Technische Universität Berlin
Fakultät II - Naturwissenschaften und Mathematik
Institut für Mathematik

Die IDR(s)-Methode zur Lösung von parametrisierten Gleichungssystemen

Diplomarbeit

vorgelegt von
Matthias Miltenberger

Erstgutachter:	Prof. Dr. Volker Mehrmann
Zweitgutachter:	Prof. Dr. Reinhard Nabben

Berlin, den 20.09.2009

Die selbstständige und eigenhändige Anfertigung dieser Arbeit versichere ich an Eides statt.

Berlin, den 20.09.2009

Inhaltsverzeichnis

Abbildungsverzeichnis

Algorithmenverzeichnis

Kapitel 1

Einleitung

Für die Lösung zahlreicher praktischer Probleme ist es notwendig, lineare Gleichungssysteme der Art

$$\mathbf{A}\,\mathbf{x} = \mathbf{b} \text{ mit } \mathbf{A} \in \mathbb{C}^{n,n} \text{ und } \mathbf{b} \in \mathbb{C}^n$$

numerisch zu lösen. Trotz dieser sehr kurzen und einfachen Aufgabenstellung, ist die Lösung solcher Systeme mitunter sehr kompliziert und aufwändig. Vor allem unter dem Gesichtspunkt einer möglichst schnellen und doch präzisen Lösung, stellt diese Problematik große Herausforderungen, sowohl an die verwendeten Algorithmen, als auch an die jeweilige Hardware. So ist es keine Seltenheit, Gleichungssysteme mit über einer Million Unbekannter zu lösen. Mit fortschreitenden Entwicklungen steigen auch die Anforderungen an die Lösungsverfahren, so dass auch auf diesem Gebiet der numerischen Mathematik stets nach neuen Möglichkeiten der effizienten Lösungsberechnung gesucht wird.
Des Weiteren gibt es nur wenige Problemklassen, für deren Lösung ein optimales Verfahren existiert. Es muss vielmehr von Problem zu Problem mithilfe umfangreicher Testläufe herausgefunden werden, welche Methode sich gut eignet und die gewünschten Resultate liefert.

In dieser Arbeit wird eine neue Klasse von iterativen Lösungsverfahren für lineare Gleichungssysteme vorgestellt. Sie basieren auf einer Idee aus dem Jahr 1980. Die sogenannten IDR-Verfahren sind in dieser Form erstmalig im Jahr 2007 präsentiert worden. Aufgrund ihrer neuartigen und ungewöhnlichen Konzeption stellt die Analyse der Verfahren noch immer große Herausforderungen. Trotz allem ist das Prinzip, auf dem die IDR-Methoden basieren, nicht kompliziert, sondern vergleichsweise nachvollziehbar, was die Verfahren attraktiv und interessant macht.
In Kapitel 2 werden grundlegende mathematische Definitionen und Konzepte dargelegt, welche das Verständnis der weiteren Arbeit erleichtern sollen.
Im folgenden Kapitel wird die Idee, die hinter den IDR-Verfahren steckt,

vorgestellt, sowie mögliche Algorithmen, die auf dem IDR-Prinzip basieren, erarbeitet.
Die Lösung von Folgen ähnlicher Gleichungssysteme ist das Thema von Kapitel 4. Hier soll aufgezeigt werden welche Möglichkeiten es gibt mithilfe des sogannten *Recyclings*, Informationen von einem System auf das nächste zu übertragen, um so die Konvergenzgeschwindigkeit zu verbessern. Bei praktischen Anwendungen treten zwar des Öfteren solche Folgen von Systemen auf, jedoch gibt es bislang nur wenige Ansätze, die sich damit beschäftigen wie dieser Umstand effizient zur Lösung der einzelnen Gleichungssysteme ausgenutzt werden kann. Es wird außerdem gezeigt, wie man die IDR-Methoden diesbezüglich modifizieren kann.
Anschließend werden in Kapitel 5 die Ergebnisse der numerischen Experimente präsentiert. Dabei werden die IDR-Methoden sowohl untereinander, als auch mit anderen etablierten Lösungsverfahren verglichen. Dazu werden verschiedene Testsysteme verwendet, um ein breites Problemspektrum abzudecken. Unter Anderem werden realistische Gleichungssysteme benutzt, die aus einem gegenwärtigen Industrieprojekt stammen. Außerdem wird anhand von Testproblemen aus dem vorherigen Kapitel die Effektivität der implementierten Recycling-Technik vorgestellt.
Abschließend werden die Ergebnisse zusammengefasst und ein Ausblick auf weitere Forschungsmöglichkeiten gegeben.

Kapitel 2

Grundlagen

Ein Teilgebiet der Numerischen Mathematik besteht darin, lineare Gleichungssysteme der folgenden Form

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}, \qquad \mathbf{A} \in \mathbb{C}^{n,n},\ \mathbf{b} \in \mathbb{C}^{n} \tag{2.1}$$

zu lösen. Dazu existieren eine Vielzahl von unterschiedlichen Methoden, die sich in Abhängigkeit der Eigenschaften und Beschaffenheit der Matrix $\mathbf{A}$ unterschiedlich gut eignen.
In diesem Kapitel soll ein kurzer Überblick über dieses Gebiet gegeben werden. Zudem werden notwendige Grundlagen und Notationen dargelegt.

2.1 Bezeichnungen und Definitionen

Um die Lesbarkeit dieser Arbeit zu vereinfachen, werden hier einige häufig vorkommende Bezeichnungen und Definitionen aufgelistet:

$\mathbf{A}$:	Matrix des linearen Gleichungssystems, invertierbar
$\mathbf{b}$:	rechte Seite des Gleichungssytems
n:	Dimension des linearen Gleichungssystems ($\mathbf{A} \in \mathbb{C}^{n,n},\ \mathbf{b} \in \mathbb{C}^n$)
$\mathbf{x}_0$:	Startapproximation zur Initialisierung des Verfahrens
$\mathbf{x}_k$:	Approximation an die Lösung im k-ten Schritt
$\mathbf{r}_k$:	Residuum im k-ten Schritt ($\mathbf{r}_k := \mathbf{b} - \mathbf{A}\mathbf{x}_k$)
$\mathbf{A}^*$:	komplex Konjugierte der Transponierten von $\mathbf{A}$
$\mathbf{a} \perp \mathbf{b}$:	$\mathbf{a}$ und $\mathbf{b}$ sind orthogonal zueinander
$\mathbf{P}$:	Matrix mit Schattenvektoren $\mathbf{p}_1, \ldots, \mathbf{p}_s$ als Spalten
$\mathcal{N}(\cdot)$:	linker Nullraum einer Matrix ($\mathbf{x} \in \mathcal{N}(\mathbf{P}) \Rightarrow \mathbf{P}^*\mathbf{x} = 0$)
$\mathcal{S}$:	$(n-s)$-dimensionaler Unterraum von $\mathbb{C}^n$ ($\mathcal{S} = \mathcal{N}(\mathbf{P})$)
$\mathbb{P}^k$:	Raum der Polynome von Grad k oder kleiner
$\lceil\cdot\rceil$:	aufrunden

Außerdem sei angemerkt, dass Matrizen und Vektoren stets fett gedruckt sind, um sie besser von kursiv geschriebenen Skalaren und Indices zu unterscheiden.

Definition 1. Ein Unterraum $\mathcal{U} \subseteq \mathbb{C}^n$ heißt $\mathbf{A}$-*invariant*, wenn für alle Vektoren $\mathbf{v} \in \mathcal{U}$ gilt

$$\mathbf{A}\mathbf{v} \in \mathcal{U}.$$

Definition 2. Gilt für ein Paar (λ, $\mathbf{v}$) mit $\lambda \in \mathbb{C}$ und $\mathbf{v} \in \mathbb{C}^n$ die Beziehung

$$\mathbf{A}\mathbf{v} = \lambda\,\mathbf{v},$$

so heißen λ und $\mathbf{v}$ *Eigenwert*, beziehungsweise *Eigenvektor* der Matrix $\mathbf{A}$.

Definition 3. Die Zahl $\kappa \in \mathbb{R}$ mit $\kappa = \|\mathbf{A}\|\|\mathbf{A}^{-1}\|$ heißt *Konditionszahl* von $\mathbf{A}$. Für die euklidische Norm gilt $\kappa_2 = \frac{\lambda_{\max}}{\lambda_{\min}}$, wobei $\lambda_{\max}$ und $\lambda_{\min}$ den betragsmäßig größten, beziehungsweise kleinsten Eigenwert bezeichnen.

2.2 Verschiedene Lösungsansätze

Es gibt viele verschiedene Möglichkeiten, die Lösung eines linearen Gleichungssystems zu berechnen. Man kann sie in die beiden Klassen der direkten und der iterativen Verfahren einordnen.
Eine der populärsten direkten Methoden stellt dabei das nach Carl Friedrich Gauß benannte Gauß-Verfahren dar. Hier wird die Matrix $\mathbf{A}$ durch schrittweise Umformungen auf eine obere Zeilenstufenform gebracht. Danach wird mittels Rückwärtseinsetzen, das heißt, mit der Bestimmung der Variablen

von unten nach oben, die Lösung berechnet. Es ist also für dieses Verfahren notwendig, auf die gesamte Matrix zugreifen zu können, was nicht bei allen Gleichungssystemen möglich ist. Außerdem ist der Gesamtaufwand zur Lösung eines allgemeinen Gleichungssystems kubisch in der Dimension des Systems, das heißt, er liegt in der Größenordnung von $O(n^3)$. Hat die Systemmatrix dagegen eine spezielle Struktur, kann der Aufwand auch weit darunter liegen und damit das Verfahren auch für größere Systeme eine gute Wahl sein.
In vielen praktischen Anwendungen hat man es mit großen, dünn besetzten Matrizen zu tun, also mit Matrizen, bei denen die Anzahl der Einträge ungleich Null, linear in n ist. Oft ist zudem die Matrix nicht direkt bekannt, sondern liegt lediglich als Matrix-Vektor-Multiplikation vor. Iterative Verfahren benötigen nur diese Funktionalität und sind daher für diese Probleme zu bevorzugen. Aufgrund der Struktur der Matrix sind zudem die Matrix-Vektor-Multiplikationen relativ schnell auszuführen, was die Lösungsberechnung beschleunigt. Iterative Verfahren beginnen mit einer Startapproximation an die Lösung und verbessern diese schrittweise, bis eine festgelegte Fehlertoleranz unterschrittten wird. Allerdings ist die Anzahl der Iterationen bis dieser Fall eintritt, stark von meist schwer zu beeinflussenden Eigenschaften der Matrix $\mathbf{A}$ abhängig. Auch kann oftmals eine Konvergenz nicht garantiert werden, so dass stets problemabhängig das jeweilig bestgeeignetste Verfahren ausgewählt werden muss.
Eine Möglichkeit, die Konvergenz und Stabilität des Verfahrens zu verbessern, besteht in der Verwendung eines Vorkonditionierers. In vielen praktischen Anwendungen dominiert dabei sogar die Berechnung und Anwendung des Vorkonditionierers die Gesamtlaufzeit des Verfahrens.

2.3 Krylovlöser

Zur Lösung des Gleichungssytems (2.1) eignen sich besonders Verfahren, die auf dem Konzept des Krylovraums aufbauen. Dieser Raum ist nach dem russischen Mathematiker und Schiffsbauingenieur Alexei Nikolaevich Krylov (1863-1945) benannt.

Definition 4. Der Raum $\mathcal{K}_k(\mathbf{A}, \mathbf{r}) = \text{span}\left\{\mathbf{r}, \mathbf{Ar}, \ldots, \mathbf{A}^{k-1}\mathbf{r}\right\} \subseteq \mathbb{C}^n$ wird als *Krylovraum* von $\mathbf{A} \in \mathbb{C}^{n,n}$ und $\mathbf{r} \in \mathbb{C}^n$ bezeichnet. Ist eindeutig, welche Vektoren und Matrizen gemeint sind, wird der Raum auch kürzer als $\mathcal{K}_k$ bezeichnet.

Ein Krylovlöser konstruiert schrittweise Annäherungen $\mathbf{x}_k$ an die Lösung, die im k-dimensionalen (affinen) Unterraum $\mathbf{x}_0 + \mathcal{K}_k(\mathbf{A}, \mathbf{r}_0)$ liegen. Die unterschiedlichen Krylovlöser können in vier verschiedene Klassen eingeteilt werden, wobei hier der Einfachheit halber $\mathbf{x}_0 = 0$ gesetzt wird, was zur Folge hat, dass $\mathbf{r}_0 = \mathbf{b}$ gilt:

Ritz-Galerkin-Verfahren:
Man konstruiert ein $\mathbf{x}_k$ mit orthogonal zum aktuellen Krylovraum stehendem Residuum, so dass also

$$\mathbf{r}_k \perp \mathcal{K}_k(\mathbf{A}, \mathbf{r}_0)$$

erfüllt ist. Ein populärer Vertreter dieser Klasse ist das Verfahren der konjugierten Gradienten, kurz CG-Verfahren.

Verfahren der minimalen Residuumsnorm:
Man berechnet dasjenige $\mathbf{x}_k$ mit minimaler Norm $\|\mathbf{r}_k\|_2$, das heißt,

$$\min_{\mathbf{x}_k \in \mathcal{K}_k} \|\mathbf{b} - \mathbf{A}\,\mathbf{x}_k\|_2$$

soll erfüllt sein. Bekannte Methoden dieser Art sind das MINRES-Verfahren und dessen Verallgemeinerung GMRES.

Petrov-Galerkin-Verfahren:
Man konstruiert ein $\mathbf{x}_k$, so dass das Residuum orthogonal zu einem anderen k-dimensionalen Raum steht, also

$$\mathbf{r}_k \perp \mathcal{L}_k.$$

Hierzu zählen unter anderem das BI-CG-Verfahren, eine Verallgemeinerung des CG-Verfahrens oder auch das QMR-Verfahren (quasi minimal residual).

Verfahren der minimalen Fehlernorm:
Man bestimmt ein $\mathbf{x}_k$ aus $\mathbf{A}^*\mathcal{K}_k(\mathbf{A}^*, \mathbf{r}_0)$, so dass $\|\mathbf{x}_k - \mathbf{x}\|_2$ minimal ist. Das SYMMLQ-Verfahren beispielsweise gehört zu dieser Klasse von Methoden.

Natürlich stellen die hier angegebenen Verfahren nur einen kleinen Ausschnitt der Vielzahl an Krylovlösern dar und so sei für weitere Informationen auf [Vor03] verwiesen. Als einen der bekanntesten Vertreter soll hier das GMRES-Verfahren kurz vorgestellt werden, auf das später Bezug genommen wird.

2.3.1 Generalized Minimal Residual

Das GMRES-Verfahren konstruiert explizit eine (orthogonale) Basis des aktuellen Krylovraums. Das bedeutet, dass in jedem Schritt ein neuer Vektor $\mathbf{v}_{k+1}$ zu den vorhandenen hinzugefügt wird, nachdem er zur alten Basis orthogonalisiert wurde. Dadurch entsteht die sogenannte Arnoldi-Relation

$$\mathbf{A}\,\mathbf{V}_k = \mathbf{V}_{k+1}\,\mathbf{H}_{k+1,k}, \tag{2.2}$$

wobei $\mathbf{V}_k = [\ \mathbf{v}_1 \dots \mathbf{v}_k\ ]$ die Basisvektoren aus $\mathcal{K}_k$ enthält und $\mathbf{H}_{k+1,k} \in \mathbb{C}^{k+1,k}$ eine obere Hessenbergmatrix ist, also $\mathbf{H} = [h_{i,j}]$ mit $h_{i,j} = 0$ für $i > j+1$ erfüllt. $\mathbf{v}_1$ wird als $\mathbf{r}_0 = \mathbf{b}$ gewählt. Das GMRES-Verfahren findet im k-ten Schritt die Approximation an die Lösung durch Minimierung der Residuumsnorm:

$$\begin{aligned} \|\mathbf{b} - \mathbf{A}\,\mathbf{x}_k\|_2 &= \|\mathbf{b} - \mathbf{A}\,\mathbf{V}_k\,\mathbf{y}\|_2 \\ &= \|\|\mathbf{r}_0\|_2\,\mathbf{V}_{k+1}\,\mathbf{e}_1 - \mathbf{V}_{k+1}\,\mathbf{H}_{k+1,k}\,\mathbf{y}\|_2 \\ &= \|\|\mathbf{r}_0\|_2\,\mathbf{e}_1 - \mathbf{H}_{k+1,k}\,\mathbf{y}\|_2 \end{aligned}$$

Der letzte Schritt ist möglich, da $\mathbf{V}_{k+1}$ orthogonale Spalten hat und damit von der Norm 1 ist. Des Weiteren bezeichnet $\mathbf{e}_k$ den k-ten Einheitsvektor der passenden Dimension. Für weitere Informationen zu dieser Lösungsmethode sei [Saa03] empfohlen.
Das GMRES-Verfahren stellt eine gute Möglichkeit dar, die Effizienz von anderen Methoden zu untersuchen, da stets die optimale Approximation an die Lösung (bezüglich der Residuumsnorm) aus dem aktuellen Krylovraum gefunden wird. Der große Nachteil dieser Methode ist der Bedarf einer vollständigen Basis des Krylovraums, das heißt, Speicher- und Berechnungsaufwand (Orthogonalisierung) steigen linear an. Eine Abhilfe bei dieser Schwierigkeit wird in Kapitel 4 vorgestellt.

2.3.2 Allgemeine Rekursionen

Mit der Bedingung $\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k$ kann eine ähnliche Beziehung für die Residuen aufgestellt werden:

$$\begin{aligned} \mathbf{x}_1 \in \mathbf{x}_0 + \mathcal{K}_1(\mathbf{A}, \mathbf{r}_0) &\Rightarrow \mathbf{x}_1 = \mathbf{x}_0 + \alpha\,\mathbf{r}_0 \\ \mathbf{x}_2 \in \mathbf{x}_0 + \mathcal{K}_2(\mathbf{A}, \mathbf{r}_0) &\Rightarrow \mathbf{x}_2 = \mathbf{x}_0 + \alpha\,\mathbf{r}_0 + \beta\,\mathbf{A}\mathbf{r}_0 \end{aligned}$$

$$\begin{aligned} \mathbf{r}_1 = \mathbf{b} - \mathbf{A}\mathbf{x}_1 = \mathbf{b} - \mathbf{A}\mathbf{x}_0 - \alpha\,\mathbf{A}\mathbf{r}_0 = \mathbf{r}_0 - \alpha\,\mathbf{A}\mathbf{r}_0 &\Rightarrow \mathbf{r}_1 \in \mathbf{r}_0 + \mathbf{A}\mathcal{K}_1 \\ \mathbf{r}_2 = \mathbf{b} - \mathbf{A}\,\mathbf{x}_2 = \mathbf{r}_0 - \alpha\,\mathbf{A}\,\mathbf{r}_0 - \beta\,\mathbf{A}^2\,\mathbf{r}_0 &\Rightarrow \mathbf{r}_2 \in \mathbf{r}_0 + \mathbf{A}\mathcal{K}_2 \end{aligned}$$

α und β sind komplexe Skalare. Induktiv folgt damit $\mathbf{r}_k \in \mathbf{r}_0 + \mathbf{A}\mathcal{K}_k$. Dies lässt sich auch mit einem Polynom $\Phi_k \in \mathbb{P}^k \backslash \mathbb{P}^{k-1}$ ausdrücken:

$$\mathbf{r}_k = \Phi_k(\mathbf{A})\,\mathbf{r}_0$$

Eine andere Möglichkeit ein Bildungsgesetz für die Residuen zu formulieren, stellt die folgende Schreibweise mit Hilfe von Vorwärtsdifferenzen

$\Delta\,\mathbf{r}_k = \mathbf{r}_{k+1} - \mathbf{r}_k$ dar.

$$\begin{aligned}
\mathbf{r}_{k+1} &= -\alpha\,\mathbf{A}\,\mathbf{r}_k + \sum_{i=0}^{\ell} \beta_i\,\mathbf{r}_{k-i} \\
&= \gamma_0\,\mathbf{r}_k - \alpha\,\mathbf{A}\,\mathbf{r}_k - \sum_{i=1}^{\ell} \gamma_i \Delta\,\mathbf{r}_{k-i}\,, \\
\text{wobei} \qquad \gamma_\ell &= \beta_\ell\,, \\
\gamma_{\ell-1} &= \beta_{\ell-1} + \beta_\ell\,, \\
&\vdots \\
\gamma_0 &= \beta_0 + \ldots + \beta_\ell
\end{aligned}$$

Um nun mithilfe der Relation $\Delta\,\mathbf{r}_k = -\,\mathbf{A}\Delta\,\mathbf{x}_k$ und der Differenzen $\Delta\,\mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ eine Berechnungsformel für die dazugehörigen Approximationen zu erhalten, muss $\gamma_0 = \sum_{i=0}^{\ell} \beta_i = 1$ erfüllt sein:

$$\begin{aligned}
\mathbf{r}_{k+1} &= \mathbf{r}_k - \alpha\,\mathbf{A}\,\mathbf{r}_k - \sum_{i=1}^{\ell} \gamma_i \Delta\,\mathbf{r}_{k-i} \\
\Delta\,\mathbf{r}_k &= -\alpha\,\mathbf{A}\mathbf{r}_k - \sum_{i=1}^{\ell} \gamma_i \Delta\,\mathbf{r}_{k-i} \\
\Delta\,\mathbf{x}_k &= \alpha\,\mathbf{r}_k - \sum_{i=1}^{\ell} \gamma_i \Delta\,\mathbf{x}_{k-i} \\
\mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha\,\mathbf{r}_k - \sum_{i=1}^{\ell} \gamma_i \Delta\,\mathbf{x}_{k-i}\,,
\end{aligned}$$

Der Parameter ℓ bestimmt hierbei die Tiefe der Rekursion, das heißt, ℓ bestimmt maßgeblich den Speicherbedarf und den Aufwand in jedem Iterationsschritt. Befindet sich ℓ in der Größenordnung von k, so spricht man von einer langen Rekursion ($\rightarrow$ GMRES). Bei konstantem $\ell \ll k$ handelt es sich um ein Verfahren mit kurzen Rekursionen ($\rightarrow$ CG).
In [FabM84] wurde gezeigt, dass es kein Verfahren für allgemeine Matrizen gibt, das sowohl kurze Rekursionen als auch eine optimale Fehlerminimierungseigenschaft besitzt (Faber-Manteuffel-Theorem). Es müssen also stets Vor- und Nachteile der verschiedenen Methoden gegeneinander abgewogen werden. Mit den IDR-Methoden aus [SonG08] und [GijS08] haben sich die Auswahlmöglichkeiten weiter vervielfältigt.
In Kapitel 3 wird erklärt auf welche Art und Weise bei diesen Verfahren die Parameter α und γ_i bestimmt werden, um effizient die Lösung eines Gleichungssystems zu berechnen.

Kapitel 3

Induzierte Dimensionsreduktion (IDR)

Die Methoden der IDR-Familie basieren ursprünglich auf einer Idee von Peter Sonneveld aus dem Jahr 1980. Im folgenden Abschnitt wird diese Idee formuliert und erklärt, wie man daraus einen funktionierenden Algorithmus konstruieren kann. Anschließend werden zwei Varianten aus der Klasse der IDR-Methoden näher vorgestellt.
Keine der hier behandelten IDR-Methoden stellt Bedingungen, wie Symmetrie oder positive Definitheit, an die Matrix $\mathbf{A}$. Auch dies ist ein großer Vorteil dieser Verfahren.

3.1 Das IDR-Prinzip

Die Idee der IDR-Verfahren besteht darin Approximationen an die Lösung so zu berechnen, dass die Residuen in speziellen Räumen mit kleiner werdenden Dimensionen liegen. Das bedeutet, dass die exakte Lösung gefunden ist, sobald der Raum, in dem das Residuum liegt, zum Nullraum zusammengefallen ist. Es wird also nicht wie bei einigen anderen Verfahren schrittweise ein Raum aufgebaut, in dem die optimale Lösung gesucht wird. Die ungewöhnliche Konstruktion der Räume grenzt dabei die IDR-Verfahren von herkömmlichen Lösungsverfahren ab.
Die Art und Weise wie man bei der Konstruktion der Residuen, beziehungsweise der Konstruktion der Räume, vorgeht, ist das Hauptunterscheidungsmerkmal der verschiedenen Methoden der IDR-Familie.

3.1.1 Das IDR-Theorem

Der folgende Satz beschreibt die Konstruktion der Räume, in denen die Residuen liegen werden und ist in dieser oder ähnlicher Form auch in [SonG08], [GijS08] und [SleSG08] zu finden. Der Beweis ist [SonG08] entnommen.

Satz 1. *Sei $\mathcal{G}_0$ der volle Krylovraum $\mathcal{K}_n(\mathbf{A}, \mathbf{r})$ und sei $\mathcal{S} \subset \mathbb{C}^n$ ein echter Unterraum von $\mathbb{C}^n$, so dass $\mathcal{G}_0 \cap \mathcal{S}$ keinen nichttrivialen Unterraum von $\mathbf{A}$ enthält.*
Definiert man für $j = 1, 2, \ldots$ die Folge der Räume $\{\mathcal{G}_j\} \subseteq \mathbb{C}^n$ als

$$\mathcal{G}_j := (\mathbf{I} - \omega_j \mathbf{A})\,(\mathcal{G}_{j-1} \cap \mathcal{S})$$

mit komplexen Skalaren $\omega_j \neq 0$, so gilt

- *$\mathcal{G}_{j+1} \subseteq \mathcal{G}_j$ für alle $j \in \mathbb{N}$ und*
- *$\mathcal{G}_J = \{0\}$ für ein $J \leq n$.*

Beweis. Durch Induktion lässt sich die Behauptung $\mathcal{G}_{j+1} \subseteq \mathcal{G}_j$ zeigen. Da $\mathcal{G}_0$ als voller Krylovraum $\mathbf{A}$-invariant ist, gilt die Beziehung

$$\mathcal{G}_1 = (\mathbf{I} - \omega_1 \mathbf{A})(\mathcal{G}_0 \cap \mathcal{S}) \subseteq (\mathbf{I} - \omega_1 \mathbf{A})\mathcal{G}_0 \subseteq \mathcal{G}_0.$$

Sei nun die Induktionsannahme $\mathcal{G}_j \subseteq \mathcal{G}_{j-1}$ für $j > 0$ erfüllt. Weiter sei $\mathbf{x}$ ein Vektor aus dem Raum $\mathcal{G}_{j+1}$. Es existiert also ein $\mathbf{v} \in \mathcal{G}_j \cap \mathcal{S}$ mit

$$\mathbf{x} = (\mathbf{I} - \omega_{j+1} \mathbf{A})\,\mathbf{v}.$$

Mit der Induktionsannahme ist $\mathbf{v}$ auch ein Element des Raums $\mathcal{G}_{j-1} \cap \mathcal{S}$ und damit liegt $(\mathbf{I} - \omega_j \mathbf{A})\,\mathbf{v}$ auch im Raum $\mathcal{G}_j$. Also muss $\mathbf{A}\,\mathbf{v}$ in $\mathcal{G}_j$ liegen und damit auch $\mathbf{x}$, was sich als Linearkombination von $\mathbf{v}$ und $\mathbf{A}\,\mathbf{v}$ darstellen lässt:

$$\mathbf{x} = (\mathbf{I} - \omega_{j+1} \mathbf{A})\,\mathbf{y} = \mathbf{y} - \omega_{j+1} \mathbf{A}\,\mathbf{y} \in \mathcal{G}_j$$

Somit ist die erste Behauptung bewiesen.
Als nächstes soll gezeigt werden, dass für ein $J \leq n$ der Raum $\mathcal{G}_J$ gleich dem Nullraum ist.
Da $\mathcal{G}_{j+1} \subseteq \mathcal{G}_j$ ist, so gibt es zwei Möglichkeiten: Entweder $\mathcal{G}_{j+1}$ ist gleich dem Raum $\mathcal{G}_j$ oder $\mathcal{G}_{j+1}$ ist ein echter Unterraum von $\mathcal{G}_j$.
Gleichheit kann nur auftreten, wenn $\mathcal{G}_j \cap \mathcal{S} = \mathcal{G}_j$, denn $\dim(\mathcal{G}_j \cap \mathcal{S}) < \dim(\mathcal{G}_j)$ würde $\dim(\mathcal{G}_{j+1}) < \dim(\mathcal{G}_j)$ implizieren, was der Gleichheit der Räume widerspräche. Also gilt $\mathcal{G}_j \cap \mathcal{S} = \mathcal{G}_j$, womit die $\mathbf{A}$-Invarianz von $\mathcal{G}_j$ folgt:

$$\mathcal{G}_j = \mathcal{G}_{j+1} = (\mathbf{I} - \omega_{j+1}\,\mathbf{A})(\mathcal{G}_j \cap \mathcal{S}) = (\mathbf{I} - \omega_{j+1}\,\mathbf{A})\mathcal{G}_j$$

Außerdem gilt $\mathcal{G}_j \subseteq \mathcal{S}$ und mit $\mathcal{G}_j \subseteq \mathcal{G}_0$ folgt, dass $\mathcal{G}_j \subseteq \mathcal{G}_0 \cap \mathcal{S}$. Da aber nach Voraussetzung des Satzes $\mathcal{G}_0 \cap \mathcal{S}$ keinen nichttrivialen $\mathbf{A}$-invarianten Unterraum enthält, muss $\mathcal{G}_j$ bereits der Nullraum sein.
Zusammenfassend lässt sich also sagen, dass in jedem Schritt von j zu $j+1$ entweder die Dimension von $\mathcal{G}_j$ reduziert wird oder bereits $\mathcal{G}_j = \{0\}$ erfüllt ist. Da die Dimension von $\mathcal{G}_0$ höchstens n beträgt, gilt also $J \leq n$. □

Bemerkung 1. Die Räume $\mathcal{G}_j$ werden auch als *Sonnveldräume* bezeichnet. Meist wird $\mathcal{S}$ als linker Nullraum einer Matrix $\mathbf{P} = [\,\mathbf{p}_1 \dots \mathbf{p}_s] \in \mathbb{C}^{n,s}$ definiert. Es gilt damit für alle Vektoren $\mathbf{v} \in \mathcal{S} = \mathcal{N}(\mathbf{P})$ die Bedingung $\mathbf{P}^*\,\mathbf{v} = 0$. Die Wahl des Parameters s wirkt sich dabei maßgeblich auf die Konvergenzgeschwindigkeit und den Speicherbedarf des Algorithmus aus.

Der nächste Satz wird in [SonG08] als „erweitertes IDR-Theorem“ bezeichnet und geht näher auf die tatsächliche Dimensionsreduktion von einem Raum zum nächsten ein.

Satz 2. *Es gelten die gleichen Bezeichnungen wie bei Satz 1, also* $\mathcal{G}_0 = \mathcal{K}_n(\mathbf{A}, \mathbf{r})$ *und* $\mathcal{G}_{j+1} = (\mathbf{I} - \omega_{j+1}\,\mathbf{A})(\mathcal{G}_j \cap \mathcal{S})$, *wobei* $(\mathbf{I} - \omega_{j+1}\,\mathbf{A})$ *nichtsingulär sei. Die Dimension der Sonneveldräume* $\mathcal{G}_j$ *sei mit* d_j *bezeichnet.*
Dann gilt für alle $j \geq 0$

$$d_j \geq d_{j+1} \quad \textit{und} \quad 0 \leq d_{j+1} - d_{j+2} \leq d_j - d_{j+1} \leq s.$$

Das heißt, die Dimensionsreduktion liegt stets zwischen 0 *und* s.

Beweis. Die erste Behauptung ist bereits im Beweis zu Satz 1 mit $\mathcal{G}_{j+1} \subseteq \mathcal{G}_j$ gezeigt worden. Für die zweite Behauptung wird der Raum $\mathcal{G}_j \cap \mathcal{S}$ umgeschrieben. Dazu sei $\mathbf{G}_j$ eine Matrix, deren Spalten eine Basis von $\mathcal{G}_j$ bilden. Damit gilt für alle $\mathbf{v} \in \mathcal{G}_j \cap \mathcal{S}$

$$\mathbf{v} = \mathbf{G}_j\,\mathbf{c} \quad \text{und} \quad \mathbf{P}^*\,\mathbf{v} = \mathbf{P}^*\,\mathbf{G}_j\,\mathbf{c} = 0 \;\text{ für }\; \mathbf{c} \in \mathbb{C}^n$$

Folglich lässt sich $\mathcal{G}_j \cap \mathcal{S}$ auch umformulieren zu $\mathbf{G}_j(\mathcal{N}(\mathbf{P}^*\,\mathbf{G}_j))$. Also gilt für $\mathcal{G}_{j+1}$

$$\mathcal{G}_{j+1} = (\mathbf{I} - \omega_{j+1}\,\mathbf{A})(\mathcal{G}_j \cap \mathcal{S}) = (\mathbf{I} - \omega_{j+1}\,\mathbf{A})\,\mathbf{G}_j(\mathcal{N}(\mathbf{P}^*\,\mathbf{G}_j))$$

und da $\mathbf{I} - \omega_{j+1}\,\mathbf{A}$ nichtsingulär ist, gilt für die Dimension d_{j+1}

$$d_{j+1} = \dim(\mathcal{G}_{j+1}) = \dim(\,\mathbf{G}_j(\mathcal{N}(\,\mathbf{P}^*\,\mathbf{G}_j))) = d_j - \mathrm{rang}(\,\mathbf{P}^*\,\mathbf{G}_j).$$

$\mathbf{P}^*\,\mathbf{G}_j \in \mathbb{C}^{s,d_j}$ hat $\mathrm{rang}(\,\mathbf{P}^*\,\mathbf{G}_j) = s - \dim(\mathcal{N}(\,\mathbf{G}_j^*\,\mathbf{P})) = s - l_j$, $l_j \leq s$ also gilt

$$d_{j+1} = d_j - (s - l_j) \;\Rightarrow\; d_j - d_{j+1} \leq s.$$

Es bleibt noch die mittlere Ungleichung der Behauptung zu zeigen. Dazu sei $\mathbf{v} \neq 0$ ein beliebiger Vektor aus $\mathcal{N}(\,\mathbf{G}_j^*\,\mathbf{P})$. Es gilt also $\mathbf{G}_j^*\,\mathbf{P}\mathbf{v} = 0$ und da $\mathcal{G}_{j+1} \subseteq \mathcal{G}_j$ auch $\mathbf{G}_{j+1}^*\,\mathbf{P}\mathbf{v} = 0$, beziehungsweise $\mathbf{v} \in \mathcal{N}(\,\mathbf{G}_{j+1}^*\,\mathbf{P})$. Also ist $\mathcal{N}(\,\mathbf{G}_{j+1}^*\,\mathbf{P}) \subseteq \mathcal{N}(\,\mathbf{G}_j^*\,\mathbf{P})$ und damit ist auch $l_{j+1} = \dim(\mathcal{N}(\,\mathbf{G}_{j+1}^*\,\mathbf{P})) \leq \dim(\mathcal{N}(\,\mathbf{G}_j^*\,\mathbf{P})) = l_j$. Es gilt wie oben

$$d_{j+2} = d_{j+1} - (s - l_{j+1}) \;\Rightarrow\; d_{j+1} - d_{j+2} \leq s$$

und so folgt

$$d_{j+1} - d_{j+2} \leq d_j - d_{j+1} \leq s. \qquad \square$$

Wie schon in [SonG08] erläutert, lässt sich leicht begründen, dass im Normalfall eine Dimensionsreduktion der Größe s stattfindet: Im Beweis ist zu sehen, dass im Schritt von $\mathcal{G}_j$ nach $\mathcal{G}_{j+1}$ die Reduktion dem Rang der Matrix $\mathbf{P}^*\,\mathbf{G}_j \in \mathbb{C}^{s,d_j}$ entspricht. Angenommen es ist $d_j \gg s$, das heißt, das Verfahren befindet sich noch nicht in der Endphase, so ist es sehr unwahrscheinlich, dass der Rang von $\mathbf{P}^*\,\mathbf{G}_j$ kleiner als s ist: $\mathbf{P}$ und $\mathbf{G}_j$ haben jeweils linear unabhängige Spaltenvektoren, so dass ein verminderter Rang nur auftreten kann, wenn $\mathbf{x}^*\,\mathbf{G}_j = 0$ für ein $\mathbf{x} \in \mathbb{C}^n \backslash \{0\}$ gilt. Da jedoch $\mathbf{x} = \mathbf{P}\mathbf{y}$ eine Linearkombination der Zeilen von $\mathbf{P}$ ist, muss $\mathbf{y}$ als Vektor aus $\mathbb{C}^s \backslash \{0\}$ d_j Bedingungen genügen, um $\mathbf{y}^*\,\mathbf{P}^*\,\mathbf{G}_j = 0$ zu erfüllen. Um ein realistisches Zahlenbeispiel zu nennnen, sei ein Problem der Größe $n = 5000$ mit der Wahl $s = 10$ zu lösen. Aller Wahrscheinlichkeit nach liegt hier die schrittweise Dimensionsreduktion bei 10, sofern nicht die letzten Iterationen betrachtet werden.
Zwar ist dies kein Beweis der Konvergenz der IDR-Methode, aber immerhin lässt sich so bei Verwendung exakter Arithmetik die Anzahl der benötigten Reduktionsschritte, um von einem n-dimensionalen Raum auf den Nullraum zu gelangen, mit $\lceil n/s \rceil$ angeben.

3.1.2 Alternative Definition

Die Sonneveldräume $\mathcal{G}_j$ können auch mithilfe von Block-Krylovräumen beschrieben werden:

$$\mathcal{G}_j = \{\,\mathbf{x} \in \mathbb{C}^n \;:\; \mathbf{x} = \Omega_j(\,\mathbf{A})\,\mathbf{v},\, \mathbf{v} \perp \mathcal{K}_j\,(\,\mathbf{A}^*,\,\mathbf{P})\}$$

Hierbei bezeichnet Ω_j ein Polynom j-ten Grades der Form

$$\Omega_j(t) = (1 - \omega_1) \cdot \ldots \cdot (1 - \omega_j).$$

Der Block-Krylovraum ist definiert durch

$$\mathcal{K}_j\,(\,\mathbf{A}^*,\,\mathbf{P}) := \left\{\,\mathbf{z} \in \mathbb{C}^n \;:\; \mathbf{z} = \sum_{i<j} (\,\mathbf{A}^*)^i\;\mathbf{P}\,\mathbf{v},\,\mathbf{v} \in \mathbb{C}^s\right\},$$

stimmt also für einen Vektor $\mathbf{P}$ ($s = 1$) anstelle einer Matrix ($s > 1$) mit dem gewöhnlichen Krylovraum überein.
Mit dieser alternativen Definition der Räume $\mathcal{G}_j$ lassen sich die IDR-Verfahren in die Klasse der Petrov-Galerkin-Verfahren einordnen.
Ist nun das Ziel, den Raum $\mathcal{G}_j$ mit wachsendem j klein werden zu lassen, so muss dazu der Block-Krylovraum $\mathcal{K}_j(\,\mathbf{A}^*,\,\mathbf{P})$ entsprechend größer werden.
Das Residuum bei den IDR-Verfahren hat also die Eigenschaft im Bild des orthogonalen Komplements eines Block-Krylovraums zu stehen:

$$\mathbf{r}_k \in \Omega_j(\,\mathbf{A})\,(\mathcal{K}_j(\,\mathbf{A}^*,\,\mathbf{P}))^{\perp}\,.$$

Dagegen ist beim GMRES-Verfahren das Residuum orthogonal zum Raum $\mathbf{A}\mathcal{K}_k$ und beim BI-CG-Verfahren steht es orthogonal zum Krylovraum $\mathcal{K}_k(\,\mathbf{A}^*,\,\tilde{\mathbf{r}}_0)$, wobei hier $\tilde{\mathbf{r}}_0$ eine vergleichbare Rolle, wie die Schattenvektoren der IDR-Verfahren hat.
Für den Beweis dieses Zusammenhangs der Sonneveldräume mit Block-Krylovräumen sei auf Abschnitt 3.2.3, beziehungsweise [Gut09] verwiesen.

3.2 Konstruktion eines Algorithmus

In den folgenden Abschnitten werden verschiedene Möglichkeiten gezeigt, wie ein auf dem IDR-Prinzip basierender Algorithmus, konstruiert werden kann und welche Zusammenhänge zwischen den einzelnen Varianten bestehen. Die aufgeführten Methoden sind aus den jeweils angegebenen Quellen übernommen und wurden jeweils nur leicht in der Notation an diese Arbeit angepasst, um Vergleiche zwischen den Methoden zu erleichtern und ein besseres Verständnis zu ermöglichen. In dieser Arbeit wird, sofern nicht anders angegeben, der aktuell betrachtete Sonneveldraum als $\mathcal{G}_j$ bezeichnet, während die einzelnen Vektoren fortlaufend mit dem Index k gekennzeichnet sind und so die aktuelle Iteration widerspiegeln.

3.2.1 Die einfache IDR(s)-Methode

Zunächst wird eine vergleichsweise intuitive Realisierung vorgestellt, die in [SonG08] beschrieben wurde. Wie bereits im vorherigen Kapitel erläutert

wurde, verwendet ein allgemeiner Krylovlöser die Iterationsvorschriften

$$\begin{aligned} \mathbf{r}_{k+1} &= \mathbf{r}_k - \alpha\,\mathbf{A}\,\mathbf{r}_k - \sum_{i=1}^{s} \gamma_i \Delta\,\mathbf{r}_{k-i} \text{ und} \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha\,\mathbf{r}_k - \sum_{i=1}^{s} \gamma_i \Delta\,\mathbf{x}_{k-i}. \end{aligned}$$

Um nun zu erreichen, dass $\mathbf{r}_{k+1}$ im neuen Raum $\mathcal{G}_{j+1}$ liegt, müssen $\mathbf{v}_k$ und γ_i für $i = 1, \ldots, s$ so bestimmt werden, dass die IDR-Bedingung

$$\mathbf{r}_{k+1} = (\mathbf{I} - \omega_{j+1}\,\mathbf{A})\,\mathbf{v}_k, \qquad \mathbf{v}_k \in \mathcal{G}_j \cap \mathcal{S} \tag{3.1}$$

erfüllt ist.
Unter der Annahme, dass alle vorherigen Residuen im Raum $\mathcal{G}_j$ liegen, gilt das gleiche für die Linearkombinationen $\Delta\,\mathbf{r}_{k-i}$. Eine gute Wahl für $\mathbf{v}_k$ ist also eine Kombination aus ebendiesen Vektoren, da somit $\mathbf{v}_k \in \mathcal{G}_j$ erfüllt wäre. Mit

$$\mathbf{v}_k = \mathbf{r}_k - \sum_{i=1}^{s} \gamma_i \Delta\,\mathbf{r}_{k-i} \tag{3.2}$$

entspricht die IDR-Bedingung einer Verallgemeinerung der obigen allgemeinen Krylov-Iterationvorschrift

$$\mathbf{r}_{k+1} = (\mathbf{I} - \omega_{j+1}\,\mathbf{A})\,\mathbf{v}_k = \mathbf{r}_k - \omega_{j+1}\,\mathbf{A}\,\mathbf{v}_k - \sum_{i=1}^{s} \gamma_i \Delta\,\mathbf{r}_{k-i}.$$

Es findet also zusätzlich zur Multiplikation von $\mathbf{A}$ mit $\mathbf{r}_k$ auch eine Multiplikation mit den s vorherigen Residuen $\mathbf{r}_{k-s}, \ldots, \mathbf{r}_{k-1}$ statt. Dies ist auch ein Grund für das Fehlen einer umfassenden Analyse der Methode für $s > 1$. Eine Analyse auf Grundlage der Störungstheorie, das heißt, unter Verwendung einer gestörten Matrix $\tilde{\mathbf{A}}$ wird somit schnell sehr kompliziert und war bislang noch nicht erfolgreich.
Der Parameter ω_{j+1} kann im Übrigen frei gewählt werden, vorzugsweise so dass $\mathbf{r}_{k+1}$ minimiert wird.
Allerdings muss nun noch gesichert werden, dass $\mathbf{v}_k$ auch in $\mathcal{S}$ liegt. Um dies zu erreichen, werden die verbleibenden Freiheitsgrade in Form der γ_{k-i} ausgeschöpft. Mithilfe der Matrix $\mathbf{P}$, dessen Nullraum $\mathcal{S}$ definiert, kann die Beziehung $\mathbf{v}_k \in \mathcal{S}$ als $\mathbf{P}^*\,\mathbf{v}_k = 0$ geschrieben werden. Durch Einsetzen der Darstellung (3.2) von $\mathbf{v}_k$ lässt sich diese Gleichung zu

$$\mathbf{P}^*\,\mathbf{r}_k = \mathbf{P}^* \sum_{i=1}^{s} \gamma_i \Delta\,\mathbf{r}_{k-i}$$

umformen. Dies kann übersichtlicher in Matrizenschreibweise dargestellt werden, so dass sich ein lineares Gleichungssystem der Größe $s \times s$ ergibt:

$$\left(\begin{bmatrix} - & \mathbf{p}_1^* & - \\ - & \mathbf{p}_2^* & - \\ & \vdots & \\ - & \mathbf{p}_s^* & - \end{bmatrix} \begin{bmatrix} | & | & | & | \\ \Delta\,\mathbf{r}_{k-1} & \Delta\,\mathbf{r}_{k-2} & \dots & \Delta\,\mathbf{r}_{k-s} \\ | & | & | & | \end{bmatrix} \right) \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_s \end{bmatrix} = \mathbf{P}^*\,\mathbf{r}_k.$$

Mit der IDR-Abbildung (3.1) kann aus diesem $\mathbf{v}_k$ ein neues Residuum $\mathbf{r}_{k+1}$ berechnet werden. Dieses Residuum $\mathbf{r}_{k+1}$ ist das erste Residuum im neuen Raum $\mathcal{G}_{j+1}$ und heißt *Primärresiduum*. Um ein weiteres Primärresiduum im Raum $\mathcal{G}_{j+2} \subset \mathcal{G}_{j+1}$ zu finden, müssen zunächst sogenannte *Zwischenresiduen* berechnet werden, damit genug Vektoren in $\mathcal{G}_{j+1}$ vorhanden sind, um diesen Sonneveldraum mit $\mathcal{S}$ zu schneiden. So entsteht ein neues $\mathbf{v}_{k+s+1} \in \mathcal{G}_{j+1} \cap \mathcal{S}$, mit welchem der IDR-Schritt $(\mathbf{I} - \omega_{j+2}\,\mathbf{A})\,\mathbf{v}$ ausgeführt werden kann.
Wie in dem obigen $(s \times s)$-Gleichungssystem zu erkennen ist, müssen $s+1$ Vektoren in $\mathcal{G}_{j+1}$ aufgebaut werden. Das bedeutet, dass zum Primärresiduum zusätzlich s Zwischenresiduen benötigt werden. Die Zwischenresiduen werden auf die gleiche Art wie das Primärresiduum erzeugt. Für die IDR-Abbildung wird das gleiche ω_{j+1} verwendet, während bei der Konstruktion von $\mathbf{v}_{k+i} \in \mathcal{G}_{j+1} \cap \mathcal{S}$, $i = 1, \dots, s+1$ die „ältesten“ Δ-Vektoren aus $\mathcal{G}_j$ durch die neu berechneten Δ-Vektoren aus $\mathcal{G}_{j+1}$ ersetzt werden. Dies ist aufgrund der Inklusionseigenschaft der Sonneveldräume möglich. Durch dieses Vorgehen werden nur die aktuellsten Vektoren verwendet, so dass zusätzlich zur aktuellen Approximation nebst Residuum, stets nur $s\ \Delta\,\mathbf{r}$- und $s\ \Delta\,\mathbf{x}$-Vektoren gespeichert werden müssen.
Um den Algorithmus in Gang zu setzen, bzw. das erste Primärresiduum im Raum $\mathcal{G}_1$ zu konstruieren, werden die benötigten s Vektoren $\Delta\,\mathbf{r}_0$, …, $\Delta\,\mathbf{r}_{s-1} \in \mathcal{G}_0 = \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$ durch ein frei wählbares Krylovraum-Verfahren aufgebaut.
Unter der Annahme, dass die Dimensionsreduktion konstant bei s liegt und n durch s teilbar ist, kann eine einfache Aufwandsabschätzung für dieses Verfahren angegeben werden.

for $j = 0$ to $n/s - 1$ **do**
  baue Raum $\mathcal{G}_j$ auf: → s Iterationen
  führe IDR-Schritt durch: → eine Iteration
end for

Es werden demnach $(s+1) \cdot n/s = n + n/s$ Iterationen für die Berechnung der Lösung in exakter Arithmetik benötigt.
Der folgende Pseudocode, angelehnt an [SonG08], beschreibt dieses Vorgehen detaillierter, während die nachstehenden Grafiken das IDR-Verfahren für den dreidimensionalen Fall illustrieren.

Input: $\mathbf{A},\ \mathbf{b},\ s,\ \mathbf{x}_0,\ \mathbf{P}$, tol
Output: $\mathbf{x}$, sodass $\|\mathbf{b} - \mathbf{A}\,\mathbf{x}\| < \text{tol}$

1: $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\,\mathbf{x}_0$
2: **for** $k = 1$ to s **do**
3: $\quad \mathbf{v} = \mathbf{A}\,\mathbf{r}_{k-1}$
4: $\quad \omega = \frac{\langle \mathbf{r}_{k-1}, \mathbf{v}\rangle}{\|\mathbf{v}\|^2}$
5: $\quad \Delta\,\mathbf{x}_{k-1} = \omega\,\mathbf{r}_{k-1}$
6: $\quad \Delta\,\mathbf{r}_{k-1} = -\omega\,\mathbf{v}$
7: $\quad \mathbf{r}_k = \mathbf{r}_{k-1} + \Delta\,\mathbf{r}_{k-1}$
8: $\quad \mathbf{x}_k = \mathbf{x}_{k-1} + \Delta\,\mathbf{x}_{k-1}$
9: **end for** /* *genug Vektoren in* $\mathcal{G}_0$ *berechnet* */
10: $\Delta\,\mathbf{R}_k = [\Delta\,\mathbf{r}_{k-1} \ldots \Delta\,\mathbf{r}_0]$
11: $\Delta\,\mathbf{X}_k = [\Delta\,\mathbf{x}_{k-1} \ldots \Delta\,\mathbf{x}_0]$
12: **while** $\|\mathbf{r}_k\| > \text{tol}$ **do**
13: $\quad$ **for** $i = 0$ to s **do**
14: $\quad\quad$ löse nach $\mathbf{y}$ auf: $\mathbf{P}^*\Delta\,\mathbf{R}_k\,\mathbf{y} = \mathbf{P}^*\,\mathbf{r}_k$ /* $\mathbf{y} = [\gamma_1 \ldots \gamma_s]^T$ */
15: $\quad\quad \mathbf{v} = \mathbf{r}_k - \Delta\,\mathbf{R}_k\,\mathbf{y}$
16: $\quad\quad$ **if** $k = 0$ **then**
17: $\quad\quad\quad \omega = \frac{\langle \mathbf{v}, \mathbf{A}\,\mathbf{v}\rangle}{\|\mathbf{A}\,\mathbf{v}\|^2}$ /* ω *für neuen Raum* $\mathcal{G}_{j+1}$ */
18: $\quad\quad\quad \Delta\,\mathbf{r}_k = -\omega\,\mathbf{A}\,\mathbf{v} - \Delta\,\mathbf{R}_k\,\mathbf{y}$ /* $= (\mathbf{I} - \omega\,\mathbf{A})\,\mathbf{v} - \mathbf{r}_k$ */
19: $\quad\quad\quad \Delta\,\mathbf{x}_k = \omega\,\mathbf{v} - \Delta\,\mathbf{X}_k\,\mathbf{y}$ /* $\Delta\,\mathbf{x}_k = -\mathbf{A}^{-1}\Delta\,\mathbf{r}_k$ */
20: $\quad\quad$ **else**
21: $\quad\quad\quad$ /* *berechne weitere Vektoren in* $\mathcal{G}_j$ */
22: $\quad\quad\quad \Delta\,\mathbf{x}_k = \omega\,\mathbf{v} - \Delta\,\mathbf{X}_k\,\mathbf{y}$
23: $\quad\quad\quad \Delta\,\mathbf{r}_k = -\mathbf{A}\Delta\,\mathbf{x}_k$
24: $\quad\quad$ **end if**
25: $\quad\quad \mathbf{r}_{k+1} = \Delta\,\mathbf{r}_k + \mathbf{r}_k$ /* $= (\mathbf{I} - \omega\,\mathbf{A})\,\mathbf{v}$ */
26: $\quad\quad \mathbf{x}_{k+1} = \Delta\,\mathbf{x}_k + \mathbf{x}_k$
27: $\quad\quad \Delta\,\mathbf{r}_k = (\mathbf{I} - \omega\,\mathbf{A})\,\mathbf{v} - \mathbf{r}_k = -\Delta\,\mathbf{R}_k\,\mathbf{y} - \omega\,\mathbf{A}\,\mathbf{v}$
28: $\quad\quad \Delta\,\mathbf{x}_k = -\Delta\,\mathbf{X}_k\,\mathbf{y} + \omega\,\mathbf{v}$
29: $\quad$ **end for**
30: $\quad$ /* *Aktualisierung der* Δ*-Matrizen:* */
31: $\quad \Delta\,\mathbf{R}_{k+1} = [\Delta\,\mathbf{r}_k \ldots \Delta\,\mathbf{r}_{k-s+1}]$
32: $\quad \Delta\,\mathbf{X}_{k+1} = [\Delta\,\mathbf{x}_k \ldots \Delta\,\mathbf{x}_{k-s+1}]$
33: $\quad k = k + 1$
34: **end while**

Algorithmus 3.1: einfaches IDR(s)

Wie man an den Zeilen 18/19 und 21/22 sehen kann, unterscheiden sich die Berechnungen der Primärresiduen von denen der Zwischenresiduen nur durch die Wahl eines neuen Wertes für ω. Die Umformulierung der Berechnung soll (bei Gleitkommaarithmetik) die Beziehung $\Delta\,\mathbf{r}_k = -\mathbf{A}\Delta\,\mathbf{x}_k$ sichern und einem sogenannten *residual gap*, also einer zu großen Differenz

zwischen dem tatsächlichen und dem berechneten Residuum, vorbeugen. Die Aktualisierung der Δ-Matrizen bewirkt, dass im Falle der Zwischenresiduen im nächsten Schritt ein neues Residuum konstruiert wird, das allerdings noch im selben Raum $\mathcal{G}_j$ wie das vorhergehende Residuum liegt. Erst durch eine neue Wahl von ω erfolgt der Übergang zum neuen Raum $\mathcal{G}_{j+1}$.
Die folgenden Grafiken veranschaulichen diesen Zusammenhang visuell für den Fall $\mathcal{G}_0 = \mathbb{R}^3$ und $s = 1$.
In Abbildung 3.1 aus [Gut09] sind die verschiedenen Räume $\mathcal{G}_j$ und die dazugehörigen Abbildungen $(\mathbf{I} - \omega_j \mathbf{A})$ dargestellt. Gut zu sehen ist hier die Inklusionseigenschaft $\mathcal{G}_{j+1} \subset \mathcal{G}_j$ der einzelnen Sonneveldräume, sowie die Schnittbildung mit dem Raum $\mathcal{S}$. Aufgrund der Wahl von $s = 1$ ist $\mathcal{S}$ als linker Nullraum eines einzelnen Vektors in $\mathbb{R}^3$ zweidimensional.

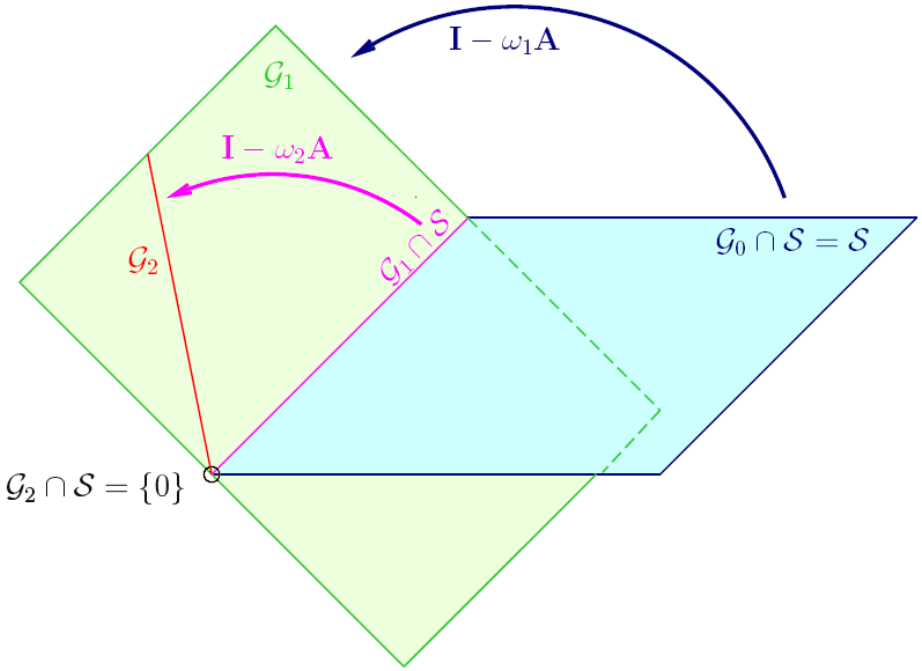

Abbildung 3.1: IDR(1), Abbildungen und Räume

In der nächsten Grafik (Abbildung 3.2, übernommen aus [Gut09]) sind die Konstruktionen der Residuen der ersten beiden Schritte zu sehen. $\mathbf{r}_0$ und $\mathbf{r}_1$ sind als Startwerte gegeben. Im ersten Schritt wird daraus der Vektor $\mathbf{v}_1$ gebildet, welcher durch die IDR-Abbildung $(\mathbf{I} - \omega_1 \mathbf{A})$ das erste Primärresiduum erzeugt. Mit den Residuen $\mathbf{r}_1$ und $\mathbf{r}_2$ wird im zweiten Schritt über die Konstruktion von $\mathbf{v}_2$ das Zwischenresiduum $\mathbf{r}_3$ konstruiert.

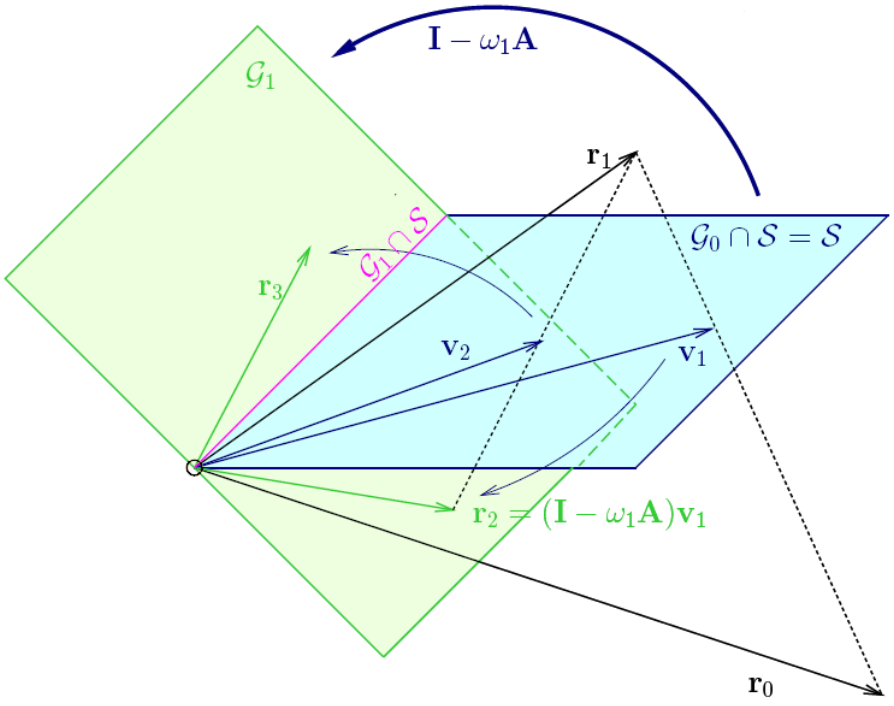

Abbildung 3.2: IDR(1), Konstruktion der Vektoren

Abbildung 3.3 aus [Gut09] zeigt schließlich die letzten Schritte bis der Raum $\mathcal{G}_2\cap\mathcal{S}$ zum Nullraum zusammenfällt. Hier ist auch der Sonderfall einer linearen Abhängigkeit zweier $\mathbf{v}$-Vektoren zu sehen, da die Systemdimension mit 3 sehr klein gewählt ist und dadurch bereits $\mathcal{G}_1\cap\mathcal{S}$ eindimensional ist. Diese lineare Abhängigkeit führt zu einer singulären Matrix $\mathbf{P}^*\Delta\,\mathbf{R}$, so dass das $s\times s$-Gleichungssystem in Zeile 14 von Algorithmus 3.1 nicht mehr gelöst werden kann. Bei realistischen Größenordnungen des Gleichungssytems tritt dieses Problem nicht auf. Sollte es doch dazu kommen, kann das Problem umgangen werden, indem die Anzahl der Schattenvektoren verringert und damit $\mathcal{S}$ vergrößert wird (siehe Abschnitt 3.2.5).

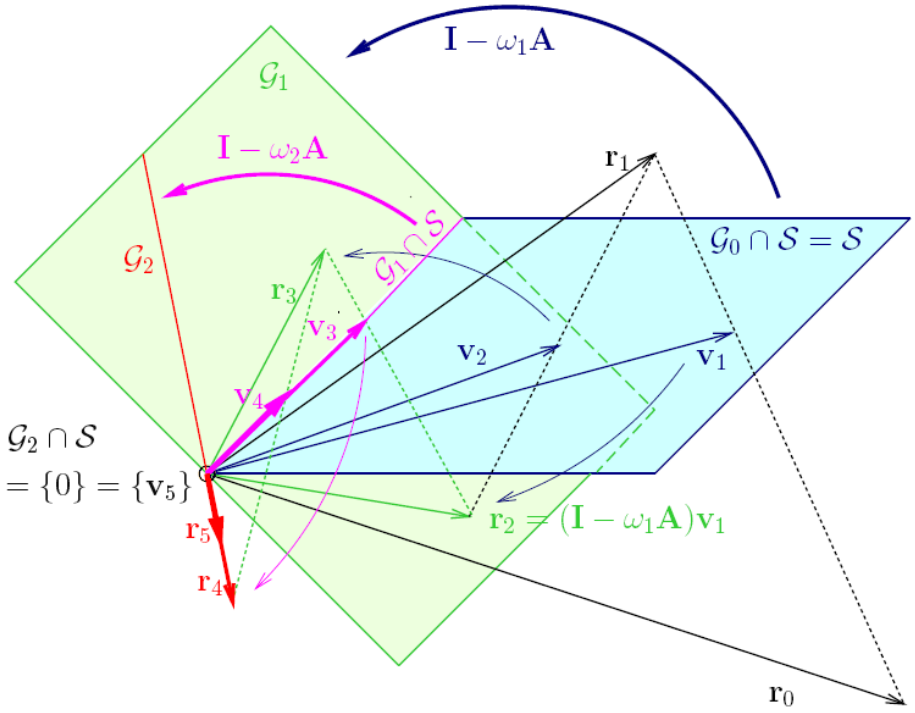

Abbildung 3.3: IDR(1), die letzten Schritte

3.2.2 IDR(s) mit Biorthogonalisierung

Diese Variante der IDR-Methoden wurde in [GijS08] vorgestellt. Hierbei wird durch die geschickte Ausnutzung zweier Orthogonalitäten eine Reduzierung des Rechenaufwands bei gleich bleibendem Speicherbedarf ermöglicht. Außerdem ist diese Variante stabiler als das einfache IDR-Verfahren, was vor allem bei höheren Werten von s zum Tragen kommt.

Änderungen in der Notation

Die Vektoren $\Delta\,\mathbf{r}_k = \mathbf{r}_{k+1} - \mathbf{r}_k$, die beim einfachen IDR-Verfahren die jeweiligen Sonneveldräume $\mathcal{G}_j$ aufgebaut haben, werden hier durch die Vektoren $\mathbf{g}_k$ ersetzt. Durch eine aufwändigere Berechnung der einzelnen Vektoren ist die elegante Verwendung der Vorwärtsdifferenzen nicht mehr möglich. Analog zu den Residuen ersetzen die Vektoren $\mathbf{u}_k$ die Approximationsdifferenzen $\Delta\,\mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$.
Dementsprechend werden auch die Matrizen, in denen die Vektoren gespeichert werden, in $\mathbf{G}$ und $\mathbf{U}$ umbenannt.

Idee des Verfahrens

Die Zwischenresiduen des IDR-Verfahrens müssen vor allem die Bedingung erfüllen, im gleichen Raum $\mathcal{G}_j$ wie das Primärresiduum zu liegen. Dies lässt vermuten, dass deren Konstruktion auch modifiziert werden kann, um daraus Vorteile ziehen zu können.
Anstatt für die Berechnung eines neuen Zwischenresiduums $\mathbf{r}_{k+1}$ nur das Residuum $\mathbf{r}_k$ aus dem vorgehenden Schritt sowie einen neuen Vektor $(\Delta\,\mathbf{r}_k)$ aus $\mathcal{G}_j$ zu benutzen (vergleiche dazu Zeile 25 von Algorithmus 3.1), können auch alle bereits berechneten Vektoren aus $\mathcal{G}_j$ miteinbezogen werden. Das neue Residuum liegt dann als Linearkombination von mehreren Vektoren aus $\mathcal{G}_j$ noch immer im Raum $\mathcal{G}_j$. Sind beispielsweise k Vektoren $\mathbf{g}_i \in \mathcal{G}_j$ bereits vorhanden, kann das neue Residuum wie folgt berechnet werden:

$$\mathbf{r}_{k+1} = \mathbf{r}_k - \sum_{i=1}^{k} \beta_i\,\mathbf{g}_i$$

Die Skalare β_i können dabei beispielsweise so gewählt werden, dass das neue Residuum orthogonal zu den ersten k Schattenvektoren der Matrix $\mathbf{P}$ steht. Die gleiche Idee kann dabei auch auf die Konstruktion der Vektoren $\mathbf{g}_k$ angewendet werden, welche ebenfalls der Bedingung $\mathbf{g}_k \in \mathcal{G}_j$ genügen müssen.

$$\mathbf{g}_k = -\Delta\,\mathbf{r}_k - \sum_{i=1}^{k-1} \alpha_i\,\mathbf{g}_i$$

Hierbei können die Skalare α_i ebenfalls so gewählt werden, dass $\mathbf{g}_k$ orthogonal zu den ersten $k-1$ Schattenvektoren steht.
Es bestehen also im Schritt $k+1$ nach dem Übergang zu einem neuen Sonneveldraum die beiden Orthogonalitäten

$$\mathbf{r}_{k+1} \perp \{\,\mathbf{p}_1, \ldots, \mathbf{p}_k\} \quad \text{und} \quad \mathbf{g}_k \perp \left\{\,\mathbf{p}_1, \ldots, \mathbf{p}_{k-1}\right\}.$$

Ein positiver Nebeneffekt dieser Beziehung besteht darin, dass für die Konstruktion eines neuen Primärresiduums $\mathbf{r}_{k+1} = (\,\mathbf{I} - \omega_{j+1}\,\mathbf{A})\,\mathbf{v}$ der Vektor $\mathbf{v}$ als letztes Zwischenresiduum $\mathbf{r}_k$ gewählt werden kann, da $\mathbf{r}_k \in \mathcal{G}_j \cap \mathcal{S}$ gilt. Es muss also in diesem Schritt nicht explizit ein $\mathbf{v}$ berechnet werden.
Durch geschicktes Überspeichern alter, nicht mehr benötigter Vektoren, bleibt der Speicherbedarf bei dieser IDR-Variante im Vergleich zum einfachen IDR-Verfahren unverändert.

Konstruktion der Zwischenresiduen

In diesem Abschnitt soll gezeigt werden, wie sich die beschriebenen Orthogonalitätsbeziehungen in den IDR-Algorithmus einfügen lassen.

Angenommen es sind bereits $k < s$ Residuen im aktuellen Raum $\mathcal{G}_j$ vorhanden, so hat die Matrix $\mathbf{G}_k \in \mathbb{C}^{n,s}$ die Form

$$\mathbf{G}_k = \left[\mathbf{g}_1 \ \cdots \ \mathbf{g}_{k-1} \ \mathbf{g}_{-s-k-1} \ \cdots \ \mathbf{g}_{-1} \right].$$

Hierbei bezeichnen die Vektoren g mit negativem Index Vektoren aus dem vorangegangen Schritt. Das heißt $\mathbf{g}_{-s-k-1}, \ldots, \mathbf{g}_{-1} \in \mathcal{G}_{j-1}$. Diese Anordnung der neu berechneten Vektoren in die Matrix $\mathbf{G}$ bewirkt, dass die i-te Spalte von $\mathbf{G}$ stets orthogonal zu den $(i-1)$-ten Schattenvektoren $\mathbf{p}_1 \ldots \mathbf{p}_i$ ist.
Die Berechnung der neuen Vektoren $\mathbf{g}_k$, $\mathbf{u}_k$, $\mathbf{r}_{k+1}$, $\mathbf{x}_{k+1}$ geschieht durch folgende Schritte:

1) v berechnen: Um $\mathbf{v} \in \mathcal{G}_{j-1} \cap \mathcal{S}$ zu finden, muss wie beim einfachen IDR-Verfahren ein Gleichungssystem gelöst werden. Hier hat das System aufgrund der Orthogonalitäten jedoch eine leichter zu berechnende Dreiecksform. $\mu_{i,k}$ steht abkürzend für das Skalarprodukt des i-ten Schattenvektors mit der k-ten Spalte von $\mathbf{G}$, während ϕ_i das entsprechende Skalarprodukt mit dem aktuellen Residuum bezeichnet.

$$\mathbf{P}^* \mathbf{G}_k \mathbf{y} = \mathbf{P}^* \mathbf{r}_k$$

$$\begin{bmatrix} \mu_{1,1} & 0 & \cdots & \cdots & 0 \\ \mu_{2,1} & \mu_{2,2} & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ \mu_{s,1} & \mu_{s,2} & \cdots & \cdots & \mu_{s,s} \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \vdots \\ \vdots \\ \vdots \\ \gamma_s \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ \phi_k \\ \vdots \\ \phi_s \end{bmatrix}$$

In dieser Darstellung ist zu sehen, dass die Werte $\gamma_1, \ldots, \gamma_{k-1}$ gleich Null sind, wodurch sich die Berechnung von $\mathbf{v}$ verkürzt:

$$\mathbf{v} = \mathbf{r}_k - \sum_{i=k}^{s} \gamma_i \, \mathbf{g}_{-s-i-1}$$

Es werden für die Konstruktion von $\mathbf{v}$ nur die *alten* Vektoren $\mathbf{g}_{-s-k-1}$, …, $\mathbf{g}_{-1}$ aus $\mathcal{G}_{j-1}$ benutzt, wohingegen beim einfachen IDR-Verfahren alle aktuellen Spaltenvektoren von ΔR bei der Berechnung von $\mathbf{v}$ zum Einsatz kamen.
Anschaulich bedeutet dies, dass durch die vorangegangene Orthogonalisierung die Schnittbildung von $\mathcal{G}_{j-1}$ mit $\mathcal{S}$ mit weniger Aufwand möglich ist.

2) Vektoren in $\mathcal{G}_j$: Nun wird durch die IDR-Abbildung $(\mathbf{I} - \omega_j \mathbf{A})\,\mathbf{v}$ ein neuer Vektor in $\mathcal{G}_j$ erzeugt. Theoretisch ist dies bereits das neue Residuum. Allerdings wird hier, wie beim einfachen IDR-Verfahren, die

Berechnung von $\mathbf{g}_k$ und $\mathbf{u}_k$ der Berechnung von $\mathbf{r}_{k+1}$ und $\mathbf{x}_{k+1}$ vorgezogen. Mit

$$\mathbf{r}_{k+1} = (\mathbf{I} - \omega_j \mathbf{A})\,\mathbf{v} = \mathbf{r}_k - \omega_j \mathbf{A}\,\mathbf{v} - \sum_{i=k}^{s} \gamma_i \mathbf{g}_{-s-i-1}$$

lässt sich $\mathbf{g}_k$ als

$$\mathbf{g}_k = -\Delta \mathbf{r}_k = \mathbf{r}_k - \mathbf{r}_{k+1} = \omega_j \mathbf{A}\,\mathbf{v} + \sum_{i=k}^{s} \gamma_i \mathbf{g}_{-s-i-1}$$

schreiben. Da auch hier für alle i die Beziehung $\mathbf{g}_i = \mathbf{A}\,\mathbf{u}_i$ gilt, können die Formeln weiter umgeformt werden, um so auch $\mathbf{u}_k$ zu erhalten. Die folgenden Formeln zeigen die Konstruktionen, die auch in der Implementierung des Verfahrens verwendet werden:

$$\begin{aligned} \mathbf{u}_k &= \omega_j \mathbf{v} + \sum_{i=k}^{s} \gamma_i \mathbf{u}_{-s-i-1}, \\ \mathbf{g}_k &= \mathbf{A}\,\mathbf{u}_k. \end{aligned}$$

3) Orthogonalisierung von $\mathbf{g}_k$: Dieser Vektor $\mathbf{g}_k$ muss nun mithilfe der $k-1$ bereits vorhandenen Vektoren $\mathbf{g}_1, \ldots, \mathbf{g}_{k-1}$ gegen die ersten $k-1$ Schattenvektoren aus $\mathbf{P}$ orthogonalisiert werden. Außerdem wird auch $\mathbf{u}_k$ aktualisiert, um die Beziehung $\mathbf{g}_k = \mathbf{A}\,\mathbf{u}_k$ zu erhalten:

for $i = 1$ to $k-1$ **do**
  $\alpha = \mathbf{p}_i^*\,\mathbf{g}_k / \mu_{i,i} = \mathbf{p}_i^*\,\mathbf{g}_k / \mathbf{p}_i^*\,\mathbf{g}_i$
  $\mathbf{g}_k = \mathbf{g}_k - \alpha\,\mathbf{g}_i$
  $\mathbf{u}_k = \mathbf{u}_k - \alpha\,\mathbf{u}_i$
end for

Es sei hier jedoch angemerkt, dass der erste Vektor $\mathbf{g}_1$ in einem neuen Sonneveldraum nicht orthogonalisiert werden kann, sondern, dass die Orthogonalitätsbedingung erst mit $\mathbf{g}_2$ erfüllt wird. Dass das obige Vorgehen tatsächlich die Orthogonalität

$$\mathbf{g}_k \perp \left\{\mathbf{p}_1, \ldots, \mathbf{p}_{k-1}\right\}$$

erzeugt, lässt sich induktiv zeigen:

Beweis. Es gilt für ein festes k, dass $\mathbf{g}_{k-1} \perp \left\{\mathbf{p}_1, \ldots, \mathbf{p}_{k-2}\right\}$. Es muss nun induktiv gezeigt werden, dass im i-ten Schritt ($i \leq k-1$) die Orthogonalität von $\mathbf{g}_k$ zu $\mathbf{p}_i$ erzeugt wird. Für den ersten Schritt gilt

$$\mathbf{g}_k^{\text{neu}} = \mathbf{g}_k - \frac{\mathbf{p}_1^*\,\mathbf{g}_k}{\mathbf{p}_1^*\,\mathbf{g}_1}\,\mathbf{g}_1.$$

Die Multiplikation mit $\mathbf{p}_1^*$ von links zeigt $\mathbf{g}_k \perp \mathbf{p}_1$:

$$\mathbf{p}_1^* \, \mathbf{g}_k^{\text{neu}} = \mathbf{p}_1^* \, \mathbf{g}_k - \frac{\mathbf{p}_1^* \, \mathbf{g}_k}{\mathbf{p}_1^* \, \mathbf{g}_1} \, \mathbf{p}_1^* \, \mathbf{g}_1 = 0.$$

Mit der Induktionsvoraussetzung $\mathbf{g}_k \perp \left\{ \mathbf{p}_1, \ldots, \mathbf{p}_{i-1} \right\}$ gilt für $\mathbf{g}_k^{\text{neu}}$ im i-ten Schritt $(i \leq k-1)$

$$\mathbf{g}_k^{\text{neu}} \quad = \quad \mathbf{g}_k - \frac{\mathbf{p}_i^* \, \mathbf{g}_k}{\mathbf{p}_i^* \, \mathbf{g}_i} \, \mathbf{g}_i$$

die Orthogonalität $\mathbf{g}_k^{\text{neu}} \perp \mathbf{p}_i$, wie die Multiplikation mit $\mathbf{p}_i^*$ von links zeigt:

$$\mathbf{p}_i^* \, \mathbf{g}_k^{\text{neu}} = \mathbf{p}_i^* \, \mathbf{g}_k - \frac{\mathbf{p}_i^* \, \mathbf{g}_k}{\mathbf{p}_i^* \, \mathbf{g}_i} \, \mathbf{p}_i^* \, \mathbf{g}_i = 0.$$

Die Orthogonalität von $\mathbf{g}_k$ zu vorherigen Vektoren $\mathbf{p}_1, \ldots, \mathbf{p}_{i-1}$ wird im Übrigen durch die Addition im i-ten Schritt nicht beeinflusst, da nur ein Vielfaches von $\mathbf{g}_i$ addiert wird und dieser Vektor $\mathbf{g}_i$ bereits orthogonal zu $\mathbf{p}_1, \ldots, \mathbf{p}_{i-1}$ steht. □

4) neues Zwischenresiduum $\mathbf{r}_{k+1}$: Das neue Residuum soll der Bedingung $\mathbf{r}_{k+1} \perp \{ \mathbf{p}_1, \ldots, \mathbf{p}_k \}$ genügen. Durch folgende Konstruktion kann außerdem auch die neue Approximation $\mathbf{x}_{k+1}$ berechnet werden:

$$\begin{aligned} \mathbf{r}_{k+1} &= \mathbf{r}_k - \frac{\phi_k}{\mu_{k,k}} \, \mathbf{g}_k, \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \frac{\phi_k}{\mu_{k,k}} \, \mathbf{u}_k \end{aligned}$$

Da nach Voraussetzung $\mathbf{r}_k, \ \mathbf{g}_k \perp \left\{ \mathbf{p}_1, \ldots, \mathbf{p}_{k-1} \right\}$ schon erfüllt ist, muss lediglich $\mathbf{r}_{k+1} \perp \mathbf{p}_k$ gezeigt werden:

$$\mathbf{p}_k^* \, \mathbf{r}_{k+1} = \mathbf{p}_k^* \, \mathbf{r}_k - \frac{\mathbf{p}_k^* \, \mathbf{r}_k}{\mathbf{p}_k^* \, \mathbf{g}_k} \, \mathbf{p}_k^* \, \mathbf{g}_k = 0$$

Damit bestehen nun nach jedem Schritt k innerhalb eines Sonneveldraums $\mathcal{G}_j$ die beiden Orthogonaliäten

$$\mathbf{r}_{k+1} \perp \{ \mathbf{p}_1, \ldots, \mathbf{p}_k \} \ \text{ und } \ \mathbf{g}_k \perp \left\{ \mathbf{p}_1, \ldots, \mathbf{p}_{k-1} \right\}.$$

Ein weiterer Vorteil dieses Verfahrens gegenüber der einfachen IDR-Methode ist eine schnellere Berechnung der neuen rechten Seite $[\phi_1 \ \ldots \ \phi_s]^T$ des $s \times s$-Gleichungssystems:

$$\phi_i^{\text{neu}} = \mathbf{p}_i^* \, \mathbf{r}_{k+1} = \mathbf{p}_i^* \left(\mathbf{r}_k - \frac{\phi_k^{\text{alt}}}{\mu_{k,k}} \, \mathbf{g}_k \right) = \phi_i^{\text{alt}} - \frac{\phi_k^{\text{alt}} \mu_{i,k}}{\mu_{k,k}} \ , \ i = 1, \ldots, s$$

Das Matrix-Vektorprodukt $\mathbf{P}^* \, \mathbf{r}_{k+1}$ kann also durch diese Skalaroperation ersetzt werden.
Der Algorithmus 3.2, der größtenteils aus [GijS08] entnommen ist, fasst alle Schritte der beschriebenen IDR-Variante zusammen.

Input: $\mathbf{A}$, $\mathbf{b}$, s, $\mathbf{x}_0$, $\mathbf{P}$, tol
Output: $\mathbf{x}$, sodass $\|\mathbf{b} - \mathbf{A}\,\mathbf{x}\| < \text{tol}$

$\mathbf{r} = \mathbf{b} - \mathbf{A}\,\mathbf{x}_0$
$[\mathbf{g}_1 \dots \mathbf{g}_s] = \mathbf{G} = 0$ /* $\mathbf{G} \in \mathbb{C}^{n,s}$ */
$[\mathbf{u}_1 \dots \mathbf{u}_s] = \mathbf{U} = 0$ /* $\mathbf{U} \in \mathbb{C}^{n,s}$ */
$\mathbf{M} = \mathbf{I}_s$ /* $\mathbf{M} = [\mu_{i,j}]$ */
$\omega = 1$
while $\|\mathbf{r}\| > \text{tol}$ **do**
  /* *rechte Seite des* $s \times s$ *Gleichungssystems* */
  $[\phi_1 \dots \phi_s]^T = \mathbf{f} = \mathbf{P}^*\,\mathbf{r}$
  for $k = 1$ to s **do**
    löse nach $\mathbf{y}$ auf: $\mathbf{M}\,\mathbf{y} = \mathbf{f}$
    $\mathbf{v} = \mathbf{r} - \sum_{i=k}^{s} \gamma_i\,\mathbf{g}_i$ /* $\mathbf{v} \in \mathcal{G}_{j-1} \cap \mathcal{S}$ */
    $\mathbf{u}_k = \omega\,\mathbf{v} + \sum_{i=k}^{s} \gamma_i\,\mathbf{u}_i$
    $\mathbf{g}_k = \mathbf{A}\,\mathbf{u}_k$ /* $\mathbf{g}_k \in \mathcal{G}_j$ */
    /* *Orthogonalisierung von* $\mathbf{g}_k$ *gegen* $\mathbf{p}_1, \dots, \mathbf{p}_{k-1}$ */
    for $i = 1$ to $k - 1$ **do**
      $\alpha = \mathbf{p}_i^*\,\mathbf{g}_k / \mu_{i,i}$
      $\mathbf{g}_k = \mathbf{g}_k - \alpha\,\mathbf{g}_i$
      $\mathbf{u}_k = \mathbf{u}_k - \alpha\,\mathbf{u}_i$
    end for
    /* *Aktualisierung der* $s \times s$*-Matrix* $\mathbf{M}$ */
    for $i = k$ to s **do**
      $\mu_{i,k} = \mathbf{p}_i^*\,\mathbf{g}_k$ /* $\mathbf{M} = \mathbf{P}^*\,\mathbf{G} = [\mu_{i,j}]$ */
    end for
    $\beta = \phi_k / \mu_{k,k}$
    /* *neues Zwischenresiduum, orthogonal zu* $\mathbf{p}_1, \dots, \mathbf{p}_k$ */
    $\mathbf{r} = \mathbf{r} - \beta\,\mathbf{g}_k$
    $\mathbf{x} = \mathbf{x} + \beta\,\mathbf{u}_k$
    if $k \neq s$ **then**
      $[\phi_1 \dots \phi_k] = [0 \dots 0]$
      /* *Aktualisierung der rechten Seite des* $s \times s$*-Systems* */
      for $i = k + 1$ to s **do**
        $\phi_i = \phi_i - \beta\mu_{i,k}$
      end for
    end if
  end for
  $\omega = \frac{\langle \mathbf{r}, \mathbf{A}\,\mathbf{r} \rangle}{\|\mathbf{A}\,\mathbf{r}\|^2}$ /* ω *für neuen Raum* $\mathcal{G}_{j+1}$ */
  $\mathbf{r} = \mathbf{r} - \omega\,\mathbf{A}\,\mathbf{r}$ /* *neues Primärresiduum* $\mathbf{r}^{neu} = (\,\mathbf{I} - \omega\,\mathbf{A})\,\mathbf{r}^{alt}$ */
  $\mathbf{x} = \mathbf{x} + \omega\,\mathbf{r}$
end while

Algorithmus 3.2: IDR(s) mit Biorthogonalisierung

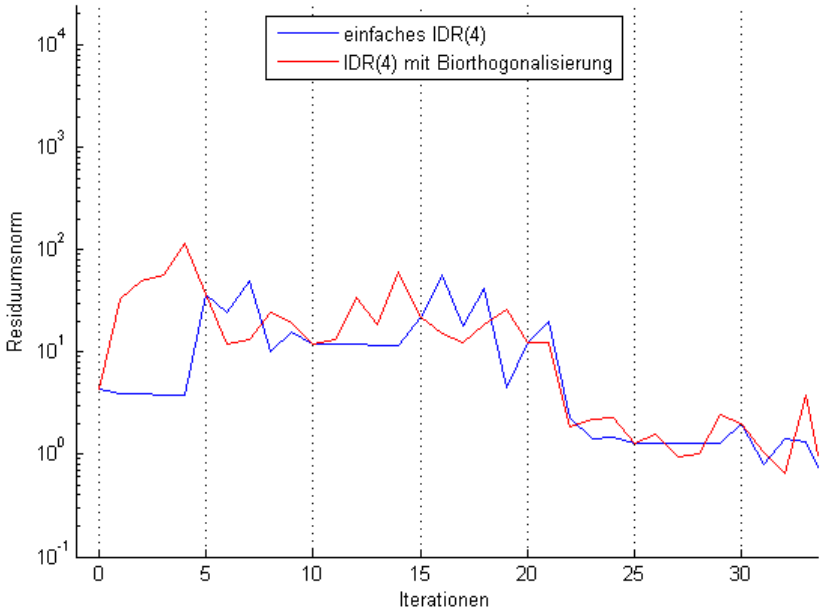

Abbildung 3.4: Zusammenhang der Primärresiduen

3.2.3 Zusammenhang der Varianten

Die Anzahl der Matrix-Vektor-Multiplikationen für einen kompletten Zyklus, das heißt, für die Berechnung von s Zwischenresiduen und einem Primärresiduum, bleibt mit $s+1$ unverändert. Allerdings werden beim IDR-Verfahren mit Biorthogonalisierung etwas weniger Vektor-Updates benötigt. Ein Vektor-Update bezeichnet dabei eine Operation der Form $a\,\mathbf{x} + \mathbf{y}$.
Überraschenderweise produzieren beide Verfahren die gleichen Primärresiduen: Jeweils jedes $(s+1)$-te Residuum und jede $(s+1)$-te Approximation stimmen überein. Dazu wurde für Abbildung 3.4 ein System mit einer tridiagonalen Toeplitzmatrix der Dimension $n = 50$ und dem IDR-Parameter $s = 4$ gelöst. Die Primärresiduen scheinen also unabhängig von der Berechnung der Zwischenresiduen konstruiert zu werden. Das bedeutet, dass die Biorthogonalisierung keine Verbesserung im Hinblick auf die benötigte Anzahl an Iterationen erreicht. Allerdings wirkt sich die Orthogonalität dieser Variante positiv auf die numerische Stabilität der Berechnung der Zwischenresiduen und damit auch auf die des gesamten Verfahrens aus.
In [Gut09] und [SleSG08] wird unter der Voraussetzung einer konstanten Dimensionsreduktion von s gezeigt, dass sich die (eindeutig bestimmten) Primärresiduen, beziehungsweise die Vektoren $\mathbf{v} \in \mathcal{G}_j \cap \mathcal{S}$, die unmittelbar zuvor berechnet worden sind, auf einen bestimmten Vektor in $\mathcal{G}_0 = \mathcal{K}_n$ zurückführen lassen:

Für jedes Residuum $\mathbf{r} \in \mathcal{G}_j$ gilt $\mathbf{r} = (\mathbf{I} - \omega_j \mathbf{A})\, \mathbf{v}_{j-1}^{(\mathbf{r})}$ mit $\mathbf{v}_{j-1}^{(\mathbf{r})} \in \mathcal{G}_{j-1} \cap \mathcal{S}$. Aufgrund des Bildungsgesetzes der Sonneveldräume gibt es für $\mathbf{v}_{j-1}^{(\mathbf{r})}$ ein $\mathbf{v}_{j-2}^{(\mathbf{r})} \in \mathcal{G}_{j-2} \cap \mathcal{S}$, so dass $\mathbf{v}_{j-1}^{(\mathbf{r})} = (\mathbf{I} - \omega_{j-1} \mathbf{A})\, \mathbf{v}_{j-2}^{(\mathbf{r})}$. Dies lässt sich induktiv fortsetzen bis $\mathbf{v}_1^{(\mathbf{r})} = (\mathbf{I} - \omega_1 \mathbf{A})\, \mathbf{v}_0^{(\mathbf{r})}$ mit $\mathbf{v}_0^{(\mathbf{r})} \in \mathcal{G}_0 \cap \mathcal{S}$. Indem man die Reihenfolge dieser Kette umkehrt, gelangt man zur folgenden Darstellung:

$$\begin{aligned}
\mathbf{v}_0^{(\mathbf{r})} &= \Omega_0(\mathbf{A})\, \mathbf{v}_0^{(\mathbf{r})}, \\
\mathbf{v}_1^{(\mathbf{r})} &= \Omega_1(\mathbf{A})\, \mathbf{v}_0^{(\mathbf{r})}, \\
&\vdots \\
\mathbf{v}_{j-1}^{(\mathbf{r})} &= \Omega_{j-1}(\mathbf{A})\, \mathbf{v}_0^{(\mathbf{r})}, \\
\mathbf{r} &= \Omega_j(\mathbf{A})\, \mathbf{v}_0^{(\mathbf{r})}.
\end{aligned}$$

Die Vektoren $\mathbf{v}_k^{(\mathbf{r})}$, $k = 0, \ldots, j-2$, werden nicht explizit berechnet und sind für kein $\mathbf{r}$ bekannt. Die Polynome $\Omega_k(\mathbf{A}) = (\mathbf{I} - \omega_1 \mathbf{A}) \cdot \ldots \cdot (\mathbf{I} - \omega_k \mathbf{A}) \in \mathbb{P}^k$ für $k = 1, \ldots, j-1$ bilden mit $\Omega_0(\mathbf{A}) = \mathbf{I}$ eine Basis des Raums $\mathbb{P}^{j-1}$ der Polynome von Grad $\leq j-1$. Des Weiteren liegen alle $\mathbf{v}_k$ für $k = 0, \ldots, j-1$ in $\mathcal{S}$ (unabhängig von $\mathbf{r}$), so dass für alle Polynome $\Omega \in \mathbb{P}^{j-1}$ gilt:

$$\Omega(\mathbf{A})\, \mathbf{v}_0 \in \mathcal{S} \;\Leftrightarrow\; \mathbf{P}^* \Omega(\mathbf{A})\, \mathbf{v}_0 = 0 \;\Leftrightarrow\; \mathbf{v}_0 \perp \overline{\Omega}(\mathbf{A}^*)\, \mathbf{P}$$

Der dritte Ausdruck dieser Äquivalenzen ist gleichbedeutend mit

$$\mathbf{v}_0 \perp \mathcal{K}_j(\mathbf{A}^*, \mathbf{P}),$$

was auch die alternative Definition der Sonneveldräume als Block-Krylovräume rechtfertigt.
Im Falle eines Primärresiduums $\mathbf{r}_{j(s+1)}$, welches aufgrund der Krylov-Iterationsvorschrift im Raum $\mathbf{r}_0 + \mathbf{A}\mathcal{K}_{j(s+1)}$ liegt, gilt für den korrespondierenden Vektor $\mathbf{v}_0$

$$\mathbf{v}_0 \in \mathbf{r}_0 + \mathbf{A}\mathcal{K}_{j(s+1)-j} = \mathbf{r}_0 + \mathbf{A}\mathcal{K}_{js}, \tag{3.3}$$

da $\mathbf{r}_{j(s+1)} = \Omega_j(\mathbf{A})\, \mathbf{v}_0$ mit dem Polynom Ω_j von Grad j. Das bedeutet, dass hier $\mathbf{v}_0$ durch $(j \cdot s)$ Parameter bestimmt wird, welche jedoch durch die Orthogonalitätsbedingung von $\mathbf{v}_0$ an den $(j \cdot s)$-dimensionalen Block-Krylovraum $\mathcal{K}_j(\mathbf{A}^*, \mathbf{P})$ eindeutig festgelegt sind. Dies gilt für alle IDR-Verfahren und hat zur Folge, dass die jeweiligen Primärresiduen übereinstimmen. Außerdem kann man an der Beziehung (3.3) sehen, dass die Anzahl der Freiheitsgrade bei der Konstruktion der Zwischenresiduen im jeweils aktuellen Sonneveldraum mit fortschreitender Iterationszahl stetig zunimmt bis ein neuer Reduktionschritt ausgeführt wird. Beim IDR-Verfahren mit Biorthogonalisierung kann man diese Eigenschaft gut an der länger werdenden Schleife zur Orthogonalisierung der Zwischenresiduen sehen (vergleiche

Zeilen 15-19 in Algorithmus 3.2). Es kommt mit jedem Schritt ein neuer Vektor hinzu, der gegen die vorherigen orthogonalisiert werden muss. Selbstverständlich kann anstelle der Orthogonalisierung auch eine beliebige andere Taktik zur Verbesserung des Verfahrens angewendet werden, solange die Vektoren im gleichen Raum $\mathcal{G}_j$ bleiben. So wäre es beispielsweise möglich, die Vektoren $\mathbf{g}_k$ innerhalb eines Sonneveldraums gegeneinander zu orthogonalisieren, um die Berechnung der Lösung des $s \times s$-Gleichungssytems zu stabilisieren. Es ist jedoch noch nicht gelungen, eine effizientere Modifikation als die hier vorgestellte zu implementieren.

3.2.4 Berechnung von ω

Der Parameter ω aus der IDR-Abbildung $\mathbf{I} - \omega\,\mathbf{A}$ wird in den beiden beschriebenen IDR-Varianten so gewählt, dass er die Norm des neuen Residuums $\mathbf{r} = (\mathbf{I} - \omega\,\mathbf{A})\,\mathbf{v}$ minimiert:

$$\min \|\,\mathbf{r}\|_2 = \min \|\,\mathbf{v} - \omega\,\mathbf{A}\,\mathbf{v}\|_2 \;\Rightarrow\; \omega = \frac{\langle\,\mathbf{v},\,\mathbf{A}\,\mathbf{v}\rangle}{\|\,\mathbf{A}\,\mathbf{v}\|_2}$$

Das bedeutet, dass ω stark vom Winkel zwischen $\mathbf{v}$ und $\mathbf{A}\,\mathbf{v}$ abhängt, da

$$\langle\,\mathbf{v},\,\mathbf{A}\,\mathbf{v}\rangle = \|\,\mathbf{v}\|_2 \|\,\mathbf{A}\,\mathbf{v}\|_2 \cdot \cos\angle(\,\mathbf{v},\,\mathbf{A}\,\mathbf{v}).$$

Problematisch wird diese Beziehung, wenn $\mathbf{A}$ (beinahe) schiefhermitesch ist, denn für eine schiefhermitesche Matrix gilt $\mathbf{A}^* = -\,\mathbf{A}$ und somit ist das Skalarprodukt $\langle\,\mathbf{v},\,\mathbf{A}\,\mathbf{v}\rangle$ gleich Null:

$$\langle\,\mathbf{v},\,\mathbf{A}\,\mathbf{v}\rangle = \,\mathbf{v}^*\,\mathbf{A}^*\,\mathbf{v} = -\,\mathbf{v}^*\,\mathbf{A}\,\mathbf{v} = -\,\langle\,\mathbf{v},\,\mathbf{A}\,\mathbf{v}\rangle \;\Rightarrow\; \langle\,\mathbf{v},\,\mathbf{A}\,\mathbf{v}\rangle = 0$$

In solchen Fällen muss die Berechnung entsprechend verändert werden. Sind jedoch keine oder nur wenige Informationen über die Matrix $\mathbf{A}$ bekannt, so kann ω durch Festlegen eines Minimalwertes daran gehindert werden, zu klein zu werden, um damit eine mögliche Stagnation des Verfahrens zu vermeiden. So wäre im Extremfall von $\omega = 0$ die IDR-Abbildung $(I - \omega A)$ gleich der Einheitsmatrix. In den in Kapitel 5 verwendeten Algorithmen ist diese Abfrage implementiert.

3.2.5 Wahl der Schattenvektoren

Da in jedem Schritt eine Multiplikation mit der Matrix $\mathbf{P}$ durchgeführt werden muss, ist die Struktur dieser Matrix bei der Geschwindigkeit einer einzelnen Iteration ausschlaggebend. Wählt man $\mathbf{P}$ so, dass in jeder Zeile nur maximal ein Element ungleich Null steht, kann die Laufzeit des Algorithmus leicht verbessert werden, da die Multiplikationen effizienter ausgeführt werden können und auch leichter parallelisiert werden können.
Des Weiteren kann das $s \times s$-System, welches in jedem Schritt gelöst werden

muss, schlecht konditioniert sein, da beispielsweise die Spalten der Matrix $\mathbf{G}$, beziehungsweise $\Delta\mathbf{R}$, (fast) linear abhängig sind. In diesem Fall kann s und damit die Anzahl der Schattenvektoren reduziert werden. Dazu berechnet man die Singulärwertzerlegung von $\mathbf{M}$, wobei $\mathbf{M}$ die Matrix des $s \times s$-Systems bezeichnet und entfernt die Singulärwerte, die unter einer gewissen Schranke δ liegen:

$$\mathbf{M} = \mathbf{Q} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_s \end{bmatrix} \mathbf{V}^* \text{ mit } \mathbf{Q}, \mathbf{V} \in \mathbb{C}^{s,s} \text{ unitär },$$

$$\mathbf{P}^{\text{neu}} = \mathbf{P}\,[\,\mathbf{p}_1 \dots \mathbf{p}_{s^{\text{neu}}}], \text{ mit } s^{\text{neu}}, \text{ so dass } \sigma_1, \dots \sigma_{s^{\text{neu}}} > \delta.$$

Die Matrizen $\mathbf{G}$ und $\mathbf{U}$, beziehungsweise $\Delta\mathbf{R}$ und $\Delta\mathbf{X}$ müssen daher ebenfalls verkürzt werden.
In zahlreichen numerischen Tests hat sich die Wahl von $\mathbf{P}$ als orthogonalisierte komplexe Zufallsmatrix bewährt. Sämtliche Versuche, Informationen der Systemmatrix $\mathbf{A}$ in die Konstruktion von $\mathbf{P}$ einfließen zu lassen, sind gescheitert, da sie zu starken Geschwindigkeitseinbußen, beziehungsweise zu völliger Stagnation geführt haben.
Die oben beschriebene Wahl von $\mathbf{P}$ als „quasi dünn besetzte“ Matrix scheint neben der einer orthogonalisierten Zufallsmatrix zurzeit die sinnvollste und erfolgversprechendste zu sein.
Allerdings sei hier angemerkt, dass durch die Verwendung einer speziell an das System angepassten Matrix $\mathbf{P}$ im Vergleich zu einer Zufallsmatrix, die Effizienz der IDR-Methoden sehr wahrscheinlich gesteigert werden kann. Da jedoch die Berechnungen der einzelnen Vektoren stark von $\mathbf{P}$ abhängig sind, führt eine spezielle Wahl von $\mathbf{P}$ meist zu einer (fast) singulären Matrix $\mathbf{M}$, so dass das Verfahren infolgedessen numerischen Ungenauigkeiten ausgesetzt ist, beziehungsweise völlig stagniert.
In einer früheren Version von [SonG08], die online zu finden ist (siehe [Gij08]), befindet sich ein Beweis, der die Wahl von $\mathbf{P}$ als Zufallsmatrix rechtfertigt. So findet bei allen bis auf endlich vielen Wahlen von $\mathbf{P}$ eine reguläre Dimensionsreduktion von s statt, vorausgesetzt man betrachtet nicht die letzten Schritte.
Um die Konvergenzeigenschaften der IDR-Verfahren durch Modifikation der Schattenvektoren zu verbessern ist daher ein tiefergehendes analytisches Verständnis des genauen Einflusses dieser Vektoren erforderlich.

3.2.6 IDR(s) mit nichtlinearen Stabilisierungspolynomen

Numerische Tests haben gezeigt, dass das IDR-Verfahren und insbesondere die Variante mit Biorthogonalisierung gängigen Methoden zur Lösung linearer Gleichungssysteme, wie beispsielsweise GMRES und BiCGstab, beziehungsweise BiCGstab(ℓ), überlegen oder zumindest ebenbürtig ist. Allerdings gibt es auch Fälle, in denen BiCGstab(ℓ) bessere Ergebnisse erzielt.

Dies tritt vor allem bei schiefsymmetrischen, beziehungsweise schiefhermiteschen Matrizen auf, wenn im IDR-Verfahren die lineare Residuumsminimierung mithilfe von ω schlecht funktioniert.
Zeitgleich sind zwei Verfahren entwickelt worden, die dieses Problem mit höhergradigen Stabilisierungspolynomen lösen: Zum einen ein Verfahren names IDR(s)stab(ℓ), das in [SleG09] beschrieben ist, und zum anderen das Verfahren GBi-CGSTAB(s, ℓ) (siehe [TanS09]). Die beiden Verfahren wurden unabhängig voneinander entwickelt und stehen laut [TanS09] im gleichen Verhältnis zueinander, wie das Jacobi- und das Gauß-Seidel-Verfahren (siehe dazu [Saa03], Seite 104 ff). Beide Verfahren sind sehr kompliziert und da sie leider für diese Arbeit zu spät veröffentlicht wurden, konnten sie nicht mehr implementiert werden. Für numerische Ergebnisse muss daher auf die Artikel der jeweiligen Autoren verwiesen werden. Beide Verfahren scheinen jedoch ein großes Potential für die numerische Mathematik zu haben und sollten weiter untersucht werden.

Idee der Verfahren

Vom ursprünglichen Konzept der IDR-Methoden ist bei dieser Variation nicht mehr viel zu erkennen, da grundlegende Änderungen durchgeführt werden mussten, um die höhergradige Minimierung zu ermöglichen. Für nähere Informationen seien die schon genannten Artikel [SleG09] und [TanS09] empfohlen.
Grob umrissen, lässt sich die IDR(s)stab(ℓ)-Methode auf die folgenden zwei Schritte zusammenfassen:

1. Führe ℓ IDR(s)-Schritte durch. Dies benötigt also $\ell \cdot s$ Iterationen und liefert ℓ Primärresiduen $\mathbf{r}_1, \ldots \mathbf{r}_\ell$. Dieser Schritt wird von den Autoren als *IDR-Schritt* bezeichnet.

2. Konstruiere ein neues Residuum $\mathbf{r} = \mathbf{r}_0 - \sum_{i=1}^{\ell} \gamma_i \mathbf{r}_i$, wobei die Parameter $\gamma_1, \ldots, \gamma_\ell$ so gewählt werden, dass die Norm $\| \mathbf{r}\|_2$ minimiert wird. Diesen Schritt bezeichnen die Autoren als *Polynomschritt*.

Wählt man $\ell = 1$, so ist das jeweilige Verfahren (mathematisch) äquivalent zur einfachen IDR(s)-Methode. Wählt man dagegen $s = 1$, so erhält man ein Verfahren, das der BiCGstab(ℓ)-Methode entspricht.

Kapitel 4

Krylov-Recycling

Dieses Kapitel befasst sich mit der Technik des Krylov-Recyclings. Man versteht darunter die Wiederverwendung von Informationen aus einem vorherigen (Krylov-)Raum bei der Konstruktion eines neuen.
Anwendung findet diese Technik zum einen bei Lösungsverfahren, die während der Laufzeit auf Neustarts angewiesen sind (→ GMRES) und zum anderen, und dies bildet den Schwerpunkt dieses Kapitels, bei der Lösung von parametrisierten Gleichungssytemen $\mathbf{A}_i\,\mathbf{x} = \mathbf{b}_i$, bei denen sich die einzelnen Matrizen nur wenig voneinander unterscheiden.
In beiden Fällen können durch geschickten Einsatz von Recycling-Techniken erhebliche Verbesserungen erzielt werden.

4.1 Recycling bei Neustarts

Am Beispiel von GMRES lässt sich diese Art des Krylov-Recyclings gut veranschaulichen. Da beim GMRES-Verfahren in jeder neuen Iteration zwei neue Vektoren gespeichert werden müssen, deren Länge außerdem linear ansteigt, ist der Speicheraufwand vor allem bei großen Systemen zunehmend schwieriger zu bewältigen. Man ist in diesen Fällen darauf angewiesen, das Verfahren mit der aktuellen Approximation als neuem Startvektor neu zu beginnen. Allerdings leidet die Konvergenzeigenschaft von GMRES unter dieser Strategie, so dass es sein kann, dass das Verfahren aufgrund von Neustarts stagniert oder zumindest die Konvergenzgeschwindigkeit sinkt.
Eine Möglichkeit diesem Problem zu begegnen, besteht darin anstelle eines einzigen Vektors, einen ganzen Raum in das neugestartete Verfahren zu integrieren. Dadurch können wünschenswerte Eigenschaften mit vergleichsweise wenig Aufwand in den folgenden Zyklus übertragen werden, um die Konvergenzgeschwindigkeit zu erhöhen, beziehungsweise zu erhalten. Der dabei benötigte Speicheraufwand bleibt konstant gering.

4.1.1 Optimal Truncation

In [Stu99] wurde ein Verfahren entwickelt, welches die Informationsverluste durch Neustarts auf ein Minimum reduzieren soll. Zur Anschauung werden die Residuen mit und ohne Verwendung von Neustarts miteinander verglichen:

- Start mit Residuum $\mathbf{r}_0$
  $\Rightarrow$ nach $s < m$ Schritten: Residuum $\mathbf{r}_s$
  $\Rightarrow$ nach m Schritten: Residuum $\mathbf{r}_m$

- Neustart nach $s < m$ Schritten mit Residuum $\mathbf{r}_s$
  $\Rightarrow$ nach weiteren $m - s$ Schritten: Residuum $\tilde{\mathbf{r}}_m$

Das heißt, nach m Schritten sind die beiden Residuen $\mathbf{r}_m$ (ohne Neustart) und $\tilde{\mathbf{r}}_m$ (mit Neustart) berechnet. Da beim Neustart der vorher berechnete Raum $\mathbf{A}\mathcal{K}_s(\mathbf{A}, \mathbf{r}_0)$ nicht mehr bei der Orthogonalisierung berücksichtigt wurde, kann man erwarten, dass $\|\tilde{\mathbf{r}}_m\|_2$ größer ist als $\|\mathbf{r}_m\|_2$. Die Differenz $\mathbf{e} = \tilde{\mathbf{r}}_m - \mathbf{r}_m$, welche es zu minimieren gilt, wird in [Stu99] und [ParSMJM06] als *Residuumsfehler* bezeichnet. Gelingt es $\mathbf{e}$ zu minimieren, so erhält man ein Verfahren, das bei gleichbleibendem Berechnungs- und Speicheraufwand eine ähnliche Konvergenzgeschwindigkeit wie das volle GMRES-Verfahren aufweist. Der Residuumsfehler ist abhängig von den kanonischen Winkeln zwischen den Räumen $\mathbf{A}\mathcal{K}_s(\mathbf{A}, \mathbf{r}_0)$ und $\mathbf{A}\mathcal{K}_{m-s}(\mathbf{A}, \mathbf{r}_s)$, also von den Winkeln zwischen dem Raum, der nicht mehr berücksichtigt wird und dem neuen Raum.
Um nun den Fehler zu verbessern, wird nicht der ganze Raum $\mathbf{A}\mathcal{K}_m(\mathbf{A}, \mathbf{r}_0)$

verworfen, sondern es wird die Orthogonalität zu einem bestimmten k-dimensionalen Unterraum in den nächsten Schritten gewahrt. So entsteht ein neues Residuum $\tilde{\mathbf{r}}_m^{(\mathrm{ot})}$ mit dem neuen Residuumsfehler $\mathbf{e}^{(\mathrm{ot})} = \tilde{\mathbf{r}}_m^{(\mathrm{ot})} - \mathbf{r}_m$. Wird dieser Unterraum so gewählt, dass er die größten kanonischen Winkel erhält, kann $\mathbf{e}^{(\mathrm{ot})}$ minimiert werden, da der ausschlaggebende Teil in die folgenden Schritte übertragen wird. Für nähere Informationen zu dieser Technik sei auf [Stu99] verwiesen. Eine sehr umfassende Analyse über die bei solchen Verfahren auftretenden Räume und deren Zusammenhänge ist in [EieES00] zu finden.

4.1.2 GMRES mit deflated restarting

Einen anderen Ansatz verfolgt eine in [Mor02] entwickelte Variante. Es werden hierfür sogenannte *harmonische Ritzvektoren* benötigt.

Definition 5. Sei $\mathbf{y}_k$ ein Eigenvektor der Matrix $\mathbf{H}_{k,k}$ aus der Arnoldi-Relation (2.2), wobei hier $\mathbf{H}_{k,k}$ die Matrix $\mathbf{H}_{k+1,k}$ ohne die letzte Zeile ist. Der Vektor

$$\mathbf{u}_k = \mathbf{V}_k \, \mathbf{y}_k,$$

mit $\mathbf{V}_k$ ebenfalls aus der Relation (2.2) heißt dann *Ritzvektor* von $\mathbf{A}$. *Ritzwerte* von $\mathbf{A}$ sind entsprechend Eigenwerte von $\mathbf{H}_{k,k}$ und haben die Eigenschaft, die äußeren Eigenwerte von $\mathbf{A}$ zu approximieren. *Harmonische Ritzwerte* approximieren dagegen die äußeren Eigenwerte von $\mathbf{A}^{-1}$, also die inneren Eigenwerte von $\mathbf{A}$.

Das GMRES-Verfahren hat die Konvergenzrate

$$\frac{\|\mathbf{r}_n\|_2}{\|\mathbf{r}_0\|_2} \leq \min_{\phi_n \in \mathbb{P}_n^0} \max_{\lambda \in \sigma(\mathbf{A})} \|\phi_n(\lambda)\|_2 \,,$$

wobei $\mathbb{P}_n^0$ die Menge aller Polynome ϕ_n von Grad n mit $\phi_n(0) = 1$ und $\sigma(\mathbf{A})$ die Menge aller Eigenwerte von $\mathbf{A}$ bezeichnet. Um diese Rate zu erhöhen, kann das Spektrum von $\mathbf{A}$ dahingehend verändert werden, dass Eigenwerte nahe bei Null entfernt werden (*deflation*). Dazu wird der benutzte Krylovraum um den Raum $\mathcal{A}$ erweitert, indem harmonische Ritzvektoren, welche Eigenvektoren zu kleinen Eigenwerten approximieren, hinzugefügt werden. Diese Methoden werden deshalb auch als *Augmented Krylov Solvers* bezeichnet. Es ergibt sich dann die verbesserte Konvergenzrate

$$\frac{\|\mathbf{r}_n\|_2}{\|\mathbf{r}_0\|_2} \leq \min_{\phi_n \in \mathbb{P}_n^0} \max_{\lambda \in (\sigma(\mathbf{A}) \setminus \mathcal{A})} \|\phi_n(\lambda)\|_2 \,.$$

Einzelheiten und weitergehende Informationen finden sich unter Anderem in [Mor02].

4.2 Recycling bei parametrisierten Gleichungssystemen

In [KilS06] wurde der Zusammenhang zwischen invarianten Unterräumen und Eigenwerten zweier Matrizen untersucht, wobei sich die Matrizen nur durch eine kleine Störung unterscheiden. Sie haben einen Satz formuliert, der den Zusammenhang zwischen unterschiedlichen $\mathbf{A}$-invarianten Räumen nach einer kleinen Störung erhellt.
Es folgen nun einige Voraussetzungen und Definitionen für diesen Satz.

Definition 6. Der *größte kanonische Winkel* θ zwischen zwei Räumen $\mathcal{X}$ und $\mathcal{Y}$ ist definiert als

$$\theta = \max_{\mathbf{x}\in\mathcal{X}\setminus\{0\}} \min_{\mathbf{y}\in\mathcal{Y}\setminus\{0\}} \angle(\mathbf{x}, \mathbf{y}).$$

Sei nun $\mathbf{A}$ eine symmetrische positiv definite Matrix mit der Faktorisierung

$$\mathbf{A} = \begin{bmatrix} \mathbf{V}_1 & \mathbf{V}_2 & \mathbf{V}_3 \end{bmatrix} \begin{bmatrix} \Lambda_1 & & \\ & \Lambda_2 & \\ & & \Lambda_3 \end{bmatrix} \begin{bmatrix} \mathbf{V}_1^T \\ \mathbf{V}_2^T \\ \mathbf{V}_3^T \end{bmatrix}$$

die Diagonalisierung von $\mathbf{A}$, wobei $\mathbf{V} = [\ \mathbf{V}_1\ \mathbf{V}_2\ \mathbf{V}_3\ ]$ eine orthogonale Matrix ist. $\Lambda_j = \text{ diag } (\lambda_1^{(j)}, \ldots, \lambda_{k_j}^{(j)})$ enthält die der Größe nach geordneten Eigenwerte von $\mathbf{A}$:

$$\lambda_1^{(1)} \leq \ldots \leq \lambda_{k_1}^{(1)} < \lambda_1^{(2)} \leq \ldots \leq \lambda_{k_2}^{(2)} < \lambda_1^{(3)} \leq \ldots \leq \lambda_{k_3}^{(3)}.$$

Sei weiter $\mathbf{E}$ eine symmetrische Matrix, die eine Störung auf $\mathbf{A}$ darstellt, so dass $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{E}$ gilt. Die Störung soll dabei hauptsächlich die großen Eigenwerte $\lambda_1^{(3)}, \ldots, \lambda_{k_3}^{(3)}$ beeinflussen, also $\|\mathbf{E}\|_F \approx \eta = \|\mathbf{E}\,\mathbf{V}_3\|_F$ und $\|\mathbf{E}\,[\ \mathbf{V}_1\ \mathbf{V}_2\ ]\|_F \leq \varepsilon$. Außerdem wird vorausgesetzt, dass $\|\mathbf{E}\|_F$ im Vergleich zum Abstand zwischen Λ_1 und Λ_3, also $\lambda_1^{(3)} - \lambda_{k_1}^{(1)}$ klein ist, und dass die Störung auf $[\mathbf{V}_1\ \mathbf{V}_2]$ verglichen mit dem Abstand zwischen $\mathbf{V}_1$ und $\mathbf{V}_2$, also $\lambda_1^{(2)} - \lambda_{k_1}^{(1)}$, gering ist.
Desweiteren seien δ und $\tilde{\delta}$ folgendermaßen definiert:

$$\begin{aligned} \delta &= \min(\lambda_1^{(2)} - \varepsilon,\ \lambda_1^{(3)} - \eta) - 2\varepsilon - (\lambda_{k_1}^{(1)} + \varepsilon) \gg \varepsilon, \\ \tilde{\delta} &= \delta\left(1 - \frac{2\varepsilon^2}{\delta^2}\right) \end{aligned}$$

Satz 3. *Sei* $\mathbf{A}$ *eine symmetrisch positiv definite Matrix mit obiger Diagonalisierung und sei* $\mathbf{E}$ *eine symmetrische Matrix mit* $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{E}$ *und den Parametern* $\varepsilon, \eta, \delta$ *und* $\tilde{\delta}$*. Dann gibt es eine Matrix* $\tilde{\mathbf{V}}_1$*, deren Spaltenvektoren einen* $(\mathbf{A} + \mathbf{E})$*-invarianten Unterraum aufspannen. Für den größten kanonischen Winkel* θ *zwischen* $\text{span}(\mathbf{V}_1)$ *und* $\text{span}(\tilde{\mathbf{V}}_1)$ *gilt*

$$\tan\theta\left(\text{span}(\mathbf{V}_1),\ \text{span}(\tilde{\mathbf{V}}_1)\right) \leq \frac{\varepsilon}{\tilde{\delta}}.$$

Außerdem gilt für die kleinen Eigenwerte $\tilde{\lambda}_j^{(1)}$ *des gestörten Systems die Beziehung*

$$\forall \tilde{\lambda}_j^{(1)} \; \exists \lambda_i^{(1)} \; mit \; |\tilde{\lambda}_j^{(1)} - \lambda_i^{(1)}| \leq \varepsilon + \frac{2\varepsilon^2}{\delta}.$$

Für den Beweis sei auf [KilS06] verwiesen.
Dieser Satz zeigt, dass sich auch durch große Störungen im Bereich der großen Eigenwerte die kleinen Eigenwerte nur sehr wenig ändern. Es ist hier nicht erforderlich, dass der Abstand zwischen Λ_1 und Λ_2, also $\lambda_1^{(2)} - \lambda_{k_1}^{(1)}$ groß ist.
Bei solchen Störungen ist es also von Vorteil einen Unterraum zu recyceln, der zu den kleinen Eigenwerten korrespondiert, da diese auch im nächsten System auftreten.
Es ist jedoch auch oft problemabhängig, welche Recyclingstrategie gute Ergebnisse erzielt. Je mehr über die Veränderungen zwischen den Systemen bekannt ist, desto besser können Recycling-Strategien daran angepasst werden.

4.3 Eignung von IDR-Methoden

In diesem Abschnitt soll die Eignung der IDR-Methoden im Hinblick auf die Verwendung von Krylov-Recycling-Strategien untersucht werden.
Verfahren, welche unterschiedliche Recycling-Techniken implementieren, haben den Anspruch die Lösung mit nur sehr wenigen Iterationen, beziehungsweise Matrix-Vektor-Multiplikationen zu berechnen, indem unter Anderem bereits mit einem sehr guten Suchraum gestartet wird. Es ist dabei wichtig, dass das schon von Anfang an relativ kleine Residuum für die folgenden Iterationen nicht wieder größer wird. Eine gewisse Minimierungseigenschaft ist also von Vorteil. Den IDR-Methoden fehlt leider ebendiese Eigenschaft, was sich in den zum Teil stark oszillierenden Residuumsnormen für verschiedene Iterationen widerspiegelt (siehe Konvergenzverläufe in Kapitel 5).
Desweiteren beruhen Recycling-Techniken auch darauf, dass ein bereits guter Suchraum für das nächste System weiter verbessert wird. Dies bedeutet, dass neue Vektoren hinzugenommen werden müssen. Bei der Klasse der IDR-Verfahren ist jedoch eine solche Vergrößerung nicht ohne Weiteres möglich, da auch die Anzahl der Schattenvektoren erhöht werden müsste. Die Dimensionen der einzelnen Matrizen und Räume der IDR-Verfahren sind durch den Parameter s nach oben beschränkt.
Außerdem haben alle IDR-Methoden gemein, dass in den ersten s Schritten ein größtenteils beliebig wählbares Krylovverfahren benutzt wird, um die s, beziehungsweise $2s$, Startvektoren zu konstruieren. Wenn also die eigentliche IDR-Idee der schrumpfenden Sonneveldräume zum Einsatz kommt, haben andere Verfahren unter Umständen die Lösung bereits gefunden.
Es besteht also ein Konflikt zwischen einer großen Freiheit bei der Konstruk-

tion der Räume und Matrizen und einem schnellen Start des eigentlichen IDR-Verfahrens.
Eine mögliche Lösung besteht in der Verwendung und Konstruktion von Vektoren aus dem alten System, so dass zur Lösung des neuen Systems ein bereits fortgeschrittener Sonneveldraum $\mathcal{G}_j$, $j > 0$ simuliert wird, den das IDR-Verfahren weiter schrumpfen lässt. Dabei ist zu beachten, dass die verwendeten Schattenvektoren für die gesamte Sequenz konstant bleiben, da sonst die Informationen eines Systems über die Sonneveldräume für die Lösung des nächsten Systems nicht mehr geeignet sind.
Ein anderer Ansatz besteht darin, die Startvektoren auf herkömmliche Art zu berechnen und stattdessen die Schattenvektoren $\mathbf{p}_1, \ldots, \mathbf{p}_s$ mit Informationen aus dem vorherigen System zu optimieren ($\rightarrow$ flexible Schattenvektoren).

4.3.1 Recycling des Lösungsraums

Es ist beim einfachen IDR-Verfahren, sowie beim IDR-Verfahren mit Biorthogonalisierung, möglich, die Anzahl der benötigten Iterationen für die Berechnung der Lösung stark zu reduzieren. Vereinfachend gesagt, besteht die Idee darin, die Vektoren, die Informationen über den letzten Sonneveldraum enthalten, als Startvektoren für das neue System zu benutzen.

Anwendung beim einfachen IDR-Verfahren

Beim einfachen IDR-Verfahren eignen sich die Spaltenvektoren der Matrix $\Delta\,\mathbf{X} = [\Delta\,\mathbf{x}_1 \ldots \Delta\,\mathbf{x}_s] \in \mathbb{C}^{n,s}$ dafür sehr gut. Da für $\Delta\,\mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, $k = 1, \ldots, s$ gilt und außerdem der finale Lösungsvektor $\mathbf{x} = \mathbf{x}_{s+1}$ mit an das neue Lösungsverfahren übergeben wird, sind die letzten $s + 1$ Approximationen des alten Systems für die Berechnung der Lösung des zweiten Systems verfügbar. Um der für das einfache IDR-Verfahren wichtigen Bedingung $\Delta\,\mathbf{R} = -\,\mathbf{A}\Delta\,\mathbf{X}$ zu genügen, ist es unnötig, auch die Matrix $\Delta\,\mathbf{R}$ zu recyceln, da sich schließlich die Systemmatrix $\mathbf{A}$ geändert hat. Durch eine Multiplikation mit der neuen Matrix $-\,\mathbf{A}_2$ kann diese Beziehung jedoch für die recycelten Vektoren wiederhergestellt werden. Diese s Matrix-Vektor-Multiplikationen $\Delta\,\mathbf{r}_k = -\,\mathbf{A}_2\Delta\,\mathbf{x}_k^{(\mathrm{rec})}$ ersetzen dabei die s Schritte des Krylovverfahrens zur Generierung der Startvektoren in $\mathcal{G}_0$.
Sollten weniger als s Vektoren aus dem alten System vorhanden sein, da die Lösung sehr schnell gefunden wurde, muss die zu recycelnde Matrix $\Delta\,\mathbf{X}$ um entsprechend viele Vektoren ergänzt werden. Hierbei kann wieder ein frei wählbares Krylovraumverfahren benutzt werden. Zusätzlich kann die numerische Stabilität erhöht werden, indem die Spaltenvektoren der Matrix $\Delta\,\mathbf{X}$ orthogonalisiert werden, bevor sie als Startvektoren für das nächste System benutzt werden. Der folgende Pseudocode beschreibt dieses Vorgehen.

```
1: /* Daten des ersten Systems: */
2: Δ X^(rec) ∈ C^{n,s̃}, x^(rec), r^(rec)
3: /* Orthogonalisierung zur Stabilisierung */
4: Δ X = orth(Δ X^(rec))
5: for k = 1 to s̃ do
6:    /* Konstruktion der korrespondierenden Matrix Δ R */
7:    Δ R(:, k) = − A_2 Δ X(:, k)
8: end for
9: for s̃ to s do
10:   /* restliche Vektoren mit Krylovverfahren erzeugen */
11: end for
```

Anwendung beim IDR-Verfahren mit Biorthogonalisierung

In der sehr effizienten Implementierung von Martin B. van Gijzen und Peter Sonneveld werden die ersten s Startvektoren implizit durch den gleichen Code erzeugt, der auch die späteren IDR-Schritte ausführt. Es scheint daher schwieriger den vorherigen Lösungsraum in das Verfahren zu integrieren. Numerische Tests haben jedoch bestätigt, dass es eine gute Wahl ist, die Initialisierung der Matrizen $\mathbf{G}$ und $\mathbf{U}$ als Nullmatrizen zu ersetzen. Stattdessen werden die Matrizen $\mathbf{G}$ und $\mathbf{U}$ des vorherigen Systems direkt, das heißt ohne Modifikationen, für das folgende System verwendet. Um diese Matrizen in den bestehenden Algorithmus integrieren zu können, muss außerdem noch die $s \times s$ Matrix $\mathbf{M}$, die das in jedem Schritt zu lösende Gleichungssystem repräsentiert und daher als $\mathbf{M} = \mathbf{P}^* \mathbf{G}$ definiert ist, einmalig mit der recycelten Matrix $\mathbf{G}$ initialisiert werden. Wird kein Recycling benutzt, so wird mit $\mathbf{M}$ als Einheitsmatrix gestartet, um die ersten s Vektoren in $\mathcal{G}_0$ zu erzeugen.

4.3.2 Flexible Schattenvektoren

Wie bereits in Abschnitt 3.2.5 erläutert wurde, ist noch unbekannt wie sich die Wahl der Schattenvektoren auf die Konvergenzeigenschaften der IDR-Methoden auswirkt. Theoretisch bestünde in Gestalt von $\mathbf{P}$ eine gute Möglichkeit, Informationen aus einem oder mehreren bereits gelösten Systemen zu integrieren, da keine besonderen Anforderungen an die Matrix gestellt werden. Es wären keine Modifikationen innerhalb der IDR-Methoden notwendig, um die Information aufzunehmen.
Es stellt sich die Frage, wie $\mathbf{P}$ überhaupt „besser“ gewählt werden kann. Im Hinblick auf das IDR-Prinzip sind die Schattenvektoren bereits optimal gewählt, wenn die Dimensionsreduktion in jedem $s + 1$-ten Schritt s beträgt. Es gibt noch keinen theoretischen Ansatz, wie man das Residuum schneller reduzieren könnte, obwohl es noch in einem höherdimensionalen Sonneveldraum liegt.

4.4 Ein Testproblem

In diesem Abschnitt wird die Herkunft der Sequenz von Gleichungssystemen erläutert, welche als Testproblem für die IDR-Recycling-Technik verwendet wird. Es handelt sich dabei um tridiagonale Toeplitzmatrizen, also um Matrizen der Form

$$\begin{bmatrix} d & d_1 & & & \\ d_{-1} & d & d_{+1} & & \\ & d_{-1} & \ddots & \ddots & \\ & & \ddots & \ddots & d_{+1} \\ & & & d_{-1} & d \end{bmatrix}.$$

Diese Matrizen treten bei der Lösung von eindimensionalen parameterabhängigen Differentialgleichungen aus dem Bereich der Konvektions-Diffusions-Probleme auf. Die Gleichung

$$-u''(x) + au'(x) = f,\ a > 0 \ \text{ auf dem Intervall } [0,1]\,,$$

mit den Randbedingungen $u(0) = 0 = u(1)$ kann mithilfe einer Diskretisierung numerisch gelöst werden. Dazu wird das Gebiet, in diesem Fall das Intervall, in $n+1$ äquidistante Gitterpunkte aufgeteilt, die den Abstand h zueinander haben. Mit der Taylorformel können dann die einzelnen Differentiationen abgeschätzt werden:

$$\begin{aligned} \text{Taylorformel:} \quad u(x+h) &= u(x) + h * u'(x) + \frac{h^2}{2}u''(x) + \dots \\ \Rightarrow \quad u'(x) &\approx \frac{u(x+h) - u(x-h)}{2h} \\ \Rightarrow \quad u''(x) &\approx \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} \end{aligned}$$

Dies ist die Formulierung mittels zentraler Differenzen und entsteht durch Addition zweier Vorwärtsdifferenzen mit h und $-h$. Nun kann die obige Differentialgleichung mit diesen Approximation umgeschrieben werden, um ein lineares Gleichungssystem aufzustellen:

$$\begin{aligned} i = 0, \dots n: \quad & -\tfrac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2} + a\tfrac{u(x_{i+1}) - u(x_{i-1})}{2h} && = f \\ i = 0, \dots n: \quad & -(1-c)u(x_{i+1}) + 2u(x_i) - (1+c)u(x_{i-1}) && = h^2 f \end{aligned}$$

$$\begin{bmatrix} 2 & -1+c & & & \\ -1-c & 2 & -1+c & & \\ & -1-c & \ddots & \ddots & \\ & & \ddots & \ddots & -1+c \\ & & & -1-c & 2 \end{bmatrix} \begin{bmatrix} u(x_0) \\ \vdots \\ u(x_n) \end{bmatrix} = h^2 f$$

Es gilt $n = \frac{1}{h}$ und es wurde $c = a\frac{h}{2}$ gesetzt, um h^2 ausmultiplizieren zu können. Die so definierte Matrix $\mathbf{A}(c)$ ist für $c = 0$ symmetrisch und wird mit größer werdendem c immer „unsymmetrischer“. Die Eigenwerte tridiagonaler Toeplitzmatrizen der Form 4.4 sind

$$k = 1, \ldots, n: \quad \lambda_k = \quad d + 2\sqrt{d_{+1}d_{-1}}\cos\left(\frac{k\pi}{n+1}\right).$$

In Kapitel 5.2 wird unter anderem anhand von Gleichungssequenzen dieser Art die Wirksamkeit der Recycling-Techniken für die IDR-Verfahren demonstriert.

4.5 Das SFE-Problem

Die Berliner Softwarefirma SFE GmbH stellt unter anderem Simulationsverfahren für die Innenraumakustik von Fahrzeugen her. Ein Teil dieser Simulationen besteht darin Eigenwerte einer parameterabhängigen Matrix zu berechnen, die als Schwingungen betrachtet werden können und daher minimert werden sollen. In Folge dieser Berechnungen ist es notwendig Gleichungssysteme zu lösen, welche sich jeweils vom vorhergehenden nur wenig unterscheiden. Es ist also angebracht, Krylov-Recycling-Strategien zu implementieren.
Es gibt verschiedene Möglichkeiten, die Matrizen für dieses Problem zu konstruieren, von denen hier eine kurz beschrieben werden soll. Es handelt sich dabei um drei Blockmatrizen, die wie folgt aufgebaut sind:

$$\begin{bmatrix} \mathbf{M}_s & \\ & \mathbf{M}_f \end{bmatrix} = \mathbf{M} \quad \text{(Massematrix)}$$

$$\begin{bmatrix} \mathbf{D}_s & \mathbf{A}_{sf}^T \\ \mathbf{A}_{sf} & \end{bmatrix} = \mathbf{D} \quad \text{(Dämpfung und Kopplung)}$$

$$\begin{bmatrix} \mathbf{K}_s & \\ & \mathbf{K}_f \end{bmatrix} = \mathbf{K} \quad \text{(Steifigkeitsmatrix)}$$

Die Indizes s und f bezeichnen dabei die Teilmatrizen für *structure* (Karosserie, Innenraum) und *fluid* (Luft). Für eine genauere Beschreibung der physikalischen Zusammenhänge sei [Koh04] empfohlen.
Es soll nun das Problem

$$\mathbf{P}(\lambda)\,\mathbf{x} = (\lambda^2\,\mathbf{M} + \lambda\,\mathbf{D} + \mathbf{K})\,\mathbf{x} = 0$$

gelöst werden, nachdem verschiedene Umformungen vorgenommen wurden:

- shift ($\lambda \to \lambda - \sigma$) und Taylor-Approximation:

$$\mathbf{P}(\sigma + \lambda - \sigma) \approx \mathbf{P}(\sigma) + \mathbf{P}'(\sigma)(\lambda - \sigma) + \mathbf{P}''(\sigma)(\lambda - \sigma)^2$$

- Linearisierung:

$$\left((\lambda-\sigma)\begin{bmatrix}\mathbf{M} & \\ & \mathbf{I}\end{bmatrix}+\begin{bmatrix}\mathbf{P}'(\sigma) & \mathbf{P}(\sigma)\\ -\mathbf{I} & \end{bmatrix}\right)\begin{bmatrix}(\lambda-\sigma)\,\mathbf{x}\\ \mathbf{x}\end{bmatrix} = 0$$

$$\Leftrightarrow \quad \left(\begin{bmatrix}\mathbf{M} & \\ & \mathbf{I}\end{bmatrix}+\tfrac{1}{\lambda-\sigma}\begin{bmatrix}\mathbf{P}'(\sigma) & \mathbf{P}(\sigma)\\ -\mathbf{I} & \end{bmatrix}\right)\begin{bmatrix}(\lambda-\sigma)\,\mathbf{x}\\ \mathbf{x}\end{bmatrix} = 0$$

Mit $\mu = \frac{1}{\lambda-\sigma}$ und $\hat{\mathbf{x}} = [(\lambda-\sigma)\,\mathbf{x},\ \mathbf{x}]^T$ ergibt sich das quadratische Eigenwertproblem

$$-\begin{bmatrix}\mathbf{P}'(\sigma) & \mathbf{P}(\sigma)\\ -\mathbf{I} & \end{bmatrix}^{-1}\begin{bmatrix}\mathbf{M} & \\ & \mathbf{I}\end{bmatrix}\hat{\mathbf{x}} = \mu\,\hat{\mathbf{x}}.$$

Die Inverse der Matrix $\begin{bmatrix}\mathbf{P}'(\sigma) & \mathbf{P}(\sigma)\\ -\mathbf{I} & \end{bmatrix}$ ist von σ abhängig und ändert sich daher von einem shift σ zum nächsten. Es ist nun die Aufgabe eines linearen Gleichungssystemlösers die für die Berechnung der Eigenwerte benötigten Matrix-Vektormultiplikationen bereitzustellen. Dabei ist es natürlich für die Lösung des Eigenwertproblems von entscheidender Wichtigkeit, dass diese Multiplikationen mit möglichst hoher Geschwindigkeit ausgeführt werden. Aufgrund der sehr großen Anzahl Unbekannter (bis zu einer Million) stellt die Bearbeitung dieses Problems eine große Herausforderung an die Hardware dar, wobei insbesondere die Größe des verfügbaren Arbeitsspeichers Schwierigkeiten bereitet. Da sich die Matrizen für kleine Änderungen des Parameters σ nur wenig voneinadner unterscheiden, bietet sich die Verwendung eines Gleichungssytemlösers an, der Recycling-Strategien implementiert.
In Kapitel 5 wird ein vereinfachtes Modell der Dimension $n = 10746$ des SFE-Problems betrachtet, bei dem alle Matrizen außer $\mathbf{M}_s$ und $\mathbf{K}_s$ gleich Null sind. Dies beschleunigt die Berechnungen, so dass auch häufige Testläufe durchgeführt werden können.

Kapitel 5

Numerische Ergebnisse

In diesem Kapitel werden die Implementierungen der vorgestellten Verfahren untereinander und mit anderen Methoden verglichen. Alle Berechnungen wurden mit MATLAB in der Version 2007b durchgeführt und können mithilfe der beigefügten Programme überprüft und um weitere Ergebnisse erweitert werden. Es handelt sich dabei im Wesentlichen um Erweiterungen sowohl der einfachen IDR-Methode aus [SonG08], als auch der IDR-Methode mit Biorthogonalisierung, welche in [GijS08] beschrieben und unter [Gij08] zu finden ist. Zu den beschriebenen Recycling-Strategien wurde außerdem die Verwendung von Vorkonditionierern implementiert. Beide Programme lassen sich über ein Hauptprogramm ansteuern, so dass verschiedene Testläufe bequem durchgeführt werden können. Sofern nicht anders angegeben, wurde für alle Systeme die gleiche orthogonalisierte komplexe Matrix $\mathbf{P}$ verwendet, um die Ergebnisse vergleichbarer zu machen.

Notation

In diesem Kapitel werden für die beiden vorgestellten IDR-Verfahren die Bezeichnungen „IDR(s)simple“, beziehungsweise „IDR(s)biortho“ verwendet, da deren Implementierungen ebenfalls so benannt wurden.

5.1 Vergleiche mit anderen Lösern

In diesem Abschnitt werden die vorgestellten IDR-Varianten mit den in MATLAB bereits implementierten Gleichungssystemlösern GMRES und BI-CGSTAB (siehe [Vor92]) und dem BiCGstab(l)-Verfahren aus [SleVF94] verglichen. Abbildung 5.1 zeigt die Ergebnisse für die Lösung eines Systems mit der diagonaldominanten, schlecht konditionierten, tridiagonalen Matrix „dorr“ aus der MATLAB Galerie (Dimension $n = 1000$, $\theta = 0.01$). Es ist gut zu erkennen, dass die IDR-Verfahren im Hinblick auf die Iterationsanzahl Schwierigkeiten bei der Lösung des Systems haben und die Konvergenzgeschwindigkeit stark von der Wahl von s abhängt. Die traditionellen Lösungsverfahren GMRES und BI-CGSTAB konvergieren zwar mit weniger Iterationen, allerdings ist die benötigte Rechenzeit verglichen mit den IDR-Verfahren ungefähr zehnmal so lang. Weiterhin ist deutlich zu sehen, dass durch mehr Schattenvektoren die Konvergenzgeschwindigkeit gemessen an der Iterationsanzahl gesteigert werden kann, damit allerdings eine längere Berechnungszeit einhergeht. Im Fall von IDR(20)simple liegt die vergleichsweise hohe Anzahl an Iterationen an numerischen Ungenauigkeiten bei der Berechnung der Zwischenresiduen, die mit größer werdendem Parameter s an Bedeutung gewinnen. BiCGstab(10) ist im Hinblick auf die Iterationsanzahl den anderen Verfahren deutlich überlegen.

Verfahren	Zeit(sec)	Iterationen
GMRES	8.757210	505
BICGSTAB	8.583279	644.5
IDR(4)simple	0.827593	2360
IDR(10)simple	0.683947	1603
IDR(20)simple	1.058202	1905
IDR(4)biortho	0.739793	1962
IDR(10)biortho	0.714357	1672
IDR(20)biortho	1.215313	1535
BiCGstab(10)	0.842231	65

Abbildung 5.1: Vergleich der benötigten Rechenzeit verschiedener Verfahren

5.1.1 Schaltkreisdesign

In diesem Abschnitt werden Gleichungssysteme gelöst, die aus dem Design elektronischer Schaltkreise stammen. Die Matrizen sowie die korrespondierenden rechten Seiten, sind aus der Matrix-Market-Kollektion und daher frei zugänglich. Als Lösungsverfahren wurden für alle Systeme beide IDR-Varianten mit verschiedenen Werten für s, sowie das GMRES- und das BI-CGSTAB-Verfahren verwendet.
Das erste System bezieht sich auf einen 20-Bit Addierer. Die Matrix ist reell und unsymmetrisch von der Dimension 2395×2395 mit 17319 Einträgen ungleich Null. Die Konditionszahl ist mit $1.76 \cdot 10^4$ vergleichsweise groß. Die Matrix ist also schlecht konditioniert. Abbildung 5.2 zeigt deutlich die Überlegenheit der IDR-Methoden im Vergleich zum BI-CGSTAB-Verfahren, wobei auch zu sehen ist, dass größere Werte für s einen positiven Einfluss auf die benötigte Anzahl an Iterationen haben. Außerdem ist zu erkennen, dass die beiden IDR-Varianten für gleiche Werte von s einen sehr ähnlichen Konvergenzverlauf haben, was durch die eindeutig bestimmten Primärresiduen bedingt ist. Das (volle) GMRES-Verfahren hat zwar eindeutig die geringste Iterationanzahl, jedoch ist hier der Speicherbedarf sehr viel größer als bei bei den übrigen Methoden.

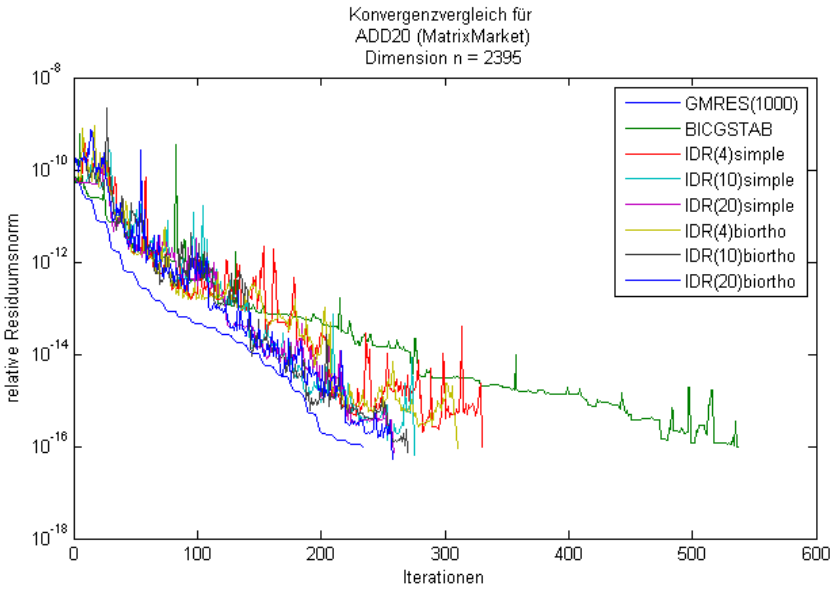

Abbildung 5.2: Konvergenzvergleich für ADD20

Auf der nächsten Grafik (Abbildung 5.3) ist der gleiche Versuch mit einer Matrix, die einen 32-Bit Addierer repräsentiert, dargestellt. Es handelt sich hierbei um eine reelle, unsymmetrische Matrix der Dimension 4960×4960

mit 23884 Einträgen ungleich Null. Diese Matrix ist also dünner besetzt als im vorherigen Beispiel und mit einer Konditionszahl von $2.14 \cdot 10^2$ besser konditioniert. Dieser Umstand drückt sich in der allgemein geringeren Iterationsanzahl aus. Obwohl sich hier ein qualitativ ähnlicher Konvergenzverlauf ergibt, ist aufgrund der relativ guten Kondition kein großer Unterschied zwischen den einzelnen Methoden festzustellen.

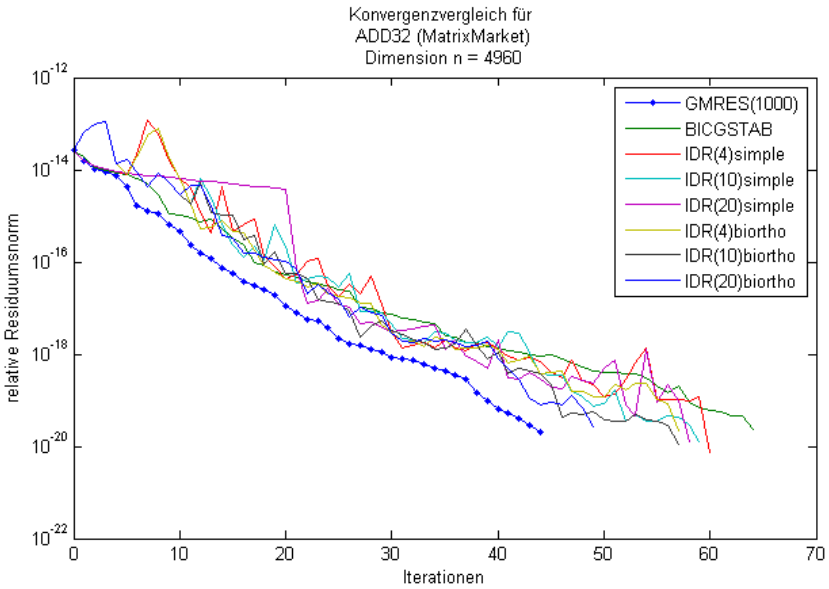

Abbildung 5.3: Konvergenzvergleich für ADD32

Als letztes Beispiel dieser Klasse von Gleichungssytemen, wird eine Repräsentation eines Speicherschaltkreises verwendet. Die Systemmatrix ist wieder reell und unsymmetrisch und hat die Dimension 17758×17758 mit 126150 Einträgen ungleich Null. Es handelt sich also um eine weitaus größere Matrix, die außerdem mit einer Konditionszahl von $2.67 \cdot 10^5$ schlecht konditioniert ist. In der Abbildung 5.4 zeigt sich daher, dass die IDR-Methoden für solche Systeme besser geeignet sind als die beiden traditionellen Krylovlöser. Jedoch wirkt sich dieser Effekt erst bei einer Wahl von $s > 4$ aus.

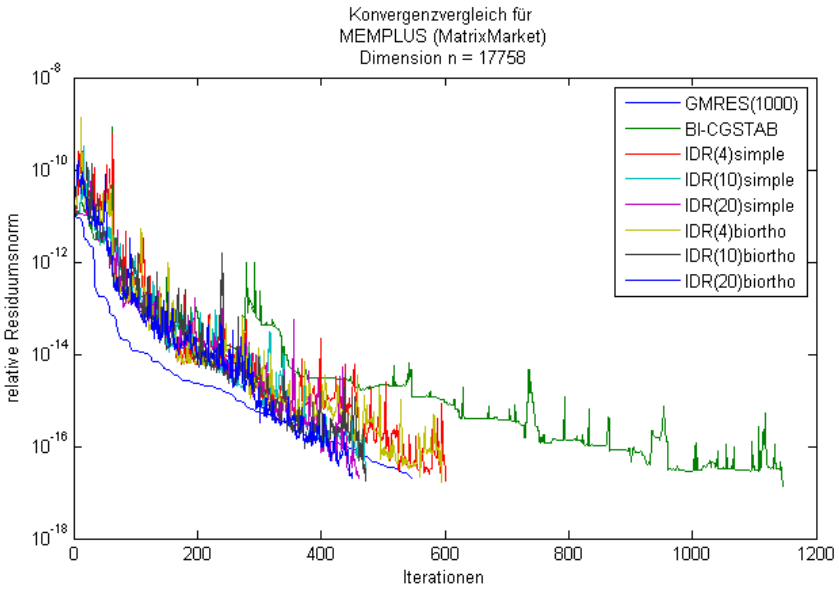

Abbildung 5.4: Konvergenzvergleich für MEMPLUS

5.2 Anwendung auf Toeplitzmatrizen

Dieser Abschnitt zeigt die Effektivität der IDR-Methoden und insbesondere deren Erweiterungen um das Krylov-Recycling auf. Als Testsystem wurden hier Toeplitzmatrizen der Form 4.4 verwendet, wie sie in Kapitel 4.4 vorgestellt wurden.
Zuerst sind in Abbildung 5.5 die (reellen) Lösungsvektoren einer gesamten Sequenz von 20 Gleichungssystemen dargestellt, um die Auswirkung der Veränderungen der Matrizen auf die Lösung der einzelnen Systeme zu veranschaulichen. Hierbei wurde mit einer Toeplitzmatrix mit Parameter $c = 10^{-4}$ als Startsystem und einer graduellen Veränderung von c um eine Addition von ebenfalls 10^{-4} gearbeitet. Die rechte Seite, also der Vektor b, wurde bei allen Systemen konstant auf 1 gesetzt.

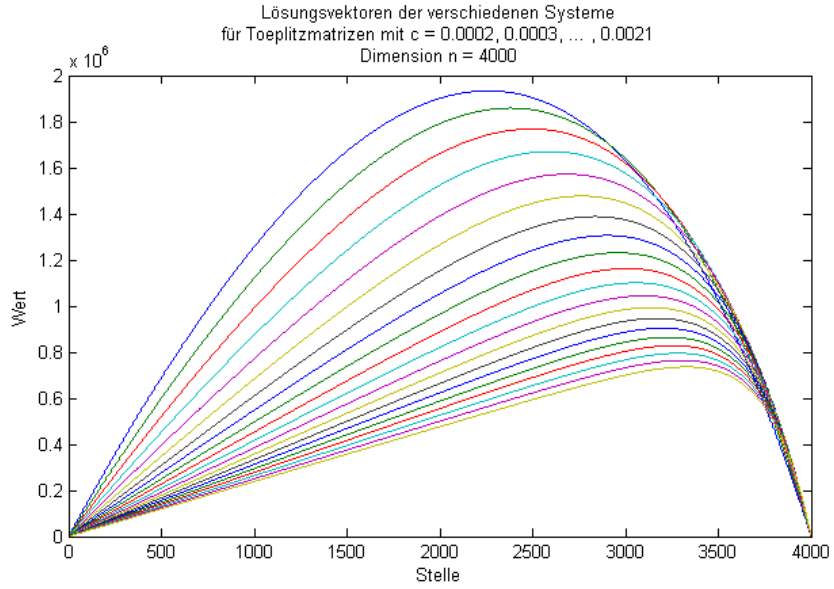

Abbildung 5.5: Lösungsvektoren einer Toeplitzsequenz ($n = 4000$)

Die Wirksamkeit der Recycling-Technik beruht mitunter darauf, dass die Schattenvektoren für die gesamte Folge von Gleichungssystemen konstant bleibt. In Abbildung 5.6 ist deutlich zu sehen, dass durch eine neue Wahl von $\mathbf{P}$ die Konvergenz des Verfahrens mit Recycling schlechter als ausfällt als beim Verfahren ohne die Benutzung von Recycling.

In Abbildung 5.7 sind die Konvergenzverläufe der einzelnen Systeme dargestellt, wobei angemerkt werden muss, dass hier keine Vorkonditionierung angewandt wurde und deshalb nur mit einer Dimension von 200 gearbeitet wurde. In der Grafik ist die in Kapitel 3.1 bereits prognostizierte reguläre Dimensionsreduktion von n/s zu erkennen. So ergibt sich eine ungefähre Laufzeit von $(s+1)\cdot n/s = n+n/s$ Iterationen und damit in diesem Fall mit $s = 10$ eine Iterationsanzahl von 220, was durch die numerischen Resultate bestätigt wird. Die Verbesserung des Startresiduums und damit einhergehend auch eine leichte Verbesserung der Gesamtkonvergenz für alle Systeme nach dem Ersten, ist in der Übergabe der Lösung als Startwert für das nächste System begründet.

Durch die Implementierung der in Kapitel 4.3.1 vorgestellten Möglichkeiten des Krylov-Recyclings kann diese Konvergenzgeschwindigkeit jedoch stark verbessert werden. So ist in Abbildung 5.8 zu sehen, dass im Vergleich mit Abbildung 5.7 die gleiche Sequenz sehr viel schneller gelöst werden kann. Auch hier wurde noch nicht mit einem Vorkonditionierer gearbeitet, so dass

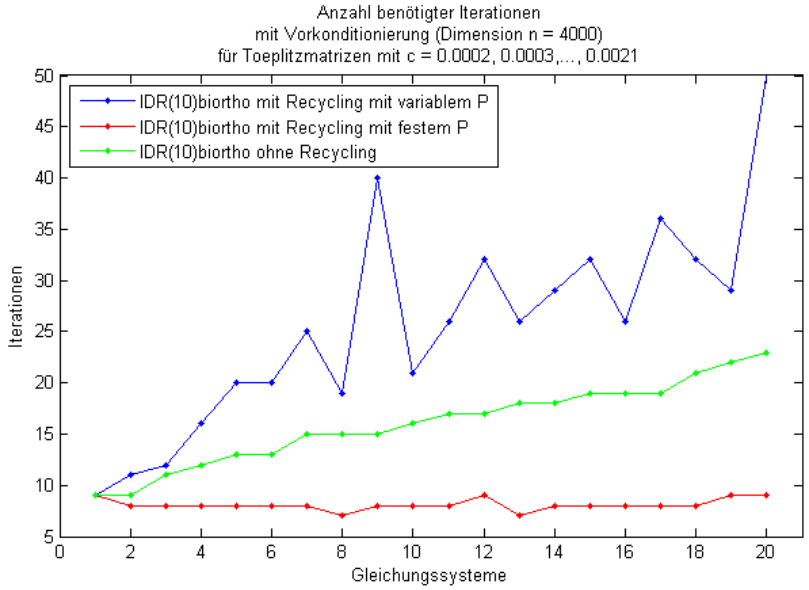

Abbildung 5.6: Auswirkung einer variablen Matrix $\mathbf{P}$

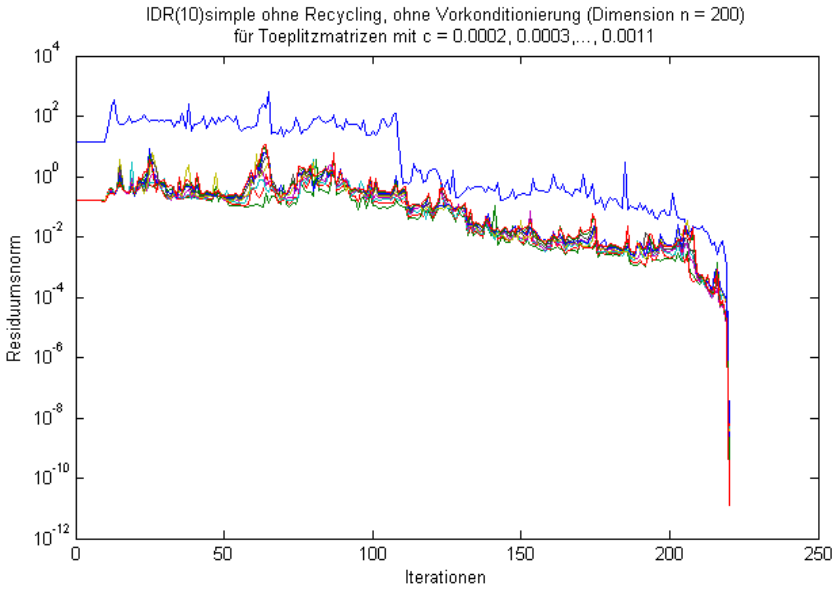

Abbildung 5.7: Konvergenzverlauf der einfachen IDR-Methode

alle Verbesserungen allein auf das Recycling des vorherigen Lösungsraums zurückzuführen sind.

Auch die IDR-Variante mit Biorthogonalisierung profitiert von der Imple-

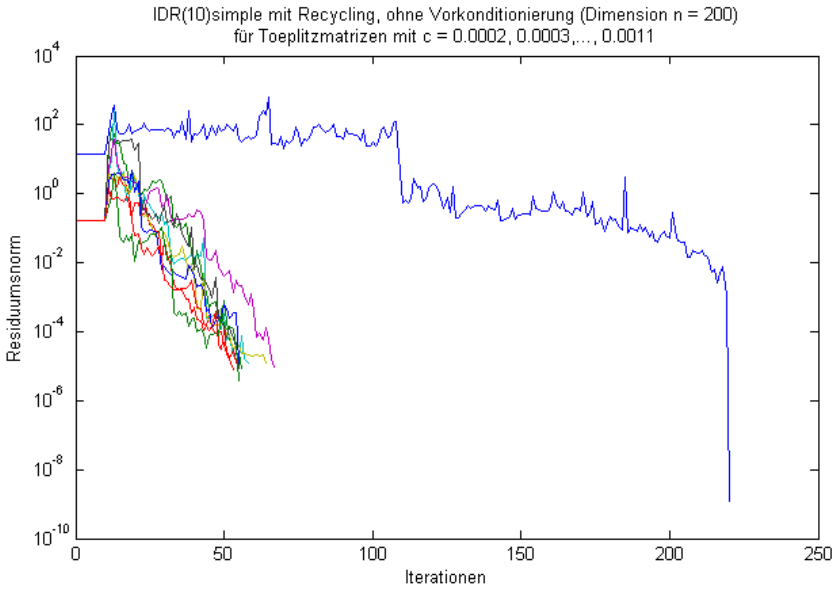

Abbildung 5.8: Konvergenzverlauf der einfachen IDR-Methode mit Recycling

mentierung des Recyclings des vorherigen Lösungsraums, wie in Abbildung 5.9 zu sehen ist. Allerdings hat das IDR-Verfahren mit Biorthogonalisierung den Vorteil einer besseren numerischen Stabilität - insbesondere für größere Werte von s. Wie sich später zeigen wird, verhält sich diese Variante bei großen Systemen besser als die einfache IDR-Methode.

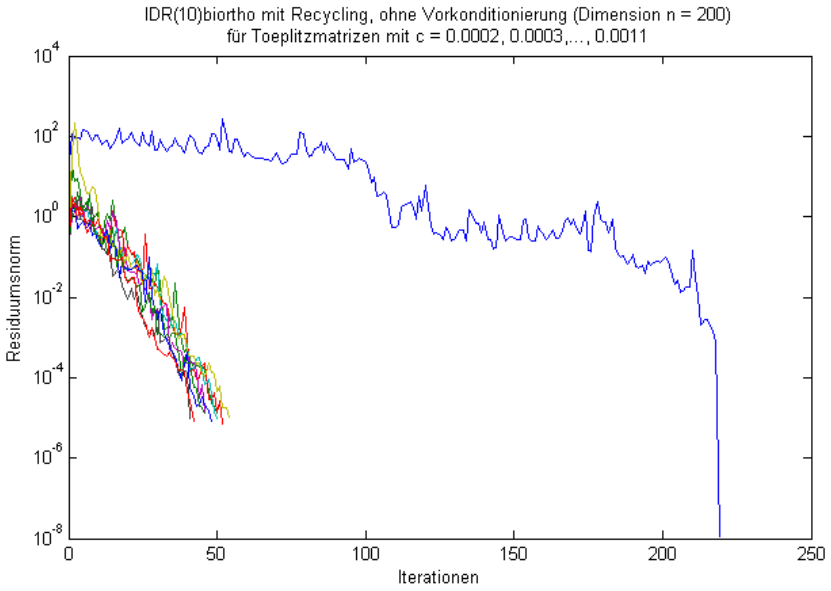

Abbildung 5.9: Konvergenzverlauf der IDR-Methode mit Biorthogonalisierung und Recycling

5.2.1 Anwendung eines Vorkonditionierers

In diesem Abschnitt wird die gleiche Art von Testsequenzen in Form von Toeplitzmatrizen wie zuvor betrachtet. Allerdings wird hier zusätzlich zum Krylov-Recycling auch eine Vorkonditionierung angewandt. Dazu wird die LU-Zerlegung der Matrix des ersten Systems berechnet und mit dieser von links vorkonditioniert. Eine LU-Zerlegung ist die Faktorisierung einer Matrix in eine untere und eine obere Dreiecksmatrix:

$$\mathbf{A} = \mathbf{L}\,\mathbf{U} = \begin{bmatrix} l_{1,1} & & \\ \vdots & \ddots & \\ l_{n,1} & \dots & l_{n,n} \end{bmatrix} \begin{bmatrix} r_{1,1} & \dots & r_{1,n} \\ & \ddots & \vdots \\ & & r_{n,n} \end{bmatrix}.$$

Das Inverse dieses Vorkonditionierers wird dann (in diesem Fall von links) mit dem zu lösenden Gleichungssystem multipliziert

$$\mathbf{U}^{-1}\,\mathbf{L}^{-1}\,\mathbf{A}\,\mathbf{x} = \mathbf{U}^{-1}\,\mathbf{L}^{-1}\,\mathbf{b},$$

wobei die Multiplikation durch Vorwärts-, beziehungsweise Rückwärtseinsetzen durchgeführt wird.
Da der Vorkonditionierer auf Basis der ersten Matrix aufgebaut ist und jedes System mit dem gleichen Vorkonditionierer gestartet wird, nimmt die Wirkung der Vorkonditionierung mit fortschreitender Veränderung innerhalb der Sequenz stetig ab. In Abbildung 5.10 ist dies anhand der von System zu

System steigenden Iterationsanzahl der beiden Verfahren ohne Verwendung von Recycling gut zu erkennen. Durch das Recycling des Lösungsraums wird dieser Effekt aufgehoben, so dass die Iterationsanzahl auch bei späteren Systemen der Sequenz gleich bleibt.

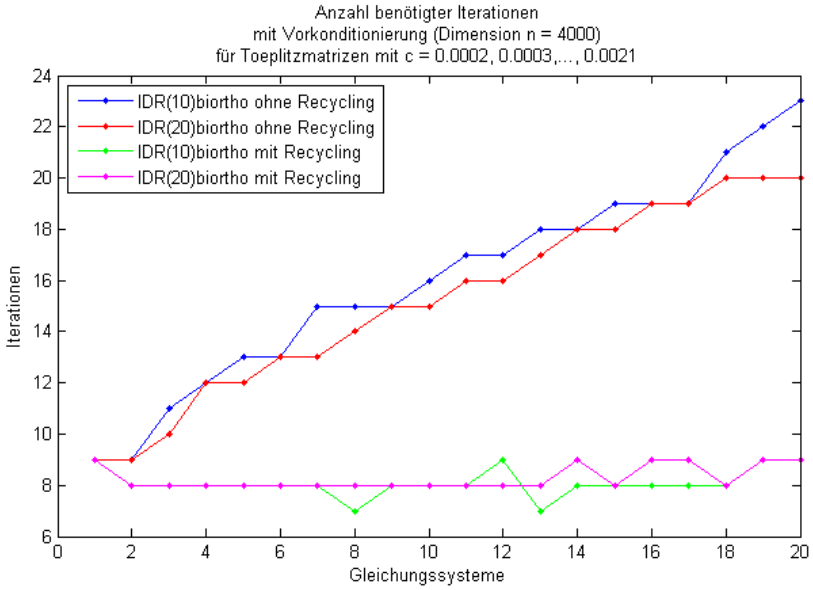

Abbildung 5.10: Iterationsanzahl der IDR-Methode mit Biorthogonalisierung

5.3 Anwendung auf das SFE-Problem

In Abbildung 5.11 wird die Lösung des sogenannten „Box“-Modells berechnet. Es sollen Eigenwerte im Bereich von $100-500$ auf der imaginären Achse gefunden werden, was für den Gleichungssystemlöser bedeutet, eine Sequenz zu lösen, bei der sich der Parameter σ in einem Bereich zwischen $100i$ und $500i$ bewegt. Es wurde hier eine Schrittweite von 10 verwendet, so dass insgesamt 41 Gleichungssysteme gelöst werden mussten. Der Vorkonditionierer wurde basierend auf dem System mit $\sigma = 90i$ berechnet. Man hätte auch den Vorkonditionierer auf dem ersten System basieren lassen können. Dies hätte die Lösung dieses Systems jedoch zu sehr vereinfacht und damit die Testergebnisse schlechter vergleichbar gemacht. Bei einer zu kleinen Wahl von s kann es passieren, dass die Verwendung der hier beschriebenen Recycling-Technik die Konvergenz des Verfahren sogar verschlechtert. Der Grund dafür ist, dass der recycelte Raum zu klein ist und deshalb nicht gut als Startraum für das folgende System funktioniert.
In der Grafik ist zu erkennen, dass die IDR-Verfahren nit Recycling und ei-

nem Wert für s größer als 4 eine, auf die gesamte Gleichungsfolge gesehen, relativ konstante Iterationsanzahl benötigen. Bei den klassischen Krylovraum-Verfahren und den IDR-Verfahren ohne Recycling ist dagegen zu beobachten, dass die Iterationsanzahl mit wachsender Entfernung des Systems zum Vorkonditionierer zunimmt. Die Implementierung der Recycling-Strategie kann diesen Effekt also ausgleichen.

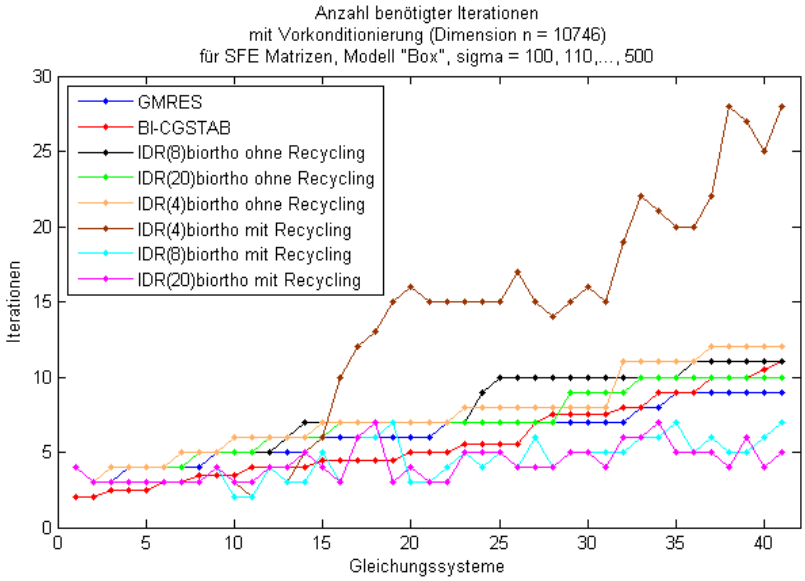

Abbildung 5.11: Vergleich mehrerer Verfahren für das „Box“-Modell

Kapitel 6

Zusammenfassung und Ausblick

In dieser Arbeit wurde die neuartige Klasse der IDR-Verfahren vorgestellt, die trotz ihrer einfachen Konzeption viel Raum für Modifikationen bietet. Anhand zahlreicher numerischer Experimente konnten die erwarteten guten Resultate der IDR-Verfahren belegt werden. Außerdem wurden die Gemeinsamkeiten und Unterschiede der beiden IDR-Verfahren mit und ohne Biorthogonalisierung analysiert und begründet. Es wurde dargestellt, dass die Biorthogonalisierung die wünschenwerten Eigenschaften der einfachen IDR-Methode erhält und gleichzeitig für eine verbesserte Stabilität des Verfahrens sorgt.
Während des Verfassens dieser Arbeit wurde das IDR-Prinzip in [SleG09] und [TanS09] weiter verallgemeinert, indem die Integration höhergradiger Stabilisierungspolynome realisiert wurde. Obwohl in diesen Artikeln die neuen Verfahren als besonders effektiv dargestellt werden, müssen deren Implementierungen noch umfasssend numerisch getestet werden. Zudem sollte die zugrunde liegende Theorie genau analysiert werden.

Obwohl anzunehmen ist, dass noch effizientere Implementierungen und Modifikationen der IDR-Verfahren möglich sind, stellen die derzeit verwendeten IDR-Methoden eine sinnvolle Ergänzung zu den bestehenden Gleichungssystemlösern dar. Sie haben die Eigenschaft mit vergleichsweise wenigen Iterationen in kurzer Zeit auch große und schlecht konditionierte Gleichungssysteme lösen zu können. Dennoch lassen sie sich, wie anhand der numerischen Beispiele in Kapitel 5 dargestellt, nicht uneingeschränkt auf alle Arten von Gleichungssystemen anwenden.
Im Rahmen dieser Arbeit konnten Recycling-Strategien erfolgreich in bestehende IDR-Varianten integriert werden. Dadurch konnte der Aufwand bei der Lösung von Folgen ähnlicher Gleichungssysteme stark reduziert werden. Bei der Verwendung von Recycling-Strategien sollte allerdings immer darauf geachtet werden, dass die Anzahl der Schattenvektoren und damit auch die

Anzahl der recycelten Vektoren nicht zu klein gewählt wird.
Es gelang also, die beiden relativ neuen Techniken des Krylov-Recyclings und der IDR-Methoden miteinander zu kombinieren, um so ein effektives Lösungsverfahren für Folgen von Gleichungssystemen zu erhalten.

Insbesondere bei der Implementierung von Recycling-Strategien sollte die Möglichkeit der Verwendung von flexiblen Schattenvektoren weiter untersucht werden. Im Gegensatz zur Nutzung von Zufallsvektoren sollten diese Freiheitsgrade genutzt werden, um die Effektivität der IDR-Methoden weiter zu steigern.
Sollte es gelingen, eine Recycling-Strategie basierend auf der Wahl der schattenvektoren umzusetzen, so ließe sich dies auch auf die verallgemeinerten IDR-Varianten mit höhergradigen Stabilisierungspolynomen anwenden.

Literaturverzeichnis

[EieES00] Michael Eiermann, Oliver G. Ernst, and Olaf Schneider. Analysis of acceleration strategies for restarted minimal residual methods. Journal of Computational and Applied Mathematics, 123:261-292, 2000.

[FabM84] Vance Faber and Thomas Manteuffel. Necessary and sufficient conditions for the existence of a conjugate gradient method. *SIAM J. Numer. Anal.*, 21(2):352–362, 1984.

[Gij08] Martin B. van Gijzen. MATLAB Code von IDR(s)biortho . http://ta.twi.tudelft.nl/nw/users/gijzen/IDR.html, 2008.

[GijS08] Martin B. van Gijzen and Peter Sonneveld. An elegant IDR(s) variant that efficiently exploits bi-orthogonality properties. Technical report, Delft University of Technology, Reports of the Department of Applied Mathematical Analysis, Report 08-21, 2008.

[Gut09] Martin H. Gutknecht. IDR explained. Technical report, Seminar for Applied Mathematics, ETH Zürich, 2009.

[KilS06] Misha E. Kilmer and Eric de Sturler. Recycling subspace information for diffuse optical tomography. *SIAM J. Sci. Comput.*, 27(6):2140–2166 (electronic), 2006.

[Koh04] Peter Kohnke. *ANSYS, Inc. Theory Reference.* SAS IP, Inc, 12th edition, 2004.

[Mor02] Ronald B. Morgan. GMRES with deflated restarting. *SIAM J. Sci. Comput.*, 24(1):20–37 (electronic), 2002.

[ParSMJM06] Michael L. Parks, Eric de Sturler, Greg Mackey, Duane D. Johnson, and Spandan Maiti. Recycling Krylov subspaces for sequences of linear systems. *SIAM J. Sci. Comput.*, 28(5):1651–1674 (electronic), 2006.

[Saa03] Yousef Saad. *Iterative methods for sparse linear systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, second edition, 2003.

[SleG09] Gerard L. G. Sleijpen and Martin B. van Gijzen. Exploiting BiCGstab(ℓ) strategies to induce dimension reduction. Technical report, Delft University of Technology, Reports of the Department of Applied Mathematical Analysis, Report 09-02, 2009.

[SleSG08] Gerard L. G. Sleijpen, Peter Sonneveld, and Martin B. van Gijzen. Bi-CGSTAB as an induced dimension reduction method. Technical report, Delft University of Technology, Reports of the Department of Applied Mathematical Analysis, Report 08-07, 2008.

[SleVF94] G. L. G. Sleijpen, H. A. van der Vorst, and D. R. Fokkema. bicgstab(l) and other hybrid Bi-CG methods. *Numerical Algorithms*, 7(1):75–109, June 1994.

[SonG08] Peter Sonneveld and Martin B. van Gijzen. IDR(s): a family of simple and fast algorithms for solving large nonsymmetric systems of linear equations. *SIAM J. Sci. Comput.*, 31(2):1035–1062, 2008/09.

[Stu99] Eric De Sturler. Truncation strategies for optimal Krylov subspace methods. *SIAM J. Numer. Anal.*, 36(3):864–889 (electronic), 1999.

[TanS09] Masaaki Tanio and Masaaki Sugihara. GBi-CGSTAB(s, l): IDR(s) with Higher-Order Stabilization Polynomials. Technical report, Department of Mathematical Informatics Graduate School of Information Science and Technology University of Tokyo, 2009.

[Vor03] Henk A. van der Vorst. *Iterative Krylov methods for large linear systems*, volume 13 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, UK, 2003.

[Vor92] H. A. van der Vorst. BI-CGSTAB: A fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems. *SIAM J. Scient. Stat. Comput.*, 13(2):631–644, March 1992.