OPTIMIERUNG DER NUMERIK EINES LINEAREN GLEICHUNGSSYSTEMS FÜR DIE SIMULATION DES SCHALLFELDES IM VOKALTRAKT

Johann August Marwitz, Simon Stone, Peter Birkholz

Institut für Akustik und Sprachkommunikation, Technische Universität Dresden
Johann_August.Marwitz@mailbox.tu-dresden.de

Kurzfassung: Im Gegensatz zur in Forschung und Industrie weit verbreiteten konkatenativen Sprachsynthese besitzt die artikulatorische Sprachsynthese alle Freiheiten, die auch ein natürlicher Sprechapparat besitzt. Durch die vollständige aeroakustische Simulation der Sprache entsteht nachteilig ein hoher Rechenaufwand. Maßgeblich ist die Berechnung eines großen linearen Gleichungssystems (LGS) zur Simulation des Schallfeldes im Vokaltrakt. Um die Berechnungszeit zu reduzieren wurden für den artikulatorischen Sprachsynthesizer VocalTractLab verschiedene numerische Verfahren, Vorkonditionierungen und Speicherstrategien untersucht. Insgesamt ist es gelungen, mittels der Cholesky-Zerlegung und einer auf den Algorithmus sowie auf das LGS angepassten Speichermethode die Berechnungszeit um das Fünf- bis Sechsfache gegenüber dem bisher verwendeten Gauß-Seidel-Relaxationsverfahren zu verkürzen. Verglichen mit der Dauer des simulierten akustischen Signals nimmt die Berechnung des LGS nur noch ca. die Hälfte der simulierten Zeit in Anspruch - womit die gesamte Synthese deutlich näher an die Echtzeitfähigkeit rückt und auf aktueller PC-Hardware einen Echtzeitfaktor von 2 erreicht (vgl. vorher 4,5).

1 Motivation und Besonderheiten der Artikulatorischen Synthese

Bei der künstlichen Erzeugung von Sprache wird grundsätzlich zwischen zwei Ansätzen unterschieden: Der Signalmodellierung anhand einer Datenbank von Sprachaufnahmen und der vollständigen artikulatorischen Modellierung, bei der die Spracherzeugung durch den Rechner simuliert wird. In Alltag und Praxis herrscht gegenwärtig noch der erste Ansatz vor, zumeist in Form der konkatenativen "Unit-Selection" (Google Assistant, Amazon Alexa, Apple Siri, Mary TTS [1]). Diese nutzt eine Datenbasis von einem menschlichen Sprecher eingesprochenen lautsprachlicher Einheiten, wie Silben, Wörtern oder sogar Sätzen, aus denen die jeweils passendste Einheit ausgewählt und mit weiteren Einheiten zu Äußerungen verkettet wird. Unit Selection Synthese erreicht grundsätzlich eine gute Verständlichkeit und Natürlichkeit, birgt allerdings konzeptionelle Nachteile. Bedingt durch die feste Lautbasis ist sie in den Freiheiten stark beschränkt. Um beispielsweise prosodische und phonetische Veränderungen zu erzeugen, müssen Signalmanipulationen durchgeführt werden, welche zu Artefakten führen. Aufgrund des festen Lautinventars ist es i.d.R. nicht möglich, mehrere unterschiedliche Stimmen und Sprachen darzustellen. Deutlich flexibler ist hier die artikulatorische Sprachsynthese, welche auf einer vollständigen Simulation der Spracherzeugung beruht und damit Manipulationen auf der Ebene des "Signalgenerators" erlaubt, die artefaktfreie, realistische Ergebnisse erzielen.
Die Sprachproduktion des Menschen beruht auf einem zeitlich und räumlich präzise koordinierten Zusammenspiel aus Energiequelle (Lunge und Stimmlippen) und einer Vielzahl von Artikulatoren wie der Zunge, dem Kiefer, dem Gaumensegel und den Lippen, die gemeinsam den Vokaltrakt formen. Für die Simulation von Sprache muss ein parametrisierbares Modell

des Vokaltrakts erstellt werden. Der Detaillierungsgrad des Modells entscheidet schlussendlich über die Natürlichkeit der simulierten Sprache. Im Allgemeinen wird zwischen drei Gruppen unterschieden, der 3D-, 2D- und 1D-Modellierung. Am wenigsten Vereinfachungen werden bei der 3D-Modellierung vorgenommen, welche auf drei verschiedene Arten realisiert wird. Aus Bilddaten (i.d.R. MRT) können explizite 3D-Netze für jede Form nachgebildet werden [2, 3, 4]. Eine weitere Möglichkeit ist die statistische Modellierung, welche die Vokaltaktform aus den Hauptkomponenten der möglichen Vokaltraktformen nachbildet [5, 6]. Letztlich können auch die biomechanischen Aktoren simuliert werden, durch welche die Vokaltraktform bestimmt wird [7, 8]. Unter Berücksichtigung der Symmetrie des Vokaltrakts, kann das 3D-Modell auf ein 2D-Modell reduziert werden. Die lateralen Dimension wird vernachlässigt und der Vokaltrakt in der mediosagittalen Ebene dargestellt. Die prinzipiellen Modellierungsmethoden sind die gleichen — direkte geometrische Modellierung [9, 10] und statistische Überlagerung verschieden gewichteter Vokaltaktformen [11, 12]. Unter der Annahme von ebenen Schallwellen kann das Vokaltraktmodell weiter vereinfacht werden. In diesem Fall ist nur der Verlauf der Querschnittsfläche orthogonal zur Schallausbreitungsrichtung relevant. Der Vokaltrakt wird folglich als eindimensionale Funktion der Fläche über der Position entlang der Schallausbreitungsrichtung gesehen. Nachteilig können höhere Moden im 1D-Modell nicht mehr dargestellt werden [13], aus perzeptiver Sicht sind diese jedoch vernachlässigbar [14].
Das 1D-Modell ist, aufgrund seiner Einfachheit in der Berechnung sehr Ressourcen schonend und eignet sich besonders gut für die rechnergestützte Simulation von Sprache. Die Modellierung der Vokaltaktformen zur Gewinnung der 1D Querschnittsfunktion wird allerdings meist mit Hilfe des 3D- oder 2D-Modells vorgenommen. Auch das weltweit eingesetzte artikulatorische Synthesesystem *VocalTractLab (VTL)*[1] berechnet die Querschnittsfunktion eines 3D Vokaltraktmodells in 129 Schnittebenen. Die so gewonnene diskrete Querschnittsfunktion wird dann in ein Rohrsystem überführt, das aus 97 zylindrischen Abschnitten unterschiedlichen Querschnitts besteht. Zur Berechnung der Druckverhältnisse und Volumenströme in diesem Rohrsystem (und damit zur Berechnung des Schallfeldes im Vokaltrakt) kann in elektroakustischer Analogie aus dem diskreten 1D-Modell ein elektrisches Netzwerk hergeleitet werden, welches das aero-akustische Verhalten abbildet. Für dieses Netzwerk wird ein lineares Gleichungssystem (LGS) erstellt, das nur von den Volumenströmen an den Knotenpunkten des Netzwerks, also am Übergang von einem Rohrabschnitts zum nächsten, abhängt. Zur Berechnung des Sprachsignals muss dieses Gleichungssystem in jedem Zeitschritt der Simulation (übliche Abtastfrequenz 44,1 kHz) nach den Volumenströmen aufgelöst werden. Dementsprechend entsteht bei der Simulation des Schallfeldes im Vokaltrakt einen enormer Rechenaufwand, der signifikant von einer effizienten Lösung des Gleichungssystems abhängt.

2 Eigenschaften des linearen Gleichungssystems

Um ein effizientes Lösungsverfahren zu finden, muss man sich zunächst mit der Struktur des gegebenen linearen Gleichungssystems auseinandersetzen. Im Allgemeinen wird ein lineares Gleichungssystem in der Form

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{1}$$

dargestellt. Die Koeffizientenmatrix $\boldsymbol{A}$ und der Vektor der rechte Seite $\boldsymbol{b}$ sind im VTL bekannt, da sie in jedem Zeitschritt der Simulation aus den Netzwerkkomponenten berechnet werden. Der Lösungsvektor $\boldsymbol{x}$ ist hingegen unbekannt und enthält in VTL die Volumenströme durch die einzelnen Rohrsegmente [15].
Das im vorhergehenden Abschnitt beschriebenen LGS besteht aus $n = 97$ Gleichungen, die

[1] `www.vocaltractlab.de`

Koeffizientenmatrix $\boldsymbol{A}$ hat folglich eine Größe von 97 Zeilen $\times$ 97 Spalten. Eine Besonderheit ist, dass die Koeffizientenmatrix mit nur 306 Elementen ungleich null (von 9409) schwachbesetzt ist. Dabei sind die Koeffizienten in Tridiagonalgestalt angeordnet, das heißt, dass nur die Hauptdiagonale der Matrix und ihre beiden Nebendiagonalen mit von null verschiedenen Koeffizienten besetzt sind. Diese Struktur bleibt in jedem Zeitschritt erhalten und lediglich die Werte der Koeffizienten ändern sich. Bedingt durch Verzweigungen im elektro-akustischen Netzwerk wird in mehren Zeilen von dieser Struktur abgewichen. Weiterhin ist die Koeffizientenmatrix symmetrisch, da $\boldsymbol{A} = \boldsymbol{A}^T$ gilt und negativ definit, da $\boldsymbol{v}^T\boldsymbol{A}\boldsymbol{v} < 0$ erfüllt ist, wobei $\boldsymbol{v}$ ein beliebiger vom Nullvektor verschiedener Vektor ist. Einige Lösungsverfahren benötigen allerdings eine positiv definite Koeffizientenmatrix. Diese erhält man durch negieren sämtlicher Einträge von $\boldsymbol{A}$ sowie von $\boldsymbol{b}$ — der Lösungsvektor bzw. das Ergebnis bleibt unverändert.

3 Übersicht verschiedener Klassen von Lösungsverfahren

Ziel ist nun die Lösung des LGS, also $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$. Im Allgemeinen werden die Verfahren in zwei große Klassen aufgeteilt: die direkten und die iterativen Verfahren.
Direkte Verfahren ermitteln nach einer endlichen bestimmbaren Anzahl von Rechenoperation die exakte Lösung des LGS. Der theoretisch exakten Lösungsfindung und dem determinierten Verhalten steht nachteilig ein hoher Berechnungsaufwand gegenüber, der kubisch mit der Anzahl der Gleichungen bzw. Unbekannten des LGS steigt. Die untersuchte Koeffizientenmatrix hat die Größe von n = 97 und stellt einen Grenzfall dar, bei dem man die direkte Berechnung des LGS noch in Betracht ziehen sollte. In dieser Arbeit wurden zwei direkte Verfahren untersucht: die LR-Zerlegung und die Cholesky-Zerlegung. Im Vergleich ist die Cholesky-Zerlegung das konzeptionell effizientere Verfahren, da sie durch Ausnutzen der Symmetrie von $\boldsymbol{A}$ nur etwa halb so viele Operationen benötigt.
Den direkten Verfahren stehen die iterativen Verfahren gegenüber. Die Verfahren dieser Klasse haben alle gemein, dass ausgehend von einer ersten Lösungsschätzung $\boldsymbol{x}^{(0)}$ in jeder Iteration des Algorithmus eine Näherung an die Lösung aus der letzten Näherungslösung des LGS bestimmt wird.

$$\boldsymbol{x}^{(k+1)} = \Phi(\boldsymbol{x}^{(k)}) \text{ mit } k = 0, 1, ... \tag{2}$$

Die aktuelle Iterationszahl wird hier durch k dargestellt. Wenn alle Anforderungen des Verfahrens an die Koeffizientenmatrix des LGS erfüllt werden konvergiert die Näherungslösung mit den Iterationen gegen die exakte Lösung.
Die iterativen Verfahren werden weiter in Unterklassen aufgeteilt. In dieser Untersuchung wurden die Klassen Splitting-Verfahren und Krylov-Unterraum-Verfahren betrachtet.
Die Splitting-Verfahren werden heutzutage nur noch selten zur Lösung von LGS eingesetzt [16, S. 133]. Sie spielen jedoch eine große Rolle bei der Vorkonditionierung von Projektionsmethoden [17, S. 284]. Strukturell sind die Splitting-Verfahren einfach zu implementieren und benötigen wenig Rechenzeit für eine Iteration. Allerdings verringert sich der Fehler pro Iteration sehr wenig, insbesondere wenn eine hohe Genauigkeit gefordert ist, und somit werden viele Iterationen benötigt, bis das gewünschte Resultat vorliegt.
Die Krylov-Unterraum-Verfahren sind sogenannte semi-direkte Verfahren, das heißt nach spätestens n Iterationen (n entspricht der Dimension von $\boldsymbol{A}$) wird die exakte Lösung gefunden. In der Regel ist eine hinreichend genaue Lösung wesentlich schneller gefunden. Die Konvergenzgeschwindigkeit kann durch die Vorkonditionierung der Koeffizientenmatrix noch erheblich gesteigert werden.
In der folgenden Tabelle wird ein Überblick gegeben, welche Verfahren aus den zuvor beschriebenen Klassen untersucht wurden und wie diese einzuordnen sind.

Tabelle 1 – Klassifizierung unterschiedlicher Verfahren

Verfahren	Anforderungen an die Koeffizienten-matrix	Verfahrensklasse	Nutzbar zur Vorkondi-tionierung
Cholesky-Zerlegung	symmetrisch und positiv definit	direkt	ja
LR-Zerlegung	regulär	direkt	ja
Jacobi-Verfahren	regulär	iterativ, Splitting-Verfahren	ja
Gauß-Seidel-Relaxations (GSR)-Verfahren	regulär	iterativ, Splitting-Verfahren	ja
Symmeterisches Gauß-Seidel (SGS)-Verfahren	regulär	iterativ, Splitting-Verfahren	ja
Conjugate-Gradients (CG)-Verfahren	symmetrisch und positiv definit	iterativ, Krylov-Unterraum-Verfahren	nein
BiCG-Verfahren	regulär	iterativ, Krylov-Unterraum-Verfahren	nein
RFOM-Verfahren	regulär	iterativ, Krylov-Unterraum-Verfahren	nein
GMRES-Verfahren	regulär	iterativ, Krylov-Unterraum-Verfahren	nein
MINRES-Verfahren	symmetrisch	iterativ, Krylov-Unterraum-Verfahren	nein

4 Vorkonditionierung und Speicherstrategie

Die Vorkonditionierung ist eine Technik die Kondition $cond()$ der Koeffizientenmatrix zu verkleinern und dadurch die Krylov-Unterraum-Verfahren wesentlich zu beschleunigen. Die Kondition bestimmt sich aus dem Quotient des größten (σ_n) und kleinsten (σ_1) Eigenwerts der Koeffizientenmatrix zu

$$cond(\boldsymbol{A}) = \frac{\sigma_n}{\sigma_1}. \tag{3}$$

[18, S. 25-29]. Es kann gezeigt werden, dass durch Annähern der Koeffizientenmatrix $\boldsymbol{A}$ an die Identität die Eigenwerte immer näher beieinander liegen und somit die Konditionszahl kleiner wird. Daher versucht man eine Matrix zu finden, die sich leicht berechnen lässt und eine Näherung an die Inverse der Koeffizientenmatrix des LGS ist. Aus verschiedenen Verfahren zur Lösung von LGS lassen sich solche Matrizen ableiten.
Weiterhin wurden die Algorithmen hinsichtlich effizienter Speicherstrukturen der Koeffizientenmatrix untersucht. Insbesondere bei der Cholesky-Zerlegung hat es sich als effektiv erwiesen, nur die Einhüllende der Koeffizientenmatrix zu speichern, da diese nur die Elemente enthält die zur Berechnung der Cholesky-Zerlegung benötigt werden. Bei iterativen Verfahren werden hingegen nur die von null verschiedenen Elemente benötigt. Die Sternchen sollen in folgender

Matrix von null verschiedene Elemente sein. Die Freiräume sind mit Nullelementen besetzt.

$$\boldsymbol{A} = \begin{bmatrix} * & * & & & * & & \\ * & * & * & & 0 & * & \\ & * & * & * & 0 & 0 & \\ & & * & * & * & 0 & \\ * & 0 & 0 & * & * & * & \\ & * & 0 & 0 & * & * & * \\ & & & & & * & * \\ & & & & & & * \end{bmatrix} \tag{4}$$

Zur Einhüllenden einer Matrix gehören neben den von null verschiedenen Elementen auch die Einträge, welche sich in der linken unteren Dreiecksmatrix in einer Zeile sowie in der rechten oberen Dreiecksmatrix in einer Spalte zwischen zwei von null verschiedenen Elementen befinden. In der dargestellten Matrix wären das genau die Elemente, welche explizit durch eine Null dargestellt sind. Speichert man nur die Einhüllende der Koeffizientenmatrix, lassen sich folglich sehr viele Rechenoperationen sparen wenn sich die Elemente nah an der Hauptdiagonalen von $\boldsymbol{A}$ befinden.

5 Ergebnisse und Diskussion

Zur Untersuchung, welche Kombination aus Vorkonditionierung, Speicherstruktur und Lösungsverfahren am effizientesten ist, wurden zwei Strategien verfolgt. Ein gemitteltes relatives Fehlermaß (das Residuum) wurde über der Iteration dargestellt, um ein anschauliches Diagramm zu erhalten, in dem erkennbar ist, wie schnell ein Verfahren gegen eine hinreichend genaue Lösung konvergiert. Da die benötigte Zeit zur Durchführung einer Iteration in den unterschiedlichen Verfahrensweisen zum einen teilweise nicht konstant ist und zum anderen Vergleichbarkeit mit den direkten Verfahren gewährleistet werden sollte, wurde ein Vergleich der Berechnungszeit angestellt. Als Grundlage für den Vergleich dienten 26457 LGS, welche für die Synthese von 600 ms des Vokals „a“ im VTL verwendet werden. Der zuvor im VTL implementierte Algorithmus, das GSR-Verfahren, diente als Referenz. Um zu beurteilen, welcher der zuvor erwähnten Algorithmen für die Simulation des Schallfeldes im Vokaltrakt am besten geeignet ist, wurden zunächst alle Algorithmen zum Vergleichen im MATLAB implementiert. Auf dieser Grundlage war es möglich, verschiedene Vorkonditionierungen und Speicherstrukturen in Kombination mit den Algorithmen zu bewerten. Für einen abschließenden Vergleich wurden die erfolgversprechendsten Verfahren in C++ in das VTL implementiert und hinsichtlich ihrer Laufzeit beurteilt.

5.1 Residuenverläufe

Im Folgenden sind verschiedene gemittelte Residuenverläufe über der Iteration dargestellt. Der Mittelwert der Residuen wird aus den in Iteration k berechneten Residuen der 26457 LGS gebildet. Die schwarze, gestrichelte, horizontale Linie stellt dabei den Wert des Residuums dar, bei dem die Berechnung des LGS bzw. der Volumenströme hinreichend genau ist. Abbildung 1 zeigt einen ersten Vergleich verschiedener Verfahren. Effiziente Verfahren konvergieren mit möglichst wenig Iterationen gegen eine hinreichend genau Lösung und haben somit einen steilen Residuenverlauf. Der Referenzalgorithmus weist im Bereich einer sehr ungenauen Approximation der Lösung sogar die höchste Konvergenzgeschwindigkeit auf, wird jedoch im weiteren Verlauf von anderen Algorithmen überholt. Das Restarted Full Orthogonalization (RFOM)-Verfahren scheint im Vergleich am besten abzuschneiden, da bei diesen Verfahren die benötigte

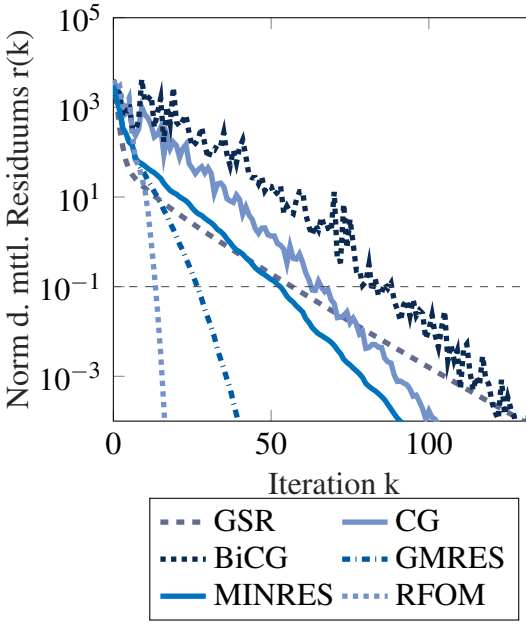

Abbildung 1 – Verfahren im Vergleich

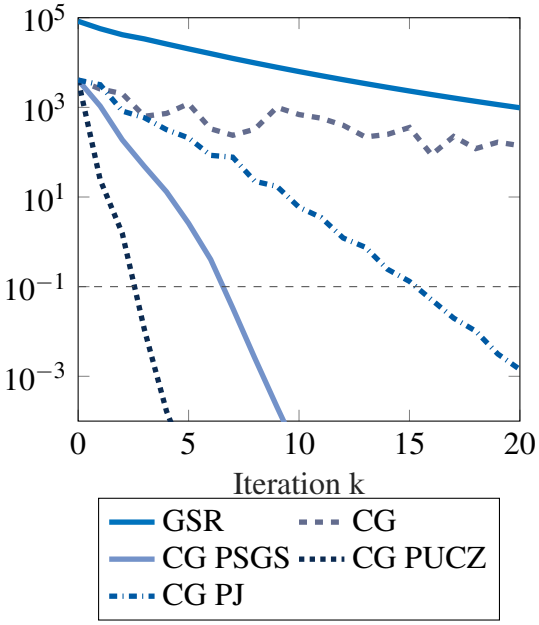

Abbildung 2 – Vorkonditionierte CG-Verfahren im Vergleich

Zeit für eine Iteration sehr variabel ist, ist diese Betrachtungsweise jedoch nur bedingt aussagekräftig. Die geringste Konvergenzgeschwindigkeit und einen stark oszillierenden Verlauf hat das Bi-Conjugate Gradients (BiCG)-Verfahren und wurde deshalb nicht weiter betrachtet.

In der Abbildung 2, sind die Residuenverläufe verschiedener Vorkonditionierungen beim CG-Verfahren zu sehen. Die erste Lösungsschätzung wurde durch lineare Extrapolation aus den Lösungen der zwei vorherigen Simulationszeitschritten berechnet. Zum Vergleich dient der Referenzalgorithmus, das GSR-Verfahren, wie er ursprünglich im VTL implementiert war und somit ohne Lösungsapproximation. Es wird ersichtlich, dass durch die Vorkonditionierung viel weniger Iterationen benötigt werden. Schon durch einfache Jakobi-Vorkonditionierung verringert sich die durchschnittliche Iterationsanzahl von ca. 65 (siehe Abb. 1) auf 15, um eine hinreichend genaue Lösung zu erreichen. Bei der Vorkonditionierung durch das SGS-Verfahren werden im Mittel nur noch ca. sieben und mit der unvollständigen Cholesky-Zerlegung (UCZ) sogar nur ca. drei Iterationen benötigt. Das Verhältnis von eingesparten Iterationen zum Berechnungsaufwand der Vorkonditionierung wird im anschließenden Laufzeitvergleich ersichtlich.
Beim GMRES- und MINRES-Verfahren konnten ähnlich gute Ergebnisse erzielt werden. Bezogen auf den Grundalgorithmus sind die Vorteile der Vorkonditionierung jedoch wesentlich geringer.

5.2 Laufzeitvergleich

Wie eingehend erwähnt wurden alle Algorithmen zusätzlich hinsichtlich der benötigten Rechenzeit untersucht, da die Zeit zur Durchführung einer Iteration in den unterschiedlichen Verfahrensweisen zum einen teilweise nicht konstant ist und zum anderen Vergleichbarkeit mit den Direkten-Verfahren (bei denen die exakte Lösung des LGS bestimmt wird) gewährleistet werden sollte. Zunächst wurde die in MATLAB implementierten Algorithmen für einen Vergleich heran gezogen. Die gemessenen Zeiten sind in Tabelle 2 dargestellt. Es wird ersichtlich, dass die Verfahren GMRES und RFOM trotz kleiner Iterationsanzahl sehr viel Zeit benötigen um die 26457 LGS zu lösen. Aufgrund dessen wurde das RFOM-Verfahren nicht weiter betrachtet. Die übrigen Krylov-Unterraum-Verfahren konnten durch Vorkonditionierung wesentlich beschleunigt werden. Auf das MINRES-Verfahren war es jedoch nicht möglich die Jacobi-Vorkonditionierung anzuwenden und bei der symmetrischen Gauß-Seidel (SGS) Vorkonditionierung kam es bei 46 LGS nicht zur Konvergenz. Mittels der unvollständigen

Tabelle 2 – Laufzeitvergleich verschiedener Verfahren; Abbruchbed.:$||\boldsymbol{r}|| < 0,1$; Erste Näherung: linear extrapoliert; Mittelung: 5 Zeitwerte; $*$46 Mal nicht konvergiert bei max. 100 Iterationen

	CG	GMRES	MINRES	RFOM	GSR	CZ
Vorkonditionierung :						
ohne	42,34 s	91,00 s	47,38 s	149,18 s	16,73 s	3,96 s
Jacobi	17,33 s	42,30 s	-	-	-	-
SGS	13,55 s	21,00 s	19,92 s $*$	-	-	-
UCZ	9,41 s	12,41 s	16,06 s	-	-	-

Cholesky-Zerlegung (USZ) ist es gelungen den Referenzalgorithmus (GSR) mit allen bis hierhin betrachteten Algorithmen hinsichtlich der Berechnungszeit zu unterbieten. Als der mit Abstand schnellste Algorithmus stellte sich jedoch die Cholesky-Zerlegung heraus. Der bei großen Koeffizientenmatrizen eigentlich ineffiziente Algorithmus konnte durch ausschließliches Speichern der Einhüllenden der Matrix wesentlich beschleunigt werden. Die zwei besten Verfahren CZ und CG wurde als auswählbare Löser im VTL in C++ implementiert. Anhand von vier zu synthetisierenden Sätzen wurde ein erneuter Laufzeitvergleich angestellt, wobei die Länge der Sprachsequenz der Berechnungszeit gegenüber gestellt wurde (siehe Abbildung 3). Durch Approximation der ersten Näherungslösung durch lineare Extrapolation konnte die Berechnungszeit bereits auf ca. 70 % reduziert werden. Der Effekt wird jedoch durch einen steileren Residuenverlauf vermindert (beispielsweise durch SGS-Vorkonditionierung auf ca. 90 %). Mittels SGS-Vorkonditionierung konnte die Berechnungszeit auf 40 % gesenkt werden. Somit ist die Sprachsequenz nur noch geringfügig länger als die Berechnungszeit. In der C++ Implementierung erzielt die SGS- gegenüber der UCZ-Vorkonditionierung die besseren Resultate. Am schnellsten ist aber auch hier die Cholesky-Zerlegung, welche lediglich die halbe Zeit der Sprachsequenz zu Berechnung benötigt. Die Berechnung konnte im Vergleich zum ursprünglich im VTL implementierten Algorithmus um das Sechsfache beschleunigt werden.

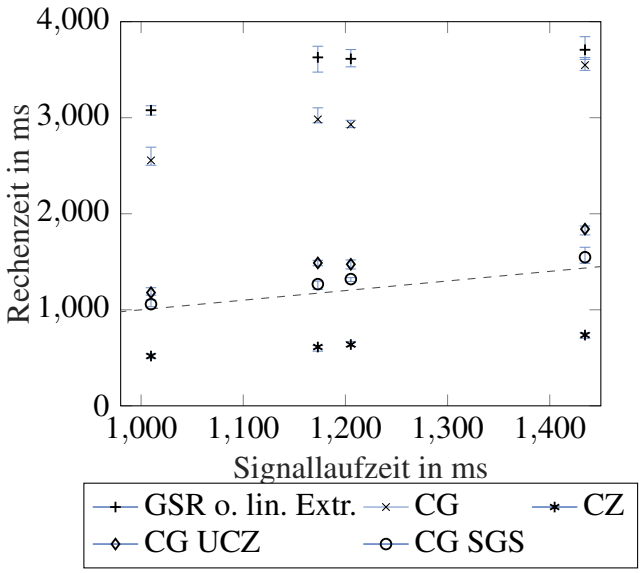

Abbildung 3 – Berechnugszeiten im VTL; Algorithmen mit linearer Extrapolation bis auf GSR

6 Zusammenfassung und Ausblick

Bisher erreichte die artikulatorische Synthese mit dem VTL einen Echtzeitfaktor von ca. 4,5, das heißt die Berechnungen beanspruchten 4,5 mal soviel Zeit wie die zu synthetisierende Äußerung. Im Rahmen dieser Untersuchung ist es gelungen, die benötigte Rechenzeit soweit zu reduzieren, dass bis zu Echtzeitfaktor 2 erreicht wurde. Ausschlaggebend war ein effizienteres Verfahren zur Simulation des Schallfeldes im Vokaltrakt. Mittels der Cholesky-Zerlegung und einer auf den Algorithmus sowie auf das LGS angepassten Speichermethode wurde es möglich, die Berechnungszeit um das Fünf- bis Sechsfache gegenüber dem bisher verwendeten Gauß-Seidel-Relaxationsverfahren zu verkürzen. Außerdem erreicht die Cholesky-Zerlegung eine theoretisch exakte Lösung, die nur noch von der Maschinengenauigkeit abhängt.
Mit den Krylov-Unterraum-Verfahren konnte keine Echtzeitfähigkeit erreicht werden. Das mit-

hilfe des symmetrischen Gauß-Seidel-Verfahrens vorkonditionierte Verfahren der konjugierten Gradienten führte trotzdem zu guten Resultaten. Im Vergleich zum ursprünglichen Verfahren ist es um ca. Faktor 2,4 schneller. Um die Lösung des LGS noch weiter zu beschleunigen, könnten in folgenden Untersuchungen weitere Vorkonditionierung betrachtet werden.

Literatur

[1] SCHRÖDER, M., M. CHARFUELAN, S. PAMMI, und I. STEINER: *Open source voice creation toolkit for the MARY TTS Platform*. In *12th Annual Conference of the International Speech Communication Association-Interspeech 2011*, S. 3253–3256. ISCA, 2011.

[2] ENGWALL, O.: *Synthesizing static vowels and dynamic sounds using a 3d vocal tract model*. In *4th ISCA Tutorial and Research Workshop (ITRW) on Speech Synthesis*. 2001.

[3] BIRKHOLZ, P.: *Modeling consonant-vowel coarticulation for articulatory speech synthesis*. *PLoS ONE*, 8(4), 2013.

[4] AALTO, D., O. AALTONEN, R.-P. HAPPONEN, P. JÄÄSAARI, A. KIVELÄ, J. KUORTTI, J.-M. LUUKINEN, J. MALINEN, T. MURTOLA, R. PARKKOLA ET AL.: *Large scale data acquisition of simultaneous MRI and speech*. *Applied Acoustics*, 83, S. 64–75, 2014.

[5] BADIN, P., G. BAILLY, M. RAYBAUDI, und C. SEGEBARTH: *A three-dimensional linear articulatory model based on MRI data*. In *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*. 1998.

[6] BADIN, P., G. BAILLY, L. REVERET, M. BACIU, C. SEGEBARTH, und C. SAVARIAUX: *Three-dimensional linear articulatory modeling of tongue, lips and face, based on MRI and video images*. *Journal of Phonetics*, 30(3), S. 533–553, 2002.

[7] PAYAN, Y. und P. PERRIER: *Synthesis of vv sequences with a 2d biomechanical tongue model controlled by the equilibrium point hypothesis*. *Speech Communication*, 22(2-3), S. 185–205, 1997.

[8] FELS, S., F. VOGT, K. VAN DEN DOEL, J. LLOYD, I. STAVNESS, und E. VATIKIOTIS-BATESON: *Artisynth: A biomechanical simulation platform for the vocal tract and upper airway*. In *International Seminar on Speech Production, Ubatuba, Brazil*, Bd. 138. Citeseer, 2006.

[9] MERMELSTEIN, P.: *Articulatory model for the study of speech production*. *The Journal of the Acoustical Society of America*, 53(4), S. 1070–1082, 1973.

[10] TEIXEIRA, A. J., R. MARTINEZ, L. N. SILVA, L. M. JESUS, J. C. PRÍNCIPE, und F. A. VAZ: *Simulation of human speech production applied to the study and synthesis of european portuguese*. *EURASIP Journal on Applied Signal Processing*, 2005, S. 1435–1448, 2005.

[11] ÖHMAN, S. E.: *Coarticulation in VCV utterances: Spectrographic measurements*. *The Journal of the Acoustical Society of America*, 39(1), S. 151–168, 1966.

[12] ÖHMAN, S. E.: *Numerical model of coarticulation*. *The Journal of the Acoustical Society of America*, 41(2), S. 310–320, 1967.

[13] BLANDIN, R., M. ARNELA, R. LABOISSIÈRE, X. PELORSON, O. GUASCH, A. V. HIRTUM, und X. LAVAL: *Effects of higher order propagation modes in vocal tract like geometries*. *The Journal of the Acoustical Society of America*, 137(2), S. 832–843, 2015.

[14] ARNELA, M., S. DABBAGHCHIAN, R. BLANDIN, O. GUASCH, O. ENGWALL, A. VAN HIRTUM, und X. PELORSON: *Influence of vocal tract geometry simplifications on the numerical simulation of vowel sounds*. *The Journal of the Acoustical Society of America*, 140(3), S. 1707–1718, 2016.

[15] BIRKHOLZ, P.: *3D-Artikulatorische Sprachsynthese*. Logos-Verlag, Berlin, 2005.

[16] KANZOW, C.: *Numerik linearer Gleichungssysteme: Direkte und iterative Verfahren*. Springer-Verlag, Heidelberg, 2007.

[17] SAAD, Y.: *Iterative methods for sparse linear systems*. SIAM, Philadelphia, PA, 2003.

[18] MEISTER, A.: *Numerik linearer Gleichungssysteme*. Springer-Verlag, Wiesbaden, 2015.