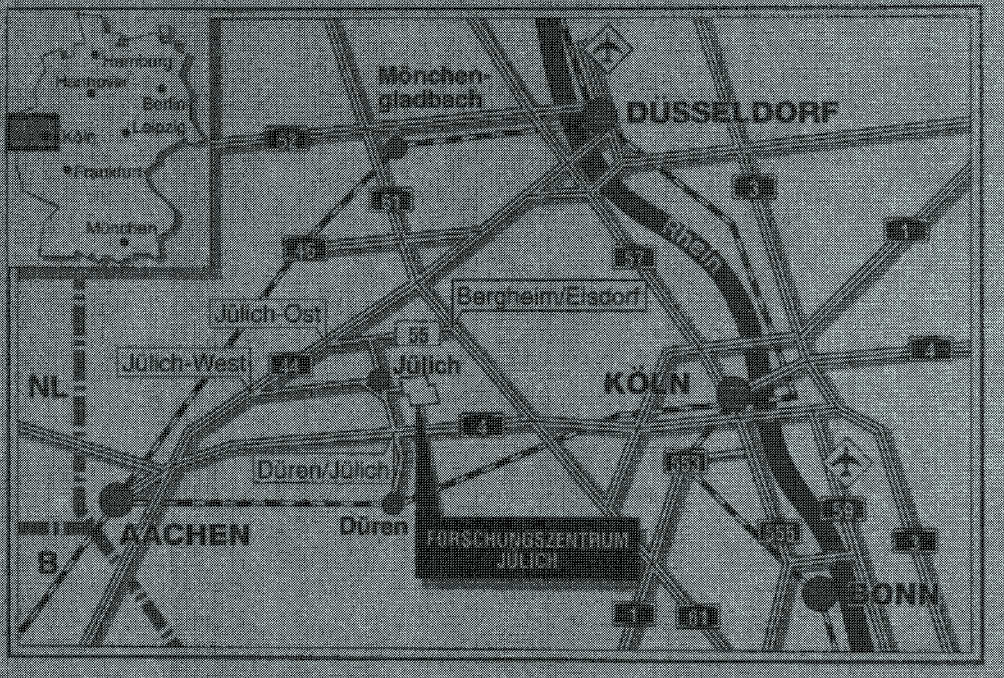
Forschungszentrum Jülich

Zentralinstitut für Angewandte Mathematik

Kommunikationsminimale Algorithmen zur Lösung großer dünnbesetzter linearer Gleichungssysteme auf massiv-parallelen Systemen

Manfred Sauren

ISSN 0944-2952



Berichte des Forschungszentrums Jülich ; 3396
ISSN 0944-2952
Zentralinstitut für Angewandte Mathematik Jül-3396
D 82 (Diss. RWTH Aachen)

Zu beziehen durch: Forschungszentrum Jülich GmbH · Zentralbibliothek
D-52425 Jülich · Bundesrepublik Deutschland
☎ 02461/61-6102 · Telefax: 02461/61-6103 · e-mail: zb-publikation@fz-juelich.de

Kurzfassung

Die numerische Lösung partieller Differentialgleichungen kann auf die Lösung dünnbesetzter linearer Gleichungssysteme zurückgeführt werden, deren Größe den Einsatz von massiv-parallelen Systemen erfordert. Für diese Problemstellung sind iterative Verfahren geeignet, in denen die Koeffizientenmatrix ausschließlich zur Berechnung von Matrix-Vektor-Produkten verwendet wird. In dieser Verfahrensklasse werden Krylov–Teilraumverfahren entworfen, die auf einem Parallelrechner mit verteiltem Speicher nur einen globalen Synchronisationspunkt in jedem Iterationsschritt besitzen. In diesem Zusammenhang wird eine Variante des Verfahrens der quasi-minimalen Residuen entwickelt, dessen Minimierungsproblem darüber hinaus in allen l_p-Normen rekursiv gelöst werden kann. Auf diese Weise wird eine neue Klasse der quasi-l_p-minimalen Verfahren eingeführt.

Bei der Implementierung iterativer Verfahren auf parallelen Systemen mit verteiltem Speicher ist die Parallelisierung des Matrix-Vektor-Produktes die aufwendigste Operation. Hierzu werden sowohl die Einträge der Matrix als auch die Komponenten der Vektoren auf die einzelnen Prozessoren verteilt. Der Datenaustausch, der bei der parallelen Berechnung dieses Produktes erforderlich ist, induziert ein für das iterative Verfahren charakteristisches Kommunikationsschema, das in jedem Iterationsschritt des Verfahrens benutzt wird. Es werden Techniken zur Beschleunigung iterativer Verfahren entwickelt, bei denen keine zusätzlichen Kommunikationskosten entstehen sollen. Dazu wird ein Modell beschrieben, in dem das lineare Gleichungssystem um die Variablen erweitert wird, die für den erforderlichen Datenaustausch auf einem Parallelrechner mit verteiltem Speicher benötigt werden. So wird es möglich, die zusätzlichen Variablen zur Konvergenzbeschleunigung des iterativen Verfahrens einzusetzen und gleichzeitig das Kommunikationsschema zu berücksichtigen, wodurch weitere Kommunikationskosten vermieden werden. Auf diese Weise werden zunächst bei Gebietszerlegungsmethoden Beschleunigungstechniken beschrieben, die anschließend auf iterative Verfahren zur Lösung linearer Gleichungssysteme übertragen werden. Die daraus resultierenden Methoden können als Vorkonditionierer in den dargestellten Krylov–Teilraumverfahren eingesetzt werden. Das Verhalten der entwickelten Algorithmen wird abschließend an Modellproblemen auf dem massiv-parallelen Rechner Intel Paragon XP/S 10 demonstriert.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die systematische Nutzung von Rechnern zur Lösung komplexer Probleme in Naturwissenschaft und Technik durch mathematische Modellierung und Computersimulation entwickelt sich neben Theorie und Experiment zu einer dritten eigenständigen wissenschaftlichen Disziplin. Die zunehmende Verfügbarkeit leistungsfähiger Systeme eröffnet so bedeutende Anwendungsperspektiven. Insbesondere ergeben sich hierdurch in den Lebenswissenschaften, beispielsweise in der Molekularchemie und der Medizin völlig neue Möglichkeiten. Auch mittelfristige Wettervorhersagen, komplizierte Strömungsvorgänge oder aufwendige Crash-Simulationen sind auf diese Weise durchführbar.

Die meisten dieser Probleme werden dabei durch partielle Differentialgleichungen modelliert. Verbrennungsvorgänge werden beispielsweise durch sehr große Systeme dieser Gleichungen beschrieben; durch gezielte Optimierung der Brennkammergeometrie könnte so die Schadstoffbildung reduziert werden. Die Simulation von Crash-Tests ist heute in der Automobilbranche eine Standardanwendung bei der Fahrzeugentwicklung. Auch hier wird die Simulation mit partiellen Differentialgleichungen modelliert. Mischvorgänge beherrschen in der chemischen Industrie die Herstellung vieler Produkte. Produktqualität, Herstellungszeiten und -kosten werden deshalb stark durch effiziente und kostengünstige Mischverfahren beeinflußt. Komplexe Mischvorgänge können so mit der Lösung von partiellen Differentialgleichungen simuliert werden. Selbst die Gleiteigenschaften von Fallschirmen werden durch die Modellierung mit solchen Gleichungen studiert.

Neben der mathematischen Modellierung bildet die numerische Lösung der Gleichungen eine zentrale Rolle in den Ingenieur- und Naturwissenschaften [59]. Bei realistischen und großen Anwendungen ist dies äußerst rechen- und speicherintensiv, so daß hierzu massiv-parallele Systeme eingesetzt werden müssen. Bei Systemen mit verteiltem Speicher wird aber die Ausführungszeit eines Programmes nicht allein durch die Rechenzeit bestimmt, sondern auch durch Kommunikationszeiten, die durch den Austausch von Daten zwischen den Prozessoren entstehen. Man sucht daher schnelle Algorithmen, die niedrige Kommunikationskosten auf Parallelrechnern besitzen.

Zur Lösung einer partiellen Differentialgleichung gibt es verschiedene Ansätze,

beispielsweise den der Diskretisierung der Gleichung auf ihrem Definitionsgebiet mit Hilfe eines Differenzenverfahrens, der Methode der finiten Elemente oder der Methode der finiten Volumen. Die Behandlung des so gewonnenen diskreten Problems kann dann auf die Lösung eines oder mehrerer linearer Gleichungssysteme mit dünnbesetzter Koeffizientenmatrix reduziert werden. Dabei bedeutet dünnbesetzt, daß der überwiegende Teil der Matrixelemente Null ist und nur wenige Einträge jeder Zeile von Null verschieden sind. Für eine möglichst genaue Lösung werden dabei sehr große Gleichungssysteme benötigt, so daß die Größe und der Besetzungsgrad der Matrix bei den verwendeten Speichertechniken und Lösungsmethoden berücksichtigt werden müssen.

Grundsätzlich kann man zwei verschiedene Arten von Verfahren zur Lösung linearer Gleichungssysteme unterscheiden, nämlich direkte Verfahren und iterative Verfahren. Bei direkten Verfahren, wie zum Beispiel der Gauß–Elimination, entstehen im allgemeinen in jedem Schritt der Elimination zusätzliche Nicht-Null-Elemente, die den niedrigen Besetzungsgrad der Matrix zerstören und so den Speicherbedarf um ein Vielfaches erhöhen können. Für große Gleichungssysteme sind daher iterative Verfahren die bessere Wahl. Solche Methoden starten mit einer Anfangsnäherung an die Lösung und berechnen in jedem Iterationsschritt eine neue Näherung (*Iterierte*), die schließlich gegen die Lösung des Gleichungssystems konvergieren soll. Der Vorteil dieser Methoden besteht darin, daß sie die Matrix ausschließlich zur Berechnung von Matrix-Vektor-Produkten benutzen, so daß oft sehr effiziente Implementierungen bezüglich des Speicherplatzbedarfes der Matrix und der Berechnungszeiten für Matrix-Vektor-Produkte möglich sind.

Die iterativen Verfahren lassen sich nun weiter unterteilen in die Gruppe der klassischen Verfahren, wie zum Beispiel das Gauß–Seidel– oder Jacobi–Verfahren, und die Gruppe der Krylov–Teilraumverfahren. Die klassischen Verfahren basieren auf einer Zerlegung der Matrix und haben den Nachteil, daß sie nur für spezielle Matrizen entsprechend schnell konvergieren; sie sind daher für viele reale Anwendungen nicht geeignet. Bei den Krylov–Teilraumverfahren hingegen wird iterativ ein Teilraum des Lösungsraumes konstruiert, in welchem die Lösung des Gleichungssystems approximiert wird. Ein bekanntes Beispiel für ein solches Verfahren ist das Verfahren der konjugierten Gradienten (CG) [41], das aber nur für Gleichunssysteme mit symmetrisch positiv definiter Koeffizientenmatrix benutzt werden kann. Da viele Matrizen praxisrelevanter Probleme diese Eigenschaft nicht erfüllen, werden im folgenden Verfahren für allgemeine Matrizen untersucht.

In der Klasse der Krylov–Teilraummethoden unterscheidet man Verfahren, die lange oder kurze Rekursionen zur Berechnung der Iterierten benötigen. Dabei verwenden Verfahren mit langen Rekursionen alle zuvor erzeugten Vektoren des Teilraumes, Verfahren mit kurzen Rekursionen dagegen nur einige wenige. Aus Speicherplatzgründen sind deshalb letztere vorzuziehen. Diese haben aber den Nachteil, daß der Approximationsfehler im erzeugten Teilraum nicht minimiert werden kann [26]. Folglich ist ein monotoner Konvergenzverlauf nicht garantiert. Ein Beispiel für eine Methode mit kurzen Rekursionen ist das Verfahren der bikonjugierten Gradienten (BCG) [27, 53].

Wie angedeutet, ist dessen Konvergenzverlauf jedoch nicht monoton; es ist sogar durch starkes Oszillieren geprägt. Anders verhält sich dagegen das von Freund und Nachtigal entwickelte Verfahren der quasi-minimalen Residuen (QMR) [30]. In dieser auch auf kurzen Rekursionen basierenden Methode kann das Residuum – also der Fehler – als ein Produkt von zwei Faktoren angegeben werden. Anstatt nun den gesamten Fehler zu minimieren, was nur mit Hilfe von langen Rekursionen möglich ist, wird hier nur einer der beiden Faktoren minimiert. Dieser Ansatz führt zu einem nahezu monotonen Konvergenzverlauf.

In dieser Arbeit wird die Minimierung des Faktors, d.h. die Quasi-Minimierung des Residuums, in verschiedenen Normen durchgeführt. So entsteht eine erweiterte Klasse von QMR-Verfahren, mit deren Hilfe sich auch das oben erwähnte BCG-Verfahren interpretieren läßt, so daß ein neuer Einblick in die Zusammenhänge der beiden Algorithmen gewährt wird.

Bei der Implementierung eines iterativen Verfahrens auf einem Parallelrechner mit verteiltem Speicher stellt sich die parallele Ausführung des Matrix-Vektor-Produktes als die aufwendigste Operation heraus. Hierzu sind sowohl die Elemente der Matrix als auch die Komponenten der Vektoren auf die einzelnen Prozessoren verteilt, so daß neben den Rechenzeiten auch Kommunikationskosten durch den Datenaustausch zwischen den Prozessoren entstehen. Diese werden im wesentlichen durch die Besetzungsstruktur der Matrix bestimmt, so daß bei dünnbesetzten Matrizen oft nur benachbarte Prozessoren Daten austauschen müssen. Die Verteilung der Einträge der Matrix und Komponenten der Vektoren sollte daher so vorgenommen werden, daß die Gesamtausführungszeit des Matrix-Vektor-Produktes minimiert wird. Das Kommunikationsmuster, das sich bei dieser Operation auf einem massiv-parallelen System mit verteiltem Speicher ergibt, wird im folgenden als Kommunikationsschema oder als Kommunikationsstruktur bezeichnet.

Zusätzlich zu den aus Matrix-Vektor-Produkten resultierenden Kommunikationskosten fallen bei vielen Iterationsverfahren noch sogenannte globale Synchronisationspunkte an. Unter einem solchen Punkt versteht man eine Stelle im Programm, an der alle Prozessoren eines Parallelrechners miteinander kommunizieren müssen, um nachfolgende Rechnungen ausführen zu können. Diese Punkte sind beispielsweise zur Berechnung von Skalarprodukten notwendig.

Zusammenfassend ergibt sich also, daß die Kommunikationszeiten bei der Implementierung eines iterativen Verfahrens auf einem Parallelrechner mit verteiltem Speicher durch Matrix-Vektor-Produkte und globale Synchronisationspunkte bestimmt werden.

Die im Rahmen dieser Arbeit entwickelten Algorithmen werden deshalb so konstruiert, daß sie in jedem Iterationsschritt höchstens einen globalen Synchronisationspunkt besitzen. Darüber hinaus werden Beschleunigungsmethoden für iterative Verfahren bereitgestellt, die keine weiteren Kommunikationszeiten benötigen, und so nur die zur Berechnung des Matrix-Vektor-Produktes vorgegebene Kommunikationsstruktur benutzen. Dazu wird in dieser Arbeit ein Modell entwickelt, das eine Integration des Kommunikationsschemas in das zu lösende lineare Gleichungssystem erlaubt. Die

Idee ist, das in jeder Iteration notwendigerweise auszuführende Kommunikationsschema weitmöglichst auszunutzen.

Zur Beschleunigung der iterativen Verfahren unter dem Aspekt der Kommunikationsstrukturerhaltung werden sogenannte Gebietszerlegungsmethoden [33, 11, 12, 34, 49, 63, 50, 13, 71] analysiert, die eine dem Matrix-Vektor-Produkt ähnliche Kommunikationsstruktur aufweisen. Die Grundidee der Gebietszerlegung ist dabei, das Definitionsgebiet der partiellen Differentialgleichung in verschiedene Teilgebiete zu zerlegen, auf denen Teilprobleme formuliert und gelöst werden und die so zur Lösung des Gesamtproblems beitragen. Das bekannteste Verfahren dieser Art ist das alternierende Schwarz–Verfahren [70], das zur Lösung von Randwertproblemen eingesetzt werden kann (siehe Erläuterung zu Abbildung 1.1). Werden die Teilprobleme des alternierenden Prozesses nicht nacheinander, sondern gleichzeitig gelöst und läßt man dabei mehr als zwei sich überlappende Teilgebiete zu, so erhält man das sogenannte parallele Schwarz–Verfahren. Dieses Verfahren besitzt aber – wie später erläutert wird – eine dem Matrix-Vektor-Produkt ähnliche Kommunikationsstruktur, die durch den Austauch der aktuellen Randwerte der einzelnen Teilgebiete bestimmt wird. Der Vorteil dieses Verfahrens ist, daß zu seiner Beschreibung nur die auszutauschenden Randwerte benötigt werden. Die Werte, die nicht ausgetauscht werden, brauchen in dieser Formulierung nicht mehr berücksichtigt zu werden. Aufgrund der Reduzierung der Gesamtdatenmenge auf die auszutauschenden Daten können jetzt gezielte Techniken zur Beschleunigung des parallelen Schwarz–Verfahrens angegeben werden. Auf diese Weise ergeben sich kommunikationsstrukturerhaltende beschleunigte parallele Schwarz–Verfahren, deren Konvergenzverhalten ohne zusätzliche Kommunikation verbessert wird.

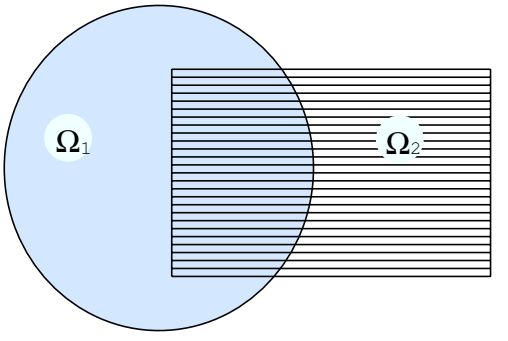


Abbildung 1.1: Durch abwechselndes Lösen der Probleme auf den überlappenden Teilgebieten Ω_1 und Ω_2 mit den jeweils aktuellen Randwerten wird das Gesamtproblem gelöst.

Da auch das Kommunikationsmodell bei der Lösung der linearen Gleichungssysteme integriert wurde, kann in der Arbeit gezeigt werden, wie sich die gewonnenen Beschleunigungstechniken auf die Lösung der linearen Gleichungssysteme übertragen lassen.

Die Zwischenlösungen, die beim parallelen Schwarz–Verfahren auf den Teilgebieten berechnet werden, können auch in einem Krylov–Teilraumverfahren zur Lösung des linearen Gleichungssystems, das durch Diskretisierung der partiellen Differentialgleichung auf dem gesamten Gebiet entstanden ist, verwendet werden. Die Ausnutzung von Teillösungen für die Lösung des Gesamtproblems wird im Kontext der linearen Gleichungssysteme als Vorkonditionierung bezeichnet. Da die Teilprobleme auf den Teilgebieten im parallelen Schwarz–Verfahren unabhängig und ohne zusätzliche Kommunikation gelöst werden, überträgt sich diese Eigenschaft auch auf den Vorkonditionierer. Aus diesem Grund werden die so konstruierten Vorkonditionierer häufig

bei der Implementierung eines Krylov–Teilraumverfahrens auf einem Parallelrechner mit verteiltem Speicher eingesetzt.

Genauso wie das parallele Schwarz–Verfahren als Vorkonditionierer in den bereitgestellten Krylov–Teilraumverfahren benutzt werden kann, wird in der Arbeit gezeigt, daß auch die neu entwickelten beschleunigten Schwarz–Verfahren als Vorkonditionierer eingesetzt werden können.

Die Arbeit ist wie folgt aufgebaut: Im zweiten Kapitel wird zunächst ein formaler Rahmen geschaffen, der die Vektoren, die in einem iterativen Verfahren benötigt werden, auf die Prozessoren eines Parallelrechners verteilt. Insbesondere liefert eine andere Darstellung des Matrix-Vektor-Produktes ein äquivalentes Gleichungssystem, das durch das Kommunikationsschema erweitert und im folgenden als erweitertes Gleichungssystem bezeichnet wird. Die Kommunikationsstruktur kann so beim Lösen des Systems berücksichtigt werden.

Im dritten Kapitel werden Krylov–Teilraumverfahren bereitgestellt. Dabei übernimmt eine neue Variante des QMR-Verfahrens [68] eine zentrale Rolle, die im Rahmen dieser Arbeit durch die Verwendung der l_p-Normen verallgemeinert wird. Neben numerischen Vergleichen, die an dreidimensionalen Beispielen durchgeführt werden, wird die Herleitung der hier vorgestellten neuen Methoden vollständig beschrieben. Einen Schwerpunkt bilden neu entwickelte parallele Varianten der Krylov–Teilraumverfahren [9, 8, 10], von denen gezeigt wird, daß sie mit nur einem globalen Synchronisationspunkt auskommen.

Im vierten Kapitel wird das parallele Schwarz–Verfahren als Grundlage verschiedener Gebietszerlegungsmethoden benutzt. Im eindimensionalen Fall kann dieses Verfahren durch die analytische Darstellung der jeweiligen Iterierten als Jacobi–Verfahren interpretiert werden, das lediglich die Funktionswerte in den Endpunkten der überlappenden Intervalle – also der auszutauschenden Variablen – benutzt. Geeignete Hilfsmittel aus der linearen Algebra führen schließlich zu einer deutlichen Beschleunigung des Verfahrens. Auf diese Weise entsteht ein paralleles Verfahren, das sich in die Klasse der Schwarz–Verfahren einreiht.

Im fünften Kapitel wird zunächst gezeigt, wie das parallele Schwarz–Verfahren als Vorkonditionierer in Krylov–Teilraumverfahren benutzt werden kann. Anschließend werden die zu diesem Schwarz–Verfahren gehörigen Vorkonditionierer sowohl für das betrachtete lineare Gleichungssystem als auch für das erweiterte Gleichungssystem hergeleitet. Durch die im vierten Kapitel vorgestellte neue leistungsfähige Technik, bei der nur die auszutauschenden Daten berücksichtigt werden müssen, werden beschleunigte Lösungsverfahren für das erweiterte Gleichungssystem gewonnen. Diese können wiederum als Vorkonditionierer in den hier entwickelten Krylov–Teilraumverfahren eingesetzt werden. Abschließend werden die neuen Verfahren an dreidimensionalen Modellproblemen auf dem massiv-parallelen System Intel Paragon XP/S 10 getestet. Dabei weist die neue QMR-Variante ähnlich gute numerische Eigenschaften wie das von Freund und Nachtigal [31] entwickelte Verfahren auf. Durch die Reduktion der globalen Synchronisationspunkte wird der Vorteil der bereitgestellten Varianten speziell beim Einsatz von massiv-parallelen Systemen deutlich. Auch wird gezeigt, daß die

zur Konvergenz benötigte Iterationsanzahl des parallelen Schwarz–Verfahrens durch die beschleunigten Methoden drastisch reduziert werden kann. Da in allen Verfahren Rechen- und Kommunikationsaufwand in jedem Iterationsschritt identisch sind, führen die neuen Methoden wegen ihrer höheren Konvergenzgeschwindigkeit zu wesentlich kürzeren Rechenzeiten. Darüber hinaus wird nachgewiesen, daß sich diese Eigenschaften auch auf Krylov–Teilraumverfahren übertragen, wenn sie mit den entsprechenden Methoden vorkonditioniert werden.

Abschließend sei betont, daß das parallele Schwarz–Verfahren in der parallelen Version des Finite-Elemente-Programmpaketes TRACE [75] verwendet wird, das am Forschungszentrum Jülich zur Simulation von Schadstoffausbreitungen im Boden eingesetzt wird. Auch findet das QMR-Verfahren eine Anwendung in PARTRACE [76], das den Stofftransport in porösen Medien simuliert. Die in dieser Arbeit vorgestellten Ergebnisse zeigen, daß die neu entwickelten Konzepte und Methoden auch für reale Anwendungen erfolgversprechend sind. Sie sollen zukünftig bei der Weiterentwicklung von TRACE und PARTRACE eingesetzt werden.

Kapitel 2

Aspekte der Parallelverarbeitung

In diesem Kapitel werden neben der Beschreibung von Rechnersystemen mit verteiltem Speicher Parallelisierungstechniken für Krylov–Teilraumverfahren angegeben. Insbesondere führt die formale Darstellung des Matrix-Vektor-Produktes auf einem System mit verteiltem Speicher zu dem Begriff der erweiterten Koeffizientenmatrix, deren Eigenschaften beschrieben werden.

2.1 Parallelrechner

Parallelrechner lassen sich, wie in [14, 43] beschrieben, nach mehreren Kriterien klassifizieren. In der Klasse der MIMD-Rechner (*Multiple Instruction Stream Multiple Data Stream*) unterscheidet man zwischen Systemen mit gemeinsamem und mit verteiltem Speicher. Erstere (*Shared Memory*) sind Rechner mit mehreren Prozessoren, die auf einen gemeinsamen physikalischen Speicher mit globalem Adreßraum zugreifen. Folglich werden Informationen zwischen den Prozessoren durch den gemeinsamen Zugriff auf die Daten ausgetauscht. Da aus praktischen Gründen nur wenige Verbindungen von den Prozessoren zu dem gemeinsamen Speicher bestehen, ist ihre Anzahl in Shared-Memory-Systemen nur gering.

2.1.1 Systeme mit verteiltem Speicher

Im Gegensatz zu Systemen mit gemeinsamem Speicher verfügen Parallelrechner mit verteiltem Speicher (*Distributed Memory*) über keinen globalen physikalischen Adreßraum. Jedem Prozessor ist seine eigene Speichereinheit zugeordnet, so daß er nur die Daten bearbeiten kann, die in seinem lokalen Adreßraum enthalten sind. Benötigt ein Prozessor die Daten eines anderen, so müssen diese durch explizite *Kommunikation* über ein Verbindungsnetzwerk ausgetauscht werden. Folglich können Daten anderer Prozessoren erst dann bearbeitet werden, wenn sie entsprechend verschickt und vom Adressaten empfangen wurden. Das Programmiermodell, das einen solchen Austausch von Nachrichten beinhaltet, heißt *Message Passing*.

Die Anzahl der Prozessoren, auch *Knoten* genannt, kann in einem Distributed-Memory-System sehr unterschiedlich sein. Es gibt Rechnersysteme mit bis zu einigen tausend Prozessoren, die daher durch den Begriff *massiv-parallel* charakterisiert werden.

Die Rechenleistung eines Parallelrechners mit verteiltem Speicher ist aber nicht nur von der Knotenleistung abhängig, sondern auch von der Leistungsfähigkeit des Netzwerkes, das die einzelnen Prozessoren verbindet. Die Struktur der Verbindung wird durch die Topologie bestimmt, wobei meistens Baum-, Ring-, Hypercube- oder Gitterstrukturen [14] verwendet werden. Letztere weist auch der im nachfolgenden Abschnitt beschriebene Rechner auf.

2.1.2 Der Rechner Intel Paragon XP/S 10

In diesem Abschnitt wird das am Forschungszentrum Jülich installierte System Intel Paragon XP/S 10 skizziert, auf dem die im Rahmen dieser Arbeit entwickelten Algorithmen implementiert wurden.

Der Intel Paragon ist ein massiv-paralleles System mit verteiltem Speicher, dessen Knoten durch ein Netzwerk in Form eines zweidimensionalen Gitters miteinander verbunden sind. Jeder Rechenknoten besitzt zwei RISC-Prozessoren (*Reduced Instruction Set Computer*) vom Typ i860 XP, die eine maximale Rechenleistung von 75 MFLOPS in 64-Bit-Arithmetik erreichen und mit einer Taktfrequenz von 50 MHz arbeiten. Neben dem *Applikationsprozessor*, der die arithmetischen und logischen Operationen ausführt, ist der *Kommunikationsprozessor* für den Austausch von Nachrichten zuständig. Jedem solchen Prozessor-Paar ist ein lokaler Hauptspeicher von 32 Megabyte zugeordnet. Die Zugriffszeit auf diesen Speicher beträgt 60 ns.

Der Rechnner Intel Paragon XP/S 10 am Zentralinstitut für Angewandte Mathematik des Forschungszentrums Jülich besitzt neben einigen Service- und I/O-Knoten 138 Rechenknoten. Auf diesen ist das Betriebssystem Paragon OSF/1 der Open Software Foundation installiert, das die von der Firma Intel zur Verfügung gestellt Message Passing-Bibliothek NX unterstützt. Hierfür wurde unter dem derzeitigen Release 1.4 von OSF/1 eine Bandbreite des Netzwerks von 90 Megabyte/s und eine Latenzzeit (*Startup Time*), die zum Aufbau einer Netzwerkverbindung benötigt wird, von 38 μs gemessen.

Weitere Informationen zum Rechner sind im Handbuch [46] zu finden. Die Systemarchitektur und der verwendete Prozessortyp sind zum Beispiel in [5, 58] beschrieben.

2.2 Parallele Algorithmen

Der Einsatz von parallelen Systemen erfordert die Entwicklung neuer Strategien und Methoden zur Lösung technisch-wissenschaftlicher Probleme. Parallele Algorithmen lassen sich einerseits bezüglich der verwendeten Rechnerarchitekturen und andererseits bezüglich der inhärenten Parallelität des zu lösenden Problems klassifizieren [44].

Falls beispielsweise ein Algorithmus die gleichen Operationen auf unterschiedlichen Daten ausführt, liegt *Datenparallelismus* vor, dessen Maß durch die *Datengranularität* gegeben wird. Sie bestimmt die Größe der Segmente eines Datensatzes, die parallel ausgeführt werden können. Die im nächsten Abschnitt entwickelten Parallelisierungsstrategien nutzen z.B. den grob-granularen Datenparallelismus von Krylov-Teilraumverfahren.

Ein weiterer wichtiger Aspekt beim Einsatz paralleler Algorithmen ist der Lastausgleich (*Load Balancing*), der sicherstellt, daß die Rechenlast gleichmäßig auf die Prozessoren verteilt ist. Der *Speedup* mißt dabei die Güte eines parallelen Algorithmus auf der verwendeten Rechnerarchitektur. Bezeichnet t_p die Ausführungszeit eines parallelen Algorithmus zur Lösung eines Problems auf p Prozessoren und ist t_{seq} die Zeit, die der sequentielle Algorithmus zur Lösung desselben Problems auf einem Knoten des Parallelrechners benötigt, so ergibt sich der Speedup als t_{seq}/t_p. Die Größe $t_{seq}/(p \cdot t_p)$ wird dann als *Effizienz* eines parallelen Algorithmus bezüglich der benutzten Rechnerarchitektur bezeichnet.

2.3 Parallelisierungskonzepte für iterative Verfahren

Im nachfolgenden Kapitel werden Krylov–Teilraumverfahren entwickelt, die zur Lösung des linearen Gleichungssystems

$$\mathbf{A}\mathbf{x} = \mathbf{b}\,, \qquad \mathbf{A} \in \mathbb{R}^{N\times N} \quad \text{und} \quad \mathbf{x}, \mathbf{b} \in \mathbb{R}^N$$

mit allgemeiner regulärer Koeffizientenmatrix geeignet sind. Um dem speziellen mathematischen Hintergrund der einzelnen Methoden nicht vorgreifen zu müssen, sollen zunächst formale Parallelisierungsansätze für Verfahren des folgenden Typs bereitgestellt werden.

Definition 2.3.1. *Ein iteratives Verfahren heißt cg-ähnlich, falls jeder Iterationsschritt durch eine konstante Anzahl von*

- *Linearkombinationen von Vektoren,*
- *Skalarprodukten,*
- *Matrix-Vektor-Produkten und*
- *skalaren Operationen*

bestimmt wird. Darüber hinaus soll die Anzahl der skalaren Operationen so klein sein, daß eine Parallelisierung dieser Operationen nicht notwendig ist.

Das Verfahren der konjugierten Gradienten ist folglich cg-ähnlich. Methoden, wie das auf langen Rekursionen basierende Verfahren der verallgemeinerten minimalen Residuen [67], werden durch sie nicht erfaßt.

In dieser Arbeit werden ausschließlich Verfahren beschrieben, die sich durch Definition 2.3.1 charakterisieren lassen.

Da die Parallelisierung skalarer Operationen entfällt, wird nun ein Formalismus bereitgestellt, der das Aufteilen der Vektoren des $\mathbb{R}^N$ auf p Prozessoren vornimmt.

Definition 2.3.2. *Sei $J = \{j_1, \ldots, j_m\} \subset I = \{1, 2, \ldots, N\}$ eine Indexmenge, deren Elemente in aufsteigender Reihenfolge gegeben sind, und $\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} \in \mathbb{R}^N$. Eine lineare Abbildung $\mathbf{R}_J : \mathbb{R}^N \to \mathbb{R}^m$ heißt Restriktion bezüglich der Indexmenge J, falls*

$$\mathbf{R}_J \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} = \mathbf{x}_J := \begin{pmatrix} x_{j_1} \\ \vdots \\ x_{j_m} \end{pmatrix} .$$

Die Restriktion bildet somit einen Vektor $\mathbf{x} \in \mathbb{R}^N$ auf die Komponenten ab, deren Indizes in $J = \{j_1, \ldots, j_m\}$ mit $j_1 < \ldots < j_m$ enthalten sind. Die Abbildung wird durch die Matrix $\mathbf{R}_J = (r_{ij}) \in \mathbb{R}^{m \times N}$ mit

$$r_{ij} = \begin{cases} 1 & , \; j = j_i \, , \\ 0 & , \; \text{sonst} \, , \end{cases}$$

beschrieben. So bewirkt das Produkt von $\mathbf{R}_J$ mit einer Matrix $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N \times N}$, daß die Zeilen von $\mathbf{A}$ mit den Indizes aus der Menge J herausgegriffen werden. Ist $K = \{k_1, \ldots, k_l\} \subset I$ mit $k_1 < \ldots < k_l$ eine weitere Indexmenge, so liefert die Komposition $\mathbf{R}_J \mathbf{A} \mathbf{R}_K^T = \mathbf{R}_J (\mathbf{R}_K \mathbf{A}^T)^T$ die Teilmatrix von $\mathbf{A}$, deren Spaltenindizes zu K und Zeilenindizes zu J gehören. Folglich gilt

$$\mathbf{R}_J \mathbf{A} \mathbf{R}_K^T = \begin{pmatrix} a_{j_1 k_1} & \ldots & a_{j_1 k_l} \\ \vdots & \ldots & \vdots \\ a_{j_m k_1} & \ldots & a_{j_m k_l} \end{pmatrix} . \tag{2.1}$$

Definition 2.3.3. *Eine Menge $\{I_i \, , \; i = 1, \ldots, p\}$ heißt p-disjunkte Zerlegung der Indexmenge $I = \{1, 2, \ldots, N\}$, falls*

$$\bigcup_{i=1}^{p} I_i = I \, ,$$
$$I_i \cap I_j = \emptyset \quad \textit{für} \quad i \neq j$$

und die Elemente jeder Teilmenge I_i in aufsteigender Reihenfolge gegeben sind.

Ein Vektor $\mathbf{x} \in \mathbb{R}^N$ heißt p-disjunkt verteilt auf p Prozessoren eines Parallelrechners, falls nur die Komponenten von $\mathbf{x}$ bezüglich I_i, d.h. $\mathbf{x}_{I_i} = \mathbf{R}_{I_i} \mathbf{x}$, im lokalen Speicher von Prozessor P_i vorhanden sind.

Die obige Definition läßt also zu, daß nicht nur zusammenhängende Blöcke eines Vektors auf die Prozessoren verteilt werden, sondern daß Komponenten mit beliebigen Indizes auf einem Prozessor vorliegen können. Damit kommt man der Forderung nach, daß die zur Problembeschreibung gewählte Numerierung eines Vektors nicht mit der Numerierung übereinstimmen muß, die aus Sicht der Parallelverarbeitung für eine günstige Verteilung der Komponenten auf die Prozessoren notwendig wäre.

2.3.1 Linearkombinationen von Vektoren

Im folgenden seien die beiden Vektoren $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$ auf die gleiche Weise p-disjunkt auf p Prozessoren verteilt, so daß die Linearkombination

$$\mathbf{z} = \alpha\,\mathbf{x} + \beta\,\mathbf{y}\,, \qquad \alpha, \beta \in \mathbb{R}\,,$$

durch

$$\mathbf{z}_{I_i} = \alpha\,\mathbf{x}_{I_i} + \beta\,\mathbf{y}_{I_i}\,, \quad i = 1, \ldots, p,$$

parallel auf den einzelnen Prozessoren durchgeführt werden kann. Sind die Komponenten des Ergebnisvektors $\mathbf{z}$ in gleicher Weise auf die Knoten des Rechners verteilt, so kann diese Addition ohne Kommunikation durchgeführt werden, was bei gleichmächtigen Mengen I_i zu einer optimalen Lastverteilung führt.

2.3.2 Skalarprodukte von Vektoren

Das euklidische Skalarprodukt der Vektoren $\mathbf{x}$ und $\mathbf{y}$ läßt sich aufgrund der Beziehung

$$\mathbf{x}^T\mathbf{y} = \sum_{i=1}^{p} \mathbf{x}_{I_i}^T \mathbf{y}_{I_i}$$

in zwei Phasen bestimmen. In der ersten Phase berechnet jeder Prozessor P_i die Teilsumme $\alpha_i = \mathbf{x}_{I_i}^T\mathbf{y}_{I_i}$ unabhängig von den anderen. In der zweiten Phase müssen die p lokalen Summen zu $\mathbf{x}^T\mathbf{y} = \alpha_1 + \ldots + \alpha_p$ addiert werden. Mit Hilfe eines *globalen Synchronisationspunktes* wird das Ergebnis auf allen Prozessoren zur Verfügung gestellt. Unter einem solchen Punkt versteht man eine Stelle im Programm, an der alle Prozessoren eines Parallelrechners miteinander kommunizieren müssen, um die nachfolgenden Rechnungen ausführen zu können.

Aufgrund der endlichen Gleitpunktdarstellung reeller Zahlen ist das Ergebnis einer Summation davon abhängig, in welcher Reihenfolge die Summanden addiert werden, so daß verschiedene Zerlegungen zu unterschiedlichen Ergebnissen führen können. Die Berechnung von Skalarprodukten unter dem Aspekt der numerischen Stabilität wird z.B. in [42] diskutiert.

2.3.3 Matrix-Vektor-Produkte

Im folgenden Abschnitt wird das Matrix-Vektor-Produkt $\mathbf{z} = \mathbf{A}\mathbf{x}$ mit der Koeffizientenmatrix $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N\times N}$ beschrieben. Hierzu wird gefordert, daß die Komponenten des Vektors

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_{I_1} \\ \vdots \\ \mathbf{x}_{I_p} \end{pmatrix} \in \mathbb{R}^N \qquad \text{und des Ergebnisvektors} \qquad \mathbf{z} = \begin{pmatrix} \mathbf{z}_{I_1} \\ \vdots \\ \mathbf{z}_{I_p} \end{pmatrix} \in \mathbb{R}^N$$

beide zu der gleichen p-disjunkten Zerlegung $\{I_i,\ i = 1,\ldots,p\}$ der Indexmenge $I = \{1,2,\ldots,N\}$ auf den p Prozessoren vorliegen. Auch sollen sich die Zeilen von $\mathbf{A}$, deren Zeilenindizes in I_i enthalten sind, im lokalen Speicher von P_i befinden.

Beispiel 2.3.1. *Wählt man zur Durchführung des Matrix-Vektor-Produktes*

$$\begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 0 & 0 & 3 & 0 \\ 4 & 5 & 6 & 0 & 0 & 0 \\ 7 & 8 & 9 & 0 & 10 & 11 \\ 0 & 0 & 0 & 12 & 13 & 14 \\ 15 & 0 & 0 & 16 & 0 & 17 \\ 0 & 0 & 0 & 0 & 0 & 18 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix}$$

auf zwei Prozessoren die Zerlegung $I_1 = \{1,2,3\}$ und $I_2 = \{4,5,6\}$, so erhält der erste Prozessor die Komponenten $\mathbf{x}_{I_1} = (x_1,x_2,x_3)^T$, $\mathbf{z}_{I_1} = (z_1,z_2,z_3)^T$ und die ersten drei Zeilen der Matrix. Der Menge I_2 entsprechend kann der zweite Prozessor auf die übrigen Daten zugreifen. Mit Hilfe von $\mathbf{x}_{I_1}$ läßt sich auf dem ersten Knoten nur die Komponente z_2 bestimmen, zur Berechnung von z_1 und z_3 fehlen x_5 und x_6.

Der Vektor $\mathbf{z}_{I_i}$ läßt sich lokal berechnen, falls alle dafür notwendigen Komponenten von $\mathbf{x}$ auf Prozessor P_i zur Verfügung stehen. Neben den lokal gespeicherten Daten $\mathbf{x}_{I_i}$ werden i.allg. weitere Komponenten $\mathbf{x}_{\partial I_i}$ benötigt. Die nachfolgende Definition ermittelt die Indizes der fehlenden Komponenten.

Definition 2.3.4. *Für $J \subset I = \{1,2,\ldots,N\}$ und $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N\times N}$ wird die Menge*

$$\partial J = \{k \in I \setminus J : \quad a_{jk} \neq 0 \quad \text{für ein } j \in J\}$$

als Nachbar von J bezüglich $\mathbf{A}$ oder kurz als Rand von J bezeichnet.

Die nicht vorhandenen Komponeneten $\mathbf{x}_{\partial I_i}$ müssen nun von den anderen Prozessoren angefordert werden. Nach Erhalt der Daten bezeichne $\widetilde{\mathbf{x}}_{\partial I_i}$ eine bei Prozessor P_i vorliegende Kopie von $\mathbf{x}_{\partial I_i}$. Der so ausgeführte Datenaustausch wird durch die Gleichung

$$\begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_{\partial I_1}\mathbf{R}_{I_1}^T & \ldots & \mathbf{R}_{\partial I_1}\mathbf{R}_{I_p}^T \\ \vdots & \ldots & \vdots \\ \mathbf{R}_{\partial I_p}\mathbf{R}_{I_1}^T & \ldots & \mathbf{R}_{\partial I_p}\mathbf{R}_{I_p}^T \end{pmatrix} \begin{pmatrix} \mathbf{x}_{I_1} \\ \vdots \\ \mathbf{x}_{I_p} \end{pmatrix} \tag{2.2}$$

beschrieben. Wird keine Nachricht von P_i nach P_j übermittelt, so ist $\mathbf{R}_{\partial I_j}\mathbf{R}_{I_i}^T = \mathbf{0}$, also ist insbesondere auch $\mathbf{R}_{\partial I_i}\mathbf{R}_{I_i}^T = \mathbf{0}$ für $i = 1, \ldots, p$. Findet ein Datentransfer statt, treten an den entsprechenden Stellen der Matrizen $\mathbf{R}_{\partial I_j}\mathbf{R}_{I_i}^T = \mathbf{R}_{\partial I_j}\mathbf{I}\,\mathbf{R}_{I_i}^T$ Einsen auf, vergleiche (2.1).

Definition 2.3.5. *Die in Gleichung* (2.2) *auftretende Matrix heißt Kommunikationsmatrix von* $\mathbf{A}$ *zur Zerlegung* $\{I_i\,,\ i = 1, \ldots, p\}$.

Weiter nennt man $\left(\widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \widetilde{\mathbf{x}}_{\partial I_p}\right)^T$ *den Empfangsvektor von* $\mathbf{x} = \left(\mathbf{x}_{I_1}, \ldots, \mathbf{x}_{I_p}\right)^T$ *und* $\mathbf{x}^{\mathbf{e}} = \left(\mathbf{x}_{I_1}, \widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \mathbf{x}_{I_p}, \widetilde{\mathbf{x}}_{\partial I_p}\right)^T$ *den durch den Empfangsvektor erweiterten Vektor* $\mathbf{x}$ *bezüglich* $\mathbf{A}$ *und* $\{I_i\,,\ i = 1, \ldots, p\}$.

Wie bereits erwähnt, kann das Matrix-Vektor-Produkt nach Erhalt der fehlenden Komponenten lokal durchgeführt werden. Da die Komposition $\mathbf{R}_J\mathbf{A}\mathbf{R}_K^T$ die Teilmatrix von $\mathbf{A}$ ist, deren Zeilenindizes in J und Spaltenindizes in K vorkommen, läßt sich nun das Produkt mit Hilfe des durch den Empfangsvektors erweiterten Vektor $\mathbf{x}^{\mathbf{e}}$ formulieren:

$$\begin{pmatrix} \mathbf{z}_{I_1} \\ \mathbf{z}_{I_2} \\ \vdots \\ \mathbf{z}_{I_p} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_{I_1}\mathbf{A}\mathbf{R}_{I_1}^T & \mathbf{R}_{I_1}\mathbf{A}\mathbf{R}_{\partial I_1}^T & \ldots & \mathbf{0} & \mathbf{0} \\ & & \ddots & & \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{R}_{I_p}\mathbf{A}\mathbf{R}_{I_p}^T & \mathbf{R}_{I_p}\mathbf{A}\mathbf{R}_{\partial I_p}^T \end{pmatrix} \begin{pmatrix} \mathbf{x}_{I_1} \\ \widetilde{\mathbf{x}}_{\partial I_1} \\ \vdots \\ \mathbf{x}_{I_p} \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix}, \tag{2.3}$$

d.h.

$$\mathbf{z}_{I_i} \;=\; \sum_{j=1}^{p} \mathbf{R}_{I_j}\mathbf{A}\mathbf{R}_{I_j}^T\,\mathbf{x}_{I_j} \;=\; \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T\,\mathbf{x}_{I_j} \;+\; \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{\partial I_i}^T\,\widetilde{\mathbf{x}}_{\partial I_i}\,. \tag{2.4}$$

Bei der Ausführung des Matrix-Vektor-Produktes können i.allg. Kommunikation und Rechnung überlappt werden, denn während Prozessor P_i auf die Daten $\mathbf{x}_{\partial I_i}$ wartet, können schon die Komponenten von $\mathbf{z}_{I_i}$ bestimmt werden, die nicht von $\mathbf{x}_{\partial I_i}$ abhängen. Das sind also die Werte von $\mathbf{z}_{I_i}$, die sich ausschließlich aus der Kenntnis von $\mathbf{x}_{I_i}$ berechnen lassen. Die folgende Definition bestimmt die zugehörige Indexmenge $\overset{\circ}{I}_i$ dieser Komponenten.

Definition 2.3.6. *Sei* $J \subset I = \{1, 2, \ldots, N\}$ *und* $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N\times N}$. *Das Innere von* J *ist die Menge*

$$\overset{\circ}{J} = \{k \in J \;:\quad a_{kl} = 0 \quad \textit{für alle } l \in I \setminus J\}\,.$$

Das Matrix-Vektor-Produkt kann nun auf jedem Prozessor P_i wie folgt durchgeführt werden:

- Verschicke die entsprechende Komponenten von $\mathbf{x}_{I_i}$, die andere Prozessoren für ihre Rechnung benötigen.

- Berechne $\mathbf{z}_{\overset{\circ}{I}_i}$ aus der lokalen Kenntnis von $\mathbf{x}_{I_i}$.
- Empfange Komponenten $\mathbf{x}_{\partial I_i}$ als Kopie in $\widetilde{\mathbf{x}}_{\partial I_i}$.
- Berechne $\mathbf{z}_{I_i \setminus \overset{\circ}{I}_i}$.

Entsteht die Matrix $\mathbf{A}$ durch Diskretisierung einer partiellen Differentialgleichung, so lassen sich die Begriffe Rand und Inneres einer Indexmenge auch anschaulich erläutern. Maßgebend dabei ist das Diskretisierungsschema.

Beispiel 2.3.2. *Diskretisiert man beispielsweise die Laplace–Gleichung $-\Delta u = 0$ mit Dirichlet–Randbedingung auf dem Einheitsquadrat mit der üblichen Fünfpunktformel [39] zum gegebenen 6×6 Gitter, siehe Abbildung 2.1, so entsteht bei natürlicher Anordnung der Unbekannten u_i, $i = 1, \ldots, 36$, ein Gleichungssystem mit der Koeffizientenmatrix*

$$\mathbf{A} = \begin{pmatrix} \mathbf{T} & -\mathbf{I} & & & & \\ -\mathbf{I} & \mathbf{T} & -\mathbf{I} & & & \\ & -\mathbf{I} & \mathbf{T} & -\mathbf{I} & & \\ & & -\mathbf{I} & \mathbf{T} & -\mathbf{I} & \\ & & & -\mathbf{I} & \mathbf{T} & -\mathbf{I} \\ & & & & -\mathbf{I} & \mathbf{T} \end{pmatrix} \quad \textit{mit } \mathbf{T} = \begin{pmatrix} 4 & -1 & & & & \\ -1 & 4 & -1 & & & \\ & -1 & 4 & -1 & & \\ & & -1 & 4 & -1 & \\ & & & -1 & 4 & -1 \\ & & & & -1 & 4 \end{pmatrix}.$$

Verteilt man die Komponenten des Vektors $\mathbf{u} = (u_1, \ldots, u_{36})$ auf einen Parallelrechner mit 4 Prozessoren gemäß der in Abbildung 2.1 angegebenen Blockzerlegung des Gebietes, so erhält der erste Prozessor die Komponenten des Vektors $\mathbf{u}$ bezüglich der Indexmenge $I_1 = \{1, 2, 3, 7, 8, 9, 13, 14, 15\}$. Das Innere $\overset{\circ}{I}_1$ dieser Menge sind alle Komponenten von $\mathbf{z} = \mathbf{A}\mathbf{u}$, die sich nur aus der Kenntnis von $\mathbf{u}_{I_1}$ bestimmen lassen, also $\overset{\circ}{I}_1 = \{1, 2, 7, 8\}$. Der Rand von I_1 sind schließlich die Komponenten von $\mathbf{u}$, die zusätzlich zu $\mathbf{u}_{I_1}$ benötigt werden, um $\mathbf{z}$ auf $I_1 \setminus \overset{\circ}{I}_1$ berechnen zu können, somit die Indexmenge $\partial I_1 = \{4, 10, 16, 19, 20, 21\}$.

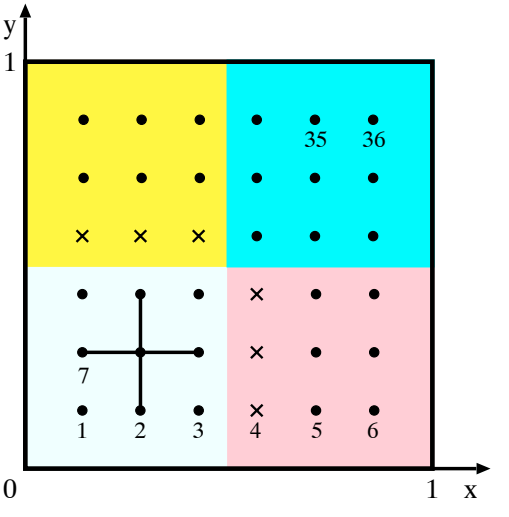


Abbildung 2.1: *In vier Teilgebiete zerlegtes Einheitsquadrat mit zweidimensionalem 6×6 Gitter und natürlicher Numerierung der Gitterpunkte*

Für die in der Gleichung (2.3) angegebene Durchführung des Matrix-Vektor-Produktes ist es erforderlich, daß jeder Prozessor P_i auf die Zeilen der Matrix $\mathbf{A}$ zugreifen kann, deren Indizes zur Indexmenge I_i gehören. Falls die Matrix durch Diskretisierung einer partiellen Differentialgleichung entstanden ist, so ist sie i.allg. dünnbesetzt

(*sparse*). Das bedeutet, daß der überwiegende Teil der Matrixelemente den Wert Null besitzt oder genauer, daß man zur Durchführung des Marix-Vektor-Produktes $\mathcal{O}(N)$ Additionen und Multiplikationen benötigt, wenn N die Ordnung der Matrix bezeichnet. Eine oft verwendete Speichertechnik für solche Matrizen ist das sogenannte CRS-Schema [66]. Weitere Speichertechniken befinden sich in [80].

Ein schwieriges Problem ist das Auffinden einer Zerlegung $\{I_i, i = 1, \ldots, p\}$ der Indexmenge $I = \{1, 2, \ldots, N\}$, die zur Durchführung des Matrix-Vektor-Produktes mit der Koeffizientenmatrix $\mathbf{A}$ sinnvoll erscheint. Es muß eine Zerlegung gefunden werden, die die Gesamtlaufzeit, bestehend aus Berechnungs– und Kommunikationszeit, auf p Prozessoren minimiert. Entsteht die Matrix durch Diskretisierung einer partiellen Differentialgleichung, so ist das Matrix-Vektor-Produkt bezüglich I_i i.allg. proportional zur Mächtigkeit dieser Menge, so daß das Minimierungsproblem vereinfacht dargestellt werden kann. Man sucht dann eine Zerlegung $\{I_i\,,\ i = 1, \ldots, p\}$, die sowohl die maximale Mächtigkeit der Mengen I_i

$$\max_{i=1,\ldots,p} |I_i| \longrightarrow \min\,,$$

als auch die Kommunikationskosten minimiert,

$$\sum_{i=1}^{p} |\partial I_i| \longrightarrow \min\ .$$

Mit Hilfe des von $\mathbf{A}$ induzierten Graphen läßt sich hieraus ein Problem formulieren, das der Partitionierung von Graphen entspricht. Bei der durch Diskretisierung einer partiellen Differentialgleichung auf dem Gebiet Ω entstandenen Matrix $\mathbf{A}$ impliziert die Aufteilung des Gebietes oder des Diskretisierungsgitters eine i.allg. gute Zerlegung der Indexmenge I. Partitionierungsalgorithmen, die auf der physikalischen Zerteilung des Gebietes Ω basieren, findet man in [32, 57]; Arbeiten zur Gitterpartitionierung sind in [3] bzw. zur Graphpartitionierung in [47, 62] beschrieben.

2.4 Die erweiterte Koeffizientenmatrix

Sendevorgang (2.2) und Matrix-Vektor-Produkt (2.3) können in der nachfolgenden Gleichung zusammengefaßt werden:

$$\begin{pmatrix} \mathbf{z}_{I_1} \\ \mathbf{0} \\ \mathbf{z}_{I_2} \\ \mathbf{0} \\ \vdots \\ \mathbf{z}_{I_{p-1}} \\ \mathbf{0} \\ \mathbf{z}_{I_p} \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_1 & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & -\mathbf{R}_{1,2} & \mathbf{0} & \ldots & -\mathbf{R}_{1,p} & \mathbf{0} \\ & & & & & & \\ & & & \ddots & & & \\ & & & \ddots & & & \\ & & & & & & \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} & \mathbf{A}_{pp} & \mathbf{A}_p \\ -\mathbf{R}_{p,1} & \mathbf{0} & \ldots & -\mathbf{R}_{p,p-1} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{x}_{I_1} \\ \widetilde{\mathbf{x}}_{\partial I_1} \\ \mathbf{x}_{I_2} \\ \widetilde{\mathbf{x}}_{\partial I_2} \\ \vdots \\ \mathbf{x}_{I_{p-1}} \\ \widetilde{\mathbf{x}}_{\partial I_{p-1}} \\ \mathbf{x}_{I_p} \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix}. \tag{2.5}$$

Hierbei sind die Matrizen $\mathbf{A}_{ii}$, $\mathbf{A}_i$ und $\mathbf{R}_{i,j}$ durch

$$\mathbf{A}_{ii} = \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T \qquad \mathbf{A}_i = \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{\partial I_i}^T \qquad \text{für } i = 1,\ldots,p$$

und

$$\mathbf{R}_{i,j} = \mathbf{R}_{\partial I_i}\mathbf{R}_{I_j}^T\,, \quad i,j = 1,\ldots,p\,,$$

gegeben. Werden in dieser Gleichung ausschließlich die Werte $\mathbf{x}_{I_i}$ für $i = 1,\ldots,p$ als bekannt vorausgesetzt, so ergeben sich äquivalent zu (2.2) die Werte $\widetilde{\mathbf{x}}_{\partial I_i}$ mit den Blockzeilen $2,\ 4,\ \ldots\ ,\ 2p$. Schließlich läßt sich $\mathbf{z}_{I_i}$ gemäß (2.3) mit Hilfe der Blockzeilen $1,\ 3,\ \ldots\ ,\ 2p-1$ berechnen.

Definition 2.4.1. *Die Matrix aus Gleichung* (2.5) *nennt man die erweiterte Koeffizientenmatrix* $\mathbf{A}^{\mathbf{e}}$ *von* $\mathbf{A}$ *bezüglich der Zerlegung* $\{I_i\,,\ i = 1,\ldots,p\}$.

Entsprechend heißt der Vektor $\mathbf{z}^{\mathbf{e}} = \left(\mathbf{z}_{I_1},\mathbf{0},\ldots,\mathbf{z}_{I_p},\mathbf{0}\right)^T$ *der mit Null erweiterte Vektor von* $\left(\mathbf{z}_{I_1},\ldots,\mathbf{z}_{I_p}\right)^T$ *bezüglich* $\mathbf{A}$ *und* $\{I_i\,,\ i = 1,\ldots,p\}$.

Die erweiterte Koeffizientenmatrix $\mathbf{A}^{\mathbf{e}}$ der Matrix $\mathbf{A}$ wird in Kapitel 5 eine zentrale Rolle bei der Lösung linearer Gleichungssysteme übernehmen. Für einen sequentiellen Rechner ($p = 1$) sind beide Matrizen identisch. Die gerade eingeführten Begriffe werden nun an einem Beispiel erläutert.

Beispiel 2.4.1. *Diskretisiert man die gewöhnliche Differentialgleichung* $-u'' = f$ *im Gebiet* $\Omega = (0,1)$ *mit homogenen Dirichlet–Randwerten durch die sogenannte zweite Differenz*

$$\frac{u(x+h) - 2u(x) + u(x-h)}{h^2} = u''(x) + \mathcal{O}(h^2)$$

mit $h = 1/7$*, so entsteht das Gleichungssystem*

$$\underbrace{\begin{pmatrix} 2/h^2 & -1/h^2 & & & & \\ -1/h^2 & 2/h^2 & -1/h^2 & & & \\ & -1/h^2 & 2/h^2 & -1/h^2 & & \\ & & -1/h^2 & 2/h^2 & -1/h^2 & \\ & & & -1/h^2 & 2/h^2 & -1/h^2 \\ & & & & -1/h^2 & 2/h^2 \end{pmatrix}}_{\mathbf{A}} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{pmatrix} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \\ f(x_4) \\ f(x_5) \\ f(x_6) \end{pmatrix},$$

wobei die Werte u_i *die Funktion* u *an den Stellen* $x_i = ih$ *für* $i = 1,\ldots,6$ *approximieren. Soll dieses Problem auf zwei Prozessoren gelöst werden, so liegt folgende Zerlegung der Indexmenge* $I = \{1,\ldots,6\}$ *nahe:*

$$\begin{aligned} I_1 &= \{1,2,3\}, & I_2 &= \{4,5,6\}\,, \\ \partial I_1 &= \{4\}, & \partial I_2 &= \{3\}\,. \end{aligned}$$

Das Matrix-Vektor-Produkt $\mathbf{z} = \mathbf{A}\mathbf{u}$ *läßt sich mit der zu* $\mathbf{A}$ *erweiterten Matrix* $\mathbf{A}^{\mathbf{e}}$ *durchführen:*

$$\begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ 0 \\ z_4 \\ z_5 \\ z_6 \\ 0 \end{pmatrix} = \begin{pmatrix} 2/h^2 & -1/h^2 & & & & & & \\ -1/h^2 & 2/h^2 & -1/h^2 & & & & & \\ & -1/h^2 & 2/h^2 & -1/h^2 & & & & \\ & & & 1 & -1 & & & \\ & & & & 2/h^2 & -1/h^2 & & -1/h^2 \\ & & & & -1/h^2 & 2/h^2 & -1/h^2 & \\ & & & & & -1/h^2 & 2/h^2 & \\ & & -1 & & & & & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \widetilde{u}_4 \\ u_4 \\ u_5 \\ u_6 \\ \widetilde{u}_3 \end{pmatrix}.$$

Dabei entspricht die vierte und achte Zeile dem Sendevorgang. Die vierte Zeile lautet

$$\widetilde{u}_4 = u_4\,,$$

d.h. sende vom zweiten Prozessor den Wert von u_4 an den ersten Prozessor und speichere diesen in $\widetilde{u}_4$. Entsprechend läßt sich die achte Zeile interpretieren. Anschließend wird das Matrix-Vektor-Produkt auf den einzelnen Prozessoren lokal durchgeführt. Auf dem ersten Prozessor ergeben sich die Komponenten von $\mathbf{z}_{I_1} = \{z_1, z_2, z_3\}$ durch Kenntnis von $\mathbf{u}_{I_1} = \{u_1, u_2, u_3\}$ und der Variablen $\widetilde{u}_4$, was durch die ersten drei Zeilen der Matrix $\mathbf{A^e}$ beschrieben wird. Mit der fünften, sechsten und siebten Zeile von $\mathbf{A^e}$ erhält man auf dem zweiten Prozessor die Komponenten $\mathbf{z}_{I_2} = \{z_4, z_5, z_6\}$.

Zum Abschluß des Kapitels wird erläutert, wie die erweiterte Matrix zur Lösung des linearen Gleichungssystems $\mathbf{Ax} = \mathbf{b}$ eingesetzt werden kann. Es wird sich herausstellen, daß das Lösen der beiden Systeme mit den Koeffizientenmatrizen $\mathbf{A}$ und $\mathbf{A^e}$ zu derselben Lösung führt. Der nachfolgende Satz präzisiert diesen Sachverhalt.

Satz 2.4.1. *Sei $\{I_i,\, i = 1, \ldots, p\}$ eine Zerlegung der Indexmenge $I = \{1, 2, \ldots, N\}$. Weiter sei $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{x} = \left(\mathbf{x}_{I_1}, \ldots, \mathbf{x}_{I_p}\right)^T \in \mathbb{R}^N$ und $\mathbf{b} = \left(\mathbf{b}_{I_1}, \ldots, \mathbf{b}_{I_p}\right)^T \in \mathbb{R}^N$. Dann gilt:*

(i) $\mathbf{A}$ *regulär* $\iff$ $\mathbf{A^e}$ *regulär.*

(ii) Ist $\mathbf{x}$ *Lösung von* $\mathbf{Ax} = \mathbf{b}$ *und* $\begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix} := \begin{pmatrix} \mathbf{x}_{\partial I_1} \\ \vdots \\ \mathbf{x}_{\partial I_p} \end{pmatrix}$, *so löst der erweiterte Vektor* $\mathbf{x^e} = \left(\mathbf{x}_{I_1}, \widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \mathbf{x}_{I_p}, \widetilde{\mathbf{x}}_{\partial I_p}\right)^T$ *das Gleichungssystem* $\mathbf{A^e x^e} = \mathbf{b^e}$ *mit* $\mathbf{b^e} = \left(\mathbf{b}_{I_1}, \mathbf{0}, \ldots, \mathbf{b}_{I_p}, \mathbf{0}\right)^T$.

(iii) Ist $\mathbf{x^e}$ *Lösung von* $\mathbf{A^e x^e} = \mathbf{b^e}$, *so löst* $\mathbf{x}$ *die Gleichung* $\mathbf{Ax} = \mathbf{b}$.

Beweis. Die Behauptungen (ii) und (iii) folgen unmittelbar aus der Darstellung des Matrix-Vektor-Produktes (2.5). Da die Regularität einer Matrix dadurch gekennzeichnet ist, daß die homogene Gleichung nur die triviale Lösung besitzt, folgt (i) als Spezialfall von (ii) und (iii). □

Teil (iii) von Satz 2.4.1 bietet die Möglichkeit, die Lösung von $\mathbf{A}\mathbf{x} = \mathbf{b}$ mit Hilfe des erweiterten Systems $\mathbf{A}^{\mathbf{e}}\mathbf{x}^{\mathbf{e}} = \mathbf{b}^{\mathbf{e}}$ zu ermitteln. Anstatt das ursprüngliche Gleichungssystem zu lösen, kann ein Krylov–Teilraumverfahren verwendet werden, das die Lösung des erweiterten Gleichungssystem $\mathbf{A}^{\mathbf{e}}\mathbf{x}^{\mathbf{e}} = \mathbf{b}^{\mathbf{e}}$ bestimmt. Die zusätzlichen Variablen $\widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \widetilde{\mathbf{x}}_{\partial I_p}$, die bei der Durchführung des Matrix-Vektor-Produktes auf einem Parallelrechner mit verteiltem Speicher hinzugenommen werden, werden später zur Beschleunigung des eingesetzten Verfahrens benutzt.

Kapitel 3

Krylov–Teilraumverfahren

Die Lösung großer linearer Gleichungssysteme stellt besondere Anforderungen an die Algorithmen, wobei direkte Methoden aufgrund ihres hohen Speicherbedarfs oft nicht verwendet werden können. Eine Alternative bieten daher iterative Verfahren, die die Koeffizientenmatrix ausschließlich in der Form von Matrix-Vektor-Produkten benötigen. Viele solcher Methoden zur Lösung des linearen Gleichungssystems

$$\mathbf{A}\mathbf{x} = \mathbf{b} \qquad \text{mit } \mathbf{A} \in \mathbb{R}^{N\times N} \text{ und } \mathbf{x}, \mathbf{b} \in \mathbb{R}^N \tag{3.1}$$

gehören zu der Klasse der Krylov–Teilraumverfahren, die Approximationen $\mathbf{x}_n$ der Lösung $\mathbf{A}^{-1}\mathbf{b}$ in der Form

$$\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, \mathbf{A}), \quad n \in \mathbb{N},$$

liefern. Dabei bezeichnen $\mathbf{x}_0$ eine Startlösung für die Gleichung (3.1) und $\mathcal{K}_n(\mathbf{r}_0, \mathbf{A}) = \text{span}\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \mathbf{A}^2\mathbf{r}_0, \ldots, \mathbf{A}^{n-1}\mathbf{r}_0\}$ den n-ten Krylov–Teilraum, der durch den Residuumsvektor $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$ und die Matrix $\mathbf{A}$ erzeugt wird.

Ein solches Verfahren besteht im wesentlichen aus zwei Komponenten. Einerseits muß eine Basis $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ mit

$$\text{span}\{\mathbf{v}_1, \ldots, \mathbf{v}_n\} = \mathcal{K}_n(\mathbf{r}_0, \mathbf{A})$$

(iterativ) konstruiert werden und andererseits muß eine Vorschrift angegeben werden, die die Approximation in dem Raum $\mathbf{x}_0 + \text{span}\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ bestimmt. Die Basis sollte daher so gewählt werden, daß $\mathbf{x}_n$ mit wenig Aufwand berechnet werden kann.

Die bekannteste Krylov–Teilraummethode ist das Verfahren der konjugierten Gradienten zur Lösung linearer Gleichungssysteme mit symmetrisch positiv definiter Koeffizientenmatrix $\mathbf{A}$. Der Raum $\mathcal{K}_n(\mathbf{r}_0, \mathbf{A})$ wird dabei durch die generierten $\mathbf{A}$-konjugierten Vektoren aufgespannt. Mit den drei folgenden äquivalenten Bedingungen lassen sich die Iterierten $\mathbf{x}_n$ dieses Verfahrens charakterisieren. Zum einen wird der Vektor $\mathbf{x}_n$ so gewählt, daß der Fehler $\mathbf{x}_n - \mathbf{A}^{-1}\mathbf{b}$ im erzeugten affinen Raum in der $\mathbf{A}$-Norm minimal ist, d.h.

$$\|\mathbf{x}_n - \mathbf{A}^{-1}\mathbf{b}\|_{\mathbf{A}} \quad = \quad \min_{\mathbf{x}\in\mathbf{x}_0+\mathcal{K}_n(\mathbf{r}_0,\mathbf{A})} \|\mathbf{x} - \mathbf{A}^{-1}\mathbf{b}\|_{\mathbf{A}}\,.$$

Zum anderen aber soll das zugehörige Residuum $\mathbf{b} - \mathbf{A}\mathbf{x}_n$ in der $\mathbf{A}^{-1}$-Norm minimiert werden, also

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}_n\|_{\mathbf{A}^{-1}} \quad = \quad \min_{\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, \mathbf{A})} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_{\mathbf{A}^{-1}} \, .$$

In beiden Fällen stellt $\|\mathbf{x}\|_{\mathbf{B}} = \mathbf{x}^T \mathbf{B} \mathbf{x}$, $\mathbf{x} \in \mathbb{R}^N$, für symmetrisch positiv definite Matrizen $\mathbf{B}$ eine Norm dar. Zusätzlich sind die beiden Minimierungseigenschaften zu der Galerkin–Bedingung

$$\mathbf{w}^T (\mathbf{b} - \mathbf{A}\mathbf{x}_n) = 0 \quad \text{für alle } \mathbf{w} \in \mathcal{K}_n(\mathbf{r}_0, \mathbf{A}) = \mathcal{K}_n(\mathbf{r}_0, \mathbf{A}^T) \tag{3.2}$$

äquivalent. Da für allgemeine reguläre Koeffizientenmatrizen $\mathbf{A}$ jedoch weder $\|\cdot\|_{\mathbf{A}}$ noch $\|\cdot\|_{\mathbf{A}^{-1}}$ eine Norm ist, wird nun die übliche euklidische Norm $\|\cdot\|_2$ verwendet. Ein Beispiel dafür ist GMRES, wo die Iterierte durch die Minimierungseigenschaft

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}_n\|_2 \quad = \quad \min_{\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, \mathbf{A})} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 \tag{3.3}$$

bestimmt wird. Das Verfahren der bikonjugierten Gradienten hingegen basiert auf der Orthogonalitätsbedingung (3.2) und erzeugt hierzu einen weiteren Raum $\mathcal{K}_n(\widetilde{\mathbf{r}}_0, \mathbf{A}^T)$ mit dessen Hilfe die Approximierten $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, \mathbf{A})$ durch die Forderung

$$\mathbf{w}^T (\mathbf{b} - \mathbf{A}\mathbf{x}_n) = 0 \quad \text{für alle } \mathbf{w} \in \mathcal{K}_n(\widetilde{\mathbf{r}}_0, \mathbf{A}^T) \tag{3.4}$$

ermittelt werden. Dabei bezeichnet $\widetilde{\mathbf{r}}_0$ einen beliebigen Startvektor zur Generierung des Krylov–Teilraums $\mathcal{K}_n(\widetilde{\mathbf{r}}_0, \mathbf{A}^T)$, der i.allg. in Übereinstimmung mit $\mathbf{r}_0$ gewählt wird.

Die Berechnung der Iterierten wird im dritten Abschnitt dieses Kapitels näher beschrieben. Zunächst soll auf die Erzeugung einer geeigneten Basis eingegangen werden. Speziell werden Krylov–Teilraumverfahren zur Lösung von linearen Gleichungssystemen mit allgemeiner regulärer Koeffizientenmatrix beschrieben, deren Basis $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ mit Hilfe des unsymmetrischen Lanczos–Prozesses [52] generiert wird. Dieser cg-ähnliche Prozeß benötigt die Matrix $\mathbf{A}$ ausschließlich in der Form der beiden Matrix-Vektor-Produkte $\mathbf{A}\mathbf{x}$ und $\mathbf{A}^T\mathbf{x}$ zu beliebigen Vektoren $\mathbf{x}$. Soll der Lanczos–Algorithmus parallelisiert werden, müssen bei der Durchführung dieser Operationen und der auftretenden Skalarprodukte Daten ausgetauscht werden. Ist die Matrix dünnbesetzt, so kommunizieren bei der Parallelisierung der Matrix-Vektor-Produkte i.allg. nur benachbarte Prozessoren. Die ungünstigen Parallelisierungseigenschaften der Methode lassen sich dann auf die vorhandenen Skalarprodukte zurückführen [7, 17, 18, 19, 36]. Da diese mit globalen Synchronisationspunkten bestimmt werden, d.h. den Datenaustauch zwischen allen Prozessoren erfordern, wird die Performance bei steigender Prozessoranzahl gerade durch die Berechnung dieser Skalarprodukte negativ beeinflußt.

Es können zwei grundsätzliche Strategien angegeben werden, die diese Schwierigkeiten beheben. Zum einen kann der sequentielle Algorithmus so umstrukturiert

werden, daß die anfallende Kommunikation mit sinnvoller Rechnung überlappt wird. Diese Technik wird von den meisten Parallelrechnern unterstützt. Der zweite Ansatz hingegen zielt auf die Konstruktion neuer Algorithmen, die mit einer geringeren Anzahl von globalen Synchronisationspunkten pro Iteration auskommen. Aus Gründen der Lastverteilung sind dabei Algorithmen mit nur einem globalen Synchronisationspunkt pro Iterationsschritt besonders wichtig. Zur Reduzierung der Anzahl der globalen Synchronisationspunkte werden manchmal sogar zusätzliche Rechnungen in Kauf genommen [65]. Die Vor- und Nachteile der beiden Strategien für das Verfahren der konjugierten Gradienten sind in [21] zusammengefaßt.

In dieser Arbeit wird nicht der ersten, sondern der zweiten Strategie nachgegangen. In Abschnitt 3.2 wird eine neue Version des Lanczos–Algorithmus bereitgestellt, die in jedem Iterationsschritt nur einen globalen Synchronisationspunkt benötigt, ohne dafür zusätzliche Rechnungen durchzuführen. Mit Hilfe der durch diesen Algorithmus erzeugten Basis wird das Verfahren der bikonjugierten Gradienten und der quasi-minimalen Residuen hergeleitet. Letzteres wird dann durch die Verwendung der l_p-Normen verallgemeinert.

Nachfolgend bezeichnen $\mathbf{e}_i^{(N)} \in \mathbb{R}^N$ die i-ten Einheitsvektoren und für $1 \leq p < \infty$ $\|\mathbf{x}\|_p = \left(\sum_{i=1}^N |x_i|^p\right)^{\frac{1}{p}}$ und $\|\mathbf{x}\|_\infty = \max_{i=1,\ldots,N} |x_i|$ mit $\mathbf{x} \in \mathbb{R}^N$ die l_p-Normen.

3.1 Der klassische unsymmetrische Lanczos–Prozeß

Der klassische unsymmetrische Lanczos–Prozess [52] transformiert eine beliebige Matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ iterativ in eine tridiagonale Matrix $\mathbf{T}$, deren Eigenwerte die Eigenwerte von $\mathbf{A}$ approximieren. Zur Herleitung des Lanczos–Algorithmus nehme man an, daß die Matrix

$$\mathbf{T} = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \gamma_2 & \alpha_2 & \beta_3 & & \\ & \gamma_3 & \ddots & \ddots & \\ & & \ddots & \ddots & \beta_N \\ & & & \gamma_N & \alpha_N \end{pmatrix} \in \mathbb{R}^{N \times N}$$

durch die Ähnlichkeitstransformation

$$\mathbf{T} = \mathbf{V}^{-1}\mathbf{A}\mathbf{V} \tag{3.5}$$

entstanden ist. Bezeichnet $\mathbf{W}$ die transponierte Inverse von $\mathbf{V}$, so ist (3.5) äquivalent zu den drei Gleichungen

$$\mathbf{W}^T\mathbf{V} = \mathbf{I} \tag{3.6}$$

$$\mathbf{A}\mathbf{V} = \mathbf{V}\mathbf{T} \tag{3.7}$$

$$\mathbf{A}^T\mathbf{W} = \mathbf{W}\mathbf{T}^T \quad . \tag{3.8}$$

Der unsymmetrische Lanczos–Prozeß besteht nun darin, die Matrizen $\mathbf{V}$, $\mathbf{W}$ und $\mathbf{T}$ zu berechnen, d.h. rekursiv die Spaltenvektoren $\mathbf{v}_n$ und $\mathbf{w}_n$ von

$$\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N) \quad \text{und} \quad \mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N)$$

und die skalaren Elemente α_n, β_n und γ_n der Matrix $\mathbf{T}$ zu bestimmen. Basierend auf den Gleichungen (3.6) – (3.8) werden zu zwei beliebigen Startvektoren $\mathbf{v}_1$ und $\mathbf{w}_1$, die der Biorthogonalitätseigenschaft $\mathbf{w}_1^T \mathbf{v}_1 = 1$ genügen, nun iterativ die Matrix $\mathbf{T}$ und zwei Folgen von Vektoren $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots$ und $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \ldots$ generiert. Diese werden als *Lanczos–Vektoren* bezeichnet. Der zugehörige Algorithmus kann wie folgt entwickelt werden. Die Eigenschaft (3.6) entspricht der Biorthogonalität

$$\mathbf{w}_i^T \mathbf{v}_j = \begin{cases} 0, & \text{falls } i \neq j \\ 1, & \text{falls } i = j \end{cases} , \tag{3.9}$$

der Lanczos–Vektoren. Vergleicht man die n-ten Spalten von (3.7) und (3.8), so erhält man die Gleichungen

$$\mathbf{A}\mathbf{v}_n = \gamma_{n+1}\mathbf{v}_{n+1} + \alpha_n \mathbf{v}_n + \beta_n \mathbf{v}_{n-1} \tag{3.10}$$

$$\mathbf{A}^T \mathbf{w}_n = \beta_{n+1}\mathbf{w}_{n+1} + \alpha_n \mathbf{w}_n + \gamma_n \mathbf{w}_{n-1} \ , \tag{3.11}$$

wobei $\mathbf{v}_0 = \mathbf{w}_0 = \mathbf{0}$ zu setzen sind. Werden diese nach $\mathbf{v}_{n+1}$ bzw. $\mathbf{w}_{n+1}$ aufgelöst, so ergeben sich hieraus Rekursionen für die $(n+1)$-Vektoren. Die Bedingung (3.9) bestimmt im wesentlichen die Berechnung der Koeffizienten α_n, β_{n+1} und γ_{n+1}. Der Koeffizient α_n ergibt sich nach Multiplikation der Gleichung (3.10) mit $\mathbf{w}_n^T$ oder (3.11) mit $\mathbf{v}_n^T$ aus der Biorthogonalität (3.9) zu

$$\alpha_n = \mathbf{w}_n^T \mathbf{A} \mathbf{v}_n \ . \tag{3.12}$$

Zur Berechnung von β_{n+1} und γ_{n+1} setzt man zunächst

$$\widetilde{\mathbf{v}}_{n+1} = \mathbf{A}\mathbf{v}_n - \alpha_n \mathbf{v}_n - \beta_n \mathbf{v}_{n-1} \tag{3.13}$$

und

$$\widetilde{\mathbf{w}}_{n+1} = \mathbf{A}^T \mathbf{w}_n - \alpha_n \mathbf{w}_n - \gamma_n \mathbf{w}_{n-1} \ , \tag{3.14}$$

wobei angenommen wird, daß die Größen β_n und γ_n vom vorherigen Iterationsschritt bekannt sind. Aufgrund der Gleichungen (3.10) und (3.11) folgt hieraus

$$\widetilde{\mathbf{v}}_{n+1} = \gamma_{n+1}\mathbf{v}_{n+1} \tag{3.15}$$

und

$$\widetilde{\mathbf{w}}_{n+1} = \beta_{n+1}\mathbf{w}_{n+1} \ , \tag{3.16}$$

so daß die Biorthogonalität der Lanczos–Vektoren $\mathbf{v}_{n+1}$ und $\mathbf{w}_{n+1}$ die Identität

$$\gamma_{n+1}\beta_{n+1} = \widetilde{\mathbf{w}}_{n+1}^T \widetilde{\mathbf{v}}_{n+1} \tag{3.17}$$

impliziert. So erhält man aus den Gleichungen (3.12) bis (3.17) den unsymmetrischen Lanczos–Algorithmus, der wegen der Struktur von (3.13) und (3.14) auf *Drei-Term-Rekursionen* basiert. Dieser Prozeß ist in Algorithmus 3.1 beschrieben. Dabei sind Anweisungen, die zu einem globalen Synchronisationspunkt auf einem Parallelrechner mit verteiltem Speicher führen, mit einem • gekennzeichnet.

Algorithmus 3.1 Klassischer unsymmetrischer Lanczos–Prozeß

Eingabe: $\mathbf{A}, \widetilde{\mathbf{v}}_1$ und $\widetilde{\mathbf{w}}_1$ mit $\widetilde{\mathbf{w}}_1^T \widetilde{\mathbf{v}}_1 = 1$

$\mathbf{v}_0 = \mathbf{w}_0 = \mathbf{0}$

for $n = 1, 2, 3, \ldots$ **do**

- wähle γ_n und β_n mit $\gamma_n \beta_n = \widetilde{\mathbf{w}}_n^T \widetilde{\mathbf{v}}_n$ Gl. (3.17)

  $\mathbf{v}_n = \dfrac{1}{\gamma_n} \widetilde{\mathbf{v}}_n$ Gl. (3.15)

  $\mathbf{w}_n = \dfrac{1}{\beta_n} \widetilde{\mathbf{w}}_n$ Gl. (3.16)

- $\alpha_n = \mathbf{w}_n^T \mathbf{A} \mathbf{v}_n$ Gl. (3.12)

  $\widetilde{\mathbf{v}}_{n+1} = \mathbf{A}\mathbf{v}_n - \alpha_n \mathbf{v}_n - \beta_n \mathbf{v}_{n-1}$ Gl. (3.13)

  $\widetilde{\mathbf{w}}_{n+1} = \mathbf{A}^T \mathbf{w}_n - \alpha_n \mathbf{w}_n - \gamma_n \mathbf{w}_{n-1}$ Gl. (3.14)

enddo

Algorithmus 3.1 wurde aus den Gleichungen (3.6)–(3.8) hergeleitet. Umgekehrt kann auch gezeigt werden, daß die hier erzeugten Lanczos–Vektoren diesen Eigenschaften genügen, wie z.B. in Cullum und Willoughby [15] für eine spezielle Wahl der Koeffizienten γ_n und β_n gezeigt wird. Ein analoger Beweis für eine neue Version des unsymmetrischen Lanczos–Prozesses, der die Biorthogonalität der generierten Lanczos–Vektoren demonstriert, wird im folgenden Abschnitt angegeben.

In exakter Arithmetik wird Algorithmus 3.1 beendet, falls für einen Index n die Gleichung $\widetilde{\mathbf{w}}_n^T \widetilde{\mathbf{v}}_n = 0$ erfüllt ist. Ist einer der beiden Vektoren identisch Null, so spricht man von einem regulären Abbruch des Verfahrens. Problematisch ist der Fall $\widetilde{\mathbf{w}}_n^T \widetilde{\mathbf{v}}_n = 0$ und keiner der beiden Faktoren verschwindet. Darüber hinaus kann dieses Produkt auch sehr klein werden und in der Praxis zu Rundungsfehlern führen. Diese beiden unerwünschten Fälle können mit Hilfe sogenannter *Look-Ahead*-Techniken [61, 29, 60, 6, 78, 45] überwunden werden. Gutknecht gibt in seinen Arbeiten [37, 38] eine theoretische Betrachtung dieser Abbrüche und beschreibt, wie sie behandelt werden können. Da diese Problematik jedoch nicht Gegenstand dieser Arbeit ist, werden Algorithmus 3.1 und alle weiteren beendet, falls solche Abbrüche auftreten.

Die beiden Matrix-Vektor-Produkte $\mathbf{A}\mathbf{v}_n$ und $\mathbf{A}^T\mathbf{w}_n$ in Algorithmus 3.1 führen auf einem Parallelrechner mit verteiltem Speicher bei dünnbesetzten Matrizen i.allg. nur zur Synchronisation von benachbarten Prozessoren. Globale Synchronisationspunkte treten lediglich bei der Berechnung der Koeffizienten γ_n und β_n in (3.17) und des Koeffizienten α_n in (3.12) auf. Allerdings besteht in dieser Formulierung eine direkte Datenabhängigkeit zwischen den Größen γ_n, β_n und α_n, so daß α_n nicht ohne Kenntnis von γ_n und β_n bestimmt werden kann. Da aber $\alpha_n = \mathbf{w}_n^T\mathbf{A}\mathbf{v}_n = \frac{\widetilde{\mathbf{w}}_n^T\mathbf{A}\widetilde{\mathbf{v}}_n}{\gamma_n\beta_n}$ ist, können die beiden Skalarprodukte $\widetilde{\mathbf{w}}_n^T\widetilde{\mathbf{v}}_n$ und $\widetilde{\mathbf{w}}_n^T\mathbf{A}\widetilde{\mathbf{v}}_n$ gleichzeitig mit einem globalen Synchronisationspunkt berechnet werden. Um ein drittes Matrix-Vektor-Produkt zu vermeiden, wird in Gleichung (3.13) der Term $\mathbf{A}\mathbf{v}_n$ durch $\frac{1}{\gamma_n}\mathbf{A}\widetilde{\mathbf{v}}_n$ ersetzt. Auf diese Weise erhält man den klassischen unsymmetrischen Lanczos–Prozeß mit nur einem globalen Synchronisationspunkt, siehe Algorithmus 3.2. Darüber hinaus sind auch die Matrix-Vektor-Produkte $\mathbf{A}\widetilde{\mathbf{v}}_{n+1}$ und $\mathbf{A}^T\mathbf{w}_n$ voneinander unabhängig und können gleichzeitig berechnet werden. Bei unsymmetrischen Matrizen mit symmetrischer Struktur der Nichtnullelemente kann so beispielsweise Kommunikationszeit auf einem Parallelrechner eingespart werden. Ersetzt man auch $\mathbf{A}^T\mathbf{w}_n$ durch $\frac{1}{\beta_n}\mathbf{A}^T\widetilde{\mathbf{w}}_n$, ergibt Algorithmus 3.2 den Algorithmus 2.3 aus [51]. Weitere Bemerkungen zum klassischen unsymmetrischen Lanczos–Prozeß finden sich in [9].

Algorithmus 3.2 Klassischer Lanczos–Prozeß mit *einem* globalen Synchronisationspunkt

Eingabe: $\mathbf{A}, \widetilde{\mathbf{v}}_1$ und $\widetilde{\mathbf{w}}_1$ mit $\widetilde{\mathbf{w}}_1^T\widetilde{\mathbf{v}}_1 \neq 0$

$\mathbf{v}_0 = \mathbf{w}_0 = \mathbf{0}$

for $n = 1, 2, 3, \ldots$ **do**

- wähle γ_n und β_n mit $\gamma_n\beta_n = \widetilde{\mathbf{w}}_n^T\widetilde{\mathbf{v}}_n$ Gl. (3.17)
- $\alpha_n = \dfrac{\widetilde{\mathbf{w}}_n^T\mathbf{A}\widetilde{\mathbf{v}}_n}{\beta_n\gamma_n}$ Gl. (3.12)

$$\mathbf{v}_n = \frac{1}{\gamma_n}\widetilde{\mathbf{v}}_n \qquad \text{Gl. (3.15)}$$

$$\mathbf{w}_n = \frac{1}{\beta_n}\widetilde{\mathbf{w}}_n \qquad \text{Gl. (3.16)}$$

$$\widetilde{\mathbf{v}}_{n+1} = \frac{1}{\gamma_n}\mathbf{A}\widetilde{\mathbf{v}}_n - \alpha_n\mathbf{v}_n - \beta_n\mathbf{v}_{n-1} \qquad \text{Gl. (3.13)}$$

$$\widetilde{\mathbf{w}}_{n+1} = \mathbf{A}^T\mathbf{w}_n - \alpha_n\mathbf{w}_n - \gamma_n\mathbf{w}_{n-1} \qquad \text{Gl. (3.14)}$$

enddo

Mit der Bedingung (3.17) können die Koeffizienten γ_n und β_n gemäß den Gleichungen (3.15) und (3.16) zur Skalierung der Lanczos–Vektoren $\mathbf{v}_n$ und $\mathbf{w}_n$ verwendet werden. Dabei ergeben sich beliebig viele Möglichkeiten, von denen im folgenden drei diskutiert werden sollen. Einige Autoren [35, 48, 20] normieren eine der beiden

Folgen von Lanczos–Vektoren, hier $\mathbf{v}_n$, durch die Wahl

$$\gamma_n = \|\widetilde{\mathbf{v}}_n\|_2 \quad \text{und} \quad \beta_n = \frac{1}{\gamma_n}\widetilde{\mathbf{w}}_n^T\widetilde{\mathbf{v}}_n \ .$$

Sollen jedoch beide Folgen gleich behandelt werden, vergleiche [2, 51, 61], werden die Koeffizienten durch

$$\gamma_n = \sqrt{|\widetilde{\mathbf{w}}_n^T\widetilde{\mathbf{v}}_n|} \quad \text{und} \quad \beta_n = \frac{1}{\gamma_n}\widetilde{\mathbf{w}}_n^T\widetilde{\mathbf{v}}_n$$

bestimmt. Eine dritte Möglichkeit ergibt sich schließlich aus der Forderung, daß die erzeugte Tridiagolalmatrix $\mathbf{T}$ symmetrisch sein soll. Dies wird durch die Wahl

$$\gamma_n = \beta_n = \sqrt{\widetilde{\mathbf{w}}_n^T\widetilde{\mathbf{v}}_n}$$

erfüllt. Dabei muß jedoch beachtet werden, daß sich der Definitionsbereich von $\mathbb{R}$ auf $\mathbb{C}$ erweitert, d.h. die Matrix $\mathbf{T}$ und die beiden Folgen von Lanczos–Vektoren dann komplexwertig werden können. Da die Eigenwerte der iterativ erzeugten Matrix $\mathbf{T}$ die Eigenwerte von $\mathbf{A}$ approximieren, können dann bei Eigenwertproblemen durch diese Wahl spezielle Algorithmen zur Lösung symmetrischer Tridiagonalmatrizen eingesetzt werden [15].

In der Praxis jedoch sollten aus Gründen der numerischen Stabilität beide Folgen von Lanczos–Vektoren normiert werden [74, 61, 28, 29], d.h.

$$\|\mathbf{v}_n\|_2 = \|\mathbf{w}_n\|_2 = 1 \quad \text{für } n \in \mathbb{N}\,.$$

Dies wäre mit $\gamma_n = \|\widetilde{\mathbf{v}}_n\|_2$ und $\beta_n = \|\widetilde{\mathbf{w}}_n\|_2$ zu erreichen, vergleiche (3.15) und (3.16), wenn nicht Gleichung (3.17) die Normierung nur einer der beiden Folgen zulassen würde. Dieses Problem kann aber gelöst werden, indem man einen weiteren Freiheitsgrad in Form einer diagonalen Matrix $\mathbf{D}$ bereitstellt, der dazu dient, die zweite Folge zu normieren. Geht man also wieder von der Ähnlichkeitstransformation (3.5) aus und bezeichnet nun mit $\mathbf{W}$ die skalierte Matrix $\mathbf{V}^{-T}\mathbf{D}$ anstelle von $\mathbf{V}^{-T}$, so erhält man die zu (3.6)–(3.8) korrespondierenden Gleichungen

$$\mathbf{W}^T\mathbf{V} = \mathbf{D} \tag{3.18}$$

$$\mathbf{A}\mathbf{V} = \mathbf{V}\mathbf{T} \tag{3.19}$$

$$\mathbf{A}^T\mathbf{W} = \mathbf{W}\mathbf{D}^{-1}\mathbf{T}^T\mathbf{D}\,. \tag{3.20}$$

Hieraus läßt sich nun der auf Drei-Term-Rekursionen basierende klassische unsymmetrische Lanczos–Prozeß herleiten, in dem beide Folgen von Lanczos–Vektoren normiert werden können, vergleiche Abschnitt 2.2 von [31]. Der Ansatz (3.18)–(3.20) bildet die Grundlage des nächsten Abschnittes.

3.2 Der Zwei-Term-Lanczos–Prozeß

Der klassische unsymmetrische Lanczos–Prozeß aus dem vorangegangenen Abschnitt erzeugte die Lanczos–Vektoren mit der Hilfe von Drei-Term-Rekursionen. In diesem Abschnitt führt nun eine zusätzliche Voraussetzung an die Tridiagonalmatrix $\mathbf{T}$ auf gekoppelte Zwei-Term-Rekursionen. Obwohl Lanczos [53] schon 1952 eine ähnliche Technik verwendete, geriet die Idee in Vergessenheit und wurde erst 1993 von Freund et al. [29] zur Verbesserung der numerischen Stabilität wieder aufgegriffen. Wie schon erläutert, normieren die Autoren beide Folgen von Lanczos–Vektoren, um Instabilitäten zu vermeiden. Der von Freund et al. entwickelte Algorithmus benötigt allerdings in jeder Iteration drei globale Synchronisationspunkte, die mit der oben angegebenen Technik nicht auf einen reduziert werden können. Daher wird nun ein Ansatz beschrieben, der es ermöglicht, die numerischen und parallelen Aspekte zu verbinden. Das Ergebnis ist eine neue Version des unsymmetrischen gekoppelten Zwei-Term-Lanczos–Prozesses, der es erlaubt, beide Folgen zu normieren, und der darüber hinaus nur einen globalen Synchronisationspunkt in jedem Iterationsschritt benötigt. Diese Fassung soll im folgenden vorgestellt werden, vergleiche auch [9].

Dazu wird die Existenz einer LU-Zerlegung der Tridiagonalmatrix

$$\mathbf{T} = \mathbf{L}\mathbf{U} \tag{3.21}$$

angenommen, deren Faktoren $\mathbf{L} \in \mathbb{R}^{N\times N}$ und $\mathbf{U} \in \mathbb{R}^{N\times N}$ die bidiagonale Struktur

$$\mathbf{L} = \begin{pmatrix} \tau_1 & & & \\ \gamma_2 & \tau_2 & & \\ & \ddots & \ddots & \\ & & \gamma_N & \tau_N \end{pmatrix} \quad \text{und} \quad \mathbf{U} = \begin{pmatrix} 1 & \mu_2 & & \\ & 1 & \ddots & \\ & & \ddots & \mu_N \\ & & & 1 \end{pmatrix} \tag{3.22}$$

aufweisen. Mit der Einführung der Matrizen $\mathbf{P} = \mathbf{V}\mathbf{U}^{-1}$ und $\widetilde{\mathbf{Q}} = \mathbf{W}\mathbf{D}^{-1}\mathbf{U}^T$ können die Gleichungen (3.18)–(3.20), die den klassischen Lanczos–Algorithmus mit der Normierung beider Folgen beschreiben, mit Hilfe von (3.21) in

$$\mathbf{W}^T\mathbf{V} = \mathbf{D} \tag{3.23}$$
$$\mathbf{V} = \mathbf{P}\mathbf{U} \tag{3.24}$$
$$\mathbf{A}^T\mathbf{W} = \widetilde{\mathbf{Q}}\mathbf{L}^T\mathbf{D} \tag{3.25}$$
$$\mathbf{A}\mathbf{P} = \mathbf{V}\mathbf{L} \tag{3.26}$$
$$\widetilde{\mathbf{Q}} = \mathbf{W}\mathbf{D}^{-1}\mathbf{U}^T \tag{3.27}$$

überführt werden. Bezeichnet man mit $\mathbf{p}_1, \ldots, \mathbf{p}_N$ und $\widetilde{\mathbf{q}}_1, \ldots, \widetilde{\mathbf{q}}_N$ die entsprechenden Spalten der Matrizen $\mathbf{P}$ und $\widetilde{\mathbf{Q}}$ und ist $\mathbf{D} = \operatorname{diag}(\delta_1, \ldots, \delta_N)$, so sind die n-ten

Spalten von (3.24)–(3.27) durch

$$\mathbf{v}_n = \mathbf{p}_n + \mu_n \mathbf{p}_{n-1} \tag{3.28}$$
$$\mathbf{A}^T \mathbf{w}_n = \tau_n \delta_n \widetilde{\mathbf{q}}_n + \gamma_n \delta_n \widetilde{\mathbf{q}}_{n-1} \tag{3.29}$$
$$\mathbf{A}\mathbf{p}_n = \gamma_{n+1} \mathbf{v}_{n+1} + \tau_n \mathbf{v}_n \tag{3.30}$$
$$\widetilde{\mathbf{q}}_n = \frac{\mu_{n+1}}{\delta_{n+1}} \mathbf{w}_{n+1} + \frac{1}{\delta_n} \mathbf{w}_n \tag{3.31}$$

gegeben, wobei $\mathbf{p}_0 = \widetilde{\mathbf{q}}_0 = \mathbf{0}$ zu setzen sind. Mit der Definition $\mathbf{q}_n = \tau_n \delta_n \widetilde{\mathbf{q}}_n$ erhält man eine zu (3.29) und (3.31) äquivalente Form durch

$$\begin{aligned} \mathbf{q}_n &= \mathbf{A}^T \mathbf{w}_n - \frac{\gamma_n \delta_n}{\tau_{n-1} \delta_{n-1}} \mathbf{q}_{n-1} \\ \frac{\tau_n \delta_n \mu_{n+1}}{\delta_{n+1}} \mathbf{w}_{n+1} &= \mathbf{q}_n - \tau_n \mathbf{w}_n \quad , \end{aligned}$$

wobei $\mathbf{q}_0 = \mathbf{0}$. Die Koeffizienten δ_i können mit der Substitution

$$\xi_{n+1} = \frac{\tau_n \delta_n \mu_{n+1}}{\delta_{n+1}} \, , \tag{3.32}$$

die später zur Normierung der Lanczos–Vektoren dient, ersetzt werden. Zusammengefaßt ergeben sich auf diese Weise aus (3.28)–(3.31) die vier Gleichungen

$$\mathbf{p}_n = \mathbf{v}_n - \mu_n \mathbf{p}_{n-1} \tag{3.33}$$
$$\mathbf{q}_n = \mathbf{A}^T \mathbf{w}_n - \frac{\gamma_n \mu_n}{\xi_n} \mathbf{q}_{n-1} \tag{3.34}$$
$$\gamma_{n+1} \mathbf{v}_{n+1} = \mathbf{A}\mathbf{p}_n - \tau_n \mathbf{v}_n \tag{3.35}$$
$$\xi_{n+1} \mathbf{w}_{n+1} = \mathbf{q}_n - \tau_n \mathbf{w}_n \tag{3.36}$$

mit $\mathbf{p}_0 = \mathbf{q}_0 = \mathbf{0}$, die die Grundlage des entstehenden Algorithmus bilden. Die Struktur dieser Gleichungen erklärt den Begriff der *gekoppelten Zwei-Term-Rekursion*. Nimmt man an, daß die Vektoren $\mathbf{p}_{n-1}$, $\mathbf{q}_{n-1}$, $\mathbf{v}_n$ und $\mathbf{w}_n$ und die Koeffizienten μ_n, γ_n, ξ_n und τ_n aus dem vorangegangenen Iterationsschritt bekannt sind, so werden nun deren Nachfolger bestimmt. Zunächst werden die Vektoren $\mathbf{p}_n$ und $\mathbf{q}_n$ mit (3.33) und (3.34) aktualisiert. Darüber hinaus können auch die Vektoren

$$\widetilde{\mathbf{v}}_{n+1} = \mathbf{A}\mathbf{p}_n - \tau_n \mathbf{v}_n \tag{3.37}$$
$$\widetilde{\mathbf{w}}_{n+1} = \mathbf{q}_n - \tau_n \mathbf{w}_n \tag{3.38}$$

und die Koeffizienten

$$\gamma_{n+1} = \|\widetilde{\mathbf{v}}_{n+1}\|_2 \tag{3.39}$$
$$\xi_{n+1} = \|\widetilde{\mathbf{w}}_{n+1}\|_2 \tag{3.40}$$

bestimmt werden. Ein Vergleich von (3.35) mit (3.36) liefert die normierten Lanczos–Vektoren

$$\mathbf{v}_{n+1} = \frac{1}{\gamma_{n+1}}\widetilde{\mathbf{v}}_{n+1} \quad \text{und} \quad \mathbf{w}_{n+1} = \frac{1}{\xi_{n+1}}\widetilde{\mathbf{w}}_{n+1} \,. \tag{3.41}$$

Mit der Definition

$$\varrho_{n+1} = \widetilde{\mathbf{w}}_{n+1}^T \widetilde{\mathbf{v}}_{n+1} \tag{3.42}$$

und der Biorthogonalitätsbeziehung (3.23) folgt

$$\varrho_{n+1} = \gamma_{n+1}\xi_{n+1}\delta_{n+1} \quad . \tag{3.43}$$

Somit ergibt sich aus (3.32) der Koeffizient

$$\mu_{n+1} = \frac{\gamma_n \xi_n \varrho_{n+1}}{\gamma_{n+1}\tau_n \varrho_n} \quad . \tag{3.44}$$

Man beachte, daß alle Operationen, die zu einem globalen Synchronisationspunkt führen, d.h. (3.39), (3.40) und (3.42), voneinander unabhängig sind und deshalb mit einem einzigen globalen Synchronisationspunkt berechnet werden können. Es bleibt noch zu überprüfen, daß auch die Berechnung des Koeffizienten τ_{n+1} keinen weiteren Synchronisationspunkt beansprucht. Zunächst ergibt die Multiplikation von (3.35) mit $\mathbf{w}_{n+1}^T$ den Wert

$$\tau_{n+1} = \frac{\mathbf{w}_{n+1}^T \mathbf{A}\mathbf{p}_{n+1}}{\mathbf{w}_{n+1}^T \mathbf{v}_{n+1}} \quad . \tag{3.45}$$

Da aber μ_{n+1}, und damit ϱ_{n+1}, zur Aktualisierung von $\mathbf{p}_{n+1}$ benötigt werden, kann das Skalarprodukt $\mathbf{w}_{n+1}^T \mathbf{A}\mathbf{p}_{n+1}$ nicht gleichzeitig mit den anderen Produkten ermittelt werden. Ersetzt man jedoch $\mathbf{p}_{n+1}$ durch (3.33), folgt

$$\tau_{n+1} = \frac{\mathbf{w}_{n+1}^T \mathbf{A}\mathbf{v}_{n+1}}{\mathbf{w}_{n+1}^T \mathbf{v}_{n+1}} - \mu_{n+1}\frac{\mathbf{w}_{n+1}^T \mathbf{A}\mathbf{p}_n}{\mathbf{w}_{n+1}^T \mathbf{v}_{n+1}} \quad .$$

Die Erweiterung des ersten Bruches mit $\xi_{n+1}\gamma_{n+1}$ und die Substitution von $\mathbf{A}\mathbf{p}_n$ durch Gleichung (3.35) im zweiten Term führt mit der Biorthogonalitätsbeziehung (3.23) zu

$$\tau_{n+1} = \frac{\widetilde{\mathbf{w}}_{n+1}^T \mathbf{A}\widetilde{\mathbf{v}}_{n+1}}{\widetilde{\mathbf{w}}_{n+1}^T \widetilde{\mathbf{v}}_{n+1}} - \gamma_{n+1}\mu_{n+1} \quad .$$

Die Definition

$$\varepsilon_{n+1} = \widetilde{\mathbf{w}}_{n+1}^T \mathbf{A}\widetilde{\mathbf{v}}_{n+1} \tag{3.46}$$

liefert schließlich

$$\tau_{n+1} = \frac{\varepsilon_{n+1}}{\varrho_{n+1}} - \gamma_{n+1}\mu_{n+1} \quad . \tag{3.47}$$

Das zusätzliche Skalarprodukt $\widetilde{\mathbf{w}}_{n+1}^T \mathbf{A} \widetilde{\mathbf{v}}_{n+1}$ ist jetzt unabhängig von den anderen Produkten, so daß (3.39), (3.40), (3.42) und (3.46) simultan mit einem globalen Synchronisationspunkt berechnet werden können. In Algorithmus 3.3 sind die bereitgestellten Formeln zusammengefaßt.

Dabei können die Werte $\delta_n = \mathbf{w}_n^T \mathbf{v}_n$ der diagonalen Matrix $\mathbf{D}$ durch die Anweisung $\delta_n = \varrho_n / (\gamma_n \xi_n)$, vergleiche (3.43), eingefügt werden. Erwähnenswert ist auch, daß sich die Matrix-Vektor-Produkte $\mathbf{A}^T \widetilde{\mathbf{w}}_{n+1}$ und $\mathbf{A}\mathbf{p}_n$, ähnlich wie im vorangegangenen Abschnitt, gleichzeitig berechnen lassen.

Algorithmus 3.3 Gekoppelter Zwei-Term-Algorithmus mit *einem* globalen Synchronisationspunkt

---

Eingabe: $\mathbf{A}, \widetilde{\mathbf{v}}_1$ und $\widetilde{\mathbf{w}}_1$ mit $\widetilde{\mathbf{w}}_1^T \widetilde{\mathbf{v}}_1 \neq 0$

$\mathbf{p}_0 = \mathbf{q}_0 = \mathbf{0}$

$\gamma_1 = \|\widetilde{\mathbf{v}}_1\|_2\,, \quad \xi_1 = \|\widetilde{\mathbf{w}}_1\|_2\,, \quad \varrho_1 = \widetilde{\mathbf{w}}_1^T \widetilde{\mathbf{v}}_1\,, \quad \varepsilon_1 = \left(\mathbf{A}^T \widetilde{\mathbf{w}}_1\right)^T \widetilde{\mathbf{v}}_1\,, \quad \mu_1 = 0\,, \quad \tau_1 = \dfrac{\varepsilon_1}{\varrho_1}$

for $n = 1, 2, 3, \ldots$ **do**

$$\mathbf{v}_n = \frac{1}{\gamma_n} \widetilde{\mathbf{v}}_n \qquad \text{Gl. (3.41)}$$

$$\mathbf{w}_n = \frac{1}{\xi_n} \widetilde{\mathbf{w}}_n \qquad \text{Gl. (3.41)}$$

$$\mathbf{p}_n = \mathbf{v}_n - \mu_n \mathbf{p}_{n-1} \qquad \text{Gl. (3.33)}$$

$$\mathbf{q}_n = \frac{1}{\xi_n} \mathbf{A}^T \widetilde{\mathbf{w}}_n - \frac{\gamma_n \mu_n}{\xi_n} \mathbf{q}_{n-1} \qquad \text{Gl. (3.34)}$$

$$\widetilde{\mathbf{w}}_{n+1} = \mathbf{q}_n - \tau_n \mathbf{w}_n \qquad \text{Gl. (3.38)}$$

$$\widetilde{\mathbf{v}}_{n+1} = \mathbf{A}\mathbf{p}_n - \tau_n \mathbf{v}_n \qquad \text{Gl. (3.37)}$$

- $\gamma_{n+1} = \|\widetilde{\mathbf{v}}_{n+1}\|_2 \qquad$ Gl. (3.39)
- $\xi_{n+1} = \|\widetilde{\mathbf{w}}_{n+1}\|_2 \qquad$ Gl. (3.40)
- $\varrho_{n+1} = \widetilde{\mathbf{w}}_{n+1}^T \widetilde{\mathbf{v}}_{n+1} \qquad$ Gl. (3.42)
- $\varepsilon_{n+1} = \left(\mathbf{A}^T \widetilde{\mathbf{w}}_{n+1}\right)^T \widetilde{\mathbf{v}}_{n+1} \qquad$ Gl. (3.46)

$$\mu_{n+1} = \frac{\gamma_n \xi_n \varrho_{n+1}}{\gamma_{n+1} \tau_n \varrho_n} \qquad \text{Gl. (3.44)}$$

$$\tau_{n+1} = \frac{\varepsilon_{n+1}}{\varrho_{n+1}} - \gamma_{n+1} \mu_{n+1} \qquad \text{Gl. (3.47)}$$

enddo

Die Herleitung basierte auf den Gleichungen (3.24)–(3.27) unter Verwendung der Biorthogonalitätseigenschaft (3.23). Umgekehrt läßt sich aber auch zeigen, daß die durch Algorithmus 3.3 erzeugten Lanczos–Vektoren $\mathbf{v}_n$ und $\mathbf{w}_n$ biorthogonal sind. Dies ergibt sich aus dem abschließenden Satz.

Satz 3.2.1. *Falls alle Anweisungen von Algorithmus 3.3 für $n = 1, 2, \ldots, m$ wohldefiniert sind, erfüllen die $\widetilde{\mathbf{v}}$-Vektoren und die $\widetilde{\mathbf{w}}$-Vektoren die Eigenschaft*

$$\widetilde{\mathbf{w}}_i^T \widetilde{\mathbf{v}}_j = \begin{cases} 0, & \text{falls } i \neq j\,, \\ \varrho_i \neq 0, & \text{falls } i = j\,, \end{cases} \qquad 1 \leq i, j \leq m+1\,. \tag{3.48}$$

Beweis. Mit einer Induktion über n wird bewiesen, daß die Biorthogonalität (3.48) für $1 \leq i, j \leq m+1$ gültig ist. Die Induktionsverankerung wird durch die Initialisierungsphase gewährleistet und die Behauptung $\varrho_i \neq 0$ folgt aus der Wohldefiniertheit. Für $n \geq 1$ liefern die Anweisungen, die mit Gl.(3.38) und Gl.(3.34) numeriert sind, mit Hilfe der Identität $\mathbf{q}_{n-1} = \widetilde{\mathbf{w}}_n + \frac{\tau_{n-1}}{\xi_{n-1}} \widetilde{\mathbf{w}}_{n-1}$, vergleiche (3.38), die Rekursion

$$\widetilde{\mathbf{w}}_{n+1} = \frac{1}{\xi_n} \mathbf{A}^T \widetilde{\mathbf{w}}_n - \frac{\gamma_n \mu_n + \tau_n}{\xi_n} \widetilde{\mathbf{w}}_n - \frac{\gamma_n \mu_n \tau_{n-1}}{\xi_n \xi_{n-1}} \widetilde{\mathbf{w}}_{n-1}\ , \tag{3.49}$$

wobei der letzte Term für $n = 1$ verschwindet. Eine analoge Rechnung, die die Anweisungen Gl.(3.37) und Gl.(3.33) verwendet, zeigt

$$\widetilde{\mathbf{v}}_{i+1} = \frac{1}{\gamma_i} \mathbf{A} \widetilde{\mathbf{v}}_i - \frac{\gamma_i \mu_i + \tau_i}{\gamma_i} \widetilde{\mathbf{v}}_i - \frac{\mu_i \tau_{i-1}}{\gamma_{i-1}} \widetilde{\mathbf{v}}_{i-1}\ , \qquad i = 1, \ldots, n\ .$$

Die Induktionsvoraussetzung impliziert

$$\widetilde{\mathbf{w}}_n^T \mathbf{A} \widetilde{\mathbf{v}}_i = \begin{cases} 0, & \text{falls } i = 1, \ldots, n-2\ , \\ \varrho_n \gamma_{n-1}\,, & \text{falls } i = n-1\ . \end{cases} \tag{3.50}$$

Aus (3.49) folgt zunächst

$$\widetilde{\mathbf{w}}_{n+1}^T \widetilde{\mathbf{v}}_i = \frac{1}{\xi_n} \widetilde{\mathbf{w}}_n^T \mathbf{A} \widetilde{\mathbf{v}}_i - \frac{\gamma_n \mu_n + \tau_n}{\xi_n} \widetilde{\mathbf{w}}_n^T \widetilde{\mathbf{v}}_i - \frac{\gamma_n \mu_n \tau_{n-1}}{\xi_n \xi_{n-1}} \widetilde{\mathbf{w}}_{n-1}^T \widetilde{\mathbf{v}}_i$$

und mit der Induktionsvoraussetzung und (3.50) ergibt sich schließlich

$$\widetilde{\mathbf{w}}_{n+1}^T \widetilde{\mathbf{v}}_i = \begin{cases} 0, & \text{falls } i = 1, \ldots, n-2\ , \\ \dfrac{\varrho_n \gamma_{n-1}}{\xi_n} - \dfrac{\gamma_n \mu_n \tau_{n-1}}{\xi_n \xi_{n-1}} \varrho_{n-1}\,, & \text{falls } i = n-1\ , \\ \dfrac{\varepsilon_n}{\xi_n} - \dfrac{\gamma_n \mu_n + \tau_n}{\xi_n} \varrho_n\,, & \text{falls } i = n\ . \end{cases}$$

Die Berechnung von μ_n und τ_n liefert die Behauptung in den Fällen $i = n-1$ und $i = n$. Ähnlich zeigt man $\widetilde{\mathbf{w}}_i^T \widetilde{\mathbf{v}}_{n+1} = 0$ für $i = 1, \ldots, n$. Der Wert $\widetilde{\mathbf{w}}_{n+1}^T \widetilde{\mathbf{v}}_{n+1} = \varrho_{n+1}$ ergibt sich aus der entsprechenden Anweisung von Algorithmus 3.3. □

Der symmetrische Fall

Ist die Matrix symmetrisch, d.h. gilt $\mathbf{A} = \mathbf{A}^T$, so können in Algorithmus 3.3 einige Anweisungen eingespart werden. Startet man diesen Prozeß mit der Wahl $\widetilde{\mathbf{v}}_1 = \widetilde{\mathbf{w}}_1$, so ergibt sich aus den Gleichungen (3.33) – (3.36) induktiv die Übereinstimmung der Folgen $\mathbf{v}_n = \mathbf{w}_n$ und $\mathbf{q}_n = \mathbf{A}\mathbf{p}_n$. Also sind die erzeugten Lanczos Vektoren orthogonal, d.h.

$$\mathbf{v}_i^T \mathbf{v}_j = \begin{cases} 0, & \text{falls } i \neq j \ , \\ 1, & \text{falls } i = j \ . \end{cases} \tag{3.51}$$

Weiter impliziert $\mathbf{v}_n = \mathbf{w}_n$ die Gleichheit der Koeffizienten $\gamma_n = \xi_n$ bzw. $\gamma_n^2 = \varrho_n$, so daß (3.44) in vereinfachter Form durch $\mu_n = \frac{\gamma_n}{\tau_{n-1}}$ gegeben wird. Damit folgt aus (3.33)

$$\mathbf{p_n} = \mathbf{v}_n - \frac{\gamma_n}{\tau_{n-1}} \mathbf{p}_{n-1} \, . \tag{3.52}$$

Hieraus schließt man mit $\mathbf{p}_0 = \mathbf{0}$ induktiv $\mathbf{p}_n \in \mathbf{v}_n + \text{span}\{\mathbf{v}_1, \ldots, \mathbf{v}_{n-1}\}$, so daß sich aus (3.35) und der Orthogonalitätsbedingung (3.51) der Koeffizient

$$\tau_n = \mathbf{p}_n^T \mathbf{A} \mathbf{p}_n \tag{3.53}$$

ermitteln läßt. Multipliziert man Gleichung (3.37) mit $\widetilde{\mathbf{v}}_{n+1}^T$, wird γ_{n+1} durch die Formel

$$\gamma_{n+1}^2 = \widetilde{\mathbf{v}}_{n+1}^T \widetilde{\mathbf{v}}_{n+1} = \widetilde{\mathbf{v}}_{n+1}^T \mathbf{A} \mathbf{p}_n = (\mathbf{A}\mathbf{p}_n)^T \mathbf{A} \mathbf{p}_n - \tau_n \mathbf{v}_n^T \mathbf{A} \mathbf{p}_n$$

bestimmt. Da aber auch $\mathbf{v}_n^T \mathbf{A} \mathbf{p}_n = \tau_n$ gilt, vergleiche (3.35), folgt mit der Definition

$$\varepsilon_n = (\mathbf{A}\mathbf{p}_n)^T \mathbf{A} \mathbf{p}_n \tag{3.54}$$

die Identität

$$\gamma_{n+1} = \sqrt{\varepsilon_n - \tau_n^2} \, . \tag{3.55}$$

Auf diese Weise erhält man den symmetrischen Lanczos–Prozeß mit einem globalen Synchronisationspunkt, in dem die Folge der Lanczos–Vektoren normiert wird. In Algorithmus 3.4 sind die hergeleiteten Formeln zusammengestellt.

3.3 Lanczos–basierte Krylov–Teilraumverfahren

Der auf gekoppelten Zwei-Term-Rekursionen basierende unsymmetrische Lanczos–Prozeß reduziert eine Matrix $\mathbf{A}$ auf Tridiagonalgestalt $\mathbf{T}$ mit der Zerlegung $\mathbf{T} = \mathbf{L}\mathbf{U}$. Die durch diesen Algorithmus erzeugten Lanczos–Vektoren sollen nun als Basis des Krylov–Teilraumverfahrens zur Lösung des linearen Gleichungssystems (3.1), d.h.

$$\mathbf{A}\mathbf{x} = \mathbf{b} \qquad \text{mit } \mathbf{A} \in \mathbb{R}^{N \times N} \text{ und } \mathbf{x}, \mathbf{b} \in \mathbb{R}^N$$

Algorithmus 3.4 Symmetrischer gekoppelter Zwei-Term-Algorithmus mit *einem* globalen Synchronisationspunkt

Eingabe: $\mathbf{A}, \widetilde{\mathbf{v}}_1$ mit $\widetilde{\mathbf{v}}_1 \neq 0$

$\mathbf{p}_0 = \mathbf{0}$

$\gamma_1 = \|\widetilde{\mathbf{v}}_1\|_2\,, \quad \tau_0 = 1$

for $n = 1, 2, 3, \ldots$ **do**

$\mathbf{v}_n = \dfrac{1}{\gamma_n}\widetilde{\mathbf{v}}_n$ Gl. (3.41)

$\mathbf{p}_n = \mathbf{v}_n - \dfrac{\gamma_n}{\tau_{n-1}}\mathbf{p}_{n-1}$ Gl. (3.52)

- $\tau_n = \mathbf{p}_n^T \mathbf{A}\mathbf{p}_n$ Gl. (3.53)
- $\varepsilon_n = (\mathbf{A}\mathbf{p}_n)^T \mathbf{A}\mathbf{p}_n$ Gl. (3.54)

$\gamma_{n+1} = \sqrt{\varepsilon_n - \tau_n^2}$ Gl. (3.55)

$\widetilde{\mathbf{v}}_{n+1} = \mathbf{A}\mathbf{p}_n - \tau_n \mathbf{v}_n$ Gl. (3.37)

enddo

verwendet werden. Bezeichnen wie früher $\mathbf{x_0}$ eine Startlösung für die Gleichung (3.1) und $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x_0}$ den zugehörigen Residuumsvektor, so erzeugt der unsymmetrische Lanczos–Prozeß zu den Startvektoren

$$\mathbf{v}_1 = \frac{1}{\gamma_1}\mathbf{r}_0 \quad \text{und} \quad \mathbf{w}_1 = \frac{1}{\gamma_1}\mathbf{r}_0 \quad \text{mit} \quad \gamma_1 = \|\mathbf{r}_0\|_2 \neq 0 \tag{3.56}$$

die Basis der entsprechenden Krylov–Teilräume

$$\mathcal{K}_n(\mathbf{r}_0, \mathbf{A}) = \text{span}\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$$

und

$$\mathcal{K}_n(\mathbf{r}_0, \mathbf{A}^T) = \text{span}\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}\,,$$

von denen zunächst nur der erste explizit benötigt wird. Werden die n Lanczos–Vektoren $\mathbf{v}_1, \ldots, \mathbf{v}_n$ in der Matrix $\mathbf{V}_n = (\mathbf{v}_1, \ldots, \mathbf{v}_n) \in \mathbb{R}^{N \times n}$ als Spalten verwendet, so kann die noch zu bestimmende Approximierte $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, \mathbf{A})$ durch die Wahl eines $\mathbf{z}_n \in \mathbb{R}^n$ in der Form

$$\mathbf{x}_n = \mathbf{x}_0 + \mathbf{V}_n \mathbf{z}_n \tag{3.57}$$

beschrieben werden. Neben den Lanczos–Vektoren generiert der Lanczos–Prozeß im n-ten Iterationsschritt auch die Teilmatrizen

$$\mathbf{L}_n = \begin{pmatrix} \tau_1 & & \\ \gamma_2 & \ddots & \\ & \ddots & \tau_n \\ & & \gamma_{n+1} \end{pmatrix} \in \mathbb{R}^{(n+1)\times n}, \quad \mathbf{U}_n = \begin{pmatrix} 1 & \mu_2 & & \\ & 1 & \ddots & \\ & & \ddots & \mu_n \\ & & & 1 \end{pmatrix} \in \mathbb{R}^{n\times n}$$

von $\mathbf{L}$ und $\mathbf{U}$ aus (3.22) und die Matrix $\mathbf{P}_n = (\mathbf{p}_1, \ldots, \mathbf{p}_n) \in \mathbb{R}^{N\times n}$, die mit $\mathbf{V}_n$ durch

$$\mathbf{P}_n = \mathbf{V}_n \mathbf{U}_n^{-1} \tag{3.58}$$

$$\mathbf{A}\mathbf{P}_n = \mathbf{V}_{n+1} \mathbf{L}_n \,, \tag{3.59}$$

in Beziehung steht, vergleiche (3.24) und (3.26). Mit der Definition

$$\mathbf{y}_n = \mathbf{U}_n \mathbf{z}_n$$

wird $\mathbf{x}_n$ aus (3.57) wegen (3.58) auch äquivalent durch

$$\mathbf{x}_n = \mathbf{x}_0 + \mathbf{P}_n \mathbf{y}_n \tag{3.60}$$

gegeben. Für den zugehörigen Residuumsvektor gilt also

$$\mathbf{r}_n \;=\; \mathbf{b} - \mathbf{A}\mathbf{x}_n \;=\; \mathbf{r}_0 - \mathbf{A}\mathbf{P}_n \mathbf{y}_n \,.$$

Aus (3.56) und (3.59) folgt schließlich eine Faktorisierung des Residuums

$$\mathbf{r}_n = \mathbf{V}_{n+1} \left(\gamma_1 \mathbf{e}_1^{(n+1)} - \mathbf{L}_n \mathbf{y}_n \right) \,. \tag{3.61}$$

Da die Minimierung von $\mathbf{r}_n$ sehr aufwendig ist, sollen nun verschiedene Strategien zur Wahl von $\mathbf{y}_n$ und damit von $\mathbf{x}_n$ angegeben werden, die mit weniger Aufwand durchführbar sind. Bezeichnet

$$\mathbf{s}_n = \gamma_1 \, \mathbf{e}_1^{(n+1)} - \mathbf{L}_n \, \mathbf{y}_n \tag{3.62}$$

den rechten Faktor aus (3.61) und s_n^i für $i = 1, \ldots, n+1$ dessen Komponenten, so gilt zunächst

$$\mathbf{r}_n = \mathbf{V}_{n+1} \, \mathbf{s}_n \tag{3.63}$$

und wegen $\|\mathbf{v}_i\|_2 = 1$ für $i = 1, \ldots, n+1$ ist

$$\begin{aligned} \|\mathbf{r}_n\|_2 \quad &= \quad \|\mathbf{V}_{n+1} \, \mathbf{s}_n\|_2 \quad &= \quad \Big\| \sum_{i=1}^{n+1} s_n^i \, \mathbf{v}_i \Big\|_2 \\ &\leq \quad \sum_{i=1}^{n+1} \|s_n^i \, \mathbf{v}_i\|_2 \quad &= \quad \sum_{i=1}^{n+1} |s_n^i| \, \|\mathbf{v}_i\|_2 \\ &= \quad \sum_{i=1}^{n+1} |s_n^i| \quad &= \quad \|\mathbf{s}_n\|_1 \,. \end{aligned} \tag{3.64}$$

Hieraus ergibt sich eine Möglichkeit zur Wahl des $\mathbf{y}_n$, die wie folgt formuliert werden kann.

Strategie 1. *Bestimme $\mathbf{y}_n$ so, daß die jeweils ersten n Einträge von $\mathbf{s}_n$ verschwinden.*

Weiter liefert die Hölder–Ungleichung aus (3.64) die Abschätzung

$$\|\mathbf{r}_n\|_2 \;\leq\; (n+1)^{\frac{1}{q}}\,\|\mathbf{s}_n\|_p\,, \tag{3.65}$$

wobei $\frac{1}{q} = \frac{p-1}{p} \in [0;1)$, falls $1 \leq p < \infty$ und $\frac{1}{q} = 1$ im Fall $p = \infty$.

Das klassische Verfahren der quasi-minimalen Residuen [30, 31] bestimmt $\mathbf{y}_n$ derart, daß $\mathbf{s}_n$ in der l_2-Norm minimiert wird. Das Minimierungsproblem kann aber auch für jedes $1 \leq p \leq \infty$ betrachtet werden. Die Ungleichung (3.65) führt so zur folgenden Strategie zur Wahl von $\mathbf{y}_n$ und damit $\mathbf{x}_n$.

Strategie 2. *Wähle $\mathbf{y}_n$ so, daß $\mathbf{s}_n$ in der l_p-Norm minimal wird, d.h.*

$$\left\|\gamma_1\,\mathbf{e}_1^{(n+1)} - \mathbf{L}_n\,\mathbf{y}_n\right\|_p \quad = \quad \min_{\mathbf{y}\in\mathbb{R}^n}\left\|\gamma_1\,\mathbf{e}_1^{(n+1)} - \mathbf{L}_n\,\mathbf{y}\right\|_p\,. \tag{3.66}$$

Das auf dem Lanczos–Prozeß basierende Krylov–Teilraumverfahren ist in Algorithmus 3.5 abgebildet. Da die explizite Darstellung der Lanczos–Vektoren im folgenden nicht gebraucht wird, sind $\mathbf{v}_n$ und $\mathbf{w}_n$ gemäß (3.41) ersetzt und die entsprechende Anweisung aus Algorithmus 3.3 entfernt worden. Die Berechnung des Abbruchkriteriums, beispielsweise $\|\mathbf{r}_n\|_2$ kleiner als eine vom Benutzer vorgegebene Toleranz, benötigt einen weiteren globalen Synchronisationspunkt. Wird dieses Skalarprodukt, wie in Algorithmus 3.5 dargestellt, verzögert, benötigt der gesamte Algorithmus wie bisher nur einen globalen Synchronisationspunkt. Vorgreifend soll auch schon bemerkt werden, daß zur Berechnung von $\mathbf{x}_n$ und $\mathbf{r}_n$ keine zusätzliche Synchronisation gebraucht wird.

Nimmt τ_n oder γ_{n+1} in Algorithmus 3.5 im n-ten Iterationsschritt den Wert Null an, so bricht der Lanczos–Prozeß im nächsten Schritt ab. Der Fall $\gamma_{n+1} = 0$ bedeutet, daß in beiden Strategien $\mathbf{s}_n = \mathbf{0}$ und damit die exakte Lösung $\mathbf{x}_n = \mathbf{A}^{-1}\mathbf{b}$ gefunden wird. Ist jedoch $\tau_n = 0$ ergibt sich mit der ersten Strategie $\mathbf{y}_n = \begin{pmatrix}\mathbf{y}_{n-1}\\ 0\end{pmatrix}$ und damit $\mathbf{x}_n = \mathbf{x}_{n-1}$. In der zweiten Strategie läßt sich das $\mathbf{y}_n$ nicht bestimmen. Aus diesem Grund soll im folgenden

$$\tau_1,\ldots,\tau_n \neq 0 \quad \text{und} \quad \gamma_1,\ldots,\gamma_n \neq 0\,, \tag{3.67}$$

angenommen werden.

3.4 Das Verfahren der bikonjugierten Gradienten

Wegen der bidiagonalen Struktur der Matrix $\mathbf{L}_n$, wird $\mathbf{y}_n$ in der Strategie 1 durch

$$\mathbf{y}_n \;=\; (\eta_1,\ldots,\eta_n)^T \tag{3.68}$$

Algorithmus 3.5 Lanczos–basiertes Krylov–Teilraumverfahren mit *einem* globalen Synchronisationspunkt

Eingabe $\mathbf{A}, \mathbf{b}$ und $\mathbf{x}_0$

$\mathbf{p}_0 = \mathbf{q}_0 = \mathbf{0}\,,\ \widetilde{\mathbf{w}}_1 = \widetilde{\mathbf{v}}_1 = \mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$

$\gamma_0 = \xi_0 = 0\,,\ \tau_0 = \varrho_0 = 1\,,\ \gamma_1 = \|\mathbf{r}_0\|_2\,,\ \xi_1 = \gamma_1\,,\ \varrho_1 = \gamma_1^2\,,\ \varepsilon_1 = \left(\mathbf{A}^T\mathbf{r}_0\right)^T\mathbf{r}_0$

for $n = 1, 2, 3, \ldots$ **do**

$$\mu_n = \frac{\gamma_{n-1}\xi_{n-1}\varrho_n}{\gamma_n\tau_{n-1}\varrho_{n-1}}\,, \qquad \tau_n = \frac{\varepsilon_n}{\varrho_n} - \gamma_n\mu_n$$

$$\mathbf{p}_n = \frac{1}{\gamma_n}\widetilde{\mathbf{v}}_n - \mu_n\mathbf{p}_{n-1}\,, \quad \mathbf{q}_n = \frac{1}{\xi_n}\mathbf{A}^T\widetilde{\mathbf{w}}_n - \frac{\gamma_n\mu_n}{\xi_n}\mathbf{q}_{n-1}$$

$$\widetilde{\mathbf{w}}_{n+1} = \mathbf{q}_n - \frac{\tau_n}{\xi_n}\widetilde{\mathbf{w}}_n\,, \qquad \widetilde{\mathbf{v}}_{n+1} = \mathbf{A}\mathbf{p}_n - \frac{\tau_n}{\gamma_n}\widetilde{\mathbf{v}}_n$$

- **if** $\left(\|\mathbf{r}_{n-1}\|_2 < \text{Toleranz}\right)$ **then** STOP
- $\gamma_{n+1} = \|\widetilde{\mathbf{v}}_{n+1}\|_2 \qquad \xi_{n+1} = \|\widetilde{\mathbf{w}}_{n+1}\|_2$
- $\varrho_{n+1} = \widetilde{\mathbf{w}}_{n+1}^T\widetilde{\mathbf{v}}_{n+1} \qquad \varepsilon_{n+1} = \left(\mathbf{A}^T\widetilde{\mathbf{w}}_{n+1}\right)^T\widetilde{\mathbf{v}}_{n+1}$

  Anweisung zur Berechnung von $\mathbf{x}_n$ und $\mathbf{r}_n$

enddo

mit

$$\eta_i \;=\; -\frac{\gamma_i}{\tau_i}\,\eta_{i-1} \quad i = 2, \ldots, n\,, \quad \eta_1 = \frac{\gamma_1}{\tau_1}\,, \tag{3.69}$$

eindeutig gegeben. Darüber hinaus kann dieser Vektor, da die ersten $n-1$ Komponenten von $\mathbf{y}_n$ und $\mathbf{y}_{n-1}$ übereinstimmen, durch die Rekursion

$$\mathbf{y}_n \;=\; \begin{pmatrix}\mathbf{y}_{n-1}\\ 0\end{pmatrix} + \kappa_n\,\mathbf{e}_n^{(n)} \tag{3.70}$$

mit

$$\kappa_n \;=\; -\frac{\gamma_n}{\tau_n}\,\kappa_{n-1}\,, \quad \kappa_0 = -1\,, \tag{3.71}$$

dargestellt werden. Aus Gleichung (3.62) ergibt sich hieraus das zugehörige Residuum

$$\mathbf{s}_n \;=\; -\gamma_{n+1}\kappa_n\,\mathbf{e}_{n+1}^{(n+1)}\,. \tag{3.72}$$

Mit diesen Ergebnissen können nun die Iterierten $\mathbf{x}_n$ aus (3.60) und $\mathbf{r}_n$ aus (3.63) des Krylov–Teilraumverfahrens, Algorithmus 3.5, aktualisiert werden. Setzt man (3.70)

in (3.60) ein, so ist

$$\begin{aligned} \mathbf{x}_n &= \mathbf{x}_0 + \mathbf{P}_n \mathbf{y}_n \\ &= \mathbf{x}_0 + \mathbf{P}_{n-1} \mathbf{y}_{n-1} + \kappa_n \mathbf{p}_n \\ &= \mathbf{x}_{n-1} + \kappa_n \mathbf{p}_n \,. \end{aligned} \tag{3.73}$$

Aus (3.63) und (3.72) folgt in Verbindung mit (3.41)

$$\begin{aligned} \mathbf{r}_n &= \mathbf{V}_{n+1} \mathbf{s}_n \;=\; -\gamma_{n+1} \kappa_n \, \mathbf{v}_{n+1} \\ &= -\kappa_n \, \widetilde{\mathbf{v}}_{n+1} \,. \end{aligned} \tag{3.74}$$

Wegen $\|\mathbf{v}_{n+1}\|_2 = 1$ erhält man sogar

$$\|\mathbf{r}_n\|_2 \;=\; |\gamma_{n+1} \kappa_n| \,, \tag{3.75}$$

ohne ein weiteres Skalarprodukt auswerten zu müssen. Die Formeln sind in Algorithmus 3.6 zusammengefaßt.

Algorithmus 3.6 BCG zur Lösung von $\mathbf{A}\mathbf{x} = \mathbf{b}$

Eingabe: $\mathbf{A}, \mathbf{b}, \mathbf{x}_0$

Führe Initialisierung von Algorithmus 3.5 aus

$\kappa_0 = -1$

for $n = 1, 2, 3, \ldots$ **do**

Führe n-ten Iterationsschritt von Algorithmus 3.5 aus

$\kappa_n = -\dfrac{\gamma_n}{\tau_n} \kappa_{n-1}$ Gl. (3.71)

$\mathbf{x}_n = \mathbf{x}_{n-1} + \kappa_n \mathbf{p}_n$ Gl. (3.73)

$\|\mathbf{r}_n\|_2 = |\kappa_n \gamma_{n+1}|$ Gl. (3.75)

enddo

Dieses Krylov–Teilraumverfahren, das zunächst auf einer heuristischen Wahl des Vektors $\mathbf{y}_n$ basierte, stellt sich nun als eine Variante des Verfahrens der bikonjugierten Gradienten (BCG) heraus. Zum Nachweis von (3.4) bezeichne $\mathbf{W}_n = (\mathbf{w}_1, \ldots, \mathbf{w}_n) \in \mathbb{R}^{N \times n}$ die Matrix, deren Spalten aus den Lanczos–Vektoren $\mathbf{w}_1, \ldots, \mathbf{w}_n$ bestehen. Hiermit kann jedes $\mathbf{w} \in \mathcal{K}_n(\mathbf{r}_0, \mathbf{A}^T) = \operatorname{span}\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ durch ein $\mathbf{z} \in \mathbb{R}^n$ in der Form $\mathbf{w} = \mathbf{W}_n \mathbf{z}$ beschrieben werden, so daß

$$\mathbf{w}^T (\mathbf{b} - \mathbf{A}\mathbf{x}_n) = \mathbf{w}^T \mathbf{r}_n = \mathbf{z}^T \mathbf{W}_n^T \mathbf{r}_n \,.$$

Da $\mathbf{r}_n = -\gamma_{n+1} \kappa_n \mathbf{v}_{n+1}$, folgt aus der Biorthogonalitätsbeziehung der Vektoren $\mathbf{v}_i$ und $\mathbf{w}_j$ die Gleichung

$$\mathbf{w}^T (\mathbf{b} - \mathbf{A}\mathbf{x}_n) = -\gamma_{n+1} \kappa_n \mathbf{z}^T (\mathbf{W}_n^T \mathbf{v}_{n+1}) = 0 \,.$$

Eine Variante des Verfahrens der bikonjugierten Gradienten

In Abbildung 3.1, obere Kurve, ist ein typischer, durch starkes Oszillieren geprägter Konvergenzverlauf von BCG dargestellt. Modifiziert man dieses Verfahren, indem die Iterierte $\mathbf{x}_n$ nur dann aktualisiert wird, falls sich die BCG-Iterierte $\widetilde{\mathbf{x}}_n$ nicht verschlechtert, ergibt sich die untere Kurve und das entsprechend modifizierte Verfahren der bikonjugierten Gradienten, siehe Algorithmus 3.7. Dieses wird in Abschnitt 3.6 erneut interpretiert.

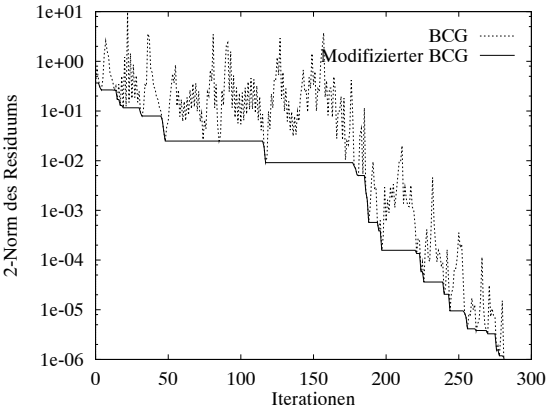


Abbildung 3.1: Typischer Konvergenzverlauf von BCG.

Algorithmus 3.7 Modifizierter BCG zur Lösung von $\mathbf{A}\mathbf{x} = \mathbf{b}$

Eingabe: $\mathbf{A}, \mathbf{b}, \mathbf{x}_0$

Führe Initialisierung von Algorithmus 3.5 aus

$\kappa_0 = -1,\ \widetilde{\mathbf{x}}_0 = \mathbf{x}_0\,,\ \|\mathbf{r}_0\|_2 = \gamma_1$

for $n = 1, 2, 3, \ldots$ **do**

Führe n-ten Iterationsschritt von Algorithmus 3.5 aus

$$\kappa_n = -\frac{\gamma_n}{\tau_n}\kappa_{n-1} \qquad \text{Gl. (3.71)}$$

$$\widetilde{\mathbf{x}}_n = \widetilde{\mathbf{x}}_{n-1} + \kappa_n \mathbf{p}_n \qquad \text{Gl. (3.73)}$$

$$\|\widetilde{\mathbf{r}}_n\|_2 = |\kappa_n \gamma_{n+1}| \qquad \text{Gl. (3.74)}$$

if $\|\widetilde{\mathbf{r}}_n\|_2 \leq \|\mathbf{r}_{n-1}\|_2$ **then**

$\mathbf{x}_n = \widetilde{\mathbf{x}}_n\,, \quad \|\mathbf{r}_n\|_2 = \|\widetilde{\mathbf{r}}_n\|_2$

else

$\mathbf{x}_n = \mathbf{x}_{n-1}\,, \quad \|\mathbf{r}_n\|_2 = \|\mathbf{r}_{n-1}\|_2$

endif

enddo

3.5 Spezielle Minimierungsprobleme in der l_p-Norm

In diesem Abschnitt wird das spezielle Minimierungsproblem (3.66) der Strategie 2 behandelt. Anders als im Fall $p = 2$ können hier Techniken wie die der Givens–Ro-

tationen [31] oder die der Normalgleichungen [68] zur Lösung des Problems nicht verwendet werden. Das Ergebnis wird dann im nachfolgenden Paragraphen zur Herleitung der verallgemeinerten Methode der quasi-minimalen Residuen verwendet. Es wird gezeigt, daß die Lösung $\mathbf{y}_n$ für $1 < p \leq \infty$ eindeutig bestimmt ist und sich ebenso wie das Residuum $\mathbf{s}_n$ mit Hilfe einer kurzen Rekursion darstellen läßt. Im Fall $p = 1$ kann eine Lösung des l_1-Minimierungsproblems explizit angegeben werden.

Im nachfolgenden Lemma wird zunächst ein Hilfsproblem gelöst, mit dem eine explizite Darstellung von $\mathbf{y}_n$ und $\mathbf{s}_n$ in Satz 3.5.1 angegeben wird. Der Satz 3.5.2 gibt die rekursive Darstellung der beiden Vektoren an, und ein abschließendes Korollar wird für den nächsten Abschnitt bereitgestellt.

Lemma 3.5.1. *Sei $n \in \mathbb{N}$ und $f : \mathbb{R}^n \to \mathbb{R}^{n+1}$ eine affine Abbildung der Form*

$$f(\mathbf{x}) = \beta_0 \left(1 - \sum_{i=1}^{n} x_i\right) \mathbf{e}_1^{(n+1)} + \sum_{i=1}^{n} \beta_i \, x_i \, \mathbf{e}_{i+1}^{(n+1)}$$

mit $\beta_0, \beta_1, \ldots, \beta_{n-1} \in \mathbb{R} \setminus \{0\}$ und $\beta_n \in \mathbb{R}$. Für $1 \leq p \leq \infty$ bezeichne der Vektor $\boldsymbol{\alpha}^{(n)} = \left(\alpha_1^{(n)}, \ldots, \alpha_n^{(n)}\right)^T \in \mathbb{R}^n$ eine Lösung des l_p-Minimierungsproblems

$$\|f(\boldsymbol{\alpha}^{(n)})\|_p = \min_{\mathbf{x} \in \mathbb{R}^n} \|f(\mathbf{x})\|_p \, .$$

Ist $m \in \mathbb{N}$ ein Index, der durch die Gleichung $|\beta_m| = \min_{j=0,\ldots,n} |\beta_j|$ bestimmt wird, so löst

$$\alpha_i^{(n)} = \begin{cases} 1 \, , & \textit{falls } \; i = m \, , \\ 0 \, , & \textit{sonst} \, , \end{cases} \qquad i = 1, \ldots, n \, , \tag{3.76}$$

das l_1-Minimierungsproblem. Gilt $1 < p \leq \infty$, so wird die eindeutige Lösung durch

$$\alpha_n^{(n)} = \frac{1}{1 + \sum_{j=0}^{n-1} \left|\frac{\beta_n}{\beta_j}\right|^q} \, , \tag{3.77}$$

$$\alpha_i^{(n)} = \left|\frac{\beta_n}{\beta_i}\right|^q \alpha_n^{(n)} \qquad , \; i = 1, \ldots, n-1, \tag{3.78}$$

gegeben, wobei $q = \frac{p}{p-1}$ für $1 < p < \infty$ und $q = 1$, falls $p = \infty$.

Beweis. Im Fall $p = 1$ ist

$$\|f(\mathbf{x})\|_1 \; = \; |\beta_0| \left|1 - \sum_{i=1}^{n} x_i\right| \, + \, \sum_{i=1}^{n} |\beta_i| \, |x_i| \; \geq \; |\beta_m| \left(\left|1 - \sum_{i=1}^{n} x_i\right| \, + \, \sum_{i=1}^{n} |x_i|\right) .$$

Nach der Dreiecksungleichung gilt $|1 - \sum_{i=1}^{n} x_i| \geq 1 - \sum_{i=1}^{n} |x_i|$, so daß

$$\|f(\mathbf{x})\|_1 \geq |\beta_m| = \|f(\boldsymbol{\alpha}^{(n)})\|_1 \, .$$

Für $1 < p < \infty$ ist das angegebene $\boldsymbol{\alpha}^{(n)}$ wegen $\nabla \|f(\boldsymbol{\alpha}^{(n)})\|_p^p = \mathbf{0}$ eine Lösung des Minimierungsproblems. Die Abbildung f wird mit

$$\mathbf{A}_n = \begin{pmatrix} \beta_0 & \dots & \beta_0 \\ -\beta_1 & & \\ & \ddots & \\ & & -\beta_n \end{pmatrix} \in \mathbb{R}^{(n+1)\times n}$$

auch durch die Gleichung $f(\mathbf{x}) = \beta_0 \mathbf{e}_1^{(n+1)} - \mathbf{A}_n \mathbf{x}$ gegeben. Die Eindeutigkeit ergibt sich deshalb aus $\operatorname{rg}(\mathbf{A}_n) = n$. Im Fall $p = \infty$ folgt die Lösungsdarstellung aus dem Grenzübergang $p \to \infty$. Ist $\beta_n = 0$, ist $\boldsymbol{\alpha}^{(n)} = (0, \dots, 0, 1)^T$ eindeutige Lösung. Für $\beta_n \neq 0$ besitzen alle $(n \times n)$-Untermatrizen von $\mathbf{A}_n$ vollen Rang, so daß nach Theorem 2.4 aus [77] auch in diesem Fall die Eindeutigkeit gewährleistet ist. □

Die Lösung des l_p-Minimierungsproblems (3.66) besitzt die folgende explizite Darstellung.

Satz 3.5.1. *Das l_p-Minimierungsproblem* (3.66) *besitzt die Lösung*

$$\mathbf{y}_n = \sum_{i=1}^{n} \alpha_i^{(n)} \, \mathbf{y}_i^{(n)} \tag{3.79}$$

mit zugehörigem Residuum

$$\mathbf{s}_n = \beta_0 \left(1 - \sum_{i=1}^{n} \alpha_i^{(n)}\right) \mathbf{e}_1^{(n+1)} + \sum_{i=1}^{n} \beta_i \, \alpha_i^{(n)} \, \mathbf{e}_{i+1}^{(n+1)} \, . \tag{3.80}$$

Hierbei sind, vergleiche (3.68)*,*

$$\mathbf{y}_i^{(n)} = (\eta_1, \dots, \eta_i, 0, \dots, 0)^T \in \mathbb{R}^n \, , \qquad i = 1, \dots, n, \tag{3.81}$$

und

$$\eta_j = -\frac{\gamma_j}{\tau_j} \, \eta_{j-1} \, , \qquad j = 2, \dots, n \, , \qquad \eta_1 = \frac{\gamma_1}{\tau_1} \, , \tag{3.82}$$

$$\beta_j = \gamma_{j+1} \, \eta_j \, , \qquad j = 1, \dots, n \, , \qquad \beta_0 = \gamma_1 \, . \tag{3.83}$$

Die Koeffizienten $\alpha_i^{(n)}$, $i = 1, \dots, n$, ergeben sich aus (3.76) *bzw.* (3.77) *und* (3.78) *unter Verwendung von* (3.83)*. Für $1 < p \leq \infty$ ist die Lösung eindeutig.*

Beweis. Sei $\mathbf{x} = (x_1, \dots, x_n)^T \in \mathbb{R}^n$. Für $\mathbf{y} = \sum_{i=1}^n x_i \, \mathbf{y}_i^{(n)}$ ist

$$\begin{aligned} \gamma_1 \, \mathbf{e}_1^{(n+1)} - \mathbf{L}_n \, \mathbf{y} \quad &= \quad \gamma_1 \, \mathbf{e}_1^{(n+1)} - \mathbf{L}_n \left(\sum_{i=1}^{n} x_i \, \mathbf{y}_i^{(n)} \right) \\ &= \quad \beta_0 \, \mathbf{e}_1^{(n+1)} - \sum_{i=1}^{n} x_i \left(\mathbf{L}_n \, \mathbf{y}_i^{(n)} \right) . \end{aligned}$$

Weiter gilt wegen $\mathbf{L}_n\,\mathbf{y}_i^{(n)} = \beta_0\,\mathbf{e}_1^{(n+1)} - \beta_i\,\mathbf{e}_{i+1}^{(n+1)}$, vergleiche (3.72),

$$\gamma_1\,\mathbf{e}_1^{(n+1)} - \mathbf{L}_n\,\mathbf{y} \quad = \quad \beta_0\left(1 - \sum_{i=1}^{n} x_i\right)\mathbf{e}_1^{(n+1)} + \sum_{i=1}^{n}\beta_i\,x_i\,\mathbf{e}_{i+1}^{(n+1)} \quad = \quad f(\mathbf{x})$$

mit f aus Lemma 3.5.1. Nach Voraussetzung (3.67) sind die Komponenten $\eta_i \neq 0$ für $i = 1, \ldots, n$. Folglich bilden die Vektoren $\mathbf{y}_1^{(n)}, \ldots, \mathbf{y}_n^{(n)}$ eine Basis des $\mathbb{R}^n$. Deshalb ist

$$\min_{\mathbf{y}\in\mathbb{R}^n}\left\|\gamma_1\,\mathbf{e}_1^{(n+1)} - \mathbf{L}_n\,\mathbf{y}\right\|_p \quad = \quad \min_{\mathbf{x}\in\mathbb{R}^n}\|f(\mathbf{x})\|_p\,.$$

Da nach (3.67) auch die Koeffizienten $\beta_0, \ldots, \beta_{n-1}$ von Null verschieden sind, ergibt sich die Behauptung des Satzes aus Lemma 3.5.1. □

Die Lösung $\mathbf{y}_n$ des l_p-Minimierungsproblems (3.66) mit entsprechendem Residuum $\mathbf{s}_n$ besitzt für $p \neq 1$ die folgende rekursive Darstellung.

Satz 3.5.2. *Sei $n \in \mathbb{N}$. Für $1 < p < \infty$ setze $q = \frac{p}{p-1}$ und $q = 1$, falls $p = \infty$. Die eindeutige Lösung $\mathbf{y}_n \in \mathbb{R}^n$ des l_p-Minimierungsproblems* (3.66) *mit dem zugehörigen Residuum $\mathbf{s}_n \in \mathbb{R}^{n+1}$ ist rekursiv durch*

$$\mathbf{y}_n = (1+\sigma_n)\begin{pmatrix}\mathbf{y}_{n-1}\\ 0\end{pmatrix} - \sigma_n\begin{pmatrix}\mathbf{y}_{n-2}\\ 0\\ 0\end{pmatrix} + \kappa_n\,\mathbf{e}_n^{(n)}\,, \qquad n \geq 3\,, \tag{3.84}$$

mit

$$\mathbf{y}_1 = \kappa_1 \quad \textit{und} \quad \mathbf{y}_2 = \begin{pmatrix}(1+\sigma_2)\kappa_1\\ \kappa_2\end{pmatrix}\,, \tag{3.85}$$

$$\mathbf{s}_n = \frac{\vartheta_n}{\lambda_n + \vartheta_n}\begin{pmatrix}\mathbf{s}_{n-1}\\ 0\end{pmatrix} - \gamma_{n+1}\,\kappa_n\,\mathbf{e}_{n+1}^{(n+1)}\,, \qquad n \geq 2\,, \tag{3.86}$$

mit

$$\mathbf{s}_1 = -\gamma_2\kappa_1\mathbf{e}_2^{(2)} \tag{3.87}$$

gegeben, wobei

$$\sigma_n = \frac{1-\lambda_n}{\lambda_n + \vartheta_n}\,, \qquad n \geq 1\,, \tag{3.88}$$

$$\kappa_n = -\frac{\gamma_n}{\tau_n}\cdot\frac{\kappa_{n-1}}{\lambda_n + \vartheta_n}\,, \qquad n \geq 1\,, \quad \kappa_0 = -1\,, \tag{3.89}$$

und

$$\vartheta_n = \left| \frac{\gamma_{n+1}}{\tau_n} \right|^q \quad , \qquad\qquad n \geq 1 \quad , \tag{3.90}$$

$$\lambda_n = \frac{\lambda_{n-1}}{\lambda_{n-1} + \vartheta_{n-1}} \quad , \qquad n \geq 2 \quad , \quad \lambda_1 = 1 \quad . \tag{3.91}$$

Zusätzlich läßt sich $\|\mathbf{s}_n\|_p$ iterativ durch die Rekursion

$$\|\mathbf{s}_n\|_p \;=\; \sqrt[p]{\left(\tfrac{\vartheta_n}{\lambda_n+\vartheta_n}\right)^p \|\mathbf{s}_{n-1}\|_p^p - |\gamma_{n+1}\,\kappa_n|^p} \tag{3.92}$$

berechnen.

Beweis. Für $n \in \mathbb{N}$ ergibt sich mit Hilfe einer Induktion aus (3.91), (3.77) und (3.83) die Identität $\lambda_{n+1} = \alpha_n^{(n)}$. Aus (3.89) in Verbindung mit (3.91) und (3.82) folgt hieraus induktiv

$$\kappa_n \;=\; \alpha_n^{(n)}\,\eta_n \,. \tag{3.93}$$

Es wird nun gezeigt, daß die Gleichung (3.84) mit (3.79) übereinstimmt. In den Fällen $n = 1$ und $n = 2$ ist dies offensichtlich. Für $n \geq 3$ folgt mit der Induktionsvoraussetzung aus (3.84)

$$\begin{aligned}
\mathbf{y}_n &= (1+\sigma_n) \begin{pmatrix} \sum_{i=1}^{n-1} \alpha_i^{(n-1)}\,\mathbf{y}_i^{(n-1)} \\ 0 \end{pmatrix} - \sigma_n \begin{pmatrix} \sum_{i=1}^{n-2} \alpha_i^{(n-2)}\,\mathbf{y}_i^{(n-2)} \\ 0 \\ 0 \end{pmatrix} + \kappa_n\,\mathbf{e}_n^{(n)} \\
&= (1+\sigma_n) \sum_{i=1}^{n-1} \alpha_i^{(n-1)}\,\mathbf{y}_i^{(n)} - \sigma_n \sum_{i=1}^{n-2} \alpha_i^{(n-2)}\,\mathbf{y}_i^{(n)} + \kappa_n\,\mathbf{e}_n^{(n)} \\
&= \sum_{i=1}^{n-2} \left((1+\sigma_n)\,\alpha_i^{(n-1)} - \sigma_n\,\alpha_i^{(n-2)} \right) \mathbf{y}_i^{(n)} + (1+\sigma_n)\alpha_{n-1}^{(n-1)}\,\mathbf{y}_{n-1}^{(n)} + \kappa_n\,\mathbf{e}_n^{(n)} \,.
\end{aligned}$$

Da nach (3.93) $\kappa_n\,\mathbf{e}_n^{(n)} = \alpha_n^{(n)}\,\eta_n\,\mathbf{e}_n^{(n)} = \alpha_n^{(n)} \left(\mathbf{y}_n^{(n)} - \mathbf{y}_{n-1}^{(n)} \right)$, gilt weiter

$$\begin{aligned}
\mathbf{y}_n = \sum_{i=1}^{n-2} \left((1+\sigma_n)\,\alpha_i^{(n-1)} - \sigma_n\,\alpha_i^{(n-2)} \right) \mathbf{y}_i^{(n)} \\
+ \left((1+\sigma_n)\alpha_{n-1}^{(n-1)} - \alpha_n^{(n)} \right) \mathbf{y}_{n-1}^{(n)} + \alpha_n^{(n)} \mathbf{y}_n^{(n)} \,.
\end{aligned}$$

Eine Rechnung zeigt, daß $(1+\sigma_n)\,\alpha_i^{(n-1)} - \sigma_n\,\alpha_i^{(n-2)} = \alpha_i^{(n)}$ für $i = 1, \ldots, n-2$ und $(1+\sigma_n)\alpha_{n-1}^{(n-1)} - \alpha_n^{(n)} = \alpha_{n-1}^{(n)}$, so daß

$$\mathbf{y}_n = \sum_{i=1}^{n} \alpha_i^{(n)}\,\mathbf{y}_i^{(n)}$$

bewiesen ist. Analog ergibt sich die Äquivalenz von (3.86) und (3.80). Die Rekursion (3.92) folgt unmittelbar aus Darstellung (3.86). □

Das folgende Korollar wird im nächsten Abschnitt benötigt.

Korollar 3.5.1. *Für* $1 < p \leq \infty$ *und* $n \in \mathbb{N}$ *berechnet die Iteration*

$$\mathbf{y}_n = \begin{pmatrix} \mathbf{y}_{n-1} \\ 0 \end{pmatrix} + \mathbf{g}_n \,, \qquad n \geq 2 \,, \quad \mathbf{y}_1 = \kappa_1 \,, \tag{3.94}$$

mit

$$\mathbf{g}_n = \sigma_n \begin{pmatrix} \mathbf{g}_{n-1} \\ 0 \end{pmatrix} + \kappa_n \mathbf{e}_n^{(n)} \,, \quad n \geq 2 \,, \quad \mathbf{g}_1 = \kappa_1 \,, \tag{3.95}$$

die eindeutige Lösung des l_p*-Minimierungsproblems* (3.66)*. Die Koeffizienten* κ_n *und* σ_n *ergeben sich aus den Gleichungen* (3.88)–(3.91)*.*

Beweis. Durch eine Induktion zeigt man, daß die Iteration (3.94) und (3.95) mit der Rekursion (3.84) übereinstimmt. □

3.6 Die Methode der quasi-l_p-minimalen Residuen

Mit Hilfe von Korollar 3.5.1 des vorigen Abschnittes können nun die Iterierten $\mathbf{x}_n$ und $\mathbf{r}_n$ von Algorithmus 3.5 durch Strategie 2 aktualisiert werden. Geht man zur Gleichung (3.60) zurück und ersetzt darin $\mathbf{y}_n$ durch die Rekursion (3.94), so ergibt sich für $1 < p \leq \infty$

$$\begin{aligned} \mathbf{x}_n &= \mathbf{x}_0 + \mathbf{P}_n \mathbf{y}_n \\ &= \mathbf{x}_0 + \mathbf{P}_{n-1} \mathbf{y}_{n-1} + \mathbf{P}_n \mathbf{g}_n \\ &= \mathbf{x}_{n-1} + \mathbf{d}_n \end{aligned} \tag{3.96}$$

mit dem neu eingeführten Vektor $\mathbf{d}_n = \mathbf{P}_n \mathbf{g}_n$. Dieser kann mit der zweiten Rekursion (3.95) durch

$$\begin{aligned} \mathbf{d}_n &= \sigma_n \mathbf{P}_{n-1} \mathbf{g}_{n-1} + \kappa_n \mathbf{P}_n \mathbf{e}_n^{(n)} \\ &= \sigma_n \mathbf{d}_{n-1} + \kappa_n \mathbf{p}_n \end{aligned} \tag{3.97}$$

aktualisiert werden, wobei $\mathbf{d}_0 = \mathbf{0}$ zu setzen ist. Der zu $\mathbf{x}_n$ gehörige Residuumsvektor $\mathbf{r}_n$ wird mit (3.63) und (3.86) bestimmt, so daß

$$\begin{aligned} \mathbf{r}_n &= \mathbf{V}_{n+1} \mathbf{s}_n \\ &= \frac{\vartheta_n}{\lambda_n + \vartheta_n} \mathbf{V}_n \mathbf{s}_{n-1} - \gamma_{n+1} \kappa_n \mathbf{v}_{n+1} \\ &= \frac{\vartheta_n}{\lambda_n + \vartheta_n} \mathbf{r}_{n-1} - \kappa_n \widetilde{\mathbf{v}}_{n+1} \,, \end{aligned} \tag{3.98}$$

vergleiche (3.41). Die Koeffizienten σ_n, κ_n, ϑ_n und λ_n erhält man gemäß den Rekursionen (3.88)–(3.91). Auf diese Weise ergibt sich Algorithmus 3.8, der für $p = 2$ in das sogenannte Parallel Iterativ Method package [16, Vers. 2.1] aufgenommen wurde und dadurch allgemein verfügbar ist.

Algorithmus 3.8 Verallgemeinertes QMR zur Lösung von $\mathbf{A}\mathbf{x} = \mathbf{b}$

Eingabe: $\mathbf{A}, \mathbf{b}$, $\mathbf{x}_0$ und $1 < p \leq \infty$

Führe Initialisierung von Algorithmus 3.5 aus

$\kappa_0 = -1, \ \lambda_1 = 1$

$\mathbf{d}_0 = \mathbf{0}$

if $p = \infty$ **then** $q = 1$ **else** $q = \dfrac{p}{p-1}$ **endif**

for $n = 1, 2, 3, \ldots$ **do**

Führe n-ten Iterationsschritt von Algorithmus 3.5 aus

$$\vartheta_n = \left| \frac{\gamma_{n+1}}{\tau_n} \right|^q \qquad \text{Gl. (3.90)}$$

$$\sigma_n = \frac{1 - \lambda_n}{\lambda_n + \vartheta_n} \qquad \text{Gl. (3.88)}$$

$$\kappa_n = -\frac{\gamma_n}{\tau_n} \frac{\kappa_{n-1}}{\lambda_n + \vartheta_n} \qquad \text{Gl. (3.89)}$$

$$\lambda_{n+1} = \frac{\lambda_n}{\lambda_n + \vartheta_n} \qquad \text{Gl. (3.91)}$$

$$\mathbf{d}_n = \sigma_n \mathbf{d}_{n-1} + \kappa_n \mathbf{p}_n \qquad \text{Gl. (3.97)}$$

$$\mathbf{x}_n = \mathbf{x}_{n-1} + \mathbf{d}_n \qquad \text{Gl. (3.96)}$$

$$\mathbf{r}_n = \frac{\vartheta_n}{\lambda_n + \vartheta_n} \mathbf{r}_{n-1} - \kappa_n \widetilde{\mathbf{v}}_{n+1} \qquad \text{Gl. (3.98)}$$

enddo

Eine gute Abschätzung von $\|\mathbf{r}_n\|_2$ ist durch (3.92) gegeben, so daß $\|\mathbf{r}_n\|_2$, und damit $\mathbf{r}_n$, erst berechnet werden muß, wenn $\|\mathbf{s}_n\|_p$ kleiner als eine vom Benutzer eingestellte Toleranz ist. Neben den zwei Matrix-Vektor-Produkten benötigt dieser Algorithmus, ohne die Berechnung von $\mathbf{r}_n$ und $\|\mathbf{r}_n\|_2$, $22N + c$ Gleitpunktoperationen und einen Synchronisationspunkt. Die Operationsanzahl stimmt mit dem Original, d.h. Algorithmus 7.1 aus [31], auch zu finden in [4], überein. Für $p = 2$ findet sich ein Vergleich der beiden Algorithmen in [8].

Der im Fall $p = 1$ aus Strategie 2 resultierende Algorithmus kann mit Hilfe von Satz 3.5.1 formuliert werden. Danach stimmt die Iterierte $\mathbf{x}_n$ im n-ten Iterationsschritt mit der bis dahin besten BCG-Lösung überein. Folglich entspricht diesem Verfahren der modifizierte BCG-Algorithmus 3.7. Darüber hinaus gilt wegen Satz 3.5.1

und (3.75) in Ungleichung (3.64) sogar die Gleichheit

$$\|\mathbf{r}_n\|_2 = \|\mathbf{s}_n\|_1 ,$$

d.h. die euklidische Norm des Residuums stimmt mit der l_1-Norm der Lösung des l_1-Minimierungsproblems (3.66) überein.

Der symmetrische Fall

Im Fall $\mathbf{A} = \mathbf{A}^T$ wird in Algorithmus 3.8 der benötigte Algorithmus 3.5 durch den symmetrischen Algorithmus 3.4 ersetzt. Die erzeugten Lanczos–Vektoren sind hier wegen (3.51) orthonormal, so daß aus (3.63) für $p = 2$ die Identität

$$\|\mathbf{r}_n\|_2 = \|\mathbf{V}_{n+1}\mathbf{s}_n\|_2 = \|\mathbf{s}_n\|_2$$

folgt. Somit wird die Iterierte $\mathbf{x}_n$ durch die in (3.3) gegebene Minimierungseigenschaft charakterisiert. Ist darüber hinaus die Matrix $\mathbf{A}$ positiv definit, folgt aus $\tau_n = 0$ in Algorithmus 3.4 auch $\mathbf{p}_n = \mathbf{0}$. Damit ist $\varepsilon_n = 0$ und schließlich $\gamma_{n+1} = 0$. Also findet der Algorithmus 3.8 in exakter Arithmetik nach maximal N Schritten die exakte Lösung $\mathbf{A}^{-1}\mathbf{b}$, wenn N die Ordnung der Matrix $\mathbf{A}$ beschreibt.

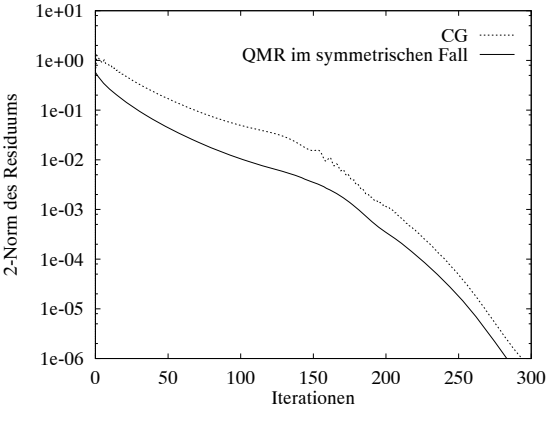


Abbildung 3.2: Typischer Konvergenzverlauf von CG und QMR im symmetrisch positiv definiten Fall.

3.7 Numerische Experimente

Das Verhalten der entwickelten Algorithmen wird in diesem Abschnitt anhand einer Diskretisierung der unten angegebenen partiellen Differentialgleichung untersucht. Die Rechnungen dazu wurden in Fortran mit doppelter Genauigkeit auf dem Parallelrechner Intel Paragon XP/S 10 des Forschungszentrums Jülich durchgeführt.

Beispiel 3.7.1. *Gegeben sei die partielle Differentialgleichung*

$$-\Delta u - 20\left(x\frac{\partial u}{\partial x} + y\frac{\partial u}{\partial y} + z\frac{\partial u}{\partial z}\right) = f \quad in \quad \Omega = (0,1)\times(0,1)\times(0,1)$$

mit homogener Dirichlet–Randbedingung, wobei die rechte Seite f so gewählt wird, daß

$$u(x,y,z) = 2\sin(4\pi x)\,\sin(6\pi y)\,\sin(4\pi z)$$

Lösung der Differentialgleichung ist.

Zur Diskretisierung werden zentrale Differenzen zweiter Ordnung auf einem $60 \times 60 \times 60$ Gitter mit der Schrittweite $1/61$ benutzt. Das resultierende Gleichungssystem besitzt eine Koeffizientenmatrix der Ordnung $N = 216000$ mit 1490400 Nichtnullelementen. Für die Testrechnungen wurde stets $\mathbf{x}_0 = \mathbf{0}$ als Startlösung gewählt, und die zugehörigen Verfahren wurden bei einer Genauigkeit des Residuums von $\|\mathbf{r}_n\|_2 < 10^{-6}$ beendet.

Zur Verteilung der Daten auf die $\widetilde{p}$ Prozessoren des Rechners wurde das Gebiet Ω in $\widetilde{p}$ gleichgroße Würfel zerlegt, so daß jeder Prozessor die zu seinem Teilgebiet gehörenden Daten verwaltet. Aufgrund der lokalen Struktur des Diskretisierungsschemas muß bei der Durchführung des Matrix-Vektor-Produktes nach Kapitel 2 jeder Prozessor mit maximal 6 Prozessoren kommunizieren.

In Abbildung 3.3 wird der Konvergenzverlauf von Algorithmus 3.8 ($p = 2$) mit dem Original, d.h. Algorithmus 7.1 aus [31], verglichen. Beide Algorithmen zeigen das gleiche Konvergenzverhalten.

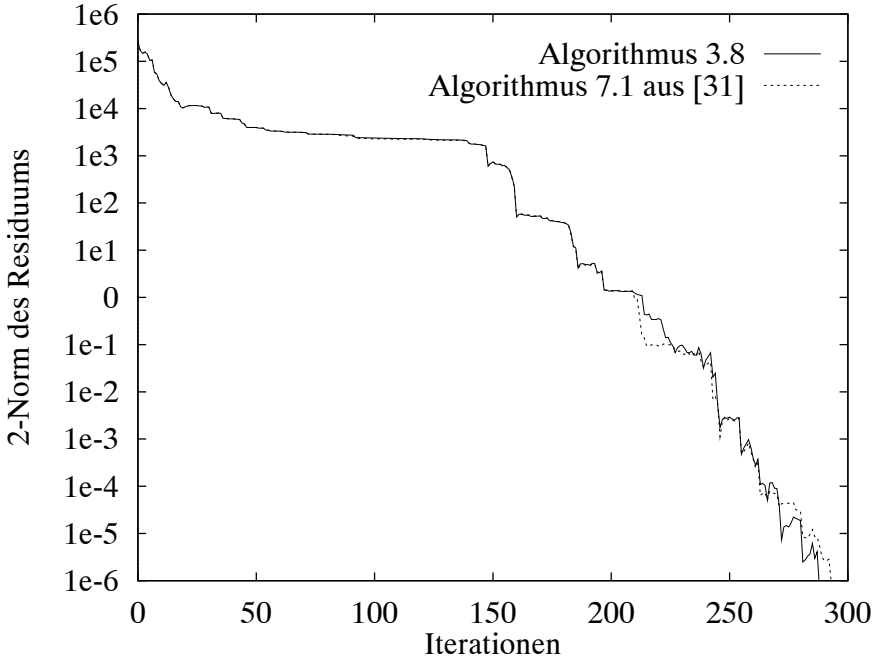


Abbildung 3.3: Vergleich des Residuumsverlaufs auf 125 Prozessoren.

Jedoch ergeben sich folgende Unterschiede bezüglich der parallelen Performance. In Abbildung 3.4 wird der Speedup der beiden Verfahren für eine feste Anzahl von Iterationsschritten angegeben. Abbildung 3.5 gibt den daraus resultierenden Zeitgewinn von Algorithmus 3.8 gegenüber Algorithmus 7.1 aus [31] wieder. Insbesondere wird darin die Größe $1 - T_1(\widetilde{p})/T_2(\widetilde{p})$ in Prozent dargestellt, wobei $T_1(\widetilde{p})$ die Laufzeit von Algorithmus 3.8 und $T_2(\widetilde{p})$ die Laufzeit von Algorithmus 7.1 aus [31] auf $\widetilde{p}$ Prozessoren angeben. Die neue Variante mit einem globalen Synchronisationspunkt ist stets

schneller und zeigt ihren Vorteil insbesondere beim Einsatz vieler Prozessoren.

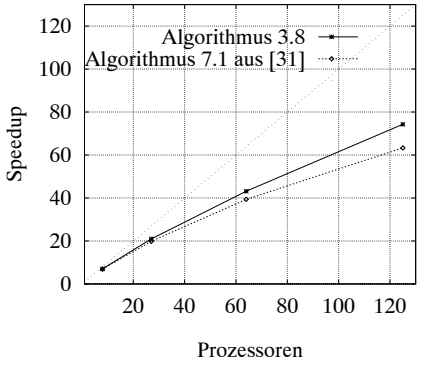


Abbildung 3.4: Speedup der beiden Verfahren

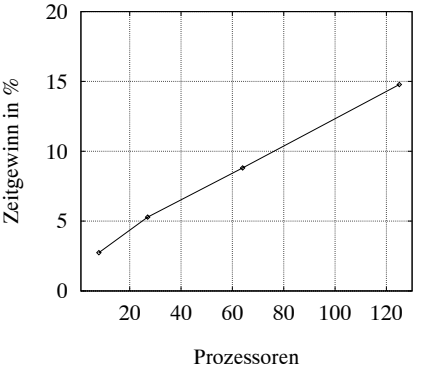


Abbildung 3.5: Zeitgewinn

Zum Abschluß des Kapitels ist in Abbildung 3.6 die euklidische Norm des Residuums beim Algorithmus 3.8 für verschiedene l_p-Normen angegeben. Für $p = 1$ muß Algorithmus 3.7 verwendet werden. Obwohl die Minimierung in den l_p-Normen nur geringen Einfluß auf den Residuumsverlauf hat, sind dennoch folgende Bemerkungen wichtig.

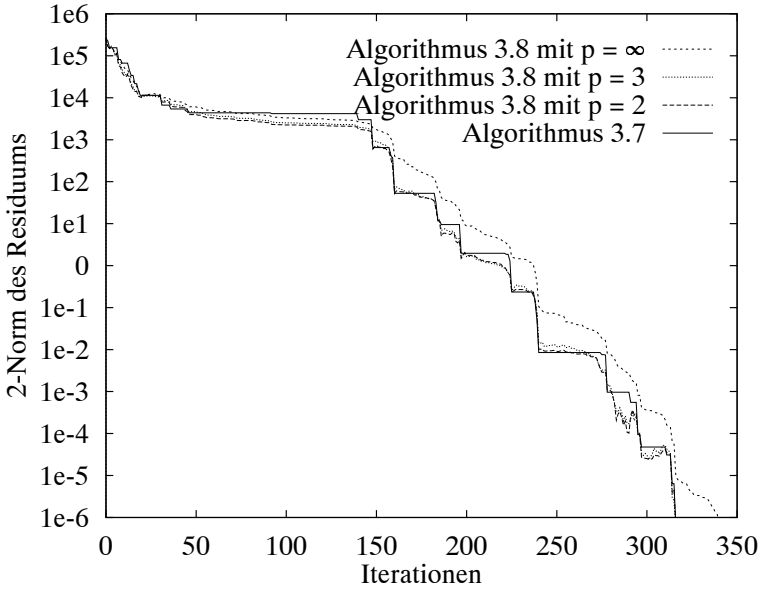


Abbildung 3.6: Vergleich des Residuumsverlaufs auf 125 Prozessoren.

Grundsätzlich benötigt Algorithmus 3.7 weniger Vektoroperationen als die anderen Algorithmen und das Verfahren der bikonjugierten Gradienten kann durch Algorithmus 3.7 in die Klasse der quasi-minimalen Residuenverfahren eingereiht werden. Insbesondere ist die Tatsache hervorzuheben, daß eine l_p-Minimierung hier überhaupt iterativ möglich ist.

Kapitel 4

Gebietszerlegungsmethoden

Gebietszerlegungsmethoden sind Techniken, die zur Lösung partieller Differentialgleichungen eingesetzt werden, und die aufgrund eines grob-granularen Parallelismus den Einsatz von parallelen Rechnersystemen, insbesondere von Systemen mit verteiltem Speicher ermöglichen. Die Grundidee kann dabei folgendermaßen beschrieben werden.

Zunächst wird der Definitionsbereich der partiellen Differentialgleichung in verschiedene Teilgebiete unterteilt. Auf jedem dieser Gebiete wird dann ein zu dieser Gleichung korrespondierendes Problem formuliert und anschließend gelöst, so daß die Teillösungen zu einer Approximation der Lösung auf dem Gesamtgebiet zusammengesetzt werden können.

Für elliptische Differentialgleichungen basiert diese Idee auf einem Verfahren, das ursprünglich von H. A. Schwarz [70] eingeführt wurde. Durch einen alternierenden Prozeß bewies er die Existenz einer Lösung der Poisson–Gleichung in einem zusammengesetzten überlappenden Gebiet $\Omega = \Omega_1 \cup \Omega_2$ mit den inneren Rändern $\Gamma_1 = \partial\Omega_1 \cap \Omega_2$ und $\Gamma_2 = \partial\Omega_2 \cap \Omega_1$, siehe Abbildung 4.1.

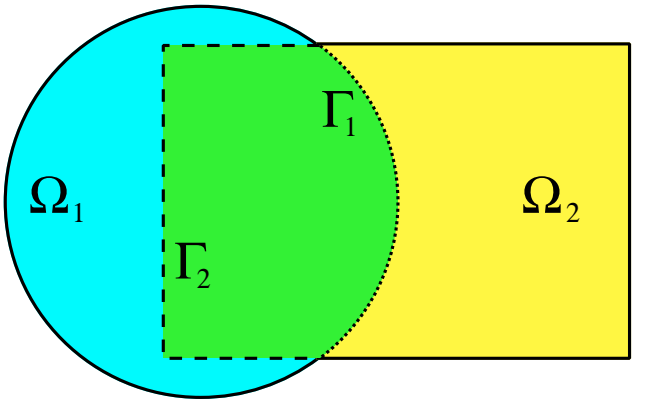


Abbildung 4.1: Überlappende Zerlegung $\Omega = \Omega_1 \cup \Omega_2$

Bezeichnet nun L einen allgemeinen elliptischen Differentialoperator, so kann das Ver-

fahren zur Lösung der linearen Differentialgleichung

$$\begin{aligned} L(u) &= f \qquad \text{in } \Omega\,, \\ u &= g \qquad \text{auf } \partial\Omega \end{aligned} \tag{4.1}$$

wie folgt beschrieben werden. Für eine auf Ω_2 gegebene Startfunktion $u_2^{(0)}$, die nur auf Γ_1 definiert werden muß, vergleiche Abbildung 4.1, werden nun nacheinander für $n = 0, 1, 2, \ldots$ die beiden Randwertprobleme

$$\begin{aligned} L\left(u_1^{(n+1)}\right) &= f \qquad && \text{in } \Omega_1\,, \\ u_1^{(n+1)} &= g \qquad && \text{auf } \partial\Omega_1 \setminus \Gamma_1\,, \\ u_1^{(n+1)} &= u_2^{(n)} \qquad && \text{auf } \Gamma_1 \end{aligned} \tag{4.2}$$

und

$$\begin{aligned} L\left(u_2^{(n+1)}\right) &= f \qquad && \text{in } \Omega_2\,, \\ u_2^{(n+1)} &= g \qquad && \text{auf } \partial\Omega_2 \setminus \Gamma_2\,, \\ u_2^{(n+1)} &= u_1^{(n+1)} \qquad && \text{auf } \Gamma_2 \end{aligned} \tag{4.3}$$

iterativ gelöst. Der gesamte Prozeß wird als alternierendes Schwarz–Verfahren bezeichnet. In jedem "halben" Schritt des Verfahrens wird die elliptische Differentialgleichung auf den beiden Teilgebieten Ω_i mit $i = 1, 2$ mit den Randwerten g auf $\partial\Omega_i \setminus \Gamma_i$ und den Werten der vorherigen Approximation auf dem inneren Rand Γ_i gelöst. Eine parallele Variante der Methode erhält man, indem die beiden Teilprobleme (4.2) und (4.3) nicht nacheinander, sondern gleichzeitig gelöst werden und dabei die Randwerte von der vorherigen Iterierten auf dem benachbarten Gebiet benutzt werden. Bezüglich ihrer Konvergenz werden diese und ähnliche Verfahren beispielsweise in [23, 24, 25] untersucht.

Wesentlich für die obige Vorgehensweise ist, daß die Ränder Γ_1 und Γ_2 im Inneren der Teilgebiete Ω_2 bzw. Ω_1 liegen, so daß hierdurch die entsprechenden Randwerte aktualisiert werden können. Soll das parallele Verfahren nun für mehr als zwei Teilgebiete formuliert werden, muß diese Eigenschaft berücksichtigt werden. Dafür ist Punkt (iii) der folgenden Definition erforderlich.

Definition 4.0.1. *Die Menge $\{\Omega_i\,,\ i = 1, \ldots, p\}$ heißt überlappende Zerlegung von $\Omega \subset \mathbb{R}^n$ mit $n \in \{1, 2, 3\}$, falls die folgenden Bedingungen erfüllt sind:*

(i) Für $i = 1, \ldots, p$ sind die Mengen $\Omega_i \subset \mathbb{R}^n$ offen und nicht leer.

(ii) $\bigcup_{i=1}^{p} \Omega_i = \Omega.$

(iii) Für jedes $x \in \partial\Omega_i$ gilt: $x \in \partial\Omega$ oder $x \in \Omega_j$ für ein j.

Die dritte Eigenschaft gewährleistet dabei, daß jeder Funktionswert auf dem Rand $\partial\Omega_i \setminus \partial\Omega$ durch einen Funktionswert im Inneren eines Gebietes Ω_j aktualisiert werden kann. Während diese Zuordnung bei zwei überlappenden Teilgebieten eindeutig ist, kann es schon bei drei überlappenden Gebieten $\Omega = \Omega_1 \cup \Omega_2 \cup \Omega_3$ möglich sein, daß beispielsweise ein $x \in \partial\Omega_3 \setminus \partial\Omega$ sowohl zum Teilgebiet Ω_1 als auch zu Ω_2 gehört. Mit Hilfe der folgenden Definition soll deshalb zu jedem $x \in \partial\Omega_i \setminus \partial\Omega$ ein $j \in \{k : x \in \Omega_k\}$ gewählt werden, mit dem der Randwert auf $\partial\Omega_i$ durch den Wert der entsprechenden Funktion auf dem Gebiet Ω_j aktualisiert wird.

Definition 4.0.2. *Sei $\{\Omega_i,\ i = 1,\ldots,p\}$ eine überlappende Zerlegung von Ω. Die Funktionen $\gamma_i : \partial\Omega_i \to \{1,\ldots,p\}$ für $i = 1,\ldots,p$ heißen Randabbildungen, falls*

$$\gamma_i(x) = i \qquad \text{für } x \in \partial\Omega_i \cap \partial\Omega$$

und

$$\gamma_i(x) = k \quad \text{mit } k \in \{j : x \in \Omega_j\} \quad \text{für } x \in \partial\Omega_i \setminus \partial\Omega\,.$$

Wählt man zu einer Zerlegung $\{\Omega_i,\ i = 1,\ldots,p\}$ von Ω also die Randabbildungen γ_i für $i = 1,\ldots,p$, so kann eine mögliche Verallgemeinerung des parallelen Schwarz–Verfahrens zur Lösung von (4.1) auf p überlappende Teilgebiete wie folgt angegeben werden.

Verfahren 4.1 Paralleles Schwarz–Verfahren

Eingabe: Überlappende Zerlegung $\{\Omega_i,\ i = 1,\ldots,p\}$ von Ω mit Randabbildungen γ_i, L, f, g und $u_i^{(0)} : \overline{\Omega}_i \to \mathbb{R}$

$u_i^{(0)}(x) = g(x)\,,\ x \in \partial\Omega_i \cap \partial\Omega$ für $i = 1,\ldots,p$

for $n = 0, 1, 2, \ldots$ **do**

Löse für $i = 1,\ldots,p$ parallel:

$$\begin{aligned} L(u_i^{(n+1)}) &= f \text{ in } \Omega_i \\ u_i^{(n+1)}(x) &= u_{\gamma_i(x)}^{(n)}(x) \text{ für } x \in \partial\Omega_i \end{aligned}$$

enddo

Das parallele Schwarz–Verfahren, vergleiche auch [71], besitzt damit eine ähnliche Struktur wie das erweiterte Matrix-Vektor-Produkt aus Kapitel 2. Da zur Lösung der Differentialgleichung auf den Teilgebieten Ω_i Daten der Funktionen auf den benachbarten Gebieten Ω_j mit $j \in \{\gamma_i(x) : x \in \partial\Omega_i \setminus \partial\Omega\}$ benötigt werden, wird hier die Kommunikationsstruktur des Verfahrens 4.1 durch die spezielle Wahl der Randabbildungen γ_i festgelegt. Der Vorteil bei der Implementierung auf einem parallelen

Rechnersystem besteht dann einerseits in der lokalen Kommunikationsstruktur und andererseits in dem meistens einfach zu realisierenden Datenaustausch, bei dem jedem Prozessor P_i genau ein Teilgebiet Ω_i zugeordnet wird. Darüber hinaus kann zur Lösung der einzelnen Teilprobleme ein bereits vorhandenes sequentielles Programm auf jedem Prozessor eingesetzt werden.

Der Nachteil des parallelen Schwarz–Verfahrens liegt jedoch, wie später verdeutlicht wird, in seiner langsamen Konvergenz. Zur Beschleunigung werden häufig Grobgitterkorrekturen eingesetzt, bei deren Verwendung dann aber das lokale Kommunikationsschema durch ein globales ersetzt wird.

In diesem Kapitel werden Beschleunigungstechniken des parallelen Schwarz–Verfahrens vorgestellt, die die lokale Kommunikationsstruktur erhalten.

Für die durchzuführenden Untersuchungen wird eine geschlossene Darstellung der Lösung benötigt. Aus diesem Grund werden die folgenden Überlegungen im eindimensionalen Fall durchgeführt, da dann die allgemeine Lösung der jetzt gewöhnlichen Differentialgleichung $L(y) = f$ mit Hilfe der Greenschen Funktion in geschlossener Form angegeben werden kann, in der die beiden Randwerte als Parameter auftreten. Durch Kenntnis dieser allgemeinen Lösung in jedem überlappenden Teilintervall läßt sich so ein Gleichungssystem nur für die unbekannten Werte der Gesamtlösung in den Randpunkten der Teilintervalle aufstellen. Das parallele Schwarz–Verfahren kann damit durch das Jacobi–Verfahren zur Lösung dieses Gleichungssystems charakterisiert werden, das so durch verschiedene Vorkonditionierungen beschleunigt werden kann. Um die lokale Kommunikationsstruktur zu erhalten, ist der Grundgedanke dabei immer die Interpretation der Vorkonditionierung als Kopplung der Randbedingungen für die Differentialgleichungen in den Teilgebieten. Der Zusammenhang mit dem verallgemeinerten Schwarz–Verfahren [22, 56, 72, 73] wird bei diesen Untersuchungen herausgestellt.

Eine von der Dimension unabhängige Interpretation dieser Vorgehensweise, die zur Lösung linearer Gleichungssysteme eingesetzt werden kann, wird im nachfolgenden Kapitel gegeben.

4.1 Das explizite parallele Schwarz–Verfahren

In diesem Abschnitt wird das parallele Schwarz–Verfahren 4.1 zur Lösung der linearen gewöhnlichen Differentialgleichung

$$L(y) = f \text{ in } (a,b) \quad \text{mit } y(a) = y(b) = 0 \tag{4.4}$$

beschrieben. Für den eindimensionalen Fall kann zunächst die Definition 4.0.1 verschärft werden, so daß sich beispielsweise jedes Teilintervall nur mit zwei weiteren Intervallen überschneidet.

Definition 4.1.1. *Eine Zerlegung* $\{(a_i,b_i)\,,\ i = 1,\ldots,p\}$ *des Intervalls* (a,b) *heißt p–überlappend, falls*

$$a_1 = a, b_p = b \text{ und } a_i < a_{i+1} < b_i < b_{i+1} \text{ für } i = 1,\ldots,p-1.$$

Diese Zerlegung heißt (ε, p)–überlappend, falls zusätzlich

$$b_i - a_{i+1} = \varepsilon \quad \textit{für } i = 1, \dots, p-1\,.$$

Eine Zerlegung heißt gleichmäßig, falls alle Teilintervalle (a_i, b_i) gleich lang sind.

Für eine p–überlappende Zerlegung des Intervalls (a, b) sind auch die Randabbildungen aus Definition 4.0.2 eindeutig bestimmt, deren Funktionswerte hier durch $\gamma_i(a_i) = i - 1$, $i = 2, \dots, p$, und $\gamma_i(b_i) = i + 1$, $i = 1, \dots, p-1$, gegeben sind.

Mit Hilfe dieser Zerlegung von (a, b) kann nun die Randwertaufgabe (4.4) zunächst mit dem auf p Teilintervallen verallgemeinerten **a**lternierenden **S**chwarz–Verfahren (**AS**) gelöst werden. Hierbei werden die p Teilprobleme auf den Teilintervallen (a_i, b_i) nacheinander gelöst, wobei jeweils die aktuellen Randwerte benutzt werden. Somit stellt dieses Verfahren für $n \in \mathbb{N}$ ein Iterationsverfahren dar, dessen Lösungen des i-ten Teilproblems $y_i^{(n)}$ in *Einzelschritten* der Form

$$\left\{ \begin{array}{rcl} L\left(y_i^{(n+1)}\right) & = & f \text{ in } (a_i, b_i) \\ y_i^{(n+1)}(a_i) & = & y_{i-1}^{(n+1)}(a_i) \\ y_i^{(n+1)}(b_i) & = & y_{i+1}^{(n)}(b_i) \end{array} \right\} \qquad \textbf{(AS)}$$

für $i = 1, \dots, p$ berechnet werden, wobei $y_0^{(n)}(a_1) = y_{p+1}^{(n)}(b_p) = 0$ für alle $n \in \mathbb{N}$ gesetzt wird. Das eindimensionale **p**arallele **S**chwarz–Verfahren (**PS**) [64, 55], das aus Verfahren 4.1 hervorgeht und im folgenden untersucht werden soll, kann durch

$$\left\{ \begin{array}{rcl} L\left(y_i^{(n+1)}\right) & = & f \text{ in } (a_i, b_i) \\ y_i^{(n+1)}(a_i) & = & y_{i-1}^{(n)}(a_i) \\ y_i^{(n+1)}(b_i) & = & y_{i+1}^{(n)}(b_i) \\ & \text{für} & i = 1, \dots, p \end{array} \right\} \qquad \textbf{(PS)}$$

mit $y_0^{(n)}(a_1) = y_{p+1}^{(n)}(b_p) = 0$ vereinfacht dargestellt werden. Hier werden alle Teilprobleme in einem *Gesamtschritt* gleichzeitig gelöst. Für einen Parallelrechner bedeutet dies, daß die p Teilprobleme auf p Prozessoren gelöst werden können und die Kommunikation der einzelnen Prozessoren lediglich aus dem Austausch der verschiedenen Randwerte besteht, so daß eine Implementierung besonders einfach ist. Wie schon erwähnt, ist jedoch die langsame Konvergenz ein Nachteil des parallelen Verfahrens. Um schneller konvergierende Varianten einführen zu können, wird eine Darstellung in Matrizenform benötigt. Sei hierfür der Differentialoperator L so gewählt, daß Lösungen von $L(y) = f$ in (a_i, b_i) mit Dirichlet–Randbedingungen existieren und eindeutig sind. Bezeichnet man insbesondere mit

$$\begin{array}{ll} \varphi_i : & \text{Lösung der DGl } L(y) = 0 \text{ in } (a_i, b_i) \text{ mit } y(a_i) = 1 \text{ und } y(b_i) = 0, \\ \psi_i : & \text{Lösung der DGl } L(y) = 0 \text{ in } (a_i, b_i) \text{ mit } y(a_i) = 0 \text{ und } y(b_i) = 1, \\ h_i : & \text{Lösung der DGl } L(y) = f \text{ in } (a_i, b_i) \text{ mit } y(a_i) = y(b_i) = 0, \end{array} \qquad (4.5)$$

so sind die Lösungen von $L(y_i) = f$ in (a_i, b_i) mit den Randwerten $y_i(a_i)$ und $y_i(b_i)$ durch

$$y_i(x) = h_i(x) + \varphi_i(x) y_i(a_i) + \psi_i(x) y_i(b_i) \quad \text{für} \quad x \in [a_i, b_i] \tag{4.6}$$

gegeben. Das parallele Schwarz–Verfahren läßt sich allein für die auszutauschenden Randwerte formulieren, falls man beachtet, daß sich die Werte an den Rändern a und b nicht ändern, also

$$\begin{cases} y_i^{(n+1)}(b_i) & = & y_{i+1}^{(n)}(b_i) \\ y_{i+1}^{(n+1)}(a_{i+1}) & = & y_i^{(n)}(a_{i+1}) \end{cases} \quad , i = 1, \ldots, p-1. \tag{4.7}$$

Da $y_{i+1}^{(n)}$ die Lösung von $L(y) = f$ im Intervall (a_{i+1}, b_{i+1}) mit den Randwerten $y_{i+1}^{(n)}(a_{i+1})$ und $y_{i+1}^{(n)}(b_{i+1})$ ist, ergibt sich mit Hilfe von (4.6)

$$y_{i+1}^{(n)}(b_i) = h_{i+1}(b_i) + \varphi_{i+1}(b_i) y_{i+1}^{(n)}(a_{i+1}) + \psi_{i+1}(b_i) y_{i+1}^{(n)}(b_{i+1}) \,. \tag{4.8}$$

Entsprechend erhält man

$$y_i^{(n)}(a_{i+1}) = h_i(a_{i+1}) + \varphi_i(a_{i+1}) y_i^{(n)}(a_i) + \psi_i(a_{i+1}) y_i^{(n)}(b_i) \,. \tag{4.9}$$

Setzt man für $i = 1, \ldots, p-1$

$$\mathbf{u}_i = \begin{pmatrix} y_i(b_i) \\ y_{i+1}(a_{i+1}) \end{pmatrix}, \qquad \mathbf{f}_i = \begin{pmatrix} h_{i+1}(b_i) \\ h_i(a_{i+1}) \end{pmatrix},$$

$$\mathbf{D}_i = \begin{pmatrix} 1 & -\varphi_{i+1}(b_i) \\ -\psi_i(a_{i+1}) & 1 \end{pmatrix},$$

und für $i = 1, \ldots, p-2$

$$\mathbf{L}_i = \begin{pmatrix} 0 & 0 \\ 0 & -\varphi_{i+1}(a_{i+2}) \end{pmatrix}, \quad \mathbf{R}_{i+1} = \begin{pmatrix} -\psi_{i+1}(b_i) & 0 \\ 0 & 0 \end{pmatrix},$$

so können die Gleichungen (4.8) und (4.9) zusammengefaßt werden zu

$$\begin{pmatrix} y_{i+1}(b_i) \\ y_i(a_{i+1}) \end{pmatrix}^{(n)} = \mathbf{f}_i + (\mathbf{I} - \mathbf{D}_i)\mathbf{u}_i^{(n)} - \mathbf{L}_{i-1}\mathbf{u}_{i-1}^{(n)} - \mathbf{R}_{i+1}\mathbf{u}_{i+1}^{(n)} \,, \tag{4.10}$$

wobei $\mathbf{L}_0$ und $\mathbf{R}_p$ entsprechende Nullmatrizen bezeichnen. Somit ergibt sich aus (4.7) das Iterationsverfahren

$$\mathbf{u}_i^{(n+1)} = \mathbf{f}_i + (\mathbf{I} - \mathbf{D}_i)\mathbf{u}_i^{(n)} - \mathbf{L}_{i-1}\mathbf{u}_{i-1}^{(n)} - \mathbf{R}_{i+1}\mathbf{u}_{i+1}^{(n)} \,.$$

Mit den Bezeichnungen

$$\mathbf{u} = \begin{pmatrix} \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{p-1} \end{pmatrix}, \qquad \mathbf{f} = \begin{pmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_{p-1} \end{pmatrix}$$

und

$$\mathbf{A} = \begin{pmatrix} \mathbf{D}_1 & \mathbf{R}_2 & & & \\ \mathbf{L}_1 & \mathbf{D}_2 & \mathbf{R}_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{L}_{p-3} & \mathbf{D}_{p-2} & \mathbf{R}_{p-1} \\ & & & \mathbf{L}_{p-2} & \mathbf{D}_{p-1} \end{pmatrix} \tag{4.11}$$

ist (4.10) äquivalent zu

$$\mathbf{u}^{(n+1)} = \mathbf{f} + \mathbf{u}^{(n)} - \mathbf{A}\mathbf{u}^{(n)} = \mathbf{f} + \left(\mathbf{I} - \mathbf{A}\right)\mathbf{u}^{(n)}$$

mit der Iterationsmatrix

$$\mathbf{I} - \mathbf{A}\,.$$

Dieses ist aber das Jacobi–Verfahren zur Lösung des Gleichungssystems $\mathbf{A}\mathbf{u} = \mathbf{f}$, dessen Lösung durch $\mathbf{u} = \left(y(b_1), y(a_2), \ldots, y(b_{p-1}), y(a_p)\right)^T$ gegeben ist, so daß im Fall der Konvergenz des Verfahrens die Iterierten $\mathbf{u}^{(n)}$ gegen die exakte Lösung konvergieren. Da die Matrix $\mathbf{A}$ blocktridiagonal ist, ergibt sich das zugehörige Block–Jacobi–Verfahren durch

$$\mathbf{D}_i\mathbf{u}_i^{(n+1)} = \mathbf{f}_i - \mathbf{L}_{i-1}\mathbf{u}_{i-1}^{(n)} - \mathbf{R}_{i+1}\mathbf{u}_{i+1}^{(n)}\,,$$

bzw. in der kompakteren Form als

$$\mathbf{u}^{(n+1)} = \mathbf{u}^{(n)} + \mathbf{D}_A^{-1}\left(\mathbf{f} - \mathbf{A}\mathbf{u}^{(n)}\right)\,,$$

falls

$$\mathbf{D}_A = \operatorname{diag}(\mathbf{D}_1, \ldots, \mathbf{D}_{p-1}) \tag{4.12}$$

gesetzt wird. In diesem Fall lautet die Iterationsmatrix

$$\mathbf{I} - \mathbf{D}_A^{-1}\mathbf{A}\,.$$

Ersetzt man $\mathbf{D}_A^{-1}$ durch die neu eingeführte Matrix $\mathbf{K} = \operatorname{diag}(\mathbf{K}_1, \ldots, \mathbf{K}_{p-1})$ mit regulären Matrizen

$$\mathbf{K}_i = \begin{pmatrix} \mu_i & \alpha_i \\ \beta_{i+1} & \nu_{i+1} \end{pmatrix}, \quad i = 1, \ldots, p-1,$$

erhält man das Richardson–Verfahren zur Lösung des vorkonditionierten Gleichungssystem $\mathbf{K}\mathbf{A}\mathbf{u} = \mathbf{K}\mathbf{f}$, also

$$\mathbf{u}^{(n+1)} = \mathbf{u}^{(n)} + \mathbf{K}\left(\mathbf{f} - \mathbf{A}\mathbf{u}^{(n)}\right)$$

mit der Iterationsmatrix

$$\mathbf{I} - \mathbf{K}\mathbf{A}\,.$$

Für die einzelnen Blöcke ergibt sich hieraus

$$\begin{aligned}\mathbf{u}_i^{(n+1)} &= \mathbf{u}_i^{(n)} + \mathbf{K}_i \left(\mathbf{f}_i - \mathbf{D}_i\mathbf{u}_i^{(n)} - \mathbf{L}_{i-1}\mathbf{u}_{i-1}^{(n)} - \mathbf{R}_{i+1}\mathbf{u}_{i+1}^{(n)}\right) \\ &= \mathbf{u}_i^{(n)} + \mathbf{K}_i \left(\mathbf{f}_i + (\mathbf{I} - \mathbf{D}_i)\mathbf{u}_i^{(n)} - \mathbf{L}_{i-1}\mathbf{u}_{i-1}^{(n)} - \mathbf{R}_{i+1}\mathbf{u}_{i+1}^{(n)}\right) - \mathbf{K}_i\mathbf{u}_i^{(n)} .\end{aligned}$$

Mit (4.10) und der Definition von $\mathbf{u}_i$ gilt schließlich

$$\begin{pmatrix} y_i(b_i) \\ y_{i+1}(a_{i+1}) \end{pmatrix}^{(n+1)} = (\mathbf{I} - \mathbf{K}_i) \begin{pmatrix} y_i(b_i) \\ y_{i+1}(a_{i+1}) \end{pmatrix}^{(n)} + \mathbf{K}_i \begin{pmatrix} y_{i+1}(b_i) \\ y_i(a_{i+1}) \end{pmatrix}^{(n)} .$$

Man erhält so das vor**k**onditionierte **e**xplizite **p**arallele **S**chwarz–Verfahren

$$\left\{\begin{array}{rcll} L\left(y_i^{(n+1)}\right) &=& f & \text{in } (a_i, b_i) \\ y_i^{(n+1)}(a_i) &=& y_i^{(n)}(a_i) & + \nu_i \left(y_{i-1}^{(n)}(a_i) - y_i^{(n)}(a_i)\right) \\ & & & + \beta_i \left(y_i^{(n)}(b_{i-1}) - y_{i-1}^{(n)}(b_{i-1})\right) \\ y_i^{(n+1)}(b_i) &=& y_i^{(n)}(b_i) & + \mu_i \left(y_{i+1}^{(n)}(b_i) - y_i^{(n)}(b_i)\right) \\ & & & + \alpha_i \left(y_i^{(n)}(a_{i+1}) - y_{i+1}^{(n)}(a_{i+1})\right) \\ & & & \text{für } i = 1, \ldots, p \end{array}\right\} \quad \textbf{(KEPS)},$$

wobei $\nu_1 = \beta_1 = \mu_p = \alpha_p = 0$ und $y_1^{(n)}(a_1) = y_p^{(n)}(b_p) = 0$ für alle $n \in \mathbb{N}$. Dieses Verfahren stimmt für $\nu_i = \mu_{i-1} = 1$ und $\beta_i = \alpha_{i-1} = \alpha$, $i = 2, \ldots, p$, mit dem sogenannten gekoppelten Verfahren aus [54] überein und soll an dieser Stelle mit

$$\left\{\begin{array}{rcll} L\left(y_i^{(n+1)}\right) &=& f & \text{in } (a_i, b_i) \\ y_i^{(n+1)}(a_i) &=& y_{i-1}^{(n)}(a_i) & + \alpha \left(y_i^{(n)}(b_{i-1}) - y_{i-1}^{(n)}(b_{i-1})\right) \\ y_i^{(n+1)}(b_i) &=& y_{i+1}^{(n)}(b_i) & + \alpha \left(y_i^{(n)}(a_{i+1}) - y_{i+1}^{(n)}(a_{i+1})\right) \\ & & & \text{für } i = 1, \ldots, p \end{array}\right\} \quad \textbf{(KEPS')}$$

bezeichnet werden. Man beachte, daß der Kommunikationsaufwand bei der Implementierung von KEPS gegenüber der Methode PS nicht steigt. Es werden lediglich Linearkombinationen der verschiedenen Randwerte ausgetauscht.

Für eine gleichmäßige (ε, p)–überlappende Zerlegung des Intervalls $(0, 1)$ mit der relativen Überlappung $\varepsilon/(b_1 - a_1) = 0.25$ wurden die Spektralradien der verschiedenen Iterationsmatrizen in Abhängigkeit des Parameters α des KEPS' für die Differentialgleichung

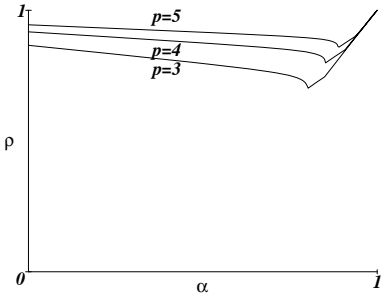


Abbildung 4.2: Die Spektralradien von KEPS' in Abhängigkeit von α.

$y'' = f$ berechnet und für $p = 3, 4, 5$ Teilintervalle in Abbildung 4.2 dargestellt.

Es soll nun das KEPS-Verfahren für die Blöcke $\mathbf{K}_i = \mathbf{D}_i^{-1}$, $i = 1, \ldots, p-1$, mit dem parallelen Schwarz–Verfahren verglichen werden. Dieses spezielle Verfahren wird im folgenden als **b**lock-**e**xplizites **p**aralleles **S**chwarz–Verfahren (**BEPS**) bezeichnet.

Ziel ist es zu zeigen, daß das block-explizite parallele Verfahren schneller konvergiert als das parallele Schwarz–Verfahren und für hinreichend kleine Überlappungen sogar schneller ist als das alternierende Schwarz–Verfahren. Dazu werden einige Hilfsmittel aus der Theorie der M–Matrizen benötigt, vergleiche [40], Kapitel 6.

Definition 4.1.2. *Eine Matrix* $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N \times N}$ *heißt M–Matrix, falls*

$$a_{ii} > 0 \quad \textit{und} \quad a_{ij} \leq 0 \qquad \textit{für } i \neq j \,,$$

$\mathbf{A}$ *regulär ist und* $\mathbf{A}^{-1} \geq 0$ *gilt.*

Definition 4.1.3. *Eine Matrix* $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N \times N}$ *heißt schwach diagonaldominant, falls*

$$\sum_{\substack{1 \leq j \leq N \\ j \neq i}} |a_{ij}| \quad \leq \quad |a_{ii}| \qquad \textit{für alle } i = 1, \ldots, N \,.$$

Definition 4.1.4. *Eine Matrix* $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N \times N}$ *heißt irreduzibel, falls ein* $m \in \mathbb{N}$ *existiert mit* $1 = i_0, i_1, \ldots, i_m = 1$*, so daß*

$$a_{i_{k-1} i_k} \neq 0 \qquad \textit{für } k = 1, \ldots, m$$

und

$$\bigcup_{k=1}^{m} \{i_k\} \quad = \quad \{1, \ldots, N\} \,.$$

Dieser Definition entspricht die Tatsache, daß der Graph einer irreduziblen Matrix zusammenhängend ist.

Definition 4.1.5. *Eine Matrix* $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N \times N}$ *heißt irreduzibel diagonaldominant, falls* $\mathbf{A}$ *irreduzibel und schwach diagonaldominant ist, und außerdem mindestens ein Index* $i \in \{1, \ldots, N\}$ *existiert mit*

$$\sum_{\substack{1 \leq j \leq N \\ j \neq i}} |a_{ij}| \quad < \quad |a_{ii}| \,.$$

Satz 4.1.1. *Ist* $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N \times N}$ *irreduzibel diagonaldominant mit*

$$a_{ii} > 0 \quad \textit{und} \quad a_{ij} \leq 0 \qquad \textit{für } i \neq j \,,$$

so ist $\mathbf{A}$ *eine M–Matrix mit* $\mathbf{A}^{-1} > 0$.

Satz 4.1.2. *Sei* $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N\times N}$ *eine M–Matrix. Bezeichnen* $\mathbf{J}$ *und* $\mathbf{B}$ *die jeweiligen Iterationsmatrizen des Jacobi– bzw. Block–Jacobi–Verfahrens, so gilt für den Spektralradius* ρ

$$\rho(\mathbf{B}) \leq \rho(\mathbf{J}) < 1\,.$$

Gilt darüber hinaus $\mathbf{A}^{-1} > \mathbf{0}$ *und* $\mathbf{J} \neq \mathbf{B}$*, so folgt*

$$\rho(\mathbf{B}) < \rho(\mathbf{J}) < 1\,.$$

Mit diesen Hilfsmitteln ergibt sich der nachfolgende Satz.

Satz 4.1.3. *Sind* φ_i, ψ_i *aus* (4.5) *positive Funktionen in* (a_i, b_i) *mit*

$$\varphi_i(x) + \psi_i(x) \leq 1 \qquad \textit{für } x \in (a_i, b_i)\,,$$

so ist $\mathbf{A}$ *aus* (4.11) *eine M–Matrix mit positiver Inverser.*

Beweis. Da die Indizes $1, 3, \ldots, 2p-3, 2p-2, 2p-4, \ldots, 2, 1$ die Bedingung von Definition 4.1.4 erfüllen, ist die Matrix $\mathbf{A}$ irreduzibel. Weil auch die weiteren Voraussetzungen von Satz 4.1.1 erfüllt sind, ergibt sich hieraus die Behauptung. □

Aus Satz 4.1.3 in Verbindung mit Satz 4.1.2 ergibt sich als wesentliche Aussage dieses Abschnittes der folgende Satz.

Satz 4.1.4. *Gegeben sei die lineare Differentialgleichung*

$$L(y) = f \qquad \textit{in } \ (a, b)$$

mit verschwindenden Dirichlet–Randbedingungen. Sei $\{(a_i, b_i)\,,\ i = 1, \ldots, p\}$ *eine* p*–überlappende Zerlegung von* (a, b) *und die Funktionen* φ_i, ψ_i *aus* (4.5) *mögen die Voraussetzungen von Satz 4.1.3 erfüllen. Bezeichnen* $\mathbf{J}$ *und* $\mathbf{B}$ *die Iterationsmatrizen des parallelen bzw. block-expliziten parallelen Schwarz–Verfahrens, so gilt*

$$\rho(\mathbf{B}) < \rho(\mathbf{J}) < 1\,.$$

Genau wie oben läßt sich zeigen, daß das alternierende Schwarz–Verfahren dem Gauß–Seidel–Verfahren zur Lösung des Gleichungssystems $\mathbf{A}\,\mathbf{u} = \mathbf{f}$ entspricht. Der folgende Satz gibt nun Aufschluß über die Verringerung der Konvergenzgeschwindigkeit beim Übergang vom alternierenden zum parallelen Schwarz–Verfahren ([79], Korollar 5.2.3).

Satz 4.1.5. *Bezeichnen* $\mathbf{G}$ *und* $\mathbf{J}$ *die jeweiligen Iterationsmatrizen des alternierenden bzw. parallelen Schwarz–Verfahrens, so gilt*

$$\rho(\mathbf{G}) = \rho(\mathbf{J})^2\,.$$

Das alternierende Schwarz–Verfahren ist folglich doppelt so schnell wie das parallele Schwarz–Verfahren.

Bemerkung 4.1.1. *Gilt* $a_{i+1} = b_i$ *für ein* $i \in \{1, \ldots, p-1\}$*, so ist die Matrix* $\mathbf{A}$ *aus* (4.11) *singulär, und die Iterationsmatrix* $\mathbf{I} - \mathbf{A}$ *des Jacobi-Verfahrens hat somit den Eigenwert* 1*, d.h. sowohl das parallele als auch das alternierende Schwarz–Verfahren divergieren, vergleiche Satz 4.1.5.*

4.2 Beispiele

In diesem Abschnitt werden Ergebnisse aus [69] zur Lösung der Wärmeleitungsgleichung aufgegriffen. Es wird gezeigt, daß das block-explizite parallele Schwarz–Verfahren bezüglich der Differentialoperatoren

$$L_1(y) = y'' \qquad \text{in} \quad (a,b) \tag{4.13}$$

und

$$L_2(y) = y'' - \kappa^2 y \qquad \text{in} \quad (a,b) \tag{4.14}$$

schneller konvergiert als das parallele Schwarz–Verfahren. Hierzu sei eine p–überlappende Zerlegung $\{(a_i,b_i)\,,\ i=1,\ldots,p\}$ des Intervalls (a,b) gegeben. Dann erhält man für die homogenen Differentialgleichungen, vergleiche (4.5), im ersten Fall die Lösungen

$$\varphi_i(x) = \frac{b_i - x}{b_i - a_i} \quad \text{und} \quad \psi_i(x) = \frac{x - a_i}{b_i - a_i} \quad \text{für } x \in (a_i,b_i)$$

und im zweiten Fall

$$\varphi_i(x) = \frac{\sinh(\kappa(b_i - x))}{\sinh(\kappa(b_i - a_i))} \quad \text{und} \quad \psi_i(x) = \frac{\sinh(\kappa(x - a_i))}{\sinh(\kappa(b_i - a_i))} \quad \text{für } x \in (a_i,b_i)\,.$$

Da die Funktionen φ_i und ψ_i in beiden Fällen positiv sind und die Ungleichung $\varphi_i(x) + \psi_i(x) \leq 1$ für $x \in (a_i,b_i)$ erfüllen, ergibt sich unmittelbar aus Satz 4.1.4 der folgende Satz.

Satz 4.2.1. *Gegeben seien die linearen Differentialgleichungen*

$$L_k(y) = f \qquad \textit{in } (a,b) \quad \textit{für } k = 1,2 \ \textit{gemäß } (4.13) \textit{ und } (4.14)$$

mit verschwindenden Dirichlet-Randbedingungen. Sei $\{(a_i,b_i)\,,\ i=1,\ldots,p\}$ eine p–überlappende Zerlegung von (a,b) und bezeichnen $\mathbf{J}_k$ und $\mathbf{B}_k$ die Iterationsmatrizen des parallelen bzw. block-expliziten parallelen Schwarz–Verfahrens, so gilt

$$\rho(\mathbf{B}_k) < \rho(\mathbf{J}_k) < 1 \quad \textit{für } k = 1,2\,.$$

Im vorigen Abschnitt ergab sich, daß weder das parallele noch das alternierende Schwarz–Verfahren konvergieren, wenn sich mindestens zwei Teilintervalle nicht überlappen. Betrachtet man den Grenzübergang ε gegen Null einer gleichmäßigen (ε,p)–überlappenden Zerlegung des Intervalls (a,b), so erhält man beim block-expliziten parallelen Schwarz–Verfahren die folgende Aussage. Obwohl dieses Verfahren für $\varepsilon = 0$ nicht existiert, da die Matrix $\mathbf{D}_A$ in diesem Fall singulär ist, existieren dennoch die entsprechenden Iterationsmatrizen für die Differentialoperatoren L_1 und L_2. Es wird sich zeigen, daß die Spektralradien dieser Matrizen auch für $\varepsilon = 0$ kleiner als Eins sind. Aufgrund der Stetigkeit des Spektralradius bedeutet dies aber, daß für kleine Überlappungen das block-explizite parallele Schwarz–Verfahren besser ist als das entsprechende alternierende Schwarz–Verfahren. Dieser Sachverhalt wird nun in der nachfolgenden Bemerkung genauer beschrieben.

Bemerkung 4.2.1. *Sei* $\{(a_i, b_i)\,,\ i = 1, \ldots, p\}$ *eine gleichmäßige* (ε, p)*–überlappende Zerlegung des Intervalls* (a, b)*. Bezeichnen* $\mathbf{B}_1$ *und* $\mathbf{B}_2$ *die Iterationsmatrizen des block-expliziten parallelen Schwarz–Verfahrens für* $\varepsilon \to 0$ *bezüglich der Differentialoperatoren* L_1 *bzw.* L_2*, dann gilt*

$$\rho(\mathbf{B}_2) = \frac{1}{\cosh\left(\frac{\kappa(b-a)}{p}\right)}\,\rho(\mathbf{B}_1) \leq \rho(\mathbf{B}_1) < 1\,.$$

Beweis. Unter der Voraussetzung einer gleichmäßigen (ε, p)–überlappenden Zerlegung erhält man aus $\mathbf{I} - \mathbf{D}_A^{-1}\mathbf{A}$ mit $\mathbf{A}$ aus (4.11) für $\varepsilon \to 0$ die Iterationsmatrizen

$$\mathbf{B}_1 = \begin{pmatrix} \mathbf{0} & \mathbf{R} & & & \\ \mathbf{L} & \mathbf{0} & \mathbf{R} & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{L} & \mathbf{0} & \mathbf{R} \\ & & & \mathbf{L} & \mathbf{0} \end{pmatrix} \in \mathbb{R}^{(2p-2)\times(2p-2)}$$

mit

$$\mathbf{L} = \begin{pmatrix} 0 & \frac{1}{2} \\ 0 & \frac{1}{2} \end{pmatrix} \qquad \text{und} \qquad \mathbf{R} = \begin{pmatrix} \frac{1}{2} & 0 \\ \frac{1}{2} & 0 \end{pmatrix}$$

und

$$\mathbf{B}_2 = \frac{1}{\cosh\left(\frac{\kappa(b-a)}{p}\right)}\,\mathbf{B}_1\,.$$

Da $\rho(\alpha\mathbf{C}) = |\alpha|\rho(\mathbf{C})$ für $\alpha \in \mathbb{C}$ und beliebige Matrizen $\mathbf{C}$ gilt, folgt

$$\rho(\mathbf{B}_2) = \frac{1}{\cosh\left(\frac{\kappa(b-a)}{p}\right)}\,\rho(\mathbf{B}_1) \leq \rho(\mathbf{B}_1)\,.$$

Somit genügt es zu zeigen, daß $\rho(\mathbf{B}_1) < 1$. Bezeichne $\mathbf{J}$ die Matrix, die man aus $\mathbf{B}_1$ durch Streichen der ersten und letzten Zeile und Spalte erhält. Durch Entwicklung der Matrix $\lambda\mathbf{I} - \mathbf{B}_1$ nach der ersten und letzten Spalte ergibt sich

$$\det(\lambda\mathbf{I} - \mathbf{B}_1) = \lambda^2 \det(\lambda\mathbf{I} - \mathbf{J})\,,$$

so daß

$$\rho(\mathbf{B}_1) = \rho(\mathbf{J})\,.$$

Die Matrix $\mathbf{J}$ ist die Iterationsmatrix des Jacobi–Verfahrens für $\mathbf{C} = (c_{ij}) := \mathbf{I} - \mathbf{J}$. Da $\mathbf{C}$ mit der Indexfolge $1, 2, 4, \ldots, 2p-4, 2p-5, 2p-7, \ldots, 3, 1$ nach Definition 4.1.4 und Definition 4.1.5 wegen $c_{ii} = 1$ und $c_{ij} \leq 0$ für $i \neq j$ irreduzibel diagonaldominant ist, folgt aus Satz 4.1.1 in Verbindung mit Satz 4.1.2 die Behauptung. □

Für eine gleichmäßige (ε, p)–überlappende Zerlegung des Intervalls (a, b) wurden die Spektralradien der verschiedenen Iterationsmatrizen in Abhängigkeit von der relativen Überlappung $z = \varepsilon/(b_1 - a_1) \in (0,1)$ mit Hilfe von Maple für die Differentialgleichung $y'' = f$ numerisch berechnet und geplottet. Es stellt sich heraus, siehe Abbildung 4.3 bis 4.5, daß das block-explizite parallele Schwarz–Verfahren insbesondere für kleine Überlappungen den anderen Verfahren überlegen ist, vergleiche Bemerkung 4.2.1.

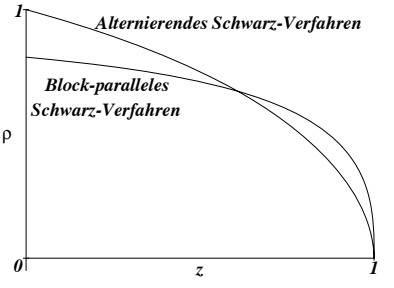


Abbildung 4.3: Die Spektralradien für $p = 5$ in Abhängigkeit von der relativen Überlappung z beim alternierenden und block-expliziten parallelen Schwarz–Verfahren.

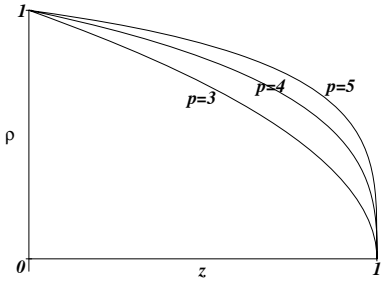


Abbildung 4.4: Der Spektralradius des parallelen Schwarz–Verfahrens in Abhängigkeit von der relativen Überlappung z.

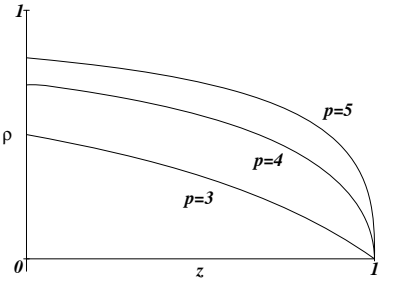


Abbildung 4.5: Der Spektralradius des block-expliziten parallelen Schwarz–Verfahrens in Abhängigkeit von der relativen Überlappung z.

4.3 Das implizite parallele Schwarz–Verfahren

In Abschnitt 4.1 erhielt man das block-parallele Schwarz–Verfahren, indem das vorkonditionierte Gleichungssystem

$$\mathbf{D}_A^{-1}\mathbf{A}\mathbf{u} = \mathbf{D}_A^{-1}\mathbf{f}$$

mit Hilfe des Jacobi–Verfahrens gelöst wurde. Die Matrix $\mathbf{D}_A^{-1}\mathbf{A}$ weist dabei die Besetzungsstruktur

auf. Somit liegt es nahe, das Gleichungssystem mit der Zerlegung $\mathbf{D}_A^{-1}\mathbf{A} = \mathbf{W}^1 - \mathbf{W}^2$ mit

$$\mathbf{W}^1 = \begin{pmatrix} * & & & & & & \\ & * & * & & & & \\ & * & * & & & & \\ & & & \ddots & & & \\ & & & & * & * & \\ & & & & * & * & \\ & & & & & & * \end{pmatrix}$$

und dem Iterationsverfahren

$$\mathbf{W}^1\mathbf{u}^{(n+1)} = \mathbf{W}^2\mathbf{u}^{(n)} + \mathbf{D}_A^{-1}\mathbf{f}$$

zu lösen. Allgemeiner wird wie im zweiten Abschnitt die Matrix $\mathbf{D}_A^{-1}$ durch eine reguläre Matrix $\mathbf{K}$ ersetzt und dann eine entsprechende Zerlegung der Matrix $\mathbf{KA}$ durchgeführt. Werden hierzu für $i = 1, \ldots, p-1$ die Matrizen

$$\mathbf{D}_i^1 = \begin{pmatrix} \mu_i - \alpha_i\psi_i(a_{i+1}) & 0 \\ 0 & \nu_{i+1} - \beta_{i+1}\varphi_{i+1}(b_i) \end{pmatrix},$$

$$\mathbf{D}_i^2 = \begin{pmatrix} 0 & \mu_i\varphi_{i+1}(b_i) - \alpha_i \\ \nu_{i+1}\psi_i(a_{i+1}) - \beta_{i+1} & 0 \end{pmatrix}$$

und für $i = 2, \ldots, p-1$ die Matrizen

$$\mathbf{R}_i^1 = \begin{pmatrix} 0 & 0 \\ -\beta_i\psi_i(b_{i-1}) & 0 \end{pmatrix}, \qquad \mathbf{R}_i^2 = \begin{pmatrix} \mu_{i-1}\psi_i(b_{i-1}) & 0 \\ 0 & 0 \end{pmatrix},$$

$$\mathbf{L}_{i-1}^1 = \begin{pmatrix} 0 & -\alpha_i\varphi_i(a_{i+1}) \\ 0 & 0 \end{pmatrix}, \quad \mathbf{L}_{i-1}^2 = \begin{pmatrix} 0 & 0 \\ 0 & \nu_{i+1}\varphi_i(a_{i+1}) \end{pmatrix}$$

definiert, ergibt sich

$$\begin{aligned} \mathbf{K}_i\mathbf{D}_i &= \mathbf{D}_i^1 - \mathbf{D}_i^2\,, & i &= 1, \ldots, p-1\,, \\ \mathbf{K}_i\mathbf{R}_{i+1} &= \mathbf{R}_{i+1}^1 - \mathbf{R}_{i+1}^2\,, & i &= 1, \ldots, p-2\,, \\ \mathbf{K}_i\mathbf{L}_{i-1} &= \mathbf{L}_{i-1}^1 - \mathbf{L}_{i-1}^2\,, & i &= 2, \ldots, p-1\,. \end{aligned}$$

Mit den Definitionen

$$\mathbf{W}_K^1 = \begin{pmatrix} \mathbf{D}_1^1 & \mathbf{R}_2^1 & & & \\ \mathbf{L}_1^1 & \mathbf{D}_2^1 & \mathbf{R}_3^1 & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{L}_{p-3}^1 & \mathbf{D}_{p-2}^1 & \mathbf{R}_{p-1}^1 \\ & & & \mathbf{L}_{p-2}^1 & \mathbf{D}_{p-1}^1 \end{pmatrix} \quad \text{und} \quad \mathbf{W}_K^2 = \begin{pmatrix} \mathbf{D}_1^2 & \mathbf{R}_2^2 & & & \\ \mathbf{L}_1^2 & \mathbf{D}_2^2 & \mathbf{R}_3^2 & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{L}_{p-3}^2 & \mathbf{D}_{p-2}^2 & \mathbf{R}_{p-1}^2 \\ & & & \mathbf{L}_{p-2}^2 & \mathbf{D}_{p-1}^2 \end{pmatrix}$$

gilt schließlich

$$\mathbf{KA} = \mathbf{W}_K^1 - \mathbf{W}_K^2\,,$$

wobei $\mathbf{W}_K^1$ die geforderte Struktur besitzt, so daß die Iterierten des induzierten Verfahrens

$$\mathbf{W}_K^1 \mathbf{u}^{(n+1)} = \mathbf{W}_K^2 \mathbf{u}^{(n)} + \mathbf{K}\mathbf{f}$$

leicht zu bestimmen sind. Für die Komponenten $\mathbf{u}_i^{(n)}$ von $\mathbf{u}^{(n)}$ ist dies gleichbedeutend mit

$$\mathbf{L}_{i-1}^1 \mathbf{u}_{i-1}^{(n+1)} + \mathbf{D}_i^1 \mathbf{u}_i^{(n+1)} + \mathbf{R}_{i+1}^1 \mathbf{u}_{i+1}^{(n+1)} = \mathbf{L}_{i-1}^2 \mathbf{u}_{i-1}^{(n)} + \mathbf{D}_i^2 \mathbf{u}_i^{(n)} + \mathbf{R}_{i+1}^2 \mathbf{u}_{i+1}^{(n)} + \mathbf{K}_i \mathbf{f}_i \,.$$

Folglich lautet die erste Zeile dieser Gleichung

$$\begin{aligned} &\mu_i y_i^{(n+1)}(b_i) - \alpha_i \varphi_i(a_{i+1}) y_i^{(n+1)}(a_i) - \alpha_i \psi_i(a_{i+1}) y_i^{(n+1)}(b_i) - \alpha_i h_i(a_{i+1}) \\ &= \mu_i \varphi_{i+1}(b_i) y_{i+1}^{(n)}(a_{i+1}) + \mu_i \psi_{i+1}(b_i) y_{i+1}^{(n)}(b_{i+1}) + \mu_i h_{i+1}(b_i) - \alpha_i y_{i+1}^{(n)}(a_{i+1}) \,. \end{aligned}$$

Daraus folgt mit (4.6)

$$\mu_i y_i^{(n+1)}(b_i) - \alpha_i y_i^{(n+1)}(a_{i+1}) = \mu_i y_{i+1}^{(n)}(b_i) - \alpha_i y_{i+1}^{(n)}(a_{i+1}) \,.$$

Analog ergibt sich aus der zweiten Zeile

$$\nu_{i+1} y_{i+1}^{(n+1)}(a_{i+1}) - \beta_{i+1} y_{i+1}^{(n+1)}(b_i) = \nu_{i+1} y_i^{(n)}(a_{i+1}) - \beta_{i+1} y_i^{(n)}(b_i) \,.$$

Da die neuen Werte $y_i^{(n+1)}(b_i)$ und $y_i^{(n+1)}(a_i)$ nicht explizit gegeben sind, erhält man insgesamt das vor**k**onditionierte **i**mplizite **p**arallele **S**chwarz–Verfahren

$$\left\{ \begin{array}{rcl} L\left(y_i^{(n+1)}\right) & = & f \quad \text{in} \quad (a_i, b_i) \\ \nu_i y_i^{(n+1)}(a_i) - \beta_i y_i^{(n+1)}(b_{i-1}) & = & \nu_i y_{i-1}^{(n)}(a_i) - \beta_i y_{i-1}^{(n)}(b_{i-1}) \\ \mu_i y_i^{(n+1)}(b_i) - \alpha_i y_i^{(n+1)}(a_{i+1}) & = & \mu_i y_{i+1}^{(n)}(b_i) - \alpha_i y_{i+1}^{(n)}(a_{i+1}) \\ & \text{für} & i = 1, \ldots, p \end{array} \right\} \quad \textbf{(KIPS)},$$

wobei $\nu_1 = \mu_p = 1$, $\beta_1 = \alpha_p = 0$ und $y_0^{(n)}(a_1) = y_{p+1}^{(n)}(b_p) = 0$ für alle $n \in \mathbb{N}$. Auch hier soll betont werden, daß der Kommunikationsaufwand bei der Implementierung von KIPS gegenüber der Methode PS nicht steigt, da ausschließlich Linearkombinationen der verschiedenen Randwerte von benachbarten Intervallen ausgetauscht werden. Da KIPS für $\nu_i = 0$ bzw. $\mu_i = 0$ nicht konvergiert, sollte man stets $\nu_i \neq 0$ und $\mu_i \neq 0$ für $i = 1, \ldots, p$ wählen, so daß die entsprechenden Gleichungen durch ν_i und μ_i geteilt werden dürfen. Folglich können diese Variablen zu Eins angenommen werden. Man erhält so ein zu KIPS äquivalentes Verfahren

$$\left\{ \begin{array}{rcl} L\left(y_i^{(n+1)}\right) & = & f \quad \text{in} \quad (a_i, b_i) \\ y_i^{(n+1)}(a_i) - \beta_i y_i^{(n+1)}(b_{i-1}) & = & y_{i-1}^{(n)}(a_i) - \beta_i y_{i-1}^{(n)}(b_{i-1}) \\ y_i^{(n+1)}(b_i) - \alpha_i y_i^{(n+1)}(a_{i+1}) & = & y_{i+1}^{(n)}(b_i) - \alpha_i y_{i+1}^{(n)}(a_{i+1}) \\ & \text{für} & i = 1, \ldots, p \end{array} \right\} \quad \textbf{(KIPS')}$$

mit $\beta_1 = \alpha_p = 0$, das für die Wahl der Koeffizienten $\beta_i = \alpha_{i-1} = 0$ für $i = 2, \ldots, p$ mit dem parallelen Schwarz–Verfahren übereinstimmt. Gilt darüber hinaus in KIPS' auch $\beta_i = \alpha_{i-1} = \alpha$ für $i = 2, \ldots, p$, so bezeichnet man dieses Verfahren mit **KIPS''**.
Wie bei der Beschreibung zu Abbildung 4.2 sind auch hier die Spektralradien in Abhängigkeit des Parameters α für KIPS'' in Abbildung 4.6 dargestellt.

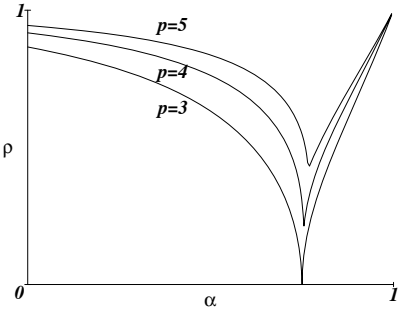


Abbildung 4.6: Die Spektralradien von KIPS'' in Abhängigkeit von α.

Bezeichnet man das KIPS-Verfahren mit $\mathbf{K}_i = \mathbf{D}_i^{-1}$ für $i = 1, \ldots, p-1$ als **b**lock-**i**mplizites **p**aralleles **S**chwarz–Verfahren (**BIPS**), so werden im folgenden die Verfahren BIPS und BEPS verglichen. Hierzu erweisen sich die nachfolgenden Lemmata als nützlich, vergleiche [1], Kapitel 6.

Lemma 4.3.1. *Sei $\mathbf{A} \in \mathbb{R}^{N\times N}$ eine M–Matrix und $\mathbf{B} = (b_{ij}) \geq \mathbf{A}$ mit $b_{ij} \leq 0$ für $i \neq j$. Dann ist auch $\mathbf{B}$ eine M–Matrix.*

Lemma 4.3.2. *Sei $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N\times N}$ mit $a_{ij} \leq 0$ für $i \neq j$. Dann gilt: $\mathbf{A}$ ist M–Matrix genau dann, wenn $\mathbf{Ax} > \mathbf{0}$ für ein positives $\mathbf{x} \in \mathbb{R}^N$.*

Satz 4.3.1. *Sei $\mathbf{A} = (\mathbf{A}_{ij})_{i,j=1,\ldots m}$ eine M–Matix mit Blockstruktur und $\mathbf{D}$ die Blockdiagonale, d.h. $\mathbf{D} = diag(\mathbf{A}_{11}, \ldots, \mathbf{A}_{mm})$. Dann ist auch $\mathbf{D}^{-1}\mathbf{A}$ eine M–Matrix.*

Beweis. Sei $\widetilde{\mathbf{A}} = (\widetilde{a}_{ij}) := \mathbf{D}^{-1}\mathbf{A}$. Nach Lemma 4.3.1 ist mit $\mathbf{A}$ auch $\mathbf{D}$ eine M–Matrix. Folglich gilt $\mathbf{D}^{-1} \geq 0$, so daß $\widetilde{a}_{ij} \leq 0$ für $i \neq j$. Weiter existiert mit Hilfe von Lemma 4.3.2 ein positiver Vektor x mit $\mathbf{Ax} > \mathbf{0}$. Weil $\mathbf{D}^{-1} \geq 0$ regulär ist, gibt es in jeder Zeile von $\mathbf{D}^{-1}$ mindestens ein positives Element, so daß auch $\mathbf{D}^{-1}\mathbf{Ax} > \mathbf{0}$. Hieraus folgt schließlich mit Lemma 4.3.2 die Behauptung. □

Damit kann die Hauptaussage dieses Abschnittes bewiesen werden.

Satz 4.3.2. *Gegeben sei die lineare Differentialgleichung*

$$L(y) = f \qquad in \quad (a,b)$$

mit verschwindenden Dirichlet–Randbedingungen. Sei $\{(a_i, b_i),\ i = 1, \ldots, p\}$ eine p–überlappende Zerlegung von (a,b) und erfüllen die Funktionen φ_i, ψ_i aus (4.5) *die Voraussetzungen von Satz 4.1.3. Bezeichnen $\mathbf{J}$, $\mathbf{B}_e$ und $\mathbf{B}_i$ die Iterationsmatrizen des parallelen, block-expliziten und block-impliziten parallelen Schwarz–Verfahrens, so gilt*

$$\rho(\mathbf{B}_i) \leq \rho(\mathbf{B}_e) < \rho(\mathbf{J}) < 1\,.$$

Beweis. Das block-explizite parallele Schwarz–Verfahren entspricht dem Block–Jacobi–Verfahren bezüglich der Matrix $\mathbf{A}$ mit $\mathbf{A}$ aus (4.11), bzw. dem Jacobi–Verfahren bezüglich der Matrix $\widetilde{\mathbf{A}} := \mathbf{D}_A^{-1}\mathbf{A}$ mit $\mathbf{D}_A^{-1}$ aus (4.12). Nun entspricht aber das block-implizite parallele Schwarz–Verfahren dem Block–Jacobi–Verfahren bezüglich $\widetilde{\mathbf{A}}$. Da nach Satz 4.3.1 mit $\mathbf{A}$ auch $\widetilde{\mathbf{A}}$ eine M–Matrix ist, folgt die Behauptung aus den Sätzen 4.1.2 und 4.1.4. □

Zum Abschluß des Abschnittes sind hier noch einmal zu einer gleichmäßigen (ε, p)–überlappenden Zerlegung des Intervalls (a, b) die Spektralradien der verschiedenen Iterationsmatrizen für das explizite und das implizite block-parallele Schwarz–Verfahren in Abhängigkeit von der relativen Überlappung $z = \varepsilon/(b_1 - a_1) \in (0, 1)$ für die gewöhnliche Differentialgleichung $y'' = f$ numerisch berechnet und in Abbildung 4.7 dargestellt. Dabei ist das BIPS-Verfahren dem BEPS-Verfahren deutlich überlegen.

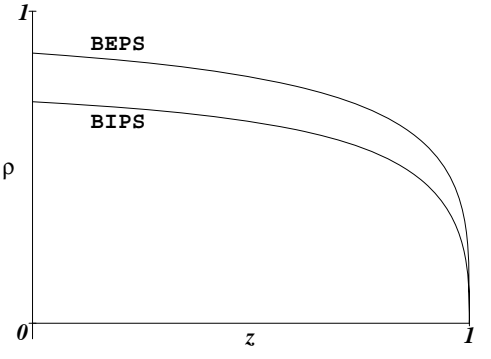


Abbildung 4.7: Die Spektralradien für $p = 6$ Teilgebiete in Abhängigkeit von der relativen Überlappung z.

Beispiel 4.3.1. *In diesem Beispiel wird die Differentialgleichung $-y'' = 2$ mit homogenen Randbedingungen auf dem Intervall $(0, 10)$ gelöst. Basierend auf einer gleichmäßigen $(0.01, 20)$-überlappenden Zerlegung des Intervalls $(0, 10)$, benötigt das parallele Schwarz–Verfahren ca. 25000 Iterationen und KIPS'' mit $\alpha = 0.995$ ca. 50 Iterationsschritte, wenn die Verfahren mit der Anfangslösung Null gestartet werden. Die nachfolgenden Abbildungen 4.8 und 4.9 zeigen einige Zwischenlösungen der verschiedenen Methoden, die mit wachsender Iterationsanzahl die Lösung besser approximieren. In beiden Fällen stellt jeweils die obere Kurve die exakte Lösung dar.*

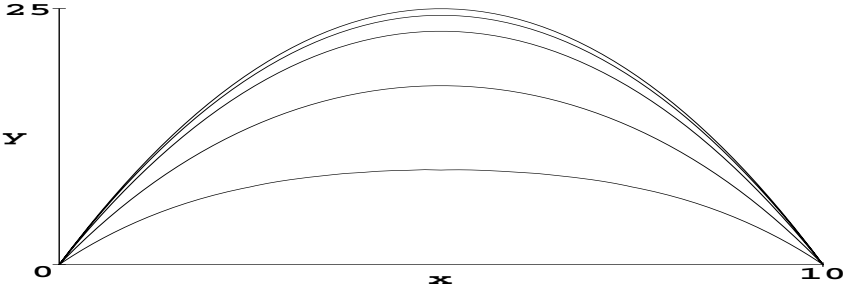


Abbildung 4.8: *Paralleles Schwarz–Verfahren nach 2000, 5000, 10000 und 15000 Iterationen.*

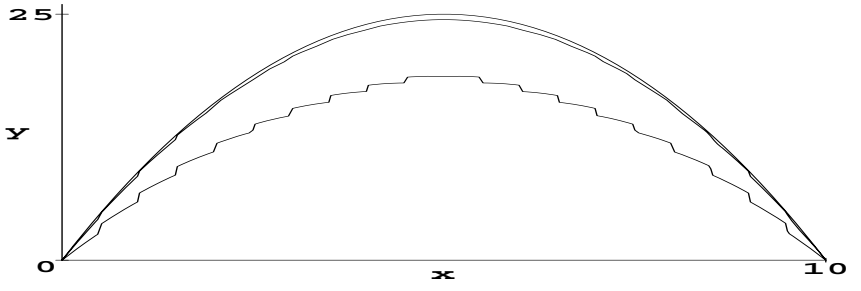


Abbildung 4.9: *KIPS'' nach 10 und 20 Iterationen.*

4.4 Das verallgemeinerte parallele Schwarz–Verfahren

Das parallele Schwarz–Verfahren tauscht an den Rändern der Intervalle (a_i, b_i) stets die Randwerte aus, also die Funktionswerte an den Stellen a_i und b_i, so daß die Teillösungen mit Hilfe von Dirichlet–Randbedingungen berechnet werden. Es sind aber z.B. auch Neumann– oder gemischte Randbedingungen möglich. Dies führt auf das **v**erallgemeinerte **p**arallele **S**chwarz–Verfahren (**VPS**)

$$\left\{ \begin{array}{rcl} L\left(y_i^{(n+1)}\right) & = & f \text{ in } (a_i, b_i) \\ L_i^l\left(y_i^{(n+1)}\right)(a_i) & = & L_i^l\left(y_{i-1}^{(n)}\right)(a_i) \\ L_i^r\left(y_i^{(n+1)}\right)(b_i) & = & L_i^r\left(y_{i+1}^{(n)}\right)(b_i) \\ & \text{für} & i = 1, \ldots, p \end{array} \right\} \quad \textbf{(VPS)},$$

wobei $L_1^l = L_p^r = 0$ ist und L_{i+1}^l, L_i^r für $i = 1, \ldots, p-1$ geeignete lineare Operatoren darstellen, die für die Funktionen aus (4.5) definiert sind. Für den Spezialfall des parallelen Schwarz–Verfahrens ist

$$L_{i+1}^l(y) = L_i^r(y) = y.$$

Neumann–Randbedingungen werden dann beispielsweise durch

$$L_{i+1}^l(y) = y' \quad \text{und} \quad L_i^r(y) = -y'$$

charakterisiert. Weitere Beispiele finden sich in [22] und [73].

Das Ziel ist es nun zu zeigen, daß die Verfahren KIPS und VPS äquivalent sind.

Satz 4.4.1. *Jedes KIPS ist ein VPS.*

Beweis. An den Endpunkten des Intervalls stimmen beide Verfahren laut ihrer Definition überein. Somit erhält man mit

$$\begin{aligned} L_i^l(y)(x) &= \nu_i y(x) - \beta_i y(x - a_i + b_{i-1}), \quad i = 2, \dots, p, \\ L_i^r(y)(x) &= \mu_i y(x) - \alpha_i y(x - b_i + a_{i+1}), \quad i = 1, \dots, p-1, \end{aligned}$$

aus VPS das KIPS. □

Zum Beweis der Umkehrung wird das folgende Lemma benötigt.

Lemma 4.4.1. *Gegeben sei die lineare Differentialgleichung*

$$L(y) = f \qquad \text{in} \quad (a,b),$$

die auf jedem Teilintervall eindeutig mit Dirichlet–Randbedingungen lösbar sei. Weiter sei $\{(a_i, b_i),\ i = 1, \dots, p\}$ *eine p–überlappende Zerlegung von* (a,b) *und seien* φ_i, ψ_i *die Funktionen aus* (4.5) *mit* $\varphi_i(a_{i+1}) \neq 0$ *und* $\psi_{i+1}(b_i) \neq 0$. *Weiter sei* $\widetilde{L}$ *ein linearer Operator, der für die Funktionen* φ_i *und* ψ_i *definiert ist. Dann gilt für* $i = 1, \dots, p-1$

(i) $$\frac{\psi_i(a_{i+1})}{\varphi_i(a_{i+1})}\widetilde{L}(\varphi_i) + \frac{1}{\psi_{i+1}(b_i)}\widetilde{L}(\psi_{i+1}) = \widetilde{L}(\psi_i)$$

(ii) $$\frac{\varphi_{i+1}(b_i)}{\psi_{i+1}(b_i)}\widetilde{L}(\psi_{i+1}) + \frac{1}{\varphi_i(a_{i+1})}\widetilde{L}(\varphi_i) = \widetilde{L}(\varphi_{i+1})$$

(iii) $$\frac{h_i(a_{i+1})}{\varphi_i(a_{i+1})}\widetilde{L}(\varphi_i) - \frac{h_{i+1}(b_i)}{\psi_{i+1}(b_i)}\widetilde{L}(\psi_{i+1}) = \widetilde{L}(h_i) - \widetilde{L}(h_{i+1})$$

im Überlappungsbereich $[a_{i+1}, b_i]$.

Beweis. Zum Beweis der Identität (i) genügt es, wegen der Linearität von $\widetilde{L}$, die Gleichungen

$$\frac{\psi_i(a_{i+1})}{\varphi_i(a_{i+1})}\varphi_i(x) + \frac{1}{\psi_{i+1}(b_i)}\psi_{i+1}(x) = \psi_i(x) \qquad \forall x \in [a_{i+1}, b_i]$$

für $i = 1, \dots, p-1$ zu verifizieren. Für die Randwerte a_{i+1} und b_i ergibt sich die Gleichheit unmittelbar aus den Definitionen der Funktionen φ_i, ψ_i und ψ_{i+1}. Da

$$L\left(\frac{\psi_i(a_{i+1})}{\varphi_i(a_{i+1})}\varphi_i + \frac{1}{\psi_{i+1}(b_i)}\psi_{i+1}\right) = L(\psi_i) \quad \text{in} \quad [a_{i+1}, b_i]$$

gilt, folgt die Behauptung aus der Voraussetzung an den Operator L. Analog lassen sich die Gleichungen (ii) und (iii) beweisen. □

Der nachfolgende Satz gibt eine Aussage über die Äquivalenz der Verfahren KIPS und VPS.

Satz 4.4.2. *Gegeben sei die lineare Differentialgleichung*

$$L(y) = f \qquad in \quad (a,b)$$

mit verschwindenden Dirichlet–Randbedingungen. Sei $\{(a_i,b_i)\,,\ i=1,\dots,p\}$ *eine p–überlappende Zerlegung von* (a,b) *und erfüllen* φ_i, ψ_i *aus* (4.5) *die Voraussetzungen* $\varphi_i(a_{i+1}) \neq 0$ *und* $\psi_{i+1}(b_i) \neq 0$ *für* $i = 1,\dots,p-1$. *Dann sind KIPS und VPS äquivalent.*

Beweis. Da an den Intervallenden a und b die Verfahren übereinstimmen und Satz 4.4.1 den ersten Teil des Satzes beweist, sollen die Werte für μ_i, ν_i, α_i und β_i angegeben werden, für welche das KIPS das VPS darstellt. Setzt man hierzu

$$\mu_i = \frac{L_i^r\left(\psi_{i+1}\right)(b_i)}{\psi_{i+1}(b_i)}\,, \quad \alpha_i = -\frac{L_i^r\left(\varphi_i\right)(b_i)}{\varphi_i(a_{i+1})}$$

und

$$\nu_{i+1} = \frac{L_{i+1}^l\left(\varphi_i\right)(a_{i+1})}{\varphi_i(a_{i+1})}\,, \quad \beta_{i+1} = -\frac{L_{i+1}^l\left(\psi_{i+1}\right)(a_{i+1})}{\psi_{i+1}(b_i)}$$

für $i = 1,\dots,p-1$, so folgt aus der Gleichung

$$\mu_i y_i^{(n+1)}(b_i) - \alpha_i y_i^{(n+1)}(a_{i+1}) = \mu_i y_{i+1}^{(n)}(b_i) - \alpha_i y_{i+1}^{(n)}(a_{i+1})$$

des KIPS mit Hilfe von (4.6) die Gleichung

$$\begin{aligned}
&\left(\mu_i - \alpha_i\psi_i(a_{i+1})\right) y_i^{(n+1)}(b_i) - \alpha_i\varphi_i(a_{i+1}) y_i^{(n+1)}(a_i) - \alpha_i h_i(a_{i+1}) \\
&= \left(\mu_i\varphi_{i+1}(b_i) - \alpha_i\right) y_{i+1}^{(n)}(a_{i+1}) + \mu_i\psi_{i+1}(b_i) y_{i+1}^{(n)}(b_{i+1}) + \mu_i h_{i+1}(b_i)\,.
\end{aligned}$$

Mit den Definitinen von μ_i und α_i ergibt sich

$$\begin{aligned}
&\left(\frac{\psi_i(a_{i+1})}{\varphi_i(a_{i+1})} L_i^r\left(\varphi_i\right)(b_i) + \frac{1}{\psi_{i+1}(b_i)} L_i^r\left(\psi_{i+1}\right)(b_i)\right) y_i^{(n+1)}(b_i) \\
&\qquad + L_i^r\left(\varphi_i\right)(b_i) y_i^{(n+1)}(a_i) + \frac{h_i(a_{i+1})}{\varphi_i(a_{i+1})} L_i^r\left(\varphi_i\right)(b_i) \\
&= \left(\frac{\varphi_{i+1}(b_i)}{\psi_{i+1}(b_i)} L_i^r\left(\psi_{i+1}\right)(b_i) + \frac{1}{\varphi_i(a_{i+1})} L_i^r\left(\varphi_i\right)(b_i)\right) y_{i+1}^{(n)}(a_{i+1}) \\
&\qquad + L_i^r\left(\psi_{i+1}\right)(b_i) y_{i+1}^{(n)}(b_{i+1}) + \frac{h_{i+1}(b_i)}{\psi_{i+1}(b_i)} L_i^r\left(\psi_{i+1}\right)(b_i)\,.
\end{aligned}$$

Nach Lemma 4.4.1 erhält man hieraus

$$\begin{aligned}
&L_i^r\left(\psi_i\right)(b_i) y_i^{(n+1)}(b_i) + L_i^r\left(\varphi_i\right)(b_i) y_i^{(n+1)}(a_i) + L_i^r\left(h_i\right)(b_i) \\
&= L_i^r\left(\varphi_{i+1}\right)(b_i) y_{i+1}^{(n)}(a_{i+1}) + L_i^r\left(\psi_{i+1}\right)(b_i) y_{i+1}^{(n)}(b_{i+1}) + L_i^r\left(h_{i+1}\right)(b_i)
\end{aligned}$$

und somit die Gleichung

$$L_i^r \left(y_i^{(n+1)} \right) (b_i) = L_i^r \left(y_{i+1}^{(n)} \right) (b_i)$$

des VPS. Entsprechend ergibt sich die Äquivalenz der zweiten Randbedingung. □

Die in diesem Kapitel erzielten Ergebnisse werden im folgenden auf das Lösen von linearen Gleichungssystemen übertragen. Auf diese Weise entstehen eigenständige Verfahren, die nur die Kommunikationsstruktur des Matrix-Vektor-Produktes benutzen werden und zusätzlich als lokale Vorkonditionierer in den entwickelten Krylov–Teilraumverfahren eingesetzt werden können.

Kapitel 5

Vorkonditionierung durch Gebietszerlegung

Konvergiert das benutzte Krylov–Teilraumverfahren zur Lösung des linearen Gleichungssystems

$$\mathbf{A}\mathbf{x} = \mathbf{b} \qquad \text{mit } \mathbf{A} \in \mathbb{R}^{N\times N} \text{ und } \mathbf{x}, \mathbf{b} \in \mathbb{R}^N \tag{5.1}$$

nur sehr langsam oder auch gar nicht, so kann eine geeignete Vorkonditionierung Abhilfe schaffen. Zu diesem Zweck wird das Gleichungssystem (5.1) in ein äquivalentes lineares Gleichungssystem

$$\mathbf{M}^{-1}\,\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\,\mathbf{b} \tag{5.2}$$

überführt, wobei die Matrix $\mathbf{M} \in \mathbb{R}^{N\times N}$ so gewählt werden sollte, daß sie $\mathbf{A}$ in einer geeigneten Weise approximiert. Das gleiche Krylov–Teilraumverfahren kann nun auf das äquivalente System $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$ angewendet werden. Algorithmus 5.1 zeigt das verallgemeinerte QMR-Verfahren zur Lösung des vorkonditionierten Systems, wobei das Residuum für das ursprüngliche System (5.1) berechnet wird, d.h. es wird iterativ $\mathbf{r}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n$ von (5.1) bestimmt.

Einfache Vorkonditionierungen können z.B. aus der Normierung der Matrix $\mathbf{A} = (a_{ij})$ mit einer Diagonalmatrix $\mathbf{M} = \mathrm{diag}(m_{11}, \ldots, m_{NN})$ bestehen, die beispielsweise durch die Elemente

$$m_{ii} \;=\; \begin{cases} a_{ii}, & \text{falls } a_{ii} \neq 0\,, \\ 1, & \text{falls } a_{ii} = 0\,, \end{cases} \qquad \text{für } i = 1, \ldots, N$$

oder

$$m_{ii} \;=\; \sqrt{\sum_{j=1}^{N} a_{ij}^2} \qquad\qquad \text{für } i = 1, \ldots, N$$

Algorithmus 5.1 Verallgemeinertes vorkonditioniertes QMR zur Lösung des Gleichungssystems $\mathbf{A}\mathbf{x} = \mathbf{b}$ mit *einem* globalen Synchronisationspunkt

Eingabe $\mathbf{A}, \mathbf{b}$ und $\mathbf{x}_0$

$\mathbf{d}_0 = \mathbf{p}_0 = \mathbf{q}_0 = \mathbf{0}\,,\ \widetilde{\mathbf{v}}_1 = \mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0\,,\ \widetilde{\mathbf{w}}_1 = \mathbf{y}_1 = \mathbf{M}^{-1}\mathbf{r}_0\,,\ \mathbf{z}_1 = \mathbf{M}^{-T}\widetilde{\mathbf{w}}_1$

$\gamma_0 = \xi_0 = 0\,,\ \tau_0 = \varrho_0 = 1\,,\ \gamma_1 = \|\mathbf{y}_1\|_2\,,\ \xi_1 = \gamma_1\,,\ \varrho_1 = \gamma_1^2\,,\ \varepsilon_1 = \left(\mathbf{A}^T\mathbf{z}_1\right)^T\mathbf{y}_1$

$\kappa_0 = -1,\ \lambda_1 = 1$

if $p = \infty$ **then** $q = 1$ **else** $q = \dfrac{p}{p-1}$ **endif**

for $n = 1, 2, 3, \ldots$ **do**

$\mu_n = \dfrac{\gamma_{n-1}\xi_{n-1}\varrho_n}{\gamma_n\tau_{n-1}\varrho_{n-1}}\,, \qquad \tau_n = \dfrac{\varepsilon_n}{\varrho_n} - \gamma_n\mu_n$

$\mathbf{p}_n = \dfrac{1}{\gamma_n}\mathbf{y}_n - \mu_n\mathbf{p}_{n-1}\,, \qquad \mathbf{q}_n = \dfrac{1}{\xi_n}\mathbf{A}^T\mathbf{z}_n - \dfrac{\gamma_n\mu_n}{\xi_n}\mathbf{q}_{n-1}$

$\widetilde{\mathbf{v}}_{n+1} = \mathbf{A}\mathbf{p}_n - \dfrac{\tau_n}{\gamma_n}\widetilde{\mathbf{v}}_n\,, \qquad \widetilde{\mathbf{w}}_{n+1} = \mathbf{q}_n - \dfrac{\tau_n}{\xi_n}\widetilde{\mathbf{w}}_n$

$\mathbf{y}_{n+1} = \mathbf{M}^{-1}\,\widetilde{\mathbf{v}}_{n+1}\,, \qquad \mathbf{z}_{n+1} = \mathbf{M}^{-T}\,\widetilde{\mathbf{w}}_{n+1}$

if $(\|\mathbf{r}_{n-1}\|_2 <$ Toleranz$)$ **then** STOP

$\gamma_{n+1} = \|\mathbf{y}_{n+1}\|_2$

$\xi_{n+1} = \|\widetilde{\mathbf{w}}_{n+1}\|_2$

$\varrho_{n+1} = \widetilde{\mathbf{w}}_{n+1}^T\mathbf{y}_{n+1}$

$\varepsilon_{n+1} = \left(\mathbf{A}^T\mathbf{z}_{n+1}\right)^T\mathbf{y}_{n+1}$

$\vartheta_n = \left|\dfrac{\gamma_{n+1}}{\tau_n}\right|^q$

$\sigma_n = \dfrac{1-\lambda_n}{\lambda_n + \vartheta_n}$

$\kappa_n = -\dfrac{\gamma_n}{\tau_n}\,\dfrac{\kappa_{n-1}}{\lambda_n + \vartheta_n}$

$\lambda_{n+1} = \dfrac{\lambda_n}{\lambda_n + \vartheta_n}$

$\mathbf{d}_n = \sigma_n\mathbf{d}_{n-1} + \kappa_n\mathbf{p}_n$

$\mathbf{x}_n = \mathbf{x}_{n-1} + \mathbf{d}_n$

$\mathbf{r}_n = \dfrac{\vartheta_n}{\lambda_n + \vartheta_n}\mathbf{r}_{n-1} - \kappa_n\widetilde{\mathbf{v}}_{n+1}$

enddo

gegeben ist. Im zweiten Fall werden die Zeilenvektoren der Matrix $\mathbf{A}$ in der euklidischen Norm zu Eins normiert.

Es soll nun gezeigt werden, wie man mit Hilfe des parallelen Schwarz–Verfahrens ein vorkonditioniertes Gleichungssystem (5.2) erhalten kann. Dazu sei die Randwertaufgabe (4.1) in $\Omega = \Omega_1 \cup \Omega_2$ mit überlappenden Teilgebieten Ω_1 und Ω_2 und homogener Dirichlet–Randbedingung in der schwachen Formulierung gegeben, d.h. es wird eine Funktion $u \in \mathrm{H}_0^1(\Omega)$ gesucht mit

$$a(u,v) = f(v) \qquad \forall\ v \in \mathrm{H}_0^1(\Omega)\,, \tag{5.3}$$

wobei $a(\cdot,\cdot)$ die zugeordnete Bilinearform und $f(\cdot)$ das entsprechende Funktional bezeichnen. Ist nun $u^{(0)}$ eine auf Ω gegebene Anfangsnäherung mit $u^{(0)} \in \mathrm{H}_0^1(\Omega)$, so werden für $n \in \mathbb{N}_0$ die zu lösenden Randwertprobleme (4.2) und (4.3) des parallelen Schwarz–Verfahrens durch die folgenden schwachen Formulierungen charakterisiert. Suche $u_1^{(n+1)} - u^{(n)} \in \mathrm{H}_0^1(\Omega_1) \subset \mathrm{H}_0^1(\Omega)$ mit

$$a(u_1^{(n+1)} - u^{(n)}, v) \;=\; f(v) - a(u^{(n)}, v) \qquad \forall\ v \in \mathrm{H}_0^1(\Omega_1) \tag{5.4}$$

und gleichzeitig eine Funktion $u_2^{(n+1)} - u^{(n)} \in \mathrm{H}_0^1(\Omega_2) \subset \mathrm{H}_0^1(\Omega)$ mit

$$a(u_2^{(n+1)} - u^{(n)}, v) \;=\; f(v) - a(u^{(n)}, v) \qquad \forall\ v \in \mathrm{H}_0^1(\Omega_2)\,. \tag{5.5}$$

Da $f(v) - a(u^{(n)}, v) = a(u - u^{(n)}, v)$ für alle $v \in \mathrm{H}_0^1(\Omega_1)$, folgt aus (5.4) die Identität

$$u_1^{(n+1)} - u^{(n)} = P_1(u - u^{(n)})\,, \tag{5.6}$$

wobei P_1 die orthogonale Projektion von $\mathrm{H}_0^1(\Omega)$ auf $\mathrm{H}_0^1(\Omega_1)$ bezeichnet. Somit ist $u_1^{(n+1)} - u^{(n)}$ die beste Approximation von $u - u^{(n)}$ im Unterraum $\mathrm{H}_0^1(\Omega_1)$bezüglich der Energienorm $a(\cdot,\cdot)^{\frac{1}{2}}$. Genauso ergibt sich aus (5.5)

$$u_2^{(n+1)} - u^{(n)} = P_2(u - u^{(n)})\,, \tag{5.7}$$

wobei jetzt $P_2 : \mathrm{H}_0^1(\Omega) \to \mathrm{H}_0^1(\Omega_2)$ die orthogonale Projektion auf den Unterraum $\mathrm{H}_0^1(\Omega_2) \subset \mathrm{H}_0^1(\Omega)$ bezeichnet. Wird nun die neue Iterierte $u^{(n+1)}$ auf Ω durch

$$u^{(n+1)} \;=\; u^{(n)} \;+\; P_1(u - u^{(n)}) \;+\; P_2(u - u^{(n)}) \tag{5.8}$$

definiert, so erhält man das sogenannte additive Schwarz–Verfahren. Sehr wichtig ist die Tatsache, daß die Projektionen der Lösung u der Randwertaufgabe (5.3) auch ohne deren Kenntnis berechnet werden können.

Gemeinsamkeiten mit dem parallelen Verfahren ergeben sich nun zum einen in der Möglichkeit, die Projektionen $P_1(u-u^{(n)})$ und $P_2(u-u^{(n)})$ gleichzeitig zu berechnen, und zum anderen in der Übereinstimmung der Iterierten in $\Omega \setminus (\Omega_1 \cap \Omega_2)$, wie die folgende Bemerkung genauer erläutert.

Bemerkung 5.0.1. *Die Iterierten $u^{(n+1)}$, $n \in \mathbb{N}_0$, des additiven Verfahrens stimmen mit den Iterierten $u_1^{(n+1)}$ und $u_2^{(n+1)}$ des parallelen Schwarz–Verfahrens in $\Omega \setminus (\Omega_1 \cap \Omega_2)$ überein, d.h.*

$$u^{(n+1)} \;=\; u_1^{(n+1)} \qquad \text{in } \Omega_1 \setminus (\Omega_1 \cap \Omega_2)$$

und

$$u^{(n+1)} \;=\; u_2^{(n+1)} \qquad \text{in } \Omega_2 \setminus (\Omega_1 \cap \Omega_2)\,.$$

Beweis. Aus der Iterationsvorschrift (5.8) folgt mit (5.6) und (5.7) die Gleichung

$$u^{(n+1)} \;=\; u_1^{(n)} \;+\; u_2^{(n)} \;-\; u^{(n)}\,.$$

Da $u_2^{(n)} - u^{(n)} \in \mathrm{H}_0^1(\Omega_2) \subset \mathrm{H}_0^1(\Omega)$, ist $u_2^{(n)} - u^{(n)} = 0$ in $\Omega \setminus \Omega_2$. Hieraus ergibt sich die erste Gleichung. Entsprechend erhält man die zweite. □

Der Unterschied der beiden Verfahren liegt nun darin, daß jede Iterierte $u^{(n)}$ des additiven Schwarz–Verfahrens eine in ganz Ω definierte Funktion ist, während die beiden Iterierten $u_1^{(n)}$ und $u_2^{(n)}$ des parallelen Verfahrens jeweils auf den Teilgebieten Ω_1 bzw. Ω_2 definiert sind und i.allg. im Überlappungsbereich $\Omega_1 \cap \Omega_2$ nicht übereinstimmen, vergleiche Abbildung 5.1.

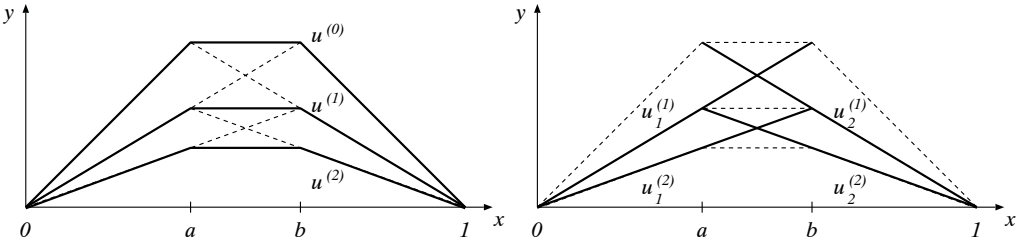


Abbildung 5.1: Iterationsfolge des additiven (links) und parallelen (rechts) Schwarz–Verfahrens für die Differentialgleichung $u'' = 0$ mit homogener Dirichlet–Randbedingung in den beiden überlappenden Teilgebieten $\Omega_1 = (0, b)$ und $\Omega_2 = (a, 1)$.

Deutet man die schwache Lösung u von (5.3) der Differentialgleichung (4.1) als Lösung der Operatorgleichung

$$A\,u \;=\; f\,, \tag{5.9}$$

so entspricht dem additiven Schwarz–Verfahren das Richardson–Verfahren zur Lösung der Gleichung

$$(P_1 \;+\; P_2)\,u \;=\; g \tag{5.10}$$

mit der bestimmbaren rechten Seite $g = P_1 u + P_2 u$, vergleiche auch das folgende Beispiel. Diese kann nun als eine vorkonditionierte Gleichung von (5.9) interpretiert werden. Im folgenden Abschnitt wird dazu ein Beispiel behandelt, bei dem auch die Projektionen explizit durch Matrizen dargestellt werden.

5.1 Das additive Schwarz–Verfahren

In diesem Abschnitt werden die orthogonalen Projektionen des additiven Schwarz–Verfahrens in einem diskreten Beispiel durch geeignete Matrizen angegeben. Außerdem wird das Verfahren für p sich überlappende Teilgebiete beschrieben.

Bezeichnet nun wie früher $\{I_i, i = 1, \ldots, p\}$ eine disjunkte Zerlegung der Indexmenge $I = \{1, \ldots, N\}$ und $\{\phi_i, i \in I\}$ die übliche Basis des Raumes V^h der stückweise linearen finiten Elemente, so läßt sich dieser Raum in die direkte Summe $V^h = V_1^h \oplus \ldots \oplus V_p^h$ mit den Teilräumen

$$V_i^h \;=\; \text{span}\{\phi_j,\ j \in I_i\}\,, \qquad i = 1, \ldots, p\,,$$

zerlegen. Bezeichnet $\text{supp}(\phi_i) = \overline{\{x \in \Omega :\ \phi_i(x) \neq 0\}}$ den Träger der Funktionen $\phi_i : \Omega \to \mathbb{R}$, so erhält man mit

$$\overline{\Omega_i} \;=\; \bigcup_{j \in I_i} \text{supp}(\phi_j)\,, \qquad i = 1, \ldots, p\,,$$

eine überlappende Zerlegung $\{\Omega_i, i = 1, \ldots, p\}$ von Ω mit

$$\overline{\Omega} \;=\; \bigcup_{i \in I} \text{supp}(\phi_i)\,,$$

vergleiche Abbildung 5.2.

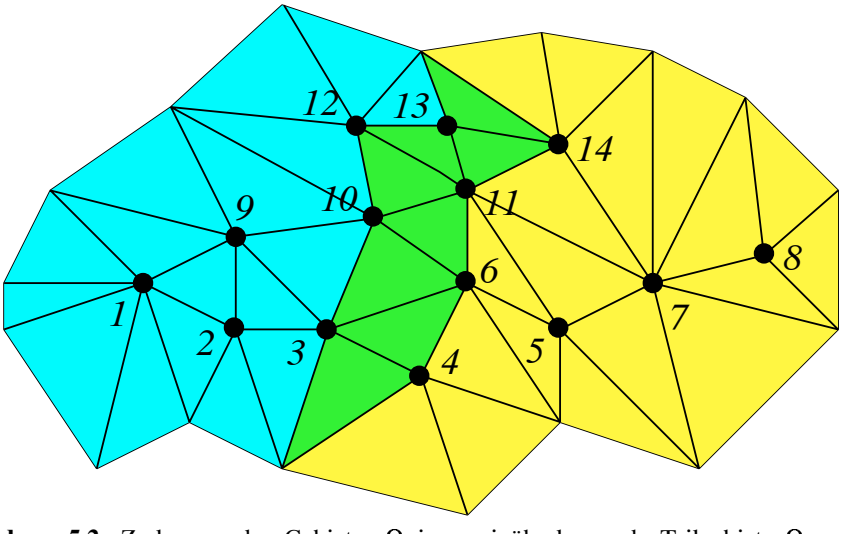


Abbildung 5.2: Zerlegung des Gebietes Ω in zwei überlappende Teilgebiete Ω_1 und Ω_2 durch Zerlegung der Indexmenge $I = \{1, \ldots, 14\}$ in $I_1 = \{1, 2, 3, 9, 10, 12, 13\}$ und $I_2 = \{4, 5, 6, 7, 8, 11, 14\}$ der zugehörigen Basisfunktionen $\{\phi_i, i \in I\}$.

Das zu (5.3) gehörige diskrete Problem bestimmt die orthogonale Projektion u^h von $u \in \text{H}_0^1(\Omega)$ in dem Raum V^h, also die Linearkombination $u^h = \sum_{i=1}^N u_i\,\phi_i \in V^h$ mit

$$a(u^h, v) = f(v) \qquad \forall\ v \in V^h\,. \tag{5.11}$$

Die Funktion u^h ist deshalb die beste Approximation von u in V^h bezüglich der Energienorm. Ist nun $\mathbf{u}$ der Vektor $\mathbf{u} = (u_1, \ldots, u_N)^T$ und $\mathbf{f} = (f(\phi_1), \ldots, f(\phi_N))^T$ und bezeichnet $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N \times N}$ die Steifigkeitsmatrix mit den Elementen

$$a_{ij} = a(\phi_j, \phi_i) \qquad \text{für } i,j = 1, \ldots, N\,,$$

so sind die Koeffizienten $u_1, \ldots, u_N$ der Lösung $u^h = \sum_{i=1}^N u_i\,\phi_i$ von (5.11) als Lösung des linearen Gleichungssystems

$$\mathbf{A}\,\mathbf{u} \;=\; \mathbf{f} \tag{5.12}$$

gegeben. Die Projektionen $P_i : V^h \to V_i^h$, $w \mapsto P_i w$, für $i = 1, \ldots, p$ sind durch die Gleichungen

$$a(P_i\,w, v) \;=\; a(w, v) \qquad \forall\ v \in V_i^h \tag{5.13}$$

eindeutig bestimmt. Gilt $w = \sum_{i=1}^N w_i\,\phi_i \in V^h$ und $z = P_i w = \sum_{j \in I_i} z_j \phi_j \in V_i^h$, so ist die Matrixdarstellung $\mathbf{P}_i$ von P_i durch die Abbildung $\mathbf{P}_i : \mathbf{w} \mapsto \mathbf{R}_{I_i}^T \mathbf{z}_{I_i}$ definiert, wobei $\mathbf{w} = (w_1, \ldots, w_N)^T$ und $\mathbf{z}_{I_i} = (z_{j_1}, \ldots, z_{j_m})^T$ mit $I_i = \{j_1, \ldots, j_m\}$. Die Matrizen $\mathbf{P}_i$ lassen sich nun mit Hilfe der zu (5.13) äquivalenten Gleichung

$$\begin{aligned}
& a(P_i\,w, \phi_j) \;=\; a(w, \phi_j) && \forall\ j \in I_i \\
\iff \qquad & \sum_{k \in I_i} z_k\,a(\phi_k, \phi_j) \;=\; \sum_{k=1}^N w_k\,a(\phi_k, \phi_j) && \forall\ j \in I_i \\
\iff \qquad & \sum_{k \in I_i} a_{jk}\,z_k \;=\; \sum_{k=1}^N a_{jk}\,w_k && \forall\ j \in I_i \\
\iff \qquad & \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T\mathbf{z}_{I_i} \;=\; \mathbf{R}_{I_i}\mathbf{A}\mathbf{w}
\end{aligned}$$

bestimmen. Sind nämlich die Matrizen $\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T$ invertierbar, so ergibt sich hieraus

$$\mathbf{z}_{I_i} \;=\; (\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T)^{-1}\mathbf{R}_{I_i}\mathbf{A}\mathbf{w}\,,$$

also ist die Matrixdarstellung von P_i durch

$$\mathbf{P}_i \;=\; \mathbf{R}_{I_i}^T(\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T)^{-1}\mathbf{R}_{I_i}\mathbf{A}$$

gegeben. Mit der Bezeichnung $\mathbf{A}_{ii} = \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T$ für $i = 1, \ldots, p$ erhält man so aus (5.10) das zu $\mathbf{A}\mathbf{u} = \mathbf{f}$ vorkonditionierte Gleichungssystem

$$\Big(\sum_{i=1}^p \mathbf{R}_{I_i}^T\mathbf{A}_{ii}^{-1}\mathbf{R}_{I_i}\Big)\,\mathbf{A}\mathbf{u} \;=\; \Big(\sum_{i=1}^p \mathbf{R}_{I_i}^T\mathbf{A}_{ii}^{-1}\mathbf{R}_{I_i}\Big)\,\mathbf{f}\,.$$

Die Matrix $\mathbf{M}_{\text{ad}}$ mit $\mathbf{M}_{\text{ad}}^{-1} = \sum_{i=1}^{p} \mathbf{R}_{I_i}^T \mathbf{A}_{ii}^{-1} \mathbf{R}_{I_i}$ wird als additiver Schwarz–Vorkonditionierer bezeichnet. Das additive Verfahren (5.8) entspricht so dem Block–Jacobi–Verfahren zur Lösung des linearen Gleichungssystems

$$\begin{pmatrix} \mathbf{A}_{11} & \dots & \mathbf{A}_{1p} \\ \vdots & \dots & \vdots \\ \mathbf{A}_{p1} & \dots & \mathbf{A}_{pp} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{I_1} \\ \vdots \\ \mathbf{u}_{I_p} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_{I_1} \\ \vdots \\ \mathbf{f}_{I_p} \end{pmatrix}$$

mit $\mathbf{A}_{ij} = \mathbf{R}_{I_i} \mathbf{A} \mathbf{R}_{I_j}^T$ für $i, j = 1, \dots, p$, dessen Blöcke den additiven Schwarz–Vorkonditionierer

$$\mathbf{M}_{\text{ad}} = \text{diag}(\mathbf{A}_{11}, \dots, \mathbf{A}_{pp}) \tag{5.14}$$

liefern.

Im folgenden soll demonstriert werden, wie das beschleunigte parallele Verfahren aus dem vierten Kapitel zur Konstruktion von weiteren Vorkonditionierern eingesetzt werden kann. Die in diesem Kapitel angegebenen Beschleunigungstechniken benötigen jedoch die Mehrdeutigkeit der Funktionswerte im Überlappungsbereich, die aber – wie bereits erwähnt – beim additiven Schwarz–Verfahren verloren geht. Folglich kann diese Methode nicht auf das lineare Gleichungssystem $\mathbf{A}\mathbf{u} = \mathbf{f}$ angewendet werden, so daß das erweiterte System $\mathbf{A}^{\mathbf{e}}\mathbf{u}^{\mathbf{e}} = \mathbf{f}^{\mathbf{e}}$ herangezogen werden muß, in dem die zusätzlichen Empfangsvektoren $(\widetilde{\mathbf{u}}_{\partial I_1}, \dots, \widetilde{\mathbf{u}}_{\partial I_1,})^T$ die geforderte Mehrdeutigkeit ermöglichen.

Zunächst soll aber der zu (5.14) analoge Vorkonditionierer für das erweiterte System erarbeitet werden. Dazu wird gezeigt, daß die jeweiligen Block–Jacobi–Verfahren für die beiden Gleichungssysteme identisch sind. Dies ist Gegenstand des folgenden Abschnittes.

5.2 Das erweiterte Gleichungssystem

Zu einer p-disjunkten Zerlegung $\{I_i, i = 1, \dots, p\}$ der Indexmenge $I = \{1, 2, \dots, N\}$ sei das lineare Gleichungssystem $\mathbf{A}\mathbf{x} = \mathbf{b}$ durch

$$\begin{pmatrix} \mathbf{A}_{11} & \dots & \mathbf{A}_{1p} \\ \vdots & \dots & \vdots \\ \mathbf{A}_{p1} & \dots & \mathbf{A}_{pp} \end{pmatrix} \begin{pmatrix} \mathbf{x}_{I_1} \\ \vdots \\ \mathbf{x}_{I_p} \end{pmatrix} = \begin{pmatrix} \mathbf{b}_{I_1} \\ \vdots \\ \mathbf{b}_{I_p} \end{pmatrix} \tag{5.15}$$

gegeben. Gilt wie früher

$$\begin{aligned} \mathbf{A}_{ii} &= \mathbf{R}_{I_i} \mathbf{A} \mathbf{R}_{I_i}^T, \quad i = 1, \dots, p, \\ \mathbf{A}_{i} &= \mathbf{R}_{I_i} \mathbf{A} \mathbf{R}_{\partial I_i}^T, \quad i = 1, \dots, p, \end{aligned}$$

und

$$\mathbf{R}_{ij} = \mathbf{R}_{\partial I_i}\mathbf{R}_{I_j}^T \qquad \text{für } i,j = 1,\ldots,p\,,$$

so ist

$$\begin{pmatrix}
\mathbf{A}_{11} & \mathbf{A}_1 & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{I} & -\mathbf{R}_{12} & \mathbf{0} & \ldots & -\mathbf{R}_{1p} & \mathbf{0} \\
 & & & \ddots & & & \\
 & & & \ddots & & & \\
\mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} & \mathbf{A}_{pp} & \mathbf{A}_p \\
-\mathbf{R}_{p1} & \mathbf{0} & \ldots & -\mathbf{R}_{pp-1} & \mathbf{0} & \mathbf{0} & \mathbf{I}
\end{pmatrix}
\begin{pmatrix}
\mathbf{x}_{I_1} \\ \widetilde{\mathbf{x}}_{\partial I_1} \\ \mathbf{x}_{I_2} \\ \widetilde{\mathbf{x}}_{\partial I_2} \\ \vdots \\ \mathbf{x}_{I_{p-1}} \\ \widetilde{\mathbf{x}}_{\partial I_{p-1}} \\ \mathbf{x}_{I_p} \\ \widetilde{\mathbf{x}}_{\partial I_p}
\end{pmatrix} =
\begin{pmatrix}
\mathbf{b}_{I_1} \\ \mathbf{0} \\ \mathbf{b}_{I_2} \\ \mathbf{0} \\ \vdots \\ \mathbf{b}_{I_{p-1}} \\ \mathbf{0} \\ \mathbf{b}_{I_p} \\ \mathbf{0}
\end{pmatrix} \tag{5.16}$$

das zu (5.15) gehörige erweiterte lineare Gleichungssystem $\mathbf{A^e x^e} = \mathbf{b^e}$ aus dem zweiten Kapitel. Sind nun

$$\mathbf{D} = \operatorname{diag}(\mathbf{A}_{11},\ \ldots\ ,\mathbf{A}_{pp})$$

und

$$\mathbf{D^e} = \operatorname{diag}\left(\begin{pmatrix}\mathbf{A}_{11} & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{I}\end{pmatrix},\ \ldots\ ,\begin{pmatrix}\mathbf{A}_{pp} & \mathbf{A}_p \\ \mathbf{0} & \mathbf{I}\end{pmatrix}\right)$$

die jeweiligen Blockdiagonalen von $\mathbf{A}$ bzw. $\mathbf{A^e}$, so lautet das zugehörige Block–Jacobi–Verfahren zur Lösung von (5.15)

$$\begin{aligned}
\mathbf{D}\mathbf{x}^{(n+1)} &= \mathbf{b} + \Big(\mathbf{D} - \mathbf{A}\Big)\mathbf{x}^{(n)} \qquad (5.17)\\
\Longleftrightarrow \qquad \mathbf{x}^{(n+1)} &= \mathbf{x}^{(n)} + \mathbf{D}^{-1}\Big(\mathbf{b} - \mathbf{A}\mathbf{x}^{(n)}\Big)
\end{aligned}$$

und das entsprechende Block–Jacobi–Verfahren zur Lösung der Gleichung (5.16)

$$\begin{aligned}
\mathbf{D^e}\mathbf{x^e}^{(n+1)} &= \mathbf{b^e} + \Big(\mathbf{D^e} - \mathbf{A^e}\Big)\mathbf{x^e}^{(n)} \qquad (5.18)\\
\Longleftrightarrow \qquad \mathbf{x^e}^{(n+1)} &= \mathbf{x^e}^{(n)} + \mathbf{D^e}^{-1}\Big(\mathbf{b^e} - \mathbf{A^e}\mathbf{x^e}^{(n)}\Big)\,.
\end{aligned}$$

Für die einzelnen Blöcke $i = 1,\ldots,p$ ergibt sich dann

$$\mathbf{x}_{I_i}^{(n+1)} = \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\Big(\mathbf{b}_{I_i} - \sum_{j=1}^{p}\mathbf{A}_{ij}\,\mathbf{x}_{I_j}^{(n)}\Big) \tag{5.19}$$

bzw.

$$\begin{pmatrix}\mathbf{x}_{I_i} \\ \widetilde{\mathbf{x}}_{\partial I_i}\end{pmatrix}^{(n+1)} = \begin{pmatrix}\mathbf{x}_{I_i} \\ \widetilde{\mathbf{x}}_{\partial I_i}\end{pmatrix}^{(n)} + \begin{pmatrix}\mathbf{A}_{ii} & \mathbf{A}_i \\ \mathbf{0} & \mathbf{I}\end{pmatrix}^{-1}\left(\begin{pmatrix}\mathbf{b}_{I_i} \\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbf{A}_{ii}\mathbf{x}_{I_i}^{(n)} + \mathbf{A}_i\widetilde{\mathbf{x}}_{\partial I_i}^{(n)} \\ \widetilde{\mathbf{x}}_{\partial I_i}^{(n)} - \sum_{j=1}^{p}\mathbf{R}_{ij}\mathbf{x}_{I_j}^{(n)}\end{pmatrix}\right), \tag{5.20}$$

wenn man beachtet, daß $\mathbf{R}_{ii} = \mathbf{0}$ für $i = 1, \ldots, p$. Es soll nun gezeigt werden, daß die beiden Verfahren (5.17) und (5.18) äquivalent sind. Dafür ist zunächst die folgende Bemerkung wichtig.

Bemerkung 5.2.1. *Das Block–Jacobi–Verfahren zur Lösung des linearen Gleichungssystems $\mathbf{A}\mathbf{x} = \mathbf{b}$ ist genau dann durchführbar, wenn das Block–Jacobi–Verfahren zur Lösung des erweiterten Gleichungssystems $\mathbf{A}^{\mathbf{e}}\mathbf{x}^{\mathbf{e}} = \mathbf{b}^{\mathbf{e}}$ durchführbar ist. Darüber hinaus gilt für die Blöcke des Verfahrens* (5.18) *bzw.* (5.20)

$$\begin{pmatrix} \mathbf{A}_{ii} & \mathbf{A}_i \\ \mathbf{0} & \mathbf{I} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{A}_{ii}^{-1} & \widetilde{\mathbf{A}}_i \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \tag{5.21}$$

mit $\widetilde{\mathbf{A}}_i = -\mathbf{A}_{ii}^{-1}\mathbf{A}_i$ für $i = 1, \ldots, p$.

Beweis. Die beiden Blockdiagonalmatrizen $\mathbf{D}$ und $\mathbf{D}^{\mathbf{e}}$ sind bei beiden Verfahren genau dann invertierbar, wenn die Matrizen $\mathbf{A}_{ii}^{-1}$ für $i = 1, \ldots, p$ existieren. Aus

$$\begin{pmatrix} \mathbf{A}_{ii}^{-1} & \widetilde{\mathbf{A}}_i \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{A}_{ii} & \mathbf{A}_i \\ \mathbf{0} & \mathbf{I} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$$

folgt schließlich der zweite Teil der Behauptung. □

Mit dieser Bemerkung kann nun der folgende Satz über die Äquivalenz der beiden Verfahren bewiesen werden.

Satz 5.2.1. *Die beiden Block–Jacobi–Verfahren* (5.17) *und* (5.18) *erzeugen bei gleichen Startvektoren dieselbe Iterationsfolge, d.h.:*
Sei $\mathbf{x}^{\mathbf{e}} = \left(\mathbf{x}_{I_1}, \widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \mathbf{x}_{I_p}, \widetilde{\mathbf{x}}_{\partial I_p}\right)^T$ der erweiterte Vektor von $\mathbf{x} = \left(\mathbf{x}_{I_1}, \ldots, \mathbf{x}_{I_p}\right)^T$ und sei $\mathbf{y} = \left(\mathbf{y}_{I_1}, \ldots, \mathbf{y}_{I_p}\right)^T$. Wird nun die Folge $\mathbf{y}^{(n)}$ mit dem Verfahren (5.17) *zum Startvektor $\mathbf{y}^{(0)} = \mathbf{y}$ und die Folge $\mathbf{x}^{\mathbf{e}(n)}$ mit* (5.18) *zum Startvektor $\mathbf{x}^{\mathbf{e}(0)} = \mathbf{x}$ erzeugt, so gilt unabhängig von der Wahl des Vektors $\left(\widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \widetilde{\mathbf{x}}_{\partial I_p}\right)^T$:*

$$\mathbf{y}^{(n)} \;=\; \mathbf{x}^{(n)} \;\; \forall n \in \mathbb{N}_0 \qquad \Longleftrightarrow \qquad \mathbf{y} \;=\; \mathbf{x}\,.$$

Beweis. Mit (5.21) erhält man aus der Blockdarstellung (5.20) für $i = 1, \ldots, p$ die Gleichungen

$$\begin{aligned} \mathbf{x}_{I_i}^{(n+1)} &= \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i} - \mathbf{A}_{ii}^{-1}\left(\mathbf{A}_{ii}\mathbf{x}_{I_i}^{(n)} + \mathbf{A}_i\widetilde{\mathbf{x}}_{\partial I_i}^{(n)}\right) - \widetilde{\mathbf{A}}_i\left(\widetilde{\mathbf{x}}_{\partial I_i}^{(n)} - \sum_{j=1}^{n}\mathbf{R}_{ij}\mathbf{x}_{I_j}^{(n)}\right) \\ &= \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i} - \mathbf{A}_{ii}^{-1}\mathbf{A}_{ii}\mathbf{x}_{I_i}^{(n)} - \mathbf{A}_{ii}^{-1}\mathbf{A}_i\widetilde{\mathbf{x}}_{\partial I_i}^{(n)} - \widetilde{\mathbf{A}}_i\widetilde{\mathbf{x}}_{\partial I_i}^{(n)} + \widetilde{\mathbf{A}}_i\sum_{j=1}^{n}\mathbf{R}_{ij}\mathbf{x}_{I_j}^{(n)}\,. \end{aligned}$$

Wegen $\widetilde{\mathbf{A}}_i = -\mathbf{A}_{ii}^{-1}\mathbf{A}_i$, vergleiche Bemerkung 5.2.1, gilt weiter

$$\mathbf{x}_{I_i}^{(n+1)} = \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i} - \mathbf{A}_{ii}^{-1}\left(\mathbf{A}_{ii}\mathbf{x}_{I_i}^{(n)} + \mathbf{A}_i\sum_{j=1}^{n}\mathbf{R}_{ij}\mathbf{x}_{I_j}^{(n)}\right).$$

Mit der Definition von $\mathbf{A}_{ii}$, $\mathbf{A}_i$ und $\mathbf{R}_{ij}$ schließt man hieraus

$$\begin{aligned}\mathbf{x}_{I_i}^{(n+1)} &= \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i} - \mathbf{A}_{ii}^{-1}\left(\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T\mathbf{x}_{I_i}^{(n)} + \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{\partial I_i}^T\sum_{j=1}^{n}\mathbf{R}_{\partial I_i}\mathbf{R}_{I_j}^T\mathbf{x}_{I_j}^{(n)}\right)\\ &= \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i} - \mathbf{A}_{ii}^{-1}\left(\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T\mathbf{x}_{I_i}^{(n)} + \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{\partial I_i}^T\mathbf{x}_{\partial I_i}^{(n)}\right).\end{aligned}$$

Da die lokalen Komponenten $\mathbf{R}_{I_i}\mathbf{A}\mathbf{x}^{(n)} = \sum_{j=1}^{p}\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_j}^T\mathbf{x}_{I_j}^{(n)}$ nur mit Kenntnis von $\mathbf{x}_{I_i}^{(n)}$ und $\mathbf{x}_{\partial I_i}^{(n)}$ berechnet werden können, vergleiche (2.4), ist

$$\sum_{j=1}^{p}\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_j}^T\mathbf{x}_{I_j}^{(n)} = \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_i}^T\mathbf{x}_{I_i}^{(n)} + \mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{\partial I_i}^T\mathbf{x}_{\partial I_i}^{(n)},$$

so daß

$$\begin{aligned}\mathbf{x}_{I_i}^{(n+1)} &= \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i} - \mathbf{A}_{ii}^{-1}\Big(\sum_{j=1}^{p}\mathbf{R}_{I_i}\mathbf{A}\mathbf{R}_{I_j}^T\mathbf{x}_{I_j}^{(n)}\Big)\\ &= \mathbf{x}_{I_i}^{(n)} + \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i} - \mathbf{A}_{ii}^{-1}\Big(\sum_{j=1}^{p}\mathbf{A}_{ij}\mathbf{x}_{I_j}^{(n)}\Big).\end{aligned}$$

Diese Iterationsvorschrift stimmt aber mit der Blockdarstellung (5.19) überein. □

Der dem additiven Schwarz–Vorkonditionierer (5.14) entsprechende Vorkonditionierer für das erweiterte Gleichungssystem ist somit nach Satz 5.2.1 durch

$$\mathbf{M}_{\text{ad}}^{\mathbf{e}} = \operatorname{diag}\left(\begin{pmatrix}\mathbf{A}_{11} & \mathbf{A}_1\\ \mathbf{0} & \mathbf{I}\end{pmatrix}, \ldots, \begin{pmatrix}\mathbf{A}_{pp} & \mathbf{A}_p\\ \mathbf{0} & \mathbf{I}\end{pmatrix}\right)$$

gegeben. Zum obigen Ergebnis sei noch bemerkt, daß im Gegensatz zum Block–Jacobi–Verfahren (5.17) die rechte Seite der Gleichung (5.18) nur aus Elementen der Kommunikationsmatrix (2.2) besteht und folglich nur mit Nullen oder Einsen besetzt ist. Außerdem ergibt der untere Block der Gleichung (5.20) die Identität

$$\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} = \widetilde{\mathbf{x}}_{\partial I_i}^{(n)} - \widetilde{\mathbf{x}}_{\partial I_i}^{(n)} + \sum_{j=1}^{n}\mathbf{R}_{ij}\,\mathbf{x}_{I_j}^{(n)} = \mathbf{x}_{\partial I_i}^{(n)}$$

und beschreibt auch hier den durchzuführenden Datenaustausch. Also kann das Block–Jacobi–Verfahren zur Lösung des erweiterten Systems $\mathbf{A}^{\mathbf{e}}\mathbf{x}^{\mathbf{e}} = \mathbf{b}^{\mathbf{e}}$ in eine dem parallelen Schwarz–Verfahren (Verfahren 4.1) ähnliche Form gebracht werden, die als algebraisches paralleles Schwarz–Verfahren bezeichnet wird, siehe Verfahren 5.1.

Da in diesem Verfahren auch die Kommunikationsvariablen benutzt werden, kann dieses nun mit den Techniken, die im vierten Kapitel bereitgestellt wurden, beschleunigt werden. Dieses Vorgehen wird im folgenden Abschnitt beschrieben. Anschließend wird das beschleunigte Verfahren als Vorkonditionierer für das erweiterte Gleichungssystem interpretiert.

Verfahren 5.1 Algebraisches paralleles Schwarz–Verfahren

Eingabe: $\mathbf{A}$, $\mathbf{b}$ und $\mathbf{x}^{(0)}$

for $n = 0, 1, 2, \ldots$ **do**

Löse für $i = 1, \ldots, p$ parallel

$$\mathbf{A}_{ii}\, \mathbf{x}_{I_i}^{(n+1)} \;=\; -\mathbf{A}_i\, \widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} \;+\; \mathbf{b}_{I_i}\,,$$

wobei

$$\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} \;=\; \mathbf{x}_{\partial I_i}^{(n)}$$

enddo

5.3 Das algebraische parallele Schwarz–Verfahren

Das Verfahren 5.1 soll nun dazu verwendet werden, das lineare Gleichungssystem $\mathbf{A}\mathbf{x} = \mathbf{b}$ bzw. das erweiterte System $\mathbf{A}^{\mathbf{e}}\mathbf{x}^{\mathbf{e}} = \mathbf{b}^{\mathbf{e}}$, deren Lösungen nach Satz 2.4.1 übereinstimmen, zu lösen. Zu einer p-disjunkten Zerlegung $\{I_i,\ i = 1, \ldots, p\}$ der Indexmenge $I = \{1, 2, \ldots, N\}$ sollen nun wie früher für $i = 1, \ldots, p$ die Komponenten $\mathbf{x}_{I_i}$ des Vektors $\mathbf{x} = (\mathbf{x}_{I_1}, \ldots, \mathbf{x}_{I_p})^T$ mit den zugehörigen Zeilen der Matrix $\mathbf{A}$ im lokalen Speicher von Prozessor P_i vorhanden sein. Das Kommunikationsschema zur Durchführung des Verfahrens 5.1 entspricht so der Kommunikationsstruktur zur Berechnung des Matrix-Vektor-Produktes $\mathbf{A}\mathbf{x}$. Basierend auf dieser Struktur soll nun das Verfahren beschleunigt werden, ohne dabei zusätzliche Kommunikation durchführen zu müssen. Ähnlich wie im vierten Kapitel soll hierzu das Verfahren 5.1 nur bezüglich der auszutauschenden Komponenten, d.h. der Empfangsvektoren $\widetilde{\mathbf{x}}_{\partial} = (\widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \widetilde{\mathbf{x}}_{\partial I_p})^T$, formuliert werden. Zunächst liefert das algebraische Verfahren für $n \in \mathbb{N}$ die Gleichungen

$$\mathbf{x}_{I_i}^{(n)} \;=\; -\mathbf{A}_{ii}^{-1}\mathbf{A}_i\widetilde{\mathbf{x}}_{\partial I_i}^{(n)} \;+\; \mathbf{A}_{ii}^{-1}\mathbf{b}_{I_i}\,, \qquad i = 1, \ldots, p\,. \tag{5.22}$$

Mit Hilfe der Kommunikationsmatrix (2.2) kann die Iterierte $\mathbf{x}^{(n)} = \left(\mathbf{x}_{I_1}^{(n)}, \ldots, \mathbf{x}_{I_p}^{(n)}\right)^T$ nur auf den zu versendenden Teil $\mathbf{x}_{\partial}^{(n)} = \left(\mathbf{x}_{\partial I_1}^{(n)}, \ldots, \mathbf{x}_{\partial I_p}^{(n)}\right)^T$ eingeschränkt werden, so daß man aus (5.22) die Gleichung

$$\begin{pmatrix} \mathbf{x}_{\partial I_1} \\ \vdots \\ \mathbf{x}_{\partial I_p} \end{pmatrix}^{(n)} \;=\; -\begin{pmatrix} \widetilde{\mathbf{A}}_{11} & \ldots & \widetilde{\mathbf{A}}_{1p} \\ \vdots & \ldots & \vdots \\ \widetilde{\mathbf{A}}_{p1} & \ldots & \widetilde{\mathbf{A}}_{pp} \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix}^{(n)} \;+\; \begin{pmatrix} \widetilde{\mathbf{b}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{b}}_{\partial I_p} \end{pmatrix} \tag{5.23}$$

erhält, wobei die Blöcke der Matrix

$$\widetilde{\mathbf{A}} = \begin{pmatrix} \widetilde{\mathbf{A}}_{11} & \ldots & \widetilde{\mathbf{A}}_{1p} \\ \vdots & \ldots & \vdots \\ \widetilde{\mathbf{A}}_{p1} & \ldots & \widetilde{\mathbf{A}}_{pp} \end{pmatrix} \in \mathbb{R}^{m \times m}$$

und die des Vektors

$$\widetilde{\mathbf{b}}_\partial = \begin{pmatrix} \widetilde{\mathbf{b}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{b}}_{\partial I_p} \end{pmatrix} \in \mathbb{R}^m$$

mit $m = |\partial I_1| + \ldots + |\partial I_p|$ durch

$$\widetilde{\mathbf{A}}_{ij} \;=\; \mathbf{R}_{\partial I_i} \mathbf{R}_{I_j}^T \mathbf{A}_{jj}^{-1} \mathbf{A}_j \qquad \text{für } i,j = 1,\ldots,p$$

bzw.

$$\widetilde{\mathbf{b}}_{\partial I_i} \;=\; \sum_{j=1}^{p} \mathbf{R}_{\partial I_i} \mathbf{R}_{I_j}^T \mathbf{A}_{jj}^{-1} \mathbf{b}_{I_j} \qquad \text{für } i = 1,\ldots,p$$

gegeben sind. Mit Hilfe der zweiten Zuweisung des Verfahrens 5.1 ergibt sich aus der Gleichung (5.23) das Iterationsverfahren

$$\begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix}^{(n+1)} = - \begin{pmatrix} \widetilde{\mathbf{A}}_{11} & \ldots & \widetilde{\mathbf{A}}_{1p} \\ \vdots & \ldots & \vdots \\ \widetilde{\mathbf{A}}_{p1} & \ldots & \widetilde{\mathbf{A}}_{pp} \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix}^{(n)} + \begin{pmatrix} \widetilde{\mathbf{b}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{b}}_{\partial I_p} \end{pmatrix}. \tag{5.24}$$

Da $\partial I_i \cap I_i = \emptyset$ ist $\widetilde{\mathbf{A}}_{ii} = \mathbf{0}$ für $i = 1,\ldots,p$, so daß insbesondere die Diagonalelemente der Matrix $\widetilde{\mathbf{A}}$ verschwinden. Folglich ist das Iterationsverfahren (5.24) das Jacobi–Verfahren zur Lösung des linearen Gleichungssystems

$$(\widetilde{\mathbf{A}} \,+\, \mathbf{I})\, \widetilde{\mathbf{x}}_\partial \;=\; \widetilde{\mathbf{b}}_\partial \,.$$

Dieses Gleichungssystem soll nun mit einer noch zu bestimmenden Matrix $\widetilde{\mathbf{K}} \in \mathbb{R}^{m \times m}$ vorkonditioniert werden und anschließend mit der durch die Blockstruktur induzierten Zerlegung der Form $\widetilde{\mathbf{K}}(\widetilde{\mathbf{A}} + \mathbf{I}) = \mathbf{W} - \mathbf{V}$ mit $\mathbf{W}, \mathbf{V} \in \mathbb{R}^{m \times m}$ mit dem Verfahren

$$\mathbf{W}\, \widetilde{\mathbf{x}}_\partial^{(n+1)} \;=\; \mathbf{V}\, \widetilde{\mathbf{x}}_\partial^{(n)} \,+\, \widetilde{\mathbf{K}}\, \widetilde{\mathbf{b}}_\partial \tag{5.25}$$

gelöst werden. Dabei soll $\mathbf{W}$ die Struktur

$$\mathbf{W} \;=\; \operatorname{diag}(\mathbf{W}_1,\ldots,\mathbf{W}_p) \qquad \text{mit} \quad \mathbf{W}_i \in \mathbb{R}^{|\partial I_i| \times |\partial I_i|} \quad \text{für } i = 1,\ldots,p$$

besitzen. Man vergleiche diese Vorgehensweise auch mit der Entwicklung des KIPS-Verfahrens. Der Vorkonditionierer $\widetilde{\mathbf{K}}$ soll nun so konstruiert werden, daß das Kommunikationsschema erhalten bleibt. Um dieser Forderung nachzukommen, ist eine andere Anordnung der Komponenten des Vektors $\widetilde{\mathbf{x}}_\partial$ sinnvoll, bei dem der Datenaustausch zwischen den einzelnen Prozessorpaaren $(P_i,\, P_j)$ mit $i < j$ erkennbar ist. Da aber

die Variablen $\widetilde{\mathbf{x}}_{\partial I_i \cap I_j}$ die empfangenen Daten von Prozesor P_j auf Prozesor P_i kennzeichnen und $\widetilde{\mathbf{x}}_{\partial I_j \cap I_i}$ die empfangenen Daten von P_i auf P_j darstellen, beschreibt die Komponente

$$\widehat{\widetilde{\mathbf{x}}}_{ij} = \begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_i \cap I_j} \\ \widetilde{\mathbf{x}}_{\partial I_j \cap I_i} \end{pmatrix} \qquad \text{mit} \quad i < j \tag{5.26}$$

den Datenaustausch zwischen den Prozessoren P_i und P_j. Die so eingeführte Anordnung $\widehat{\widetilde{\mathbf{x}}}_\partial = (\widehat{\widetilde{\mathbf{x}}}_{12}, \ldots, \widehat{\widetilde{\mathbf{x}}}_{1p}, \widehat{\widetilde{\mathbf{x}}}_{23}, \ldots, \widehat{\widetilde{\mathbf{x}}}_{2p}, \ldots, \widehat{\widetilde{\mathbf{x}}}_{p-1p})^T$ des Empfangsvektors $\widetilde{\mathbf{x}}_\partial = (\widetilde{\mathbf{x}}_{\partial I_1}, \ldots, \widetilde{\mathbf{x}}_{\partial I_p})^T$ kann durch eine Permutationsmatrix $\mathbf{S} : \widetilde{\mathbf{x}}_\partial \mapsto \widehat{\widetilde{\mathbf{x}}}_\partial$ angegeben werden. Die Erhaltung der Kommunikationsstruktur bedeutet nun, daß die Matrix $\widehat{\mathbf{K}} = \mathbf{S}\widetilde{\mathbf{K}}\mathbf{S}^{-1}$ eine Blockstruktur der Form

$$\widehat{\mathbf{K}} = \operatorname{diag}(\widehat{\mathbf{K}}_{12}, \ldots, \widehat{\mathbf{K}}_{1p}, \widehat{\mathbf{K}}_{23}, \ldots, \widehat{\mathbf{K}}_{2p}, \ldots, \widehat{\mathbf{K}}_{p-1p})$$

besitzen muß, wobei

$$\widehat{\mathbf{K}}_{ij} = \begin{pmatrix} \mathbf{K}_{ij} & \widetilde{\mathbf{K}}_{ij} \\ \widetilde{\mathbf{K}}_{ji} & \mathbf{K}_{ji} \end{pmatrix}, \qquad i < j,$$

mit

$$\mathbf{K}_{ij} \in \mathbb{R}^{|\partial I_i \cap I_j| \times |\partial I_i \cap I_j|}$$
$$\widetilde{\mathbf{K}}_{ij} \in \mathbb{R}^{|\partial I_i \cap I_j| \times |\partial I_j \cap I_i|}$$

für $i, j = 1, \ldots, p$ und $i \neq j$. Die Matrix $\widehat{\mathbf{K}}_{ij}$ wirkt so durch das Produkt

$$\widehat{\mathbf{K}}_{ij}\, \widehat{\widetilde{\mathbf{x}}}_{ij} = \begin{pmatrix} \mathbf{K}_{ij} & \widetilde{\mathbf{K}}_{ij} \\ \widetilde{\mathbf{K}}_{ji} & \mathbf{K}_{ji} \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_i \cap I_j} \\ \widetilde{\mathbf{x}}_{\partial I_j \cap I_i} \end{pmatrix}$$

nur auf die Komponenten $\widehat{\widetilde{\mathbf{x}}}_{ij}$ des Vektors $\widetilde{\mathbf{x}}_\partial$. Mit dieser Konstruktion kann das Verfahren (5.25) nun weiter analysiert werden. Bezeichnet $\widetilde{\mathbf{R}}_{\partial I_i} : \mathbb{R}^m \to \mathbb{R}^{|\partial I_i|}$ die lineare Abbildung, die durch $\begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_1} \\ \vdots \\ \widetilde{\mathbf{x}}_{\partial I_p} \end{pmatrix} \mapsto \widetilde{\mathbf{x}}_{\partial I_i}$ gegeben ist, so ergibt sich folgendes für die einzelnen Blöcke des Verfahrens (5.25):

$$\widetilde{\mathbf{R}}_{\partial I_i} \widetilde{\mathbf{K}} (\widetilde{\mathbf{A}} + \mathbf{I}) \widetilde{\mathbf{R}}_{\partial I_i}^T \widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} = -\widetilde{\mathbf{R}}_{\partial I_i} \widetilde{\mathbf{K}} (\widetilde{\mathbf{A}} + \mathbf{I}) \sum_{\substack{j=1 \\ j \neq i}}^{p} \widetilde{\mathbf{R}}_{\partial I_j}^T \widetilde{\mathbf{x}}_{\partial I_j}^{(n)} + \widetilde{\mathbf{R}}_{\partial I_i} \widetilde{\mathbf{K}} \widetilde{\mathbf{b}}_\partial$$

$\Longleftrightarrow$

$$\widetilde{\mathbf{R}}_{\partial I_i} \widetilde{\mathbf{K}} (\widetilde{\mathbf{A}} + \mathbf{I}) \widetilde{\mathbf{R}}_{\partial I_i}^T \widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} - \widetilde{\mathbf{R}}_{\partial I_i} \widetilde{\mathbf{K}} \sum_{\substack{j=1 \\ j \neq i}}^{p} \widetilde{\mathbf{R}}_{\partial I_j}^T \widetilde{\mathbf{b}}_{\partial I_j}$$
$$= -\widetilde{\mathbf{R}}_{\partial I_i} \widetilde{\mathbf{K}} (\widetilde{\mathbf{A}} + \mathbf{I}) \sum_{\substack{j=1 \\ j \neq i}}^{p} \widetilde{\mathbf{R}}_{\partial I_j}^T \widetilde{\mathbf{x}}_{\partial I_j}^{(n)} + \widetilde{\mathbf{R}}_{\partial I_i} \widetilde{\mathbf{K}} \widetilde{\mathbf{R}}_{\partial I_i}^T \widetilde{\mathbf{b}}_{\partial I_i}$$

$\Longleftrightarrow$

$$\widetilde{\mathbf{R}}_{\partial I_i}\widetilde{\mathbf{K}}\widetilde{\mathbf{R}}_{\partial I_i}^T\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} - \widetilde{\mathbf{R}}_{\partial I_i}\widetilde{\mathbf{K}}\Big(- \widetilde{\mathbf{A}}\widetilde{\mathbf{R}}_{\partial I_i}^T\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} + \sum_{\substack{j=1\\ j\neq i}}^{p}\widetilde{\mathbf{R}}_{\partial I_j}^T\widetilde{\mathbf{b}}_{\partial I_j}\Big)$$

$$= -\widetilde{\mathbf{R}}_{\partial I_i}\widetilde{\mathbf{K}}\sum_{\substack{j=1\\ j\neq i}}^{p}\widetilde{\mathbf{R}}_{\partial I_j}^T\widetilde{\mathbf{x}}_{\partial I_j}^{(n)} + \widetilde{\mathbf{R}}_{\partial I_i}\widetilde{\mathbf{K}}\Big(- \widetilde{\mathbf{A}}\sum_{\substack{j=1\\ j\neq i}}^{p}\widetilde{\mathbf{R}}_{\partial I_j}^T\widetilde{\mathbf{x}}_{\partial I_j}^{(n)} + \widetilde{\mathbf{R}}_{\partial I_i}^T\widetilde{\mathbf{b}}_{\partial I_i}\Big). \quad (5.27)$$

Bezeichnet nun

$$\mathbf{z}_{\partial}^{(n+1)} = -\widetilde{\mathbf{A}}\widetilde{\mathbf{R}}_{\partial I_i}^T\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} + \sum_{\substack{j=1\\ j\neq i}}^{p}\widetilde{\mathbf{R}}_{\partial I_j}^T\widetilde{\mathbf{b}}_{\partial I_j} \quad (5.28)$$

und

$$\mathbf{y}_{\partial}^{(n)} = -\widetilde{\mathbf{A}}\sum_{\substack{j=1\\ j\neq i}}^{p}\widetilde{\mathbf{R}}_{\partial I_j}^T\widetilde{\mathbf{x}}_{\partial I_j}^{(n)} + \widetilde{\mathbf{R}}_{\partial I_i}^T\widetilde{\mathbf{b}}_{\partial I_i}\,, \quad (5.29)$$

so folgt wegen $\mathbf{A}_{ii} = \mathbf{0}$ aus der Gleichung (5.28)

$$\mathbf{z}_{\partial I_i}^{(n+1)} = \mathbf{0}\,. \quad (5.30)$$

Der Vergleich von (5.29) mit (5.23) ergibt mit $\mathbf{A}_{ii} = \mathbf{0}$ erneut

$$\mathbf{y}_{\partial I_i}^{(n)} = \mathbf{x}_{\partial I_i}^{(n)}\,. \quad (5.31)$$

Da zur Berechnung der Komponenten $\mathbf{x}_{\partial I_j\cap I_i}^{(n+1)}$ in (5.23) nur die Komponenten $\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)}$ und $\widetilde{\mathbf{b}}_{\partial I_j}$ benötigt werden, liefert ein Vergleich von (5.28) mit (5.23)

$$\mathbf{z}_{\partial I_j\cap I_i}^{(n+1)} = \mathbf{x}_{\partial I_j\cap I_i}^{(n+1)} \qquad \text{für } j = 1,\ldots,p \quad (j\neq i)\,. \quad (5.32)$$

Aus dem gleichen Grund erhält man aus (5.29)

$$\mathbf{y}_{\partial I_i\cap I_j}^{(n)} = \mathbf{0} \qquad \text{für } j = 1,\ldots,p \quad (j\neq i)\,. \quad (5.33)$$

Zusammenfassend ergeben die Gleichungen (5.30), (5.31), (5.32) und (5.33) mit Hilfe der Notation (5.26) dann

$$\widehat{\mathbf{z}}_{ij}^{(n+1)} = \begin{pmatrix}\mathbf{z}_{\partial I_i\cap I_j}\\ \mathbf{z}_{\partial I_j\cap I_i}\end{pmatrix}^{(n+1)} = \begin{pmatrix}\mathbf{0}\\ \mathbf{x}_{\partial I_j\cap I_i}\end{pmatrix}^{(n+1)} \quad (5.34)$$

und

$$\widehat{\mathbf{y}}_{ij}^{(n)} \;=\; \begin{pmatrix} \mathbf{y}_{\partial I_i \cap I_j} \\ \mathbf{y}_{\partial I_j \cap I_i} \end{pmatrix}^{(n)} \;=\; \begin{pmatrix} \mathbf{x}_{\partial I_i \cap I_j} \\ \mathbf{0} \end{pmatrix}^{(n)} \tag{5.35}$$

für $j = 1, \ldots, p$ mit $i < j$. Mit den oben eingeführten Vektoren (5.28) und (5.29) ist Gleichung (5.27) wegen $\widetilde{\mathbf{K}} = \mathbf{S}^{-1}\widehat{\mathbf{K}}\mathbf{S}$ nun äquivalent zu

$$\begin{aligned} &\widetilde{\mathbf{R}}_{\partial I_i}\mathbf{S}^{-1}\widehat{\mathbf{K}}\mathbf{S}\widetilde{\mathbf{R}}_{\partial I_i}^{T}\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} - \widetilde{\mathbf{R}}_{\partial I_i}\mathbf{S}^{-1}\widehat{\mathbf{K}}\mathbf{S}\mathbf{z}_{\partial}^{(n+1)} \\ &\qquad = -\widetilde{\mathbf{R}}_{\partial I_i}\mathbf{S}^{-1}\widehat{\mathbf{K}}\mathbf{S}\sum_{\substack{j=1 \\ j\neq i}}^{p}\widetilde{\mathbf{R}}_{\partial I_j}^{T}\widetilde{\mathbf{x}}_{\partial I_j}^{(n)} + \widetilde{\mathbf{R}}_{\partial I_i}\mathbf{S}^{-1}\widehat{\mathbf{K}}\mathbf{S}\mathbf{y}_{\partial}^{(n+1)} \end{aligned}$$

$\Longleftrightarrow$

$$\begin{aligned} &(\mathbf{K}_{ij}, \widetilde{\mathbf{K}}_{ij})\begin{pmatrix} \widetilde{\mathbf{x}}_{\partial I_i \cap I_j} \\ \mathbf{0} \end{pmatrix}^{(n+1)} - (\mathbf{K}_{ij}, \widetilde{\mathbf{K}}_{ij})\widehat{\mathbf{z}}_{ij}^{(n+1)} \\ &\qquad = -(\mathbf{K}_{ij}, \widetilde{\mathbf{K}}_{ij})\begin{pmatrix} \mathbf{0} \\ \widetilde{\mathbf{x}}_{\partial I_j \cap I_i} \end{pmatrix}^{(n)} + (\mathbf{K}_{ij}, \widetilde{\mathbf{K}}_{ij})\widehat{\mathbf{y}}_{ij}^{(n)} \qquad \forall j \neq i\,. \end{aligned}$$

Durch Einsetzen von (5.34) und (5.35) erhält man schließlich

$$\mathbf{K}_{ij}\,\widetilde{\mathbf{x}}_{\partial I_i \cap I_j}^{(n+1)} \;-\; \widetilde{\mathbf{K}}_{ij}\,\mathbf{x}_{\partial I_j \cap I_i}^{(n+1)} \;=\; \mathbf{K}_{ij}\,\mathbf{x}_{\partial I_i \cap I_j}^{(n)} \;-\; \widetilde{\mathbf{K}}_{ij}\,\widetilde{\mathbf{x}}_{\partial I_j \cap I_i}^{(n)} \qquad \forall j \neq i$$

und somit das folgende Verfahren.

Verfahren 5.2 Vorkonditioniertes algebraisches paralleles Schwarz–Verfahren

Eingabe: $\mathbf{A}$, $\mathbf{b}$ und $\mathbf{x}^{(0)}$

for $n = 0, 1, 2, \ldots$ **do**

Löse für $i = 1, \ldots, p$ parallel

$$\mathbf{A}_{ii}\,\mathbf{x}_{I_i}^{(n+1)} \;=\; -\mathbf{A}_i\,\widetilde{\mathbf{x}}_{\partial I_i}^{(n+1)} \;+\; \mathbf{b}_{I_i}\,,$$

wobei

$$\mathbf{K}_{ij}\,\widetilde{\mathbf{x}}_{\partial I_i \cap I_j}^{(n+1)} \;-\; \widetilde{\mathbf{K}}_{ij}\,\mathbf{x}_{\partial I_j \cap I_i}^{(n+1)} \;=\; \mathbf{K}_{ij}\,\mathbf{x}_{\partial I_i \cap I_j}^{(n)} \;-\; \widetilde{\mathbf{K}}_{ij}\,\widetilde{\mathbf{x}}_{\partial I_j \cap I_i}^{(n)}$$

für $j = 1, \ldots, p$ mit $j \neq i$

enddo

5.4 Vorkonditionierung des erweiterten Systems

In Abschnitt 5.2 ergab sich der dem additiven Schwarz–Vorkonditionierer (5.14) entsprechende Vorkonditionierer für das erweiterte Gleichungssystem in der Form

$$\mathbf{M}_{\text{ad}}^{\mathbf{e}} \;=\; \operatorname{diag}\left(\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \;\ldots,\; \begin{pmatrix} \mathbf{A}_{pp} & \mathbf{A}_p \\ \mathbf{0} & \mathbf{I} \end{pmatrix}\right).$$

Aus Verfahren 5.2 erhält man den zugehörigen Vorkonditionierer nun wie folgt. Zunächst ersetzt man für $i = 1, \ldots, p$ die jeweils $2i$-te Blockzeile des erweiterten Gleichungssystems (5.16) durch

$$-\widetilde{\mathbf{K}}_{ij}\,\mathbf{x}_{\partial I_j \cap I_i} \;+\; \mathbf{K}_{ij}\,\widetilde{\mathbf{x}}_{\partial I_i \cap I_j} \;-\; \mathbf{K}_{ij}\,\mathbf{x}_{\partial I_i \cap I_j} \;+\; \widetilde{\mathbf{K}}_{ij}\,\widetilde{\mathbf{x}}_{\partial I_j \cap I_i} \;=\; \mathbf{0}\,,$$

wobei $j = 1, \ldots, p$ mit $j \neq i$. Die so aus $\mathbf{A}^{\mathbf{e}}$ entstandene Matrix wird mit $\mathbf{A}^{\mathbf{v}}$ bezeichnet. Somit ergibt sich $\mathbf{A}^{\mathbf{v}}$ als das Produkt von $\mathbf{A}^{\mathbf{e}}$ mit einer Matrix $\mathbf{K}^{\mathbf{e}}$, deren Elemente durch die Matrizen $\mathbf{K}_{ij}$ und $\widetilde{\mathbf{K}}_{ij}$ gegeben sind, und das nur auf die Blockzeilen $2, 4, \ldots, 2p$ wirkt. Bezeichnet $\mathbf{B}^{\mathbf{v}}$ die Blockdiagonale von $\mathbf{A}^{\mathbf{v}} = \mathbf{K}^{\mathbf{e}}\,\mathbf{A}^{\mathbf{e}}$ mit

$$\mathbf{B}^{\mathbf{v}} \;=\; \operatorname{diag}\left(\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_1 \\ \widetilde{\mathbf{A}}_{11} & \widetilde{\mathbf{A}}_1 \end{pmatrix},\; \ldots,\; \begin{pmatrix} \mathbf{A}_{pp} & \mathbf{A}_p \\ \widetilde{\mathbf{A}}_{pp} & \widetilde{\mathbf{A}}_p \end{pmatrix} \right),$$

wobei $\widetilde{\mathbf{A}}_{ii} \in \mathbb{R}^{|\partial I_i| \times |I_i|}$ und $\widetilde{\mathbf{A}}_i \in \mathbb{R}^{|\partial I_i| \times |\partial I_i|}$ für $i = 1, \ldots, p$, so ergibt sich der neue Vorkonditionierer von $\mathbf{A}^{\mathbf{e}}$ durch

$$\mathbf{M}^{\mathbf{e}} = {\mathbf{B}^{\mathbf{v}}}^{-1}\mathbf{K}^{\mathbf{e}}\,.$$

Hierzu wird das folgende Beispiel betrachtet. Sei

$$\mathbf{A} = \begin{pmatrix} 2/h^2 & -1/h^2 & & & & & \\ -1/h^2 & 2/h^2 & -1/h^2 & & & & \\ & -1/h^2 & 2/h^2 & -1/h^2 & & & \\ & & -1/h^2 & 2/h^2 & -1/h^2 & & \\ & & & -1/h^2 & 2/h^2 & -1/h^2 & \\ & & & & -1/h^2 & 2/h^2 & \end{pmatrix}$$

die Matrix aus Beispiel (2.4.1). Zu der dort angegebenen Zerlegung

$$\begin{aligned} I_1 &= \{1,2,3\}, & I_2 &= \{4,5,6\}\,, \\ \partial I_1 &= \{4\}, & \partial I_2 &= \{3\} \end{aligned}$$

der Indexmenge $I = \{1, \ldots, 6\}$ hat man die erweiterte Matrix

$$\mathbf{A}^{\mathbf{e}} = \begin{pmatrix} 2/h^2 & -1/h^2 & & & & & & \\ -1/h^2 & 2/h^2 & -1/h^2 & & & & & \\ & -1/h^2 & 2/h^2 & -1/h^2 & & & & \\ & & & 1 & -1 & & & \\ & & & & 2/h^2 & -1/h^2 & & -1/h^2 \\ & & & & -1/h^2 & 2/h^2 & -1/h^2 & \\ & & & & & -1/h^2 & 2/h^2 & \\ & & -1 & & & & & 1 \end{pmatrix}.$$

Da das vorkonditionierte algebraische parallele Verfahren für dieses Beispiel das KIPS-Verfahren aus Kapitel 4 ist, lautet

$$\mathbf{A^v} = \begin{pmatrix} 2/h^2 & -1/h^2 & & & & & & \\ -1/h^2 & 2/h^2 & -1/h^2 & & & & & \\ & -1/h^2 & 2/h^2 & -1/h^2 & & & & \\ & & -\alpha & \mu & -\mu & & & \alpha \\ & & & & 2/h^2 & -1/h^2 & & -1/h^2 \\ & & & & -1/h^2 & 2/h^2 & -1/h^2 & \\ & & & & & -1/h^2 & 2/h^2 & \\ & & -\nu & \beta & -\beta & & & \nu \end{pmatrix}.$$

Folglich ist $\mathbf{K^e}$ durch

$$\mathbf{K^e} = \begin{pmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & \mu & & & & \alpha \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & \beta & & & & \nu \end{pmatrix}$$

und $\mathbf{B^v}$ durch

$$\mathbf{B^v} = \begin{pmatrix} 2/h^2 & -1/h^2 & & & & & & \\ -1/h^2 & 2/h^2 & -1/h^2 & & & & & \\ & -1/h^2 & 2/h^2 & -1/h^2 & & & & \\ & & -\alpha & \mu & & & & \\ & & & & 2/h^2 & -1/h^2 & & -1/h^2 \\ & & & & -1/h^2 & 2/h^2 & -1/h^2 & \\ & & & & & -1/h^2 & 2/h^2 & \\ & & & & -\beta & & & \nu \end{pmatrix}$$

gegeben. Es liegen nun zwei Möglichkeiten zur Wahl der Parameter α, β, μ und ν nahe. Zum einen entspricht die Wahl $\mu = \nu = 1$ mit dem noch freien Parameter $\alpha = \beta \in [0, 1)$ dem Übergang von KIPS nach KIPS' im vierten Kapitel. Zum anderen ergibt $\alpha = \beta = 1/h^2$ und $\mu = \nu = 2/h^2$ die Übereinstimmung der Matrix $\mathbf{B^v}$ in den erweiterten Blöcken mit dem Diskretisierungsschema, d.h. mit der Ausgangsmatrix $\mathbf{A}$. Beide Möglichkeiten lassen sich nun auf das Verfahren 5.2 zur Lösung linearer Gleichungssysteme mit einer beliebigen Koeffizientenmatrix $\mathbf{A}$ übertragen. Hierdurch ergeben sich die beiden folgenden Verfahren mit den wie oben definierten Vorkonditionierern.

Verfahren 5.3 Algebraisches paralleles Schwarz–Verfahren mit Parameter α

Verfahren 5.2 mit der folgenden Wahl der Matrizen $\mathbf{K}_{ij}$ und $\widetilde{\mathbf{K}}_{ij}$:

$$\mathbf{K}_{ij} = \mathbf{R}_{\partial I_i \cap I_j}\, \mathbf{R}^T_{\partial I_i \cap I_j} \quad \text{und} \quad \widetilde{\mathbf{K}}_{ij} = \alpha\, \mathbf{R}_{\partial I_i \cap I_j}\, \mathbf{R}^T_{\partial I_j \cap I_i}$$

für $i, j = 1, \ldots, p$ und $i \neq j$.

Soll wie oben beschrieben die Matrix $\mathbf{B}^{\mathbf{v}}$ in deren erweiterten Blöcken mit der Ausgangsmatrix $\mathbf{A}$ übereinstimmen, d.h.

$$\mathbf{B}^{\mathbf{v}} = \operatorname{diag}\left(\begin{pmatrix} \mathbf{R}_{I_1} \\ \mathbf{R}_{\partial I_1} \end{pmatrix} \mathbf{A} \begin{pmatrix} \mathbf{R}_{I_1} \\ \mathbf{R}_{\partial I_1} \end{pmatrix}^T, \ldots, \begin{pmatrix} \mathbf{R}_{I_p} \\ \mathbf{R}_{\partial I_p} \end{pmatrix} \mathbf{A} \begin{pmatrix} \mathbf{R}_{I_p} \\ \mathbf{R}_{\partial I_p} \end{pmatrix}^T \right),$$

so erhält man Verfahren 5.4.

Verfahren 5.4 Erweitertes Block–Verfahren

Verfahren 5.2 mit der folgenden Wahl der Matrizen $\mathbf{K}_{ij}$ und $\widetilde{\mathbf{K}}_{ij}$:

$$\mathbf{K}_{ij} = \mathbf{R}_{\partial I_i \cap I_j}\, \mathbf{A}\, \mathbf{R}^T_{\partial I_i \cap I_j} \quad \text{und} \quad \widetilde{\mathbf{K}}_{ij} = -\mathbf{R}_{\partial I_i \cap I_j}\, \mathbf{A}\, \mathbf{R}^T_{\partial I_j \cap I_i}$$

für $i, j = 1, \ldots, p$ und $i \neq j$.

5.5 Numerische Experimente

Auch in diesem Abschnitt wird das Verhalten der entwickelten Verfahren anhand einer Diskretisierung der in Beispiel 3.7.1 gegebenen partiellen Differentialgleichung demonstriert. Das in Abschnitt 3.7 beschriebenen Diskretisierungsschema sowie die dort angegebene Datenverteilung werden auch hier zugrunde gelegt.

In Abbildung 5.3 ist der Residuumsverlauf der Verfahren 5.1, 5.3 und 5.4 dargestellt. Dabei zeigen die beiden neuen Verfahren ein ähnliches und deutlich besseres Konvergenzverhalten als Verfahren 5.1. Die besten Ergebnisse mit Verfahren 5.3 wurden hier mit $\alpha = 0.6$ erzielt. Für verschiedene Prozessoranzahlen wird in Abbildung 5.4 die benötigten Iterationsanzahlen der verschiedenen Verfahren angegeben,

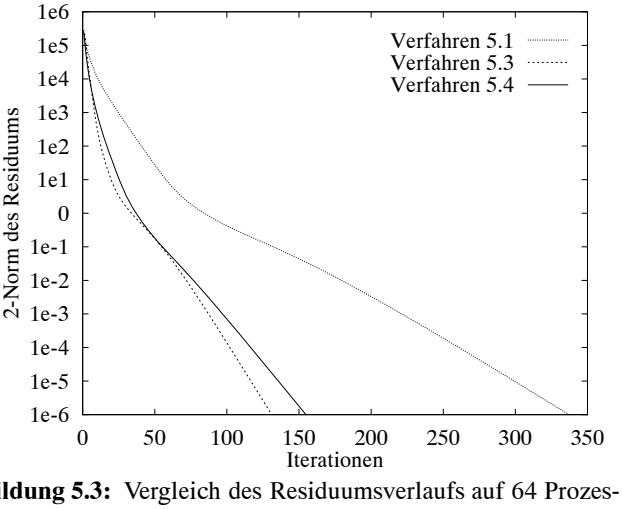


Abbildung 5.3: Vergleich des Residuumsverlaufs auf 64 Prozessoren.

um eine Genauigkeit von 10^{-6} zu erreichen. Dabei werden hier die lokalen Gleichungssysteme mit dem Gauß–Seidel–Verfahren gelöst, das nach 20 Schritten abgebrochen wird. Während Verfahren 5.1 mit steigender Prozessoranzahl deutlich mehr Iterationen benötigt, bleibt diese bei den Verfahren 5.3 und 5.4 weitgehend konstant. Dieses günstige Verhalten spiegelt sich auch in Abbildung 5.5 wider, die den Speedup der drei Verfahren zeigt.

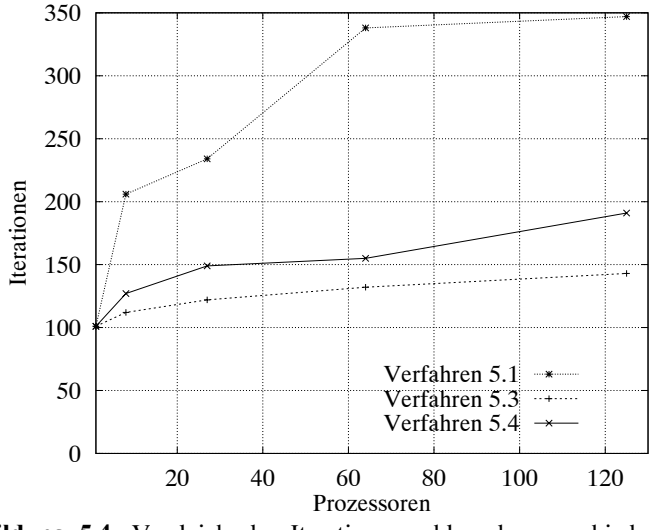


Abbildung 5.4: Vergleich der Iterationsanzahlen der verschiedenen Verfahren

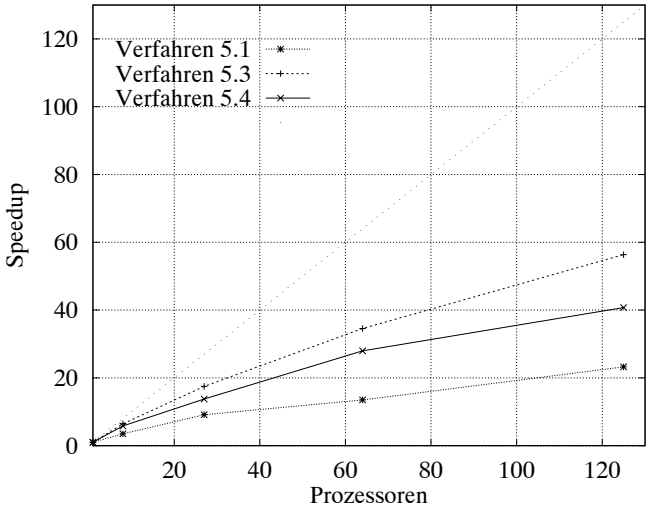


Abbildung 5.5: Speedup der verschiedenen Verfahren

Zum Abschluß des Kapitels wird das Verfahren 5.4 als Vorkonditionierer im Krylov–Teilraumverfahren Algorithmus 5.1 eingesetzt. Hierzu wird das folgende Beispiel

betrachtet.

Beispiel 5.5.1. *Gegeben sei die partielle Differentialgleichung*

$$-\Delta u \;=\; f \quad in \quad \Omega = (0,13) \times (0,1) \times (0,1)$$

mit homogener Dirichlet–Randbedingung, wobei die rechte Seite f so gewählt wird, daß

$$u(x,y,z) \;=\; x(13-x)y(1-y)z(1-z)$$

Lösung der Differentialgleichung ist.

Zur Diskretisierung werden auch hier zentrale Differenzen zweiter Ordnung auf einem $480 \times 36 \times 36$ Gitter der Schrittweite $1/37$ verwendet. Das resultierende Gleichungssystem mit der Koeffizientenmatrix $\mathbf{A}$ besitzt die Ordnung $N = 622080$ mit 4282848 Nichtnullelementen. Die Testrechnungen wurden mit $\mathbf{x}_0 = \mathbf{0}$ durchgeführt und bei einer Genauigkeit von 10^{-4} in der euklidischen Norm des Residuums beendet. Die Datenverteilung auf die p Prozessoren wurde mit Hilfe einer Streifenzerlegung des Gebietes Ω in x-Richtung vorgenommen. Die Daten wurden dabei gleichmäßig auf die Prozessoren verteilt. In Abbildung 5.6 wird die Anzahl der benötigten Iterationen für verschiedene Prozessorzahlen dargestellt. Die obere Kurve zeigt die Ergebnisse für den aus Verfahren 5.1 resultierenden additiven Schwarz–Vorkonditionierer zur Lösung des linearen Gleichungssystems $\mathbf{A}\mathbf{x} = \mathbf{b}$ mit Algorithmus 5.1. Dagegen bezieht sich die untere Kurve auf den aus Verfahren 5.4 resultierenden Vorkonditionierer für das erweiterte Gleichungssystem $\mathbf{A^e x^e} = \mathbf{b^e}$. Auch hier ist das neue Verfahren 5.4 dem ursprünglichen Verfahren 5.1 überlegen.

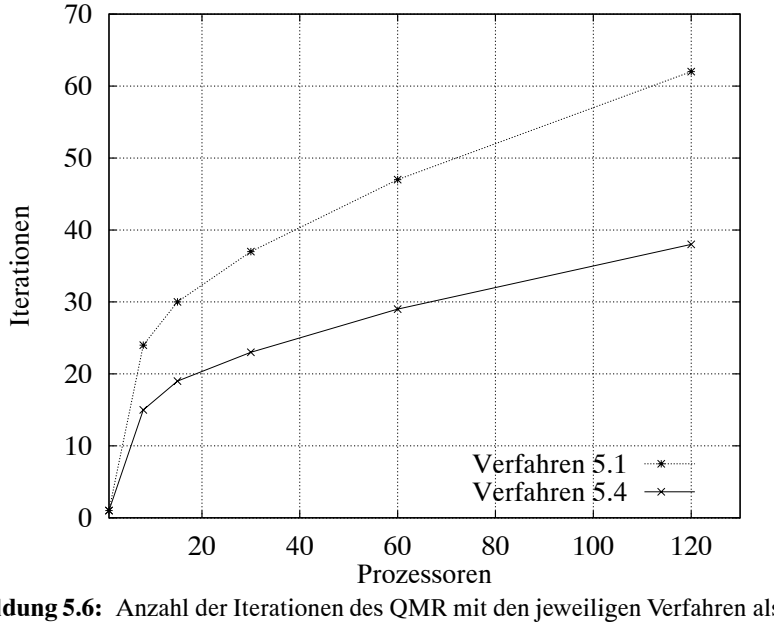


Abbildung 5.6: Anzahl der Iterationen des QMR mit den jeweiligen Verfahren als Vorkonditionierer

Kapitel 6

Zusammenfassung und Ausblick

Viele Problemstellungen in den Ingenieur- und Naturwissenschaften werden mit Hilfe von partiellen Differentialgleichungen behandelt. In praxisrelevanten Anwendungen ist aber die numerische Lösung dieser Gleichungen äußerst speicher- und rechenintensiv, so daß hierfür zunehmend massiv-parallele Systeme eingesetzt werden müssen. Da auf solchen Systemen neben den Rechenzeiten auch Kommunikationszeiten anfallen, ist die Entwicklung schneller paralleler Algorithmen mit niedrigen Kommunikationskosten von zentraler Bedeutung.

Die numerische Lösung partieller Differentialgleichungen führt nach einer geeigneten Diskretisierung auf die Lösung linearer Gleichungssysteme $\mathbf{A}\mathbf{x} = \mathbf{b}$, deren Koeffizientenmatrizen häufig groß und dünnbesetzt sind. Für diese Problemstellung eignen sich daher iterative Verfahren, in denen die Koeffizientenmatrix ausschließlich zur Berechnung von Matrix-Vektor-Produkten verwendet wird. Spezielle iterative Verfahren sind Krylov–Teilraumverfahren, die zur Lösung dieser Systeme mit allgemeiner Koeffizientenmatrix eingesetzt werden können. Aus Speicherplatzgründen werden hier Krylov–Verfahren entwickelt, deren Iterierte $\mathbf{x}_n$ mit kurzen Rekursionen berechnet werden können. In dieser Arbeit wird der im Krylov–Verfahren erzeugte Teilraum mit Hilfe des unsymmetrischen Lanczos–Prozesses generiert. Aus Gründen der numerischen Stabilität werden dazu gekoppelte Zwei-Term-Rekursionen und normierte Lanczos–Vektoren verwendet. In dem Lanczos–basierten Krylov–Teilraumverfahren (Algorithmus 3.5) kann das Residuum $\mathbf{r}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n$ äquivalent durch das Produkt $\mathbf{r}_n = \mathbf{V}_{n+1}\mathbf{s}_n$ beschrieben werden. Dabei bezeichnet $\mathbf{V}_{n+1}$ eine Matrix, deren Spalten aus den erzeugten normierten Lanczos–Vektoren bestehen, und $\mathbf{s}_n$ ist ein Vektor, der mit Hilfe eines überbestimmten linearen Gleichungssystems bestimmt wird. Dieser Vektor definiert die Iterierte $\mathbf{x}_n$ und wird in dem von Freund und Nachtigal eingeführten Verfahren der quasi-minimalen Residuen in der euklidischen Norm minimiert.

Im Vergleich zu der bekannten l_2-Minimierung liefert eine allgemeine l_p-Minimierung mit $1 \leq p \leq \infty$ einen zusätzlichen Parameter, der zur Beschleunigung des Konvergenzverhaltens ausgenutzt werden kann. In dieser Arbeit werden Techniken zur effizienten Lösung der dabei auftetenden Minimierungsprobleme bereitgestellt. Die daraus resultierenden Krylov–Teilraumverfahren zur Lösung linearer Gleichungssy-

steme können mit kurzen Rekursionen beschrieben und implementiert werden. Auf diese Weise entsteht die Klasse der quasi-l_p-minimalen Residuenverfahren (Algorithmus 3.8). Der Fall $p = 1$ stellt dabei eine Verbindung zum Verfahren der bikonjugierten Gradienten her, mit dem die jeweils beste BCG-Iterierte als eine Quasi-l_1-Minimierung charakterisiert werden kann (Algorithmus 3.7).

Bei dem Entwurf der Lanczos–basierten Krylov–Teilraumverfahren werden numerische Stabilität und Aspekte der Parallelverarbeitung miteinander verknüpft. Die numerische Stabilität wird durch eine Normierung der Lanczos–Vektoren sowie die Verwendung von gekoppelten Zwei-Term-Rekursionen gesichert. Hinsichtlich der Parallelverarbeitung wird bei der Herleitung ein Ansatz verfolgt, der es erlaubt, alle anfallenden Skalarprodukte eines Iterationsschrittes mit nur einem globalen Synchronisationspunkt zu berechnen. Es fallen dabei im Vergleich zur ursprünglichen Version keine zusätzlichen Vektoroperationen an. Durch die Reduktion der Anzahl der globalen Synchronisationspunkte wird der Vorteil der bereitgestellten Variante speziell beim Einsatz von massiv-parallelen Systemen deutlich. Doch auch aus Gründen der Lastverteilung sind Verfahren mit nur einem globalen Synchronisationspunkt von besonderer Bedeutung.

Die Implementierung von iterativen Verfahren auf Parallelrechnern mit verteiltem Speicher erfolgt durch die Parallelisierung der aufretenden Operationen, bei denen das Matrix-Vektor-Produkt die aufwendigste darstellt. Dazu werden sowohl die Elemente der Matrix als auch die Komponenten der Vektoren auf die einzelnen Prozessoren verteilt. Die Kommunikation, die nun bei der parallelen Ausführung dieses Produktes anfällt, induziert ein für das iterative Verfahren charakteristisches Kommunikationsschema, das in jedem Iterationsschritt des Verfahrens benutzt werden muß. Wie in der Arbeit beschrieben, müssen aber bei der Berechnung der Matrix-Vektor-Produkte zusätzliche Variablen für den Datenempfang bereitgestellt werden. Auf diese Weise entsteht der Begriff des durch den Empfangsvektor erweiterten Vektors $\mathbf{x}^{\mathbf{e}}$ (Definition 2.3.5). Für diesen Vektor läßt sich mit Hilfe der erweiterten Matrix $\mathbf{A}^{\mathbf{e}}$ (Definition 2.4.1) ein zu $\mathbf{Ax} = \mathbf{b}$ äquivalentes lineares Gleichungssystem $\mathbf{A}^{\mathbf{e}}\mathbf{x}^{\mathbf{e}} = \mathbf{b}^{\mathbf{e}}$ aufstellen, in dem die Kommunikationsstruktur des Verfahrens enthalten ist. Da gezeigt werden konnte (Satz 2.4.1), daß die Lösungen beider Systeme gleich sind, kann anstelle des ursprünglichen Systems das erweiterte lineare Gleichungssystem gelöst werden. Die zusätzlichen Variablen, die auf einem parallelen System mit verteiltem Speicher als Empfangsvariablen notwendig sind, werden dann dazu benutzt, das iterative Verfahren zu beschleunigen. Durch die Integration des Kommunikationsschemas in das zu lösende System können Beschleunigungsmethoden entwickelt werden, die keine zusätzlichen Kommunikationskosten verursachen. Die Kommunikationsstruktur des Algorithmus bleibt dabei erhalten.

Da das parallele Schwarz–Verfahren (Verfahren 4.1) eine dem Matrix-Vektor-Produkt ähnliche Kommunikationsstruktur besitzt, wird zur Entwicklung von Beschleunigungstechniken für iterative Verfahren zunächst eine Beschleunigung des parallelen Schwarz–Verfahrens auf analytischer Ebene durchgeführt. Dazu wird eine Randintegralmethode verwendet, die es ermöglicht, das Verfahren allein durch die auszutau-

schenden Daten zu beschreiben. Diese Datenreduktion erlaubt es, Beschleunigungen anzugeben, die auch die bestehende Kommunikationsstruktur erhalten. Die daraus resultierenden beschleunigten parallelen Schwarz–Verfahren, deren Konvergenzverhalten ohne die Verwendung zusätzlicher Kommunikation verbessert wird, sind sogar optimal (im Sinne von Satz 4.4.2).

Mit dem algebraischen Schwarz–Verfahren (Verfahren 5.1) werden schließlich die Ergebnisse der obigen analytischen Vorgehensweise auf das erweiterte Gleichungssystem übertragen, das hier mit einer diskreten Randintegralmethode auf die auszutauschenden Daten reduziert wird. Auf diese Weise sind wiederum Beschleunigungen des algebraischen Verfahrens möglich, die auch die Kommunikationsstruktur erhalten (Verfahren 5.3 und 5.4). Abschließend wird gezeigt, wie die so gewonnenen Verfahren als Vorkonditionierer in den bereitgestellten Krylov–Teilraumverfahren eingesetzt werden können. Auch hier wird bei der Beschleunigung der Verfahren keine zusätzliche Kommunikation benötigt.

Die Überlegenheit der entwickelten Verfahren wird anhand der Implementierung verschiedener dreidimensionaler Modellprobleme auf dem massiv-parallelen System Intel Paragon XP/S 10 gezeigt. Die beschleunigten Verfahren reduzieren hier die zur Konvergenz benötigte Iterationsanzahl des parallelen Schwarz–Verfahrens erheblich. Da alle Verfahren in jedem Iterationsschritt den gleichen Rechen- und Kommunikationsaufwand haben, führt dieses zu wesentlich kürzeren Gesamtausführungzeiten. In den behandelten Beispielen ergeben sich so Zeitersparnisse von über 50 Prozent. Darüber hinaus übertragen sich diese Eigenschaften auch auf entsprechend vorkonditionierte Krylov–Teilraumverfahren, so daß auch hier die Iterationsanzahl um 40 Prozent reduziert werden kann.

Das Ziel der Arbeit war es, neue Verfahren zur Lösung linearer Gleichungssysteme zu entwickeln und gleichzeitig auch Möglichkeiten aufzuzeigen, wie bestehende Verfahren beschleunigt werden können. Dabei standen Verfahren im Vordergrund, die in realen Anwendungen ihren Einsatz finden. So werden beispielsweise sowohl das parallele Schwarz–Verfahren in TRACE als auch das Verfahren der quasi-minimalen Residuen in PARTRACE im Forschungszentrum Jülich zur Simulation von Schadstoffausbreitungen im Boden und von Stofftransporten in porösen Medien verwendet. Die Rechenzeiten dieser Simulationen können so mit den aufgeführten Beschleunigungstechniken erheblich verkürzt werden.

Literaturverzeichnis

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

[2] Z. Bai. Error Analysis of the Lanczos Algorithm for the Nonsymmetric Eigenvalue Problem. *Mathematics of Computation*, 62(205):209–226, 1994.

[3] S. T. Barnard and H. D. Simon. Fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. *Concurrency: Practice and Experience*, 6(2):101–117, 1994.

[4] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, 1993.

[5] R. Berrendorf, H. C. Burg, U. Detert, R. Esser, M. Gerndt, and R. Knecht. Intel Paragon XP/S – architecture, software, and performance. Interner Bericht KFA–ZAM–IB–9605, Forschungszentrum Jülich GmbH, Jülich, Mai 1994.

[6] C. Brezinski, M. R. Zaglia, and H. Sadok. A Breakdown–Free Lanczos Type Algorithm for Solving Linear Systems. *Numerische Mathematik*, 63(1):29–38, 1992.

[7] H. M. Bücker. Isoefficiency Analysis of Parallel QMR-Like Iterative Methods and its Implications on Parallel Algorithm Design. Interner Bericht KFA–ZAM–IB–9604, Forschungszentrum Jülich GmbH, Jülich, Germany, March 1996.

[8] H. M. Bücker and M. Sauren. A Parallel Version of the Quasi–Minimal Residual Method Based on Coupled Two–Term Recurrences. In J. Waśniewski, J. Dongarra, K. Madsen, and D. Olesen, editors, *Applied Parallel Computing: Industrial Computation and Optimization, Proceedings of the Third International Workshop, PARA '96, Lyngby, Denmark, August 18–21, 1996*, volume 1184 of *Lecture Notes in Computer Science*, pages 157–165, Berlin, 1996. Springer.

[9] H. M. Bücker and M. Sauren. A Parallel Version of the Unsymmetric Lanczos Algorithm and its Application to QMR. Interner Bericht KFA–ZAM–IB–9605, Forschungszentrum Jülich GmbH, Jülich, Germany, March 1996.

[10] H. M. Bücker and M. Sauren. A Variant of the Biconjugate Gradient Method Suitable for Massively Parallel Computing. In *Fourth International Symposium on Solving Irregularly Structured Problems in Parallel, IRREGULAR'97*, to be published in Lecture Notes in Computer Science, Berlin, 1997. Springer. Also available in extended form as Internal Report KFA–ZAM–IB– 9702, Research Centre Jülich, Jülich, Germany, April 1997.

[11] T.F. Chan, R. Glowinski, J. Périaux, and O.B. Widlund, editors. *Second International Symposium on Domain Decomposition Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1989.

[12] T.F. Chan, R. Glowinski, J. Périaux, and O.B. Widlund, editors. *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1990.

[13] T.F. Chan and T.P. Mathew. Domain decomposition algorithms. In *Acta Numerica 1994*, pages 61–143. Cambridge University Press, Cambridge, 1994.

[14] M. Cosnard and D. Trystram. *Parallel Algorithms and Architectures*. International Thomson Computer Press, London, 1995.

[15] J. Cullum and R. A. Willoughby. A Practical Procedure for Computing Eigenvalues of Large Sparse Nonsymmetric Matrices. In J. Cullum and R. A. Willoughby, editors, *Large Scale Eigenvalue Problems, Proceedings of the IBM Europe Institute Workshop on Large Scale Eigenvalue Problems, Oberlech, Austria, July 8–12, 1985*, number 127 in North-Holland Mathematics Studies, pages 193–240, Amsterdam, The Netherlands, 1986. North-Holland.

[16] R. D. da Cunha and T. Hopkins. The Parallel Iterative Methods (PIM) Package for the Solution of Systems of Linear Equations on Parallel Computers. *Applied Numerical Mathematics*, 19(1–2):33–50, 1995.

[17] E. de Sturler. A Parallel Variant of GMRES(m). Reports of the Faculty of Technical Mathematics and Informatics 91–85, Delft University of Technology, Delft, The Netherlands, 1991.

[18] E. de Sturler. A Performance Model for Krylov Subspace Methods on Mesh–based Parallel Computers. Technical Report CSCS–TR–94–05, Swiss Scientific Computing Center, CH–6928 Manno, Switzerland, May 1994.

[19] E. de Sturler and H. A. van der Vorst. Reducing the Effect of Global Communication in GMRES(m) and CG on Parallel Distributed Memory Computers. Technical Report 832, University of Utrecht, Utrecht, The Netherlands, October 1993.

[20] E. F. Van de Velde. *Concurrent Scientific Computing*. Number 16 in Texts in Applied Mathematics. Springer, New–York, 1994.

[21] J. W. Demmel, M. T. Heath, and H. A. van der Vorst. Parallel Numerical Linear Algebra. In *Acta Numerica 1993*, pages 111–197. Cambridge University Press, Cambridge, 1993.

[22] B. Engquist and K.-H. Zhao. Analysis of Generalized Schwarz Alternating Procedure for Domain Decomposition. In *Proceedings of the Copper Mountain Conference on Iterative Methods*. SIAM, April 9-13 1996.

[23] D. Evans and L. Kang. The Convergence Rate of the Schwarz Alternating Procedure(III) – for Neumann Problems. *Int. J. Comput. Math.*, 21:85–108, 1987.

[24] D. Evans, L. Kang, and J. Shao. The Convergence Rate of the Schwarz Alternating Procedure(I) – for One-dimensional Problems. *Int. J. Comput. Math.*, 20:157–170, 1986.

[25] D. Evans, L. Kang, J. Shao, and Y.Chen. The Convergence Rate of the Schwarz Alternating Procedure(II) – for Two-dimensional Problems. *Int. J. Comput. Math.*, 20:325–339, 1986.

[26] V. Faber and T. Manteuffel. Necessary and sufficient conditions for the existence of a conjugate gradient method. *SIAM Journal on Numerical Analysis*, 21:352–362, 1984.

[27] R. Fletcher. Conjugate Gradient Methods for Indefinite Systems. In G. A. Watson, editor, *Numerical Analysis Dundee 1975*, volume 506 of *Lecture Notes in Mathematics*, pages 73–89, Berlin, 1976. Springer.

[28] R. W. Freund, G. H. Golub, and N. M. Nachtigal. Iterative Solution of Linear Systems. In *Acta Numerica 1992*, pages 1–44. Cambridge University Press, Cambridge, 1992.

[29] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal. An Implementation of the Look–Ahead Lanczos Algorithm for Non–Hermitian Matrices. *SIAM Journal on Scientific Computing*, 14(1):137–158, 1993.

[30] R. W. Freund and N. M. Nachtigal. QMR: A Quasi–Minimal Residual Method for Non–Hermitian Linear Systems. *Numerische Mathematik*, 60(3):315–339, 1991.

[31] R. W. Freund and N. M. Nachtigal. An Implementation of the QMR Method Based on Coupled Two–Term Recurrences. *SIAM Journal on Scientific Computing*, 15(2):313–337, 1994.

[32] J. R. Gilbert, G. L. Miller, and S. H. Teng. Geometric mesh partitioning: implementation and experiments. In *Proceedings 9th International Parallel Processing Symposium*, pages 418–427. IEEE Computer Society Press, 1995.

[33] R. Glowinski, G.H. Golub, G.A. Meurant, and J. Périaux, editors. *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1988.

[34] R. Glowinski, Y.A. Kuznetsov, G. Meurant, J. Périaux, and O.B. Widlund, editors. *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1991.

[35] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, second edition, 1989.

[36] A. Gupta, V. Kumar, and A. Sameh. Performance and Scalability of Preconditioned Conjugate Gradient Methods on Parallel Computers. Technical Report TR 92–64, Department of Computer Science, University of Minnesota, Minneapolis, MN – 55455, November 1992. Revised April 1994.

[37] M. H. Gutknecht. A Completed Theory of the Unsymmetric Lanczos Process and Related Algorithms, Part I. *SIAM Journal on Matrix Analysis and Applications*, 13(2):594–639, 1992.

[38] M. H. Gutknecht. A Completed Theory of the Unsymmetric Lanczos Process and Related Algorithms, Part II. *SIAM Journal on Matrix Analysis and Applications*, 15(1):15–58, 1994.

[39] W. Hackbusch. *Theorie und Numerik elliptischer Differentialgleichungen*. Teubner, Stuttgart, 1986.

[40] W. Hackbusch. *Iterative Lösung großer schwachbesetzter Gleichungssysteme*. Teubner, Stuttgart, 1993.

[41] M. Hestenes and E. Stiefel. Methods of Conjugate Gradients for Solving Linear Systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[42] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, 1996.

[43] R. W. Hockney and C. R. Jesshope. *Parallel Computers 2*. Adam Hilger, Bristol, second edition, 1988.

[44] F. Hoßfeld. *Parallele Algorithmen*. Informatik Fachberichte 64. Springer, Heidelberg, 1983.

[45] T. Huckle. Low-Rank Modification of the Unsymmetric Lanczos Algorithm. *Mathematics of Computation*, 64(212):1577–1588, 1995.

[46] Intel Supercomputing Systems Division, Beaverton, Oregon. *Paragon User's Guide*, April 1996. Order Number: 312491-005.

[47] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. Technical Report TR 95-035, Department of Computer Science, University of Minnesota, 1995.

[48] W. Kerner. Large–Scale Complex Eigenvalue Problems. *Journal of Computational Physics*, 85(1):1–85, 1989.

[49] D.E. Keyes, T.F. Chan, G. Meurant, J.S. Scroggs, and R.G. Voigt, editors. *Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992.

[50] D.E. Keyes and J. Xu, editors. *Domain Decomposition Methods in Science and Engineering*, volume 180 of *Contemporary Mathematics*. American Mathematical Society, 1994. Proceedings of the Seventh International Conference on Domain Decomposition.

[51] S. K. Kim and A. T. Chronopoulos. An Efficient Nonsymmetric Lanczos Method on Parallel Vector Computers. *Journal of Computational and Applied Mathematics*, 42:357–374, 1992.

[52] C. Lanczos. An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators. *Journal of Research of the National Bureau of Standards*, 45(4):255–282, 1950.

[53] C. Lanczos. Solutions of Systems of Linear Equations by Minimized Iterations. *Journal of Research of the National Bureau of Standards*, 49(1):33–53, 1952.

[54] G. Lindenmayr. Untersuchungen an Modellproblemen zur Konvergenz des Schwarzschen Verfahrens der Gebietszerlegung. Interner Bericht KFA/ICG-4 Interner Bericht Nr. 500995, Forschungszentrum Jülich GmbH, Jülich, Germany, June 1995.

[55] P.L. Lions. On the Schwarz Alternating Method. I. *First International Symposium on Domain Decomposition Methods for Partial Differential Equations. SIAM, Philadelphia*, pages 1–42, 1988.

[56] P.L. Lions. On the Schwarz Alternating Method III: A Variant for Nonoverlapping Subdomains. *Proceedings of Third International conference on Domain Decomposition Methods. SIAM, Philadelphia*, pages 202–223, 1990.

[57] G.L. Miller, S.H. Teng, W. Thurston, and S.A. Vavasis. Automatic mesh partitioning. In A. George, J. Gilbert, and J. Liu, editors, *Sparse Matrix Computations: Graph Theory Issues and Algorithms*, Volumes in Mathematics and Its Applications. IMA, 1993.

[58] A. Mlynski-Wiese. Leistungsuntersuchung des iPSC/860 RISC-Knoten-Prozessors: Architekturanalyse und Programmoptimierung. Bericht Jül-2766, Forschungszentrum Jülich GmbH, ZAM, Jülich, Mai 1993.

[59] W.E. Nagel, editor. *Partielle Differentialgleichungen, Numerik und Anwendungen*, volume 18 of *Konferenzen des Forschungszentrums Jülich*, Jülich, Deutschland, 1996. Forschungszentrum Jülich GmbH.

[60] B. N. Parlett. Reduction to Tridiagonal Form and Minimal Realizations. *SIAM Journal on Matrix Analysis and Applications*, 13(2):567–593, 1992.

[61] B. N. Parlett, D. R. Taylor, and Z. A. Liu. A Look–Ahead Lanczos Algorithm for Unsymmetric Matrices. *Mathematics of Computation*, 44(169):105–124, 1985.

[62] A. Pothen, H. D. Simon, and K. P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430–452, 1990.

[63] A. Quarteroni, J. Périaux, Y.A. Kuznetsov, and O.B. Widlund, editors. *Domain Decomposition Methods in Science and Engineering*, volume 157 of *Contemporary Mathematics*. American Mathematical Society, 1993. The Sixth International Conference on Domain Decomposition.

[64] G. Rodrigue. Inner / outer iterative methods and numerical Schwarz algorithms. *Parallel Computing*, 2:205–218, 1985.

[65] Y. Saad. Practical Use of Polynomial Preconditionings for the Conjugate Gradient Method. *SIAM Journal on Scientific and Statistical Computing*, 6(4):865–881, 1985.

[66] Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, Boston, 1996.

[67] Y. Saad and M. H. Schulz. GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[68] M. Sauren and H. M. Bücker. On Deriving the Quasi-Minimal Residual Method. Interner Bericht KFA–ZAM–IB–9608, Forschungszentrum Jülich GmbH, Jülich, Germany, April 1996. accepted for publication in *SIAM Review*.

[69] M. Sauren and R. von Seggern. Vorkonditionierung des parallelen Schwarz–Verfahrens zur Lösung parabolischer Differentialgleichungen. Interner Bericht KFA–ZAM–IB–9615, Forschungszentrum Jülich GmbH, Jülich, Germany, June 1996.

[70] H.A. Schwarz. Über einen Grenzübergang durch alternirendes Verfahren. *Vierteljahresschrift der Naturforschenden Gesellschaft in Zürich*, 15:272–286, 1870.

[71] B.F. Smith, P.E. Bjørstad, and W.D. Gropp. *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

[72] K.H. Tan and M.J.A. Borsboom. On Generalized Schwarz Coupling Applied to Advection-Dominated Problems. In Keyes and Xu [50], pages 125–130. Proceedings of the Seventh International Conference on Domain Decomposition.

[73] W.P. Tang. Generalized Schwarz Splittings. *SIAM Journal on Scientific and Statistical Computing*, 13(2):573–595, 1992.

[74] D. R. Taylor. *Analysis of the Look Ahead Lanczos Algorithm for Unsymmetric Matrices*. Ph. D. dissertation, Department of Mathematics, University of California, Berkeley, CA, November 1982.

[75] H. Vereecken, A. Braun, T. Graf, U. Jaekel, G. Lindenmayr, and R. Seidemann. Parallel Computation of Solute Transport in Heterogeneous Porous Media. In W.E. Nagel, editor, *Partielle Differentialgleichungen, Numerik und Anwendungen*, volume 18 of *Konferenzen des Forschungszentrums Jülich*, pages 131–145, Jülich, Deutschland, 1996. Forschungszentrum Jülich GmbH.

[76] H. Vereecken, G. Lindenmayr, O. Neuendorf, U. Döring, and R. Seidemann. TRACE, A mathematical model for reactive transport in 3D variably saturated porous media. Interner Bericht KFA/ICG-4–501494, Forschungszentrum Jülich GmbH, Jülich, Germany, August 1994.

[77] G. A. Watson. *Approximation Theory and Numerical Methods*. John Wiley & Sons, Chichester, 1980.

[78] Q. Ye. A Breakdown–Free Variation of the Nonsymmetric Lanczos Algorithms. *Mathematics of Computation*, 62(205):179–207, 1994.

[79] D.M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, 1971.

[80] Z. Zlatev. *Computational Methods for General Sparse Matrices*. Kluwer Academic Publishers, Dordrecht, 1991.

Dank

Die vorliegende Arbeit ist am Zentralinstitut für Angewandte Mathematik (ZAM) des Forschungszentrums Jülich entstanden.
Herrn Prof. Dr. F. Hoßfeld, dem Institutsdirektor des ZAM und Inhaber des Lehrstuhls für Technische Informatik und Computerwissenschaften der RWTH Aachen, danke ich für die Themenstellung, die Betreuung und für wichtige Impulse, die zum Gelingen der Arbeit beigetragen haben.
Herrn Prof. Dr. K. Indermark, dem Inhaber des Lehrstuhls für Informatik II der RWTH Aachen, gilt mein Dank für die Übernahme des Korreferates.
Besonderer Dank gebührt meinem Betreuer Herrn Dr. R. von Seggern; seine nahezu väterliche Betreuung hat wesentlich zum Gelingen der Arbeit beigetragen.
Herrn Dr. P. Weidner, dem Leiter der Abteilung Mathematik des ZAM, danke ich für zahlreiche Diskussionen und Ratschläge im Verlauf der Arbeit.
Weiter möchte ich meinem Kollegen und Freund Martin Bücker für die zahlreich geführten Diskussionen, die gemeinsamen Problembewältigungen und seine unermüdliche Hilfsbereitschaft während der Erstellung dieser Arbeit danken.
Auch bedanke ich mich bei meiner Kollegin Astrid Goeke, die mich im redaktionellen Teil sehr unterstützt hat.
Mehr als nur ein Dankeschön gilt meiner lieben Tina, die mich mit Ihrer Liebe durch alle Höhen und Tiefen während dieser Zeit geführt hat.