Block-Krylovraumverfahren

BACHELORARBEIT
zur Lehrveranstaltung *Numerische Mathematik I*

Institut für Mathematik und wissenschaftliches Rechnen
Karl-Franzens-Universität Graz

ELIAS WINDISCH

Betreuer: Ass.-Prof. Dr. rer.nat. Dipl.-Math. Christian Clason

Zusammenfassung

In dieser Arbeit werden zwei iterative Lösungsverfahren für ein lineares Gleichungssystem mit mehreren rechten Seiten dargestellt. Die beiden Verfahren sind eine Verallgemeinerung des CG-bzw. GMRES-Verfahrens. Durch die Verallgemeinerung entstehen neue Probleme für die Lösungsansätze beschrieben werden. Die Anwendung der beiden, in dieser Arbeit untersuchten, Verfahren ist vor allem beim Lösen von Gleichungssystemen mit mehreren rechten Seiten, beschrieben durch eine dünn besetzte Matrix, relevant. Numerische Beispiele belegen, dass die dargestellten Verfahren in bestimmten Situationen schneller die Lösung des Gleichungssystems mit mehreren rechten Seiten finden als das sukzessive Lösen der einzelnen Gleichungssysteme durch das CG-bzw. GMRES-Verfahrens.

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht. Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen. Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Graz, den 8. August 2012

Inhaltsverzeichnis

1 Einleitung

In vielen Anwendungsbereichen der Mathematik treten Problemstellungen auf, welche durch dünn besetzte Matrizen mit hoher Dimension n und mehreren rechten Seiten modelliert werden können. In dieser Arbeit betrachten wir zwei Verfahren, die das Gleichungssystem, das durch eine dünn besetzte Matrix mit mehreren rechten Seiten beschrieben ist, löst. Beide Verfahren berechnen iterativ ein X_* für das gilt:

$$AX_* = B \quad \text{mit } A \in \mathbb{R}^{n\times n} \text{ und } B \in \mathbb{R}^{n\times m}. \tag{1.1}$$

Diese Verfahren sind eine Verallgemeinerung des CG- und GMRES-Verfahrens. Wir ordnen die beiden, in dieser Arbeit betrachteten Verfahren für mehrere rechte Seiten den Block-Krylovraumverfahren zu.

Block-Krylovraumverfahren haben den Vorteil gegenüber gewöhnlichen Krylovraumverfahren, dass der Suchraum der Lösung in jeder Iteration meist viel größer ist und dadurch die Lösung meist nach weniger Iterationen gefunden werden kann. Die Nachteile der Block-Krylovraumverfahren sind:

- Das Residuum kann mitten im Algorithmus linear abhängig werden und dadurch das weitere Berechnen der Lösung erschweren.
- Das Lösen der Hilfsprobleme in jedem Iterationsschritt wird schwieriger.
- Block-Krylovraumverfahren benötigen meist mehr Speichervolumen.

Im nächsten Kapitel in dieser Arbeit, wird der verallgemeinerte CG-Algorithmus analysiert, indem zuerst ein gewöhnlicher CG-Algorithmus nur minimal erweitert wird. Dieser verallgemeinerte Algorithmus wird Block-CG-Verfahren genannt. Die neuen Probleme des Block-CG-Verfahren zum Beispiel die mögliche lineare Abhängigkeit der Residuen werden im Unterkapitel 2.3 durch Reorthogonalisierung und Deflation gelöst. Da das Block-CG-Verfahren im Allgemeinen nur bei symmetrisch positiv definiten Matrizen konvergiert, wird in Kapitel 3 eine Verallgemeinerung des GMRES-Verfahrens dargestellt, das wir nunmehr Block-GMRES-Verfahren nennen. Für diese Darstellung wird ein Block-Arnoldi-Prozess verwendet, welcher im Unterkapitel 3.1 hergeleitet wird. Lösungsansätze für Probleme die beim Block-GMRES-Verfahren zum Vorschein kommen werden im Unterkapitel 3.4 diskutiert.

Bei beiden Verfahren wird im Unterkapitel 2.2 beziehungsweise im Unterkapitel 3.3 die Konvergenz analysiert. Es wird gezeigt, dass der Suchraum der Block-Verfahren oftmals viel größer ist als bei den gewöhnlichen Verfahren. Am Ende dieser Arbeit im Kapitel 4 vergleichen wir die Block-Krylovraumverfahren mit den gewöhnlichen Krylovraumverfahren anhand der Anwendung auf verschiedene Matrizen. Wir simulieren am Rechner in welchen Problemstellungen welche Verfahren bei der Lösung von Gleichungssystemen von Vorteil sind.

Der Vollständigkeit halber sei hinzugefügt, dass wir nur Block-Krylovraumverfahren und gewöhnliche Krylovraumverfahren vergleichen. So werden in dieser Arbeit zum Beispiel direkte Verfahren zum Lösen linearer Gleichungssysteme nicht berücksichtigt. Diese werden vor allem dann angewendet, wenn das n der Gleichung (1.1) klein ist.

2 Block-CG-Verfahren

Das erste der beiden Verfahren, das untersucht wird, ist das Block-CG-Verfahren. Dieses Verfahren findet iterativ eine Lösung des Problems (1.1). Wie im gewöhnlichen CG-Verfahren wird auch hier für die Konvergenz benötigt, dass A symmetrisch und positiv definit ist. Den Algorithmus leiten wir vom gewöhnlichen CG-Verfahren ab, ohne Block-Krylovräume a priori zu definieren.

Im Unterkapitel 2.2 sehen wir, dass das Block-CG-Verfahren ein Block- Krylovraumverfahren ist, indem der dort definierte Block-Krylovraum der Suchraum der Lösung in jedem Iterationsschritt ist.

2.1 Algorithmus

Lemma 2.1. *Sei* $A \in \mathbb{R}^{n\times n}$ *symmetrisch positiv definit und sei* $P \in \mathbb{R}^{n\times m}$ *mit* $m \leqslant n$. *Falls* P *vollen Rang hat ist* P^TAP *symmetrisch positiv definit.*

Beweis. Sei $0 \neq y \in \mathbb{R}^m$ dann gilt $Py \neq 0$ und dadurch $\langle P^TAPy, y\rangle = \langle APy, Py\rangle > 0$. □

Satz 2.2. *Sei* $A \in \mathbb{R}^{n\times n}$ *symmetrisch positiv definit und sei* $B \in \mathbb{R}^{n\times m}$ *mit* $m \leqslant n$. *Sei* $P_j \in \mathbb{R}^{n\times m}$ *mit* $j \in \{0,\dots,n-1\}$ *und* ${P_j}^TAP_i = 0$ *für alle* $j \neq i$, *wobei alle* P_j *vollen Spaltenrang haben. Sei* $X_0 \in \mathbb{R}^{n\times m}$ *beliebig und* X_k *rekursiv durch*

$$X_k := X_{k-1} + P_{k-1}\alpha_k$$

bestimmt mit

$$\alpha_k := (P_{k-1}^TAP_{k-1})^{-1}P_{k-1}^TR_0 \qquad \textit{und} \qquad R_k := B - AX_k.$$

Dann folgt:

$$P_j^TR_k = 0 \qquad \textit{für} \qquad j \in \{0,\dots,k-1\}.$$

Zusätzlich ist nach höchstens n *Iterationen die Lösung* X_* *mit* $AX_* = B$ *erreicht.*

Bemerkung 2.3. *Die Matrizen* α_k *sind wohldefiniert, da alle* P_j *vollen Spaltenrang haben.*

Beweis. Für den Beweis vom Satz betrachten wir:

$$\begin{aligned}
P_{k-1}^TR_{k-1} &= P_{k-1}^T(B - AX_{k-1}) = P_{k-1}^TB - P_{k-1}^TAX_{k-1}\\
&= P_{k-1}^TB - P_{k-1}^TA(X_0 + \sum_{i=0}^{k-2} P_i\alpha_{i+1})\\
&= P_{k-1}^TB - P_{k-1}^TAX_0 + \sum_{i=0}^{k-2}(P_{k-1}^TAP_i\alpha_{i+1})\\
&= P_{k-1}^T(B - AX_0) = P_{k-1}^TR_0.
\end{aligned}$$

Aufgrund der Definition von α_k und obiger Gleichheit gilt:

$$\begin{aligned} 0 &= P_{k-1}^T A P_{k-1} \alpha_k - P_{k-1}^T R_{k-1} = P_{k-1}^T (A P_{k-1} \alpha_k - R_{k-1}) \\ &= P_{k-1}^T (A(X_{k-1} + P_{k-1}\alpha_k) - B) = -P_{k-1}^T R_k. \end{aligned}$$

Weiters gilt für $i \neq j$:

$$P_j^T (R_{i+1} - R_i) = P_j^T (AX_i - AX_i + AP_i\alpha_{i+1}) = P_j^T A P_i \alpha_{i+1} = 0.$$

Sei $j < k-1$. Aus den vorherigen beiden Gleichungen folgt mithilfe der Teleskopreihe:

$$P_j^T R_k = P_j^T R_k - P_j^T R_{j+1} = \sum_{i=j+1}^{k-1} P_j^T (R_{i+1} - R_i) = 0.$$

Um die Konvergenz zu beweisen verwenden wir, dass $P_j^T R_n = 0$ für alle $j \in \{0, \dots, n-1\}$. Insbesondere heißt das: $\left\langle P_j^{(i)}, R_n^{(i)} \right\rangle = 0$ für alle $i \in \{1, \dots, m\}$, wobei $R_n^{(i)}$ die i-te Spalte von R_n ist. Da alle P_j vollen Spaltenrang haben sind die A-orthogonalen $P_j^{(i)}$ mit $j \in \{0, \dots, n-1\}$ und i beliebig aber fix, eine Basis des $\mathbb{R}^n$, das heißt die lineare Hülle von $R_n^{(i)}$ ist orthogonal zu $\mathbb{R}^n$ woraus folgt: $B^{(i)} - AX_n^{(i)} = R_n^{(i)} = 0$.

$AX_n^{(i)} = B^{(i)}$ für alle $i \in \{1, \dots, m\}$ ist gleichbedeutend mit $AX_n = B$. □

Bemerkung 2.4. *Mit der oben definierten Rekursion von X_k genügt R_k der Rekursion:*

$$R_k = R_{k-1} - AP_{k-1}\alpha_k. \tag{2.1}$$

Obiger Satz und eine geschickte Wahl der P_j hilft uns einen Algorithmus für ein Block-CG-Verfahren herzuleiten. Wir starten mit: $P_0 = R_0$. Angenommen wir haben bereits $l+1$ Matrizen $P_0, \dots, P_l$ mit vollem Spaltenrang und $P_i^T A P_j = 0$ für alle $i \neq j$ konstruiert. Wir nehmen an, dass R_{l+1} vollen Spaltenrang hat. Wie auf nicht vollen Spaltenrang reagiert wird sehen wir im Kapitel 2.3. Wie im klassischen CG-Verfahren A-orthogonalisieren wir P_{l+1} gegen alle P_j mit:

$$P_{l+1} = R_{l+1} - \sum_{j=0}^{l} P_j \beta_j^{(l+1)}. \tag{2.2}$$

Wir definieren $\beta_j^{(l+1)} := (P_j^T A P_j)^{-1} R_{l+1}^T A P_j$.

Im nächsten Lemma zeigen wir, dass P_{l+1} nur durch $\beta_l^{(l+1)}$ und R_{l+1} und P_l definiert ist.

Lemma 2.5. *Es gilt $\beta_j^{(l+1)} = 0$ für $j \in \{0, \dots, l-1\}$.*

Beweis. Mithilfe von (2.2) gilt folgende Gleichheit:

$$R_{l+1}^T R_j = R_{l+1}^T (P_j + \sum_{i=0}^{j-1} P_i \beta_i^{(j)}) = R_{l+1}^T P_j + \sum_{i=0}^{j-1} R_{l+1}^T P_i \beta_i^{(j)} = 0, \tag{2.3}$$

für $j \leqslant l+1$. Aufgrund von (2.1) folgt für $j < l$:

$$R_{l+1}^T A P_j = R_{l+1}^T (R_j - R_{j+1})(\alpha_{j+1})^{-1} = 0, \tag{2.4}$$

womit $\beta_j^{(l+1)} = 0$ für $j \in \{0, \ldots, l-1\}$ folgt. □

Dieses Lemma liefert uns die Rekursion:

$$P_{l+1} = R_{l+1} - P_l \beta_l^{(l+1)} = R_{l+1} - P_l \beta_{l+1} \tag{2.5}$$

mit

$$\beta_{l+1} := (P_l^T A P_l)^{-1} R_{l+1}^T A P_l. \tag{2.6}$$

Dass P_{l+1} vollen Spaltenrang hat und dass $P_{l+1}^T A P_j = 0$ ist, halten wir in folgendem Lemma fest.

Lemma 2.6. *Sei P_{l+1} wie in (2.5) definiert, und R_{l+1} habe vollen Spaltenrang. Dann gilt:*

$$P_i^T A P_{l+1} = 0$$

für alle $i \in \{0, \ldots, l\}$ und P_{l+1} hat vollen Spaltenrang.

Beweis. Wir zeigen zuerst $P_i^T A P_{l+1} = 0$ denn:

$$P_i^T A P_{l+1} = P_i^T A R_{l+1} - P_i^T A P_l \beta_{l+1} = P_i^T A R_{l+1} = 0$$

für $i < l$ wegen (2.4). Für $i = l$ gilt:

$$P_l^T A P_{l+1} = P_l^T A R_{l+1} - P_l^T A P_l (P_l^T A P_l)^{-1} R_{l+1}^T A P_l = 0.$$

Somit ist nur noch zu zeigen, dass P_{l+1} vollen Spaltenrang hat. Für den Beweis verwenden wir folgende Aussage der Linearen Algebra:

P_{l+1} hat vollen Spaltenrang, genau dann, wenn $P_{l+1}^T P_{l+1}$ positiv definit ist.

Also zeigen wir, dass $P_{l+1}^T P_{l+1}$ positiv definit ist falls $R_{l+1}^T R_{l+1}$ positiv definit ist.

$$\begin{aligned} P_{l+1}^T P_{l+1} &= (R_{l+1}^T - C^T)(R_{l+1} - C) \\ &= R_{l+1}^T R_{l+1} - C^T R_{l+1} - R_{l+1}^T C + C^T C, \end{aligned}$$

wobei $C := P_l \beta_{l+1}$. Die beiden Terme:

$$C^T R_{l+1} = \beta_{l+1}^T P_l^T R_{l+1} = 0$$

wegen des letzten Matrixpoduktes und

$$R_{l+1}^T C = R_{l+1}^T P_l \beta_{l+1} = 0$$

aufgrund des ersten Matrixproduktes. Damit gilt für $0 \neq v \in \mathbb{R}^m$:

$$\begin{aligned}\langle P_{l+1}^T P_{l+1} v, v\rangle &= \langle (R_{l+1}^T R_{l+1} + C^T C)v, v\rangle \\ &= \|R_{l+1}v\|^2 + \|Cv\|^2 \geqslant \|R_{l+1}v\|^2 > 0.\end{aligned}$$

□

Bemerkung 2.7. *Den Spaltenrang von R_{l+1} kann man nicht auf vorangegangene Residuen zurückführen. Als Beispiel dafür verwenden wir*

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} \qquad \textit{mit dem Startwert} \qquad X_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$\textit{und der rechten Seite} \qquad B = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 2 & 2 \end{pmatrix}.$$

Rekursiv berechnet erhalten wir:

$$R_0 = \begin{pmatrix} 0 & 0 \\ 0 & -1 \\ 1 & 1 \end{pmatrix} \qquad \textit{und} \qquad R_1 = \begin{pmatrix} -\frac{1}{3} & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

Wie im klassischen CG-Verfahren verwenden wir stabilere in exakter Arithmetik äquivalente Berechnungen von β_{l+1} und α_{l+1}. Aus (2.1) und der Orthogonalität von R_l und R_{l+1} folgt:

$$0 = R_l^T R_{l+1} = R_l^T (R_l - A P_l \alpha_{l+1}) = R_l^T R_l - R_l^T A P_l \alpha_{l+1}.$$

Wegen (2.5) und der A-orthogonalität der P_j gilt:

$$R_l^T A P_l = (P_l + P_{l-1}\beta_l)^T A P_l = P_l^T A P_l + \beta_l^T P_{l-1}^T A P_l = P_l^T A P_l.$$

Nach α_{l+1} aufgelöst erhalten wir:

$$\alpha_{l+1} = (P_l^T A P_l)^{-1} R_l^T R_l. \tag{2.7}$$

Aus der A-orthogonalität von P_l und P_{l+1} und (2.5) folgt weiter

$$R_{l+1}^T A P_l = \beta_{l+1}^T P_l^T A P_l + P_{l+1}^T A P_l = \beta_{l+1}^T P_l^T A P_l.$$

Auflösen nach β_{l+1}^T und Einsetzen von $AP_l = (R_l - R_{l+1})\alpha_{l+1}^{-1}$ in der linken Seite ergibt

$$\beta_{l+1} = \beta_{l+1}^T = (R_l^T R_l)^{-1} R_{l+1}^T R_{l+1}.$$

Diese Gleichungen führen zu einem Block-CG-Verfahren, das wir in Algorithmus 1 zusammenfassen.

Algorithmus 1 Block-CG-Verfahren

Input: $A, B, X_0, \epsilon > 0$

1: $R_0 = P_0 = B - AX_0, k = 0$
2: **while** $\|R_k\| > \epsilon$ **do**
3: $\quad k \leftarrow k+1$
4: $\quad$ Löse: $P_{k-1}^T A P_{k-1} \alpha_k = R_{k-1}^T R_{k-1}$
5: $\quad X_k = X_{k-1} + P_{k-1}\alpha_k$
6: $\quad R_k = R_{k-1} - AP_{k-1}\alpha_k$
7: $\quad$ Löse: $R_{k-1}^T R_{k-1}\beta_k = R_k^T R_k$
8: $\quad P_k = R_k + P_{k-1}\beta_k$
9: **end while**

Output: X_k

2.2 Konvergenzanalyse

In Satz 2.2 haben wir gezeigt, dass der Algorithmus in exakter Arithmetik nach höchstens $n+1$ Iterationen konvergiert. Um nun bessere Fehlerabschätzungen zu bekommen, zeigen wir in diesem Unterkapitel, dass das Block-CG-Verfahren mindestens so schnell konvergiert wie das klassische CG-Verfahren. Des Weiteren definieren wir, was wir unter einem Block-Krylovraum verstehen.

Wir betrachten für ein $i \in \{1, \dots, m\}$ den Fehler $\left\|E_k^{(i)}\right\|_A = \left\|X_*^{(i)} - X_k^{(i)}\right\|_A$. Es gilt: $X_k \in X_0 + V_k$ mit

$$V_k := \left\{ \sum_{j=0}^{k-1} A^j P_0 \gamma_j \mid \gamma_j \in \mathbb{R}^{m\times m} \right\} = \left\{ \sum_{j=0}^{k-1} R_j \widetilde{\gamma}_j \mid \widetilde{\gamma}_j \in \mathbb{R}^{m\times m} \right\}$$

Wir nennen $V_k = V_k(A, P_0)$ den k-ten Block-Krylovraum von A und P_0. Für $X_k^{(i)}$ gilt:

$$X_k^{(i)} = X_0^{(i)} + w_k^{(i)} \text{ mit } w_k \in V_k. \tag{2.8}$$

Zusätzlich gilt:

$$R_k^{(i)} \perp v_k^{(i)} \text{ für alle } v_k \in V_k. \tag{2.9}$$

Beweis. Sei $v_k \in V_k$ beliebig. Wir betrachten:

$$\left\langle R_k^{(i)}, v_k^{(i)} \right\rangle = \sum_{j=0}^{k-1} \left\langle R_k^{(i)}, (R_j\widetilde{\gamma}_j)^{(i)} \right\rangle$$

Um $(R_j\widetilde{\gamma}_j)^{(i)}$(i-te Spalte des Matrixprodukts) zu berechnen setzen wir $R_j =: r$ und $\widetilde{\gamma}_j =: \xi$. Wir verwenden $r := (r_1, \dots, r_m)$ und $\xi = (\xi_1, \dots, \xi_m)$ und wollen $(r\xi)^{(i)}$

berechnen.

$$(r_1,\dots,r_m)(\xi_1,\dots,\xi_m) = \begin{pmatrix} \sum_{\nu=1}^{m} r_{1,\nu}\xi_{\nu,i} \\ \sum_{\nu=1}^{m} r_{2,\nu}\xi_{\nu,i} \\ \vdots \\ \sum_{\nu=1}^{m} r_{\nu,\nu}\xi_{\nu,i} \end{pmatrix} = \sum_{\nu=1}^{m} r_\nu \xi_{\nu,i}$$
$$= \sum_{\nu=1}^{m} R_j^{(\nu)} \xi_{\nu,i} \text{ mit } \xi_{\nu,i} \in \mathbb{R}.$$

Betrachten wir, das obige Skalarprodukt so gilt:

$$\left\langle R_k^{(i)}, \sum_{\nu=1}^{m} R_j^{(\nu)} \xi_{\nu,i} \right\rangle = \sum_{\nu=1}^{m} \xi_{\nu,i} \left\langle R_k^{(i)}, R_j^{(\nu)} \right\rangle \overset{(2.3)}{=} 0.$$

□

Folgende Sätze und Definition verwenden wir für die Abschätzung des Fehlers $E_k^{(i)}$:

Satz 2.8. *[Cla12, Satz 5.1]*
Sei $\mathcal{K}$ ein Unterraum von $\mathbb{R}^n$, $x_0, b \in \mathbb{R}^n$ beliebig und $A \in \mathbb{R}^{n\times n}$ selbstadjungiert und positiv definit. Setze $x^* = A^{-1}b$. Dann ist

$$\|\tilde{x} - x^*\|_A = \min_{x \in x_0+\mathcal{K}} \|x - x^*\|_A$$

genau dann, wenn

$$\tilde{x} \in x_0 + \mathcal{K} \qquad \textit{und} \qquad b - A\tilde{x} \perp \mathcal{K}$$

gilt.

Definition 2.9. *Wir definieren:*

$$(V_k)^{(i)} := \{v_k^{(i)} \mid v_k \in V_k\} = \left\{ \sum_{j=0}^{k-1} \sum_{\nu=1}^{m} R_j^{(\nu)} \xi_{\nu,j} \mid \xi_{\nu,j} \in \mathbb{R} \right\}$$
$$= \left\{ \sum_{j=0}^{k-1} \sum_{\nu=1}^{m} A^j P_0^{(\nu)} \xi_{\nu,j} \mid \xi_{\nu,j} \in \mathbb{R} \right\}.$$

Satz 2.10. *Seien $(V_k)^{(i)}$ wie oben definiert und $K_k(A, R_0^{(i)}) := \operatorname{span}\{R_0^{(i)}, AR_0^{(i)}, \dots, A^{k-1}R_0^{(i)}\}$ der k-te Krylovraum von A und $R_0^{(i)}$ dann folgt:*

$(V_k)^{(i)}$ ist ein Unterraum und $K_k(A, R_0^{(i)}) \subset (V_k)^{(i)}$.

Beweis. Seien $v_k^{(i)}, w_k^{(i)} \in (V_k)^{(i)}$ und $\beta, \alpha \in \mathbb{R}$. Wir betrachten:

$$\begin{aligned}\alpha v_k^{(i)} + \beta w_k^{(i)} &= \alpha \sum_{j=0}^{k-1} \sum_{\nu=1}^{m} R_j^{(\nu)} \xi_{\nu,i} + \beta \sum_{j=0}^{k-1} \sum_{\nu=1}^{m} R_j^{(\nu)} \widetilde{\xi_{\nu,i}} \\ &= \sum_{j=0}^{k-1} \sum_{\nu=1}^{m} R_j^{(\nu)} (\alpha \xi_{\nu,i} + \beta \widetilde{\xi_{\nu,i}}) \in (V_k)^{(i)}.\end{aligned}$$

Sei $b \in K_k(A, R_0^{(i)}) = K_k(A, P_0^{(i)})$ dann gilt:

$$b = \sum_{j=0}^{k-1} \lambda_j A^j P_0^{(i)} = \sum_{j=0}^{k-1} (A^j P_0 \gamma_j)^{(i)} \in (V_k)^{(i)}$$

mit

$$\mathbb{R}^{m \times m} \ni \gamma_j := \begin{cases} (\gamma_j)_{s,t} = \lambda_j & \text{für } s = i \text{ und } t = i \\ (\gamma_j)_{s,t} = 0 & \text{sonst} \end{cases}.$$

□

Aufgrund von (2.8), (2.9) und Satz 2.8 gilt:

$$\left\| X_{(i)}^* - X_k^{(i)} \right\|_A = \min_{x \in X_0^{(i)} + (V_k)^{(i)}} \left\| X_{(i)}^* - x \right\|_A \leqslant \min_{x \in X_0^{(i)} + K_k(R_0^{(i)})} \left\| X_{(i)}^* - x \right\|_A$$

wegen

$$X_0^{(i)} + (V_k)^{(i)} \supset X_0^{(i)} + K_k(A, R_0^{(i)}).$$

Wir wissen nun, dass das Block-CG-Verfahren mindestens gleich schnell konvergiert wie das normale CG-Verfahren. Wie zu Beginn der Arbeit angesprochen ist der Suchraum für das Block-CG-Verfahren oftmals viel größer als im normalen CG-Verfahren. In [O'L80, Kapitel 4] wird eine Abschätzung für den Block-CG-Algorithmus hergeleitet. Um diese Abschätzung zu formulieren führen wir folgende Definitionen ein:

Definition 2.11. *Sei X_k mit dem Block-CG-Verfahren berechnet. Wir definieren:*

- $E_k := X_k - X_* = [E_k^{(1)}, \ldots, E_k^{(m)}]$.
- *U sei die unitäre Transformationsmatrix von A, sodass $U^T A U$ eine Diagonalmatrix ist.*
- *Für $i \in \{1, \ldots, m\}$ sei e_i der Fehler E_0 ohne der i-ten Spalte, das heißt:*

$$e_i = [E_0^{(1)}, \ldots, E_0^{(i-1)}, E_0^{(i+1)} \ldots, E_0^{(m)}]$$

- *F_i sei jene Matrix für die gilt: Die Spalten von UF_i sind eine orthonormale Basis der linearen Hülle der Spalten von Ae_i.*

Die Abschätzung beschreiben wir im folgenden Satz:

Satz 2.12. *[O'L80, Theorem 5] Sei X_k mit dem Block-CG-Verfahren berechnet. Seien $0 < \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ die Eigenwerte der symmetrisch positiv definiten Matrix A der Problemstellung (1.1). Sei $\kappa_B := \frac{\lambda_n}{\lambda_m}$. Sei die Matrix F_i der Definition 2.11 unterteilt in*

$$F_i =: \begin{pmatrix} F_i^{(1)} \in \mathbb{R}^{(m-1)\times(m-1)} \\ F_i^{(2)} \in \mathbb{R}^{(n-m+1)\times(m-1)} \end{pmatrix}.$$

Sei $\sigma_{\min}(F_i^{(1)}) > 0$. Dann gilt:

$$\left\| E_k^{(i)} \right\|_A \leqslant \left(\frac{1 - \sqrt{\kappa_B^{-1}}}{1 + \sqrt{\kappa_B^{-1}}} \right)^k c,$$

wobei c eine Konstante ist, die nicht von k sehr wohl aber von m abhängt.

Im Vergleich dazu sieht das Konvergenzverhalten des gewöhnlichen CG-Verfahren folgendermaßen aus:

Satz 2.13. *[Cla12, Satz 7.3] Sei $X_k^{(i)}$ die k-te Iteration des Vektors $X_0^{(i)}$ im gewöhnlichen CG-Verfahren, dann gilt:*

$$\left\| E_k^{(i)} \right\|_A \leqslant 2 \left(\frac{1 - \sqrt{\kappa_C^{-1}}}{1 + \sqrt{\kappa_C^{-1}}} \right)^k \left\| E_0^{(i)} \right\|_A,$$

mit $\kappa_C := \lambda_n / \lambda_1$.

In beiden Abschätzungen sehen wir, dass die Konvergenz von κ_B beziehungsweise von κ_C abhängt. Je näher diese Werte bei 1 liegen, desto schneller konvergiert das Verfahren, das heißt ist A eine Matrix, in der die Eigenwerte λ_1 und λ_n weit auseinander liegen, λ_m und λ_n aber nicht, muss nach den obigen Sätzen das Block-CG-Verfahren einen klaren Vorteil gegenüber dem gewöhnlichen CG-Verfahren haben. Vorausgesetzt die Bedingungen im Satz 2.12 sind erfüllt.

Ohne näher darauf einzugehen, zeigt uns die Bedingung $\sigma_{\min}(F_i^{(1)}) > 0$ in Satz 2.12, dass beim Block-CG-Verfahren der Startwert X_0 wichtig für die Konvergenz des Verfahrens ist.

In [O'L80, Kapitel 4] wird zusätzlich gezeigt, dass das Block-CG-Verfahren nicht nur bei einer kleinen Differenz von λ_m und λ_n schnelle Konvergenz aufweist, sondern auch dann schnell konvergiert, wenn $n - m + 1$ Eigenwerte der Matrix A nah beieinander liegen, das heißt, dass das Block-CG-Verfahren zum Beispiel bei einer Matrix A deren Eigenwerte $\lambda_{m-1}, \ldots, \lambda_{n-1}$ nah beieinander liegen schnell die Näherung der Lösung X_* findet, egal wie groß die Differenz von λ_n und λ_m ist.

2.3 Reorthogonalisierung, Deflation und Präkonditionierung

Im Algorithmus 1 berücksichtigen wir nicht, dass die Residuen linear abhängig werden können und dann α_k und β_k nicht mehr wohldefiniert sind. Um dieses Problem zu lösen, orthonormalisieren wir die Spalten von P_k in jedem Schritt und können dann verifizieren ob Spalten in P_k linear abhängig sind. Tritt dieser Fall ein, entfernen wir die Spalten, die lineare Abhängigkeit verursachen, und behandeln diese separat. Mit den übrigen Spalten starten wir das Verfahren neu.

$\widetilde{P_k}$ sei P_k von Algorithmus 1, $\widetilde{\alpha_k}$ sei α_k von Algorithmus 1 und $\widetilde{\beta_k}$ sei β_k von Algorithmus 1. Wir definieren P_k reorthonormalisiert wie folgt: Sei $Q_k\theta_k = \widetilde{P_k}$ die QR- Zerlegung von $\widetilde{P_k}$. Es gilt: $Q_k \in \mathbb{R}^{n\times m}$ ist orthogonal, $\theta_k \in \mathbb{R}^{m\times m}$ hat obere Dreiecksform. Wir definieren P_k reorthonormalisiert als das Q_k der QR-Zerlegung, das heißt:

$$P_k := \widetilde{P_k}(\theta_k)^{-1}. \tag{2.10}$$

Wir definieren α_k reorthonormalisiert und β_k reorthonormalisiert wie folgt:

$$\alpha_k := (P_{k-1}^T A P_{k-1})^{-1}(\theta_{k-1}^T)^{-1} R_{k-1}^T R_{k-1} \quad \text{und} \tag{2.11}$$

$$\beta_k := \theta_{k-1}(R_{k-1}^T R_{k-1})^{-1} R_k^T R_k. \tag{2.12}$$

Die Definitionen (2.10), (2.11) und (2.12) führen uns zu Algorithmus 2. Folgender Satz

Algorithmus 2 Block-CG-Verfahren mit Reorthonormalisierung

Input: $A, B, X_0, \epsilon > 0$

1: $R_0 = \pi_0 = B - AX_0, k = 0$
2: Berechne Q_0 und θ_0 mit QR-Zerlegung von R_0
3: $P_0 = Q_0$
4: **while** $\|R_k\| > \epsilon$ **do**
5: $\quad k \leftarrow k+1$
6: $\quad$ Löse: $\theta_{k-1}^T P_{k-1}^T A P_{k-1}\alpha_k = R_{k-1}^T R_{k-1}$
7: $\quad X_k = X_{k-1} + P_{k-1}\alpha_k$
8: $\quad R_k = R_{k-1} - AP_{k-1}\alpha_k$
9: $\quad$ Löse: $R_{k-1}^T R_{k-1}\widetilde{\beta_k} = R_k^T R_k$
10: $\quad \beta_k = \theta_{k-1}\widetilde{\beta_k}$
11: $\quad \pi_k = R_k + P_{k-1}\beta_k$
12: $\quad$ Berechne Q_k und θ_k mit QR-Zerlegung von π_k
13: $\quad P_k = Q_k$
14: **end while**

Output: X_k

zeigt uns, dass die berechneten X_k und R_k mit Reothonormalisierung, mit den X_k und R_k ohne Reorthonormalisierung übereinstimmen.

Satz 2.14. *Sei α_k, β_k und P_k wie in Algorithmus 2 definiert. Dann gilt:*

X_k und R_k sind dieselben wie in Algorithmus 1 und $\pi_k = \widetilde{P_k}$.

Beweis. Wir beweisen diesen Satz mit Induktion. Sei $k = 0$ dann gilt: $\pi_0 = R_0$ und X_0 beziehungsweise R_0 sind gleich wie im Algorithmus 1. Angenommen es gelte für k. Wir betrachten:

$$X_{k+1} = X_k + P_k \alpha_{k+1}.$$

Nach Induktionsvorraussetzung ist X_k ident mit X_k von Algorithmus 1 und $\pi_k = \widetilde{P_k}$.

$$\begin{aligned} P_k \alpha_{k+1} &= \pi_k \theta_k^{-1} (P_k^T A P_k)^{-1} (\theta_k^T)^{-1} R_k^T R_k \\ &= \pi_k (\theta_k^T P_k^T A P_k \theta_k)^{-1} R_k^T R_k = \pi_k \widetilde{\alpha}_{k+1}. \end{aligned}$$

Somit bleibt uns noch zu zeigen, dass $\pi_k = \widetilde{P}_{k+1}$.

$$\pi_{k+1} = R_{k+1} + P_k \theta_k \widetilde{\beta}_{k+1} = R_{k+1} + \pi_k \widetilde{\beta}_{k+1}.$$

□

Um den Rang von P_k in der QR-Zerlegung zu bestimmen vernachlässigen wir jene Spalten von P_k die lineare Abhängigkeit verursachen in Q. Zum Beispiel können wir das im modifiziertem Gram-Schmidt-Verfahren[Cla12, Algorithmus 2.2] vollziehen, indem wir anstatt bei $r_{j,j} = 0$ den Algorithmus abzubrechen, q_j und r_j nicht berechnen und den Index j speichern um diese Spalte separat behandeln zu können. Das modifizierte Gram-Schmidt-Verfahren führen wir mit einer Indexverschiebung(da eine Spalte gelöscht wurde) fort.

Durch das separate Behandeln jener Spalten, die lineare Abhängigkeit verursachen, sind α_{k+1} und β_{k+1} wegen (2.7) und (2.6) wohldefiniert. Dadurch kann der Algorithmus bei linear abhängigen Spalten in P_k trotzdem fortgeführt werden wie in Algorithmus 3 angeführt. Das Entfernen der Spalten, die lineare Abhängigkeit verursachen, nennen wir Deflation.

Es soll angemerkt sein, dass in verschiedenen anderen Algorithmen der Begriff Deflation in einer anderen Bedeutung verwendet wird. Hier assoziieren wir Deflation immer nur mit der obigen Definition.

Da wir im Satz 2.12 eine ähnliche Konvergenzanalyse wie im normalen CG-Verfahren verwenden, macht es auch beim Block-CG-Verfahren Sinn eine Matrix als Vorkonditionierer zu wählen. Auch hier hängt die Gestalt dieser Matrix stark von der Matrix A ab.

Sei M symmetrisch positiv definit als Vorkonditionierer gewählt, dann kommen wir auf Algorithmus 3.

Algorithmus 3 Präkonditioniertes Block-CG-Verfahren mit Reorthogonalisierung und Deflation

Input: $A, B, X_0, \epsilon > 0$

1: X_0 ist so gewählt, dass R_0 vollen Spaltenrang hat.
2: $R_0 = \widetilde{P_0} = B - AX_0, k = 0$
3: Berechne Q_0 und θ_0 mit QR-Zerlegung von MR_0.
4: **while** $\|R_k\| > \epsilon$ **do**
5: $\quad k \leftarrow k + 1$
6: $\quad$ Löse: $\theta_{k-1}^T P_{k-1}^T A P_{k-1} \alpha_k = R_{k-1}^T M R_{k-1}$
7: $\quad X_k = X_{k-1} + P_{k-1}\alpha_k$
8: $\quad R_k = R_{k-1} - AP_{k-1}\alpha_k$
9: $\quad$ Löse: $R_{k-1}^T M R_{k-1} \widetilde{\beta_k} = R_k^T M R_k$
10: $\quad \widetilde{\beta_k} = \theta_{k-1}\widetilde{\beta_k}$
11: $\quad \widetilde{P_k} = MR_k + P_{k-1}\beta_k$
12: $\quad$ Berechne Q_k und θ_k mit QR-Zerlegung von $\widetilde{P_k}$
13: $\quad$ **if** Spaltenrang$(\widetilde{P_k}) < m$ and $\|R_k\| > \epsilon$ **then**
14: $\quad\quad$ Betrachte l.a. verursachende Spalten separat und starte das Verfahren mit den restlichen Spalten neu.
15: $\quad$ **else**
16: $\quad\quad P_k = Q_k$
17: $\quad$ **end if**
18: **end while**

Output X_k

3 Block-GMRES-Verfahren

Das Block-CG-Verfahren konvergiert wie das klassische CG-Verfahren nur bei symmetrischen, positiv definiten Matrizen, das heißt: Ist A in der Problemstellung (1.1) nicht symmetrisch positiv definit, konvergiert das Block-CG-Verfahren im Allgemeinen nicht. Dies motiviert ein Block-GMRES-Verfahren für invertierbare Matrizen.

In diesem Kapitel wird ein Block-GMRES-Verfahren hergeleitet und analysiert. Die Herleitung des Verfahrens orientiert sich an [Cla12, Kapitel 8]. Das klassische GMRES-Verfahren verwendet den Arnoldi-Prozess um eine orthonormale Basis des entsprechenden Krylovraums zu finden. Im Block-GMRES-Verfahren verwenden wir den Block-Arnoldi-Prozess, um eine blockorthonormale Basis des Block-Krylovraums zu finden. Wie dieses Verfahren aussieht, wollen wir im Folgenden beschreiben.

3.1 Block-Arnoldi-Prozess

Um einen Block-Arnoldi-Prozess herzuleiten, verwenden wir folgende Definitionen:

Definition 3.1. *Seien $X, Y \in \mathbb{R}^{n\times m}$. Wir definieren:*

- *Die Matrizen X, Y sind blockorthogonal, falls $X^T Y = 0$.*
- *Wenn $X^T X = E$ mit E als Einheitsmatrix in $\mathbb{R}^{m\times m}$, dann heißt X orthogonal.*
- *Seien $Y_k \in \mathbb{R}^{n\times m}$ und gelte*

$$Y_l^T Y_j = \begin{cases} E & \text{für } j = l \\ 0 & \text{sonst} \end{cases},$$

dann nennen wir die Matrizen Y_k blockorthonormal.

Bemerkung 3.2. *Falls die Matrizen Y_k blockorthogonal sind heißt das, dass die $Y_k^{(j)}$ orthonormal zueinander sind für alle k und alle j.*

Der nächste Schritt ist es für den Block-Krylovraum

$$V_k(A, W) = \left\{ \sum_{j=0}^{k-1} A^j W \gamma_j \mid \gamma_j \in \mathbb{R}^{m\times m} \right\}$$

blockorthonormale Matrizen Y_j zu finden sodass:

$$V_k(A, W) = \left\{ \sum_{j=0}^{k-1} Y_j \gamma_j \mid \gamma_j \in \mathbb{R}^{m\times m} \right\}.$$

Falls die blockorthonormalen Matrizen Y_j vollen Spaltenrang haben, nennen wir sie blockorthonormale Basis von V_k. Den Algorithmus zur Bestimmung der Y_j bezeichnen

wir als Block-Arnoldi-Prozess. Wie im normalen Arnoldi-Prozess orthonormalisieren wir in jedem Schritt und erzeugen durch die Multiplikation mit A den nächsten Vektor. Die QR-Zerlegung ist wie im Block-CG-Verfahren, jene Zerlegung bei der q_j gelöscht und eine Indexverschiebung vollzogen wird, falls im modifiziertem Gram-Schmidt-Verfahren $r_{j,j} = 0$.

Algorithmus 4 Block-Arnoldi-Prozess

Input: $W \in \mathbb{R}^{n\times m}$ mit vollem Spaltenrang

1: $Y_1\theta = W$ mit $Y_1\theta$ QR-Zerlegung von W.
2: **for** j=1,...,k **do**
3: $\quad\widetilde{Y}_{j+1} = AY_j$
4: $\quad$**for** i=1,...,j **do**
5: $\quad\quad h_{i,j} = Y_i^T\widetilde{Y}_{j+1}$
6: $\quad\quad\widetilde{Y}_{j+1} \leftarrow \widetilde{Y}_{j+1} - Y_i h_{i,j}$
7: $\quad$**end for**
8: $\quad Y_{j+1}h_{j+1,j} = \widetilde{Y}_{j+1}$ mit $Y_{j+1}h_{j+1,j}$ QR-Zerlegung von $\widetilde{Y}_j$.
9: **end for**

Output: $Y_1, \ldots, Y_k, h_{i,j}$

Im Algorithmus 4 gilt $Y_j \in \mathbb{R}^{n\times m}$ und $h_{i,j} \in \mathbb{R}^{m\times m}$. In der weiteren Herleitung nehmen wir an, dass die berechneten Y_j vollen Spaltenrang haben. Es ist wichtig im Block-GMRES-Verfahren den Fall, dass die berechneten Y_j nicht vollen Spaltenrang haben zu berücksichtigen und eine Lösung für dieses Problem zu finden.

Zu Beginn wollen wir aber die leichtere Variante wählen um diese dann in weiterer Folge auszuweiten. Wir definieren:

$$y_k := (Y_1, \ldots, Y_k) \in \mathbb{R}^{n\times km} \tag{3.1}$$

und

$$\overline{H}_k := \begin{pmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,k} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,k} \\ & h_{3,2} & \ddots & \vdots \\ & & \ddots & h_{k,k} \\ & & & h_{k+1,k} \end{pmatrix} \in \mathbb{R}^{(k+1)m\times km}$$

Aus Algorithmus 4 folgt damit:

$$Ay_k = y_{k+1}\overline{H}_k.$$

Beweis. Es gilt in Algorithmus 4:

$$AY_j = \widetilde{Y}_{j+1} + \sum_{i=1}^{j} Y_i h_{i,j} = Y_{j+1}h_{j+1,j} + \sum_{i=1}^{j} Y_i h_{i,j} = \sum_{i=1}^{j+1} Y_i h_{i,j}.$$

□

Wir sind mit diesen Erkenntnissen in der Lage ein Block-GMRES-Verfahren herzuleiten.

3.2 Block-GMRES-Verfahren

Die Grundidee des Block-GMRES-Verfahren ist es mithilfe des Block-Arnoldi-Prozesses das Residuum $\left\|B^{(i)} - AX^{(i)}\right\|$ mit $X^{(i)} \in X_0^{(i)} + (V_k(A, R_0))^{(i)}$ für alle i zu minimieren.

Wir gehen im Block-GMRES-Verfahren zu Beginn gleich vor wie im normalen GMRES-Verfahren. Es verändern sich nur die Dimensionen der einzelnen Komponenten der Gleichungen. Da wir jedes $X \in X_0 + V_k(A, R_0)$ schreiben können als $X = X_0 + y_k\xi$ für ein $\xi \in \mathbb{R}^{km\times m}$ gilt:

$$\begin{aligned} B - AX &= B - A(X_0 + y_k\xi) = R_0 - Ay_k\xi \\ &= Y_1\theta - y_{k+1}\overline{H}_k\xi = y_{k+1}(E_1\theta - \overline{H}_k\xi) \end{aligned} \tag{3.2}$$

mit $E_1 \in \mathbb{R}^{(k+1)m\times m}$ und den ersten $m \times m$ Einträgen als Einheitsmatrix. In der nächsten Definition wird eine Matrixnorm und ein Matrixskalarprodukt eingeführt um die Norm dann auf $B - AX$ anwenden zu können.

Definition 3.3. *Seien $X, Y \in \mathbb{R}^{n\times m}$. Für X, Y definieren wir das Skalarprodukt $\langle ., . \rangle_F$ und $\|.\|_F$ (Forbenius Norm) wie folgt:*

$$\langle X, Y\rangle_F := \operatorname{spur}(X^TY), \quad \|X\|_F = \sqrt{\operatorname{spur}(X^TX)}.$$

Die F-Norm von X genauer betrachtet ergibt:

$$\|X\|_F = \sqrt{\sum_{j=1}^{m} \left\|X^{(j)}\right\|_2^2}.$$

Die F-Norm ist keine Operatornorm also keine von einer Vektornorm induzierte Norm sondern wird als Block-Vektornorm betrachtet.

Greifen wir die Grundidee des Block-GMRES-Verfahren wieder auf sehen wir, dass wir $\|B - AX\|_F$ für $X \in X_0 + V_k(A, R_0)$ minimieren wollen. Das in der Gleichung (3.1) definierte blockorthogonale y_k ist bezüglich der F-Norm Längentreu.

Beweis. Für ein beliebiges $J \in \mathbb{R}^{km\times m}$ gilt:

$$\|y_kJ\|_F = \sqrt{\operatorname{spur}(J^Ty_k^Ty_kJ)} = \sqrt{\operatorname{spur}(J^TJ)} = \|J\|_F.$$

□

Deswegen und Aufgrund von Gleichung (3.2) gilt:

$$\begin{aligned} \|B - AX\|_F &= \left\|E_1\theta - \overline{H}_k\xi\right\|_F \\ &= \sqrt{\sum_{j=1}^{m} \left\|E_1\theta^{(j)} - \overline{H}_k\xi^{(j)}\right\|_2^2} \end{aligned} \tag{3.3}$$

Daraus können wir schließen, dass wir für das Block-GMRES-Verfahren folgende Ausgleichsprobleme lösen müssen:

$$\left\| E_1\theta^{(j)} - \overline{H}_k \xi^{(j)} \right\|_2 \qquad \text{für alle } j \in \{1, ..., m\}$$

Wir fassen das abstrakte Block-GMRES-Verfahren in Algorithmus 5 zusammen. Die im

Algorithmus 5 Abstraktes Block-GMRES-Verfahren

Input: $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}, X_0 \in \mathbb{R}^{n\times m}, \epsilon > 0$

1: $R_0 = B - AX_0, s = 1$
2: **while** $\|R_s\|_F > \epsilon$ **do**
3: Berechne $Y_s, \overline{H}_s$ mit Block-Arnoldi-Prozess 4
4: **for** $j = 1, \ldots, m$ **do**
5: Löse $\min_{\xi^{(j)}\in\mathbb{R}^{km}} \left\|E_1\theta^{(j)} - \overline{H}_s\xi^{(j)}\right\|_2$ für ξ_s
6: **end for**
7: Setze $X_s = X_0 + Y_s\xi_s$, $s \leftarrow s+1$
8: **end while**

Output: X_s

Algorithmus 5 berechneten $X_s^{(j)}$ sind daher die Lösungen des in der Grundidee beschriebenen Problems. Da wir vorausgesetzt haben, dass die Y_k vollen Spaltenrang haben, können wir schließen, dass die Matrix X_s von Algorithmus 5 in exakter Arithmetik spätestens nach n Schritten die Lösung X_* annimmt. Denn die Dimension von $V_n(A, R_0)^{(j)} \geqslant n$ für alle $j \in \{1, \ldots, m\}$ und wir minimieren $\left\|B^{(j)} - AX^{(j)}\right\|_2$ für $X^{(j)} \in V_n(A, R_0)^{(j)}$.

In der Praxis wird in jedem Schritt die bereits konstruierte Blockorthonormalbasis $Y_1, \ldots, Y_s$ verwendet, sodass jeweils nur eine neue $n \times m$ Matrix für Y_{s+1} und $\overline{H}_{s+1}$ berechnet werden muss. Um ξ_s im Algorithmus 5 zu berechnen wird eine Folge von orthogonalen Matrizen $G_{i+1,i}$ mit

$$G_s[\overline{H}_s, E_1\theta] = \begin{pmatrix} \mathcal{R}_s & d_s \\ 0 \in \mathbb{R}^{sm\times m} & \rho_s \end{pmatrix}, \quad G_s = G_{s+1,s}G_{s,s-1}\cdots G_{2,1}$$

konstruiert, sodass $\mathcal{R}_s \in \mathbb{R}^{sm\times sm}$ eine obere Dreiecksmatrix ist

$$d_s \in \mathbb{R}^{sm\times m}\ ,\ \rho_s \in \mathbb{R}^{m\times m}$$

und $G_{i+1,i} \in \mathbb{R}^{(s+1)m\times(s+1)m}$ so gewählt ist, dass die Matrix $h_{i+1,i}$ zu Null gemacht wird. Für die Lösung und das Residuum gilt dann:

$$\xi_s = \mathcal{R}_s^{-1}d_s, \qquad \|R_s\|_F = \|\rho_s\|_F$$

Im Folgenden wird gezeigt, dass die explizite Berechnung von X_s und ξ_s erst bei $\|\rho_s\| < \epsilon$ notwendig ist. Angenommen wir haben bereits eine QR Zerlegung von $\overline{H}_{s-1}$ konstruiert mit

$$G_{s-1}\overline{H}_{s-1} = \begin{pmatrix} \mathcal{R}_{s-1} \\ 0 \end{pmatrix}, \qquad G_{s-1}E_1\theta = \begin{pmatrix} d_{s-1} \\ \rho_{s-1} \end{pmatrix}$$

Definieren wir

$$h_s := (h_{1,s}, \cdots, h_{s,s})^T \in \mathbb{R}^{sm \times m} \qquad , G_{s-1} h_s =: (c_{s-1}, c_{s,s})^T$$

mit $c_{s-1} \in \mathbb{R}^{m(s-1) \times m}$ und $c_{s,s} \in \mathbb{R}^{m \times m}$, so ist

$$\begin{aligned} G_s \overline{H}_s &= G_{s+1,s} \begin{pmatrix} G_{s-1} \overline{H}_{s-1} & G_{s-1} h_s \\ 0 \in \mathbb{R}^{m \times (s-1)m} & h_{s+1,s} \end{pmatrix} \\ &= G_{s+1,s} \begin{pmatrix} \mathcal{R}_{s-1} & c_{s-1} \\ 0 & c_{s,s} \\ 0 & h_{s+1,s} \end{pmatrix}. \end{aligned}$$

Wir bestimmen nun $G_{s+1,s}$ so, dass

$$G_{s+1,s} \begin{pmatrix} c_{s,s} \\ h_{s+1,s} \end{pmatrix} = \begin{pmatrix} \gamma_s \\ 0 \end{pmatrix}$$

ist, und setzen

$$\mathcal{R}_s = \begin{pmatrix} \mathcal{R}_{s-1} & c_{s-1} \\ 0 & \gamma_s \end{pmatrix}.$$

Für die rechte Seite erhalten wir:

$$\begin{aligned} G_s E_1 \theta &= G_{s+1,s} \begin{pmatrix} G_{s-1} E_1 \theta \\ 0 \in \mathbb{R}^{m \times m} \end{pmatrix} \\ &= G_{s+1,s} \begin{pmatrix} d_{s-1} \\ \rho_{s-1} \\ 0 \end{pmatrix} = \begin{pmatrix} d_{s-1} \\ \delta_s \\ \rho_s \end{pmatrix} =: \begin{pmatrix} d_s \\ \rho_s \end{pmatrix} \end{aligned}$$

mit $\delta_s \in \mathbb{R}^{m \times m}$.

Für den Algorithmus bedeutet das, dass wir ξ_s und X_s erst dann berechnen, wenn das Residuum $\|R_s\|_F = \|\rho_s\|_F$ klein genug ist. Wie im normalen GMRES-Verfahren wird auch hier das Verfahren mit steigendem s sehr aufwendig. Daher wird oft ein Algorithmus verwendet, der das Verfahren, nach einer festen Anzahl von Schritten(k), neu startet. Der neue Startwert X_0 ist dabei die alte Näherung X_k bezogen auf Algorithmus 5.

Wir führen die wesentlichen Schritte des Algorithmus im Block-GMRES-Verfahren mit Neustart an. Die Einschränkung, dass Y_j vollen Spaltenrang für alle j hat, muss im Algorithmus beinhaltet sein.

3.3 Konvergenzanalyse

Aus Satz 2.10 können wir folgern, dass beim Block-GMRES-Verfahren der Suchraum über welchen das Residuum minimiert wird mindestens gleich groß ist wie im normalen

Algorithmus 6 Block-GMRES-Verfahren mit Neustart nach k Schritten

Input: $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}, X_0 \in \mathbb{R}^{n\times m}, s^*, s = 0, k\epsilon > 0$

$R_0 = B - AX_0$
Berechne θ
$d_0 = E_1\theta$
while $\left\|\rho_j\right\|_F > \epsilon$ und $s < s^*$ **do**
    $s \leftarrow s + 1$
    $j = 1$
    while $\left\|\rho_j\right\|_F > \epsilon$ und $j \leqslant k$ **do**
        Berechne $Y_j, h_{1,j}, \ldots, h_{j+1,j}$ nach Block-Arnoldi mit R_s
        Wende G_j auf $[h_{1,j}, \ldots, h_{j,j}]^T$ an$\rightsquigarrow c_{j-1}$
        Berechne $G_{j+1,j}$, das $h_{j+1,j}$ annulliert $\rightsquigarrow \mathcal{C}_j = (c_{j-1}, \gamma_j)^T$
        Wende $G_{j+1,j}$ auf $[\rho_{j-1}, 0]^T$ an $\rightsquigarrow \delta_j, \rho_j$
        $j \leftarrow j + 1$
    end while
    $\mathcal{R}_s = [\mathcal{C}_1, \ldots, \mathcal{C}_{j-1}]$, $\mathcal{Y}_s = [Y_1, \ldots, Y_{j-1}]$, $d_s = [\delta_1, \ldots, \delta_{j-1}]^T$
    $X_s = X_{s-1} + \mathcal{Y}_s\mathcal{R}_s^{-1}d_s$
    Berechne $R_s,\ \theta,\ d_0$
end while

Output:X_s

GMRES-Verfahren ($K_k(A, R_0^{(i)}) \subset (V_k)^{(i)}$). Auch hier können Situationen entstehen die das Block-Verfahren dadurch effizienter machen als das sukzessive Lösen der Gleichung $Ax = B^{(i)}$ für $i \in \{1, \ldots, m\}$. Einige solcher Situationen wollen wir noch analysieren.

Bevor wir Beispiele betrachten, um zu sehen, wann die oben genannten Block-Verfahren sinnvoll sind und wann nicht, wollen wir uns noch einmal dem Block-GMRES-Verfahren zuwenden. Das Block-GMRES-Verfahren startet nach k Iterationen neu. Im Fall, dass das Residuum R_s in Algorithmus 6 linear abhängige Spalten hat, können wir jene Spalten, die lineare Abhängigkeit verursachen, ausnehmen und mit den restlichen das Verfahren starten. Hinzu kommt, dass wir durch eine geschickte Zerlegung zwar nur mit den linear unabhängigen Spalten rechnen müssen, die Lösung X_{s+1} dann aber so konstruieren können, dass gilt: $X_{s+1} \in \mathbb{R}^{n\times m}$. Das nächste Kapitel wird sich mit der Rekonstruierung der Lösung beschäftigen.

3.4 Anfangsdeflation und Rekonstruktion der Lösung

Wir sprechen hier wieder von Deflation, da wir wieder Spalten, die lineare Abhängigkeit verursachen, von den Berechnungen ausnehmen. Anders als im Block-CG-Verfahren werden wir diese Spalten, wenn sie durch Algorithmus 7 neu berechnet sind, wieder in die Lösung aufnehmen, das heißt wir rekonstruieren die Lösung. Wie der Name verrät, wird die Anfangsdeflation immer nur zu Beginn eines Neustarts nach k Schritten durchgeführt. Kapitel 3.4 orientiert sich an [Gut07, Kapitel 12].

Wir zerlegen R_s mit einer Rang erhaltenden QR-Zerlegung, und zwar soll es eine QR-Zerlegung sein die den numerischen Rang, mit einer gewissen Toleranz tol, der Matrix aufschlüsselt. Für eine vorgegebene Toleranz tol und eine Matrix A ist der numerische Rang $\text{rank}_{\text{tol}}(A)$ definiert als die Anzahl der Singulärwerte von A, deren Betrag größer als tol ist, das heißt auch bei fast linearer Abhängigkeit wird Deflation angewendet. Die Rang erhaltende QR-Zerlegung sieht wie folgt aus:

$$R_s =: [Y_1, Y_1^{\triangle}] \begin{pmatrix} \theta & \theta^{\square} \\ 0 & \theta^{\triangle} \end{pmatrix} \pi^T =: [Y_1, Y_1^{\triangle}] \begin{pmatrix} h_0 \\ h_0^{\triangle} \end{pmatrix}. \tag{3.4}$$

Es gilt für obige Gleichung, dass $\pi \in \mathbb{R}^{m \times m}$ eine Permutationsmatrix ist. Die Matrix $[Y_1, Y_1^{\triangle}] \in \mathbb{R}^{n \times m}$ ist das Q der QR-Zerlegung, wobei die Spalten von Y_1 den Raum der Spalten von R_s aufspannt und $Y_1^{\triangle}$ im Algorithmus nicht berücksichtigt wird. Weiters ist $\theta \in \mathbb{R}^{m_0 \times m_0}$ eine reguläre obere Dreiecksmatrix und $\theta^{\triangle} \in \mathbb{R}^{(m-m_0)\times(m-m_0)}$ für das gilt: $\left\|\theta^{\triangle}\right\|_F < \sqrt{m - m_0}$ tol. Die Zerlegung in Gleichung (3.4) wird auch RRQR-Zerlegung(Rank-Revealing QR-Zerlegung) genannt.

Wir teilen die Permutationsmatrix π aus (3.4) in eine Matrix $\pi^{\square} \in \mathbb{R}^{m \times m_0}$ und eine Matrix $\pi^{\triangle} \in \mathbb{R}^{m \times (m-m_0)}$ und zwar so, dass folgende Gleichungen erfüllt sind:

$$R_s\pi = R_s[\pi^{\square}, \pi^{\triangle}] = [R_s\pi^{\square}, R_s\pi^{\triangle}] = [B\pi^{\square}, B\pi^{\triangle}] - A[X_s\pi^{\square}, X_s\pi^{\triangle}]. \tag{3.5}$$

Wegen (3.4) gilt:

$$R_s\pi = [Y_1\theta, Y_1\theta^{\square} + Y_1^{\triangle}\theta^{\triangle}]. \tag{3.6}$$

Für den Neustart oder Beginn des Block-GMRES-Verfahren wird nur der Teil $Y_1\theta$ beachtet. Nachher wollen wir aber eine neue Approximation $X_s \in \mathbb{R}^{n \times m}$ haben. Wie wir zu dieser Lösung kommen wollen wir im Folgenden herleiten. Wir verwenden folgende Definition in den weiteren Überlegungen.

Definition 3.4. *Sei $\pi^{\square}$ und $\pi^{\triangle}$ wie in Gleichung (3.5), wobei s fix ist. Wir definieren:*

- $X_s^{\triangle} := X_s\pi^{\triangle}$.
- $X_*^{\triangle} := A^{-1}B\pi^{\triangle}$.
- $X_s^{\square} := X_s\pi^{\square}$.
- $X_*^{\square} := A^{-1}B\pi^{\square}$.

Aus der Definition folgt, dass $[X_*^{\square}, X_*^{\triangle}]\pi^T$ die Lösung des Gleichungssystems (1.1) ist.

Wir vergleichen jeweils die Blöcke von (3.6) und (3.5) in $R_s\pi$ multipliziert mit A^{-1}. Die beiden ersten Blöcke mit A^{-1} multipliziert und gleichgesetzt ergibt

$$A^{-1}Y_1\theta = A^{-1}B\pi^{\square} - X_s\pi^{\square} = X_*^{\square} - X_s^{\square}. \tag{3.7}$$

Bei den beiden zweiten Blöcken gilt nach Umformungen folgende Gleichheit:

$$\begin{aligned} X_*^{\triangle} &= A^{-1}B\pi^{\triangle} = X_s\pi^{\triangle} + (A^{-1}Y_1\theta)(\theta^{-1}\theta^{\square}) + A^{-1}Y_1^{\triangle}\theta^{\triangle} \\ &= X_s^{\triangle} + (A^{-1}Y_1\theta)(\theta^{-1}\theta^{\square}) + A^{-1}Y_1^{\triangle}\theta^{\triangle}. \end{aligned} \tag{3.8}$$

Angenommen wir haben die neue Approximation $X_{s+1}^{\square}$ bereits berechnet. Wir wollen mit dieser Approximation $X_{s+1}^{\triangle}$ ermitteln. Wir betrachten den letzten Term der Gleichung (3.8). Falls der Rang von R_s exakt m_0 ist, so ist $\theta^{\triangle} = 0$ und damit auch der betrachtete Term. Da wir aber auch bei einer fast lineare Abhängigkeit der Spalten Deflation anwenden ist dieser Term meist nicht 0. Um die Norm dieses Terms nach oben abzuschätzen müssen wir erst zeigen, dass die Frobeniusnorm submultiplikativ ist.

Lemma 3.5. *Für alle Matrizen* $C \in \mathbb{R}^{n\times m}$ *und* $D \in \mathbb{R}^{m\times l}$ *gilt:*

$$\|CD\|_F \leqslant \|C\|_F \|D\|_F .$$

Beweis. Wir beweisen das Lemma mithilfe der Cauchy-Schwarz Ungleichung:

$$\begin{aligned} \|CD\|_F^2 &= \sum_{i=1}^{n}\sum_{k=1}^{l}\left|\sum_{j=1}^{m} c_{i,j}d_{j,k}\right|^2 = \sum_{i=1}^{n}\sum_{k=1}^{l}\left|\left\langle (C^T)^{(i)}, D^{(k)}\right\rangle\right| \\ &\leqslant \sum_{i=1}^{n}\sum_{k=1}^{l}\left\|(C^T)^{(i)}\right\|_2^2\left\|D^{(k)}\right\|_2^2 = \sum_{i=1}^{n}\left\|(C^T)^{(i)}\right\|_2^2\sum_{k=1}^{l}\left\|D^{(k)}\right\|_2^2 \\ &= \left\|C^T\right\|_F^2\|D\|_F^2 = \|C\|_F^2\|D\|_F^2 \end{aligned}$$

□

Wir schätzen den letzten Term der Gleichung (3.8) folgendermaßen ab:

$$\left\|A^{-1}Y_1^{\triangle}\theta^{\triangle}\right\|_F \leqslant \left\|A^{-1}\right\|_F \left\|Y_1^{\triangle}\right\|_F \left\|\theta^{\triangle}\right\|_F \leqslant \left\|A^{-1}\right\|_F (m-m_0)\,\text{tol}. \tag{3.9}$$

Diesen Term werden wir in der Bestimmung von $X_{s+1}^{\triangle}$ vernachlässigen. Die letzte Ungleichung stimmt wegen:

$$\left\|Y_1^{\triangle}\right\|_F \leqslant \sqrt{m-m_0} \qquad \text{und} \qquad \left\|\theta^{\triangle}\right\|_F \leqslant \sqrt{m-m_0}\,\text{tol}.$$

Für die Berechnung von $X_{s+1}^{\triangle}$ verwenden wir, dass $X_{s+1}^{\square}$ eine Näherung von $X_*^{\square}$ ist:

$$\begin{aligned} X_*^{\triangle} &\overset{(3.7),\,(3.8),\,(3.9)}{\approx} X_s^{\triangle} + (X_*^{\square} - X_0^{\square})(\theta^{-1}\theta^{\square}) \\ &\approx X_s^{\triangle} + (X_{s+1}^{\square} - X_0^{\square})(\theta^{-1}\theta^{\square}) =: X_{s+1}^{\triangle}. \end{aligned}$$

Um die neue Approximation X_{s+1} zu bestimmen müssen wir die Spalten von

$$\widetilde{X}_{s+1} = [X_{s+1}^{\square}, X_{s+1}^{\triangle}]$$

noch in die richtige Position permutieren:

$$X_{s+1} = \widetilde{X}_{s+1}\pi^T. \tag{3.10}$$

Im Block-Arnoldi-Prozess werden wir nicht mit Deflation arbeiten, da sich der Aufwand dafür nicht lohnt. (Für ein kleines k wird nicht viel Rechenaufwand gespart.) Da aber die Spalten von Y_j linear abhängig werden können, lösen wir dieses Problem indem wir das Verfahren neu starten und die vorläufige Approximation X_{s+1} mit $Y_1, \ldots, Y_{j-1}$ berechnen, das heißt im Falle, dass die Spalten von Y_j linear abhängig sind gilt:

$$\mathcal{R}_s = [\mathcal{C}_1, \ldots, \mathcal{C}_{j-1}] \quad , \quad \mathcal{Y}_s = [Y_1, \ldots, Y_{j-1}] \quad \text{und} \quad d_s = [\delta_1, \ldots, \delta_{j-1}]^T$$
$$\text{mit} \qquad X_s = X_{s-1} + \mathcal{Y}_s \mathcal{R}_s^{-1} d_s.$$

Diese Überlegungen führen zu Algorithmus 7.

4 Numerische Beispiele

In diesem Kapitel wenden wir die beiden vorgestellten Block-Krylovraumverfahren auf bestimmte Gleichungssysteme an und analysieren für welche Matrizen das Block-CG-Verfahren und das Block-GMRES-Verfahren einen Vorteil gegenüber dem gewöhnlichen CG-Verfahren und dem gewöhnlichen GMRES-Verfahren haben und für welche sie nicht von Vorteil sind. Die Lösung der Block-Verfahren werden mit Algorithmus 3($M = E$) beziehungsweise Algorithmus 7 berechnet. Alle Rechnungen wurden mit *Matlab R2009b* unter *Windows Vista* auf einem *Intel(R) Core(TM)2 Duo CPU T5850 @ 2.16GHz 2,17 GHz* mit 2 GByte RAM und einem 32Bit Betriebssystem durchgeführt.

4.1 Block-CG-Verfahren und gewöhnliches CG-Verfahren

Für folgende Problemstellungen vergleichen wir das Block-CG-Verfahren mit dem gewöhnlichen CG-Verfahren:

- $A_1 \in \mathbb{R}^{2000\times 2000}$ mit den Eigenwerten

  $$0.5, 1 \ldots, 2.5, 1000001, 1000002, \ldots, 1001995, 1001996$$

  und $B = X_0 = E_1 \in \mathbb{R}^{2000\times m}$.
- $A_2 \in \mathbb{R}^{1600\times 1600}$ als Poisson-Matrix $B = X_0 = E_1 \in \mathbb{R}^{1600\times m}$.

A_1 wird konstruiert indem eine Matrix mit den Eigenwerten von A_1 in der Diagonale durch Koordinatentransformation mit der orthogonalen Matrix Q des *Matlab*-Befehls `gallery( 'orthog' , 2000 , 4 )` zur symmetrisch positiv definiten Matrix A_1 transformiert wird.

Algorithmus 7 Block-GMRES-Verfahren mit Neustart nach k Schritten und Anfangsdeflation

Input: $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}, X_0 \in \mathbb{R}^{n\times m}, s^*, s = 0, k\epsilon > 0$

$R_0 = B - AX_0$
Berechne θ
$d_0 = E_1\theta$
while $\|\rho_j\|_F > \epsilon$ und $s < s^*$ **do**
    $R_s = [Y_1, Y_1^{\triangle}] \begin{pmatrix} \theta & \theta^{\square} \\ 0 & \theta^{\triangle} \end{pmatrix} \pi^T$
    $\pi = [\pi^{\square}, \pi^{\triangle}]$
    $X_s^{\triangle} = X_s\pi^{\triangle}$
    $X_s^{\square} = X_s\pi^{\square}$
    $s \leftarrow s + 1$
    $j = 1$
    while $\|\rho_j\|_F > \epsilon$ und $j \leqslant k$ **do**
        Berechne $Y_j, h_{1,j}, \ldots, h_{j+1,j}$ nach Block-Arnoldi und einer Rang erhaltenen QR-Zerlegung mit Y_1 schon berechnet
        if Spaltenrang Y_j nicht voll nach Rang erhaltener QR-Zerlegung **then**
            Verlasse innere Schleife.
        end if
        Wende G_j auf $[h_{1,j}, \ldots, h_{j,j}]^T$ an$\rightsquigarrow c_{j-1}$
        Berechne $G_{j+1,j}$, das $h_{j+1,j}$ annulliert $\rightsquigarrow \mathcal{C}_j = (c_{j-1}, \gamma_j)^T$
        Wende $G_{j+1,j}$ auf $[\rho_{j-1}, 0]^T$ an $\rightsquigarrow \delta_j, \rho_j$
        $j \leftarrow j + 1$
    end while
    $\mathcal{R}_s = [\mathcal{C}_1, \ldots, \mathcal{C}_{j-1}]$, $\mathcal{Y}_s = [Y_1, \ldots, Y_{j-1}]$, $d_s = [\delta_1, \ldots, \delta_{j-1}]^T$
    $X_s^{\square} = X_{s-1}^{\square} + \mathcal{Y}_s\mathcal{R}_s^{-1}d_s$
    $X_s^{\triangle} = X_{s-1}^{\triangle} + (X_s^{\square} - X_0^{\square})(\theta^{-1}\theta^{\square})$
    $X_s = [X_s^{\square}, X_s^{\triangle}]\pi^T$
    Berechne $R_s,\ \theta,\ d_0$
end while

Output: X_s

Die Matrix A_2, mit dem *Matlab*-Befehl `gallery('poisson',40)` erzeugt, sieht folgendermaßen aus

$$A_2 := \begin{pmatrix} T & -I & 0 & \cdots & 0 \\ -I & T & -I & \ddots & \vdots \\ 0 & -I & T & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -I \\ 0 & \cdots & 0 & -I & T \end{pmatrix} \quad \text{mit} \quad T := \begin{pmatrix} 4 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & \ddots & \vdots \\ 0 & -1 & 4 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 4 \end{pmatrix} \in \mathbb{R}^{40\times 40}.$$

Nach Satz 2.12 ist A_1 so konstruiert, dass das Block-CG-Verfahren ab $m = 5$ einen klaren Vorteil gegenüber dem gewöhnlichen CG-Verfahren hat(λ_6 und λ_{2000} liegen relativ nah beieinander). In Abbildung 1(a), in der die Konvergenzzeiten der beide Verfahren verglichen werden, ist dieser Vorteil klar erkennbar.

Das gewöhnliche CG-Verfahren benötigt für $m = 10$ insgesamt **87** Iterationen, das sind im Durchschnitt **8.7** Iterationen um ein Gleichungssystem $AX^{(i)} = B^{(i)}$ zu lösen. In Abbildung 1(b) sehen wir, dass die Iterationen des Block-CG-Verfahren bei steigendem m bis auf 4 verringert werden. Obwohl das Lösen der Hilfsprobleme mit steigendem m schwieriger wird, garantiert die Größe des Block-Krylovraums bei A_1 eine schnellere Konvergenz des Block-CG-Verfahrens, da nur 4 Iterationen benötigt werden um die Lösung zu finden.

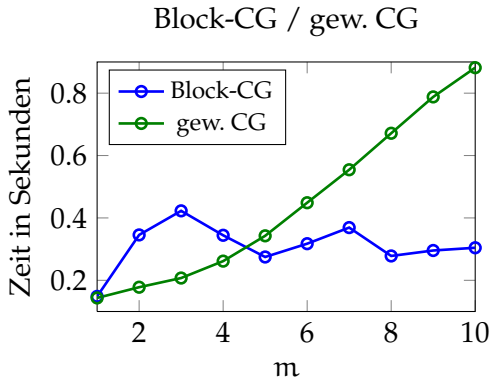
Die Matrix A_2 ist eine klassische, dünn besetzte, symmetrisch positiv definite Matrix. Wir sehen in Abbildung 2(a), dass das gewöhnliche CG-Verfahren beim Berechnen der Lösung des Gleichungssystems (1.1), für alle m, schneller konvergiert. Zurückzuführen ist dies auf die Anzahl der Iterationen der Verfahren. Während das gewöhnliche CG-Verfahren **739** Iterationen für $m = 10$ benötigt (im Durchschnitt **73.9**) sehen wir in Abbildung 2(b), dass die Iterationenanzahl beim Block-CG-Verfahren bei steigendem m sogar vergrößert wird, was auf numerische Ungenauigkeiten schließen lässt. Da das Lösen der Hilfsproblem im Block-CG-Verfahren bei steigendem m sehr aufwendig wird und die Iterationenanzahl in diesem Beispiel nicht kleiner wird, ist das gewöhnliche CG-Algorithmus die klar bessere Variante beim Lösen des Gleichungssystems (1.1).

Diese zwei Beispiele bestärken, dass die Verteilung der Eigenwerte bei der Wahl des Verfahrens eine große Rolle spielt.

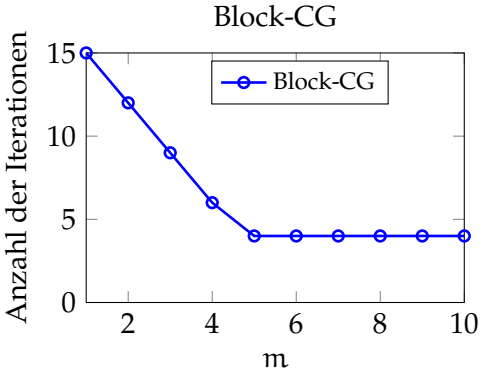
4.2 Block-GMRES-Verfahren und gewöhnliches GMRES-Verfahren

Im folgenden Abschnitt wollen wir das Block-GMRES-Verfahren mit dem gewöhnlichen GMRES-Verfahren vergleichen.

In der Praxis wird im gewöhnlichen GMRES-Algorithmus und im Block-GMRES-Algorithmus der Parameter k, der angibt wie oft die innere Schleife durchlaufen wird,

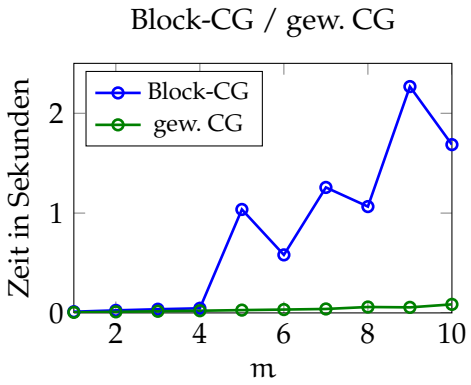


(a) Konvergenzzeitenvergleich bei A_1

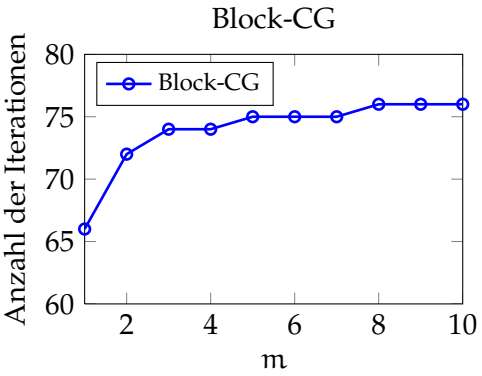


(b) Iterationen des Block-CG bei A_1

Abbildung 1: Vergleich der Konvergenzzeiten bei A_1 bzw. Iterationenanzahl des Block-CG bei A_1.



(a) Konvergenzzeitenvergleich bei A_2



(b) Iterationen des Block-CG bei A_2

Abbildung 2: Vergleich der Konvergenzzeiten bei A_2 bzw. Iterationenanzahl des Block-CG bei A_2.

zwischen 10 und 30 gewählt. In den meisten Beispielen, konvergiert das Block-GMRES-Verfahren schneller als das gewöhnliche GMRES-Verfahren. Zurückzuführen ist dies auf folgende Gründe:

- Das Lösen der Hilfsprobleme wird bei steigendem m nicht gravierend aufwendiger.
- Das Anwachsen des Block-Krylovraums bei steigendem m bringt oft eine signifikante Einsparung bei der Anzahl der Iterationen, denn der Suchraum ist durch $k \leqslant 30$ ohnehin begrenzt und wenn der Block-Krylovraum groß genug ist können viele Restarts eingespart werden.

Wir verwenden für den Vergleich folgende Beispiele mit A_3, A_4 aus Matrix Market[1]

- $A_3 \in \mathbb{R}^{765\times765}$(MCFE)[2], $X_0 = B = E_1 \in \mathbb{R}^{765\times m}$ und $k = 15$.
- $A_4 \in \mathbb{R}^{317\times317}$(CAVITY01)[3], $X_0 = E_1 \in \mathbb{R}^{317\times m}$, $k = 30$ und $B \in \mathbb{R}^{317\times m}$ ist eine Zufallsmatrix.

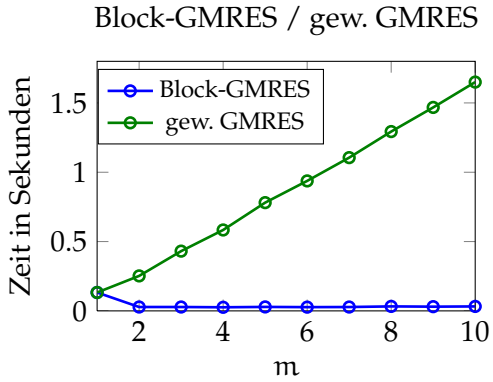
Für die Berechnung der Lösung des Gleichungssystems (1.1), mit $A = A_3$, bei $m = 10$, braucht das gewöhnliche GMRES-Verfahren insgesamt **400** Iterationen (der inneren Schleife) also im Durchschnitt **40** Iterationen um ein Gleichungssystem $AX^{(i)} = B^{(i)}$ zu lösen. In Abbildung 3(b) ist beim Block-GMRES-Verfahren der Rückgang der Anzahl der Iterationen mit steigendem m sehr schön zu sehen, was sich positiv auf die Konvergenzzeit des Block-GMRES-Verfahrens auswirkt die in Abbildung 3(a) visualisiert ist. Wir sehen in Abbildung 3(a), dass in diesem Beispiel für alle $1 < m \leqslant 10$ das Block-GMRES-Verfahren dem gewöhnlichen GMRES-Verfahren vorzuziehen ist.

Betrachten wir Abbildung 4(a) sehen wir, dass für $m = 2, 3$ das gewöhnliche GMRES-Verfahren schneller die Lösung des Gleichungssystems (1.1), mit $A = A_4$, berechnet als das Block-GMRES-Verfahren. Bei größerem m ist aber das Block-GMRES-Verfahren ganz klar das schnellere Verfahren. Das gewöhnliche GMRES-Verfahren benötigt, bei $m = 10$, zum Lösen des Gleichungssystems (1.1), mit $A = A_4$, **14335** Iterationen (im Durchschnitt **1433.5**). Da das Block-Verfahren nur bei $m = 1, 2, 3$ auch viele Iterationen benötigt, dann ab $m = 4$ aber nur mehr sehr wenige Iterationen braucht(Abbildung 4(b)) erklärt den Verlauf der Zeitkurve des Block-GMRES-Verfahrens in Abbildung 4(a). Aus dieser Überlegung folgt auch, dass das Block-GMRES-Verfahren bei $m = 10$ signifikant schneller X_* berechnet als bei $1 \leqslant m \leqslant 7$.

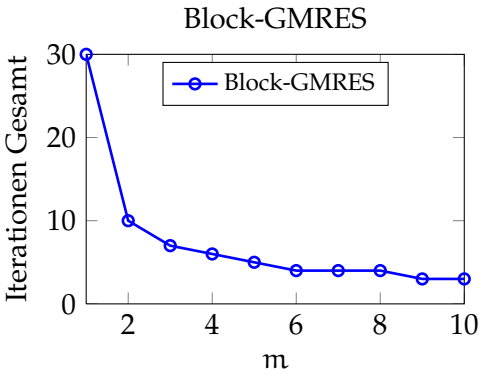
Schlussfolgerung

Die beiden, in dieser Arbeit betrachteten, Block-Krylovraumverfahren sind weitere Methoden das Gleichungssystem in (1.1) zu lösen. Ob eines der Verfahren dem sukzessiven

[1] http://math.nist.gov/MatrixMarket/
[2] http://math.nist.gov/MatrixMarket/data/Harwell-Boeing/astroph/mcfe.html
[3] http://math.nist.gov/MatrixMarket/data/SPARSKIT/drivcav_old/cavity01.html

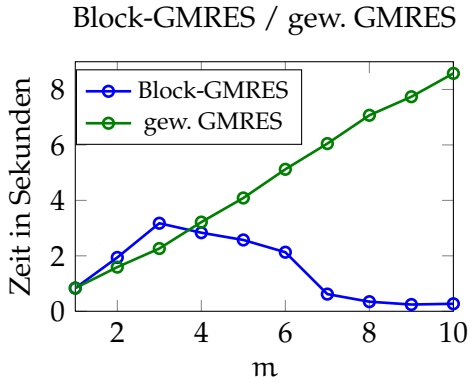


(a) Konvergenzzeitenvergleich des Block-GMRES und gew. GMRES bei A_3

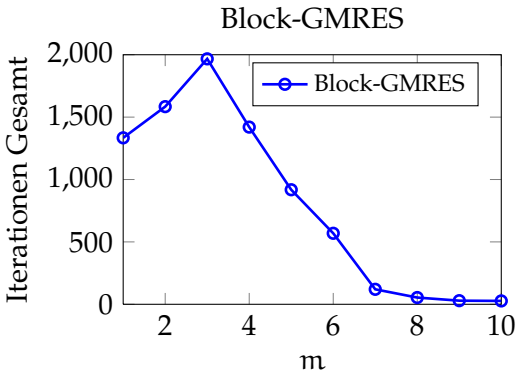


(b) Iterationen des Block-GMRES bei A_3

Abbildung 3: Vergleich der Konvergenzzeiten bei A_3 bzw. Iterationenanzahl des Block-GMRES bei A_3.



(a) Konvergenzzeitenvergleich des Block-GMRES und gew. GMRES bei A_4



(b) Iterationen des Block-GMRES bei A_4

Abbildung 4: Vergleich der Konvergenzzeiten bei A_4 bzw. Iterationenanzahl des Block-GMRES bei A_4.

Lösen durch das klassische CG-Verfahren oder klassische GMRES-Verfahren vorgezogen wird, hängt ganz stark von der Problemstellung ab. Oft ist auch nur a posteriori erkennbar welches Verfahren sich besser eignet, wobei in den meisten Fällen das Block-GMRES-Verfahren schneller die Lösung findet als der gewöhnliche GMRES-Algorithmus. Weitere Block-Krylovraumverfahren werden in verschiedenen Veröffentlichungen vorgestellt, wie zum Beispiel das Block-Bi-CG-Verfahren in [O'L80].

Literatur

[Cla12] Christian Clason. Numerische Mathematik I. Vorlesungsskript, "http://www.uni-graz.at/people/clason/teaching/numa12/NuMaSkript.pdf", Karl-Franzens-Universität Graz, Graz, Austria, 2012.

[Gut07] Martin H. Gutknecht. Block Krylov space methods for linear systems with multiple right-hand sides: An introduction. In A. H. Siddiqi, I. S. Duff, and O. Christensen, editors, *Modern Mathematical Models, Methods and Algorithms for Real World Systems*. Anamaya Publishers, New Delhi, http://www.sam.math.ethz.ch/~mhg/pub/delhipap.pdf, 2007.

[Hin01] Michael Hintermüller. Numerische Verfahren in der Optimierung. Vorlesungsskript, "http://www.uni-graz.at/imawww/hintermueller/optimierung.pdf, Karl-Franzens-Universität Graz, Graz, Austria, 2001.

[O'L80] Dianne P. O'Leary. The Block Conjugate Gradient Algorithm and Related Methods. In *Linear Algebra and its applications*, volume 29, pages 239–322. Elsevier North Holland, 1980.