Technische Universität Berlin
Institut für Mathematik

Bachelorarbeit
im Studiengang Technomathematik

Vorkonditionierte Krylov-Unterraum-Verfahren zur Lösung linearer Gleichungssysteme

Markus Wolff
323994

Betreut von: Prof. Dr. Volker John (WIAS Berlin)
Erstgutachter: Prof. Dr. Etienne Emmrich

Die selbstständige und eigenständige Anfertigung versichert an Eides statt:
Berlin, den

_____

Markus Wolff

Inhaltsverzeichnis

1 Einleitung

In dieser Arbeit beschäftigen wir uns mit numerischen Verfahren zur Lösung von linearen Gleichungssystemen. Für wenige Gleichungen kann man eine Lösung auf einem Blatt Papier bestimmen, aber wenn man ein System bestehend aus tausenden oder millionen von Gleichungen betrachtet, benötigt man spezielle Algorithmen.

Ein wichtiger Spezialfall sind dünnbesetzte Matrizen, das heißt die Einträge der Matrix, die das Gleichungssystem repräsentiert, sind meist Null. Systeme von dünnbesetzten Matrizen lassen sich leichter und schneller lösen als bei Systemen von vollbesetzten Matrizen, bei denen fast jeder Eintrag von Null verschieden ist. Dieser Spezialfall ist von praktischer Bedeutung, da diese dünnbesetzten Matrizen unter anderem bei der Lösung von Differentialgleichungen mit dem Finite-Elemente-Verfahren auftreten. Die Grundidee des Finite-Element-Verfahren ist, dass man die Gleichungen in einem endlich-dimensionalen Raum betrachtet. Als Basis des Raumes wählt man Funktionen mit einer besonderen Gestalt, um dünnbesetzte Matrizen zu erhalten. Jede Funktion des Raumes lässt sich durch eine Linearkombination der Basisfunktionen darstellen. Die Koeffizienten der Gewichte kann man in einem Vektor zusammenfassen. Damit lässt sich jede Funktion des Raumes durch einen Vektor darstellen. Damit man eine möglichst gute Lösung erhält, muss die Dimension des Raumes meist sehr groß gewählt werden und man erhält große lineare Gleichungssysteme.

Als Grundlage und Orientierung dieser Arbeit diente uns das Buch „Iterative Methods for Sparse Linear Systems“ von Y. Saad [1]. Dabei beschränken wir uns auf die drei Verfahren BiCGStab, CG und GMRES. Bei den drei Verfahren handelt es sich um Krylov-Unterraum-Verfahren, weswegen wir in Kapitel 3 eine kurze Einführung zu Krylov-Unterräumen geben werden. Ein wichtiges Mittel, die Lösung von linearen Gleichungssystemen zu beschleunigen, ist das Vorkonditionieren. Die Grundidee wird kurz in Kapitel 4 erklärt.

In Kapitel 5, dem Hauptteil dieser Arbeit, werden wir die drei Verfahren mit Vorkonditionierung herleiten und deren Vor- und Nachteile zusammenfassen. Die zum Testen der Algorithmen ausgewählten Beispiele werden in Kapitel 6 kurz erläutert und im nachfolgendem Kapitel werden die Ergebnisse ausgewertet.

2 Notation

In dieser Arbeit betrachten wir verschiedene Verfahren zur Bestimmung einer Lösung $x \in \mathbb{R}^n$ des Problems

$$Ax = b, \tag{1}$$

wobei $A \in \mathbb{R}^{n \times n}$ eine nicht singuläre und dünnbesetzte Matrix ist und der Vektor $b \in \mathbb{R}^n$ gegeben ist. Da A nicht singulär ist, gibt es für jedes $b \in \mathbb{R}^n$ eine eindeutige Lösung. Um die Güte einer berechneten Lösung x^* zu ermitteln, definieren wir das

Residuum

$$r^* := b - Ax^*. \tag{2}$$

Ist die Norm des Residuums Null, so stimmt unsere berechnete Lösung x^* mit der exakten Lösung überein. Die Hoffnung ist, dass mit kleiner werdender Norm des Residuums, auch die zugehörige berechnete Lösung stärker mit der exakten Lösung übereinstimmt. Als Norm wählen wir meist die eukldische Norm.

In dieser Arbeit ist mit $\|\cdot\|$ beziehungsweise $\langle\cdot,\cdot\rangle$ stets die euklidische Norm beziehungsweise das euklidische Skalarprodukt gemeint. Da wir oft Annahmen, Zuweisungen aus Algorithmen oder zuvor gezeigte Gleichheiten bei Beweisen oder Herleitungen benötigen, benutzen wir folgende Schreibweise

$$\text{Term A} \overset{\text{III}}{=} \text{Term B}.$$

Dies bedeutet, dass aus Term A mit der Zuweisung aus Zeile III des Algorithmus Term B wird. Römische Zahlen werden bei der Nummerierung von Zeilen der Algorithmen benutzt und arabische Zahlen bei der Nummerierung von Anforderungen und Gleichheiten.

Im Verlauf der Arbeit werden auch einige Eigenschaften von Matrizen benötigt, welche im folgenden kurz definiert werden.

Definition 2.1. *Sei A eine reelle $n \times m$ Matrix mit Einträgen $a_{i,j}$. Dann heißt A*

- *obere Dreiecksmatrix, falls $a_{i,j} = 0$ für $i > j$,*
- *obere Hessenbergmatrix, falls $a_{i,j} = 0$ für $i > j + 1$,*
- *quadratisch, falls $n = m$.*

Eine quadratische Matrix A heißt

- *symmetrisch, falls $A = A^T$, das heißt $a_{i,j} = a_{j,i}$ für alle $i, j = 1, 2, \ldots, n$,*
- *positiv definit, falls $x^T Ax > 0$ für alle $x \neq 0$.*

3 Krylov-Unterräume

Als erstes benötigen wir ein paar Definitionen.

Definition 3.1. *Seien A und b wie in* (1)*. Dann nennt man*

$$K_m(A, b) = \operatorname{span}\left\{b, Ab, \ldots, A^{m-1}b\right\}$$

den m-ten Krylov-Unterraum.

Definition 3.2. *Seien A und b wie oben und p ein nichttriviales Polynom kleinsten Grades mit*

$$p(A)b = 0.$$

Dann nennt man den Grad von p auch Grad von b bezüglich A, kurz grade(b).

Definition 3.3. *Seien $U \subseteq \mathbb{R}^n$ ein Unterraum und A wie oben. Dann ist U invariant unter A, falls für alle $x \in U$ gilt*

$$Ax \in U.$$

Damit lässt sich eine Aussage über die Dimension von Krylov-Unterräumen treffen.

Proposition 3.4. *Seien A,b wie oben und $k \in \mathbb{N}$ der Grad von b bezüglich A. Dann ist $K_k(A, b)$ invariant unter A und für alle $m \geq k$ gilt*

$$K_m(A, b) = K_k(A, b).$$

Beweis. Da k der Grad von b bezüglich A ist, existiert ein Polynom $p(x) = \sum_{i=0}^{k} \alpha_i x^i$ mit

- $p(A)b = \sum_{i=0}^{k} \alpha_i A^i b = 0$,
- $\alpha_k \neq 0$.

Somit ist

$$A^k b = -\frac{\sum_{i=0}^{k-1} \alpha_i A^i}{\alpha_k},$$

und damit ist $A^k b$ ein Element des Krylov-Unterraums $K_k(A, b)$. Also ist $K_k(A, b)$ invariant unter A. Damit folgt nun auch, dass $A^{k+1} b \in K_k(A, b)$ ist und so weiter. Somit ist für alle $m \geq k$ $A^m b \in K_k(A, b)$, also $K_m(A, b) = K_k(A, b)$. □

Proposition 3.5. *Seien A,b wie oben und $k \in \mathbb{N}$ der Grad von b bezüglich A. Dann gilt für die Dimension des m-ten Krylov-Unterraums*

$$\dim(K_m(A, b)) = \min\{m, k\}.$$

Des Weiteren gilt für $m \geq k$

$$K_m(A, b) = K_k(A, b)$$

und $K_m(A, b)$ ist invariant unter A.

Beweis. Wir zeigen, dass für $m \leq k$

$$dim(K_m(A, b)) = m$$

gilt. Der Rest der Aussage folgt dann mit Proposition 3.4.

Sei also $m \leq k$. Dann ist für alle Polynome $p(x) = \sum_{i=0}^{m-1} \alpha_i x^i$, die einen kleineren Grad als m haben, äquivalent

- $p(A)b = 0$,
- $p = 0$.

Insbesondere gilt also

$$\sum_{i=0}^{m-1} \alpha_i A^i b = 0,$$

genau dann, wenn für $i = 0, 1, \ldots, m-1$

$$\alpha_i = 0$$

gilt. Daraus folgt, dass $(b, Ab, \ldots, A^{m-1}b)$ eine Basis ist und somit gilt

$$dim(K_m(A, b)) = m.$$

□

Die Grundidee von Krylov-Unterraum-Verfahren ist, dass man in der i-ten Iteration eine Approximation der exakten Lösung $\hat{x}$ im i-ten Krylov-Unterraum sucht:

$$\hat{x} \approx x_i \in x_0 + K_i(A, r_0),$$

wobei x_0 der Startwert ist und $r_0 = b - Ax_0$ das Startresiduum. Somit erhält man nach spätestens $k = grade(r_0)$ Iterationen die exakte Lösung unter der Annahme, dass man exakt rechnet. Damit sind Krylov-Unterraum-Verfahren iterative Verfahren, das heißt, sie berechnen in jeder Iteration eine Approximation an die exakte Lösung. Im Gegensatz dazu liefern direkte Löser nur am Ende die Lösung und keine Zwischenlösungen.

Als Nächstes betrachten wir, wie man eine orthonormale Basis eines m-dimensionalen Krylov-Unterraums bestimmt. Dazu verwenden wir das Arnoldi-Verfahren (siehe Algorithmus 6.1 in [2]).

Proposition 3.6. *Angenommen der Algorithmus 3.1 läuft bis zum Ende durch. Dann bilden die Vektoren $v_1, v_2, \ldots, v_m$ eine orthonormal Basis des Krylov-Unterraums $K_m(A, v_1)$.*

Beweis. Die Vektoren sind normiert, da v_1 so gewählt wird und für v_j mit $j = 2, 3, \ldots, m$ gilt

$$\|v_j\| \overset{\text{XI}}{=} \left\| \frac{w_{j-1}}{h_{j,j-1}} \right\| \overset{\text{VII}}{=} \left\| \frac{w_{j-1}}{\|w_{j-1}\|} \right\| = 1.$$

Die Orthogonalität kann man mittels vollständiger Induktion nach den ersten Vektoren $v_1, v_2, \ldots, v_i$ zeigen. Für $i = 1$ ist die Aussage klar. Zeigen wir nun den

Algorithm 3.1: Arnoldi-Verfahren

```
I   Wähle v_1 mit ||v_1|| = 1
II  for j = 1, 2, ..., m − 1 do
III     for i = 1, 2, ..., j do
IV          h_{i,j} = ⟨A v_j, v_i⟩
V       end
VI      w_j = A v_j − Σ_{i=1}^{j} h_{i,j} v_i
VII     h_{j+1,j} = ||w_j||
VIII    if h_{j+1,j} = 0 then
IX          Stop
X       end
XI      v_{j+1} = w_j / h_{j+1,j}
XII end
```

Induktionsschritt von $i \to i+1$

$$\begin{aligned} h_{i+1,i}\langle v_{i+1}, v_l\rangle &\overset{\text{XI}}{=} \langle w_i, v_l\rangle \\ &\overset{\text{VI}}{=} \langle Av_i, v_l\rangle - \left\langle \sum_{k=1}^{i} h_{k,i} v_k, v_l \right\rangle \\ &= \langle Av_i, v_l\rangle - \sum_{k=1}^{i} h_{k,i} \underbrace{\langle v_k, v_l\rangle}_{\substack{=\delta_{k,l} \\ \text{I.V.}}} \\ &= \langle Av_i, v_l\rangle - h_{l,i} \overset{\text{IV}}{=} \langle Av_i, v_l\rangle - \langle Av_i, v_l\rangle = 0, \end{aligned}$$

wobei $l \in \{1, 2, \ldots, i\}$ beliebig gilt.

Als Letztes muss noch gezeigt werden, dass

$$\operatorname{span}\{v_1, v_2, \ldots, v_m\} = \operatorname{span}\{v_1, Av_1, \ldots, A^{m-1}v_1\}$$

gilt. Das heißt, für alle v_j mit $j \in \{1, 2, \ldots, m\}$ existiert ein Polynom p_{j-1} mit Grad $j-1$ mit $v_j = p_{j-1}(A)v_1$. Der Induktionsanfang für $j = 1$ ist klar. Zeigen wir nun den Induktionsschritt von $j \to j+1$

$$\begin{aligned} h_{j+1,j} v_{j+1} &\overset{\text{XI,VI}}{=} Av_j - \sum_{i=1}^{j} h_{i,j} v_i \\ &\overset{\text{I.V.}}{=} Ap_{j-1}(A)v_1 - \sum_{i=1}^{j} h_{i,j} p_{i-1}(A) v_1 \\ &= \underbrace{\left[Ap_{j-1}(A) - \sum_{i=1}^{j} h_{i,j} p_{i-1}(A) \right]}_{:= p_j(A)} v_1. \end{aligned}$$

□

Als nächstes zeigen wir noch einige Zusammenhänge zwischen den im Algorithmus 3.1 vorkommenden Matrizen. Diese werden wir später für GMRES benötigen.

Proposition 3.7. *Seien A eine $n \times n$ Matrix und $V_m = [v_1, v_2, \ldots, v_m]$, $\tilde{H}_m = \{h_{i,j}\}$ mittels des Algorithmus 3.1 berechnet. Weiter sei H_m eine $m \times m$ Matrix die durch Streichung der letzten Zeile von $\tilde{H}_m$ entsteht. Dann gelten folgende Gleichungen*

$$\begin{aligned} AV_m &= V_m H_m + w_m e_m^T = V_{m+1}\tilde{H}_m, \\ V_m^T A V_m &= H_m. \end{aligned}$$

Beweis. Zunächst erhalten wir wie zuvor im Beweis von Proposition 3.6 aus den Zeilen XI und VI des Algorithmus 3.1

$$h_{j+1,j} v_{j+1} = Av_j - \sum_{i=1}^{j} h_{i,j} v_i$$

und damit

$$Av_j = \sum_{i=1}^{j+1} h_{i,j} v_i.$$

Betrachten wir nun die j-te Spalte von $V_{m+1}\tilde{H}_m$, wobei $v_{i,j}$ für die i-te Komponente des Vektors v_j steht.

$$\begin{aligned} (V_{m+1}\tilde{H}_m)_j &= \begin{pmatrix} \sum_{i=1}^{m+1} v_{1,i} h_{i,j} \\ \vdots \\ \sum_{i=1}^{m+1} v_{n,i} h_{i,j} \end{pmatrix} \\ &= \sum_{i=1}^{m+1} v_i h_{i,j} \\ &= \sum_{i=1}^{j+1} v_i h_{i,j} + \underbrace{\sum_{i=j+2}^{m+1} v_i h_{i,j}}_{\substack{=0 \\ \text{da, } h_{i,j}=0 \text{ für } i>j+1}} \\ &= Av_j \\ &= (AV_m)_j, \\ (V_m H_m)_j &= \sum_{i=1}^{m} v_i h_{i,j} \\ &= \sum_{i=1}^{j+1} v_i h_{i,j} + \underbrace{\sum_{i=j+2}^{m} v_i h_{i,j}}_{\substack{=0 \\ \text{da, } h_{i,j}=0 \text{ für } i>j+1}} \\ &= \begin{cases} \sum_{i=1}^{j+1} v_i h_{i,j}, & \text{für } j = 1, 2, \ldots, m-1, \\ \sum_{i=1}^{m+1} v_i h_{i,j} - h_{m+1,m} v_{m+1}, & \text{für } j = m \end{cases} \\ &= \begin{cases} Av_j, & \text{für } j = 1, 2, \ldots, m-1, \\ Av_j - \underbrace{h_{m+1,m} v_{m+1}}_{:= w_m}, & \text{für } j = m \end{cases} \\ &= \left(AV_m - w_m e_m^T\right)_j. \end{aligned}$$

Die zweite Behauptung folgt aus der bereits bewiesenen Gleichung

$$AV_m = V_m H_m + w_m e_m^T$$

und der Orthonormalität von V_m bzw. V_{m+1}.

$$\begin{aligned} V_m^T A V_m &= \underbrace{V_m^T V_m}_{=\mathbb{1}} H_m + \underbrace{V_m^T w_m e_m^T}_{\substack{=0, \\ \text{Def. } w_m}} \\ &= H_m. \end{aligned}$$

□

Es stellt sich nun noch die Frage, ob das Verfahren vorzeitig abbricht. Dies geschieht nur, wenn der Grad von v_1 bezüglich A kleiner ist als m. Der Unterraum, der sich aus den gebildeten Vektoren ergibt, ist dann invariant unter A. Somit stellt ein vorzeitiger Abbruch des Verfahrens kein Problem dar.

Proposition 3.8. *Das Arnoldi-Verfahren bricht im Schritt j ab, genau dann, wenn $grade(v_1) = j$ gilt. Des Weiteren ist dann $K_j(A, v_1)$ invariant unter A.*

Beweis. Sei $grade(v_1) = j$ und angenommen der Algorithmus bricht nicht im j-ten Schritt vorzeitig ab. Dann kann v_{j+1} berechnet werden und wir erhalten

$$K_{j+1}(A, v_1) \supset K_j(A, v_1).$$

Dies ist aber ein Widerspruch zu Proposition 3.5. Breche nun das Verfahren im j-ten Schritt vorzeitig ab, das heißt $w_j = 0$. Somit gilt

$$0 = Av_j - \sum_{i=1}^{j} h_{i,j} v_i.$$

Da nach Proposition 3.6 die Vektoren $v_1, v_2, \ldots, v_j$ eine Basis des Krylov-Unterraums $K_j(A, v_1)$ bilden, existieren Polynome $p_{(1)}, p_{(2)}, \ldots, p_{(j)}$ mit maximal Grad $j - 1$, sodass für $i = 1, 2, \ldots, j$ gilt

$$v_i = p_{(i)}(A) v_1.$$

Damit ergibt sich

$$\begin{aligned} Av_j - \sum_{i=1}^{j} h_{i,j} v_i &= A p_{(j)}(A) v_1 - \sum_{i=1}^{j} h_{i,j} p_{(i)}(A) v_1 \\ &= \underbrace{\left(A p_{(j)}(A) - \sum_{i=1}^{j} h_{i,j} p_{(i)}(A) \right)}_{:= p(A)} v_1 \end{aligned}$$

und damit ist p ein nichttrivials Polynom j-ten Grades mit

$$p(A) v_1 = 0.$$

Damit ist nach der Definition des Grades

$$grade(v_1) \leq j.$$

Angenommen $grade(v_1) = k < j$. Wir haben gerade gezeigt, dass dann aber bereits $v_k = 0$ gelten muss. Somit hätte das Verfahren schon im k-ten Schritt abbrechen müssen. Damit folgt $grade(v_1) = j$. Das der entstehende Krylov-Unterraum invariant unter A ist, folgt aus Proposition 3.5. □

Damit haben wir nun alle Aussagen über Krylov-Unterräume, die wir benötigen.

4 Vorkonditionierung

Bevor wir uns den Verfahren zuwenden, erläutern wir noch kurz die verschiedenen Möglichkeiten der Vorkonditionierung um die iterativen Verfahren robuster und effektiver zu gestalten. Dazu betrachten wir anstelle des Problems

$$Ax = b$$

ein transformiertes Problem

$$M^{-1}Ax = M^{-1}b. \tag{3}$$

Dabei muss das System $Mx = b$ leicht zu lösen sein, da dies später in jeder Iteration des Verfahrens nötig ist. Zusätzlich sollte die Matrix M Ähnlichkeiten mit A haben.

Je nachdem von welcher Seite man das Ausgangsproblem vorkonditioniert, erhält man verschiedene transformierte Systeme. Das obige System wurde von links vorkonditioniert. Wenn man von rechts vorkonditioniert, erhält man folgendes System

$$AM^{-1}u = b \text{ mit } x = M^{-1}u. \tag{4}$$

Eine weitere Möglichkeit gibt es, falls sich M günstig faktorisieren lässt, zum Beispiel in eine obere und untere Dreiecksmatrix $M = M_L M_R$. Dann ergibt sich folgendes System

$$M_L^{-1}AM_R^{-1}u = M_L^{-1}b \text{ mit } x = M_R^{-1}u.$$

Des Weiteren unterscheidet man noch zwischen fester und flexibler Vorkonditionierung. Bei der festen Vorkonditionierung wird in jeder Iteration des Verfahrens die gleiche Vorkonditionierungsmatrix M benutzt. Dagegen kann bei der flexiblen Variante in jeder Iteration eine andere Vorkonditionierungsmatrix M_i benutzt werden.

Generell werden meist iterative Verfahren selbst als Vorkonditionierer benutzt, zum Beispiel das Jacobi-Verfahren oder SSOR. Für eine ausführlichere Einführung von Vorkonditionierern siehe beispielsweise Kapitel 10 in [1].

5 Verfahren

Nun widmen wir uns den speziellen Verfahren zur Lösung von linearen Gleichungssystemen.

5.1 GMRES

5.1.1 Grundalgorithmus

Als erstes betrachten wir das Generalized-Minimal-Residual-Verfahren, welches 1986 von Saad und Schultz [3] entwickelt wurde. Die Idee ist, dass man die Minimierung des Residuums $r = b - Ax$ auf ein Problem der kleinsten Quadrate zurückführt, indem man eine orthonormale Krylov-Basis konstruiert. Dazu benutzt man das Arnoldi-Verfahren. Als Startvektor v_1 wählt man dabei das normierte Startresiduum

$$v_1 = \frac{r_0}{\beta} = \frac{b - Ax_0}{\beta},$$

wobei β die Norm des Startresiduums r_0 ist. Dann minimiert man das Residuum über den von der Basis aufgespannten und um x_0 verschobenen Krylov-Unterraum. Jedes Element x lässt sich schreiben als

$$x = x_0 + V_m y,$$

wobei V_m die orthonormalen Basisvektoren als Spalten hat. Damit ergibt sich für das Residuum

$$\begin{aligned} r &= b - Ax \\ &= b - A(x_0 + V_m y) \\ &\overset{(2)}{=} r_0 - AV_m y \\ &\overset{\text{Prop.3.7}}{=} \beta v_1 - V_{m+1}\tilde{H}_m y \\ &= V_{m+1}(\beta e_1 - \tilde{H}_m y). \end{aligned}$$

Da V_{m+1} orthonormal ist, folgt für die Norm

$$\|r\| = \|\beta e_1 - \tilde{H}_m y\|.$$

Damit muss man nur noch folgendes Problem der kleinsten Quadrate lösen

$$y^* = \underset{y \in \mathbb{R}^m}{\operatorname{argmin}}\{\|\beta e_1 - \tilde{H}_m y\|\}$$

und die Lösung x^* lässt sich leicht berechnen aus

$$x^* = x_0 + V_m y^*.$$

Daraus ergibt sich Algorithmus 5.1 (s. Algorithmus 6.11 aus [1]), indem zuerst die Krylov-Basis berechnet wird und dann das Minimierungsproblem gelöst wird. Da das Residuum über ein pro Iteration wachsenden Unterraum minimiert wird, kann das Residuum in keiner Iteration wachsen. Jedoch haben Greenbaum,Pták und Strakoš [4] gezeigt, dass jeder andere Verlauf möglich ist. Insbesondere kann die Norm des Residuums über mehrere Iterationen konstant bleiben.

Algorithm 5.1: GMRES

I $r_0 = b - Ax_0, \beta = \|r_0\|$ und $v_1 = \frac{r_0}{\beta}$
II **for** $j = 1$ **to** m **do**
III $w_j = Av_j$
IV **for** $i = 1$ *to* j **do**
V $h_{i,j} = \langle w_j, v_i \rangle$
VI $w_j = w_j - h_{i,j} v_i$
VII **end**
VIII $h_{j+1,j} = \|w_j\|$
IX **if** $h_{j+1,j} = 0$ **then**
X $m = j$
XI Stop
XII **end**
XIII $v_{j+1} = \frac{w_j}{h_{j+1,j}}$
XIV **end**
XV Definiere $V_m := [v_1, \ldots, v_m]$, $\tilde{H}_m := \{h_{i,j}\}_{1 \leq i \leq m+1, 1 \leq j \leq m}$
XVI $y_m = \underset{y \in \mathbb{R}^m}{\operatorname{argmin}} \{\|\beta e_1 - \tilde{H}_m y\|\}$
XVII $x_m = x_0 + V_m y_m$
XVIII **if** *Lösung nicht gut genug* **then**
XIX $x_0 = x_m$ und starte neu bei I
XX **end**

Die einzige Möglichkeit eines vorzeitigen Abbruchs von GMRES ist, falls das Arnoldi-Verfahren vorzeitig abbricht. In diesem Fall hat man bereits die exakte Lösung.

Proposition 5.1. *Sei A eine invertierbare Matrix. Dann bricht GMRES vorzeitig in Iteration j ab, d.h.* $h_{j+1,j} = 0$*, genau dann, wenn die berechnete Lösung* x_j *bereits die exakte Lösung ist.*

Beweis. Siehe Kapitel 6.5.4 Proposition 6.10 im Buch von Y. Saad [1]. □

5.1.2 Fester Vorkonditionierer

Als nächstes betrachten wir nun das von rechts vorkonditionierte Problem (4), da sich aus diesem später eine flexible Variante ableiten lässt. Am ursprünglichen Al-

gorithmus muss nicht viel verändert werden. Das Startresiduum kann genauso wie vorher berechnet werden, da

$$r_0 = b - AM^{-1}u = b - Ax. \tag{5}$$

Bei der Berechnung der Krylov-Basis ersetzt man A durch AM^{-1}. Somit berechnet man eine orthonormale Basis des Krylov-Raumes $K_m(AM^{-1}, v_1)$. Die Minimumsberechnung verändert sich nicht. Die Lösung x_m des ursprünglichen Problems berechnet sich aus der Lösung u_m des vorkonditionierten Systems wie folgt

$$\begin{aligned} u_m &= u_0 + V_m y \\ x_m &= M^{-1}u_m \\ &= M^{-1}u_0 + M^{-1}V_m y \\ &= x_0 + M^{-1}V_m y. \end{aligned}$$

Somit benötigt man die Lösung des transformierten Systems nicht explizit und es ergibt sich der Algorithmus 5.2 (s. Algorithmus 9.5 aus [1]).

Algorithm 5.2: GMRES von rechts vorkonditioniert

I $r_0 = b - Ax_0, \beta = \|r_0\|$ und $v_1 = \frac{r_0}{\beta}$
II **for** $j = 1$ *to* m **do**
III $w = AM^{-1}v_j$
IV **for** $i = 1$ *to* j **do**
V $h_{i,j} = \langle w, v_i \rangle$
VI $w = w - h_{i,j}v_i$
VII **end**
VIII $h_{j+1,j} = \|w\|$
IX $v_{j+1} = \frac{w}{h_{j+1,j}}$
X **end**
XI Definiere $V_m := [v_1, \ldots, v_m]$, $\tilde{H}_m := \{h_{i,j}\}_{1 \le i \le m+1, 1 \le j \le m}$
XII $y_m = \underset{y \in \mathbb{R}^m}{\operatorname{argmin}} \{\|\beta e_1 - \tilde{H}_m y\|\}$
XIII $x_m = x_0 + M^{-1}V_m y_m$
XIV **if** *Lösung nicht gut genug* **then**
XV $x_0 = x_m$ und starte neu bei I
XVI **end**

5.1.3 Flexibler Vorkonditionierer

Um eine flexible Variante von GMRES zu erhalten, muss die Berechnung der Lösung verändert werden. Da M in jeder Iteration verschieden sein kann, ist die Berechnung der Lösung am Ende wie in Algorithmus 5.2 nicht mehr möglich. Die Lösung x_m ist eine Linearkombination der $M_i^{-1}v_i$. Diese berechnen wir bereits bei

der Bildung der orthonormalen Basis. Ist in jeder Iteration $M_i = M$, so speichern wir diese nicht explizit ab, da wir sie später aus den v_i erzeugen können mit Hilfe von M. Bei der flexiblen Variante geht dies nicht mehr. Somit speichert man die $M_i^{-1} v_i$ in einem Zwischenschritt ab. Damit ergibt sich der Algorithmus 5.3 (s. Algorithmus 9.6 aus [1]). Bei der Basis, die in der flexiblen Variante berechnet wird, handelt es sich dann nicht mehr um eine Krylov-Basis, da durch die mögliche Änderung des Vorkonditionierers in jedem Schritt der entstehende Raum kein Krylov-Unterraum mehr ist.

Algorithm 5.3: GMRES flexibel vorkonditioniert

I $r_0 = b - Ax_0, \beta = \|r_0\|$ und $v_1 = \frac{r_0}{\beta}$
II **for** $j = 1$ *to* m **do**
III $z_j = M_j^{-1} v_j$
IV $w = Az_j$
V **for** $i = 1$ *to* j **do**
VI $h_{i,j} = \langle w, v_i \rangle$
VII $w = w - h_{i,j} v_i$
VIII **end**
IX $h_{j+1,j} = \|w\|$
X $v_{j+1} = \frac{w}{h_{j+1,j}}$
XI **end**
XII Definiere $Z_m := [z_1, \ldots, z_m]$, $\tilde{H}_m := \{h_{i,j}\}_{1 \le i \le m+1, 1 \le j \le m}$
XIII $y_m = \underset{y \in \mathbb{R}^m}{\operatorname{argmin}} \{\|\beta e_1 - \tilde{H}_m y\|\}$
XIV $x_m = x_0 + Z_m y_m$
XV **if** *Lösung nicht gut genug* **then**
XVI $x_0 = x_m$ und starte neu bei I
XVII **end**

Damit für die berechnete Lösung x_m folgendes gilt

$$\begin{aligned} \|b - Ax_m\| &\overset{(XIV)}{=} \|b - A(x_0 + Z_m y_m)\| \\ &\overset{(2)}{=} \|r_0 - AZ_m y_m\| \\ &= \|\beta v_1 - V_{m+1} \tilde{H}_m y_m\| \\ &= \|V_{m+1}\| \cdot \|\beta e_1 - \tilde{H}_m y_m\| \\ &= \|\beta e_1 - \tilde{H}_m y_m\|, \end{aligned}$$

benötigen wir

$$AZ_m = V_{m+1} \tilde{H}_m.$$

Der Beweis ist ähnlich zum Beweis von Proposition 3.7. Es gilt

$$Az_j = \sum_{i=1}^{j+1} h_{i,j} v_i$$

da

$$h_{j+1,j}v_{j+1} = w = Az_j - \sum_{i=1}^{j} h_{i,j}v_i.$$

Dann betrachtet man die j-te Spalte der Matrix $V_{m+1}\tilde{H}_m$ und es folgt die Behauptung analog wie in Proposition 3.7.

Für die exakte Lösung im Falle eines vorzeitigen Abbruchs der flexiblen Variante benötigt man zusätzlich noch die Invertierbarkeit der Matrix H, da dies nicht mehr aus der Invertierbarkeit von A folgt.

Proposition 5.2. *Sei A invertierbar, die Norm des Startresiduums $\beta \neq 0$ und die ersten $j-1$ Iterationen des Algorithmus erfolgreich, d.h. $h_{i+1,i} \neq 0$ für $i < j$. Sei zusätzlich H_j invertierbar, wobei H_j durch Streichung der letzten Zeile von $\tilde{H}_j$ entsteht. Dann ist x_j exakt, genau dann, wenn $h_{j+1,j} = 0$ gilt.*

Beweis. Siehe Buch von Y. Saad [1] Kapitel 9.4.1 Proposition 9.3. □

5.1.4 Abschließende Bemerkungen zu GMRES

Bei den von uns vorgestellten Varianten von GMRES handelt es sich um die „restarted“ Version von GMRES, das heißt, wie in den Algorithmen zu sehen, fängt man nach m Schritten neu an. Der Grund dafür ist Speicherplatz zu sparen. Der Nachteil ist, dass der Algorithmus stagnieren kann, falls die Matrix A nicht positiv definit ist. Um die Stagnation zu vermeiden wird die Vorkonditionierung benutzt, da diese die Anzahl der benötigten Iterationen reduzieren soll. Als Letztes stellt sich die Frage, wie man das Problem der kleinsten Quadrate löst. Zusätzlich liefern unsere bisher vorgestellten Algorithmen nicht explizit in jedem Schritt eine Approximation an die Lösung. Dafür sei auf das Kapitel 6.5.3 „Practical Implementation Issues“ im Buch von Y. Saad [1] verwiesen. Dort wird erklärt, wie man das Problem der kleinsten Quadrate lösen kann. Dazu formt man mittels Rotationsmatrizen die obere Hessenbergmatrix $\tilde{H}$ in eine obere Dreiecksmatrix um. Zusätzlich liefert diese Methode auch das Residuum in jedem Schritt, wodurch es möglich ist den Algorithmus zu beenden, sobald das Residuum klein genug ist.

5.2 CG-Verfahren

5.2.1 Grundalgorithmus

Nun betrachten wir das Verfahren der konjugierten Gradienten. Es wurde 1952 von Hestenes und Stiefel [5] zur Lösung von linearen Gleichungssystemen, deren Matrix A symmetrisch und positiv definit ist, entwickelt. Man nutzt diese zusätzlichen Eigenschaften der Matrix aus, um ein äquivalentes Problem zu formulieren.

Proposition 5.3. *Seien $A \in \mathbb{R}^{n\times n}$ symmetrisch und positiv definit und $b \in \mathbb{R}^n$. Dann sind äquivalent*

1. *x^* löst $Ax = b$,*

2. x^* *minimiert* $f(x) \coloneqq \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$.

Beweis. Als erstes zeigen wir, dass $Ax - b$ der Gradient von f ist.

$$\begin{aligned} f(x) &= \frac{1}{2}x^T Ax - b^T x \\ &= \frac{1}{2}\sum_{k=1}^n x_k \sum_{i=1}^n a_{k,i}x_i - \sum_{k=1}^n b_k x_k \\ \frac{\partial f(x)}{\partial x_j} &= \frac{1}{2}\sum_{i=1}^n a_{j,i}x_i + \frac{1}{2}\sum_{k=1}^n a_{k,j}x_k - b_j \\ &\overset{\text{A symm.}}{=} \sum_{i=1}^n a_{j,i}x_i - b_j \\ &= (Ax-b)_j. \end{aligned}$$

Damit können wir nun die Äquivalenz zeigen. Sei x^* ein Minimum von f. Dann muss x^* eine Nullstelle des Gradienten von f sein. Das heißt

$$Ax^* - b = 0$$

und somit ist x^* eine Lösung des Problems $Ax = b$. Sei nun x^* eine Lösung von $Ax = b$. Damit hat f im Punkt x^* ein Extremum und da A positiv definit ist, muss es sich dabei um ein Minimum handeln. □

Die Grundidee von CG ist es f zu minimieren, indem man pro Iteration eine Suchrichtung p_i hinzufügt. Die neue Suchrichtung hängt von den vorherigen Suchrichtungen und dem aktuellen Residuum r_i ab. Der Algorithmus 5.4 lässt sich herleiten, indem man fordert, dass folgende Gleichungen gelten

$$\langle r_{i+1}, r_i\rangle = 0, \tag{6}$$

$$\langle p_{i+1}, Ap_i\rangle = 0. \tag{7}$$

Als erste Suchrichtung wählt man das Startresiduum $r_0 = b - Ax_0$. Unsere alte Lösung x_i korrigieren wir mit Hilfe der Suchrichtung p_i

$$x_{i+1} = x_i + \alpha_i p_i.$$

Damit ergibt sich für das neue Residuum r_{i+1}

$$r_{i+1} = r_i - \alpha_i A p_i. \tag{8}$$

Als neue Suchrichtung ergibt sich dann

$$p_{i+1} = r_{i+1} + \beta_i p_i. \tag{9}$$

Wählt man

$$\alpha_i = \frac{\langle r_i, r_i\rangle}{\langle r_i, Ap_i\rangle}, \tag{10}$$

$$\beta_i = -\frac{\langle r_{i+1}, Ap_i\rangle}{\langle p_i, Ap_i\rangle}, \tag{11}$$

so sind die Forderungen ((6), (7)) erfüllt. Es gelten

$$\begin{aligned}
\langle r_{i+1}, r_i\rangle &\overset{(8)}{=} \langle r_i - \alpha_i A p_i, r_i\rangle \\
&= \langle r_i, r_i\rangle - \alpha_i \langle A p_i, r_i\rangle \\
&\overset{(10)}{=} 0, \\
\langle p_{i+1}, A p_i\rangle &\overset{(9)}{=} \langle r_{i+1} + \beta_i p_i, A p_i\rangle \\
&= \langle r_{i+1}, A p_i\rangle + \beta_i \langle p_i, A p_i\rangle \\
&\overset{(11)}{=} 0.
\end{aligned}$$

Außerdem gilt

$$\begin{aligned}
\langle A p_i, r_i\rangle &\overset{(9)}{=} \langle A p_i, p_i - \beta_{i-1} p_{i-1}\rangle \\
&= \langle A p_i, p_i\rangle - \beta_{i-1} \langle A p_i, p_{i-1}\rangle \\
&\overset{(7)}{=} \langle A p_i, p_i\rangle
\end{aligned}$$

und damit lässt sich α_i auch schreiben als

$$\alpha_i = \frac{\langle r_i, r_i\rangle}{\langle A p_i, p_i\rangle}. \tag{12}$$

Aus (8) folgt

$$A p_i = -\frac{r_{i+1} - r_i}{\alpha_i}. \tag{13}$$

Damit vereinfacht sich β_i zu

$$\begin{aligned}
\beta_i &\overset{(11)}{=} -\frac{\langle r_{i+1}, A p_i\rangle}{\langle p_i, A p_i\rangle} \\
&\overset{(13)}{=} -\frac{\left\langle r_{i+1}, -\frac{r_{i+1} - r_i}{\alpha_i}\right\rangle}{\langle p_i, A p_i\rangle} \\
&= \frac{1}{\alpha_i} \frac{\langle r_{i+1}, r_{i+1} - r_i\rangle}{\langle p_i, A p_i\rangle} \\
&\overset{(6)}{=} \frac{1}{\alpha_i} \frac{\langle r_{i+1}, r_{i+1}\rangle}{\langle p_i, A p_i\rangle} \\
&\overset{(12)}{=} \frac{\langle r_{i+1}, r_{i+1}\rangle}{\langle r_i, r_i\rangle}
\end{aligned}$$

und es ergibt sich der Algorithmus 5.4.

Algorithm 5.4: CG-Verfahren

I $r_0 = b - Ax_0$, $p_0 = r_0$
II **for** $i = 0, 1, \ldots$ **do**
III $\alpha_i = \frac{\langle r_i, r_i \rangle}{\langle p_i, Ap_i \rangle}$
IV $x_{i+1} = x_i + \alpha_i p_i$
V $r_{i+1} = r_i - \alpha_i A p_i$
VI **if** *Residuum* r_{i+1} *ist klein genug* **then**
VII Stop
VIII **end**
IX $\beta_i = \frac{\langle r_{i+1}, r_{i+1} \rangle}{\langle r_i, r_i \rangle}$
X $p_{i+1} = r_{i+1} + \beta_i p_i$
XI **end**

Proposition 5.4. *Für die Residuen $r_0, r_1, \ldots$ und Suchrichtugen $p_0, p_1, \ldots$ des Algorithmus 5.4 gelten*

1. *die Residuen sind orthogonal, das heißt $\langle r_i, r_j \rangle = 0, i \neq j$,*
2. *die Suchrichtungen sind A-konjugiert, das heißt $\langle p_i, Ap_j \rangle = 0, i \neq j$,*
3. $\langle p_i, r_j \rangle = \begin{cases} 0, i < j, \\ \langle r_i, r_i \rangle, i \geq j. \end{cases}$
4. $\langle r_i, Ap_i \rangle = \langle p_i, Ap_i \rangle$
5. $\langle r_i, Ap_j \rangle = 0$, *für $i \neq j$ und $i \neq j + 1$.*

Beweis. Siehe [5] Theorem 5.1. Der Beweis benötigt die folgende Propositon 5.5. □

Proposition 5.5. *Sei $p_0 = r_0$. Dann gilt für $k = 0, 1, \ldots$*

$$p_{k+1} = r_{k+1} + \frac{\|r_{k+1}\|^2}{\|r_k\|^2} p_k, \tag{14}$$

genau dann, wenn für $k = 0, 1, \ldots$ folgendes gilt:

$$p_k = \|r_k\|^2 \sum_{j=0}^{k} \frac{r_j}{\|r_j\|^2}. \tag{15}$$

Beweis. Gelte (15). Dann ist

$$\begin{aligned} p_{k+1} &\overset{(15)}{=} \|r_{k+1}\|^2 \sum_{j=0}^{k+1} \frac{r_j}{\|r_j\|^2} \\ &= \|r_{k+1}\|^2 \left(\frac{r_{k+1}}{\|r_{k+1}\|^2} + \sum_{j=0}^{k} \frac{r_j}{\|r_j\|^2} \right) \\ &= r_{k+1} + \frac{\|r_{k+1}\|^2}{\|r_k\|^2} \underbrace{\|r_k\|^2 \sum_{j=0}^{k} \frac{r_j}{\|r_j\|^2}}_{=p_k} \\ &= r_{k+1} + \frac{\|r_{k+1}\|^2}{\|r_k\|^2} p_k. \end{aligned}$$

Die Rückrichtung wird per vollständiger Induktion gezeigt. Gelte also (14). Dann ist nach Voraussetzung

$$\begin{aligned} p_0 &= r_0 \\ &= \|r_0\|^2 \sum_{j=0}^{0} \frac{r_j}{\|r_j\|^2} \\ p_1 &\overset{(14)}{=} r_1 + \frac{\|r_1\|^2}{\|r_0\|^2} p_0 \\ &= r_1 + \frac{\|r_1\|^2}{\|r_0\|^2} r_0 \\ &= \|r_1\|^2 \sum_{j=0}^{1} \frac{r_j}{\|r_j\|^2}. \end{aligned}$$

Die Behauptung gelte für $p_0, p_1, \ldots, p_k$. Dann folgt

$$\begin{aligned} p_{k+1} &\overset{(14)}{=} r_{k+1} + \frac{\|r_{k+1}\|^2}{\|r_k\|^2} p_k \\ &\overset{\text{I.V.}}{=} r_{k+1} + \frac{\|r_{k+1}\|^2}{\|r_k\|^2} \|r_k\|^2 \sum_{j=0}^{k} \frac{r_j}{\|r_j\|^2} \\ &= r_{k+1} + \|r_{k+1}\|^2 \sum_{j=0}^{k} \frac{r_j}{\|r_j\|^2} \\ &= \|r_{k+1}\|^2 \left(\frac{r_{k+1}}{\|r_{k+1}\|^2} + \sum_{j=0}^{k} \frac{r_j}{\|r_j\|^2} \right) \\ &= \|r_{k+1}\|^2 \left(\sum_{j=0}^{k+1} \frac{r_j}{\|r_j\|^2} \right). \end{aligned}$$

□

Nun wird noch gezeigt, dass die Wahl von α in Algorithmus 5.4 in der Tat f minimiert.

Proposition 5.6. *Für $i = 0, 1, \ldots$ seien α_i, x_i, r_i, p_i berechnet nach Algorithmus 5.4. Dann nimmt die Funktion $h\colon \mathbb{R} \to \mathbb{R}$*

$$\begin{aligned} h(\alpha) &\coloneqq f(x_i + \alpha_i p_i) \\ &= \frac{1}{2}\langle A(x_i + \alpha_i p_i), x_i + \alpha_i p_i\rangle - \langle b, x_i + \alpha_i p_i\rangle \end{aligned}$$

ihr Minimum bei

$$\alpha_i = \frac{\langle r_i, r_i\rangle}{\langle Ap_i, p_i\rangle}$$

an.

Beweis.

$$\begin{aligned} h(\alpha) \quad &= \quad \frac{1}{2}\langle A(x_i + \alpha_i p_i), x_i + \alpha_i p_i\rangle - \langle b, x_i + \alpha_i p_i\rangle \\ &= \quad \frac{1}{2}\Big(\langle Ax_i, x_i\rangle + \alpha_i\langle Ax_i, p_i\rangle + \alpha_i\langle Ap_i, x_i\rangle + \alpha_i^2\langle Ap_i, p_i\rangle\Big) \\ &\quad\quad -\langle b, x_i\rangle - \alpha_i\langle b, p_i\rangle \\ &\overset{\text{A symm.}}{=} \quad \frac{1}{2}\Big(\langle Ax_i, x_i\rangle + \alpha_i^2\langle Ap_i, p_i\rangle\Big) - \langle b, x_i\rangle + \alpha_i\langle Ax_i, p_i\rangle - \alpha_i\langle b, p_i\rangle \\ &\overset{(2)}{=} \quad \frac{1}{2}\Big(\langle Ax_i, x_i\rangle + \alpha_i^2\langle Ap_i, p_i\rangle\Big) - \langle b, x_i\rangle - \alpha_i\langle r_i, p_i\rangle \\ &\overset{\text{Prop.5.4}}{=} \quad \frac{1}{2}\Big(\langle Ax_i, x_i\rangle + \alpha_i^2\langle Ap_i, p_i\rangle\Big) - \langle b, x_i\rangle - \alpha_i\langle r_i, r_i\rangle \end{aligned}$$

Als Ableitungen ergeben sich

$$\begin{aligned} h'(\alpha) &= \alpha_i\langle Ap_i, p_i\rangle - \langle r_i.r_i\rangle, \\ h''(\alpha) &= \langle Ap_i, p_i\rangle. \end{aligned}$$

Da A positiv definit ist, ist

$$\langle Ap_i, p_i\rangle > 0$$

und somit liegt ein Minimum bei

$$\alpha_i = \frac{\langle r_i.r_i\rangle}{\langle Ap_i, p_i\rangle}$$

vor. □

5.2.2 Fester Vorkonditionierer

Nun betrachten wir das von links vorkonditionierte Problem (3). Zusätzlich fordern wir, dass die Vorkonditionierungsmatrix M symmetrisch und positiv definit ist. Damit können wir nun für $x, y \in \mathbb{R}^n$ ein neues Skalarprodukt

$$\langle x, y\rangle_M := \langle Mx, y\rangle = \langle x, My\rangle \tag{16}$$

und eine neue Norm

$$\|x\|_M := \sqrt{\langle x, x\rangle_M}$$

definieren. Bezüglich des neuen Skalarprodukts ist $M^{-1}A$ selbstadjungiert, das heißt

$$\left\langle M^{-1}Ax, y\right\rangle_M = \left\langle x, M^{-1}Ay\right\rangle_M \text{ für alle } x, y \in \mathbb{R}^n,$$

da

$$\begin{aligned}
\left\langle M^{-1}Ax, y\right\rangle_M &\overset{(16)}{=} \left\langle MM^{-1}Ax, y\right\rangle \\
&= \langle Ax, y\rangle \\
&= \left\langle x, A^T y\right\rangle \\
&\overset{\text{A symm.}}{=} \langle x, Ay\rangle \\
&= \left\langle x, MM^{-1}Ay\right\rangle \\
&\overset{(16)}{=} \left\langle x, M^{-1}Ay\right\rangle_M .
\end{aligned}$$

Anstelle des Problems (3) wird wieder ein äquivalentes Minimierungsproblem gelöst.

Proposition 5.7. *Seien $A, M \in \mathbb{R}^{n\times n}$ symmetrisch und positiv definit und $b \in \mathbb{R}^n$. Dann sind äquivalent*

1. *x^* löst $M^{-1}Ax = M^{-1}b$,*
2. *x^* minimiert $f_M(x) := \frac{1}{2}\left\langle M^{-1}Ax, x\right\rangle_M - \left\langle M^{-1}b, x\right\rangle_M$.*

Beweis. Es gelten

$$\begin{aligned}
f_M(x) &= \frac{1}{2}\left\langle M^{-1}Ax, x\right\rangle_M - \left\langle M^{-1}b, x\right\rangle_M \\
&\overset{(16)}{=} \frac{1}{2}\left\langle MM^{-1}Ax, x\right\rangle - \left\langle MM^{-1}b, x\right\rangle \\
&= \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle \\
&= f(x)
\end{aligned}$$

und

$$M^{-1}Ax = M^{-1}b \Longleftrightarrow Ax = b.$$

Somit folgt

$$\begin{aligned} x^* \text{ löst } M^{-1}Ax = M^{-1}b \quad &\Longleftrightarrow \quad x^* \text{ löst } Ax = b \\ &\overset{\text{Prop.5.3}}{\Longleftrightarrow} \quad x^* \text{ minimiert } f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle \\ &\Longleftrightarrow \quad x^* \text{ minimiert} \\ &\qquad\quad f_M(x) = \frac{1}{2}\left\langle M^{-1}Ax, x\right\rangle_M - \left\langle M^{-1}b, x\right\rangle_M . \end{aligned}$$

□

Um den Algorithmus zu erhalten, ändert man die Folgerungen (6) und (7) leicht ab

$$\langle z_{j+1}, z_j\rangle_M = 0, \tag{17}$$

$$\left\langle p_{i+1}, M^{-1}Ap_i\right\rangle_M = 0. \tag{18}$$

In jeder Iteration wird wieder die alte Lösung x_j durch eine Suchrichtung p_j verbessert, also

$$x_{j+1} = x_j + \alpha_j p_j.$$

Dadurch ergibt sich als neues Residuum z_{j+1}

$$z_{j+1} = z_j - \alpha_j M^{-1}Ap_j \tag{19}$$

beziehungsweise

$$z_{j+1} = M^{-1}b - M^{-1}Ax_{j+1} = M^{-1}(b - Ax_{j+1}) = M^{-1}r_{j+1}, \tag{20}$$

wobei r_{j+1} das Residuum des ursprünglichen Problems (1) ist. Die neue Suchrichtung p_{j+1} hängt wie zuvor von den alten Suchrichtungen und dem aktuellen Residuum z_{j+1} ab

$$p_{j+1} = z_{j+1} + \beta_j p_j. \tag{21}$$

Die Bestimmung von α_j und β_j erfolgt ähnlich wie zuvor

$$\begin{aligned} \langle z_{j+1}, z_j\rangle_M &\overset{(19)}{=} \left\langle z_j - \alpha_j M^{-1}Ap_j, z_j\right\rangle_M \\ &= \langle z_j, z_j\rangle_M - \alpha_j \left\langle M^{-1}Ap_j, z_j\right\rangle_M , \\ \left\langle p_{j+1}, M^{-1}Ap_j\right\rangle_M &\overset{(21)}{=} \left\langle z_{j+1} + \beta_j p_j, M^{-1}Ap_j\right\rangle_M \\ &= \left\langle z_{j+1}, M^{-1}Ap_j\right\rangle_M + \beta_j \left\langle p_j, M^{-1}Ap_j\right\rangle_M \end{aligned}$$

und aus (17) und (18) folgt

$$\alpha_j = \frac{\langle z_j, z_j \rangle_M}{\left\langle M^{-1}Ap_j, z_j \right\rangle_M},$$
$$\beta_j = -\frac{\left\langle z_{j+1}, M^{-1}Ap_j \right\rangle_M}{\left\langle p_j, M^{-1}Ap_j \right\rangle_M}.$$

Außerdem gilt

$$\begin{aligned}
\left\langle M^{-1}Ap_j, z_j \right\rangle_M &\overset{(21)}{=} \left\langle M^{-1}Ap_j, p_j - \beta_{j-1}p_{j-1} \right\rangle_M \\
&= \left\langle M^{-1}Ap_j, p_j \right\rangle_M - \beta_{j-1} \left\langle M^{-1}Ap_j, p_{j-1} \right\rangle_M \\
&\overset{M^{-1}A \text{ selbstadj.}}{=} \left\langle M^{-1}Ap_j, p_j \right\rangle_M - \beta_{j-1} \underbrace{\left\langle p_j, M^{-1}Ap_{j-1} \right\rangle_M}_{\overset{(18)}{=}0} \\
&= \left\langle M^{-1}Ap_j, p_j \right\rangle_M \qquad (22)
\end{aligned}$$

und somit gilt

$$\alpha_j = \frac{\langle z_j, z_j \rangle_M}{\left\langle M^{-1}Ap_j, p_j \right\rangle_M}. \qquad (23)$$

Aus (19) folgt

$$M^{-1}Ap_j = -\frac{z_{j+1} - z_j}{\alpha_j} \qquad (24)$$

und β_j vereinfacht sich zu

$$\begin{aligned}
\beta_j &= -\frac{\left\langle z_{j+1}, M^{-1}Ap_j \right\rangle_M}{\left\langle p_j, M^{-1}Ap_j \right\rangle_M} \\
&\overset{(24)}{=} \frac{\langle z_{j+1}, z_{j+1} - z_j \rangle_M}{\alpha_j \left\langle p_j, M^{-1}Ap_j \right\rangle_M} \\
&\overset{(23),(17)}{=} \frac{\langle z_{j+1}, z_{j+1} \rangle_M}{\langle z_j, z_j \rangle_M}.
\end{aligned}$$

Somit muss man nie direkt das neue Skalarprodukt ausrechnen, da

$$\alpha_j = \frac{\langle z_j, z_j \rangle_M}{\left\langle M^{-1}Ap_j, p_j \right\rangle_M} \overset{(16),(20)}{=} \frac{\langle r_j, z_j \rangle}{\langle Ap_j, p_j \rangle},$$
$$\beta_j = \frac{\langle z_{j+1}, z_{j+1} \rangle_M}{\langle z_j, z_j \rangle_M} \overset{(16),(20)}{=} \frac{\langle r_{j+1}, z_{j+1} \rangle}{\langle r_j, z_j \rangle}$$

und es ergibt sich der Algorithmus 5.5.

Algorithm 5.5: CG-Verfahren von links vorkonditioniert

I $r_0 = b - Ax_0, z_0 = M^{-1}r_0, p_0 = z_0$
II **for** $i = 0, 1, \ldots$ **do**
III $\alpha_i = \frac{\langle r_i, z_i \rangle}{\langle p_i, Ap_i \rangle}$
IV $x_{i+1} = x_i + \alpha_i p_i$
V $r_{i+1} = r_i - \alpha_i A p_i$
VI $z_{i+1} = M^{-1} r_{i+1}$
VII **if** *Residuum* z_{i+1} *ist klein genug* **then**
VIII Stop
IX **end**
X $\beta_i = \frac{\langle r_{i+1}, z_{i+1} \rangle}{\langle r_i, z_i \rangle}$
XI $p_{i+1} = z_{i+1} + \beta_i p_i$
XII **end**

Ähnlich wie bei Algorithmus 5.4 kann man zeigen, dass die Wahl von α im Algorithmus 5.5 f_M minimiert. Dazu benötigen wir eine für den vorkonditionierten Fall angepasste Version von Proposition 5.4 und 5.5.

Proposition 5.8. *Sei $p_0 = z_0$. Dann gilt für $k = 0, 1, \ldots$*

$$p_{k+1} = z_{k+1} + \frac{\|z_{k+1}\|_M^2}{\|z_k\|_M^2} p_k,$$

genau dann, wenn für $k = 0, 1, \ldots$ folgendes gilt:

$$p_k = \|z_k\|_M^2 \sum_{j=0}^{k} \frac{z_j}{\|z_j\|_M^2}.$$

Beweis. Der Beweis geht analog zum Beweis von Proposion 5.5. Es muss im Beweis nur das euklidische Skalarprodukt $\|.\|$ durch das neue Skalarprodukt $\|.\|_M$ und r_k durch z_k ersetzt werden. □

Damit lassen sich nun ähnliche Aussagen, zu denen in Proposition 5.4, über die Suchrichtungen und Residuen des Algorithmus 5.5 treffen.

Proposition 5.9. *Für die Residuen $z_0, z_1, \ldots$ und die Suchrichtungen $p_0, p_1, \ldots$ des Algorithmus 5.5 gelten*

a) $\langle z_i, z_j \rangle_M = 0, i \neq j,$

b) $\left\langle p_i, M^{-1} A p_j \right\rangle_M = 0, i \neq j,$

c) $\langle p_i, z_j \rangle_M = \begin{cases} 0, i < j, \\ \|z_i\|_M^2, i \geq j, \end{cases}$

d) $\left\langle z_i, M^{-1}Ap_j \right\rangle_M = 0, i \neq j, i \neq j+1,$

e) $\left\langle z_i, M^{-1}Ap_i \right\rangle_M = \left\langle p_i, M^{-1}Ap_i \right\rangle_M.$

Beweis. Die Gleichheit in e) wurde bereits bewiesen bei der Herleitung von α (siehe (22)). Die restlichen Aussagen werden per vollständiger Induktion gezeigt. Seien z_0, z_1 und p_0 nach dem Algorithmus 5.5 berechnet. Aus dem Algorithmus folgt

$$\begin{aligned}
\langle p_0, z_1\rangle_M &\overset{\text{I}}{=} \langle z_0, z_1\rangle_M \\
&\overset{\text{V,VI}}{=} \left\langle z_0, z_0 - \alpha_0 M^{-1}Ap_0 \right\rangle_M \\
&\overset{\text{I}}{=} \langle z_0, z_0\rangle_M - \alpha_0 \left\langle p_0, M^{-1}Ap_0 \right\rangle_M \\
&\overset{(23)}{=} 0, \\
\langle p_0, z_0\rangle_M &\overset{\text{I}}{=} \langle z_0, z_0\rangle = \|z_0\|_M^2
\end{aligned}$$

und damit gelten die Aussagen a) bis d) für z_0, z_1 und p_0. Gelten die Aussagen a) bis d) nun für $z_0, z_1, \ldots, z_k$ und $p_0, p_1, \ldots, p_{k-1}$.

Wir zeigen zunächst, dass man p_k hinzufügen kann und die Aussagen immer noch gelten. Es muss somit überprüft werden, dass folgende Behauptungen gelten

A) $\langle p_k, z_i\rangle_M = \|z_k\|_M^2, i \leq k,$

B) $\left\langle p_i, M^{-1}Ap_k \right\rangle_M = 0, i < k,$

C) $\left\langle z_i, M^{-1}Ap_k \right\rangle_M = 0, i < k.$

Es gilt A), da für $i < k$ folgt

$$\begin{aligned}
\langle p_k, z_i\rangle_M &\overset{\text{Prop.5.8}}{=} \left\langle \|z_k\|_M^2 \sum\nolimits_{j=0}^{k} \frac{z_j}{\|z_j\|_M^2}, z_i \right\rangle_M \\
&\overset{\text{a)}}{=} \left\langle \|z_k\|_M^2 \frac{z_i}{\|z_i\|_M^2}, z_i \right\rangle_M \\
&= \frac{\|z_k\|_M^2}{\|z_i\|_M^2} \langle z_i, z_i\rangle_M = \|z_k\|_M^2.
\end{aligned}$$

Aus A) folgt B), da für $i < k$ gilt

$$\begin{aligned}
\|z_k\|_M^2 &\overset{\text{A)}}{=} \langle z_{i+1}, p_k\rangle_M \\
&\overset{\text{V,VI}}{=} \langle z_i, p_k\rangle_M - \alpha_{i-1} \left\langle M^{-1}Ap_i, p_k \right\rangle_M \\
&\overset{\text{A)}}{=} \|z_k\|_M^2 - \alpha_{i-1} \left\langle M^{-1}Ap_i, p_k \right\rangle_M,
\end{aligned}$$

da $\alpha_{i-1} \neq 0$, folgt

$$\left\langle M^{-1}Ap_i, p_k \right\rangle_M = 0.$$

Damit folgt mit B) die Aussage C). Sei $i < k$ dann gilt

$$\begin{aligned} \left\langle z_i, M^{-1}Ap_k \right\rangle_M &\overset{\text{XI}}{=} \left\langle p_i, M^{-1}Ap_k \right\rangle_M - \beta_{i-1} \left\langle p_{i-1}, M^{-1}Ap_k \right\rangle_M \\ &\overset{\text{B)}}{=} 0. \end{aligned}$$

Somit gelten die Aussagen a) bis d) für $p_0, p_1, \ldots, p_k$ und $z_0, z_1, \ldots, z_k$. Damit man z_{k+1} hinzufügen kann, muss gezeigt werden, dass

I) $\langle z_{k+1}, z_i \rangle_M = 0, i \leq k$,

II) $\langle p_i, z_{k+1} \rangle_M = 0, i \leq k$,

III) $\left\langle z_{k+1}, M^{-1}Ap_i \right\rangle_M = 0, i < k$,

gelten. Es ist

$$\begin{aligned} \langle z_{k+1}, z_i \rangle_M &\overset{\text{V,VI}}{=} \langle z_k, z_i \rangle_M - \alpha_k \left\langle M^{-1}Ap_k, z_i \right\rangle_M \\ &= \begin{cases} 0, \text{ für } i < k, \text{ wegen a) und d)}, \\ \langle z_k, z_k \rangle_M - \alpha_k \left\langle M^{-1}Ap_k, z_k \right\rangle_M \overset{\text{e),(23)}}{=} 0, \text{ falls } i = k, \end{cases} \\ \langle p_i, z_{k+1} \rangle_M &\overset{\text{Prop.5.8}}{=} \left\langle \|z_i\|_M^2 \sum_{j=0}^{i} \frac{z_j}{\|z_j\|_M^2}, z_{k+1} \right\rangle_M \\ &\overset{\text{I)}}{=} 0, i \leq k, \\ \langle z_{k+1}, z_{i+1} \rangle_M &\overset{\text{V,VI}}{=} \langle z_{k+1}, z_i \rangle_M - \alpha_i \left\langle z_{k+1}, M^{-1}Ap_i \right\rangle_M \end{aligned}$$

und für $i < k$ gilt nach I)

$$\langle z_{k+1}, z_{i+1} \rangle_M = \langle z_{k+1}, z_i \rangle_M = 0,$$

wonach

$$\left\langle z_{k+1}, M^{-1}Ap_i \right\rangle_M = 0, i < k$$

gelten muss, da $\alpha_i \neq 0$ gilt. Somit kann z_{k+1} hinzugefügt werden und es folgt die Behauptung. □

Damit lässt sich zeigen, dass die Wahl von α_i die Funktion f_M minimiert.

Proposition 5.10. *Für $i = 0, 1, \ldots$ seien α_i, x_i, z_i, p_i berechnet nach Algorithmus 5.5. Dann nimmt die Funktion $h_M \colon \mathbb{R} \to \mathbb{R}$*

$$\begin{aligned} h_M(\alpha_i) &:= f_M(x_i + \alpha_i p_i) \\ &= \frac{1}{2} \left\langle M^{-1}A(x_i + \alpha_i p_i), x_i + \alpha_i p_i \right\rangle_M - \left\langle M^{-1}b, x_i + \alpha_i p_i \right\rangle_M \end{aligned}$$

ihr Minimum bei

$$\alpha_i = \frac{\langle z_i, z_i\rangle_M}{\langle M^{-1}Ap_i, p_i\rangle_M}$$

an.

Beweis.

$$\begin{aligned}
h_M(\alpha_i) &= \frac{1}{2}\left\langle M^{-1}A(x_i+\alpha_i p_i), x_i+\alpha_i p_i\right\rangle_M - \left\langle M^{-1}b, x_i+\alpha_i p_i\right\rangle_M \\
&= \frac{1}{2}\Big(\left\langle M^{-1}Ax_i, x_i\right\rangle_M + \alpha_i\left\langle M^{-1}Ax_i, p_i\right\rangle_M + \alpha_i\left\langle M^{-1}Ap_i, x_i\right\rangle_M \\
&\quad +\alpha_i^2\left\langle M^{-1}Ap_i, p_i\right\rangle_M\Big) - \left\langle M^{-1}b, x_i\right\rangle_M - \alpha_i\left\langle M^{-1}b, p_i\right\rangle_M \\
&\overset{\text{M}^{-1}\text{A selbstadj.}}{=} \frac{1}{2}\Big(\left\langle M^{-1}Ax_i, x_i\right\rangle_M + \alpha_i^2\left\langle M^{-1}Ap_i, p_i\right\rangle_M\Big) \\
&\quad -\left\langle M^{-1}b, x_i\right\rangle_M + \alpha_i\left\langle M^{-1}Ax_i, p_i\right\rangle_M - \alpha_i\left\langle M^{-1}b, p_i\right\rangle_M \\
&\overset{(20)}{=} \frac{1}{2}\Big(\left\langle M^{-1}Ax_i, x_i\right\rangle_M + \alpha_i^2\left\langle M^{-1}Ap_i, p_i\right\rangle_M\Big) \\
&\quad -\left\langle M^{-1}b, x_i\right\rangle_M - \alpha_i\langle z_i, p_i\rangle_M \\
&\overset{\text{Prop.5.9}}{=} \frac{1}{2}\Big(\left\langle M^{-1}Ax_i, x_i\right\rangle_M + \alpha_i^2\left\langle M^{-1}Ap_i, p_i\right\rangle_M\Big) \\
&\quad -\left\langle M^{-1}b, x_i\right\rangle_M - \alpha_i\langle z_i, z_i\rangle_M\,.
\end{aligned}$$

Als Ableitungen ergeben sich

$$\begin{aligned}
h'(\alpha) &= \alpha_i\left\langle M^{-1}Ap_i, p_i\right\rangle_M - \langle z_i.z_i\rangle_M\,, \\
h''(\alpha) &= \left\langle M^{-1}Ap_i, p_i\right\rangle_M = \langle Ap_i, p_i\rangle\,.
\end{aligned}$$

Da A positiv definit ist, ist

$$\langle Ap_i, p_i\rangle > 0$$

und somit liegt ein Minimum bei

$$\alpha_i = \frac{\langle z_i.z_i\rangle_M}{\langle M^{-1}Ap_i, p_i\rangle_M}$$

vor. □

5.2.3 Flexibler Vorkonditionierer

Jetzt betrachten wir noch den Fall, dass sich der Vorkonditionierer in jeder Iteration ändern kann, das heißt wir ersetzen im Algorithmus 5.5 M durch M_i. Damit ergibt sich das Residuum in der i-ten Iteration durch

$$z_i = M_i^{-1}(b - Ax_i) = M_i^{-1} r_i$$

beziehungsweise

$$z_i = z_{i-1} - \alpha_{i-1} M_i^{-1} A p_{i-1}.$$

Unsere Forderungen (17) und (18) bleiben gleich, jedoch ändert sich in jeder Iteration das Skalarprodukt

$$\langle z_{i+1}, z_i \rangle_{M_{i+1}} = 0, \tag{25}$$

$$\left\langle p_{i+1}, M_{i+1}^{-1} A p_i \right\rangle_{M_{i+1}} = 0. \tag{26}$$

Aus diesen Forderungen ergibt sich der Algorithmus 5.6. Diese Variante von CG wird auch IPCG (Inexact Preconditioned Conjugate Gradient) genannt. Diese Variante von CG ist besonders stabil bezüglich flexibler Vorkonditionierung. Für eine genauere Analyse des Verfahrens siehe Golub und Ye „Inexact preconditioned conjugate gradient method with inner-outer iterations“ [9].

Algorithm 5.6: flexibles CG-Verfahren von links vorkonditioniert

I $r_0 = b - Ax_0, z_0 = M_0^{-1} r_0, p_0 = z_0$
II **for** $i = 0, 1, \ldots$ **do**
III $\alpha_i = \frac{\langle r_i, z_i \rangle}{\langle p_i, A p_i \rangle}$
IV $x_{i+1} = x_i + \alpha_i p_i$
V $r_{i+1} = r_i - \alpha_i A p_i$
VI $z_{i+1} = M_{i+1}^{-1} r_{i+1}$
VII **if** *Residuum* z_{i+1} *ist klein genug* **then**
VIII Exit
IX **end**
X $\beta_i = \frac{\langle r_{i+1} - r_i, z_{i+1} \rangle}{\langle r_i, z_i \rangle}$
XI $p_{i+1} = z_{i+1} + \beta_i p_i$
XII **end**

Aus (26) ergibt sich

$$\begin{aligned} 0 &= \left\langle p_{i+1}, M_{i+1}^{-1} A p_i \right\rangle_{M_{i+1}} \\ &= \langle p_{i+1}, A p_i \rangle \end{aligned} \tag{27}$$

$$\tag{28}$$

und damit folgt, wie zuvor

$$\begin{aligned}\left\langle M_{i+1}^{-1}Ap_i, z_i\right\rangle_{M_{i+1}} &\overset{\text{XI}}{=} \left\langle M_{i+1}^{-1}Ap_i, p_i\right\rangle_{M_{i+1}} - \beta_{i-1}\left\langle M_{i+1}^{-1}Ap_i, p_{i-1}\right\rangle_{M_{i+1}} \\ &\overset{(16)}{=} \left\langle M_{i+1}^{-1}Ap_i, p_i\right\rangle_{M_{i+1}} - \beta_{i-1}\langle Ap_i, p_{i-1}\rangle \\ &\overset{(27)}{=} \left\langle M_{i+1}^{-1}Ap_i, p_i\right\rangle_{M_{i+1}} \end{aligned} \tag{29}$$

Aus Forderung (25) ergibt sich die Formel in Schritt III, denn

$$\begin{aligned} 0 &= \langle z_{i+1}, z_i\rangle_{M_{i+1}} \\ &\overset{\text{VI}}{=} \left\langle M_{i+1}^{-1}r_{i+1}, z_i\right\rangle_{M_{i+1}} \\ &\overset{\text{V}}{=} \left\langle M_{i+1}^{-1}r_i, z_i\right\rangle_{M_{i+1}} - \alpha_i\left\langle M_{i+1}^{-1}Ap_i, z_i\right\rangle_{M_{i+1}} \\ &\overset{(29)}{=} \left\langle M_{i+1}^{-1}r_i, z_i\right\rangle_{M_{i+1}} - \alpha_i\left\langle M_{i+1}^{-1}Ap_i, p_i\right\rangle_{M_{i+1}} \\ &= \langle r_i, z_i\rangle - \alpha_i\langle Ap_i, p_i\rangle. \end{aligned}$$

Zunächst folgt aus Forderung (26)

$$\begin{aligned} 0 &= \left\langle p_{i+1}, M_{i+1}^{-1}Ap_i\right\rangle_{M_{i+1}} \\ &\overset{\text{XI}}{=} \left\langle z_{i+1}, M_{i+1}^{-1}Ap_i\right\rangle_{M_{i+1}} + \beta_j\left\langle p_j, M_{i+1}^{-1}Ap_i\right\rangle_{M_{i+1}} \end{aligned} \tag{30}$$

und somit

$$\beta_i = -\frac{\left\langle z_{i+1}, M_{i+1}^{-1}Ap_i\right\rangle_{M_{i+1}}}{\left\langle p_j, M_{i+1}^{-1}Ap_i\right\rangle_{M_{i+1}}}. \tag{31}$$

Da

$$M_{i+1}^{-1}Ap_i \overset{\text{V,VI}}{=} \frac{M_{i+1}^{-1}(r_{i+1} - r_i)}{-\alpha_i}, \tag{32}$$

$$\tag{33}$$

kann man β_i vereinfachen zu

$$\begin{aligned}\beta_i &\overset{(32)}{=} \frac{\left\langle z_{i+1}, M_{i+1}^{-1}(r_{i+1} - r_i)\right\rangle_{M_{i+1}}}{\alpha_i\left\langle p_i, M_{i+1}^{-1}Ap_i\right\rangle_{M_{i+1}}} \\ &\overset{(16)}{=} \frac{\langle z_{i+1}, (r_{i+1} - r_i)\rangle}{\alpha_i\langle p_i, Ap_i\rangle} \\ &\overset{\text{III}}{=} \frac{\langle z_{i+1}, (r_{i+1} - r_i)\rangle}{\langle z_i, r_i\rangle}. \end{aligned}$$

Um zu zeigen, dass die Wahl von α im Algorithmus 5.6 die Funktion f_{M_i} minimiert, benötigt man die folgende Proposition.

Proposition 5.11. *Seien z_i, p_i berechnet nach Algorithmus 5.6. Dann gilt*

$$\langle z_i, p_i\rangle_{M_i} = \langle z_i, z_i\rangle_{M_i}.$$

Beweis. Es gilt

$$\langle z_i, p_i\rangle_{M_i} \overset{\text{XI}}{=} \langle z_i, z_i\rangle_{M_i} + \beta_{i-1}\langle z_i, p_{i-1}\rangle_{M_i}.$$

Es bleibt somit zu zeigen, dass

$$\langle z_i, p_{i-1}\rangle_{M_i} = 0.$$

Dies gilt nach vollständiger Induktion, da für z_0, z_1, p_0 aus dem Algorithmus 5.6 gilt

$$\langle z_1, p_0\rangle_{M_1} = \langle z_1, z_0\rangle_{M_1} = 0.$$

Gelte also für $z_1, z_2, \ldots, z_k, p_0, p_1, \ldots, p_{k-1}$

$$\langle z_i, p_{i-1}\rangle_{M_i} = 0 \text{ für } i = 1, 2, \ldots, k.$$

Dann folgt

$$\begin{aligned}
\langle z_{k+1}, p_k\rangle_{M_{k+1}} &\overset{\text{VI,(16)}}{=} \langle r_{k+1}, p_k\rangle \\
&\overset{\text{V}}{=} \langle r_k, p_k\rangle - \alpha_k\langle Ap_k, p_k\rangle \\
&\overset{\text{III}}{=} \langle r_k, p_k\rangle - \langle r_k, z_k\rangle \\
&\overset{\text{XI}}{=} \langle r_k, p_k\rangle - \langle r_k, p_k\rangle + \beta_{k-1}\langle r_k, p_{k-1}\rangle \\
&\overset{\text{VI,(16)}}{=} \beta_{k-1}\langle z_k, p_{k-1}\rangle_{M_k} \\
&\overset{\text{I.V.}}{=} 0.
\end{aligned}$$

□

Proposition 5.12. *Für $i = 0, 1, \ldots$ seien α_i, x_i, z_i, p_i berechnet nach Algorithmus 5.6 und M_i seien symmetrische, positiv definite Matrizen. Dann nimmt die Funktion $h_{M_i}\colon \mathbb{R} \to \mathbb{R}$*

$$\begin{aligned}
h_{M_i}(\alpha_i) &:= f_{M_i}(x_i + \alpha_i p_i) \\
&= \frac{1}{2}\left\langle M_i^{-1}A(x_i + \alpha_i p_i), x_i + \alpha_i p_i\right\rangle_{M_i} - \left\langle M_i^{-1}b, x_i + \alpha_i p_i\right\rangle_{M_i}
\end{aligned}$$

ihr Minimum bei

$$\alpha_i = \frac{\langle z_i, r_i\rangle}{\langle Ap_i, p_i\rangle}$$

an.

Beweis. Der Beweis ist analog zum Beweis von Proposition 5.10. Man ersetze M durch M_i und anstelle von Proposition 5.9 benutzen wir Proposition 5.11. □

5.2.4 Abschließende Bemerkungen zu CG

Da CG ein Krylov-Unterraum-Verfahren ist, zeigen wir nun noch, dass die im m-ten Schritt berechnete Lösung x_m im um x_0 verschobenen Krylov-Unterraum $K_m(A, r_0)$ liegt. Es lässt sich zeigen, dass die Suchrichtungen den Krylov-Unterraum aufspannen.

Proposition 5.13. *Seien $p_0, p_1, \ldots, p_k, r_0, r_1, \ldots, r_k$ berechnet nach Algorithmus 5.4. Dann spannen die Suchrichtungen den Krylov-Unterraum auf. Es gilt also für $i = 0, 1, \ldots, k$*

$$\operatorname{span}\left\{r_0, Ar_0, \ldots, A^i r_0\right\} = \operatorname{span}\left\{p_0, p_1, \ldots, p_i\right\}.$$

Zusätzlich stammt das i-te Residuum aus dem Krylov-Unterraum $K_{i+1}(A, r_0)$.

Beweis. Wir zeigen die Aussagen per vollständiger Induktion. Der Induktionsanfang folgt direkt aus dem ersten Schritt des Algorithmus, da p_0 auf r_0 gesetzt wird. Gelte die Aussage der Proposition also für $i = 0, 1, \ldots, j$. Zunächst folgt mit Zeile V des Algorithmus

$$r_{j+1} = r_j - \alpha_j A p_j.$$

Nach der Induktionsvoraussetzung gilt

- $r_j \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^j r_0\right\}$,
- $p_j \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^j r_0\right\}$

und damit ist

$$Ap_j \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^{j+1} r_0\right\}.$$

Insgesamt ergibt sich also

$$r_{j+1} \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^{j+1} r_0\right\}.$$

Damit ist p_{j+1} ein Element des Krylov-Unterraums $K_{j+2}(A, r_0)$, da nach Zeile X des Algorithmus

$$p_{j+1} = r_{j+1} + \beta_j p_j$$

gilt, wobei nach Voraussetzung p_j aus dem Raum $K_{j+1}(A, r_0)$ und r_{j+1}, wie eben gezeigt, aus $K_{j+2}(A, r_0)$ sind.

Als Letztes zeigen wir, dass $A^{j+1} r_0$ im Spann der Suchrichtungen $p_0, p_1, \ldots, p_{j+1}$ ist. Für $i < j$ folgt aus der Voraussetzung

$$\operatorname{span}\left\{r_0, Ar_0, \ldots, A^i r_0\right\} = \operatorname{span}\left\{p_0, p_1, \ldots, p_i\right\}$$

direkt

$$Ap_i \in \operatorname{span}\{p_0, p_1, \ldots, p_{i+1}\}.$$

Aus Zeile X und V folgt

$$p_{j+1} = r_j - \alpha_j A p_j + \beta_j p_j$$

und somit ist

$$Ap_j \in \operatorname{span}\left\{p_0, p_1, \ldots, p_{j+1}\right\}.$$

Damit folgt $A^{j+1} r_0 \in \operatorname{span}\{p_0, p_1, \ldots, p_{j+1}\}$ aus

- $A^{j+1} r_0 = A A^j r_0$,
- $A^j r_0 \in \operatorname{span}\{p_0, p_1, \ldots, p_j\}$,
- $Ap_i \in \operatorname{span}\{p_0, p_1, \ldots, p_{i+1}\}$ für $i = 0, 1, \ldots, j$.

□

Nach der Definition des Residuums ist

$$\begin{aligned} r_m &= b - Ax_m \\ &\overset{(2)}{=} r_0 + Ax_0 - Ax_m \end{aligned}$$

und somit folgt

$$Ax_m = Ax_0 + \underbrace{r_0 - r_m}_{\in K_{m+1}(A, r_0)}.$$

Also ist x_m aus dem um x_0 verschoben Krylov-Unterraum $K_m(A, r_0)$

Analog kann man für den festvorkonditionierten Fall zeigen, dass die Suchrichtungen $p_0, p_1, \ldots, p_k$ den Krylov-Unterraum $K_{k+1}(M^{-1}A, r_0)$ aufspannen. Dazu ersetze man r_i durch z_i und A durch $M^{-1}A$.

Jetzt betrachten wir noch den Fall, dass es zu einem Abbruch des Algorithmus kommt, dass heißt $\langle r_i, r_i \rangle$ oder $\langle p_i, Ap_i \rangle$ sind Null. Im Fall $\langle r_i, r_i \rangle = 0$ folgt sofort, dass das Residuum der Nullvektor ist und damit ist unsere berechnete Lösung exakt. Betrachten wir also den Fall $\langle p_i, Ap_i \rangle = 0$. Da A symmetrisch und positiv definit ist, muss also p_i der Nullvektor sein. Aus dem Algorithmus 5.4 folgt somit nach X

$$0 = r_i + \beta_{i-1} p_{i-1}.$$

Aus Proposition 5.4 wissen wir, dass r_i senktrecht auf p_{i-1} steht. Somit muss $r_i = 0$ und $\beta_{i-1} p_{i-1} = 0$ gelten. Damit ist unsere berechnete Lösung exakt. Insgesamt ergibt sich damit, dass der Algorithmus 5.4 nur dann abbricht, wenn er die exakte Lösung berechnet hat. Dasselbe gilt, falls man fest oder flexibel vorkonditioniert. Man ersetze r_i durch z_i und A durch $M^{-1}A$ beziehungsweise $M_i^{-1}A$ für den flexiblen Fall. Die Argumentation ist dann analog wie zuvor. Man benötigt erneut, dass z_i senkrecht auf p_{i-1} steht. Dies wurde für den festen Fall in Proposition 5.9 und für den flexiblen Fall im Beweis von Proposition 5.11 gezeigt.

5.3 BiCGStab

5.3.1 Grundalgorithmus

Als letztes Verfahren betrachten wir BiCG-Stabilized (BiCGStab), welches 1992 von van der Vorst [6] erfunden wurde. Es ist abgeleitet von BiCG [7] und benötigt keine zusätzlichen Anforderungen an A wie dies bei CG der Fall war. Als Grundlage für unsere Herleitung benutzten wir zusätzlich zum Buch von Y. Saad den Artikel [6] von van der Vorst.

In BiCG werden zwei Folgen von Residuen und Suchrichtungen berechnet

$$r_i = P_i(A)r_0, \tag{34}$$

$$\hat{r}_i = P_i\left(A^T\right)\hat{r}_0, \tag{35}$$

$$p_{i+1} = T_i(A)r_0, \tag{36}$$

$$\hat{p}_{i+1} = T_i\left(A^T\right)\hat{r}_0, \tag{37}$$

wobei P_i, T_i Polynome vom Grade i sind.

Proposition 5.14. *Für die berechneten Residuen und Suchrichtungen gilt für $i \neq j$*

- *die Residuen sind biorthogonal, das heißt $\langle r_i, \hat{r}_j\rangle = 0$,*
- *die Suchrichtungen sind bikonjugiert bezüglich A, das heißt $\langle Ap_i, \hat{p}_j\rangle = 0$.*

Beweis. Siehe den Beweis auf Seite 81 von [7]. □

Daraus lässt sich auch der Name BiCG ableiten, da in CG die Residuen orthogonal und die Suchrichtungen A-konjugiert sind.

Die Idee ist nun, dass man das Residuum mit einem weiteren Polynom Q_i vom Grad i multipliziert

$$r_i = Q_i(A)P_i(A)r_0. \tag{38}$$

Wählt man $Q_i \equiv P_i$, so erhält man das von Sonneveld 1989 entwickelte CG-S Verfahren [8]. CG-S hat den Nachteil, dass es zu einem schlechtem Konvergenzverhalten kommen kann, wenn die Startlösung x_0 nahe an der exakten Lösung liegt.

Bei BiCGStab wählt man

$$Q_i(x) = (1 - \omega_i x)(1 - \omega_{i-1}x) \cdot \ldots \cdot (1 - \omega_1 x), \tag{39}$$

um eine möglichst einfache Rekursionsformel zu erhalten. Außerdem wählt man ω_i so, dass die euklidische Norm von r_i minimiert wird. Zunächst leiten wir die Rekursionsformeln für P_i, T_i aus dem BiCG Algorithmus (siehe Algorithmus 5.7) her.

Proposition 5.15. *Nach Algorithmus 5.7 gelten die folgenden Rekursionsformeln für $i = 1, 2, \ldots$*

$$P_i(A) = P_{i-1}(A) - \alpha_i A T_{i-1}(A), \tag{40}$$

$$T_i(A) = P_i(A) + \beta_{i+1} T_{i-1}(A). \tag{41}$$

Algorithm 5.7: BiCG-Verfahren

I $r_0 = b - Ax_0$, $p_0 = 0, \rho_0 = 1$
II $\hat{r}_0 = r_0, \hat{p}_0 = p_0$
III **for** $i = 1, 2, \ldots$ **do**
IV $\rho_i = \langle \hat{r}_{i-1}, r_{i-1} \rangle$
V $\beta_i = \frac{\rho_i}{\rho_{i-1}}$
VI $p_i = r_{i-1} + \beta_i p_{i-1}$
VII $\hat{p}_i = \hat{r}_{i-1} + \beta_i \hat{p}_{i-1}$
VIII $\alpha_i = \frac{\rho_i}{\langle \hat{p}_i, A p_i \rangle}$
IX $x_i = x_{i-1} + \alpha_i p_i$
X $r_i = r_{i-1} - \alpha_i A p_i$
XI **if** *Residuum r_i ist klein genug* **then**
XII Stop
XIII **end**
XIV $\hat{r}_i = \hat{r}_{i-1} - \alpha_i A^T \hat{p}_i$
XV **end**

Beweis.

$$
\begin{aligned}
P_i(A)r_0 &\overset{(34)}{=} r_i \\
&\overset{\text{X}}{=} r_{i-1} - \alpha_i A p_i \\
&\overset{(34),(36)}{=} P_{i-1}(A)r_0 - \alpha_i A T_{i-1}(A)r_0 \\
&= (P_{i-1}(A) - \alpha_i A T_{i-1}(A))r_0, \\
T_i(A)r_0 &\overset{(36)}{=} p_{i+1} \\
&\overset{\text{VI}}{=} r_i + \beta_{i+1} p_i \\
&\overset{(34),(36)}{=} P_i(A)r_0 + \beta_{i+1} T_{i-1}(A)r_0 \\
&= (P_i(A) + \beta_{i+1} T_{i-1}(A))r_0.
\end{aligned}
$$

□

Später benötigt man noch den Term mit der höchsten Potenz von A von $P_i(A)$ und $Q_i(A)$. Für $Q_i(A)$ sieht man direkt aus (39), dass dies

$$(-1)^i \omega_1 \cdot \ldots \cdot \omega_i A^i \tag{42}$$

ist. Für $P_i(A)$ folgt aus den eben bewiesenen Rekusionsformeln (40) und (41)

$$
\begin{aligned}
P_i(A) &\overset{40}{=} P_{i-1}(A) - \alpha_i A T_{i-1}(A) \\
&\overset{41}{=} P_{i-1}(A) - \alpha_i A P_{i-1}(A) - \beta_i \alpha_i A T_{i-2}(A).
\end{aligned}
$$

Somit enthält der Term $-\alpha_i A P_{i-1}(A)$ die höchste Potenz. Durch mehrmaliges Anweden der Rekursionsformeln (40) und (41), erhält man als höchste Potenz

$$(-1)^i \alpha_1 \cdot \ldots \cdot \alpha_i A^i. \tag{43}$$

Zusätzlich benötigen wir für den Algorithmus noch Rekursionsformeln für $Q_i(A)P_i(A)$ und $Q_i(A)T_i(A)$. Dazu wird die folgende Proposition benötigt.

Proposition 5.16. *Für alle Matrizen $A \in \mathbb{R}^{n\times n}$ und Polynome $S_i(x) = \sum_{k=0}^{i} s_k x^k$, $i \in \mathbb{N}$, gilt*

$$AS_i(A) = S_i(A)A.$$

Beweis. Es ist

$$\begin{aligned} AS_i(A) &= A \sum\nolimits_{k=0}^{i} s_k A^k \\ &= \sum\nolimits_{k=0}^{i} s_k A^{k+1} \\ &= \left(\sum\nolimits_{k=0}^{i} s_k A^k\right) A \\ &= S_i(A)A. \end{aligned}$$

□

Damit folgt nun

$$\begin{aligned} Q_i(A)P_i(A)r_0 &\overset{(40)}{=} Q_i(A)(P_{i-1}(A) - \alpha_i A T_{i-1}(A))r_0 \\ &\overset{(39)}{=} (1-\omega_i A)Q_{i-1}(A)(P_{i-1}(A) - \alpha_i A T_{i-1}(A))r_0 \\ &= [Q_{i-1}(A)(P_{i-1}(A) - \alpha_i A T_{i-1}(A)) \\ &\quad -\omega_i A Q_{i-1}(A)(P_{i-1}(A) - \alpha_i A T_{i-1}(A))]r_0 \\ &\overset{\text{Prop.5.16}}{=} [Q_{i-1}(A)P_{i-1}(A) - \alpha_i A Q_{i-1}(A) T_{i-1}(A)) \\ &\quad -\omega_i A(Q_{i-1}(A)P_{i-1}(A) - \alpha_i A Q_{i-1}(A) T_{i-1}(A))]r_0, \\ Q_i(A)T_i(A)r_0 &\overset{(41)}{=} Q_i(A)(P_i(A) + \beta_{i+1} T_{i-1}(A))r_0 \\ &\overset{(39)}{=} [Q_i(A)P_i(A) + \beta_{i+1}(1-\omega_i A)Q_{i-1}T_{i-1}(A)]r_0 \\ &= [Q_i(A)P_i(A) + \beta_{i+1}(Q_{i-1}T_{i-1}(A) - \omega_i A Q_{i-1}T_{i-1}(A))]r_0 \end{aligned}$$

Um die Parameter $\alpha_i, \beta_i, \omega_i, \rho_i$ zu bestimmen, benötigen wir noch die folgende Proposition.

Proposition 5.17. *Seien $r_0, \hat{r}_0 \in \mathbb{R}^n$ beliebig und $r_i = P_i(A)r_0$ und $p_i = T_{i-1}(A)r_0$ berechnet nach dem BiCG-Verfahren. Dann gilt*

$$P_i(A)r_0 \perp K_i\left(A^T, \hat{r}_0\right), \tag{44}$$

$$AT_i(A)r_0 \perp K_i\left(A^T, \hat{r}_0\right). \tag{45}$$

Beweis. Seien $P_i(x) = \sum_{k=0}^{i} b_k x^k, T_i(x) = \sum_{k=0}^{i} c_k x^k$. Aus Proposition 5.14 wissen wir, dass für $j < i$ gilt

$$\begin{aligned} 0 &= \langle A p_i, \hat{p}_j \rangle, \\ 0 &= \langle r_i, \hat{r}_j \rangle \end{aligned}$$

und damit

$$\begin{aligned} 0 &= \langle r_i, \hat{r}_j \rangle \\ &\overset{(34),(35)}{=} \left\langle P_i(A) r_0, P_j\left(A^T\right) \hat{r}_0 \right\rangle \\ &= \left\langle P_i(A) r_0, \sum_{k=0}^{j} b_k \left(A^T\right)^k \hat{r}_0 \right\rangle \\ &= \sum_{k=0}^{j} b_k \langle P_i(A) r_0, \left(A^T\right)^k \hat{r}_0 \rangle. \end{aligned}$$

Da dies für alle $j < i$ gilt, folgt damit

$$\left\langle P_i(A) r_0, \left(A^T\right)^j \hat{r}_0 \right\rangle = 0 \text{ für alle } j < i$$

und somit die Behauptung (44).

Die Aussage (45) folgt analog aus

$$\begin{aligned} 0 &= \langle A p_{i+1}, \hat{p}_{j+1} \rangle \\ &\overset{(36),(37)}{=} \left\langle A T_i(A) r_0, T_j\left(A^T\right) \hat{r}_0 \right\rangle \end{aligned}$$

□

Als Erstes betrachten wir die Parameter β_i und ρ_i. In BiCG ist

$$\begin{aligned} \rho_{i+1,BiCG} &\overset{\text{IV}}{=} \langle \hat{r}_i, r_i \rangle \\ &\overset{(34),(35)}{=} \left\langle P_i\left(A^T\right) \hat{r}_0, P_i(A) r_0 \right\rangle \end{aligned}$$

und da nach Proposition 5.17 $P_i(A) r_0$ senkrecht auf $K_i\left(A^T, \hat{r}_0\right)$ steht, benötigt man für die Berechnung nur die höchste Potenz von $P_i\left(A^T\right) \hat{r}_0$. Somit ist nach (43)

$$\rho_{i+1,BiCG} = \left\langle (-1)^i \alpha_1 \cdot \ldots \cdot \alpha_i \left(A^T\right)^i \hat{r}_0, P_i(A) r_0 \right\rangle.$$

Da man bei BiCGStab, wie in CG-S nur eine Folge von Residuen berechnen will, wählt man

$$\begin{aligned} \rho_{i+1,BiCGStab} &= \langle \hat{r}_0, r_i \rangle \\ &\overset{(38)}{=} \langle \hat{r}_0, Q_i(A) P_i(A) r_0 \rangle \\ &= \left\langle Q_i\left(A^T\right) \hat{r}_0, P_i(A) r_0 \right\rangle. \end{aligned}$$

Wie zuvor kann man die Orthogonalität aus Proposition 5.17 ausnutzen, so dass man nur die höchste Potenz von $Q_i\left(A^T\right)$ benötigt. Somit ist mit (42)

$$\rho_{i+1,BiCGStab} = \left\langle (-1)^i \omega_1 \cdot \ldots \cdot \omega_i \left(A^T\right)^i \hat{r}_0, P_i(A) r_0 \right\rangle.$$

Damit gilt die folgende Relation zwischen $\rho_{i+1,BiCG}$ und $\rho_{i+1,BiCGStab}$

$$\rho_{i+1,BiCG} = \frac{\alpha_1 \cdot \ldots \cdot \alpha_i}{\omega_1 \cdot \ldots \cdot \omega_i} \rho_{i+1,BiCGStab}. \tag{46}$$

Also ergibt sich

$$\beta_i = \frac{\rho_{i+1,BiCG}}{\rho_{i,BiCG}} = \frac{\rho_{i+1,BiCGStab}}{\rho_{i,BiCGStab}} \frac{\alpha_i}{\omega_i}.$$

Als nächstes betrachten wir α_i. In BiCG ist

$$\begin{aligned} \alpha_{i+1} &= \frac{\langle r_i, \hat{r}_i \rangle}{\langle A p_{i+1}, \hat{p}_{i+1} \rangle} \\ &\overset{(34)-(37)}{=} \frac{\left\langle P_i(A) r_0, P_i\left(A^T\right) \hat{r}_0 \right\rangle}{\left\langle A T_i(A) r_0, T_i\left(A^T\right) \hat{r}_0 \right\rangle}. \end{aligned}$$

Mit Proposition 5.17 folgt, dass nur die höchsten Potenz bedeutend ist. Diese stimmen wegen $T_i(A) = P_i(A) + \beta_{i+1} T_{i-1}(A)$ für P_i und T_i überein und somit ist

$$\alpha_{i+1} = \frac{\left\langle P_i(A) r_0, P_i\left(A^T\right) \hat{r}_0 \right\rangle}{\left\langle A T_i(A) r_0, P_i\left(A^T\right) \hat{r}_0 \right\rangle}$$

Wie zuvor bei ρ_i kann man mit der Orthogonalität aus Proposition 5.17 folgendes folgern

$$\begin{aligned} \left\langle A T_i(A) r_0, P_i\left(A^T\right) \hat{r}_0 \right\rangle &\overset{\text{Prop.5.17}}{=} \left\langle A T_i(A) r_0, (-1)^i \alpha_1 \cdot \ldots \cdot \alpha_i \left(A^T\right)^i \hat{r}_0 \right\rangle \\ &= \frac{\alpha_1 \cdot \ldots \cdot \alpha_i}{\omega_1 \cdot \ldots \cdot \omega_i} \left\langle A T_i(A) r_0, (-1)^i \omega_1 \cdot \ldots \cdot \omega_i \left(A^T\right)^i \hat{r}_0 \right\rangle \\ &\overset{\text{Prop.5.17}}{=} \frac{\alpha_1 \cdot \ldots \cdot \alpha_i}{\omega_1 \cdot \ldots \cdot \omega_i} \left\langle A T_i(A) r_0, Q_i\left(A^T\right) \hat{r}_0 \right\rangle. \end{aligned} \tag{47}$$

Damit folgt insgesamt

$$\begin{aligned} \alpha_{i+1} &\overset{(46),(47)}{=} \frac{\left\langle P_i(A) r_0, Q_i\left(A^T\right) \hat{r}_0 \right\rangle}{\left\langle A T_i(A) r_0, Q_i\left(A^T\right) \hat{r}_0 \right\rangle} \\ &\overset{\text{Prop.5.16}}{=} \frac{\langle Q_i(A) P_i(A) r_0, \hat{r}_0 \rangle}{\langle A Q_i(A) T_i(A) r_0, \hat{r}_0 \rangle}. \end{aligned}$$

Als Letztes minmieren wir noch die euklidische Norm des Residuums bezüglich ω_{i+1}. Dazu definieren wir

$$\begin{aligned} s &:= Q_i(A)P_i(A) - \alpha_{i+1}AQ_i(A)T_i(A)r_0, \\ h(\omega_{i+1}) &:= \|r_{i+1}\|^2 \\ &\overset{(38)}{=} \langle Q_{i+1}(A)P_{i+1}(A)r_0, Q_{i+1}(A)P_{i+1}(A)r_0\rangle \\ &\overset{(39),(40)}{=} \langle (1-\omega_{i+1}A)Q_i(A)(P_i(A) - \alpha_{i+1}AT_i(A))r_0, \\ &\quad (1-\omega_{i+1}A)Q_i(A)(P_i(A) - \alpha_{i+1}AT_i(A))r_0\rangle \\ &= \langle s - \omega_{i+1}As, s - \omega_{i+1}As\rangle \\ &= \langle s, s\rangle - 2\omega_{i+1}\langle As, s\rangle + \omega_{i+1}^2\langle As, As\rangle. \end{aligned}$$

Dann ist

$$\begin{aligned} h'(\omega_{i+1}) &= -2\langle As, s\rangle + 2\omega_{i+1}\langle As, As\rangle, \\ h''(\omega_{i+1}) &= 2\langle As, As\rangle = 2\|As\|^2 > 0. \end{aligned}$$

Damit liegt das Extremum bei

$$\omega_{i+1} = \frac{\langle As, s\rangle}{\langle As, As\rangle}.$$

Da $h''(\omega_{i+1}) > 0$ gilt, ist das Extremum ein Minimum. Ersetz man $Q_i(A)T_i(A)r_0$ durch p_i und $Q_i(A)P_i(A)r_0$ durch r_i, so erhält man Algorithmus 5.8.

5.3.2 Fester Vorkonditionierer

Wie schon bei GMRES, betrachten wir das von rechts vorkonditionierte System (4). Wie bereits bei GMRES gezeigt (siehe (5)), kann das Startresiduum des vorkonditionierten Systems genauso wie beim nicht vorkonditionierten System berechnet werden. Ersetzen wir nun im Rest des Algorithmus 5.8 A durch AM^{-1}, so ist die Lösung x_i, die Lösung des vorkonditionierten Problems. Die Lösung x_i^* des ursprünglichen Problems erhalten wir, indem wir von links mit M^{-1} multiplizieren

$$\begin{aligned} x_i^* &= M^{-1}x_i \\ &\overset{\text{XII}}{=} M^{-1}x_{i-1} + \alpha_i M^{-1}p_i + \omega_i M^{-1}s \\ &= x_{i-1}^* + \alpha_i M^{-1}p_i + \omega_i M^{-1}s. \end{aligned}$$

Da man meist nur an der Lösung des ursprünglichen Systems interessiert ist, speichern wir die Vektoren $M^{-1}p_i, M^{-1}s$ zwischen, damit wir sie nicht doppelt berechnen müssen. Daraus ergibt sich insgesamt der Algorithmus 5.9, wobei x_i die Lösung des ursprünglichen Problems ist. Bis auf die Bestimmung der Lösung entspricht damit Algorithmus 5.9 dem Algorithmus 5.8 angewendet auf das explizit vorkonditionierte Problem (4).

Algorithm 5.8: BiCGStab

I $r_0 = b - Ax_0$, $v_0 = p_0 = 0$, $\hat{r}_0 = r_0$
II $\rho_0 = \alpha = \omega_0 = 1$
III **for** $i = 1, 2, \ldots$ **do**
IV $\rho_i = \langle \hat{r}_0, r_{i-1} \rangle$
V $\beta_i = \frac{\rho_i}{\rho_{i-1}} \frac{\alpha}{\omega_{i-1}}$
VI $p_i = r_{i-1} + \beta_i (p_{i-1} - \omega_{i-1} v_{i-1})$
VII $v_i = Ap_i$
VIII $\alpha_i = \frac{\rho_i}{\langle \hat{r}_0, v_i \rangle}$
IX $s = r_{i-1} - \alpha_i v_i$
X $t = As$
XI $\omega_i = \frac{\langle t, s \rangle}{\langle t, t \rangle}$
XII $x_i = x_{i-1} + \alpha_i p_i + \omega_i s$
XIII $r_i = s - \omega_i t$
XIV **if** *Residuum* r_i *ist klein genug* **then**
XV Stop
XVI **end**
XVII **end**

Algorithm 5.9: BiCGStab von rechts vorkonditioniert

I $r_0 = b - Ax_0$, $v_0 = p_0 = 0$, $\hat{r}_0 = r_0$
II $\rho_0 = \alpha = \omega_0 = 1$
III **for** $i = 1, 2, \ldots$ **do**
IV $\rho_i = \langle \hat{r}_0, r_{i-1} \rangle$
V $\beta_i = \frac{\rho_i}{\rho_{i-1}} \frac{\alpha}{\omega_{i-1}}$
VI $p_i = r_{i-1} + \beta_i (p_{i-1} - \omega_{i-1} v_{i-1})$
VII $y = M^{-1} p_i$
VIII $v_i = Ay$
IX $\alpha_i = \frac{\rho_i}{\langle \hat{r}_0, v_i \rangle}$
X $s = r_{i-1} - \alpha_i v_i$
XI $z = M^{-1} s$
XII $t = Az$
XIII $\omega_i = \frac{\langle t, s \rangle}{\langle t, t \rangle}$
XIV $x_i = x_{i-1} + \alpha_i y + \omega_i z$
XV $r_i = s - \omega_i t$
XVI **if** *Residuum* r_i *ist klein genug* **then**
XVII Stop
XVIII **end**
XIX **end**

5.3.3 Flexibler Vorkonditionierer

Um eine flexible Variante von BiCGStab zu erhalten, muss man die feste Matrix M nur durch eine flexible Matrix M_i ersetzen. Dies liefert uns bereits den flexiblen Algorithmus, siehe Algorithmus 4.1 in [10].

5.3.4 Abschließende Bemerkungen zu BiCGStab

Van der Vorst weißt darauf hin, dass es bei BiCGStab, wie bei BiCG, zu einem vorzeitigen Abbruch kommen kann. Das Problem ist, dass die Bilinearform

$$B(x, y) := \left\langle P\left(A^T\right) x, P(A)y \right\rangle$$

kein Skalarprodukt definiert. Somit kann bei einer unglücklichen Wahl von $\hat{r}_0$ der Parameter ρ_i oder $\langle \hat{r}_0, v_i \rangle$ Null oder zumindest sehr klein werden, obwohl keine Konvergenz stattgefunden hat. Falls dieser Fall eintreten sollte, kann man entweder mit einem anderen $\hat{r}_0$ neustarten oder auf eine andere Methode, beispielsweise GMRES, wechseln. Ein anderer Vorschlag von van der Vorst ist, den Algorithmus leicht abzuändern. Sein Vorschlag (siehe Seite 637 [6]) ist in exakter Arithmetik äquivalent. Dieser muss jedoch noch genauer untersucht werden um die beste Variante zu finden.

Wie zuvor bei CG zeigen wir nun noch, dass die im m-ten Schritt berechnete Lösung x_m im um x_0 verschobenen Krylov-Unterraum $K_m(A, r_o)$ liegt. Dazu zeigen wir zunächst, dass das Residuum und die Suchrichtungen aus dem Krylov-Unterraum sind.

Proposition 5.18. *Seien $p_1, p_2, \ldots, p_k, r_0, r_1, \ldots, r_k$ berechnet nach Algorithmus 5.8. Dann liegen die Suchrichtungen im Krylov-Unterraum. Es gilt also für $i = 1, 2, \ldots, k$*

$$p_1, p_2, \ldots, p_i \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^{i-1} r_0\right\}.$$

Zusätzlich stammt das i-te Residuum aus dem Krylov-Unterraum $K_{i+1}(A, r_0)$.

Beweis. Wir zeigen die Aussagen per vollständiger Induktion. Der Induktionsanfang folgt direkt aus dem ersten Schritt des Algorithmus, da p_1 auf r_0 gesetzt wird. Gelte die Aussage der Proposition also für $i = 0, 1, \ldots, j$. Zunächst folgt aus den Zeilen XIII, X und IX des Algorithmus 5.8

$$\begin{aligned} r_{j+1} &= r_j - \alpha_j A p_j - \omega_j A\left(r_j - \alpha_j A p_j\right) \\ &= \left(1 - \omega_j A\right)\left(r_j - \alpha_j A p_j\right). \end{aligned}$$

Nach der Induktionsvoraussetzung gilt

- $r_j \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^j r_0\right\}$,
- $p_j \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^{j-1} r_0\right\}$

und damit ist

$$r_j - \alpha_j A p_j \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^j r_0\right\},$$

also

$$A\left(r_j - \alpha_j A p_j\right) \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^{j+1} r_0\right\}.$$

Insgesamt ergibt sich also

$$r_{j+1} \in \operatorname{span}\left\{r_0, Ar_0, \ldots, A^{j+1} r_0\right\}.$$

Damit ist p_{j+1} ein Element des Krylov-Unterraums $K_{j+1}(A, r_0)$, da nach Zeile VI und VII des Algorithmus

$$p_{j+1} = r_j + \beta_{j+1}\left(p_j - \omega_j A p_j\right)$$

gilt, denn nach Voraussetzung ist p_j aus dem Raum $K_j(A, r_0)$ und r_j aus $K_{j+1}(A, r_0)$. □

Damit folgt nun, wie zuvor bei CG, aus der Defintion des Residuums, dass x_m aus dem um x_0 verschobenen Krylov-Unterraum $K_m(A, r_0)$ stammt.

5.4 Vergleich der drei Verfahren

Bevor wir nun zu unseren Testbeispielen kommen, vergleichen wir noch die Vor- und Nachteile unserer drei Verfahren. Die wichtigen Vorteile von CG und GMRES gegenüber BiCGStab sind

- in BiCGStab kann es zu einem Abbruch des Verfahrens kommen, ohne dass eine gute Approximation an die Lösung berechnet wurde,
- BiCGStab hat einen hohen Rechenaufwand, da pro Iteration zweimal ein Matrix-Vektor-Produkt gebildet werden muss.

Der große Nachteil von CG gegenüber den beiden anderen Verfahren ist, dass die Matrix A symmetrisch und positiv definit sein muss. Der große Nachteil von GMRES ist der hohe Speicherbedarf, dadurch das eine Matrix und eine Basis eines Krylov-Unterraums gespeichert werden müssen und im flexiblen Fall sogar zwei Basen.

6 Numerische Beispiele

Als Testbeipiele dienen uns in 2D und 3D jeweils ein symmetrisches und ein nicht symmetrisches Problem. Bei den symmetrischen Problemen handelt es sich um

Laplace-Probleme. Die nicht symmetrischen Probleme sind Konvektions-Diffusionsprobleme (CD-Gleichungen). Für alle vier Probleme betrachten wir Dirichlet-Randbedingungen, das heißt für gegebenes $u_B(x)$ gilt

$$u(x) = u_B(x) \text{ für alle } x \in \partial\Omega.$$

Dabei bezeichne $\partial\Omega$ den Rand des betrachteten Gebiets Ω. In unserem Fall ist Ω das Einheitsquadrat bzw. der Einheitswürfel, das heißt

$$\Omega_2 = [0,1]^2 \text{ bzw. } \Omega_3 = [0,1]^3.$$

Die Galerkin Finite-Elemente-Methode erzeugt uns aus der Laplace-Gleichung ein lineares Gleichungssystem, wobei die Matrix A symmterisch und positiv definit ist. Für die CD-Gleichung benutzen wir die Streamline Upwind Petrov-Galerkin Finite-Elemente-Methode, kurz SUPG-FEM, welche uns ein lineares Gleichungssystem mit nicht symmetrischer Matrix liefert. Für eine Erläuterung von SUPG-FEM, auch Streamline Diffusion Finite-Elemente-Methode genannt, siehe Abschnitt III.3.2.1 in [11].

Als Finite-Element-Raum wählen wir den Raum Q_1. Dies ist in 2D beziehungsweise 3D der Raum der stetigen, stückweise bi- beziehungsweise trilinearen Funktionen.

6.1 Laplace-Gleichung in 2D

Es ist die Lösung der Differentialgleichung

$$-\Delta u(x,y) = 2\pi^2 \sin(\pi x)\sin(\pi y) \text{ für } (x,y) \in \Omega_2$$

mit

$$u(x,y) = 0 \text{ für alle } (x,y) \in \partial\Omega_2$$

gesucht. Die exakte Lösung u_e ist

$$u_e(x,y) = \sin(\pi x)\sin(\pi y),$$

denn

$$\frac{\partial^2 u_e(x,y)}{\partial x^2} = \frac{\partial^2 u_e(x,y)}{\partial y^2} = -\pi^2 \sin(\pi x)\sin(\pi y)$$

und damit gilt

$$-\Delta u_e(x,y) = -\frac{\partial^2 u_e(x,y)}{\partial x^2} - \frac{\partial^2 u_e(x,y)}{\partial y^2} = 2\pi^2 \sin(\pi x)\sin(\pi y).$$

In Abbildung 1 ist eine berechnete Lösung abgebildet.

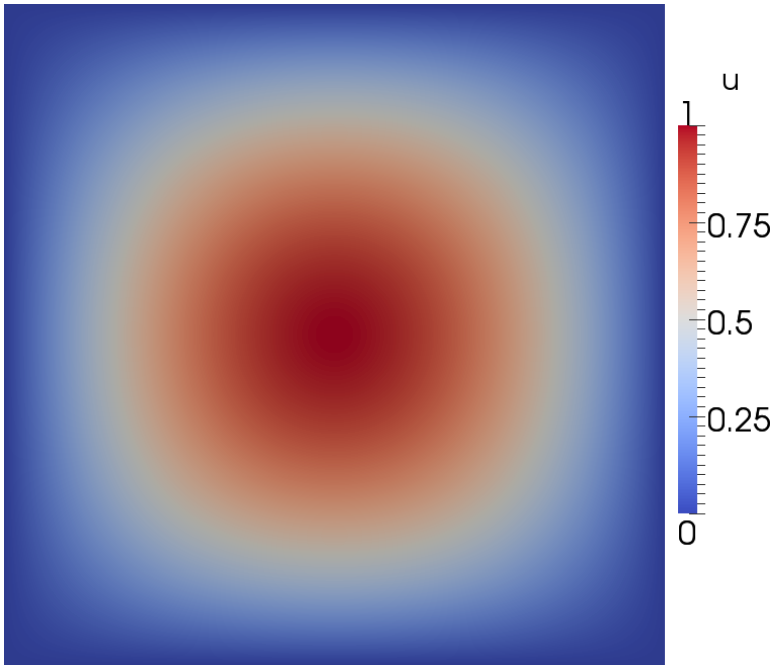

Abbildung 1: 2D Laplace-Problem: berechnete Lösung

6.2 Laplace-Gleichung in 3D

Das Problem ist ähnlich zum zweidimensional Fall. Gesucht ist die Lösung der Differentialgleichung

$$-\Delta u(x, y, z) = 3\pi^2 \sin(\pi x) \sin(\pi y) \sin(\pi z) \text{ für } (x, y, z) \in \Omega_3$$

mit der Randbedingung

$$u(x, y, z) = 0 \text{ für alle } (x, y, z) \in \partial\Omega_3.$$

Als exakte Lösung u_e ergibt sich

$$u_e(x, y, z) = \sin(\pi x) \sin(\pi y) \sin(\pi z),$$

da

$$\frac{\partial^2 u_e(x, y)}{\partial x^2} = \frac{\partial^2 u_e(x, y)}{\partial y^2} = \frac{\partial^2 u_e(x, y)}{\partial z^2} = -\pi^2 \sin(\pi x) \sin(\pi y) \sin(\pi z)$$

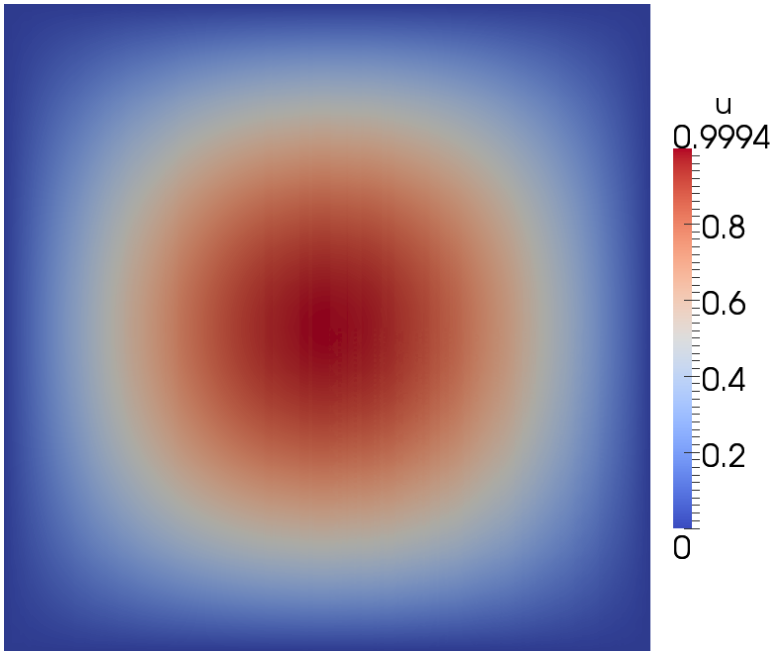

Abbildung 2: 3D Laplace-Problem: berechnete Lösung für die Schnittebene $x = 0.5$

und somit

$$\begin{aligned} -\Delta u_e(x,y,z) &= -\frac{\partial^2 u_e(x,y)}{\partial x^2} - \frac{\partial^2 u_e(x,y)}{\partial y^2} - \frac{\partial^2 u_e(x,y)}{\partial z^2} \\ &= 3\pi^2 \sin(\pi x)\sin(\pi y)\sin(\pi z). \end{aligned}$$

In Abbildung 2 ist eine berechnete Lösung für einen Schnitt durch das Gebiet Ω_3 dargestellt.

6.3 CD-Gleichung in 2D

Wir betrachten die Konvektions-Diffusions-Gleichung

$$-\epsilon \Delta u(x,y) + b \cdot \nabla u(x,y) = 0, (x,y) \in \Omega_2,$$

wobei $\epsilon = 10^{-8}$ die Diffusionskonstante ist und $b = \left(\cos\left(\frac{-\pi}{3}\right), \sin\left(\frac{-\pi}{3}\right)\right)$ ein konstantes Konvektionsfeld. Damit ist in diesem Fall der Konvektionsanteil dominant. Als Randbedingung wählen wir für $(x,y) \in \partial\Omega_2$

$$u(x,y) = \begin{cases} 0, \text{ falls } x = 1 \text{ oder } y < 0.7 \\ 1, \text{ sonst} \end{cases}.$$

Damit ergibt sich für die Lösung (siehe Abbildung 3 und 4) eine innere Grenzschicht, die im Punkt $(0, 0.7)$ beginnt, sich in Richtung $b \approx (0.5, -0.87)$ fortsetzt und in etwa im Punkt $(0.4, 0)$ endet.

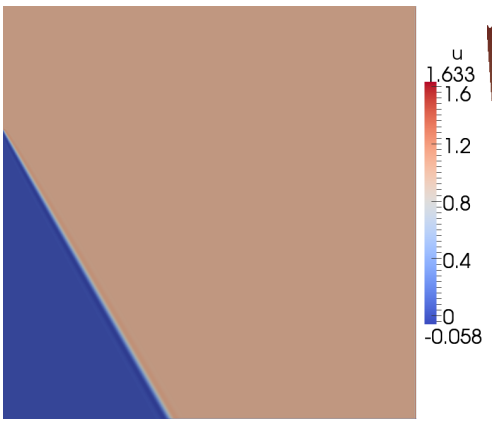

Abbildung 3: 2D CD-Problem: Grenzschicht

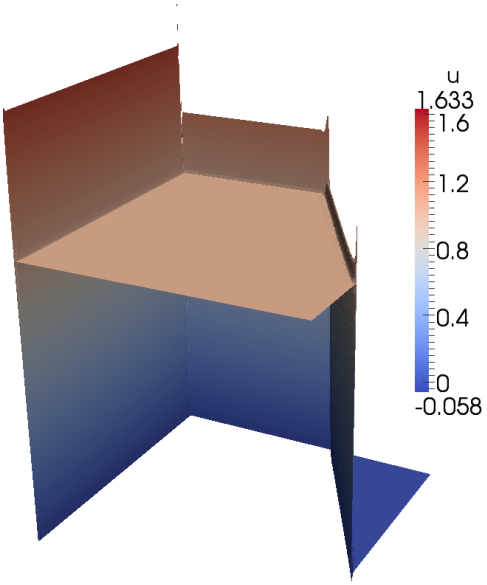

Abbildung 4: 2D CD-Problem: Randbedingungen sind erfüllt

6.4 CD-Gleichung in 3D

Als letztes Beispiel betrachten wir

$$-\epsilon\Delta u(x,y,z) + b \cdot \nabla u(x,y,z) + u(x,y,z) = f(x,y,z), (x,y,z) \in \Omega_3$$

mit $\epsilon = 10^{-6}, b = \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix}$ und f ist so gewählt, dass

$$u_e(x,y,z) = \left(x - \exp\left(\frac{2\,(x-1)}{\epsilon}\right)\right)\left(y^2 - \exp\left(\frac{3\,(y-1)}{\epsilon}\right)\right)\left(z^3 - \exp\left(\frac{4\,(z-1)}{\epsilon}\right)\right)$$

Lösung ist. Als Randbedingung muss damit

$$u(x,y,z) = u_e(x,y,z) \text{ für alle } (x,y,z) \in \partial\Omega_3$$

gewählt werden. In den Abbildungen 5 und 6 sind zwei Schnitte durch Ω_3 für eine berechnete Lösung dargestellt.

7 Auswertung

Zur Untersuchung der Algorithmen wurde jeweils die feste und flexibel vorkonditionierte Variante der 3 Verfahren in MooNMD (Mathematics and object oriented Numerics in Magdeburg) implementiert. Die flexible Variante von GMRES wurde

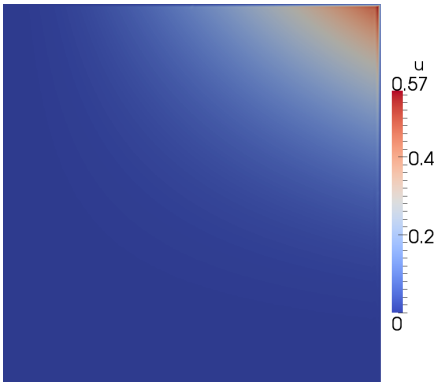

Abbildung 5: 3D CD-Problem: berechnete Lösung für die Schnittebene $x = 0.5$

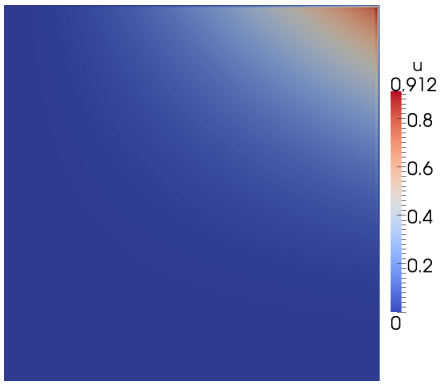

Abbildung 6: 3D CD-Problem: berechnete Lösung für die Schnittebene $x = 0.8$

bereits von Prof. John implementiert und diente mir als Vorlage für die anderen Verfahren. Da für BiCGStab der Algorithmus für flexible Vorkonditionierung mit dem für die feste Vorkonditionierung übereinstimmt, benötigt man für BiCGStab nur eine Variante. Als feste Vorkonditionierer wurde das Jacobi und das SSOR Verfahren mit $\omega = 1$, auch symmetrisches Gauß-Seidel Verfahren genannt, benutzt. Als flexibler Vorkonditionierer diente das Mehrgitter-Verfahren, welches ausführlich in Kapitel 13 im Buch von Y. Saad [1] erklärt wird.

Wie erwartet stimmen für Jacobi und SSOR als Vorkonditionierer die flexible Variante mit der festen Variante von CG beziehungsweise GMRES überein. Für das Mehrgitter-Verfahren als Vorkonditionierer ist zu erwarten, dass die flexible Variante ein besseres Ergebnis als die feste Variante liefert. Jedoch stimmen auch dort beide Varianten von CG und GMRES überein. Der Grund hierfür ist, dass das Mehrgitter-Verfahren sehr exakt gelöst wird. Dadurch verhält sich das Mehrgitter-Verfahren fast wie ein fester Vorkonditionierer. Dies erkennt man beispielsweise beim CG Verfahren. Die flexible Implementation von CG unterscheidet sich von der festen Implementation von CG nur in der Berechnung von β. Bei dem Vergleich der Werte für β stellten wir fest, dass sie sich erst in der fünften, meist sogar erst in der zehnten, signifikanten Stelle unterscheiden.

Da sich die Ergebnisse der festen Variante der Algorithmen nicht von denen der flexiblen Variante unterscheiden, sind in den Abbildungen 7 bis 13 nur die Ergebnisse der flexiblen Variante dargestellt. Die Level geben dabei den Grad der Verfeinerung des Gitters an, wobei ein höheres Level ein feineres Gitter bedeutet. GMRES wurde bei uns alle 20 Iterationen neu gestartet.

In den Abbildungen 7 bis 9 vergleichen wir die Vorkonditionierer für unsere 3 Verfahren für das dreidimensionale Laplace-Problem. Die benötigten Iterationen hängen stark vom Vorkonditionierer ab. Jacobi ist der einfachste Vorkonditionierer und benötigt deswegen auch deutlich mehr Iterationen als die anderen beiden. Mit dem Mehrgitter-Verfahren benötigen wir nie mehr als 10 Iterationen, selbst für das

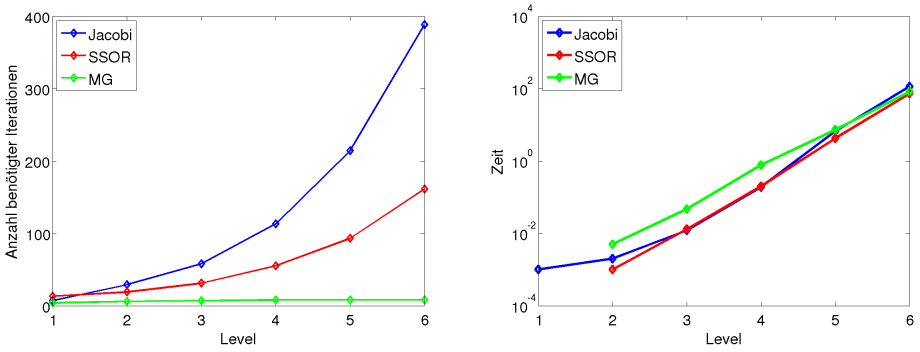

Abbildung 7: 3D Laplace: CG: Iteartionen und Zeit

letzte Level welches über zwei Millionen Freiheitsgrade besitzt. Jedoch benötigt es deutlich mehr Zeit als die anderen beiden Verfahren. Erst in den letzten beiden Leveln ist ein Unterschied in der benötigten Zeit zu erkennen, obwohl der Unterschied in der Anzahl der benötigten Iterationen für die ersten Level bereits sehr groß ist. Insgesamt scheint sich der Aufwand des Mehrgitter-Verfahrens jedoch zu lohnen, da für BiCGStab und GMRES deutlich weniger Zeit benötigt wird als mit den anderen beiden Vorkonditionierern. Für CG ist SSOR etwas schneller als das Mehrgitter-Verfahren.

Zum Vergleich der 3 Verfahren untereinander und mit einem direkten Löser untersuchen wir die Abbildungen 10 bis 13. Als direkten Löser verwendeten wir UMFPACK [12]. Für die iterativen Verfahren wurde das Mehrgitter-Verfahren als Vorkonditionierer gewählt. Als erstes sei bemerkt, dass BiCGStab genauso viel Speicher benötigt wie CG.

Der direkte Löser benötigt deutlich mehr Zeit für die letzten Level als die iterativen Verfahren. Der Zeitaufwand der iterativen Verfahren untereinander unterscheidet sich kaum. Es gibt kleine Unterschiede, jedoch scheint kein Verfahren deutlich besser als die anderen zu sein.

Beim Vergleich des Speicherbedarfs erkennt man in den letzten abgebildeten Leveln, dass GMRES am meisten Speicher benötigt und BiCGStab und CG am wenigsten. In den ersten Leveln sind die Unterschiede kaum erkennbar, jedoch mit steigenden Freiheitsgraden werden die Unterschiede sichtbarer.

Abschließend sei noch bemerkt, dass BiCGStab mit SSOR als Vorkonditioneirer bei dem zweidimensionalem CD-Problem auf dem letzten Level stagniert. Es kommt zu dem in Abschnitt 5.3.4 angesprochenen Problem, dass die Parameter $\langle \hat{r}_0, v_i \rangle$ und ρ_i sehr klein werden, ohne dass die Norm des Residuums abnimmt. In 900 Iterationen verbessert sich die Norm des Residuums nicht und erreicht nie einen Wert unter 10^{-3}, jedoch nehmen die Parameter $\langle \hat{r}_0, v_i \rangle$ und ρ_i Werte unter 10^{-20} an.

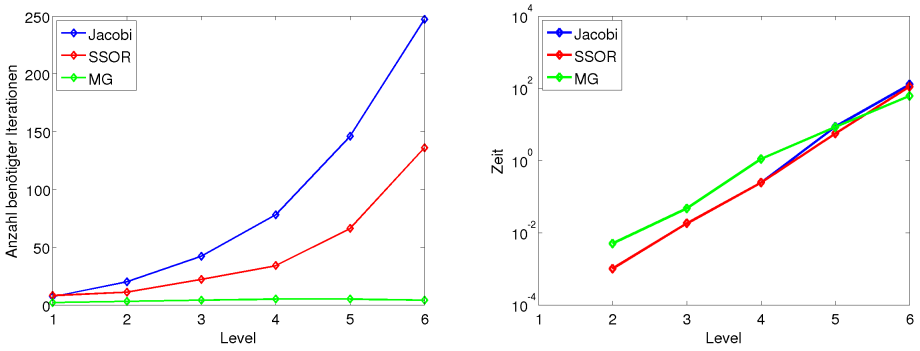

Abbildung 8: 3D Laplace: BiCGStab: Iteartionen und Zeit

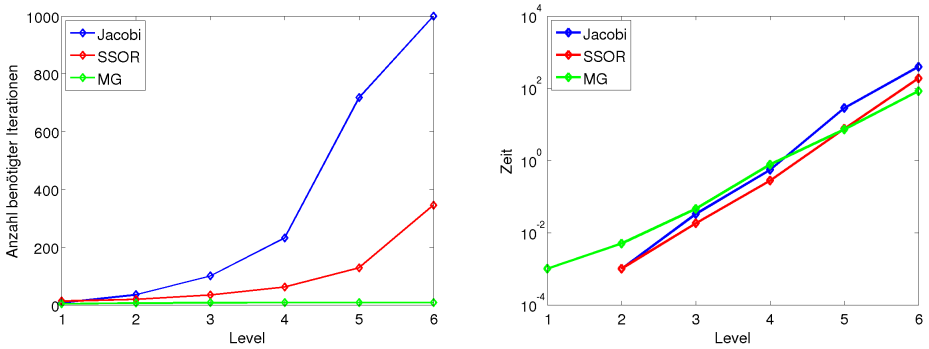

Abbildung 9: 3D Laplace: GMRES: Iteartionen und Zeit

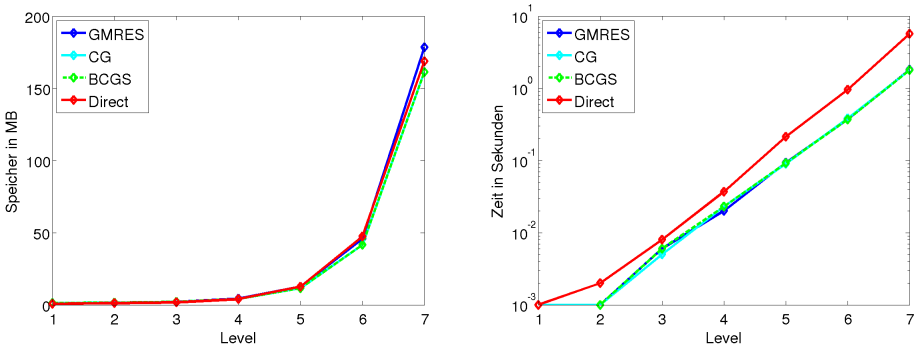

Abbildung 10: 2D Laplace: Speicher und Zeit

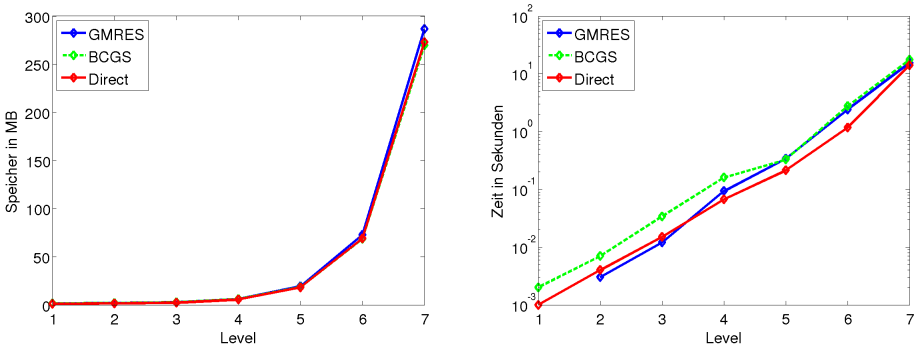

Abbildung 11: 2D CD: Speicher und Zeit

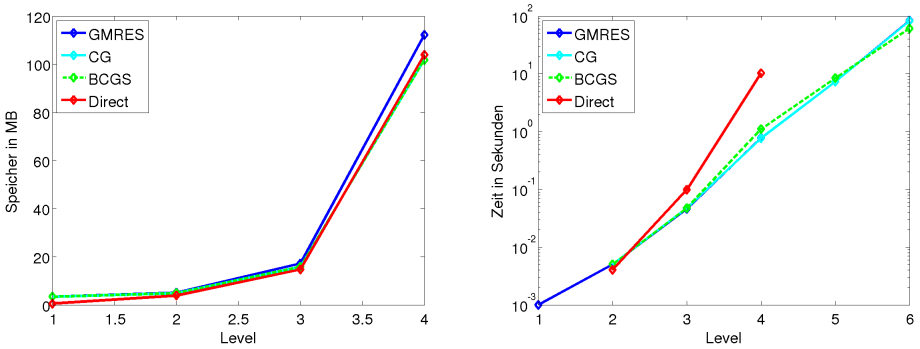

Abbildung 12: 3D Laplace: Speicher und Zeit

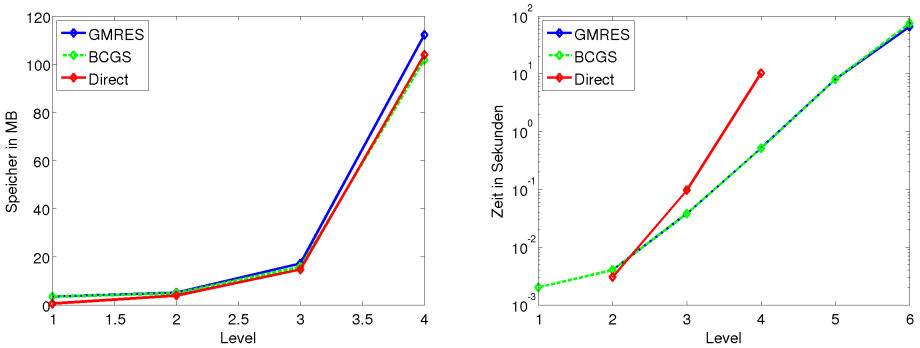

Abbildung 13: 3D CD: Speicher und Zeit

8 Fazit

Für große Systeme sind die iterativen Löser die bessere Wahl, da sie deutlich weniger Zeit zum Lösen benötigen. Des Weiteren ist die Verwendung des Mehrgitter-Verfahrens als Vorkonditionierer bei der Lösung von Differentialgleichungen eine gute Möglichkeit, die Stagnation von iterativen Verfahren zu verhindern. Wenn genug Speicher vorhanden ist, bietet sich GMRES an, da man damit beliebige Gleichungssysteme lösen kann. Falls die Matrix A symmetrisch und positiv definit ist, hat man mit CG eine gute Alternative zu GMRES, die weniger Speicher benötigt. Ist die Matrix A nicht symmetrisch und positiv definit so hat man mit BiCGStab ein Verfahren, welches nicht so viel Speicher wie GMRES benötigt. Dafür ist die Gefahr der Stagnation des Verfahrens größer als bei GMRES.

Die untersuchten Beispiele in dieser Arbeit geben keinen Hinweis darauf, dass die flexiblen Varianten von CG und GMRES bessere Ergebnisse für flexible Vorkonditionierung liefern als die Versionen von CG und GMRES für feste Vorkonditionierer. Dazu müssen noch komplexere Beispiele untersucht werden, für die das Mehrgitter-Verfahren nicht so schnell konvergiert.

Literaturverzeichnis

[1] Y. Saad, Iterative Methods for Sparse Linear Systems, second edition, SIAM, 2003

[2] W. E. Arnoldi, The principle of minimized iteration in the solution of the matrix eigenvalue problem, Q. Appl. Math., Vol. 9 (1951) No. 17, S. 17-29

[3] Y. Saad, M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Statist. Comput., Vol. 7 (1986) No. 3, S. 856-869

[4] A. Greenbaum, V. Pták, and Z. Strakoš, Any nonincreasing convergence curve is possible for GMRES, SIAM J. Matrix Anal. Appl., Vol 17 (1996) No. 3, S.465-469

[5] M. R. Hestenes, E. Stiefel, Methods of Conjugate Gradients for Solving Linear Systems, Journal of Research of the National Bureau of Standards, Vol. 49 (1952) No. 6, S. 409-436

[6] H. A. van der Vorst, Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems, SIAM J. Sci. Statist. Comput., Vol. 13 (1992) No. 2, S. 631-644

[7] R. Fletcher, Conjugate gradient methods for indefinite systems, Lecture Notes in Math., Vol. 506 (1976), S. 73-89

[8] P. Sonneveld, CGS: a fast Lanczos-type solver for nonsymmetric linear systems, SIAM J. Sci. Statist. Comput., Vol. 10 (1989) No. 1, S. 36-52.

[9] G. H. Golub, Q. Ye, Inexact Preconditioned Conjugate Gradient Method with Inner-Outer Iteration, SIAM J. Sci. Comput., Vol. 21 (1999) No. 4, S. 1305–1320

[10] J. A. Vogel, Flexible BiCG and flexible Bi-CGSTAB for nonsymmetric linear systems, Appl. Math. Comput., Vol. 188 (2007) No. 1, S. 226-233

[11] H. G. Roos, M. Stynes, L. Tobiska, Robust Numerical Methods for Singularly Perturbed Differential Equations, second Edition, Springer, 2008

[12] T. A. Davis, Algorithm 832: UMFPACK V4.3—an unsymmetric-pattern multifrontal method, ACM Trans. Math. Software, Vol. 30 (2004) No. 2, S. 196-199