3] \\ h[3] &= g[2] \\ h[4] &= -g[1] \end{aligned}$$

gefiltert. Der Filterungsprozeß funktioniert so, daß g zunächst mit den ersten vier Elementen von v multipliziert wird, dann mit den Elementen drei bis sechs, dann fünf bis acht und schließlich setzt er bei Element sieben auf. Da v nur acht Elemente hat, muß nun periodisiert werden, also sind die beiden letzten Elemente des Filters g mit den beiden ersten Elementen von v zu multiplizieren. Diese vier Filtervorgänge, bei denen der Aufsatzpunkt von g auf dem Vektor v jeweils um 2 erhöht wird, ergeben den Grobanteil von v, der ein Vektor s_1 mit vier Einträgen ist. Anschließend ermittelt man auf die gleiche Weise die Feinanteile d_1 von v mittels des Feinfilters h. Dieser Prozeß, Wavelettransformation genannt, läßt sich nun an s_1 wiederholen, man erhält zwei Vektoren s_2 und d_2 der Länge zwei. Filtert man s_2 abermals, erhält man nur noch zwei Zahlen s_3 und d_3, die sich dann nicht weiter zerlegen lassen. Dieses Vorgehen bezeichnet man auch als Pyramidenalgorithmus, da stets die entstandenen Grobanteile s_i weiter zerlegt werden, bis es nicht mehr geht, weil der Grobanteil schließlich zu einer Zahl geworden ist.

Man kann die Filterkoeffizienten auch in einer Matrix aufschreiben, sodaß sich der Filterungsprozeß als Matrix-Vektor-Multiplikation darstellen läßt. Betrachtet man v als Spaltenvektor, so würde man in die erste Zeile der Matrix, auch Waveletoperator W genannt, zunächst die Elemente des Feinfilters h schreiben und die restlichen vier Einträge mit Nullen auffüllen. Die zweite Zeile begänne dann mit zwei Nullen, dann käme h, dann noch zwei Nullen, usw. Die unteren vier Zeilen wären genauso aufgebaut wie die oberen vier, nur daß hier der Grobfilter g einzutragen wäre. Die so entstandene Matrix ist natürlich nur für Vektoren der Länge acht verwendbar, für längere oder kürzere Vektoren ist sie entsprechend aufzubauen.

Ebenso kann man auch andere Wavelets verwenden. Besonders einfach sind dabei Haarwavelets, die mit den Daubechies2-Wavelets identisch sind. Aufgrund ihrer Filterlänge von zwei eignen sie sich zwar für Rechnungsbeschleunigungen, haben dafür aber auch nur ein verschwindendes Moment. Als Beispiel sind sie ungeeignet, da sie keine Periodisierung des (zu zerlegenden) Vektors erfordern.

A.2. Eindimensionale Waveletzerlegung einer Matrix

Wie ein Operator T_0 zu transformieren ist, wurde bereits in den Abschnitten 2.2.1 und 2.2.2 beschrieben. Die Berechnung von

$$\mathcal{W}_1^{0^T} T_0 \mathcal{W}_1^0$$

wird wie folgt aufgeteilt: zuerst führt man auf T_0 zeilenweise eine Wavelettransformation durch und anschließend auf dem Ergebnis spaltenweise, denn

$$\mathcal{W}_1^{0T} T_0 = (T_0^T \mathcal{W}_1^0)^T,$$

und die Zeilen von T_0 sind die Spalten von T_0^T.

A.2.1. Nichtstandardform

Bei der Nichtstandardform wird auf jedem Level so verfahren wie oben beschrieben, und von den entstehenden Teilmatrizen A_i, B_i, C_i und T_i wird der Prozeß iterativ auf T_i fortgeführt.

In der Praxis werden die Zeilen des ursprünglichen Operators T_0 in einen Vektor geschrieben. Dieser wird dann zerlegt und die beiden Ergebnisse (Grob- und Feinanteile) als Zeilen in Zwischenspeichermatrizen D und S abgelegt. Das gleiche passiert anschließend mit den Spalten der beiden Zwischenspeicher, nur daß die Ergebnisse jetzt direkt als Spalten in die zugehörigen Matrizen A_1 und C_1 (D) beziehungsweise B_1 und T_1 (S) einsortiert werden.

A.2.2. Standardform

Die Standardform läßt sich durch weitere Zerlegung der Matrizen B_i und C_i aus der Nichtstandardform gewinnen.

A.3. Zweidimensionale Waveletzerlegung einer Matrix

A.3.1. Nichtstandardform

Eine recht ausführliche Beschreibung der zweidimensionalen Nichtstandardform liefert Gines in [24].

Die zweidimensionale Wavelettransformation läßt sich wie folgt realisieren: Der Ausgangsoperator Tt_0 besteht – abhängig von seiner Größe – aus einer Anzahl von Blöcken, von denen jeder einzelne zunächst eindimensional zu transformieren ist. Dies geschieht über zwei Zwischenspeicher. Der erste wird dazu verwendet, den aktuellen Block darin abzuspeichern, um ihn in die eindimensionale Wavelettransformation eingeben zu können, der andere dient dazu, die Ergebnisse bereits umsortiert aufzunehmen. Auf diese Weise läßt sich die Vertauschung von Block- und Innerblockkoordinaten vornehmen, ohne daß dafür eine eigene Schleife notwendig wäre.

Der zweite Transformationsschritt verläuft bis auf das Einsortieren der Ergebnisse analog. Die mit jeder Zerlegung anfallenden Teile A, B, C und T müssen zusammensortiert werden.

Für jeden zerlegten Block erhält man je eines dieser Teile. Man sortiert sämtliche A-Teile in das linke obere Viertel von Tt_0 ein, die B-Teile oben rechts, die C-Teile unten links und die T-Teile unten rechts.

In einem abschließendenen Schritt werden diese Abschnitte auf die Matrizen Aa_1 bis Tt_1 aufgeteilt.

A.3.2. Standardform

Die zweidimensionale Zerlegung in die Standardform ist ähnlich zu gestalten wie die Berechnung der zweidimensionalen Nichtstandardform. Auch hier bekommt man eine verschachtelte Struktur. Allerdings ist es im Gegensatz zu den eindimensionalen Varianten offenbar nicht möglich, aus der zweidimensionalen Nichtstandardform durch ein paar zusätzliche Rechnungen die zweidimensionale Standardform zu erhalten.

Abbildungsverzeichnis

Algorithmenverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] R. E. Alcouffe, A. Brandt, J. E. Dendy, J. W. Painter: The Multi-Grid Method for the Diffusion Equation with Strongly Discontinuous Coefficients, *SIAM J. Sci. Comput.*, 2, 1981, 430-454

[2] B. K. Alpert: Wavelets and Other Bases for Fast Numerical Linear Algebra, in: C. K. Chui:*Wavelets - A Tutorial in Theory and Applications*, Academic Press, New York 1992, 181-216

[3] O. Axelsson: *Iterative Solution Methods*, Cambridge University Press, Cambridge 1994

[4] R. E. Bank: Hierarchical Bases and the Finite Element Method, *Acta Numerica*, 1996, ed: A. Iserles, Cambridge University Press, 1996, 1-43

[5] R. E. Bank, T. F. Dupont: Analysis of a Two-Level Scheme for Solving Finite Element Equations, Report CNA-159, Center for Numerical Analysis, University of Texas at Austin 1980

[6] R. E. Bank, T. F. Dupont, H. Yserentant: The Hierarchical Basis Multigrid Method, *Numer. Math.*, 52, 1988, 427-458

[7] R. E. Bank, J. Xu: The Hierarchical Basis Multigrid Method and Incomplete LU Decomposition, Seventh International Symposium on Domain Decomposition Methods for Partial Differential Equations, eds: D. Keyes, J. Xu, AMS, Providence, Rhode Island, 1994, 163-173

[8] G. Beylkin: Wavelets and Fast Numerical Algorithms, in: *Proceedings of Symposia in Applied Mathematics*, 47, 1993, 89-117

[9] G. Beylkin: On the Representation of Operators in Bases of Compactly Supported Wavelets, *SIAM J. on Numerical Analysis*, 6, 1992, 1716-1740

[10] G. Beylkin, R. Coifman and V. Rohklin: Fast Wavelet Transforms and Numerical Algorithms I., *Comm. Pure and Appl. Math.*, 44, 1991, 141-183

[11] J. H. Bramble, J. E. Pasciak, J. Xu: Parallel Multilevel Preconditioners, *Math. Comp.*, 55, 1990, 1-22

[12] A. Brandt: Multi-Level Adaptive Solutions to Boundary-Value Problems, *Math. Comp.*, 31, 1977, 333-390

[13] C. Brezinski: *Projection Methods for Systems of Equations*, Elsevier, Amsterdam 1997

[14] T. F. Chan, W. P. Tang, W. L. Wan: Wavelet Sparse Approximate Inverse Preconditioners, BIT 37, 1997, 644-660

[15] C. K. Chui: Wavelets and Spline Interpolation, Technical Report Number CAT-227, Texas A&M University 1990

[16] J. W. Cooley, J. W. Tukey: An Algorithm for the Machine Calculation of Complex Fourier Series, *Math. Comp.* 19, 1965, 297-301

[17] N. A. Coult: A Multiresolution Strategy for Homogenization of Partial Differential Equations, Thesis, University of Colorado at Boulder 1997

[18] I. Daubechies: *Ten Lectures on Wavelets*, CBMS-NSF Reg. Conf. Series in Applied Math. 61, SIAM, Philadelphia 1992

[19] W. Dahmen: Wavelet and Multiscale Methods for Operator Equations, *Acta Numerica,* 1997, 55-228

[20] W. Dahmen, A. Kunoth: Multilevel Preconditioning, *Numer. Math.*, 63, 1992, 315-344

[21] W. Dahmen, S. Müller, T. Schlinkmann: Multigrid and Multiscale Decompositions, in: *Notes on Numerical Fluid Mechanics,* Vol. 62; *Large Scale Scientific Computations of Engineering and Environmental Problems,* eds: M. Griebel, O. P. Iliev, S. D. Margenov, P. S. Vassilevski, Vieweg-Verlag, Braunschweig 1998, 18-41

[22] J. E. Dendy: Black Box Multigrid for Nonsymmetric Problems, *Appl. Math. Comput.*, 13, 1983, 261-283

[23] I. S. Duff, A. M. Erisman, I. K. Reid: *Direct Methods for Sparse Matrices*, Clarendon Press, Oxford 1989

[24] D. L. Gines: Fast Electromagnetic Simulations Using Wavelets, Thesis, University of Colorado at Boulder 1997

[25] D. L. Gines, G. Beylkin, J. Dunn: LU Factorization of Non-Standard Forms and Direct Multiresolution Solvers, Technical Report 278, Program in Applied Mathematics, University of Colorado at Boulder 1996

[26] A. Greenbaum: *Iterative Methods for Solving Linear Systems*, SIAM, Philadelphia 1997

[27] M. Griebel: Zur Lösung von Finite-Differenzen- und Finite-Element-Gleichungen mittels der Hierarchischen-Tansformations-Mehrgitter-Methode, SFB-Bericht Nr. 342/4/90 A, Technische Universität München 1990

[28] M. Griebel: *Multilevelmethoden als Iterationsverfahren über Erzeugendensystemen*, B. G. Teubner Verlag, Stuttgart 1994

[29] M. J. Grote, T. Huckle: Parallel Preconditioning with Sparse Approximate Inverses, *SIAM J. Sci. Comput.*, 18, 1997, 838-853

[30] A. Haar: Zur Theorie der orthogonalen Funktionensysteme, *Mathematische Annalen*, 1910, 331-371

[31] W. Hackbusch: *Iterative Lösung großer schwachbesetzter Gleichungssysteme*, B. G. Teubner Verlag, Stuttgart 1993

[32] W. Hackbusch: *Multigrid Methods and Applications*, Springer-Verlag, Berlin 1985

[33] W. Hackbusch: Ein iteratives Verfahren zur schnellen Auflösung elliptischer Randwertprobleme, Report 76-12, Universität zu Köln, 1976

[34] W. Hackbusch: *Integralgleichungen – Theorie und Numerik*, B. G. Teubner Verlag, Stuttgart 1989

[35] F. Kleinert: Biorthogonal Wavelets for Fast Matrix Computations, *Appl. Comput. Harmon. Anal.* 1, 1994, 147-156

[36] S. Mallat: Multiresolution Approximation and Wavelets, Trans. Amer. Math. Soc., 315, 1989, 69-88

[37] Y. Meyer: *Ondelettes et Opérateurs*, Hermann, Paris 1990

[38] P. Oswald: *Multilevel Finite Element Approximation*, B. G. Teubner Verlag, Stuttgart 1994

[39] S. Balay, W. D. Gropp, L.C. McInnes, B. F. Smith: PETSc,
Homepage: http://www.mcs.anl.gov/petsc, 2000
Manual: PETSc 2.0 Users Manual, ANL-95/11-Revision 2.0.28, Argonne National Laboratory, 2000

[40] A. Rieder: Wavelet Multilevel Solvers for Linear Ill-Posed Problems Stabilized by Tikhonov Regularization, in: *Multiscale Wavelet Methods for Partial Diferential Equations*, Wavelet Anal. Appl., 6, Academic Press, San Diego, CA, 1997, 347-380

[41] A. Rieder, R. O. Wells Jr., X. Zhou: A Wavelet Approach to Robust Multilevel Solvers for Anisotropic Elliptic Problems, *Appl. Comput. Harmon. Anal.* 1, 1994, 355-367

[42] A. Rieder, R. Glowinski, R. O. Wells Jr., X. Zhou: A Wavelet Multigrid Preconditioner for Dirichlet Boundary Value Problems in General Domains, *RAIRO Modél. Math. Anal. Numér.* 30, 1996, 711-729

[43] Y. Saad: *Iterative Methods for Sparse Linear Systems*, 2nd edition, 2000, siehe http://www-users.cs.umn.edu/ saad/books.html

[44] R. Schneider: *Multiskalen- und Wavelet-Matrixkompression: Analysisbasierte Methoden zur effizienten Lösung großer vollbesetzter Gleichungssysteme*, Advances in Numerical Mathematics, B. G. Teubner Verlag, Stuttgart 1998

[45] G. Starke: Multilevel Minimal Residual Methods for Nonsymmetric Elliptic Problems, *Numerical Linear Algebra with Applications*, 3, 1996, 351-367

[46] G. Starke: Field-of-Values Analysis of Precondition Iterative Methods for Non-Symmetric Elliptic Problems, *Numer. Math.*, 78, 1997, 103-117

[47] G. Strang: Wavelets and Dilation Equations: A Brief Introduction, *SIAM Rev.*, 31, 1989, 613-627

[48] P. Tchamitchian: Wavelets and Differential Operators, in: *Proceedings of Symposia in Applied Mathematics*, 47, 1993, 77-88

[49] L.-N. Trefethen: Pseudospectra of Linear Operators, *SIAM Rev.*, 39, 1997, 383-406

[50] P. S. Vassilevski, J. Wang: Stabilizing the Hierarchical Basis by Approximate Wavelets, I: Theory, *Numer. Linear Algebra Appl.*, 4, 1997, 103-126

[51] P. S. Vassilevski, J. Wang: Stabilizing the Hierarchical Basis by Approximate Wavelets, II: Implementation and Numerical Results, *SIAM Rev.*, 39, 1997, 18-53

[52] B. Vidaković, P. Müller: Wavelets For Kids, 1994, erhältlich über ftp://ftp.isds.duke.edu/pub/WorkingPapers/94-13-1.ps

[53] R. Weiss: *Parameter-Free Iterative Linear Solvers*, Akademie Verlag, Berlin 1996

[54] J. Xu: Iterative Methods by Space Decomposition and Subspace Correction, *SIAM Rev.*, 34, 1992, 581-613

[55] D. M. Young: *Iterative Solution of Large Linear Systems*, Academic Press, New York and London 1971