Ein modifiziertes Newtonverfahren mit selektiver Kopplung zur Lösung von hydrogeochemischen Mehrkomponentenmodellen

Den Naturwissenschaftlichen Fakultäten
der Friedrich-Alexander-Universität Erlangen-Nürnberg
zur
Erlangung des Doktorgrades

vorgelegt von
Stephan Oßmann
aus Lichtenfels

Als Dissertation genehmigt von den Naturwissenschaftlichen
Fakultäten der Universität Erlangen-Nürnberg

Tag der mündlichen Prüfung:	28. Juli 2008
Vorsitzender der Promotionskommission:	Prof. Dr. E. Bänsch
Erstberichterstatter:	Prof. Dr. P. Knabner
Zweitberichterstatter:	PD Dr. M. Bause

Danksagung

An dieser Stelle möchte ich Herrn Prof. Dr. Knabner für die Betreuung der Arbeit danken. Ein besonderer Dank gilt Herrn Dr. Alexander Prechtel, dessen Bürotür stets offen stand und der mit zahlreichen Anregungen zum Gelingen dieser Arbeit beitrug. Des Weiteren danke ich Herrn Joachim Hoffmann für seine Hilfe bei der Einarbeitung in die Simulationssoftware *M++* und Herrn PD Dr. Markus Bause sowie Herrn Dr. Florin Radu, die mir vor allem zu Beginn der Promotionszeit hilfreich zur Seite standen.

Außerdem danke ich allen Mitarbeiterinnen und Mitarbeitern des Lehrstuhls für Angewandte Mathematik I der Friedrich-Alexander-Universität Erlangen-Nürnberg für das hervorragende Arbeitsklima.

Erlangen, Mai 2008,
Stephan Oßmann

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

1.1 Angewandte Mehrkomponentenmodelle

Mit der in den letzten Jahrzehnten immer schneller fortschreitenden Technisierung gewinnt die numerische Simulation naturwissenschaftlicher Vorgänge zunehmend an Bedeutung. Viele Vorhaben können erst mit ihrer Hilfe realisiert werden. Sie ist ein wichtiges Werkzeug bei der Entwicklung neuer Produkte und hilft uns, naturwissenschaftliche Zusammenhänge besser zu verstehen. Im Zuge des Klimawandels leisten numerische Simulationen darüber hinaus große Dienste bei der Verbesserung bestehender und der Erschaffung neuer klimaschonender Technologien.

Eine wichtige Klasse mathematischer Modelle zur Berechnung der betrachteten Vorgänge sind die Mehrkomponenten- oder Mehrphasenmodelle. Mit ihnen wird das Verhalten mehrerer miteinander wechselwirkender Komponenten (auch Phasen oder Spezies) nachgebildet.

Zur Demonstration der vielfältigen Einsatzmöglichkeiten dieser Modelle folgt eine kleine Auswahl aktueller wissenschaftlicher Arbeiten, bei denen sie verwendet wurden:

- Speicherung von CO_2 im Boden [32].
- Traglastanalyse gesättigter Böden [26].
- Simulation von Murenabgängen [50].
- Wechselwirkung der Heliosphäre mit kosmischer Strahlung [9].
- Beschreibung des Deformationsverhaltens von Asphalt [2].

Am Department für Mathematik der Friedrich-Alexander-Universität Erlangen-Nürnberg werden vor allem zwei Mehrkomponentenmodelle angewandt:

1. Das reaktive Stofftransportmodell zur Simulation von Reaktionen zwischen (chemischen und biologischen) Spezies und deren Transport [33, 47].

2. Das Modell zur Simulation von präferentiellem Fließen in strukturierten porösen Medien [46].

Beide Modelle führen nach ihrer Diskretisierung mit einem impliziten Verfahren auf ein nichtlineares Gleichungssystem, das mit dem Newton-Verfahren linearisiert wird. Die resultierenden linearen Gleichungssysteme können dann mit einem geeigneten linearen Löser berechnet werden.

Eine positive Eigenschaft des Newton-Verfahrens ist seine quadratische Konvergenz. Allerdings erfordert es einen hohen Assemblierungsaufwand und auch das Lösen der linearen Gleichungssysteme kann viel Rechenzeit in Anspruch nehmen. Ziel der vorliegenden Arbeit ist deshalb, diesen Aufwand durch Einsatz eines modifizierten Newton-Verfahrens zu reduzieren.

Der neu zu entwickelnde Newton-Löser soll dabei für beliebige Mehrkomponentenmodelle eingesetzt und mit möglichst wenigen und einfachen Parametern gesteuert werden können. Auf diese Weise soll sichergestellt werden, dass er auch von Anwendern ohne fundierte mathematische Vorkenntnisse eingesetzt werden kann.
Der angestrebte Rechenzeitgewinn soll durch Ausdünnung der Jacobi-Matrix erzielt werden. Dabei sind gezielt „kleine“ Einträge auf Null zu setzen, die für die Kopplung der durch die Spezies festgelegten Teilsysteme verantwortlich sind.
Die Kopplungsterme müssen nicht mehr assembliert werden. Außerdem zerfällt das lineare Gleichungssystem durch eine geschickte Ausdünnung in voneinander unabhängige Teilsysteme. Diese können unter Umständen mit einem geringeren Aufwand gelöst werden als das Gesamtsystem.

1.2 Aktueller Stand der Entwicklung

Als Testmodell wird der schon angesprochene reaktive Stofftransport verwendet. Auf Arbeiten zu dessen Modellierung und Diskretisierung kann dabei zurückgegriffen werden [33, 47]. Darüber hinaus existiert bereits eine Implementierung dieses Modells in der Simulationssoftware *M++* [63]. Der dort vorhandene Newtonlöser kann als Programmiergrundlage für das hier zu entwickelnde modifizierte Verfahren dienen.

In [20] befindet sich eine sehr umfassende Auflistung modifizierter Newton-Verfahren. Auch das in dieser Arbeit zu konstruierende Verfahren wird angesprochen. Allerdings sind die Ausführungen nicht sehr detailliert.

Des weiteren existieren zahlreiche Artikel zu diesem Thema, die in der Regel relativ einfache und wenig praxisrelevante Modellgleichungen verwenden und Aussagen treffen, die nur für diese gültig sind. Ein auf allgemeine (auch aufwändige) Mehrkomponentenmodelle anwendbares Verfahren ist nicht bekannt.

Im Artikel [51] und der Dissertation [47] wurde eine entsprechende Methode für den reaktiven Mehrkomponentenstofftransport im eindimensionalen Fall vorgestellt. Die dort entwickelten Vorgehensweisen eignen sich im Eindimensionalen sehr gut zur Beschleunigung des Newton-Verfahrens. Es wird ein direkter Listenlöser entwickelt, der dank der speziellen tridiagonalen Blockstruktur der Jacobi-Matrix in $1D$ äußerst effizient arbeitet.
In $2D$ ist das Band der Matrix wesentlich größer, so dass der Einsatz von direkten Lösern wegen des größeren Rechenzeitaufwandes im Vergleich zu den iterativen Verfahren kritisch zu sehen ist. Die Methoden aus [51] und [47] scheinen deshalb im in dieser Arbeit betrachteten zweidimensionalen Fall weniger geeignet.
Darüber hinaus muss in den zitierten Arbeiten die Aufteilung des gesamten linearen Gleichungssystems in Teilsysteme manuell vorgegeben werden. Dies ist bei manchen Beispielen gut möglich. Bei umfangreichen für die Anwendung relevanten Simulationen ist jedoch ein automatisches Aufteilen wünschenswert.

Das in dieser Arbeit vorgestellte modifizierte Newton-Verfahren soll deshalb auf den räumlich zweidimensionalen Fall abgestimmt sein und eine automatische Aufteilung des Gesamtsystems ohne a-priori-Wissen über das Modell ermöglichen.

1.3 Gliederung

Das folgende zweite Kapitel beinhaltet die Beschreibung der Modellgleichungen mit den benötigten Koeffizientenfunktionen und Reaktionstermen und deren Diskretisierung mit konformen Finiten Elementen. Für den räumliche zweidimensionalen Fall werden die benötigten Variationsformulierungen und das nichtlineare Gleichungssystem hergeleitet, das anschließend mit dem Newton-Verfahren linearisiert wird. Nach der Angabe des linearen Gleichungssystems und einer Diskussion der Struktur der Jacobi-Matrix schließt dieses Kapitel mit der Definition einiger wichtiger Kenngrößen zur Beurteilung des Problems und seiner Diskretisierung.

Inhalt des dritten Kapitels ist die Entwicklung des modifizierten Newton-Verfahrens. Hier wird zunächst dargelegt, dass das lineare Gleichungssystem (näherungsweise) in mehrere unabhängige Teilsysteme zerfallen kann, die getrennt voneinander lösbar sind. Anschließend wird untersucht, wie groß der mögliche Rechenzeitgewinn unter Verwendung direkter und iterativer Löser ist und wie stark der Aufwand zur Assemblierung reduziert werden kann. Die hier gewonne-

nen Erkenntnisse münden in die Konstruktion des Newton-Lösers *NewtonBlock*, der ausführlich vorgestellt wird.
Als nächster Schritt kann das Konvergenzverhalten des neuen Verfahrens untersucht werden. Hier sind lineare und nichtlineare Probleme zu unterscheiden. Die gewonnenen Konvergenzaussagen werden in beiden Fällen durch Testsimulationen verifiziert.
Als weitere Tests von *NewtonBlock* folgen schließlich zahlreiche Vergleichssimulationen zu der bereits zitierten Arbeit [47]. Zusätzlich werden noch Berechnungen mit unterschiedlichen zeitlichen und räumlichen Schrittweiten durchgeführt. Auf diese Weise soll das Potential des modifizierten Newton-Verfahrens erprobt werden.
Als Konsequenz dieser Tests wird schließlich das Broyden-Verfahren eingeführt. Mit seiner Hilfe kann der Rechenzeitgewinn in *NewtonBlock* unter Verwendung direkter Löser noch vergrößert werden. Auch hier folgen Testsimulationen zur Überprüfung der Anwendbarkeit.

Im vierten Kapitel werden die bisher manuell vorgenommenen Festlegungen der Zeitschrittweite und der Teilsysteme automatisiert. Ersteres geschieht mit einem zeitadaptiven Verfahren, zweiteres mit einer geeigneten Entkopplungsstrategie, wobei eine allgemein anwendbare Strategie und eine auf den linearen Fall beschränkte vorgestellt werden. Beide automatisierten Verfahren werden auch hier ausgiebig getestet.

Das in dieser Arbeit entwickelte modifizierte Newton-Verfahren wird im fünften Kapitel an zwei praxisorientierten Simulationen erprobt. Dabei handelt es sich um das EDTA-Problem, bei dem kein Fließen stattfindet und das Barriere-Problem mit einem komplexen Fließgeschehen. Die Performance des neuen Verfahrens wird für beide Simulationen dokumentiert und bewertet.

Im Anhang A befinden sich einige Bemerkungen zur Implementierung in *M++* und eine ausführliche Erklärung, wie *NewtonBlock* anzuwenden ist und welche Einstellungen vorzunehmen sind. Das als direkter Löser eingesetzte *SuperLU*-Verfahren wird vorgestellt und die Datenstrukturen von *M++* erläutert. Anschließend sind die wichtigsten neu implementierten Funktionen aufgelistet und kommentiert.

Zum Abschluss dieser Arbeit erfolgt im Anhang eine kurze Zusammenfassung der Ergebnisse in deutscher und englischer Sprache.

Kapitel 2

Modellgleichungen und Diskretisierung

In diesem Kapitel werden die Modellgleichungen zur Berechnung des reaktiven Mehrkomponentenstofftransports in porösen Medien kurz vorgestellt und mit der Methode der konformen Finiten Elemente diskretisiert. Die resultierenden nichtlinearen Gleichungen werden mit dem Newton-Verfahren unter Angabe der Jacobi-Matrix und der rechten Seite gelöst. Das Kapitel schließt mit einer Analyse der Struktur der Jacobi-Matrix, einigen Hinweisen zur effizienten Lösung des linearen Gleichungssystems und der Angabe dreier wichtiger Kenngrößen.

2.1 Modellierung

Im ersten Abschnitt des Kapitels Modellgleichungen und Diskretisierung stellen wir die Modellgleichungen auf. Wir verzichten auf eine ausführliche Erklärung der physikalischen Grundlagen und verweisen auf die zitierte Literatur.

Die Modellgleichungen werden im Folgenden stets auf dem Zeit-Raum-Zylinder $\mathcal{J} \times \Omega$ mit dem Zeitintervall $\mathcal{J} := (0, T)$, $T > 0$ und dem offenen zusammenhängenden Gebiet $\Omega \subset \mathbb{R}^d$, $d \in \{1, 2, 3\}$, betrachtet. Den Rand von Ω bezeichnen wir mit $\partial\Omega$, seine Normale mit $\mathbf{n}$.

2.1.1 Die Modellgleichungen

Durch die Modellgleichungen soll der reaktive Mehrkomponententransport in einem porösen Medium mit variablem Wassergehalt, Diffusion/Dispersion und Konvektion für N_S Spezies und N_R Reaktionen nachgebildet werden. Wir verwenden die z.B. in [22] hergeleitete Formulierung

$$\partial_t \left(\Theta c_i\right) - \nabla \cdot \left(D_i \nabla c_i - \mathbf{q} c_i\right) \;=\; \sum_{j=1}^{N_R} \nu_{ij} \Theta R_j \left(c_1, \ldots, c_{N_S}\right) \tag{2.1}$$

für die Spezies $i \in \{1, \ldots, N_S)$. $\Theta = \Theta\left(\mathbf{x}, t\right\} \in \mathbb{R}$ bezeichnet den Wassergehalt des Mediums, $c_i = c_i\left(\mathbf{x}, t\right) \in \mathbb{R}$ die Konzentration, $D_i = D_i\left(\mathbf{x}, t\right) \in \mathbb{R}^{d,d}$ einen Diffusions/Dispersions-Tensor, $\mathbf{q} = \mathbf{q}\left(\mathbf{x}, t\right) \in \mathbb{R}^d$ den (Wasser-)Fluss und $R_i = R_i\left(c_1, \ldots, c_{N_S}\right) \in \mathbb{R}$ einen Reaktionsterm. Die Einheiten der einzelnen Größen sind Tabelle 2.1 zu entnehmen.

Tabelle 2.1: Verwendete Größen und deren Einheiten

Größe	Bezeichnung	Einheit
c	Konzentration	Masse/Länge^2
Θ	Wassergehalt	-
D	Diffusions/Dispersions-Tensor	Länge^2/Zeit
$\mathbf{q}$	(Wasser-)Fluss	Länge/Zeit
R	Reaktionsterm	Masse/(Länge^2 $\cdot$ Zeit)

Anfangs- und Randwerte

Die Lösung der partiellen Differentialgleichungen (2.1) erfordert die Vorgabe geeigneter Anfangs- und Randwerte:

Die Anfangswerte der Stoffkonzentration c_i sind gegeben durch

$$\begin{aligned} c_i^0 : \Omega &\to \mathbb{R}, \\ \mathbf{x} &\mapsto c_i^0\left(\mathbf{x}\right). \end{aligned}$$

Der Rand $\partial\Omega$ zerfalle in die disjunkten Teile $\partial\Omega_{C,i}$ und $\partial\Omega_{N,i}$. Auf $\partial\Omega_{C,i}$ werden Dirichlet-Randwerte gegeben durch

$$\begin{aligned} f_{C,i} : \partial\Omega_{C,i} \times \mathcal{J} &\to \mathbb{R}, \\ \left(\mathbf{x}, t\right) &\mapsto f_{C,i}\left(\mathbf{x}, t\right), \end{aligned}$$

und auf $\partial\Omega_N$ homogene Neumann-Randwerte.Die jeweiligen Randwerte lauten

$$\begin{aligned} c_i &= f_{C,i} \quad \text{auf} \quad \mathcal{J} \times \partial\Omega_{C,i}, \\ D_i \nabla c_i \cdot \mathbf{n} &= 0 \quad\;\; \text{auf} \quad \mathcal{J} \times \partial\Omega_{N,i}. \end{aligned}$$

2.1.2 Die Koeffizientenfunktionen

Neben den Anfangs- und Randwerten sind zur Lösung der Modellgleichungen (2.1) die Koeffizienten Θ, $\mathbf{q}$ und D vorzugeben:

Die Koeffizienten Θ und $\mathbf{q}$

Bei der Simulation realistischer Transportprozesse werden der Wassergehalt Θ und der Wasserfluss $\mathbf{q}$ häufig mit einer Richards-Gleichung

$$\begin{aligned} \partial_t \Theta(\psi) + \nabla \cdot \mathbf{q} &= 0, \\ \mathbf{q} &= -\nabla \cdot K(\psi) \nabla(\psi + z), \end{aligned} \tag{2.2}$$

berechnet, die das Fließen von Wasser in einem porösen Medium beschreibt. $\Theta(\psi)$ und $K(\psi)$ sind hier gegebene Koeffizientenfunktionen, z die Höhe der Wassersäule über einem vordefinierten Nullniveau und ψ der gesuchte Wasserdruck.
Wir gehen im Folgenden davon aus, dass Θ und $\mathbf{q}$ explizit gegeben sind und verweisen für Einzelheiten zur Modellierung des (präferenziellen) Fließens mit der Richards-Gleichung auf die Arbeiten [5, 22, 46, 48, 57].

Der Diffusions-Dispersions-Koeffizient D

Im Eindimensionalen ist der Diffusions-Dispersions-Koeffizient definiert als

$$D_i := \alpha_l \mathbf{q} + d_{D,i}\Theta \tag{2.3}$$

mit der Dispersionslänge α_l [Länge] und der Diffusion d_D [Länge^2/Zeit].
Im Mehrdimensionalen existieren verschiedene Matrixdarstellungen wie z.B. der von Scheidegger [55] vorgeschlagene Tensor

$$D_i := -\left((\Theta D_{diff,i} + \beta_t \|\mathbf{q}\|_2) I + (\beta_l - \beta_t) \frac{\mathbf{q} \otimes \mathbf{q}}{\|\mathbf{q}\|_2} \right) \tag{2.4}$$

mit den Diffusionskoeffizienten $D_{diff,i}$, den longitudinalen und transversalen Dispersionskoeffizienten β_l [Länge] und β_t [Länge] und der Einheitsmatrix I. Er ist wie die meisten gebräuchlichen Darstellungen symmetrisch und positiv definit, so dass diese beiden Eigenschaften im Folgenden vorausgesetzt werden.

2.1.3 Die Reaktionsterme

Reaktionen zwischen Spezies werden durch die N_R Reaktionsterme R_j, $j = 1, \ldots, N_R$, beschrieben.
Wir betrachten Reaktionen $N_{S_{mob}}$ mobiler und $N_{S_{imob}}$ immobiler Spezies, zu denen auch mikrobielle Spezies zählen, mit $N_{S_{mob}} + N_{S_{imob}} = N_S$. Die Konzentrationswerte immobiler Spezies werden zur besseren Unterscheidung künftig mit $\bar{c}_i$ gekennzeichnet und wir verwenden die Notation

$$\mathbf{c} := \left(c_1, \ldots, c_{N_{S_{mob}}}\right), \qquad \bar{\mathbf{c}} := \left(\bar{c}_{N_{S_{mob}}+1}, \ldots, \bar{c}_{N_S}\right).$$

Dabei wird gefordert, dass die Indizes wie oben angegeben nach mobilen und immobilen Spezies sortiert sind.

Mobile Spezies werden transportiert und es gilt die Erhaltungsgleichung

$$\partial_t \left(\Theta c_i\right) - \nabla \cdot \left(D_i \nabla c_i - \mathbf{q}_i c_i\right) = \sum_{j=1}^{N_R} \nu_{ij} \Theta R_j \left(\mathbf{c}, \bar{\mathbf{c}}\right) \tag{2.5}$$

für die Spezies i. Die stöchiometrischen Faktoren ν_{ij} $[-]$ der $i-$ten Spezies in der $j-$ten Reaktion sind gegebene Konstanten. Sie werden z.B. in [47] ausführlich beschrieben.
Anders als die mobilen werden immobile Spezies nicht transportiert, sondern lediglich transformiert. Für sie gilt die Erhaltungsgleichung

$$\rho_B \partial_t \bar{c}_i \;=\; \Theta \sum_{j=1}^{N_{R_i}} \nu_{ij} R_{ij} \left(\mathbf{c}, \bar{\mathbf{c}}\right) \tag{2.6}$$

mit der Masse Boden pro Gesamtvolumen (Bulk Density) ρ_B [Masse/Volumen]. Wegen ihrer Immobilität müssen für sie keine Randwerte angegeben werden.

In der vorliegenden Arbeit werden drei Reaktionstypen verwendet:
Der **Abbau 0. und 1. Ordnung** der $i-$ten Spezies ist

$$R_j \;=\; -k_j c_{i(j)} - C_j, \tag{2.7}$$

wobei k_i die Abbaurate 1. Ordnung bezeichnet und C_i eine Konstante.
Die **stöchiometrische kinetische Reaktion** [6] ist definiert als

$$R_j \;=\; k_j^f \prod_{\{k|\nu_{kj}<0\}} c_k^{-\nu_{kj}} \;-\; k_j^b \prod_{\{k|\nu_{kj}>0\}} c_k^{\nu_{kj}} \tag{2.8}$$

und beschreibt eine chemische Reaktion nach dem Massenwirkungsgesetz.
Bei der **Biodegradations-Reaktion (Monod-Reaktion)** [56, 62]

$$R_j = -\mu_{max} c_{X_{i(j)}} \prod_{k \in I_j \subset \{1,\ldots,N_S\}} \left(\frac{c_k}{K_{M_k} + c_k}\right) \prod_{k \in J_j \subset \{1,\ldots,N_S\}} \frac{K_{I_k}}{K_{I_k} + c_k} \tag{2.9}$$

wirkt eine mikrobielle Spezies c_{X_i} als Katalysator einer Abbaureaktion, bei der die Spezies $I_j \subset \{1, \ldots, N_S\}$ verbraucht werden, während andere Spezies $J_j \subset \{1, \ldots, N_S\} \setminus I_j$ die Reaktion hemmen. Bei diesem Reaktionsterm sind die Parameter μ_{max} [1/Zeit] (max. spezifische Wachstumsrate) und K_I [Masse/Länge^2] (haldane Inhibitionskonzentration) vorzugeben.

Zugunsten einer einfacheren Darstellung wurde hier auf die Kennzeichnung der immobilen Spezies verzichtet.

Eine ausführliche Modellierung, Simulation und Analyse des Stofftransports und -abbaus erfolgt in der Dissertation [47] und der Diplomarbeit [33].

2.2 Diskretisierung

Ausgangspunkt für die Diskretisierung sind die Erhaltungsgleichungen (2.5) der mobilen und (2.6) der immobilen Spezies, wobei die erste Gleichung eine partielle Differentialgleichung in t und $\mathbf{x}$ darstellt und die zweite eine gewöhnliche Differentialgleichung in t.

Wir verwenden im Folgenden ausschließlich Sprechweisen des räumlich zweidimensionalen Modells. Die Überlegungen gelten jedoch auch im Ein- und Dreidimensionalen ganz entsprechend.

2.2.1 Variationsformulierung

Wir erstellen zunächst die Variationsformulierungen für (2.5) und diskretisieren anschließend (2.6) unter Verwendung der z.B. in [10, 37] vorgestellten Methode der konformen Finiten Elemente.

Kontinuierliche Variationsformulierung

Nach Multiplikation von (2.5) mit geeigneten Testfunktionen und Integration über Ω folgt mit

$$\begin{aligned}(a,b) &:= \int_{\Omega} ab \; d\mathbf{x},\\ \langle a,b\rangle &:= \int_{\partial\Omega} ab \; d\sigma,\end{aligned}$$

und dem kontinuierlichen Ansatzraum

$$H^1_{0,D}(\Omega) := \left\{ v \in H^1(\Omega) \,|\, \gamma_0(v) = 0 \text{ auf } \partial\Omega_D \right\}$$

die kontinuierliche Variationsformulierung

$$\left(\partial_t(\Theta c_i), v\right) + \left(D_i \nabla c_i, \nabla v\right) + \left(\nabla\cdot(\mathbf{q}c_i), v\right) = \left(\sum_{j=1}^{N_R} \nu_{ij}\Theta R_j, v\right) + \langle D_i \nabla c_i \cdot \mathbf{n}, v\rangle$$

für alle $v \in V$ mit dem Ansatzraum $V := H^1_{0,D}(\Omega)$.

Wir betrachten zunächst nur homogene Dirichlet-Randwerte, so dass der Randterm auf der rechten Seite wegen $v = 0$ auf $\partial\Omega_D$ wegfällt.

Die Ableitung $\partial_t\Theta$ des Wassergehaltes kann mit Hilfe der Massenerhaltung für Wasser

$$\partial_t\Theta + \nabla\cdot\mathbf{q} = 0$$

aus der Variationsformulierung eliminiert werden.

Volldiskrete Variationsformulierung

Zur Erstellung der räumlichen Diskretisierung sei $\mathcal{T}$ eine konforme Triangulierung des Gebietes Ω mit Dreiecken T und $\mathcal{K}$ die Menge der Knoten. $\mathcal{K}$ zerfalle in die Teilmengen der inneren Knoten $\mathcal{K}_I$ und der Knoten auf den einzelnen Randteilen $\mathcal{K}_D$ und $\mathcal{K}_N$, d.h.

$$\mathcal{K} \;=\; \mathcal{K}_I \cup \mathcal{K}_D \cup \mathcal{K}_N.$$

Die Knoten a_i seien so nummeriert, dass $a_i \in \mathcal{K}_I \cup \mathcal{K}_N$ für $i \in \{1, \ldots, M_1\}$ und $a_i \in \mathcal{K}_D$ für $i \in \{M_1 + 1, \ldots, M\}$.

Die zeitliche Diskretisierung erfolgt durch Aufteilung des Zeitintervalls $\mathcal{J}$ in N Teilintervalle

$$\mathcal{J}_i \;:=\; \left[t^{i-1}, t^i\right], \quad i = 1, \ldots, N \quad \text{mit}$$
$$0 = t^0 < t^1 < \cdots < t^N = T, \qquad \Delta t^n := t^n - t^{n-1}.$$

Mit diesen Unterteilungen und dem z.B. in [37] beschriebenen impliziten Eulerverfahren folgt die volldiskrete Variationsformulierung:

Finde $c^n_{i,h} \in V_h$, $i \in \{1, \ldots, N_{S_{mob}}\}$, die für alle $v_h \in V_h$ und für alle Zeitpunkte t^n, $n \in \{0, \ldots, N\}$, Lösung sind von

$$\left(\Theta c^n_{i,h}, v_h\right) + \Delta t^n \left(D \nabla c^n_{i,h}, \nabla v_h\right) + \Delta t^n \left(\mathbf{q} \cdot \nabla c^n_{i,h}, v_h\right) \;=\; \Delta t^n \sum_{j=1}^{N_R} \nu_{ij} \left(\Theta R^n_j, v_h\right) + \left(\Theta c^{n-1}_{i,h}, v_h\right).$$

Die Erhaltungsgleichung der immobilen Spezies

Auch (2.6) wird mit dem impliziten Eulerverfahren gelöst. Wir ersetzen die Zeitableitung durch einen vorwärtsgenommenen Differenzenquotienten und erhalten

$$\rho_B \bar{c}^n_{i,h} \;=\; \Delta t^n \sum_{j=1}^{N_R} \nu_{ij} \Theta R_j \left(\mathbf{c}^n_h, \bar{\mathbf{c}}^n_h\right)$$

für $i \in \{N_{S_{mob}} + 1, \ldots, N_S\}$.

2.2.2 Aufstellen des Gleichungssystems

Wir definieren nun die Ansatzfunktionen für die diskreten Ansatzräume und stellen mit deren Hilfe das zu lösende Gleichungssystem auf.

Einbringen der Basisfunktionen

Für die Basisfunktionen φ_k des endlichdimensionalen Ansatzraumes V_h gelte $\varphi_k \in \mathcal{P}_1(T)$ und

$$\varphi_i(a_j) \;=\; \delta_{ij}.$$

Die exakte Definition der Basisfunktionen kann der zitierten Literatur entnommen werden.
Mit Hilfe der φ_i können die gesuchten Konzentrationswerte dargestellt werden als

$$c_{i,h}^n(\mathbf{x}) \;=\; \sum_{k=1}^{M} c_{i,k}^n \varphi_k(\mathbf{x}).$$

Gesucht sind also die Koeffizienten dieser Basisdarstellung $c_{i,k}^n$, $i = 1, \ldots, N_{S_{mob}}$, für die folgende Gleichungen gelten:

$$\sum_{k=1}^{M} \Big[\left(c_{i,k}^n - c_{i,k}^{n-1}\right)(\Theta\varphi_k, \varphi_j) + \Delta t^n c_{i,k}^n \left(D_i \nabla\varphi_k, \nabla\varphi_j\right) + \Delta t^n c_{i,k}^n \left(\mathbf{q}\cdot\nabla\varphi_k, \varphi_j\right) - \Delta t^n \left(R_i, \varphi_j\right) \Big] \;=\; 0 \qquad \text{für } j = 1, \ldots, M_1, \quad (2.10)$$

$$c_{i,j}^n - f_{D,i}^n(a_j) \;=\; 0 \qquad \text{für } j = M_1 + 1, \ldots, M. \quad (2.11)$$

Die gewöhnliche Differentialgleichung (2.6) soll punktweise in jedem Knoten erfüllt sein und es kommen die Gleichungen

$$\rho_B\left(\bar{c}_{i,j}^n - \bar{c}_{i,j}^{n-1}\right) - \Delta t^n R_i \;=\; 0 \qquad \text{für } i = 1, \ldots, N_{S_{imob}},\ j = 1, \ldots, M, \quad (2.12)$$

hinzu.

Das Newton-Verfahren

Das Gleichungssystem bestehend aus (2.10), (2.11) und (2.12) ist ein Nullstellenproblem in der Form

$$\mathbf{G}(\mathbf{x}) \;=\; 0$$

mit einer nichtlinearen vektorwertigen Funktion $\mathbf{G}$, das mit dem Newton-Verfahren [20, 54] gelöst werden kann. Dieses Verfahren konvergiert quadratisch und erfordert die Schritte

1. Löse das lineare Gleichungssystem

$$J^{(k)} \;:=\; D\mathbf{G}\left(\mathbf{x}^{(k)}\right)\boldsymbol{\delta}^{(k)} \;=\; -\mathbf{G}\left(\mathbf{x}^{(k)}\right). \quad (2.13)$$

2. Setze $\mathbf{x}^{(k+1)} \;=\; \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}$.

Diese werden wiederholt, bis ein vorgegebenes Abbruchkriterium erfüllt ist.
Wir verwenden stets das absolute Kriterium

$$\left\|\mathbf{G}\left(\mathbf{x}^{(k+1)}\right)\right\|_2 \;\leq\; \epsilon_{abs}. \quad (2.14)$$

Alternativ hierzu ist auch das relative Kriterium

$$\left\|\mathbf{G}\left(\mathbf{x}^{(k+1)}\right)\right\|_2 \;\leq\; \epsilon_{rel}\left\|\mathbf{G}\left(\mathbf{x}^{(0)}\right)\right\|_2$$

gebräuchlich.

Die benötigte Jacobi-Matrix J_k besteht aus den Blöcken

$$(D\mathbf{G})_{jk} \;=\; \begin{pmatrix} J_{jk}^{11} & J_{jk}^{12} \\ J_{jk}^{21} & J_{jk}^{22} \end{pmatrix}, \tag{2.15}$$

mit $j,k \in \{1,\ldots,M\}$, $(D\mathbf{G})_{jk} \in \mathbb{R}^{n,n}$, $J_{jk,il}^{11} \in \mathbb{R}^{N_{S_{mob}},N_{S_{mob}}}$, $J_{jk,il}^{12} \in \mathbb{R}^{N_{S_{mob}},N_{S_{imob}}}$, $J_{jk,il}^{21} \in \mathbb{R}^{N_{S_{imob}},N_{S_{mob}}}$ und $J_{jk,il}^{22} \in \mathbb{R}^{N_{S_{imob}},N_{S_{imob}}}$.

Für innere Knoten und Knoten auf dem Neumann-Rand gilt

$$\begin{aligned}
J_{jk,il}^{11} &= \delta_{il}\left[\frac{\Theta}{\Delta t^n}\left(\varphi_k,\varphi_j\right) + \left(D\nabla\varphi_k,\nabla\varphi_j\right) + \left(\mathbf{q}\cdot\nabla\varphi_k,\varphi_j\right)\right] \\
&\quad - \left(\frac{\partial R_i}{\partial c_l}\left(\mathbf{c},\overline{\mathbf{c}}\right)\varphi_k,\varphi_j\right), \\
J_{jk,il}^{12} &= -\left(\frac{\partial R_i}{\partial \bar{c}_l}\left(\mathbf{c},\overline{\mathbf{c}}\right)\varphi_k,\varphi_j\right), \\
J_{jk,il}^{21} &= -\delta_{jk}\frac{\partial R_i}{\partial c_l}\left(\mathbf{c},\overline{\mathbf{c}}\right), \\
J_{jk,il}^{22} &= \delta_{jk}\left(\delta_{il}\frac{\rho_B}{\Delta t^n} - \frac{\partial R_i}{\partial \bar{c}_l}\left(\mathbf{c},\overline{\mathbf{c}}\right)\right),
\end{aligned}$$

mit $j \in \{1,\ldots,M_1\}$, $k \in \{1,\ldots,M\}$ und für Knoten auf dem Dirichlet-Rand

$$\begin{aligned}
J_{jk,il}^{11} &= \delta_{jk}\delta_{il}, \\
J_{jk,il}^{12} &= 0, \\
J_{jk,il}^{21} &= -\delta_{jk}\frac{\partial R_i}{\partial c_l}\left(\mathbf{c},\overline{\mathbf{c}}\right), \\
J_{jk,il}^{22} &= \delta_{jk}\left(\delta_{il}\frac{\rho_B}{\Delta t^n} - \frac{\partial R_i}{\partial \bar{c}_l}\left(\mathbf{c},\overline{\mathbf{c}}\right)\right),
\end{aligned}$$

mit $j \in \{M_1+1,\ldots,M\}$ und $k \in \{1,\ldots,M\}$. Die Indizes i und l durchlaufen alle Elemente der Teilmatrizen aus (2.15).

2.2.3 Struktur der Jacobi-Matrix

Gegeben sei der in Abbildung 2.1 dargestellte Ausschnitt der Triangulierung mit vier Elementen, acht Kanten und fünf Knoten.

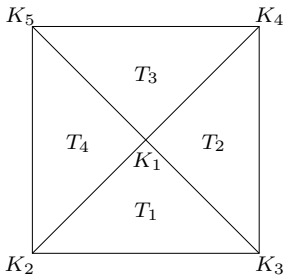

Abbildung 2.1: Ausschnitt der Triangulierung mit vier Dreiecken.

$$J \;=\; \left(\begin{array}{c|ccccc} & K_1 & K_2 & K_3 & K_4 & K_5 \\ \hline K_1 & \blacksquare & \square & \square & \square & \square \\ K_2 & \square & \blacksquare & \square & \circ & \square \\ K_3 & \square & \square & \blacksquare & \square & \circ \\ K_4 & \square & \circ & \square & \blacksquare & \square \\ K_5 & \square & \square & \circ & \square & \blacksquare \end{array}\right)$$

Abbildung 2.2: Ausschnitt aus der Struktur der Jacobi-Matrix J in knotenorientierter Darstellung mit vollbesetzten Blockmatrizen ■, Diagonalmatrizen □ und Nullmatrizen ∘.

Die gemäß Abschnitt 2.2.2 gebildete elementweise organisierte Jacobi-Matrix J hat die in Abbildung 2.2 angegebene Struktur.

Sie ist dünn besetzt und besteht aus im Allgemeinen vollbesetzten Blockmatrizen $J_{KK} \in \mathbb{R}^{N_S,N_S}$ (■), Diagonalblöcken $J_{KK'} \in \mathbb{R}^{N_S,N_S}$ (□) mit $K \neq K'$ und Nullmatrizen (∘). Pro Zeile und Spalte existiert genau ein vollbesetzter Block auf der Hauptdiagonalen und soviele Diagonalblöcke wie der Knoten K Nachbarknoten besitzt.

Die Anzahl der Nichtnullelemente γ lässt sich mit Hilfe der Graphentheorie abschätzen. Dazu interpretieren wir das durch die Triangulierung entstehende räumliche Gitter $\mathcal{M}$ (Mesh) als Graphen und benutzen

Definition 2.1 (Grad bzw. Valenz eines Knotens)**.** *Der Grad (auch Valenz) $d_{\mathcal{G}}(K)$ eines Knotens bezeichnet die Anzahl der Nachbarknoten von K im Graphen $\mathcal{G}$ und $d_{\mathcal{G}}^{max}$ den maximalen Grad eines Knotens von $\mathcal{G}$:*

$$d_{\mathcal{G}}^{\max} \;:=\; \max_{K \in \mathcal{G}} d_{\mathcal{G}}(K)\,.$$

Für die Jacobi-Matrix J ist

$$\begin{aligned}\gamma &= \sum_{K\in\mathcal{M}} \left(N_S^2 + N_S d_{\mathcal{M}}(K)\right) = N_S\left(MN_S + \sum_{K\in\mathcal{M}} d_{\mathcal{M}}(K)\right)\\ &\leq MN_S\left(N_S + d_{\mathcal{M}}^{\max}\right).\end{aligned}$$

In der verwendeten Simulationssoftware *M++* werden auch die Diagonalblöcke außerhalb der Hauptdiagonalen ($\square$) als volle Blöcke mit N_S^2 Einträgen gespeichert (siehe A.3). Die Anzahl der Nichtnullelemente erhöht sich auf

$$\begin{aligned}\gamma &= \sum_{K\in\mathcal{M}} N_S^2\left(1 + d_{\mathcal{M}}(K)\right) = N_S^2\left(M + \sum_{K\in\mathcal{M}} d_{\mathcal{M}}(K)\right)\\ &\leq MN_S^2\left(1 + d_{\mathcal{M}}^{\max}\right).\end{aligned} \tag{2.16}$$

Um eine Vergleichbarkeit der theoretischen Ausführungen mit den Simulationsergebnissen zu gewährleisten werden wir ab jetzt stets den etwas höheren Wert aus (2.16) verwenden.

Bemerkung 2.2. Bei einer Friedrichs-Keller-Triangulierung auf Rechtecksgebieten gilt für alle Knoten K

$$2 \leq d_{\mathcal{M}}(K) \leq 6.$$

$J =$

		S_1					S_2				
		K_1	K_2	K_3	K_4	K_5	K_1	K_2	K_3	K_4	K_5
	K_1	TR	T	T	T	T	R				
	K_2	T	TR	T		T		R			
S_1	K_3	T	T	TR	T				R		
	K_4	T		T	TR	T				R	
	K_5	T	T		T	TR					R
	K_1	R					TR	T	T	T	T
	K_2		R				T	TR	T		T
S_2	K_3			R			T	T	TR	T	
	K_4				R		T		T	TR	T
	K_5					R	T	T		T	TR

Abbildung 2.3: Ausschnitt aus der Struktur der Jacobi-Matrix J in speziesorientierter Darstellung. T kennzeichnet Anteile aus Transport-, R aus Reaktions- und TR aus Transport- und Reaktionstermen.

Durch Umordnen von Zeilen und Spalten kann J speziesorientiert dargestellt werden. Abbildung 2.3 zeigt eine Skizze dieser Matrix für zwei Spezies A und B

mit der Triangulierung aus Abbildung 2.1, wobei T Einträge aus Transport- und R Einträge aus Reaktionstermen symbolisieren.
Falls die Reaktionsterme aus den Nebendiagonalblöcken verschwinden, zerfällt das lineare Gleichungssystem in zwei Teilsysteme, die in der Summe unter Umständen leichter lösbar sind als das Gesamtsystem. Diese Tatsache soll im folgenden Kapitel genutzt werden, um das Gleichungssystem möglichst effizient zu lösen.

2.2.4 Kenngrößen

Nach der Durchführung der Diskretisierung und der Analyse der Struktur der Jacobi-Matrix definieren wir noch einige wichtige Kenngrößen:

Definition 2.3 (Damköhler-Zahl)**.** *Die dimensionslose Damköhler-Zahl Da ist definiert als*

$$Da \;:=\; \frac{kL}{\|\mathbf{q}\|_2} \tag{2.17}$$

mit der Ratenkonstanten k [$1/Zeit$] und der charakteristischen Länge des Gebiets L [$Länge$].

Da kennzeichnet das Verhältnis der Ratenkonstanten k (Geschwindigkeit der Reaktion) zum Betrag des konvektiven Flusses $\|\mathbf{q}\|_2$ (Geschwindigkeit des Stofftransports).

Definition 2.4 (Péclet-Zahl)**.** *Die Péclet-Zahl Pe ist definiert als*

$$Pe \;:=\; \frac{\|\mathbf{q}\|_{\infty,\Omega}}{\|D\|_{\infty,\Omega}}\; diam\,(\Omega)\,, \tag{2.18}$$

die Zell-Péclet-Zahl Pe_T als

$$Pe_T \;:=\; \frac{\|\mathbf{q}\|_{\infty,T}}{\|D\|_{\infty,T}}\; diam\,(T)\,. \tag{2.19}$$

Sowohl Pe als auch Pe_T sind dimensionslos.

Die (Zell-)Péclet-Zahl dient im folgenden als Indikator für die Konvektionsdominanz der berechneten Beispiele. Für $Pe \gg 1$ spricht man von konvektionsdominierten Problemen.

Definition 2.5 (Courant-Zahl)**.** *Die dimensionslose Courant-Zahl Cr ist definiert als*

$$Cr \;:=\; \|\mathbf{q}\|_{\infty,\Omega}\,\frac{\Delta t}{h}. \tag{2.20}$$

Cr zeigt an, ob ein Problem auch mit einem expliziten Verfahren gelöst werden könnte. Im Fall $Cr \gg 1$ sind implizite Verfahren anzuwenden.

Kapitel 3

Das modifizierte Newton-Verfahren

Inhalt des Kapitels ist die Entwicklung des modifizierten Newton-Verfahrens. Dabei soll ausgenutzt werden, dass die Jacobi-Matrix aus Abschnitt 2.2.2 unter Umständen so ausgedünnt werden kann, dass das lineare Gleichungssystem (2.13) in mehrere voneinander unabhängige Teilsysteme zerfällt und dabei die Konvergenz des Verfahrens nicht zu sehr verschlechtert wird.

Im ersten Abschnitt werden die benötigten Teilsysteme definiert. Es folgt ein Vergleich iterativer und direkter Verfahren mit dem Ziel, das jeweilige Einsparpotential einzuschätzen. Auch der mögliche Gewinn durch Nichtassemblieren der wegfallenden Einträge in J wird beurteilt. Der Abschnitt schließt mit einer Diskussion des neu entwickelten modifizierten Newtonlösers *NewtonBlock*, in den die oben gewonnenen Erkenntnisse eingehen.

Das Konvergenzverhalten wird im folgenden Abschnitt analysiert. Hier sind lineare und nichtlineare Modelle getrennt zu betrachten. Die gewonnenen Aussagen werden an kleinen Testbeispielen verifiziert.

Der dritte Abschnitt dieses Kapitels beinhaltet Vergleichssimulationen zur bereits zitierten Dissertation [47]. Mit ihrer Hilfe soll auch die Anwendbarkeit des modifizierten Newton-Verfahrens auf unterschiedliche Probleme untersucht und die zu erwartende Ersparnis abgeschätzt werden.

Im letzten Abschnitt wird zunächst die prinzipielle Funktionsweise eines weiteren modifizierten Newton-Verfahrens, des so genannten Broyden-Verfahrens, vorgestellt. Mit dessen Hilfe soll die Rechenzeit für das Lösen des linearen Gleichungssystems mit direkten Verfahren deutlich reduziert werden. Auch hier erfolgen einige Testsimulationen zur Demonstration der Wirkungsweise des Verfahrens. Es soll auch eingeschätzt werden, ob das Broyden-Verfahren zur Lösung der vorliegenden Modellgleichungen sinnvoll eingesetzt werden kann.

3.1 Entkopplung im linearen Löser

Die Lösung des in Kapitel 2 aufgestellten Problems mit dem Newton-Verfahren führt auf ein lineares Gleichungssystem der Form

$$A\mathbf{x} \;=\; \mathbf{b}, \tag{3.1}$$

wobei $A \in \mathbb{R}^{n,n}$ die globale Jacobi-Matrix J, $\mathbf{b} \in \mathbb{R}^n$ die rechte Seite $-\mathbf{G}$ und $\mathbf{x} \in \mathbb{R}^n$ den Korrekturvektor $\boldsymbol{\delta}^{(k)}$ aus Abschnitt 2.2.2 bezeichnet.
Die Matrix A habe die in 2.2.3 angegebene Struktur mit maximal γ Nichtnulleinträgen.

3.1.1 Entkoppelte lineare Gleichungssysteme

Es wurde bereits in 2.2.3 beschrieben, dass die Jacobi-Matrix J dünnbesetzt ist und aus Blöcken von im Allgemeinen vollbesetzten Matrizen (■ in Abbildung 2.2) und Diagonalmatrizen (□) besteht, wobei die Nebendiagonalelemente der vollbesetzten Matrizen die Kopplung zwischen den einzelnen Spezies durch Reaktionen realisieren.
Es ist möglich, dass im Laufe der Simulation einzelne Reaktionen in einem Zeitintervall $\mathcal{J}' \subset \mathcal{J}$ nicht stattfinden bzw. vernachlässigbar sind, so dass die entsprechenden Nebendiagonalelemente in den vollbesetzten Matrixblöcken (näherungsweise) verschwinden. Durch diese Ausdünnung der Jacobi-Matrix zerfällt das lineare Gleichungssystem unter Umständen in mehrere voneinander unabhängige Teilsysteme, die getrennt voneinander gelöst werden können.

Notation 3.1 (Teilsysteme)**.** Das lineare Gleichungssystem $A\mathbf{x} = \mathbf{b}$ zerfalle in N_T Teilsysteme

$$A_i\mathbf{x}_i \;=\; \mathbf{b}_i, \tag{3.2}$$

mit $i \in \{1, \ldots, N_T\}$, $A_i \in \mathbb{R}^{n_i,n_i}$, $\mathbf{x}_i, \mathbf{b}_i \in \mathbb{R}^{n_i}$ und

$$\sum_{i=1}^{N_T} n_i \;=\; n.$$

In der Arbeit [47] wird die Lösung entkoppelter linearer Gleichungssysteme für ein reaktives Mehrkomponentenmodell im Eindimensionalen untersucht. Als linearer Löser für die einzelnen Teilsysteme dient ein Gaußalgorithmus mit Pivotsuche, der die Tridiagonalstruktur der Systemmatrix ausnützt. Es wird nachgewiesen, dass die N_T Teilsysteme in der Summe mit weniger Rechenaufwand gelöst werden können als das Gesamtsystem.
Im Zwei- und Dreidimensionalen ist die Anzahl der Knoten, die zur Simulation realistischer Mehrkomponentenmodelle benötigt wird, in der Regel jedoch deutlich höher als im Eindimensionalen. Darüber hinaus hat die Jacobi-Matrix keine

tridiagonale Struktur, so dass ihr Band wesentlich größer ist. Die hier entstehenden linearen Gleichungssysteme lassen sich deshalb oft mit iterativen Verfahren wie dem CG-Verfahren wesentlich effizienter berechnen als mit direkten.

3.1.2 Gewinn im linearen Löser

Im Folgenden soll untersucht werden, welches der beiden erwähnten Verfahren (iterativ und direkt) zur Lösung entkoppelter linearer Gleichungssysteme besser geeignet ist. Wir verzichten dabei auf eine ausführliche Darstellung der Verfahren und verweisen auf die zitierte Literatur.

Die iterativen Verfahren [29, 54]

Exemplarisch für die Klasse der iterativen Verfahren betrachten wir das CG-Verfahren. Die Ergebnisse sind repräsentativ für alle iterativen Verfahren.

Es gibt zwei Möglichkeiten, die bei dieser Verfahrensklasse zu einer Verringerung des Rechenaufwandes führen können:

1. Die Teilmatrizen beinhalten in der Summe weniger Nichtnullelemente als A, d.h.
$$\sum_{i=1}^{N_T} \gamma_i \;<\; \gamma \quad \text{für } N_T \geq 2.$$
Es fallen gerade die Einträge in A weg, die für die Kopplung der Teilsysteme verantwortlich sind.

2. Die Anzahl der zur Lösung von (3.2) benötigten Iterationsschritte ist eventuell kleiner als für das Gesamtsystem (3.1).

3. Parallelisierungsgewinn: Die Teilsysteme können bei Einsatz eines Parallelrechners gleichzeitig gelöst werden. Dieser Ansatz wird hier nicht weiter verfolgt.

Betrachten wir zunächst die erste Möglichkeit: Der Rechenaufwand des in [54] vorgestellten CG-Verfahrens ist pro Iterationsschritt

$$Z_{CG} \;=\; (\gamma + 5)\, n$$

und es ist für $N_T \geq 2$

$$\sum_{i=1}^{N_T} Z_{CG,i} \;=\; \sum_{i=1}^{N_T} (\gamma_i + 5)\, n_i \;<\; Z_{CG},$$

da für die Nichtnullelemente mit $\sum N_{S,i}^2 < N_S^2$ gilt:

$$\gamma_i \;=\; \sum_{i=1}^{N_T} N_{S,i}^2 \left(M + \sum_{K \in \mathcal{M}} d_{\mathcal{M}}(K) \right) \;<\; N_S^2 \left(M + \sum_{K \in \mathcal{M}} d_{\mathcal{M}}(K) \right) \;=\; \gamma.$$

Beispiel 3.2. Teilt man ein System mit zwölf Spezies in vier Teilsysteme mit je drei Spezies auf, ergibt sich eine potentielle Verringerung des Rechenaufwandes pro Iterationsschritt um 75 %.

Zur Untersuchung der zweiten Möglichkeit von oben (Reduzierung der Iterationsschritte) benötigen wir

Definition 3.3 (Konditionszahl und spektrale Konditionszahl)**.** *Die Zahl*

$$\kappa(A) \;:=\; \|A\| \; \left\|A^{-1}\right\|$$

heißt Konditionszahl *von A. Falls A symmetrisch ist und daher nur reelle Eigenwerte besitzt, bezeichnet*

$$\kappa(A) \;:=\; \frac{\lambda_{max}}{\lambda_{min}}$$

die spektrale Konditionszahl *von A. Dabei sind λ_{max} und λ_{min} der größte und kleinste Eigenwert von A. Bei nichtsymmetrischen Matrizen sind die Eigenwerte λ_{min} und λ_{max} durch die Singulärwerte σ_{min} und σ_{max} zu ersetzen:*

$$\kappa(A) \;:=\; \frac{\sigma_{max}}{\sigma_{min}}.$$

Mit Hilfe dieser Kenngröße wird in [37, Satz 5.16] die folgende Abschätzung bewiesen:

Satz 3.4. *Sei κ die spektrale Konditionszahl von A und es gelte $\kappa > 1$, dann:*

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}\right\|_A \;\leq\; 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \; \left\|\mathbf{x}^{(0)} - \mathbf{x}\right\|_A .$$

Eine Einsparung von Iterationsschritten ergibt sich nur, falls zur Lösung der Teilsysteme insgesamt weniger Iterationsschritte nötig sind als für das große Gleichungssystem. Über den Zusammenhang der jeweils benötigten Schritte gibt der folgende Satz Auskunft:

Satz 3.5. *Gegeben sei ein lineares Gleichungssystem $A\mathbf{x} = \mathbf{b}$, $A \in \mathbb{R}^{n,n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ mit der Matrix*

$$A \;=\; \begin{pmatrix} A_1 & & & 0 \\ & A_2 & & \\ & & \ddots & \\ 0 & & & A_{N_T} \end{pmatrix}, \quad A_i \in \mathbb{R}^{n_i, n_i}, \quad \sum_{i=1}^{N_T} n_i = n,$$

die in N_T unabhängige Teilmatrizen zerfällt. A besitze nur reelle Eigenwerte und es gelte $\kappa(A) > 1$, $\kappa(A_i) > 1\ \forall i$.
Sei Λ die Menge der Eigenwerte von A, Λ_i die Menge der Eigenwerte von A_i. Dann ist

$$\Lambda \;=\; \bigcup_{i=1}^{N_T} \Lambda_i. \tag{3.3}$$

Beweis. Wir betrachten zunächst den Fall $N_T = 2$ mit den Matrizen

$$\begin{aligned} \tilde{A}_1 &:= \begin{pmatrix} A_1 & \\ & I_1 \end{pmatrix}, \quad I_1 \in \mathbb{R}^{n_2,n_2} \text{ Einheitsmatrix,} \\ \tilde{A}_2 &:= \begin{pmatrix} I_2 & \\ & A_2 \end{pmatrix}, \quad I_2 \in \mathbb{R}^{n_1,n_1} \text{ Einheitsmatrix.} \end{aligned}$$

Es ist $A = \tilde{A}_1\tilde{A}_2$ und $\det \tilde{A}_i = \det A_i$ (rekursive Entwicklung der auftretenden Determinanten mit dem Laplaceschen Entwicklungssatz jeweils nach der letzten Zeile für A_1 bzw. der ersten Zeile für A_2) und aus der linearen Algebra folgt

$$\det A \;=\; \left(\det \tilde{A}_1\right)\left(\det \tilde{A}_2\right) \;=\; (\det A_1)\,(\det A_2)\,.$$

Die Matrix $A - \lambda I$ hat die gleiche Struktur wie A, so dass mit

$$\det(A - \lambda I) \;=\; (\det(A_1 - \lambda I_2))\,(\det(A_2 - \lambda I_1))$$

die Behauptung folgt.
Eine Erweiterung auf den Fall $N_T > 2$ ist leicht möglich. □

Folgerung 3.6. Eine Reduzierung der Iterationsschritte durch Entkopplung des linearen Gleichungssystems ist nur zu erzielen, falls eine der beiden Bedingungen erfüllt ist:

1. Sei A die originale Matrix und A_d die ausgedünnte Matrix, die aus A durch Streichen kleiner Einträge wie oben beschrieben entsteht. Das ausgedünnte Gleichungssystem benötigt zur Lösung weniger Schritte, falls

$$\kappa(A_d) \;\ll\; \kappa(A)\,.$$

2. Seien A_i die Teilmatrizen, in die A_d zerfällt. Das $i-$te Teilsystem benötigt weniger Schritte, falls

$$\kappa(A_i) \;\ll\; \kappa(A_d)\,, \quad i \in \{1,\ldots,N_T\}\,.$$

Die erste Bedingung würde einen Vorteil aus der Ausdünnung von A ziehen, die zweite aus der Aufteilung von A_d in kleinere Teilsysteme A_i.

Ob eine der beiden Bedingungen allgemein erfüllt ist, soll anhand des folgenden akademischen Beispiels überprüft werden. Es ist an eine Simulation aus [51] angelehnt und wird in ähnlicher Weise auch in der Dissertation [47] verwendet. Die Berechnungen wurden mit der zweidimensionalen Simulationssoftware *M++* durchgeführt.

Beispiel 3.7. Gegeben sei ein Modell mit 12 mobilen Spezies $A, \ldots, L$ auf dem Gebiet $\Omega := (0; 1\ m) \times (0; 1/2\ m)$. Die Anfangswerte von E und K sind $c_{0,EK} := 100\ mg/l$, für alle anderen Spezies gelten homogene Anfangswerte. A, B, G und H haben am linken Rand die Dirichlet-Randwerte $c_{l,ABGH} := 1\ mg/l$, auf allen anderen Randteilen gelten homogene Neumann-Randwerte. Der Diffusionskoeffizient ist für alle Spezies $D := 1 \cdot 10^{-4}\ m^2/s$, der konvektive Fluss $\mathbf{q} := (1/10; 0)^T$.

Die Péclet-Zahl lautet für die gewählten Parameter

$$Pe \ = \ 1000.$$

Es finden sechs Reaktionen statt:

$$\begin{array}{llll} R_1: & A + B & \longleftrightarrow & C, \\ R_2: & C & \longleftrightarrow & D + I, \\ R_3: & D + E & \longleftrightarrow & F, \end{array} \qquad \begin{array}{llll} R_4: & G + H & \longleftrightarrow & I, \\ R_5: & I & \longleftrightarrow & J, \\ R_6: & J + K & \longleftrightarrow & L. \end{array}$$

Die Spezies A, B, G und H fließen von links in das Gebiet und verlassen es am rechten Rand wieder. Dazwischen reagieren sie wie oben beschrieben.

In der zitierten Literatur entsteht dieses System, indem die Spezies $A, \ldots, F$ und die Reaktionen R_1, R_2, R_3 in die Spezies $G, \ldots, L$ und die Reaktionen R_4, R_5, R_6 kopiert werden. Die Symmetrie zwischen den beiden Teilsystemen wird einzig durch R_2 durchbrochen. Diese Reaktion sorgt für die Verbindung zwischen den beiden Hälften.

Die Speziesgruppen (ABC), (DEF), (GHI) und (JKL) werden nur durch die Reaktionen R_2 und R_5 gekoppelt. Falls diese nicht stattfinden, zerfällt das zu lösende lineare Gleichungssystem in vier Blöcke. Fällt nur R_2 weg, entstehen die unabhängigen Teilsysteme (ABC), (DEF) und $(G \ldots L)$ und ohne R_5 $(A \ldots I)$ und (JKL).

In den folgenden Testsimulationen werden 10 Zeitschritte mit $\Delta t = 1\ s$ berechnet. Das verwendete Gitter besitzt sofern nicht anders angegeben 297 Knoten, was einer räumlichen Schrittweite von $h = 1/32\ m$ entspricht. Das zu lösende Gleichungssystem hat somit 3564 Unbekannte und es ist

$$\begin{aligned} Pe_T &\approx 31, \\ Cr &\approx 3,2. \end{aligned}$$

Das gesamte lineare Gleichungssystem wird jeweils in vier Teilsysteme mit den Spezies (ABC), (DEF), (GHI) und (JKL) aufgeteilt. Die Damköhler-Zahlen lauten für die sechs Reaktionen $(\alpha, 1, \alpha, \alpha, 1, \alpha)$ mit $\alpha \in \mathbb{R}$, $\alpha \geq 0$. Die Variation der Da erfolgt durch Änderung der Ratenparameter k. Der Fluss $\mathbf{q}$ wird konstant gehalten. Als linearer Löser dient $BiCGStab$, als Vorkonditionierer ein Schritt mit dem symmetrischen Gauß-Seidel-Verfahren.

Tabelle 3.1: Konditionszahlen der einzelnen Matrizen für die Damköhlerzahlen $(\alpha, 1, \alpha, \alpha, 1, \alpha)$ und mittlere Anzahl der Iterationsschritte pro Newtonschritt von BiCGStab (IT/NS) ohne und mit Vorkonditionierer (symmetrisches Gauß-Seidel-Verfahren) zur Lösung des vollen Problems.

α	1	10	100	1000	10000	100000
$\kappa(A)$	$8,8 \cdot 10^2$	$1,0 \cdot 10^3$	$4,1 \cdot 10^3$	$3,3 \cdot 10^4$	$3,8 \cdot 10^5$	$3,1 \cdot 10^6$
$\kappa(A_d)$	$4,6 \cdot 10^2$	$6,7 \cdot 10^2$	$4,1 \cdot 10^3$	$3,3 \cdot 10^4$	$3,8 \cdot 10^5$	$3,1 \cdot 10^6$
$\kappa(A_1)$	$3,5 \cdot 10^2$	$1,7 \cdot 10^2$	$7,2 \cdot 10^1$	$7,1 \cdot 10^2$	$7,1 \cdot 10^3$	$7,1 \cdot 10^4$
$\kappa(A_2)$	$4,6 \cdot 10^2$	$6,7 \cdot 10^2$	$4,1 \cdot 10^3$	$3,3 \cdot 10^4$	$3,8 \cdot 10^5$	$3,1 \cdot 10^6$
$\kappa(A_3)$	$3,4 \cdot 10^2$	$1,7 \cdot 10^2$	$7,3 \cdot 10^1$	$7,2 \cdot 10^2$	$7,2 \cdot 10^3$	$7,2 \cdot 10^4$
$\kappa(A_4)$	$4,6 \cdot 10^2$	$6,7 \cdot 10^2$	$4,1 \cdot 10^3$	$3,3 \cdot 10^4$	$3,8 \cdot 10^5$	$3,1 \cdot 10^6$
IT/NS ohne Vorkond.	4,0	4,3	10,8	55,3	108,7	2223,2
IT/NS mit Vorkond.	3,0	3,0	3,0	3,0	3,0	3,0

Tabelle 3.1 zeigt die Abhängigkeit der Konditionszahlen der einzelnen (Teil-) Matrizen von den Damköhlerzahlen $(\alpha, 1, \alpha, \alpha, 1, \alpha)$ sowie die durchschnittliche Anzahl der Iterationsschritte des linearen Lösers $BiCGStab$ pro Newtonschritt (IT/NS) ohne und mit Vorkonditionierer.
Es wird deutlich, dass die erste Bedingung aus Folgerung 3.6 nicht erfüllt ist. Die Konditionszahlen der ausgedünnten Matrizen sind für alle betrachteten α (annähernd) so groß wie die der ursprünglichen Matrix A. Die zweite Bedingung ist zumindest für große α erfüllt. $\kappa(A_1)$ und $\kappa(A_3)$ sind jeweils um ca. zwei Größenordnungen kleiner als $\kappa(A)$, so dass diese beiden Teilsysteme nach Satz 3.4 mit weniger Iterationsschritten gelöst werden können. Für das zweite und vierte Teilsystem ist allerdings keine Reduzierung der Schrittzahl zu erwarten.

Die Anzahl der Iterationsschritte pro Newtonschritt steigt ohne Vorkonditionierer mit zunehmendem α deutlich an, mit Vorkonditionierer bleibt sie konstant. Für großes α entstehen zu den relativ langsamen Reaktionen mit $Da = 1$ sehr schnelle mit $Da = 100000$. Die unterschiedlichen Zeitskalen, auf denen diese Reaktionen stattfinden, sind für die hohen Konditionszahlen verantwortlich. Der Vorkonditionierer gleicht diese Unterschiede offensichtlich aus und hält die Anzahl der Iterationsschritte konstant.
Mittels Tabelle 3.2 soll die Abhängigkeit der Konditionszahlen und der Iterationsschritte von der räumlichen Schrittweite h untersucht werden. Die Zell-Péclet-

Tabelle 3.2: Konditionszahlen der einzelnen Matrizen für die Damköhlerzahlen $(1000, 1, 1000, 1000, 1, 1000)$ und mittlere Anzahl der Iterationsschritte pro Newtonschritt von BiCGStab (IT/NS) ohne und mit Vorkonditionierer (symmetrisches Gauß-Seidel-Verfahren) bei variierender räumlicher Schrittweite h.

h	$1/4\ m$	$1/8\ m$	$1/16\ m$	$1/32\ m$	$1/64\ m$	$1/128\ m$
$\kappa(A)$	$3,0 \cdot 10^2$	$3,4 \cdot 10^4$	$5,2 \cdot 10^4$	$3,3 \cdot 10^4$	$1,1 \cdot 10^5$	$1,1 \cdot 10^5$
$\kappa(A_d)$	$3,0 \cdot 10^2$	$2,5 \cdot 10^4$	$3,8 \cdot 10^4$	$3,3 \cdot 10^4$	$6,8 \cdot 10^4$	$5,4 \cdot 10^4$
$\kappa(A_1)$	$7,4 \cdot 10^1$	$1,9 \cdot 10^2$	$2,4 \cdot 10^2$	$7,1 \cdot 10^2$	$3,4 \cdot 10^3$	$1,1 \cdot 10^4$
$\kappa(A_2)$	$3,0 \cdot 10^3$	$6,4 \cdot 10^3$	$9,4 \cdot 10^3$	$3,3 \cdot 10^4$	$1,4 \cdot 10^4$	$1,4 \cdot 10^4$
$\kappa(A_3)$	$7,4 \cdot 10^1$	$1,9 \cdot 10^2$	$2,4 \cdot 10^2$	$7,2 \cdot 10^2$	$3,4 \cdot 10^3$	$1,1 \cdot 10^4$
$\kappa(A_4)$	$3,0 \cdot 10^3$	$6,4 \cdot 10^3$	$9,4 \cdot 10^3$	$3,3 \cdot 10^4$	$1,4 \cdot 10^4$	$1,4 \cdot 10^4$
IT/NS ohne Vorkond.	$41,3$	$44,0$	$44,2$	$55,3$	$64,7$	$99,1$
IT/NS mit Vorkond.	$1,8$	$2,0$	$2,6$	$3,0$	$3,3$	$4,2$

Zahlen Pe_T liegen für diese Schrittweiten zwischen $7,8$ und 250, die Courant-Zahlen Cr zwischen $0,4$ und $12,8$.
Die Ausdünnung der Jacobi-Matrix führt auch hier zu keiner nennenswerten Reduzierung der Konditionszahlen. Bei der Aufteilung in die vier Teilsysteme zeigt sich ein ähnliches Verhalten wie oben. Für zwei Teilsysteme reduziert sich κ deutlich, für die anderen beiden kaum.
Die Konditionszahl hängt mit der Ordnung $O(1/h^2)$ von h ab. κ steigt wie zu erwarten mit abnehmender Schrittweite an, was sich natürlich auch auf die Anzahl der Iterationsschritte pro Newtonschritt für den linearen Löser ohne Vorkonditionierer auswirkt. Das symmetrische Gauß-Seidel-Verfahren reduziert auch hier die Anzahl der benötigten Schritte, kann die Steigerung der Konditionszahlen jedoch nicht ausgleichen. Der Faktor, um den sich die Schrittzahl erhöht, bleibt etwa erhalten.

Zum Abschluss betrachten wir in Tabelle 3.3 die Variation der Zeitschrittweite. Die Courant-Zahlen Cr liegen für die verwendeten Δt zwischen $0,128$ und 128, Pe und Pe_T bleiben unverändert.
Hier steigen die Konditionszahlen mit zunehmendem Δt deutlich an. $\kappa(A_d)$ und $\kappa(A_i)$, $i = 1, \ldots, 4$ verhalten sich ähnlich wie in den vorangegangenen beiden Beispielen.
Der Vorkonditionierer ist anders als in Tabelle 3.1 offensichtlich nicht in der Lage, die Zunahme der κ zu kompensieren. Die Anzahl der Iterationsschritte pro Newtonschritt nimmt sowohl bei der Simulation ohne als auch mit Vorkonditionierung rapide zu, so dass bei genügend großen Δt unter Umständen ein direkter Löser kürzere Rechenzeiten aufweisen könnte als der iterative. Wir werden diese Möglichkeit später noch untersuchen.
Bei der Simulation ohne Vorkonditionierung wurde der lineare Löser nach 500 Iterationsschritten abgebrochen. Ein Weiterrechnen ist in diesem Fall wegen der

Tabelle 3.3: Konditionszahlen der einzelnen Matrizen für die Damköhlerzahlen $(1000, 1, 1000, 1000, 1, 1000)$ und mittlere Anzahl der Iterationsschritte pro Newtonschritt von BiCGStab (IT/NS) ohne und mit Vorkonditionierer (symmetrisches Gauß-Seidel-Verfahren) bei variierender Zeitschrittweite Δt.

Δt	$0,01\ s$	$0,1\ s$	$1,0\ s$	$10,0\ s$
$\kappa(A)$	$1,6 \cdot 10^1$	$1,6 \cdot 10^2$	$3,3 \cdot 10^4$	$3,9 \cdot 10^6$
$\kappa(A_d)$	$9,7 \cdot 10^0$	$1,5 \cdot 10^2$	$3,3 \cdot 10^4$	$3,9 \cdot 10^6$
$\kappa(A_1)$	$5,2 \cdot 10^0$	$5,3 \cdot 10^1$	$7,1 \cdot 10^3$	$4,6 \cdot 10^3$
$\kappa(A_2)$	$9,7 \cdot 10^0$	$1,5 \cdot 10^2$	$3,3 \cdot 10^4$	$3,9 \cdot 10^6$
$\kappa(A_3)$	$5,2 \cdot 10^0$	$5,3 \cdot 10^1$	$7,2 \cdot 10^3$	$4,9 \cdot 10^3$
$\kappa(A_4)$	$9,7 \cdot 10^0$	$1,5 \cdot 10^2$	$3,3 \cdot 10^4$	$3,9 \cdot 10^6$
IT/NS ohne Vorkond.	4	> 500	> 500	> 500
IT/NS mit Vorkond.	3	7	35	84

im Vergleich zum direkten Verfahren deutlich höheren Rechenzeiten nicht mehr sinnvoll.

Zusammenfassung 3.8. Zusammenfassend ist festzuhalten, dass unabhängig von den Damköhlerzahlen und den räumlichen und zeitlichen Schrittweiten im Allgemeinen eine Ausdünnung von A zu keiner nennenswerten Reduzierung der Konditionszahlen führt. Bei der Aufteilung können Teilsysteme mit wesentlich niedrigeren κ entstehen. Es kann aber auch vorkommen, dass einige oder alle aufgeteilten linearen Gleichungssysteme ähnlich schlecht konditioniert sind wie das ursprüngliche volle System. Ein garantierter Gewinn kann also lediglich aus der oben beschriebenen Verringerung der Rechenoperationen gezogen werden.

Die direkten Verfahren [54]

Wir betrachten das Gauß-Verfahren als typischen Vertreter dieser Klasse. Es benötigt zur Lösung des linearen Gleichungssystems (3.1) bei vollbesetzter Matrix

$$Z_{Gauß} \;=\; \frac{1}{3}n^3 + n^2 - \frac{1}{3}n$$

wesentliche Rechenoperationen (Multiplikationen + Divisionen). Für dünnbesetzte A hängt der Rechenaufwand von der Bandbreite m der Matrix ab. In [54] wird gezeigt, dass das Gauß-Verfahren in diesem Fall mit

$$O\left(m^2 n\right)$$

wesentlichen Rechenoperationen durchführbar ist.

Während des Verfahrens entsteht innerhalb des Bandes ein fill-in, der von quadratischer Ordnung in den Gesamtaufwand eingeht. Wir führen deshalb vor der

Lösung des linearen Gleichungssystems einen Cuthill-Mc-Kee-Algorithmus [17] durch, mit dessen Hilfe die Bandbreite der Matrix A durch Umnummerieren der Knoten reduziert wird.

Beispiel 3.9. Falls wie in Beispiel 3.2 ein System in vier gleichgroße Teilsysteme zerfällt, reduziert sich die Ordnung des Rechenaufwandes in der Summe zu

$$4O\left(\left(\frac{m}{4}\right)^2 \frac{n}{4}\right) \;=\; \frac{1}{16} O\left(m^2 n\right).$$

Der Aufwand verringert sich also um einen Faktor 16. Das Einsparpotential ist also um einen Faktor 4 größer als beim iterativen Verfahren.

3.1.3 Assemblierungsgewinn

Unabhängig vom gewählten linearen Löser entsteht durch die Aufteilung des Gleichungssystems in $N_T \geq 2$ Teilsysteme eine Verringerung der Assemblierungszeit. Dies kann aus zwei Gründen geschehen:

1. Die Elemente in A, die für die Kopplung der Teilsysteme verantwortlich sind, müssen nicht assembliert werden.

2. Im Allgemeinen benötigen die Teilsysteme im Newtonlöser unterschiedlich viele Schritte für ihre Lösung. Die Elemente der Teilsysteme, die das Abbruchkriterium des Newtonverfahrens bereits erfüllen, müssen nicht mehr assembliert werden.

Während der zweite Punkt sehr stark vom konkreten Beispiel abhängt, lässt sich die unter 1. beschriebene Einsparung an zu assemblierenden Einträgen in A abschätzen:
Für Friedrichs-Keller-Triangulierungen auf Rechtecksgebieten gilt für die Anzahl der zu assemblierenden Nichtnullelemente γ in der Jacobi-Matrix A

$$\gamma \;\leq\; MN_S\left(N_S + 6\right),$$

wobei M wieder die Anzahl der räumlichen Knoten bezeichnet.

Tabelle 3.4: Einsparung in der Assemblierung durch Aufteilung in N_T Teilsysteme für Beispiel 3.7 mit 12 Spezies.

N_T	γ	Einsparung
1	$216M$	–
4	$4 \cdot 27M = 108M$	50 %
12	$12 \cdot 7M = 84M$	61 %

Tabelle 3.4 zeigt die Einsparung, die sich allein durch Nichtassemblieren der Kopplungselemente ergibt. Diese bezieht sich natürlich nur auf die Anzahl der zu berechnenden Einträge γ, nicht auf die benötigte Rechenzeit.

Zahlreiche Testsimulationen haben gezeigt, dass der Assemblierungsgewinn bei dieser Ausdünnung von A sehr gering ist. Es fallen aus der vollen Jacobi-Matrix nur Terme der Form $\partial R_i/\partial c_j$ für $i \neq j$ (außerhalb der Hauptdiagonalen) der vollbesetzten Blockmatrizen weg.
In der Implementierung in *M++* werden die Ableitungen der Reaktionsterme mittels einer Schleife über alle Reaktionen berechnet, innerhalb der sich noch sechs Doppelschleifen über alle Spezies befinden. Das Schema des zugehörigen Programmausschnitts ist in Abbildung 3.1 aufgelistet.

```
for (i=0; i<NR; i++)
{
  for(j=0; j<NS; j++)
  {
    for(k=0; k<NS; k++)
    {
      ...
    }
    ...
  }
  ...
}
```

Abbildung 3.1: Programmausschnitt aus der *M++*-Implementierung zur Berechnung der Ableitungen der Reaktionsterme.

Falls nur die Kopplungsterme ignoriert werden sollen, muss in jeder Doppelschleife für alle j und k abgefragt werden, ob dieser Eintrag assembliert werden soll. Das führt zu $6N_R N_S^2$ Abfragen, deren Aufwand einen Teil des Gewinns, der durch das Nichtassemblieren der Kopplungsterme entsteht, verbraucht.

Dieses Problem kann behoben werden, indem auch die jeweiligen Anteile der Reaktionsterme auf der Hauptdiagonalen ($\partial R_i/\partial c_i$) nicht assembliert werden. Dann sind in der Implementierung aus Abbildung 3.1 nur noch N_R Abfragen nötig und der Gewinn fällt entsprechen höher aus.
Diese zusätzliche Ausdünnung der Jacobi-Matrix kann allerdings zu Konvergenzproblemen im Newton-Löser führen. Solange nur schwache Reaktionen ignoriert werden, scheint diese Gefahr aber relativ gering.

3.1.4 Der modifizierte Newtonlöser *NewtonBlock*

Die Lösung des nichtlinearen Gleichungssystems (2.10), (2.11) und (2.12) mit dem modifizierten Newton-Verfahren unter manueller Aufteilung des linearen Gleichungssystems (2.13) in die Teilsysteme (3.2) wird vom neu entwickelten Newtonlöser *NewtonBlock* durchgeführt. Abbildung 3.2 zeigt das Struktogramm dieses Lösers.

Zu Beginn werden einige Variablen definiert, die Teilsysteme manuell festgelegt und die benötigten Speicherstrukturen erzeugt. Es folgt die Assemblierung des Defekts $\mathbf{b}$ und das Kopieren von $\mathbf{b}$ und $\mathbf{x}$ in die Teilvektoren $\mathbf{bT}[i]$ und $\mathbf{xT}[i]$. Mit Hilfe der $\mathbf{bT}[i]$ ermittelt der Algorithmus die Teilsysteme, die das Abbruchkriterium (2.14) bereits erfüllen.
Die Iteration des Newton-Lösers wird durch eine while-Schleife realisiert, die zu durchlaufen ist, solange eine vorgegebene Anzahl von Iterationsschritten nicht überschritten wurde ($k < MaxIter$) und noch nicht alle Teilsysteme das Abbruchkriterium (2.14) erfüllen ($nready < NT$). In der Iterationsschleife wird die Liste der zu assemblierenden Reaktionen ermittelt, die benötigten Elemente der Matrix assembliert und A in die Teilmatrizen $AT[i]$ kopiert. Das Kopieren erfolgt durch die Funktion *BuildSubSystems*, die in A.4 aufgelistet ist und näher beschrieben wird.

Die Vorgehensweise des Kopierens hat im Vergleich zur Verwendung eines linearen Blocklösers, der direkt auf der Gesamtmatrix A operiert, einige Vorteile:

- Die von *M++* bereitgestellten linearen Löser und Vorkonditionierer können problemlos aufgerufen werden.
- Neue Löser und Vorkonditionierer können ohne Weiteres implementiert werden.
- Es ist kein zeitintensives Neuassemblieren der Jacobi-Matrix nötig.
- Der Aufwand für das Kopieren hängt von γ (bei Matrizen) bzw. von n (bei Vektoren) ab. Er ist unabhängig vom Rechenaufwand des linearen Lösers, also von der Anzahl der Iterationsschritte.

Besonders der letzte Punkt ist für große lineare Gleichungssysteme von entscheidender Bedeutung. Für allzu kleine n kann er sich jedoch negativ auswirken, da hier das Kopieren eventuell mehr Zeit benötigt als der gesamte lineare Löser. Wir gehen davon aus, dass n groß genug und die Kopierzeit klein im Vergleich zur Laufzeit des linearen Lösers ist.

Nach diesen Vorbereitungen können nun die Teilsysteme $AT_i\ \mathbf{x}T_i = \mathbf{b}T_i$, die das Abbruchkriterium (2.14) noch nicht erfüllen ($ready[i] == -1$), gelöst, die Teillösungen in $\mathbf{c}$ kopiert und die neue Näherungslösung $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{c}$ berechnet

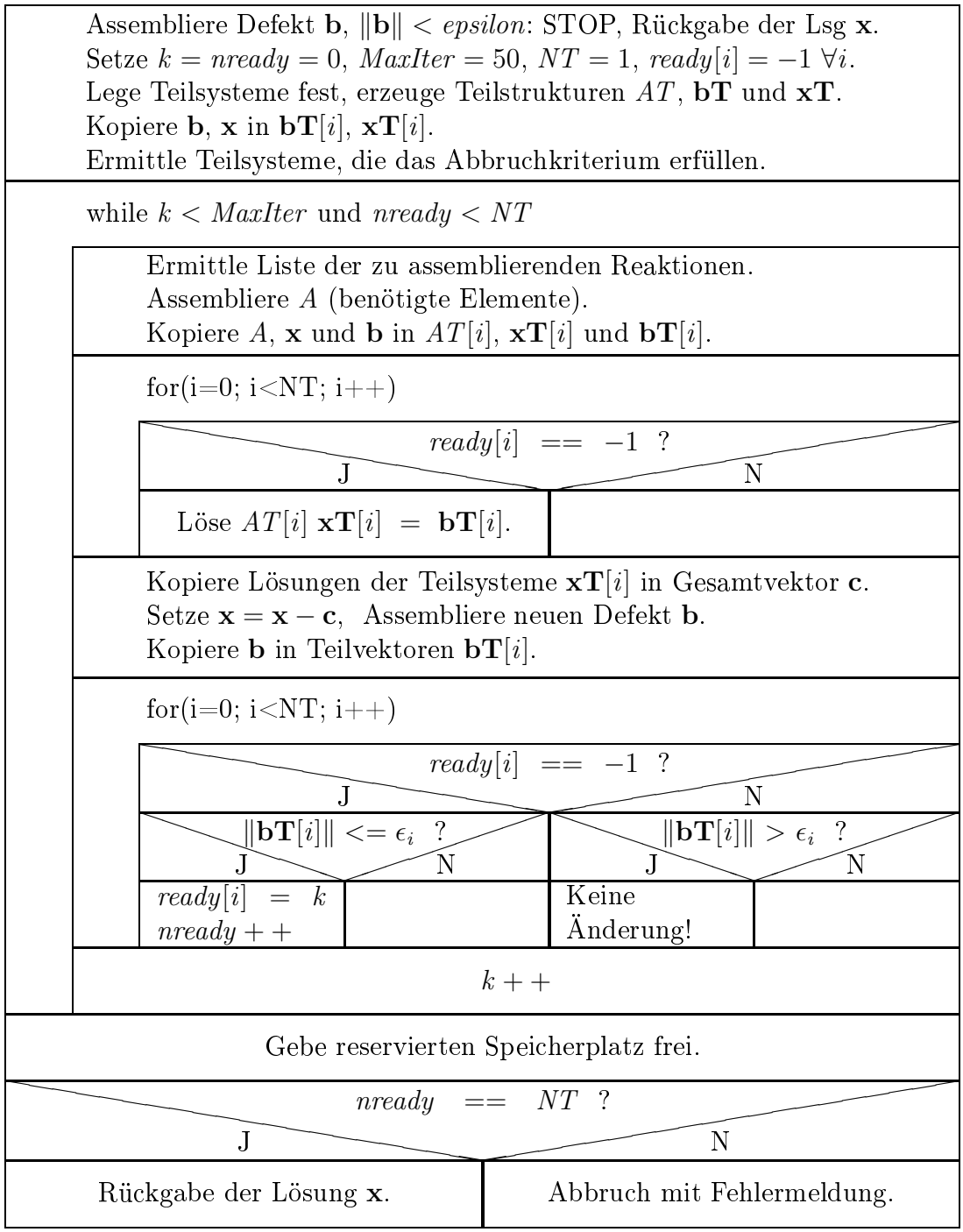

Abbildung 3.2: Struktogramm des neu entwickelten Newton-Lösers *NewtonBlock* mit manueller Festlegung der Teilsysteme.

werden. Es folgt die Neuassemblierung des Defektes $\mathbf{b}$ und das Kopieren von $\mathbf{b}$ in die Teilvektoren $\mathbf{bT}_i$.
Am Ende der Iterationsschleife ist noch zu prüfen, ob die gelösten Teilsysteme das Abbruchkriterium (2.14) erfüllen. Diese Information wird in $ready[i]$ eingetragen und $nready$ entsprechend erhöht.
Durch die Veränderung der Näherungslösung $\mathbf{x}$ ist es möglich, dass Teilsysteme, die das Abbruchkriterium bereits in einem vorhergehenden Iterationsschritt erfüllten, dieses jetzt wieder verletzen. Für sie müsste $ready[i]$ eigentlich wieder auf -1 gesetzt und der Zähler $nready$ entsprechend dekrementiert werden. Wenn wir jedoch davon ausgehen, dass nur schwache Kopplungen aufgetrennt werden, kann zu Gunsten eines größeren Assemblierungsgewinns auf diese Maßnahme verzichtet werden.

Nach Beendigung der Schleife werden reservierte Speicherstrukturen freigegeben und die gefundene Näherungslösung zurückgegeben. Falls die Schleife aufgrund des Erreichens der maximalen Iterationszahl $MaxIter$ abgebrochen wurde, endet das gesamte Programm mit einer entsprechenden Fehlermeldung.

Bemerkung 3.10. Das verwendete Abbruchkriterium des modifizierten Newtonverfahrens lautet für das $i-$te Teilsystem

$$\|\mathbf{b}_i\|_2 \;\leq\; \epsilon_{abs}\frac{n_i}{N_S} \;=:\; \epsilon_i,$$

während beim vollen Verfahren

$$\|\mathbf{b}\|_2 \;\leq\; \epsilon_{abs}$$

verwendet wurde. Für $N_T > 1$ gilt im Allgemeinen

$$\sum_{i=1}^{N_T} \|\mathbf{b}_i\|_2 \;\neq\; \|\mathbf{b}\|_2 .$$

Die Abbruchkriterien des vollen und des modifizierten Newton-Verfahrens sind also nicht identisch. Es ist jedoch in jedem Fall

$$\sum_{i=1}^{N_T} \|\mathbf{b}_i\|_2 \;\leq\; \sum_{i=1}^{N_T} \epsilon_{abs}\frac{n_i}{N_S} \;=\; \epsilon_{abs},$$

so dass der modifizierte Newton-Löser das Abbruchkriterium des vollen Verfahrens erfüllt.

In der Dissertation [47] wird im eindimensionalen Fall die Jacobi-Matrix nicht in Teilmatrizen kopiert. Statt dessen findet ein Listenlöser Anwendung (direktes Verfahren), der auf der vollen Matrix operiert und in jedem Newtonschritt alle Teilsysteme abarbeitet. Zwar sind dann die Abbruchkriterien des vollen und aufgeteilten Systems identisch, im Zweidimensionalen hat diese Vorgehensweise jedoch zwei Nachteile:

1. Der organisatorische Aufwand für das hin- und herspringen in der Matrix steigt mit dem Rechenaufwand des linearen Lösers an, der bei direkten Verfahren bekanntlich unter anderem von der Bandbreite m abhängt. Diese ist in $1D$ klein, wodurch der Einsatz eines Listenlösers gerechtfertigt ist. In $2D$ kann die Bandbreite allerdings sehr groß sein, so dass die oben beschriebene Unabhängigkeit des Kopieraufwandes von m für einen bedeutenden Zeitvorteil der in dieser Arbeit favorisierten Vorgehensweise sorgt.

2. Bei Verwendung eines Listenlösers mit Beibehaltung der Abbruchkriterien kann nur ein geringer Assemblierungsgewinn erzielt werden, da in jedem Newtonschritt alle Teilsysteme neu assembliert werden müssen. Der oben erwähnte Gewinn durch Nichtassemblieren bereits „fertiger“ Teilsysteme kann dann nicht realisiert werden.

Wir nehmen deshalb zu Gunsten des höheren Rechenzeitgewinnes den Unterschied in den Abbruchkriterien in Kauf.

Ein Listing des modifizierten Newtonlösers *NewtonBlock* befindet sich mit einigen zusätzlichen Erklärungen in A.5.

3.2 Konvergenzverhalten

In diesem Abschnitt gehen wir weiterhin davon aus, dass pro Zeitschritt ein Newton-Verfahren der Form (2.13) durchzuführen ist. Das in 3.1 vorgestellte Entkopplungsverfahren im linearen Löser führt zu einem modifizierten Newton-Verfahren:

1. Löse das lineare Gleichungssystem

$$J_d^{(k)}\ \boldsymbol{\delta}^{(k)} \;=\; -\mathbf{G}\left(\mathbf{x}^{(k)}\right). \tag{3.4}$$

2. Setze $\mathbf{x}^{(k+1)} \;=\; \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}$.

Im Vergleich zum vollen Verfahren (2.13) wurde die Jacobi-Matrix $J^{(k)}$ durch die ausgedünnte (modifizierte) Matrix $J_d^{(k)}$ ersetzt.

3.2.1 Der lineare Fall

Ein lineares Modell entsteht, wenn in den Modellgleichungen (2.1) die Koeffizientenfunktionen Θ, D und $\mathbf{q}$ als unabhängig von c und die Reaktionsterme R als linear in c angenommen werden. Es dürfen insbesondere nur Abbaureaktionen 0. und 1. Ordnung (2.7) oder stöchiometrische kinetische Reaktionen (2.8) der Form

$$A \quad \longleftrightarrow \quad B$$

stattfinden.

Für diese Modelle ist $\mathbf{G}(\mathbf{x}) = J(\mathbf{x}) - \mathbf{b}$ und es folgt aus (3.4)

$$\boldsymbol{\delta}^{(k)} \;=\; -J_d^{-1} J \mathbf{x}^{(k)} + J_d^{-1}\mathbf{b}.$$

mit der ausgedünnten Jacobi-Matrix J_d.
Eingesetzt in die untere Gleichung von (3.4) ergibt sich die explizite Vorschrift zur Berechnung der neuen Iterierten

$$\mathbf{x}^{(k+1)} \;=\; \left[I - J_d^{-1} J\right] \mathbf{x}^{(k)} + J_d^{-1}\mathbf{b}.$$

Nach [37, Satz 5.1] konvergieren Verfahren dieser Form global und linear, falls für den Spektralradius $\rho(.)$ der Iterationsmatrix

$$\rho\left(I - J_d^{-1} J\right) \;<\; 1 \tag{3.5}$$

gilt für alle Iterationsschritte k. ρ bezeichnet hier den Maximalbetrag der Singulärwerte der betreffenden Matrix.

Eine hinreichende Bedingung zur Erfüllung von (3.5) ist für eine Matrixnorm $\|.\|$

$$\left\|I - J_d^{-1} J\right\| \;<\; 1$$

und ein hinreichender Ausdruck mit $J_R^{(k)} := J - J_d$

$$\left\|J_d^{-1}\right\| \;\|J_R\| \;<\; 1.$$

Mit der in [37, Glng (5.8)] aufgestellten Konvergenzbedingung gilt für das modifizierte Newton-Verfahren (3.4) im linearen Fall die Konvergenzabschätzung

$$\left\|\mathbf{x}^{(k+1)} - \mathbf{x}\right\| \;\leq\; \left\|J_d^{-1}\right\| \;\|J_R\| \;\left\|\mathbf{x}^{(k)} - \mathbf{x}\right\|.$$

In der Spektralnorm $\|.\|_2$ für Matrizen bzw. der induzierenden euklidischen Norm für Vektoren folgt mit

$$C_{lin} \;:=\; \frac{\sigma_{max}(J_R)}{\sigma_{min}(J_d)} \tag{3.6}$$

die Abschätzung

$$\left\|\mathbf{x}^{(k+1)} - \mathbf{x}\right\|_2 \;\leq\; C_{lin} \left\|\mathbf{x}^{(k)} - \mathbf{x}\right\|_2, \tag{3.7}$$

wobei σ_{max} und σ_{min} den größten bzw. kleinsten Singulärwert der betreffenden Matrizen bezeichnet.

Im eingangs beschriebenen Spezialfall sind $J(\mathbf{x})$ und $J_d(\mathbf{x})$ unabhängig von $\mathbf{x}$. Somit sind $J^{(k)}$ und $J_d^{(k)}$ für alle Iterationsschritte gleich.
Für $J = J_d$ hätten wir in diesem Spezialfall Konvergenz in einem Schritt. Allgemein kann man erwarten, dass $C_{lin}^{(k)} < 1$ eine im Wesentlichen notwendige Bedingung für Konvergenz ist.

3.2.2 Beispiele für den linearen Fall

Zur Überprüfung der Abschätzung (3.7) betrachten wir

Beispiel 3.11. Gegeben sei ein Modell mit zwölf mobilen Spezies $A, \ldots, L$ auf dem Gebiet $\Omega := (0; 1\ m) \times (0; 1/2\ m)$. Für alle Spezies gelten homogene Anfangswerte. $A, \ldots, F$ haben am linken Rand die Dirichlet-Randwerte $c_l := 1\, mg/l$, auf allen anderen Randteilen gelten homogene Neumann-Randwerte. Der Diffusionskoeffizient ist für alle Spezies $D := 1 \cdot 10^{-4}\ m^2/s$, der konvektive Fluss $\mathbf{q} := (1/10; 0)^T$. Es finden sechs Abbaureaktionen statt:

$$\begin{array}{llll} R_1: & A & \longleftrightarrow & G, \\ R_2: & B & \longleftrightarrow & H, \\ R_3: & C & \longleftrightarrow & I, \end{array} \qquad \begin{array}{llll} R_4: & D & \longleftrightarrow & J, \\ R_5: & E & \longleftrightarrow & K, \\ R_6: & F & \longleftrightarrow & L. \end{array}$$

Es werden die Schrittweiten $\Delta t = 1\ s$ und $h = 1/32\ m$ verwendet. Die Courant- und (Zell-)Péclet-Zahl sind aus Beispiel 3.7 zu entnehmen.

Die Simulation wird durchgeführt mit den Damköhler-Zahlen

$$\begin{array}{lcl} Da_1 & := & (1, 1, 1, 1, 1, 1) \quad \text{und} \\ Da_2 & := & (1000, 1, 1000, 1000, 1, 1000)\,. \end{array}$$

Das Gesamtsystem $(A \ldots L)$ wird nach den beiden Partitionierungen

$$\begin{array}{lcl} P_1 & := & (AG)\,(B)\,(H)\,(CI)\,(DJ)\,(E)\,(K)\,(FL) \quad \text{und} \\ P_2 & := & (A)\ldots(L) \end{array}$$

aufgeteilt. P_1 ignoriert die relativ schwachen Reaktionen R_2 und R_5 aus Da_2, P_2 vernachlässigt alle Reaktionen und führt somit zu einem voll entkoppelten Verfahren.

Tabelle 3.5 zeigt die Singulärwerte $\sigma_{max}\,(J_R)$ und $\sigma_{min}\,(J_d)$ sowie den Quotienten C_{lin} aus (3.6) für die unterschiedlichen Damköhler-Zahlen Da_1 und Da_2 und die Blockbildungen P_1 und P_2.

Tabelle 3.5: Singulärwerte und Quotienten C_{lin} aus Abschätzung (3.7) bei verschiedenen Damköhlerzahlen und Blockbildungen für das lineare Beispiel 3.11.

Da	Blockbildung	$\sigma_{max}\,(J_R)$	$\sigma_{min}\,(J_d)$	C_{lin}
Da_1	P_1	$1,49 \cdot 10^{-10}$	$1,55 \cdot 10^{-6}$	$9,55 \cdot 10^{-5}$
Da_1	P_2	$1,49 \cdot 10^{-10}$	$1,54 \cdot 10^{-6}$	$9,68 \cdot 10^{-5}$
Da_2	P_1	$1,49 \cdot 10^{-10}$	$1,55 \cdot 10^{-6}$	$9,55 \cdot 10^{-5}$
Da_2	P_2	$3,82 \cdot 10^{-2}$	$9,98 \cdot 10^{-4}$	$38,3$

Bei den ersten drei Simulationen bleiben die Singulärwerte und C_{lin} im Rahmen der Rechengenauigkeit konstant. Hier werden nur schwache Reaktionen mit $D_a = 1$ vernachlässigt. Offensichtlich spielt es für die Konvergenz des Newtonverfahrens, die durch C_{lin} bestimmt wird, keine Rolle, wie viele solcher Reaktionen ignoriert werden.
Die Konvergenz hängt aber natürlich von der Stärke der aufgetrennten Kopplungen ab, wie die vierte Simulation aus Tabelle 3.5 zeigt. Hier wurden die sechs Reaktionen mit den Damköhler-Zahlen Da_2 vollständig vernachlässigt, also auch die starken Reaktionen mit $Da = 1000$. Der Quotient C_{lin} nimmt den sehr großen Wert $C_{lin} = 38,3$ an und das Verfahren konvergiert nicht mehr.

3.2.3 Der nichtlineare Fall

Konvergenzabschätzungen wie in Abschnitt 3.2.1 für den linearen Fall sind im Nichtlinearen weitaus schwerer zu finden. Allerdings können häufig Voraussetzungen gefunden werden, unter denen das Verfahren überhaupt konvergiert.
Theorem 2.6 aus [20] beinhaltet eine Konvergenzaussage für ein modifiziertes Newton-Verfahren der Form (3.4). Wir zitieren im Folgenden diesen Satz und bringen die in der vorliegenden Arbeit üblichen Bezeichnungen ein:

Satz 3.12. *Sei $\mathbf{G} : \mathcal{D} \to \mathbb{R}^n$ eine stetig-differenzierbare Abbildung mit $\mathcal{D} \subset \mathbb{R}^n$ offen und konvex. Sei J_d eine Approximation an $D\mathbf{G}$. Es existiere ein Startwert $\mathbf{x}^{(0)} \in \mathcal{D}$ mit $J_d\left(\mathbf{x}^{(0)}\right)$ invertierbar und Konstanten α, $\bar{\omega}_0$, δ_0, δ_1, $\delta_2 \geq 0$, so dass für alle $\mathbf{x}, \mathbf{y} \in \mathcal{D}$ folgende Bedingungen erfüllt sind:*

$$\begin{aligned}
\left\| J_d^{-1}\left(\mathbf{x}^{(0)}\right) \mathbf{G}\left(\mathbf{x}^{(0)}\right) \right\| &\leq \alpha, \\
\left\| J_d^{-1}\left(\mathbf{x}^{(0)}\right) \left(J\left(\mathbf{y}\right) - J\left(\mathbf{x}\right)\right) \right\| &\leq \bar{\omega}_0 \left\| \mathbf{y} - \mathbf{x} \right\|, \\
\left\| J_d^{-1}\left(\mathbf{x}^{(0)}\right) J_R\left(\mathbf{x}\right) \right\| &\leq \delta_0 + \delta_1 \left\| \mathbf{x} - \mathbf{x}^{(0)} \right\|, \\
\left\| J_d^{-1}\left(\mathbf{x}^{(0)}\right) \left(J_d\left(\mathbf{x}\right) - J_d\left(\mathbf{x}^{(0)}\right)\right) \right\| &\leq \delta_2 \left\| \mathbf{x} - \mathbf{x}^{(0)} \right\|,
\end{aligned}$$

$$\delta_0 < 1,\ \sigma := \max\left(\bar{\omega}_0, \delta_1 + \delta_2\right),\ h := \frac{2\alpha\sigma}{\left(1-\delta_0\right)^2} \leq 1,$$

$$\bar{\mathcal{S}}\left(\mathbf{x}^{(0)}, \rho\right) \subset \mathcal{D} \text{ mit } \rho := \frac{2\alpha}{1-\delta_0} / \left(1 + \sqrt{1-h}\right).$$

Dann ist die durch das modifizierte Newton-Verfahren (3.4) erzeugte Folge $\left\{\mathbf{x}^{(k)}\right\}$ wohldefiniert, verbleibt in $\bar{\mathcal{S}}\left(\mathbf{x}^{(0)}, \rho\right)$ und konvergiert gegen eine Lösung $\mathbf{x}^*$ mit $\mathbf{G}\left(\mathbf{x}^*\right) = 0$. Mit den Bezeichnungen

$$\bar{h} := \frac{\bar{\omega}_0}{\sigma} h,\ \rho_{\pm} = \frac{2\alpha}{1-\delta_0} / \left(1 \mp \sqrt{1-\bar{h}}\right)$$

ist die Lösung $\mathbf{x}^* \in \bar{\mathcal{S}}\left(\mathbf{x}^0, \rho_-\right)$ eindeutig in

$$\bar{\mathcal{S}}\left(\mathbf{x}^{(0)}, \rho\right) \cup \left(\mathcal{D} \cap \mathcal{S}\left(\mathbf{x}^{(0)}, \rho_+\right)\right).$$

Beweis. Siehe [20]. □

Die ersten beiden Bedingungen aus Satz 3.12 sind in ähnlicher Weise auch beim vollen Newton-Verfahren zu fordern. Die Ungleichungen wurden auf der linken Seite lediglich mit einem Gewichtungsfaktor

$$\left\| J_d^{-1}\left(\mathbf{x}^{(0)}\right)\right\|$$

multipliziert. Wir gehen davon aus, dass diese erweiterten Bedingungen bei maßvoller Ausdünnung der Jacobi-Matrix und der damit verbundenen Aufteilung des linearen Gleichungssystems in Teilsysteme weiterhin erfüllt sind.

Für die Beurteilung der Konvergenz des modifizierten Newton-Verfahrens ist die dritte Bedingung von entscheidender Bedeutung. Sie ist für das volle Verfahren in jedem Fall erfüllt, da wegen $J_R(\mathbf{x}) = 0\ \forall \mathbf{x} \in \mathcal{D}$

$$\left\| J_d^{-1}\left(\mathbf{x}^{(0)}\right) J_R(\mathbf{x})\right\| \ = \ 0 \ \leq \ \delta_0 + \delta_1 \left\|\mathbf{x} - \mathbf{x}^{(0)}\right\| \quad \forall\, \mathbf{x}, \mathbf{x}^{(0)} \in \mathcal{D},\ \delta_0, \delta_1 \geq 0$$

gilt. Hier können δ_0 und δ_1 unter Beachtung der weiteren Bedingungen so gesetzt werden, dass ρ maximal wird. Dadurch erreicht der Konvergenzbereich

$$\bar{\mathcal{S}}\left(\mathbf{x}^{(0)}, \rho\right)$$

die größtmögliche Ausdehnung und das Konvergenzverhalten wird optimal.

Aus den gleichen Gründen kann das Konvergenzverhalten des modifizierten Newton-Verfahrens verbessert werden, indem im ersten Newtonschritt das volle lineare Gleichungssystem gelöst wird und die Ausdünnung erst ab dem zweiten Iterationsschritt erfolgt.
Dieser Vorgehensweise steht allerdings entgegen, dass zur Lösung des nichtlinearen Gleichungssystems aus dem reaktiven Stofftransportmodell typischerweise nur wenige Newtonschritte erforderlich sind. Ein Verzichten auf die Aufteilung im ersten Schritt würde deshalb die Performance des Verfahrens zu sehr verschlechtern und sollte nur bei Auftreten von Konvergenzproblemen in Erwägung gezogen werden.

Die vierte Bedingung aus Satz 3.12 ist wieder weniger kritisch zu sehen. Wir setzen voraus, dass sie bei nicht zu starker Ausdünnung erfüllt ist mit einem δ_2, das den Konvergenzbereich des modifizierten Verfahrens nicht zu sehr verkleinert.

3.2.4 Beispiel für den nichtlinearen Fall

Anders als im linearen Fall fehlt im nichtlinearen eine Konvergenz-Abschätzung der Form (3.7). Es liegt aber nahe, die Kontraktionszahl C_{lin} aus (3.6) auch im Nichtlinearen als Indikator für den Grad der Ausdünnung zu verwenden. In

diesem Abschnitt soll nun anhand eines Testbeispiels demonstriert werden, dass diese Vorgehensweise nicht zulässig ist:

Wir berechnen wieder die bereits in Beispiel 3.7 (Seite 22) beschriebene Testsimulation mit zwölf Spezies und sechs Reaktionen. Für Plots der Ergebnisse verweisen wir auf die Dissertation [47].

Die Simulation wird durchgeführt mit den Damköhler-Zahlen Da_1 und Da_2 aus Abschnitt 3.2.2. Das Gesamtsystem wird nach der bereits bekannten Partitionierung P_2 und

$$P_3 \;:=\; (ABC)\ldots(JKL)$$

aufgeteilt, so dass bei Verwendung von Da_2 und P_3 nur die langsamen Reaktionen R_2 und R_5 ignoriert werden (siehe Beschreibung des Beispiels 3.7).

Nach Tabelle 3.6 verhalten sich die Singulärwerte $\sigma_{max}(J_R)$ und $\sigma_{min}(J_d)$ unter Verwendung der Damköhler-Zahlen Da_2 ähnlich wie im Linearen. Solange nur die schwachen Reaktionen R_2 und R_5 ignoriert werden (P_3) ist der Wert von C_{lin} sehr klein. Vernachlässigt man aber auch die starken Reaktionen R_1, R_3, R_4 und R_6, wird C_{lin} mit 195 deutlich größer und das modifizierte Newton-Verfahren konvergiert nicht mehr.

Tabelle 3.6: Singulärwerte und Quotienten C_{lin} aus Abschätzung (3.7) für verschiedene Damköhlerzahlen und Blockbildungen für das nichtlineare Beispiel 3.7.

Da	Blockbildung	$\sigma_{max}(J_R)$	$\sigma_{min}(J_d)$	C_{lin}
Da_2	P_3	$5,60 \cdot 10^{-10}$	$1,25 \cdot 10^{-4}$	$4,48 \cdot 10^{-6}$
Da_2	P_2	$3,00 \cdot 10^{-4}$	$1,54 \cdot 10^{-6}$	$1,95 \cdot 10^{+2}$
Da_1	P_3	$3,92 \cdot 10^{-10}$	$1,25 \cdot 10^{-6}$	$3,14 \cdot 10^{-4}$
Da_1	P_2	$2,98 \cdot 10^{-6}$	$1,54 \cdot 10^{-6}$	$1,94$

Für Da_1 stellt sich die Situation grundlegend anders dar. Während bei der Partitionierung P_3 in vier Teilsysteme die Kontraktionszahl noch einen plausiblen Wert annimmt, ist C_{lin} bei der Aufteilung P_2 in zwölf Teilsysteme mit $C_{lin} = 1,94$ unerwartet hoch. Schließlich werden auch hier nur langsame Reaktionen vernachlässigt. Die Ergebnisse aus Abschnitt 3.2.2 hätten einen ähnlichen Wert wie bei P_3 Nahe gelegt (vergleiche Tabelle 3.5).

Wir werden im nächsten Abschnitt sehen, dass das modifizierte Newton-Verfahren trotzdem konvergiert und die Anzahl der benötigten Newtonschritte nur geringfügig ansteigt.

Folgerung 3.13. Aus diesen Beobachtungen folgt, dass die für den linearen Fall hergeleitete Kontraktionszahl C_{lin} aus (3.6) im Allgemeinen nicht auf ein nichtlineares Problem übertragen werden kann.

3.3 Vergleichssimulationen zu [47]

Nach der Beschreibung des modifizierten Newton-Verfahrens und den Erläuterungen zur Implementierung und Konvergenz des neu entwickelten Block-Newtonlösers erfolgen in diesem Abschnitt einige Beispielsimulationen:

Wie bereits erwähnt wurde ein ähnliches Entkopplungsverfahren im Eindimensionalen in der Dissertation [47] entwickelt und mit Beispiel 3.7 getestet. In den nächsten beiden Abschnitten soll das Beispiel zunächst mit der ursprünglichen Zeitschrittweite aus [47] und dann mit vergrößertem Δt berechnet werden.

3.3.1 Ursprüngliche Schrittweite

In den folgenden Beispielen wird eine Zeitschrittweite von $\Delta t = 0,01\ s$ im Zeitintervall $\mathcal{J} = (0, 30\ s)$ (3000 Zeitschritte) verwendet. Das Gitter besitzt 4257 Knoten (51084 Unbekannte) und es ist $h = 1/128\ m$. Diese Schrittweiten führen zu den Kennzahlen

$$\begin{aligned} Pe &= 1000, \\ Pe_T &= 7,8 \quad \text{und} \\ Cr &= 0,128. \end{aligned}$$

Die relativ kleine Courant-Zahl indiziert den Einsatz eines expliziten Verfahrens. Allerdings wurde im Referenzbeispiel aus [47] ein implizites verwendet, so dass wir an dieser Stelle ebenfalls unser implizites Verfahren anwenden.

Die Anzahl der Nichtnullelemente und deren Anteil an den Gesamteinträgen in J bzw. J_d ist für verschiedene Konfigurationen Tabelle 3.7 zu entnehmen.

Tabelle 3.7: Anzahl der Nichtnullelemente und deren Anteil an den Gesamteinträgen in J bzw. J_d für verschiedene Konfigurationen.

Konfiguration	γ	Anteil
$(A \dots L)$	4198032	0,16 %
$(ABC) \dots (JKL)$	$4 \cdot 262377 = 1049508$	0,64 %
$(A) \dots (L)$	$12 \cdot 29153 = 349836$	1,93 %

In den Tabellen 3.8 bis 3.10 werden die Anzahl der Newtonschritte pro Zeitschritt, die Assemblierungszeit, die Zeit des linearen Lösers und die Gesamtrechenzeit für *SuperLU* und *BiCGStab* mit einem symmetrischen Gauß-Seidel-Verfahren als Vorkonditionierer (ein Iterationsschritt) bei Verwendung verschiedener Konfigurationen (Blockbildungen) aufgelistet. Bei Aufteilung der Jacobi-Matrix in mehrere Teilsysteme wird zwischen Berechnungen mit

- Assemblierung der vollen Jacobi-Matrix J und

- Assemblierung der ausgedünnten Jacobi-Matrix J_d (ohne die vernachlässigten Reaktionen) gemäß Abschnitt 3.1.3

unterschieden. Im ersten Fall werden durch die Aufteilung des Gleichungssystems lediglich die für die Kopplung zuständigen Elemente von J ignoriert, während im zweiten Fall alle zu einer vernachlässigten Reaktion gehörenden Einträge nicht assembliert werden. Die zweite Variante führt also zu einer stärkeren Ausdünnung. Es soll getestet werden, wie sich diese auf das Konvergenzverhalten des modifizierten Verfahrens und auf die Assemblierungszeit auswirkt.

In Tabelle 3.8 zerfällt das lineare Gleichungssystem dank der speziell gewählten Damköhler-Zahlen auf natürliche Weise in vier Teilsysteme mit je drei Spezies. Die Anzahl der Newtonschritte pro Zeitschritt bleibt konsequenterweise konstant. Die Assemblierungszeit steigt bei Aufteilung und vollständiger Assemblierung etwas an. Ursache hierfür ist der zusätzliche Aufwand für das Kopieren der Gesamtmatrix in die einzelnen Teilmatrizen. Die Abweichungen der Assemblierungszeiten von direktem und iterativem Löser werden durch die nicht exakte Zeitmessung vom $M{+}{+}$ verursacht.

Tabelle 3.8: Newtonschritte pro Zeitschritt (NS/TS), CPU-Zeiten für Assemblierung (Ass), linearen Löser (LS) und Gesamtzeit (Ges) in Minuten für verschiedene Reaktionen und Konfigurationen (Blockbildungen) mit Assemblierung der vollständigen Jacobi-Matrix J und der ausgedünnten Matrix J_d für die Damköhlerzahlen $(1000, 0, 1000, 1000, 0, 1000)$.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	vollständig	1,9	157	1076	1235
SuperLU	$(ABC) \dots (JKL)$	vollständig	1,9	165	52	221
SuperLU	$(ABC) \dots (JKL)$	J_d	1,9	164	53	220
BiCGStab	$(A \dots L)$	vollständig	1,9	156	60	228
BiCGStab	$(ABC) \dots (JKL)$	vollständig	1,9	165	10	179
BiCGStab	$(ABC) \dots (JKL)$	J_d	1,9	164	10	178

Bei dieser Simulation besteht kein Zeitunterschied zwischen voller Assemblierung (J) und Assemblierung der ausgedünnten Matrix (J_d). Durch die Aufteilung von J fallen nur ca. 7% der Einträge weg, d.h.

$$\gamma\left(J_d\right) \;\approx\; 0,93\;\gamma\left(J\right).$$

Diese relativ kleine Einsparung wird durch den erhöhten organisatorischen Aufwand zunichte gemacht, der zur Identifikation der gelöschten Einträge nötig ist. Es fällt weiter auf, dass der Beschleunigungsfaktor des linearen Lösers beim direkten SuperLU-Verfahren mit ca. 18 deutlich höher ist als beim iterativen BiCGStab-Verfahren mit 6. Das bestätigt die Überlegungen aus den Beispielen 3.2 und 3.9.

Die Gesamtrechenzeit beschleunigt sich bei *SuperLU* um den Faktor $5,5$ und bei *BiCGStab* um den Faktor $1,2$. Das direkte Verfahren kann durch die Aufteilung wie erwartet stärker beschleunigt werden als das iterative. Trotzdem erreicht es nicht die kurzen Rechenzeiten von *BiCGStab*.

Tabelle 3.9: Performance für die Damköhlerzahlen $(1000, 1, 1000, 1000, 1, 1000)$ mit den Bezeichnungen aus Tabelle 3.8.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	vollständig	$3,0$	304	2371	2679
SuperLU	$(ABC) \dots (JKL)$	vollständig	$3,0$	329	200	534
SuperLU	$(ABC) \dots (JKL)$	J_d	$3,0$	327	200	531
BiCGStab	$(A \dots L)$	vollständig	$3,0$	303	125	433
BiCGStab	$(ABC) \dots (JKL)$	vollständig	$3,0$	329	36	371
BiCGStab	$(ABC) \dots (JKL)$	J_d	$3,0$	327	36	368

Tabelle 3.9 zeigt die Performance für leicht veränderte Damköhlerzahlen. Die Reaktionen R_1, R_3, R_4 und R_6 bleiben bei $Da = 1000$, R_2 und R_5 werden auf $Da = 1$ erhöht, so dass durch die Aufteilung diesmal tatsächlich Kopplungsterme aus J eliminiert werden.
Bei beiden linearen Lösern erhöht sich die Anzahl der Newtonschritte pro Zeitschritt nicht. Das ist für modifizierte Newtonverfahren nicht selbstverständlich. Die Kopplung aufgrund der niedrigen Damköhlerzahlen ($Da = 1$) ist wohl zu schwach, um größere Probleme im Newtonverfahren zu verursachen.
Auch hier ist die Assemblierungszeit bei voller Assemblierung der Matrix J im Rahmen der Toleranz identisch mit der Assemblierungszeit der ausgedünnten Matrix J_d. Da die Blockbildung genauso erfolgt wie im ersten Beispiel, ist diese Beobachtung wie oben zu erklären.
Der Aufwand für den linearen Löser sinkt um das 12-fache beim direkten Verfahren und um das $3,5$-fache beim iterativen. Auch hier zeigt sich das größere Einsparpotential des direkten Lösers. Trotzdem kann er nicht mit der Performance des iterativen Verfahrens konkurrieren.
Die Gesamtzeit beschleunigt sich bei *SuperLU* um das $4,9$- und bei *BiCGStab* um das $1,1$-fache.

Bei der anschließenden Simulationsgruppe aus Tabelle 3.10 wurde für alle ablaufenden Reaktionen die relativ niedrige Damköhlerzahl $Da = 1$ verwendet. Die Reaktionen sind also gleichberechtigt und verglichen mit den Damköhlerzahlen $Da = 1000$ aus den bisherigen Simulationen deutlich schwächer.
Auch hier zeigt sich das bereits anhand von Tabelle 3.9 dokumentierte Verhalten. Darüber hinaus ist zu bemerken, dass das ursprüngliche Problem (volle Assemblierung) selbst bei Aufteilung in 12 Teilsysteme vom Newtonverfahren noch

Tabelle 3.10: Performance für die Damköhlerzahlen $(1, 1, 1, 1, 1, 1)$ mit den Bezeichnungen aus Tabelle 3.8.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	vollständig	$2, 2$	193	1392	1588
SuperLU	$(ABC) \dots (JKL)$	vollständig	$3, 0$	324	197	526
SuperLU	$(ABC) \dots (JKL)$	J_d	$3, 0$	326	230	561
SuperLU	$(A) \dots (L)$	vollständig	$4, 0$	453	98	557
SuperLU	$(A) \dots (L)$	J_d	$3, 0$	317	75	399
BiCGStab	$(A \dots L)$	vollständig	$2, 2$	188	92	295
BiCGStab	$(ABC) \dots (JKL)$	vollständig	$3, 0$	317	33	354
BiCGStab	$(ABC) \dots (JKL)$	J_d	$3, 0$	318	35	359
BiCGStab	$(A) \dots (L)$	vollständig	$4, 0$	455	27	488
BiCGStab	$(A) \dots (L)$	J_d	$3, 0$	317	22	345

Tabelle 3.11: Anteil der Transport- und Reaktionsterme an der Gesamtassemblierungszeit im Mittel für die Beispiele aus diesem Abschnitt.

Term	Anteil
Assemblierung der Transportterme	75 %
Assemblierung der Reaktionsterme	15 %
Sonstiges (Speicherverwaltung, Ausgabe, ...)	10 %

gelöst werden kann. Allerdings liegt die dazu benötigte Gesamtzeit beim iterativen Verfahren deutlich höher als die Zeit zur Lösung des vollen Problems (ohne Aufteilung).

Obwohl in dieser Simulation bei Aufteilung in 12 Teilsysteme mit Assemblierung der ausgedünnten Matrix alle Reaktionsterme in J ignoriert werden, unterscheidet sich die Assemblierungszeit pro Newtonschritt nicht von der mit vollständiger Assemblierung. Bei der Simulation mit *BiCGStab* und Konfiguration $(A \dots L)$ (keine Aufteilung) benötigt die Assemblierung insgesamt 188 Minuten. Davon entfallen weniger als 2 Minuten auf die Berechnung der Ableitungen der Reaktionsterme, die bei teilweiser Assemblierung nicht ermittelt werden müssen.
Aus Tabelle 3.11 sind die Anteile für die Assemblierung der Transport- und Reaktionsterme an der gesamten Assemblierungszeit zu entnehmen. Demnach benötigt die Berechnung der Transportanteile bereits 75 % der Gesamtzeit, so dass die minimalen Unterschiede zwischen den Assemblierungszeiten von J und J_d aus den drei Beispielen verständlich sind.

Die Anzahl der Newtonschritte pro Zeitschritt zeigt bei voller Aufteilung des linearen Gleichungssystems ein untypisches Verhalten: Bei vollständiger Assemblierung der Jacobi-Matrix werden mehr Newtonschritte benötigt als bei Assemblierung von J_d. Trotz dieser zusätzlichen Ausdünnung konvergiert das modifizierte

Verfahren also besser als bei voller Assemblierung von J.
Als Ursache ist zu vermuten, dass die (im Nichtlinearen unbekannte) Kontraktionszahl im vorliegenden Spezialfall mit der zusätzlichen Ausdünnung kleiner ist als ohne. Allerdings konnte ein vergleichbares Phänomen bei keiner anderen Simulation beobachtet werden.

3.3.2 Simulationen mit veränderten Schrittweiten

Die Vergleichssimulationen von oben haben unter anderem gezeigt, dass sich das modifizierte Newton-Verfahren im Zweidimensionalen für direkte lineare Löser ähnlich verhält wie im Eindimensionalen [47]. Allerdings war der iterative lineare Löser ohne Entkopplung meist schneller als der direkte, selbst wenn das Gleichungssystem maximal entkoppelt wurde. Wegen der Dominanz der Assemblierungszeit und des relativ geringen Gewinns im iterativen Verfahren konnte zudem durch die Aufteilung kein nennenswerter Gewinn erzielt werden.
Diese unbefriedigende Situation lässt sich durch veränderte Schrittweiten verbessern. In folgenden werden nochmals die Testsimulationen aus den Tabellen 3.8 bis 3.10 mit $\Delta t = 1,0\ s$ (vorher $0,01\ s$) und 16705 Knoten (vorher 4257) berechnet. Die Kennzahlen nehmen hier die Werte

$$\begin{aligned} Pe &= 1000, \\ Pe_T &\approx 3,9 \quad \text{und} \\ Cr &= 25,6 \end{aligned}$$

an. Für diese großen Courant-Zahlen ist der Einsatz eines impliziten Verfahrens gerechtfertigt.

Bei der Lösung des vollständigen linearen Gleichungssystem assemblieren wir die Jacobimatrix weiterhin vollständig (klassisches Newton-Verfahren), während bei der Lösung des aufgeteilten Systems nur J_d aufgebaut wird.

Tabelle 3.12: Performance für die Damköhlerzahlen $(1000, 0, 1000, 1000, 0, 1000)$ mit den Bezeichnungen aus Tabelle 3.8.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	vollständig	Speicher	über	lauf	
SuperLU	$(ABC) \dots (JKL)$	J_d	3,4	15	17	34
BiCGStab	$(A \dots L)$	vollständig	3,4	14	66	83
BiCGStab	$(ABC) \dots (JKL)$	J_d	3,4	15	11	28

Zunächst ist festzustellen, dass der direkte Löser *SuperLU* bei Verwendung des vollständigen linearen Gleichungssystems bei allen drei Simulationen aus den Tabellen 3.12 bis 3.14 einen Speicherüberlauf erzeugt. Der fill-in ist trotz der Bandreduzierung mit dem Cuthill-Mc-Kee-Algorithmus so groß, dass der verfügbare

Speicherplatz nicht ausreicht. Bei der Aufteilung in vier bzw. zwölf kleinere Gleichungssysteme tritt dieses Problem nicht auf. Das Entkopplungsverfahren führt also erst dazu, dass die Simulationen mit einem direkten Verfahren durchgeführt werden können.

Tabelle 3.13: Performance für die Damköhlerzahlen $(1000, 1, 1000, 1000, 1, 1000)$ mit den Bezeichnungen aus Tabelle 3.8.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	vollständig	Speicher	über	lauf	
SuperLU	$(ABC) \dots (JKL)$	J_d	$4,5$	22	47	70
BiCGStab	$(A \dots L)$	vollständig	$3,7$	15	162	180
BiCGStab	$(ABC) \dots (JKL)$	J_d	$4,5$	22	64	87

Anders als bei den Berechnungen mit den ursprünglichen Schrittweiten ist das entkoppelte Verfahren mit dem direkten linearen Löser in allen Fällen deutlich schneller als das vollständige mit iterativem Löser. Die Gesamtrechenzeit verringert sich um mehr als die Hälfte.
Auch die Simulationen mit iterativem Löser selbst beschleunigen sich deutlich. Die Assemblierungszeit ist hier klein im Vergleich mit der Zeit für den linearen Löser, so dass der Gewinn in *BiCGStab* einen deutlicheren Einfluss auf die Gesamtzeit hat als vorher.

Tabelle 3.14: Performance für die Damköhlerzahlen $(1, 1, 1, 1, 1, 1)$ mit den Bezeichnungen aus Tabelle 3.8.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	vollständig	Speicher	über	lauf	
SuperLU	$(ABC) \dots (JKL)$	J_d	$5,2$	26	57	85
SuperLU	$(A) \dots (L)$	J_d	konver	giert	nicht	
BiCGStab	$(A \dots L)$	vollständig	$3,1$	12	154	168
BiCGStab	$(ABC) \dots (JKL)$	J_d	$5,2$	26	78	105
BiCGStab	$(A) \dots (L)$	J_d	konver	giert	nicht	

Die Simulation mit voller Aufteilung des linearen Gleichungssystems aus Tabelle 3.14 konvergiert für die verwendete hohe Zeitschrittweite nicht mehr.

3.4 Das Broyden-Verfahren

Bei den letzten Simulationen in 3.3.2 war das modifizierte Verfahren mit einem direkten linearen Löser schneller als mit einem iterativen. Hier soll nun versucht werden, diesen Vorteil mit Hilfe eines weiteren modifizierten Newton-Verfahrens, des sogenannten Broyden-Verfahrens, noch zu vergrößern.

3.4.1 Beschreibung des Verfahrens

In Abschnitt 2.2.2 wurde die Lösung des Gleichungssystems (2.10), (2.11) und (2.12) mit dem Newton-Verfahren (2.13) beschrieben. Der große Vorteil dieses „exakten“ Newton-Verfahrens ist seine quadratische Konvergenz. Es erfordert allerdings in jedem Newtonschritt die rechenzeitintensive Neuassemblierung der Jacobi-Matrix (2.15). Falls das lineare Gleichungssystem (2.13) mit einem direkten Verfahren gelöst werden soll, muss hierfür außerdem in jedem Schritt ein Aufwand der Ordnung $O\left(n^3\right)$ in Kauf genommen werden.
Das Broyden-Verfahren kann diesen Nachteil eventuell ausgleichen, indem es nur im ersten Newtonschritt eine $LU-$Zerlegung von J berechnet. In den folgenden Schritten wird lediglich ein Rang-1-Update der Jacobi-Matrix ermittelt. Mit dessen Hilfe und der $LU-$Zerlegung aus dem ersten Schritt kann schließlich eine bessere Näherungslösung gefunden werden.

Broydens Rang-1-Modifikation [12, 20]:

Zu lösen ist das Nullstellenproblem finde $\mathbf{x} \in \mathbb{R}^n$ mit

$$\mathbf{f}\left(\mathbf{x}\right) \;=\; 0.$$

Bei der Methode von Broyden wird zunächst eine $LU-$Zerlegung der Jacobi-Matrix des ersten Newtonschrittes berechnet, d.h.

$$LU \;=\; D\mathbf{f}\left(\mathbf{x}_1\right) \;=:\; J_1.$$

Mit Hilfe dieser Zerlegung erfolgt die Berechnung des Korrekturvektors δ_1:

$$LU\boldsymbol{\delta}_1 \;=\; -\mathbf{f}\left(\mathbf{x}_1\right).$$

Ab dem zweiten Newton-Schritt wird die Jacobi-Matrix nicht mehr neu assembliert, sondern gemäß der Rang-1-Modifikation

$$J_k \;:= J_{k-1} + \frac{\mathbf{F}_k\boldsymbol{\delta}_{k-1}^T}{\|\boldsymbol{\delta}_{k-1}\|^2} \quad \text{für } k \geq 2, \tag{3.8}$$

rekursiv berechnet, wobei $\mathbf{F}_k := \mathbf{f}\left(\mathbf{x}_k\right)$.
Setzt man (3.8) in das Newton-Verfahren (2.13) ein, ergibt sich der neue Korrekturvektor

$$\boldsymbol{\delta_k} \;=\; -\left[J_{k-1} + \frac{\mathbf{F}_k\boldsymbol{\delta}_{k-1}^T}{\|\boldsymbol{\delta}_{k-1}\|^2}\right]^{-1}\mathbf{F}_k.$$

Außerdem erhält man

$$J_k^{-1} \;=\; \left(I + \frac{\boldsymbol{\delta}_k\boldsymbol{\delta}_{k-1}^T}{\|\boldsymbol{\delta}_{k-1}\|^2}\right)J_{k-1}^{-1} \quad \text{für } k \geq 2 \tag{3.9}$$

zur Berechnung der Inversen J_k^{-1} aus der inversen Jacobi-Matrix J_{k-1}^{-1} des vorhergehenden Newton-Schrittes.
Durch rekursive Anwendung von (3.9) folgt

$$J_{k-1}^{-1} \;=\; \left[I + \frac{\boldsymbol{\delta}_{k-1}\boldsymbol{\delta}_{k-2}^T}{\|\boldsymbol{\delta}_{k-2}\|^2}\right] \ldots \left[I + \frac{\boldsymbol{\delta}_2\boldsymbol{\delta}_1^T}{\|\boldsymbol{\delta}_1\|^2}\right] J_1^{-1} \quad \text{für } k \geq 2. \tag{3.10}$$

Nach Einsetzen von (3.9) in (2.13) lautet der Korrekturvektor des $k-$ten Newtonschrittes

$$\boldsymbol{\delta}_k \;=\; \left[I - \frac{\boldsymbol{\delta}_{k-1}\left(-J_{k-1}^{-1}\mathbf{F}_k\right)^T}{\|\boldsymbol{\delta}_{k-1}\|^2}\right]^{-1} \left(-J_{k-1}^{-1}\mathbf{F}_k\right) \quad \text{für } k \geq 2. \tag{3.11}$$

Der Vektor $J_{k-1}^{-1}\mathbf{F}_k$ wird mit Hilfe von (3.10) berechnet. Dabei kann die bereits aus dem ersten Newtonschritt bekannte LU-Zerlegung von J_1 verwendet werden, so dass nur noch eine Vorwärts-Rückwärts-Substitution mit LU sowie eine Reihe von Skalarprodukten und Vektoradditionen berechnet werden müssen. Der Aufwand hierfür ist von der Ordnung $O\left(n^2\right)$.
Des weiteren ist noch die inverse Matrix aus (3.11) zu bestimmen. Dies geschieht mit der Sherman-Morrison-Woodbury-Formel [28], mit der die Lösung eines Rang-1-modifizierten Gleichungssystems der Form

$$\left(A + \mathbf{u}\mathbf{v}^T\right)\mathbf{x} \;=\; \mathbf{b}$$

wie folgt berechnet werden kann:

$$\mathbf{x} \;=\; \left(A + \mathbf{u}\mathbf{v}^T\right)^{-1}\mathbf{b} \;=\; A^{-1}\mathbf{b} - \frac{A^{-1}\mathbf{u}\mathbf{v}^T A^{-1}\mathbf{b}}{1 + \mathbf{v}^T A^{-1}\mathbf{u}}.$$

Der Aufwand ist $O\left(n\right)$, da in unserem Fall $A := I$ gilt. Eingesetzt in (3.11) ist der Korrekturvektor für $k \geq 2$

$$\boldsymbol{\delta}_k \;=\; -J_{k-1}^{-1}\mathbf{F}_k - \frac{\|J_{k-1}^{-1}\mathbf{F}_k\|^2}{\|\boldsymbol{\delta}_{k-1}\|^2 + \left(J_{k-1}^{-1}\mathbf{F}_k\right)^T\boldsymbol{\delta}_{k-1}}\,\boldsymbol{\delta}_{k-1}. \tag{3.12}$$

Insgesamt lässt sich feststellen, dass der Aufwand für die Lösung des linearen Gleichungssystems im ersten Newtonschritt nach wie vor von der Ordnung $O\left(n^3\right)$ ist. Alle folgenden Newtonschritte lassen sich allerdings mit $O\left(n^2\right)$ durchführen.

Konvergenz:

In [12] wird die Fehlerabschätzung

$$\|\mathbf{x}_{k+1} - \mathbf{x}\| \;\leq \Theta\|\mathbf{x}_k - \mathbf{x}\| \quad \text{mit } 0 < \Theta < 1$$

gezeigt und es gilt

$$\|\boldsymbol{\delta}_{k+1}\| \;\leq\; \Theta\|\boldsymbol{\delta}_k\|.$$

Wie bereits erwähnt geht die quadratische Konvergenz verloren. Das Verfahren konvergiert aber immer noch superlinear, d.h.

$$\lim_{k\to\infty} \frac{\|\boldsymbol{\delta}_{k+1}\|}{\|\boldsymbol{\delta}_k\|} \;=\; 0.$$

3.4.2 Beispielsimulationen

Betrachtet wird auch hier die Testsimulation aus Beispiel 3.7 mit $M = 4257$ (ursprüngliche räumliche Schrittweite aus Abschnitt 3.3.1).

Das Broyden-Verfahren konvergiert für die Beispielsimulationen aus den Tabellen 3.8 bis 3.10 nicht. Als Ursache konnte der im Vergleich zum vollen Newton-Verfahren eingeschränkte Konvergenzbereich identifiziert werden.
Um das Einsparpotential beim Broyden-Verfahren im Vergleich zum Newton-Verfahren mit iterativem Löser abschätzen zu können, wurde die Testsimulation mit der kleineren Schrittweite $\Delta t = 0,001s$ und 3000 Zeitschritten berechnet. Die Damköhler-Zahlen entsprechen denen aus Tabelle 3.8. Für die übrigen Kenngrößen gilt

$$\begin{aligned} Pe &= 1000, \\ Pe_T &\approx 7,8 \quad \text{und} \\ Cr &= 0,0128. \end{aligned}$$

Die Courant-Zahl ist noch kleiner als in Abschnitt 3.3.1. Wir werden in Tabelle 3.15 sehen, dass das Broyden-Verfahren selbst in diesem Fall keine befriedigenden Resultate liefert.

Tabelle 3.15: Performance für die Damköhlerzahlen $(1000, 0, 1000, 1000, 0, 1000)$ mit den Bezeichnungen aus Tabelle 3.8.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
SuperLU	$(A \ldots L)$	vollständig	3,0	301	2600	2904
SuperLU	$(ABC) \ldots (JKL)$	vollständig	3,0	321	244	612
SuperLU	$(ABC) \ldots (JKL)$	A_d	3,0	264	173	443
Broyden	$(A \ldots L)$	vollständig	4,1	235	1317	1579
Broyden	$(ABC) \ldots (JKL)$	vollständig	4,1	247	123	375
Broyden	$(ABC) \ldots (JKL)$	A_d	konver	giert	nicht	
BiCGStab	$(A \ldots L)$	vollständig	3,0	301	87	413
BiCGStab	$(ABC) \ldots (JKL)$	vollständig	3,0	314	15	336
BiCGStab	$(ABC) \ldots (JKL)$	A_d	3,0	262	15	283

Der Vergleich von *SuperLU* und *BiCGStab* zeigt das bereits beschriebene Verhalten. Das Broyden-Verfahren benötigt im Vergleich mit dem vollen Newton-Verfahren mehr Newtonschritte zur näherungsweisen Lösung. Diese Beobachtung ist durch die inexakte Berechnung der Jacobi-Matrix ab dem zweiten Newtonschritt zu erklären.
Die Assemblierungszeiten sind etwas niedriger als beim vollen Newton-Verfahren. Die für uns entscheidende Zeit für den linearen Löser ist nur halb so groß wie beim exakten Verfahren mit *SuperLU*. Leider ist sie immer noch größer als beim iterativen linearen Löser.

Folgerung 3.14. Direkte Löser sind demzufolge für kleine Zeitschrittweiten auch unter Verwendung des Broyden-Verfahrens nicht konkurrenzfähig. Bei größeren Δt entstehen kaum behebbare Konvergenzprobleme. Wir benutzen deshalb im Folgenden nur noch das Newtonverfahren aus (2.13) mit geeignet großem Δt.

Kapitel 4

Automatisierungen

Die Testsimulationen aus Abschnitt 3.3 haben gezeigt, dass für das modifizierte Newton-Verfahren aus Kapitel 3 (bei einfachen Beispielen) nur unter Verwendung großer Zeitschrittweiten ein nennenswerter Gewinn im Vergleich zum vollen Newton-Verfahren entsteht. Diese Tatsache wirft die Frage nach einer geeigneten Wahl von Δt auf, die im Allgemeinen nur durch Einführung eines adaptiven Verfahrens zu beantworten ist. Dessen Beschreibung ist ebenso wie die Durchführung einiger Testsimulationen Inhalt des ersten Abschnittes des vorliegenden Kapitels.

In 4.2 wird davon ausgegangen, dass kein a-priori-Wissen über das zu lösende lineare Gleichungssystem vorhanden ist. Eine manuelle Aufteilung ist dann nicht mehr möglich. Deshalb soll in diesem Abschnitt einleitend ein Algorithmus vorgestellt werden, mit dessen Hilfe die Zusammenhangskomponenten des noch zu definierenden Reaktionsgraphen ermittelt werden können.
Anschließend erfolgt die Entwicklung einer auf den Konvergenzbetrachtungen aus Abschnitt 3.2 beruhenden Entkopplungsstrategie. Mit deren Hilfe kann das lineare Gleichungssystem unter Verwendung des in Abschnitt 4.2.1 vorgestellten Algorithmus automatisch in Teilsysteme partitioniert werden.

4.1 Zeitadaption

4.1.1 Das Verfahren

Das adaptive Verfahren soll die Anzahl der Newtonschritte NS_i für alle Zeitschritte i in den Grenzen

$$NS_{min} \leq NS_i \leq NS_{max}$$

halten. Falls das (modifizierte) Newton-Verfahren im $i-$ten Zeitschritt weniger als NS_{min} Newtonschritte benötigt, falls also $NS_i < NS_{min}$ ist, wird die Zeitschrittweite Δt mit einem Faktor $fact_1 > 1$ multipliziert. Sollten jedoch mehr als

NS_{max} Newtonschritte benötigt werden, d.h. $NS_i > NS_{max}$, erfolgt eine Multiplikation von Δt mit $fact_2 < 1$.
Um die Veränderung der Zeitschrittweite in sinnvollen Grenzen halten zu können, werden außerdem obere und untere Schranken für Δt eingeführt. Die Schrittweite kann Δt_{min} nicht unter- und Δt_{max} nicht überschreiten.

Die Parameter NS_{min}, NS_{max}, $fact_1$ und $fact_2$ können frei gewählt werden. Während für $fact_1$ und $fact_2$ Werte in den Bereichen

$$\begin{aligned} 1,2 &\leq fact_1 \leq 2, \\ 0,25 &\leq fact_2 \leq 0,75, \end{aligned}$$

in der Regel gute Ergebnisse liefern, hängen NS_{min} und NS_{max} sehr stark vom konkreten Beispiel ab und müssen bei jeder Simulation abgeschätzt werden. Aus zahlreichen Testsimulationen kann allerdings der Schluss gezogen werden, dass NS_{min} und NS_{max} bei Verwendung des modifizierten Verfahrens etwas höher gewählt werden sollten als beim vollen Newton-Verfahren.
Durch die Aufteilung des Gesamtsystems erhöht sich meist die Anzahl der Newtonschritte (siehe Testsimulationen aus 3.3), so dass bei identischen Grenzen die Zeitschrittweite beim vollen Verfahrens deutlich höher gewählt werden würde als beim aufgeteilten. Der Gewinn durch das größere Δt hat in allen durchgeführten Simulationen den Gewinn durch die Aufteilung überwogen und das modifizierte Verfahren hat zur Lösung deutlich mehr Zeit benötigt.

4.1.2 Beispielsimulationen

Wir führen hier nochmals die Testsimulationen aus Abschnitt 3.3 durch und verwenden diesmal das oben beschriebene zeitadaptive Verfahren. Die untere Schranke NS_{min} wird dabei auf Grundlage der Ergebnisse aus den Tabellen 3.8 bis 3.10 gesetzt und kann wegen der oben beschriebenen erhöhten Anzahl der Newtonschritte beim modifizierten Verfahren von der Schranke des vollen Verfahrens abweichen.
Diese unterschiedliche Wahl von NS_{min} führt dazu, dass die Performance der Simulationen mit bzw. ohne Entkopplung untereinander nicht direkt verglichen werden können. Durch die Berechnungen soll lediglich die Wirksamkeit der Zeitadaption demonstriert werden. Die Werte aus den folgenden Tabellen sind deshalb nur mit den entsprechenden Werten aus 3.3 zu vergleichen.
Die obere Schranke wurde für alle folgenden Simulationen auf $NS_{max} := 10$ gesetzt. Sie spielt jedoch in den meisten Fällen keine Rolle, da die Anzahl der Newtonschritte beim verwendeten Beispiel 3.7 mit zunehmender Simulationszeit abnimmt.
Die verwendeten Faktoren für die Vergrößerung bzw. Verkleinerung von Δt lauten

für alle Testbeispiele

$$\begin{aligned} fact_1 &:= 2, \\ fact_1 &:= 0,5. \end{aligned}$$

Die Startschrittweite ist für alle Beispiele $\Delta t_0 := 0,01\ s$, was gerade der Schrittweite aus den Tabellen 3.8 bis 3.10 entspricht. Die minimale und maximale Schrittweite lautet

$$\begin{aligned} \Delta t_{min} &:= \Delta t_0 = 0,01\ s, \\ \Delta t_{max} &:= 5\ s. \end{aligned}$$

Für die Kenngrößen aus Abschnitt 2.2.4 gilt

$$\begin{aligned} Pe &= 1000, \\ Pe_T &\approx 7,8 \quad \text{und} \\ Cr &\in [0,128\ ;\ 7,8]\,. \end{aligned}$$

In Tabelle 4.1 zerfällt das Gleichungssystem wieder auf natürliche Weise in vier Teilsysteme. NS_{min} wurde deshalb für die Simulationen mit und ohne Entkopplung identisch gewählt. Der obere Plot von Abbildung 4.1 zeigt den zugehörigen Verlauf von Δt.

Tabelle 4.1: Schranke NS_{min}, Anzahl der Zeitschritte (TS), Newtonschritte pro Zeitschritt (NS/TS), CPU-Zeiten für Assemblierung (Ass), linearen Löser (LS) und Gesamtzeit (Ges) in Minuten für verschiedene Konfigurationen für die Damköhler-Zahlen $(1000, 0, 1000, 1000, 0, 1000)$ mit Zeitadaption.

Löser	Konfiguration	NS_{min}	TS	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	3	291	4,0	42	336	379
SuperLU	$(ABC) \dots (JKL)$	3	265	4,0	40	15	56
BiCGStab	$(A \dots L)$	3	291	4,0	42	21	66
BiCGStab	$(ABC) \dots (JKL)$	3	265	4,0	40	3	44

Bei der Berechnung ohne Entkopplung wird aufgrund der verwendeten Damköhler-Zahlen das gleiche Problem gelöst wie im entkoppelten Fall. Trotzdem unterscheidet sich die Anzahl der Zeitschritte. Die Kurven von Δt in Abbildung 4.1 oben verlaufen zwar fast gleich, geringe Unterschiede sind aber dennoch zu erkennen. Dieser scheinbare Widerspruch ist mit den in Abschnitt 3.1.4 beschriebenen unterschiedlichen Abbruchkriterien zu erklären.
Gegen Ende der Simulation ($t > 21\ s$) sinkt die Zeitschrittweite wieder ab. Diese Reduzierung ist nötig, um den Endzeitpunkt T exakt zu erreichen. Sie wird nicht durch eine steigende Anzahl von Newtonschritten verursacht.

Tabelle 4.2: Performance für die Damköhlerzahlen $(1, 1, 1, 1, 1, 1)$ mit Zeitadaption und den Bezeichnungen aus Tabelle 4.1.

Löser	Konfiguration	NS_{min}	TS	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	3	18	3,3	2	16	19
SuperLU	$(ABC) \dots (JKL)$	4	31	5,6	7	5	12
SuperLU	$(A) \dots (L)$	4	1108	4,9	224	160	394
BiCGStab	$(A \dots L)$	3	18	3,3	2	9	11
BiCGStab	$(ABC) \dots (JKL)$	4	31	5,6	7	8	15
BiCGStab	$(A) \dots (L)$	4	1108	4,9	225	87	318

Der Vergleich der Tabellen 4.1 und 3.8 (Seite 38, $\Delta t = 0,01\ s$) zeigt ein deutliches Absinken der Rechenzeiten. Die Gesamtzeiten verringern sich um 70 % bis 75 %.

In Tabelle 4.3 mit schwacher Kopplung der vier Teilsysteme $(ABC) \dots (JKL)$ wurden unterschiedliche Werte für NS_{min} gewählt. Es ist zu erkennen, dass die Anzahl der Newtonschritte pro Zeitschritt bei Entkopplung zwar ansteigt, die Anzahl der Zeitschritte ist jedoch niedriger.
Der mittlere Plot in Abbildung 4.1 zeigt, dass Δt im entkoppelten Fall vor allem am Anfang der Simulation stärker steigt als beim voll gekoppelten Verfahren. Später steigt die Zeitschrittweite beim vollen Verfahren stärker an und wird bei $t \approx 12\ s$ sogar größer als beim entkoppelten. Den vorher erworbene Vorsprung des aufgeteilten Verfahrens kann es jedoch nicht mehr einholen.

Tabelle 4.3: Performance für die Damköhlerzahlen $(1000, 1, 1000, 1000, 1, 1000)$ mit Zeitadaption und den Bezeichnungen aus Tabelle 4.1.

Löser	Konfiguration	NS_{min}	TS	NS/TS	Ass	LS	Ges
SuperLU	$(A \dots L)$	3	273	4,0	40	313	353
SuperLU	$(ABC) \dots (JKL)$	4	44	4,8	9	6	15
BiCGStab	$(A \dots L)$	3	273	4,0	40	25	69
BiCGStab	$(ABC) \dots (JKL)$	4	44	4,8	9	6	15

Der Vergleich mit Tabelle 3.8 (Seite 39, $\Delta t = 0,01\ s$) zeigt ein ähnliches Verhalten wie oben. Die Gesamtzeiten sinken sogar noch stärker um 84 % bis 97 %.

Bei der Simulation mit identischen Damköhlerzahlen aus Tabelle 4.2 steigen bei Aufteilung in vier Teilsysteme sowohl die Anzahl der Zeitschritte als auch die Newtonschritte pro Zeitschritt an. Durch die Entkopplung kann deshalb im Vergleich zum vollen Verfahren mit *BiCGStab* kein Gewinn erzielt werden.
Bei voller Entkopplung steigt die Zeitschrittweite kaum noch (siebe Abbildung 4.1 unten) und die Gesamtzeit ist fast so hoch wie bei fester Zeitschrittweite aus Tabelle 3.8 (Seite 39, $\Delta t = 0,01\ s$).

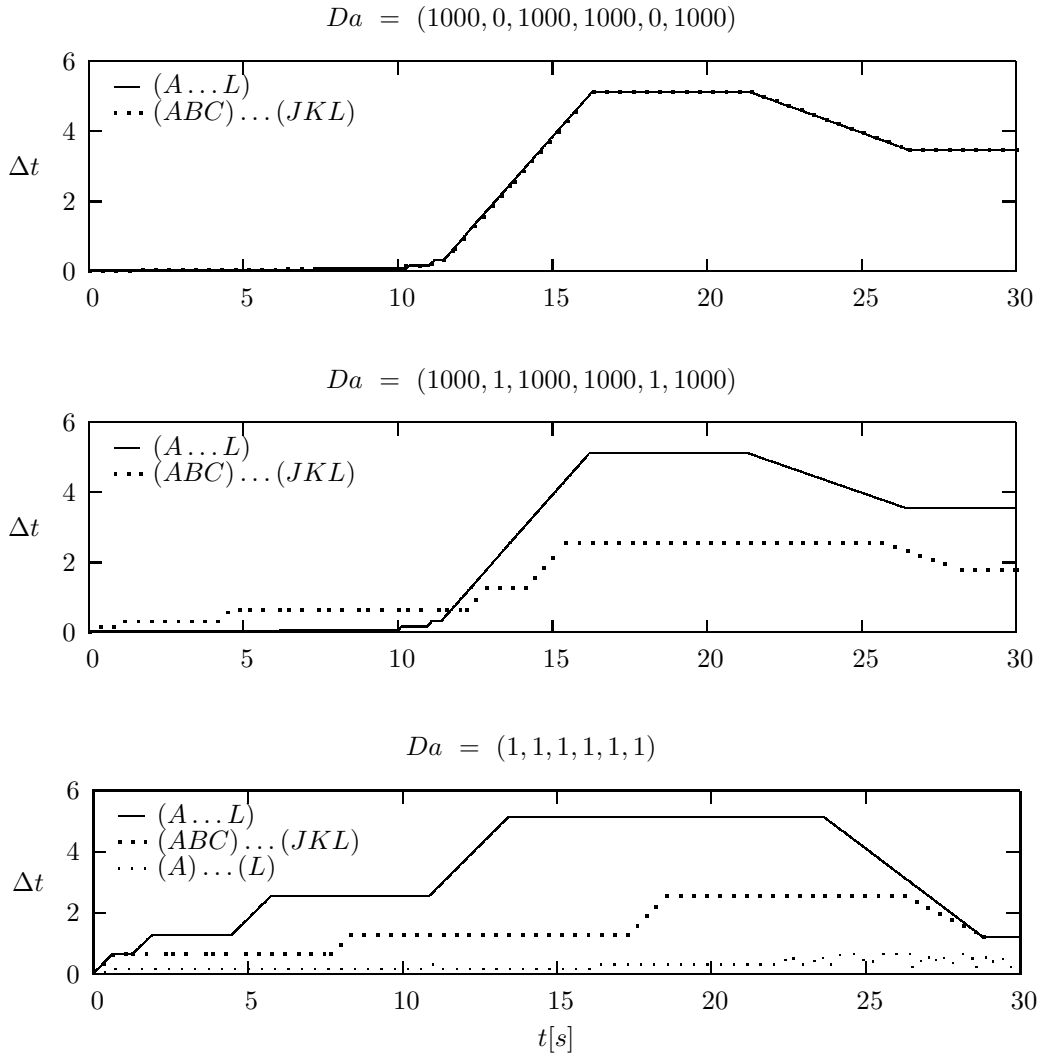

Abbildung 4.1: Zeitschrittweiten für die drei Beispielsimulationen mit und ohne Entkopplung.

4.2 Automatisierte Entkopplungsstrategien

In den Beispielen des letzten Abschnitts wurden die linearen Gleichungssystems noch manuell in Teilsysteme aufgeteilt. Im Folgenden soll die Blockbildung automatisiert werden.

4.2.1 Reaktionsgraphen und deren Aufteilung

Unter Verwendung der elementorientierten Sichtweise aus 2.2.3 kann die Matrix J als Adjazenzmatrix eines nichtorientierten kantenbewerteten Graphen $\mathcal{G}$ interpretiert werden. $\mathcal{G}$ setzt sich zusammen aus dem räumlichen Gitter $\mathcal{M}$ und den Reaktionsgraphen $\mathcal{R}_K$, wobei jeder räumliche Knoten $K \in \mathcal{M}$ wie in Abbildung 4.2 dargestellt Wurzel eines Reaktionsgraphen ist.

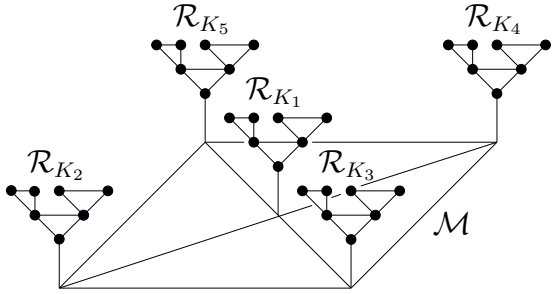

Abbildung 4.2: Gesamtgraph $\mathcal{G}$ aus räumlichem Netz $\mathcal{M}$ und Reaktionsgraphen $\mathcal{R}_K$.

Im Allgemeinen stimmen die einzelnen $\mathcal{R}_K$ nicht überein. Allerdings besitzen sie eine durch die Reaktionen vorgegebene gemeinsame Grundstruktur. Für das akademische Testbeispiel 3.7 ist diese in Abbildung 4.3 dargestellt. Es ist zu beachten, dass die Kantengewichte der einzelnen $\mathcal{R}_K$ stark unterschiedlich sein können.

Ziel der zu entwickelnden Strategie ist es, den Gesamtreaktionsgraphen $\mathcal{R}$ bis zu einem gewissen Grad in Teilgraphen $\mathcal{R}_i$ aufzuteilen. Dabei beginnen wir mit $\mathcal{R}$ und suchen eine Partitionierung in $\mathcal{R}_1$ und $\mathcal{R}_2$, so dass das Gesamtgewicht der aufgetrennten Kanten möglichst klein ist. $\mathcal{R}_1$ und $\mathcal{R}_2$ können auf die gleiche Weise weiter aufgeteilt werden.
Ein Verfahren, das die beschriebene Aufgabe erfüllt, ist der z.B. in [60] beschriebene Algorithmus von Stoer und Wagner. Er bestimmt den minimalen Schnitt von $\mathcal{R}$, d.h. eine Partition der Ecken (Spezies) des Graphen in die Teilgraphen $\mathcal{R}_1$ und $\mathcal{R}_2$. Außerdem werden die Kosten dieser Aufteilung (Summe der Gewichte der trennenden Kanten) ermittelt.

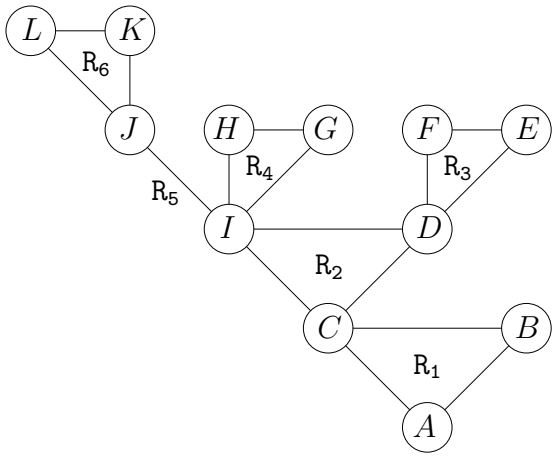

Abbildung 4.3: Reaktionsnetzwerk $\mathcal{R}$ für das akademische Testbeispiel 3.7.

Selbstverständlich können $\mathcal{R}_1$ und $\mathcal{R}_2$ durch nochmaligen Aufruf des Algorithmus weiter zerlegt werden. Es entsteht ein iteratives Verfahren, das den Gesamtgraphen $\mathcal{R}$ unter Angabe der Kosten so lange partitioniert, bis die kleinsten Teilgraphen nur noch eine Ecke enthalten.

Der Aufwand zur Zerlegung eines Graphen $\mathcal{R}$ mit N_S Knoten in die Teilgraphen $\mathcal{R}_1$ und $\mathcal{R}_2$ ist folgendem Satz zu entnehmen, den wir aus [60] zitieren:

Satz 4.1. *Der Algorithmus von Stoer und Wagner bestimmt mit Aufwand*

$$O\left(d_{\mathcal{R}}^{max} N_S \log N_S\right)$$

die Kosten eines minimalen Schnittes eines ungerichteten (Reaktions-)Graphen $\mathcal{R}$*.*

Beweis. Siehe [60, S. 229]. □

Bemerkung 4.2. $d_{\mathcal{R}}^{max}$ bezeichnet die maximale Anzahl von Spezies, die an einer Reaktion beteiligt sind. Bei realistischen Simulationen ist diese in der Regel deutlich kleiner als die Anzahl der Spezies N_S. Für das Beispiel 3.7 ist $N_S = 12$ und $d_{\mathcal{R}}^{max} = 3$.

Für weitere Ausführungen zur Funktionsweise des Algorithmus sei auf die zitierte Literatur verwiesen. In A.6 finden sich einige Anmerkungen zur Implementierung in *M++*.

4.2.2 Die Strategie

Ziel der zu entwickelnden Entkopplungsstrategie ist, das Gesamtsystem (3.1) in N_T Teilsysteme (3.2) zu zerlegen. Dabei sollen „schwache Bindungen“ zwischen den Spezies ignoriert werden, während „starke Bindungen“ erhalten bleiben.

Wir setzen ab jetzt voraus, dass entweder ein lineares Modell zu lösen ist oder ein nichtlineares, für das C_{lin} aus (3.6) sinnvolle Werte liefert (siehe Abschnitt 3.2.4). Sollten diese Bedingungen nicht erfüllt sein, sind alle folgenden Überlegungen zur Einbindung von C_{lin} in die Strategie zu ignorieren.

Die Teilsysteme werden wie bisher auf dem gesamten räumlichen Gitter $\mathcal{M}$ gelöst. Die aufzutrennenden Verbindungen müssen deshalb in jedem lokalen Reaktionsgraphen $\mathcal{R}_{K_i}$, $i = 1, \ldots, M$, schwach sein. Aufgrund der hohen Anzahl räumlicher Knoten ist es nicht möglich, alle $\mathcal{R}_{K_i}$ zu analysieren und daraus eine globale Aufteilung zu ermitteln.
Wir fassen statt dessen die lokalen Reaktionsgraphen zu einem repräsentierenden Graphen $\widetilde{\mathcal{R}}$ zusammen. Für die Matrizen bedeutet dies

$$A \;\mapsto\; \widetilde{A} \qquad \text{mit} \qquad \widetilde{A} \;\in\; \mathbb{R}^{N_S,N_S}, \tag{4.1}$$

wobei $\widetilde{A}$ als Adjazenzmatrix eines ungerichteten Graphen obere Dreiecksform besitzen soll.
Eine sinnvolle Wahl zur Identifizierung global schwacher Verbindungen zwischen den einzelnen Spezies ist

$$\widetilde{A}_{ij} \;:=\; \begin{cases} \max\limits_{\substack{\{k,l\in\{1,\ldots,n\}:\\ k \bmod N_S=i,\\ l \bmod N_S=j\}}} \{|A_{kl}|, |A_{lk}|\} & \text{für } i = 1, \ldots, N_S,\; j = i, \ldots, N_S, \\ \\ 0 & \text{sonst.} \end{cases} \tag{4.2}$$

Das Gewicht der Kanten von $\widetilde{\mathcal{R}}$ entspricht also dem betragsmäßig maximalen Gewicht der betreffenden Kanten über alle lokalen Reaktionsgraphen $\mathcal{R}_{K_i}$. Das bedeutet insbesondere, dass Kanten in $\widetilde{\mathcal{R}}$ mit kleinen Gewichten aufgetrennt werden können.

Bei der konkreten Bestimmung der Teilgraphen stellt sich die Frage, was der Ausdruck „kleines Gewicht“ in diesem Zusammenhang bedeutet. Wir betrachten zwei Möglichkeiten, um die Größe der Kantengewichte zu beurteilen:

1. Das Gewicht einer aufzutrennenden Kantenmenge ist klein, wenn das Ignorieren dieser Kanten zu einem modifizierte Newton-Verfahren führt, zu dessen Lösung nur geringfügig mehr Newtonschritte nötig sind als für das volle System. Als Indikator dient die Kontraktionszahl C_{lin} aus (3.6). Kanten werden aufgetrennt, solange

   $$C_{lin} \;\leq\; C_{lin}^{max} \tag{4.3}$$

   mit einer vorgegebenen Schranke $C_{lin}^{max} \geq 0$.

2. Das Gewicht einer aufzutrennenden Kantenmenge ist klein, wenn es einen gegebenen Anteil des Gesamtgewichts von $\widetilde{A}$ nicht überschreitet, d.h. wenn für alle $i, j = 1, \ldots, N_S$

$$\left(\widetilde{A}_{trenn}\right)_{ij} \;\leq\; \epsilon_B^{rel} \;\left|\widetilde{A}\right| \tag{4.4}$$

mit vorgegebenem $\epsilon_B^{rel} \geq 0$ gilt. In $\widetilde{A}_{trenn}$ sind dabei nur die Elemente aus $\widetilde{A}$ eingetragen, die vernachlässigt werden sollen. $\left|\widetilde{A}\right|$ bezeichnet das Gesamtgewicht des Graphen $\widetilde{A}$, d.h.

$$\left|\widetilde{A}\right| \;:=\; \sum_{i=1}^{N_S} \sum_{j=i}^{N_S} \left|\widetilde{A}_{ij}\right| .$$

Im Falle ihrer Anwendbarkeit ist die erste Möglichkeit „exakter“ als die zweite. So kann man z.B. Kanten auftrennen, solange

$$C_{lin} \;<\; 1 \cdot 10^{-5}$$

erfüllt ist. Schon nach wenigen zusätzlichen Newtonschritten wird der Fehler gemäß (3.7) das Abbruchkriterium für das Newton-Verfahren erfüllen.
Allerdings ist diese Möglichkeit recht aufwändig und es gelten die oben beschriebenen Einschränkungen im nichtlinearen Fall (siehe Abschnitt 3.2). Für jede mögliche Aufteilung, die der Algorithmus von Stoer und Wagner vorschlägt, müssen zudem die Matrizen J_d und J_R neu bestimmt und die beiden Singulärwerte $\sigma_{max}(J_R)$ und $\sigma_{min}(J_d)$ neu berechnet werden.
Die Singulärwerte können in *M++* mit dem Von-Mises-Verfahren [19] ermittelt werden. Dabei handelt es sich um ein iteratives Verfahren, dass einen Näherungswert von σ_{min} bzw. σ_{max} in sehr wenigen Schritten $(1-3)$ findet. Der Hauptaufwand bei der Berechnung von C_{lin} steckt deshalb in der Ermittlung der Blöcke und dem Kopieren von J in J_d und J_R.

Die zweite Möglichkeit ist demgegenüber sehr schnell zu berechnen und führte in Testsimulationen ebenfalls zu akzeptablen Ergebnissen. Allerdings besteht hier die Gefahr eines nichtkonvergenten Verfahrens. Man wird deshalb die obere Schranke ϵ_B^{rel} vorsichtig wählen und lieber ein gewisses Einsparpotential durch eine mögliche weitere Aufteilung ungenutzt lassen.
Es besteht auch die Möglichkeit, zur adaptiven Bestimmung von ϵ_B^{rel} eine ähnliche Vorgehensweise wie beim zeitadaptiven Verfahren aus Abschnitt 4.1 zu verwenden. Vor dem Start der Simulation sind ϵ_B^{rel} sowie Schranken $\epsilon_{B,min}^{rel}$ und $\epsilon_{B,max}^{rel}$ vorzugeben. Als Indikator für die Veränderung von ϵ_B^{rel} dient wieder die Anzahl der Newtonschritte NS.

Sollte sie unter einem gegebenen NS^B_{min} liegen, wird ϵ^{rel}_B mit dem Faktor $fact^B_1 > 1$ multipliziert. Falls sie NS^B_{max} überschreitet, erfolgt eine Multiplikation mit $fact^B_2 < 1$. Dabei soll ϵ^{rel}_B die Schranken $\epsilon^{rel}_{B,min}$ nicht unter- und $\epsilon^{rel}_{B,max}$ nicht überschreiten.

Bemerkung 4.3. Wir verwenden hier bewusst das oben angegebene Kriterium (4.4) und nicht etwa das auch mögliche

$$\left|\widetilde{A}_{trenn}\right| \;\leq\; \epsilon^{rel}_B \;\left|\widetilde{A}\right|.$$

Diese Wahl ist mit der Beobachtung aus Abschnitt 3.2.2 (Tabelle 3.5) begründet. Dort wurde festgestellt, dass die Anzahl der vernachlässigten schwachen Verbindungen keine Rolle spielt, sondern nur deren maximale Stärke.

Aufgrund der beiden oben beschriebenen Einschränkungen der Verfahren mit den Abbruchkriterien (4.3) bzw. (4.4) (hohe Rechenzeit bzw. fehlende Exaktheit) empfiehlt sich im Falle der Anwendbarkeit von (4.3) eine Kombination der beiden Kriterien:
Die vom Algorithmus von Stoer und Wagner vorgeschlagene Kantenmenge wird aufgetrennt, falls eine der folgenden beiden Bedingungen für alle $i, j = 1, \ldots, N_S$ erfüllt ist:

$$\left(\widetilde{A}_{trenn}\right)_{ij} \;\leq\; \epsilon^{rel}_{B,1}\left|\widetilde{A}\right| \qquad \text{oder} \tag{4.5}$$

$$\epsilon^{rel}_{B,1}\left|\widetilde{A}\right| \;<\; \left(\widetilde{A}_{trenn}\right)_{ij} \;<\; \epsilon^{rel}_{B,2}\left|\widetilde{A}\right| \qquad \text{und} \qquad C_{lin} \;\leq\; C^{max}_{lin}. \tag{4.6}$$

Sehr schwache Verbindungen mit (4.5) werden hier ohne Prüfung des Konvergenzkriteriums C_{lin} vernachlässigt, sehr starke mit

$$\left(\widetilde{A}_{trenn}\right)_{ij} \;\geq\; \epsilon^{rel}_{B,2}\left|\widetilde{A}\right| \qquad \forall i, j = 1, \ldots, N_S$$

bleiben erhalten. Die zeitaufwändige Überprüfung von C_{lin} erfolgt nur, falls das Gewicht der maximal aufgetrennten Kante im Intervall aus Gleichung (4.6) liegt.

Bei geeigneter Wahl der beiden Schranken $\epsilon^{rel}_{B,1}$ und $\epsilon^{rel}_{B,2}$ kombiniert diese Vorgehensweise die Vorteile der Kriterien (4.3) und (4.4). Verglichen mit der ausschließlichen Verwendung von (4.3) entsteht eine sehr effiziente Entkopplungsstrategie, bei der die Konsistenz des Verfahrens immer noch gewährleistet bleibt.

Zusammenfassung 4.4. Zusammenfassend lautet die automatische Entkopplungsstrategie zur Erkennung schwacher Verbindungen wie folgt:

1. Bestimme die Adjazenzmatrix $\widetilde{A}$ des repräsentierenden Reaktionsgraphen $\widetilde{\mathcal{R}}$ gemäß (4.1) und (4.2).

2. Bilde Teilsysteme durch Aufteilung des Graphen $\widetilde{\mathcal{R}}$ mit dem Algorithmus von Stoer und Wagner aus 4.2.1. Dabei wird der Graph partitioniert, solange nur Kanten aufgetrennt werden, die

   (a) zu einer Kontraktionszahl C_{lin} mit

   $$C_{lin} \;\leq\; C_{lin}^{max}$$

   führen mit einer vorgegebenen Schranke $C_{lin}^{max} \geq 0$ oder

   (b) zu einer Matrix $\widetilde{A}_{trenn}$ führen mit

   $$\left(\widetilde{A}_{trenn}\right)_{ij} \;\leq\; \epsilon_B^{rel} \left|\widetilde{A}\right| \qquad \forall i,j = 1,\ldots,N_S$$

   mit vorgegebenem $\epsilon_B^{rel} \geq 0$.

Die beiden Kriterien können durch Anwendung von (a) im Intervall (4.6) und (b) in (4.5) kombiniert werden.

4.2.3 Beispielsimulationen

Wir betrachten wieder die schon in Abschnitt 3.3 verwendete Simulation aus Beispiel 3.7 mit der kleinen Zeitschrittweite $\Delta t = 0,01\ s$. Zwar kann hier kein großer Rechenzeitgewinn erzielt werden, die Strategie lässt sich aber gut analysieren. Um die Funktionsfähigkeit der in 4.2.2 entwickelten automatischen Entkopplungsstrategie besser testen zu können, werden variable Damköhlerzahlen verwendet. Der zeitliche Verlauf der Da ist für die sechs Reaktionen in Tabelle 4.4 angegeben.

Tabelle 4.4: Variation der Damköhlerzahlen für die Testsimulation im Intervall $0 \leq t < 15\ s$. Für $t \geq 15\ s$ wird die Tabelle periodisch fortgesetzt.

Zeitintervall $[s]$	$Da\,(R_1)$	$Da\,(R_2)$	$Da\,(R_3)$	$Da\,(R_4)$	$Da\,(R_5)$	$Da\,(R_6)$
0 - 1	0	0	0	0	0	0
1 - 2	1000	0	0	0	0	0
2 - 3	1000	0	0	1000	0	0
3 - 4	1000	1	0	1000	0	0
4 - 5	1000	1	0	1000	1	0
5 - 6	1000	1	1000	1000	1	0
6 - 10	1000	1	1000	1000	1	1000
10 - 11	0	1	1000	1000	1	1000
11 - 12	0	1	1000	0	1	1000
12 - 13	0	0	1000	0	1	1000
13 - 14	0	0	1000	0	0	1000
14 - 15	0	0	0	0	0	1000

In den Simulationen werden nur (fast) verschwindende Kopplungen gemäß Kriterium (4.4) aufgetrennt. Als Schranke für die Aufteilung wurde

$$\epsilon_B^{rel} := 1 \cdot 10^{-8}$$

gewählt. Eine Anwendung des Kriteriums (4.3) mit

$$C_{lin}^{max} := 1 \cdot 10^{-5} \tag{4.7}$$

liefert die gleiche Aufteilung des linearen Gleichungssystems, verursacht jedoch deutlich längere Rechenzeiten.

Tabelle 4.5 zeigt die Performance bei Verwendung von *BiCGStab* mit einem symmetrischen Gauß-Seidel-Verfahren (ein Schritt) als Vorkonditionierer. Dabei wurden das volle Newton-Verfahren ohne Aufteilung sowie das modifizierte mit automatischer Aufteilung wie oben beschrieben und teilweiser Assemblierung der Jacobi-Matrix verwendet.

Tabelle 4.5: Performance für akademisches Testbeispiel mit variablen Damköhlerzahlen aus Tabelle 4.4 und automatischer Entkopplung mit den Bezeichnungen aus Tabelle 3.8.

Löser	Konfiguration	Assemblierung	NS/TS	Ass	LS	Ges
BiCGStab	$(A \dots L)$	vollständig	$2,9$	295	123	442
BiCGStab	automatisch	J_d	$2,9$	336	71	422

Die Assemblierungszeit steigt um 41 Minuten. Davon entfallen ca. 20 Minuten auf das Kopieren der Gesamtmatrix in die Teilmatrizen und 21 Minuten auf das Finden der Teilsysteme mit der oben beschriebenen Strategie. Das entspricht etwa $5,2$ % der Rechenzeit.
Die Zeit für den linearen Löser verringert sich bei Aufteilung auf ca. 59 % des ursprünglichen Wertes. Bei den Beispielsimulationen in Abschnitt 3.3 war diese Einsparung teilweise deutlich höher, jedoch wurden dort auch Reaktionen mit Damköhlerzahlen $Da = 1$ vernachlässigt.
In diesem Beispiel wäre es durch Wahl einer höheren Schranke ϵ_B auch möglich, die schwachen Reaktionen R_2 und R_5 immer aufzutrennen. Allerdings hätte das eine Erhöhung der Anzahl der Newtonschritte pro Zeitschritt vom $3,0$ auf $3,9$ zur Folge. Wegen des stark ansteigenden Assemblierungsaufwandes würde das entkoppelte Verfahren ähnlich wie in den Beispielen aus Tabelle 3.10 eine deutlich höhere Gesamtzeit benötigen als das ursprüngliche.
Selbst bei der vorsichtigen Auftrennung kann nur ein geringfügiger Gewinn von etwa $4,3$ % erzielt werden. Ursache ist auch hier die bereits beschriebene Dominanz der Transportterme in der Assemblierung.

Die Wirkungsweise der automatischen Entkopplungsstrategie kann an Abbildung 4.4 abgelesen werden. In der ersten Sekunde der Simulation finden keine Reaktionen statt, so dass für jede Spezies ein eigenes Teilsystem gelöst werden kann ($N_T = 12$). Im Folgenden werden immer mehr Reaktionen zugeschaltet (vergleiche Tabelle 4.4) und die Anzahl der Teilsysteme sinkt ab $t = 6\ s$ auf $N_T = 1$. Es wird also das volle System gelöst.

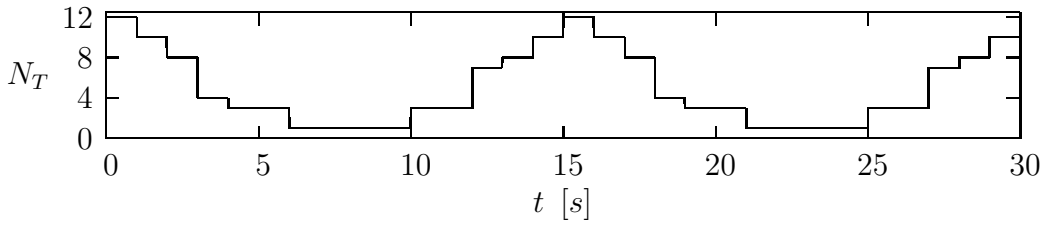

Abbildung 4.4: Anzahl der Teilsysteme N_T für die Simulation mit Damköhler-Zahlen aus Tabelle 4.4.

Ab $t = 10\ s$ werden Reaktionen deaktiviert und die Anzahl der Teilsysteme steigt wieder auf $N_T = 12$. Für $t \geq 15\ s$ wiederholt sich dieses Verhalten.

Kapitel 5

Praxisorientierte Simulationen

Im abschließenden fünften Kapitel werden zwei praxisorientierte Simulationen durchgeführt. Mit ihrer Hilfe soll das in dieser Arbeit vorgestellte Verfahren an relevanten Beispielen getestet werden.

Beim ersten Beispiel, dem EDTA-Problem, handelt es sich um eine Simulation ohne Fluss. Wir ermitteln die Lösung des reaktiven Modells im Zweidimensionalen und vergleichen die Performance des Entkopplungsverfahrens mit und ohne Zeitadaption nochmals mit den Ergebnissen aus [47], wo die gleiche Simulation bereits im Eindimensionalen durchgeführt wurde.

Beim nachfolgenden Barriere-Problem wird das Verfahren auf eine echte zweidimensionale Simulation angewandt. Hier werden die Ergebnisse geplottet und die Performance des Entkopplungsverfahrens für verschiedene Parameter dokumentiert.

5.1 Das EDTA-Problem

In [59] wurde auf Basis der Artikel [15] und [25] ein Problem vorgestellt, dass Reaktionen mit dem Stoff Ethylendiamintetraessigsäure ($EDTA$, Summenformel $C_{10}H_{16}N_2O_8$) im Boden modelliert. $EDTA$ dient als Reinigungs- und Bleichmittel und wird auch als Konservierungsmittel eingesetzt.
Der Stoff ist für Menschen unbedenklich. Falls er jedoch über Abwässer in die Umwelt gelangt, kann er dort große Schäden verursachen. So entstehen in kontaminierten Böden und im Grundwasser vermehrt Mikroorganismen, die sich von den Abbauprodukten Kohlenstoff und Stickstoff ernähren, wodurch die natürliche Biodiversität des Mediums nachhaltig gestört wird.
Das von $EDTA$ ausgehende Hauptrisiko für Mensch und Umwelt besteht jedoch in seiner Eigenschaft als Komplexbildner. Der Stoff geht sehr stabile Bindungen mit mindestens $2-$fach positiv geladenen Kationen ein und bildet sogenannte Metallkomplexe.

Aufgrund dieser Eigenschaft ist $EDTA$ in der Lage, fest im Boden gebundene Schwermetallsalze zu lösen und in das Grundwasser zu transportieren. Dort zerfällt das $EDTA$ und setzt die hochgiftigen Schwermetalle wieder frei. Eine Vorhersage der Boden- und Grundwasserverschmutzung durch $EDTA$ ist deshalb von großem Interesse.

5.1.1 Problembeschreibung

Betrachtet wird der in [15] und [25] beschriebene Abbau des Metallkomplexes $Co\,(II)\,EDTA\,(aq)$ mit Hilfe von positiv und negativ geladenen Schwefelionen. Das $EDTA$ hat sich hier mit Cobalt-Ionen gebunden und diese aus dem Boden gelöst. Co wird z.B. in der Rüstungsindustrie zur Erhöhung der Wärme- und Verschleissfestigkeit von legierten Stählen verwendet. Rückstände dieses Stoffes werden deshalb oft in militärisch genutzten Gebieten gefunden.

Zur Simulation der Reaktionen gehen wir von einem Modell mit 14 Spezies und 10 Reaktionen aus, die aus [15] und [25] entnommen wurden:

$$
\begin{aligned}
R_1: &\quad Co\,(II)\,(aq) + S_{neg} &&\rightleftharpoons\ S_{neg\_}Co,\\
R_2: &\quad Co\,(II)\,EDTA\,(aq) + S_{pos} &&\rightleftharpoons\ S_{pos\_}Co\,(II)\,EDTA,\\
R_3: &\quad Fe\,(III)\,EDTA\,(aq) + S_{pos} &&\rightleftharpoons\ S_{pos\_}Fe\,(III)\,EDTA\,(aq)\,,\\
R_4: &\quad EDTA\,(aq) + S_{pos} &&\rightleftharpoons\ S_{pos\_}EDTA,\\
R_5: &\quad Co\,(III)\,EDTA\,(aq) + S_{pos} &&\rightleftharpoons\ S_{pos\_}Co\,(III)\,EDTA,\\
R_6: &\quad S_{pos\_}Co\,(II)\,EDTA &&\leftrightarrow\ Co\,(II)\,(aq) + S_{pos\_}EDTA,\\
R_7: &\quad S_{pos\_}EDTA &&\rightarrow\ Fe\,(III)\_\,EDTA\,(aq) + S_{pos},\\
R_8: &\quad Co\,(II)\,EDTA\,(aq) &&\rightarrow\ Co\,(III)\,EDTA\,(aq)\,,\\
R_9: &\quad Fe\,(III)\_\,EDTA\,(aq) + 6O_2 &&\rightarrow\ 3CO_2 + \text{Biomasse},\\
R_{10}: &\quad EDTA\,(aq) + 6O_2 &&\rightarrow\ 3CO_2 + \text{Biomasse}.
\end{aligned}
$$

Die Schreibweise der Pfeile entspricht der aus [47]: $\rightleftharpoons$ repräsentiert schnelle kinetische Reaktionen, $\leftrightarrow$ langsame kinetische Reaktionen und $\rightarrow$ Abbaureaktionen, wobei R_7 und R_8 kinetische Reaktionen sind, R_9 und R_{10} unumkehrbare Monod-Reaktionen.
Die Parameter der zehn Reaktionen R_1 bis R_{10} werden gemäß Tabelle 5.1 gesetzt. Es findet kein Fluss statt, d.h. $D := 0$, $\mathbf{q} := 0$ und es gelten die Anfangswerte aus Tabelle 5.2.

Abbildung 5.1 zeigt das Reaktionsnetzwerk R für das $EDTA-$Problem. Zusammen mit den Reaktionsgleichungen R_1 bis R_{10} ist zu erkennen, dass den Schwefelionen S_{neg} (I) und S_{pos} (K) eine zentrale Bedeutung zukommt.
Die Reaktion R_1 mit S_{neg} läuft nach Tabelle 5.1 ebenso wie die Reaktionen R_2 bis R_5 mit S_{pos} sehr schnell ab, während R_6 bis R_8 über deutlich kleinere Reaktionsraten verfügen. Das Reaktionsnetzwerk sollte also auch hier zumindest zeitweise

Tabelle 5.1: Reaktionsparameter für das $EDTA-$Problem. Die Vorwärts- und Rückwärtsraten k^f bzw. k^b werden ebenso wie μ_{max} in $1/h$ angegeben, K_{M_1} und K_{M_2} in mM/l.

Reaktion	Parameter
R_1	$k^f = 12000, \quad k^b = 1000,$
R_2	$k^f = 25000, \quad k^b = 1000,$
R_3	$k^f = 9000, \quad k^b = 1000,$
R_4	$k^f = 25000, \quad k^b = 1000,$
R_5	$k^f = 2.500, \quad k^b = 1000,$
R_6	$k^f = 1, \quad k^b = 0,001,$
R_7	$k^f = 2,5, \quad k^b = 0,$
R_8	$k^f = 0,001, \quad k^b = 0,$
R_9	$\mu_{max} = 2,5 \cdot 10^{-4}$, $K_{M_1} = 1,0 \cdot 10^{-5}$, $K_{M_2} = 1,0 \cdot 10^{-5}$,
R_{10}	$\mu_{max} = 2,5 \cdot 10^{-2}$, $K_{M_1} = 1,0 \cdot 10^{-5}$, $K_{M_2} = 1,0 \cdot 10^{-5}$.

Tabelle 5.2: Übersicht über Spezies und Anfangswerte für das $EDTA$-Problem.

Nr.	Spezies	Kürzel	mob/imob	Anfangswert
1	$Co\,(II)\,(aq)$	A	mobil	0
2	$Co\,(II)\,EDTA\,(aq)$	B	mobil	$0,032$
3	$Fe\,(III)\,EDTA\,(aq)$	C	mobil	0
4	$EDTA\,(aq)$	D	mobil	0
5	$Co\,(III)\,EDTA\,(aq)$	E	mobil	0
6	O_2	F	mobil	$0,256$
7	CO_2	G	mobil	0
8	$S_{neg-}Co$	H	immobil	0
9	S_{neg}	I	immobil	$0,0011$
10	$S_{pos-}Co\,(II)\,EDTA$	J	immobil	0
11	S_{pos}	K	immobil	$0,016$
12	$S_{pos-}Fe\,(III)\,EDTA\,(aq)$	L	mobil	0
13	$S_{pos-}EDTA$	M	immobil	0
14	$S_{pos-}Co\,(III)\,EDTA$	N	immobil	0
15	$Biomasse$	O	immobil	$0,02$

in mehrere Teilsysteme zerfallen, wobei die Aufteilung des Gesamtsystems anders als in Beispiel 3.7 nicht ohne weiteres vorhergesagt werden kann.

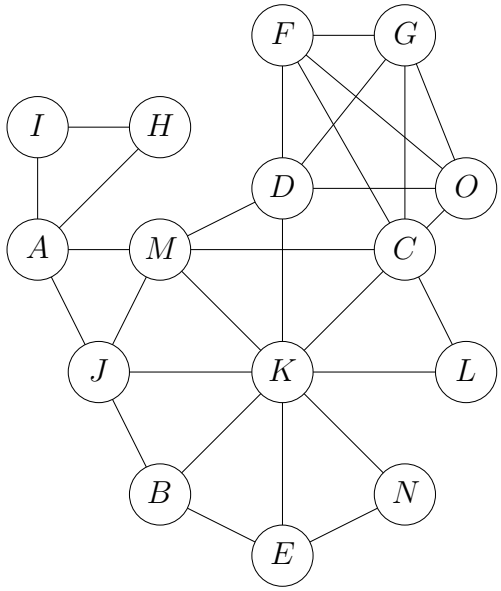

Abbildung 5.1: Reaktionsnetzwerk $\mathcal{R}$ für das $EDTA-$Problem.

5.1.2 Performance des Verfahrens

Tabelle 5.3 dokumentiert die Performance des Verfahrens bei der Lösung des EDTA-Problems. Dabei wurden in Übereinstimmung mit der Dissertation [47, 6.2.2] die Berechnungen bis zum Endzeitpunkt $T = 50\ h$ durchgeführt. Auf eine Simulation bis $T = 2300\ h$ wurde aufgrund der hohen Rechenzeiten verzichtet. Die Zeitschrittweite Δt wurde konstant mit

$$\Delta t \ := \ 0,001\ h$$

und adaptiv nach dem Verfahren aus 4.1 mit

$$\begin{array}{lll} \Delta t_{min} := 0,001\ h, & NS_{min} := 3, & fact_1 := 2,0, \\ \Delta t_{max} := 0,05\ h, & NS_{max} := 10, & fact_2 := 0,5, \end{array}$$

gewählt.
Das lineare Gleichungssystem wurde für alle Zeitschrittweiten voll gelöst (ohne Aufteilung) und mit automatischer Aufteilung wie in 4.2.2 beschrieben mit den beiden durch die oberen Schranken

$$\begin{array}{rcl} C_{lin,1}^{max} & := & 1 \cdot 10^{-4}, \\ C_{lin,2}^{max} & := & 1 \cdot 10^{-5}, \\ \epsilon_B^{rel} & := & 5 \cdot 10^{-8}, \end{array}$$

gegebenen Entkopplungsstrategien.
Die sehr kleine Wahl von ϵ_B^{rel} bewirkt, dass nur Verbindungen zwischen Spezies getrennt werden, die ohnehin annähernd wegfallen. Damit soll verhindert werden, dass bei der zu erwartenden Dominanz der Assemblierungszeit gegenüber der Zeit für den linearen Löser ein zu starker Anstieg der Assemblierungszeit zu einer Verschlechterung der Gesamtperformance führt.

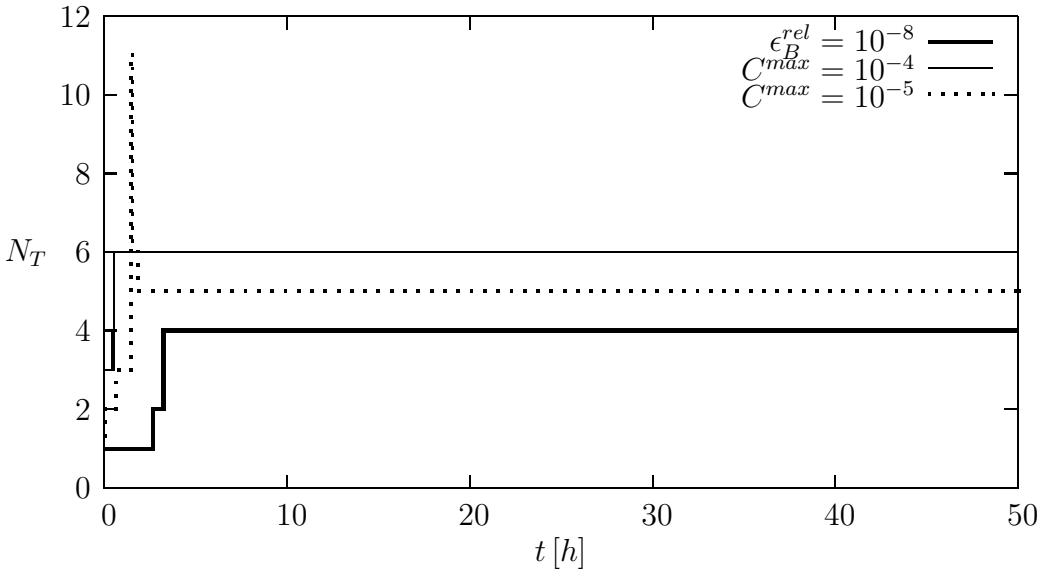

Abbildung 5.2: Anzahl der Teilsysteme N_T in Abhängigkeit von der Zeit für das EDTA-Problem bei fester Zeitschrittweite mit $\Delta t = 0,001\ h$ mit unterschiedlicher Aufteilungsstrategie.

Abbildung 5.2 visualisiert die Anzahl der Teilsysteme N_T in Abhängigkeit von der Zeit. Dabei zeigt sich, dass für $\Delta t = 0,001\ h$ und $C^{max} = 1 \cdot 10^{-4}$ während des größten Teils der Simulation das lineare Gleichungssystem in sechs Teilsysteme aufgeteilt wird. Diese lauten

$$(A)\,(HI)\,(BCDEJKLMN)\,(F)\,(G)\,(O)\,.$$

Hier wurden R_9 und R_{10} und teilweise R_1 vernachlässigt, wobei die Bindung zwischen den Spezies H und I ($S_{neg-}Co$ und S_{neg}) von R_1 erhalten bleibt. Die automatische Entkopplungsstrategie teilt also nicht exakt nach Reaktionen auf.
Die zweite Kurve zeigt N_T für $\Delta t = 0,001\ h$ und $C^{max} = 1 \cdot 10^{-5}$. Sie führt zu einer Aufteilung in die fünf Teilsysteme

$$(AHI)\,(BCDEJKLMN)\,(F)\,(G)\,(O)\,.$$

Verglichen mit oben bleibt die volle Reaktion R_1 (AHI) erhalten. Die Verringerung der Schranke C^{max} von $1 \cdot 10^{-4}$ auf $1 \cdot 10^{-5}$ führt also zu einer weniger starken Aufteilung.
Für die letzte Kurve wurde schließlich die einfache Entkopplungsstrategie mit $\epsilon_B^{rel} = 1 \cdot 10^{-8}$ verwendet. Das Gleichungssystem wird in die vier Teilsysteme

$$(ABCDEHIJKLMN)\,(F)\,(G)\,(O)$$

partitioniert. Es werden nur die beiden Monod-Reaktionen R_9 und R_{10} vernachlässigt.

Abbildung 5.3 visualisiert den Verlauf von N_T unter Verwendung der Schrittweitensteuerung. Dabei wird die maximale Zeitschrittweite bereits nach weniger als 10 s angenommen.

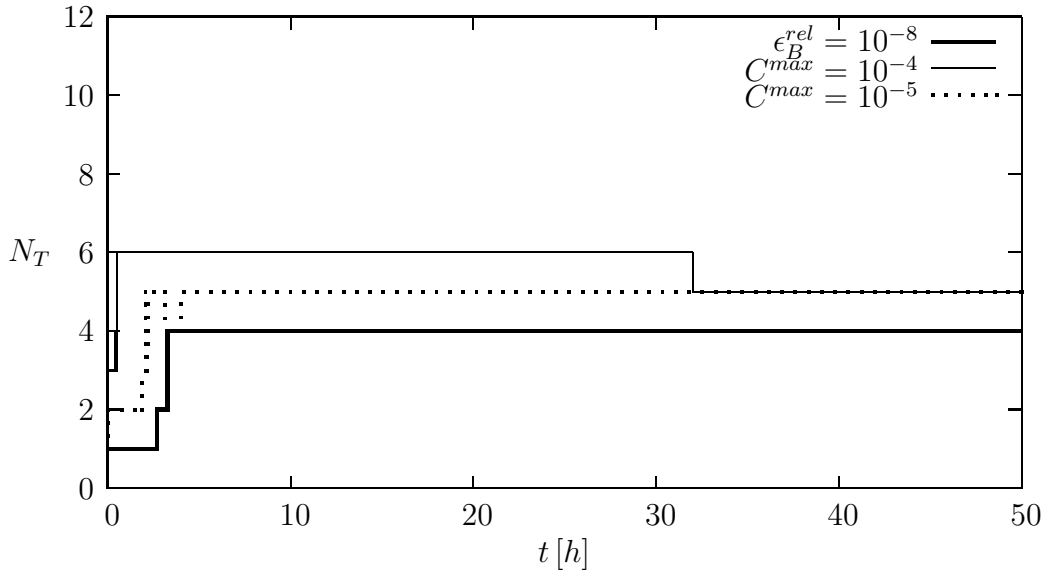

Abbildung 5.3: Anzahl der Teilsysteme N_T in Abhängigkeit von der Zeit für das EDTA-Problem bei adaptiver Zeitschrittweite mit unterschiedlicher Aufteilungsstrategie.

Die Kurve für ϵ_B^{rel} ist identisch mit der aus Abbildung 5.2 (feste Zeitschrittweite), wohingegen die für C_1^{max} und C_2^{max} einen etwas anderen Verlauf nehmen. Wichtigster Unterschied ist die Verringerung der Anzahl der Teilsysteme N_T von 6 auf 5 bei $t \approx 31$ h. Das größere Δt führt folglich dazu, dass nicht mehr ganz so stark aufgeteilt werden kann.
Die beiden Abbildungen 5.2 und 5.3 zeigen auch, dass die Strategie (4.3), bei der C_{lin} als Indikator dient, auf das nichtlineare EDTA-Beispiel angewandt werden

kann. Es entstehen sinnvolle Aufteilungen und auch die Anzahl der Teilsysteme scheint im Vergleich zur Strategie (4.4) mit ϵ_B^{rel} plausibel.

Nach der Analyse der Blockbildungen wollen wir nun die Performance der einzelnen Strategien betrachten. Diese ist in der bereits bekannten Form in Tabelle 5.3 dokumentiert.

Tabelle 5.3: Performance für das EDTA-Beispiel mit der Anzahl von Zeitschritten TS, Zeit für die automatische Ermittlung der Aufteilung des Gesamtsystems *Auto* und den übrigen Bezeichnungen aus Tabelle 3.8.

$\Delta t\,[h]$	Aufteilungsstrategie	TS	NS/TS	*Ass*	*Auto*	*LS*	*Ges*
$0,001$	–	50.000	$1,29$	186	–	76	284
$0,001$	$\epsilon_B^{rel} = 1 \cdot 10^{-8}$	50.000	$2,29$	198	19	41	279
$0,001$	$C^{max} = 1 \cdot 10^{-4}$	50.000	$3,91$	466	992	37	1.518
$0,001$	$C^{max} = 1 \cdot 10^{-5}$	50.000	$2,57$	260	681	39	1.001
adaptiv	–	1.009	$1,63$	5	–	2	8
adaptiv	$\epsilon_B^{rel} = 1 \cdot 10^{-8}$	1.009	$2,63$	5	1	1	8
adaptiv	$C^{max} = 1 \cdot 10^{-4}$	31.973	$5,61$	461	633	30	1145
adaptiv	$C^{max} = 1 \cdot 10^{-5}$	1.182	$5,43$	14	16	3	39

Der Vergleich der oberen und unteren Hälfte der Tabelle zeigt sofort die große Zeitersparnis des zeitadaptiven Verfahrens. Die Simulation muss zwar aus Konvergenzgründen mit der niedrigen Schrittweite $\Delta t = 0,001\ h$ gestartet werden, kann aber wie bereits erwähnt sehr schnell auf den maximal zulässigen Wert $\Delta t_{max} = 0,05\ h$ erhöht werden. So ergibt sich eine Beschleunigung um das bis zu $35-$fache.

Die Aufteilung des Gleichungssystems hat bei Verwendung von ϵ_B^{rel} wie erwartet keinen großen Einfluss auf die Rechenzeit. Die Ursache für den geringen Gewinn durch das Entkopplungsverfahren ist wie schon in den vorherigen Beispielen die geringe Anzahl von Iterationsschritten des verwendeten iterativen linearen Lösers (*BiCGStab* mit symmetrischem Gauß-Seidel-Verfahren als Vorkonditionierer). Die Konditionszahl der vollbesetzten Jacobi-Matrix liegt bei

$$\kappa\,(J) \;\approx\; 16.$$

BiCGStab benötigt daher weniger als fünf Schritte zur Lösung des linearen Gleichungssystems, wodurch die Dominanz der Assemblierungszeiten zu erklären ist.

Die zweite Entkopplungsstrategie mit C_{lin} verursacht hingegen sehr viel höhere Rechenzeiten. Obwohl sie theoretisch abgesichert nur auf lineare Modelle anwendbar ist, liefert sie für das nichtlineare $EDTA-$Problem brauchbare Aufteilungen des Gesamtsystems.

5.2 Das Barriere-Problem

1997 wurde von Bannett [3] ein Modell entwickelt, mit dessen Hilfe die Reinigung des Grundwassers von Trichlorethylen (TCE) und sechswertigem Chrom sowie deren umweltschädlichen Abbauprodukten simuliert werden kann. Anlass für diese Bemühungen war eine Kontamination des Grundwassers auf einem ehemaligen Flughafengelände in Elisabeth City im US-Bundesstaat North Carolina. Dort waren in einer Werkhalle, in der Metallpanzerungen hergestellt wurden, die genannten Schadstoffe in großen Mengen ausgetreten und drohten den nahe gelegenen Pasquotank River zu verschmutzen.
Bannett schlug vor, zwischen Werkhalle und Fluss eine 60 cm dicke reaktive Barriere aus Eisengranulat ($Fe^0\,(s)$) zu installieren. Das kontaminierte Grundwasser fließt durch die Barriere, die Schadstoffe reagieren mit dem Eisengranulat zu unbedenklichen Stoffen und gelangen schließlich in den Fluss.
Entsprechende numerische Simulationen wurden bereits von Mayer [44, 1999] und Blowes und Mayer [7, 1999] durchgeführt. Grundlage für die folgenden Berechnungen bildet der Artikel [45, 2001].

5.2.1 Problembeschreibung

Wir betrachten einen vertikalen zweidimensionalen Schnitt durch den Grundwasserleiter, der sich in einer Tiefe von -4 bis -7 Metern befindet und von der Werkhalle bis zum Fluss reicht. Am linken Rand befindet sich die Kontaminationsstelle, am rechten der Pasquotank River und zwischen den beiden die reaktive Barriere (siehe Abbildung 5.4).

Die schraffiert gezeichnete Barriere ($1,8\,m \leq x_1 \leq 2,4\,m$) besteht aus drei Teilen, die sich durch ihre hydraulische Leitfähigkeit und Porosität unterscheiden:

$$K := \begin{cases} 1,2 \cdot 10^{-6}\ \frac{m}{s} & \text{für } -5,5\ m \leq x_2 < -4,0\ m, \\ 1,2 \cdot 10^{-3}\ \frac{m}{s} & \text{für } -6,5\ m \leq x_2 < -5,5\ m, \\ 1,2 \cdot 10^{-6}\ \frac{m}{s} & \text{für } -7,0\ m \leq x_2 < -6.5\ m. \end{cases}$$

$$\Theta := \begin{cases} 0,5 & \text{für } -6,5\ m \leq x_2 < -4,0\ m, \\ 0,38 & \text{für } -7,0\ m \leq x_2 < -6,5\ m. \end{cases}$$

Vor und hinter der Barriere gilt

$$\begin{aligned} K &:= 1,2 \cdot 10^{-3}\ \frac{m}{s}, \\ \Theta &:= 0,38. \end{aligned}$$

Die Fluss-Randbedingungen werden am linken Rand durch einen durchschnittlichen hydraulischen Gradienten von $0,0022$ gesetzt. Am rechten Rand gelten

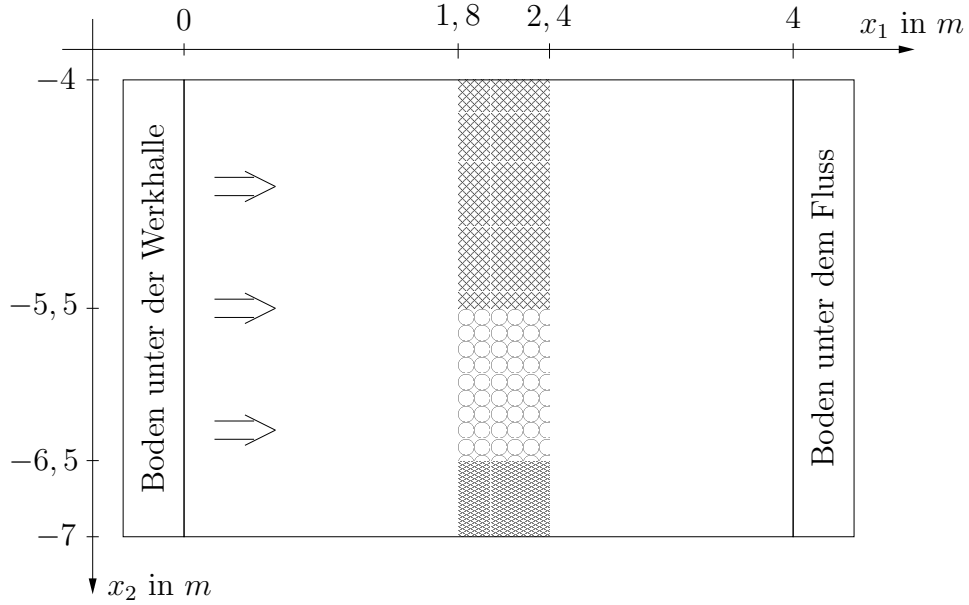

Abbildung 5.4: Gebiet für das Barriere-Problem.

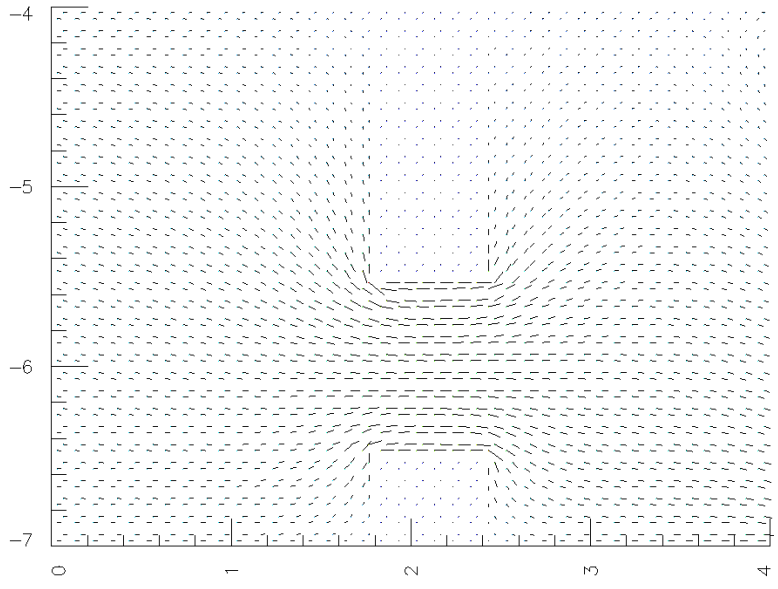

Abbildung 5.5: Fließfeld des Barriere-Problems.

homogene Neumann-Randbedingungen, am oberen und unteren homogene Fluss-Randbedingungen. Es stellt sich das Fließfeld aus Abbildung 5.5 ein.

Zur Simulation der Reaktionen wird ein Modell mit 21 Spezies und 10 Reaktionen gemäß Tabelle 5.4 verwendet.

Tabelle 5.4: Reaktionen für das Barriere-Problem.

Nr	Reaktion
1	$Fe^0\,(s) + CrO_4^{2-} + 6H^+ \rightarrow Fe^{3+} + Cr\,(OH)_2^+ + 2H_2O$
2	$Fe^0\,(s) + 0,3025C_2HCL_3 \rightarrow Fe^{2+} + 0,07C_2H_2Cl_2 + 1,2325H^+ + 0,2325C_2H_6 + 0,7675Cl^-$
3	$Fe^0\,(s) + C_2H_2Cl_2 + H^+ \rightarrow Fe^{2+} + C_2H_3Cl + Cl^-$
4	$Fe^0\,(s) + \frac{1}{2}C_2H_3Cl + \frac{3}{2}H^+ \rightarrow Fe^{2+} + \frac{1}{2}C_2H_6 + \frac{1}{2}Cl^-$
5	$Fe^0\,(s) + \frac{1}{4}O_2\,(aq) + 3H^+ \rightarrow Fe^{3+} + \frac{1}{2}H_2O + H_2\,(aq)$
6	$Fe^0\,(s) + \frac{3}{8}NO_3^- + \frac{15}{4}H^+ \rightarrow Fe^{3+} + \frac{3}{8}NH_4^+ + \frac{9}{8}H_2O$
7	$Fe^0\,(s) + \frac{1}{4}SO_4^{2-} + \frac{9}{4}H^+ \rightarrow Fe^{2+} + \frac{1}{4}HS^- + H_2O$
8	$Fe^0\,(s) + \frac{1}{2}CH_2O + 2H^+ \rightarrow Fe^{2+} + \frac{1}{2}CH_4\,(aq) + \frac{1}{2}H_2O$
9	$Fe^0\,(s) + \frac{1}{4}CO_3^{2-} + \frac{5}{2}H^+ \rightarrow Fe^{2+} + \frac{1}{4}CH_4\,(aq) + \frac{3}{4}H_2O$
10	$Fe^0\,(s) + 2H^+ \rightarrow Fe^{2+} + H_2\,(aq)$

In [45] wurde auf Diffusion und Dispersion verzichtet. Aus Konvergenzgründen setzen wir im Tensor (2.4)

$$\beta_l \;:=\; 0,02.$$

Diese Wahl beeinflusst lediglich die Diffusion in Fluss-Richtung. Die Simulationsergebnisse werden dadurch kaum verfälscht. Ohne diese zusätzliche Diffusion würden bei den Berechnungen mit den verwendeten konformen Finiten Elementen starke Oszillationen entstehen. Durch die obige Wahl von β_l werden diese gerade vermieden.

Tabelle 5.5: Ratenkonstanten k und Damköhler-Zahlen Da für die zehn Reaktionen aus Tabelle 5.4.

Nr	$\log k$	Da
1	$6,0$	$4,00 \cdot 10^{12}$
2	$-3,1$	$3,18 \cdot 10^{3}$
3	$-4,1$	$3,18 \cdot 10^{2}$
4	$-3,3$	$2,00 \cdot 10^{3}$
5	$6,5$	$1,26 \cdot 10^{13}$

Nr	$\log k$	Da
6	$-2,5$	$1,26 \cdot 10^{4}$
7	$-3,5$	$1,26 \cdot 10^{3}$
8	$-4,7$	$7,98 \cdot 10^{1}$
9	$-4,7$	$7,98 \cdot 10^{1}$
10	$-7,3$	$2,00 \cdot 10^{-1}$

Die Ratenkonstanten der Reaktionen sind inklusive der Damköhler-Zahlen in Tabelle 5.5 verzeichnet. Es ist zu erkennen, dass die Reaktionen 1 und 5 sehr große

Damköhler-Zahlen besitzen, während die der restlichen Reaktionen wesentlich kleiner sind. Diese Tatsache lässt hoffen, dass mit Hilfe des modifizierten Newton-Verfahren deutliche Einsparungen in den Rechenzeiten zu erzielen sind.

Die Randwerte der Stoffkonzentrationen am linken Rand hängen von der Tiefe ab. Deshalb wird dieser Randteil in die sieben Bereiche $RW1$ bis $RW7$ aus Tabelle 5.6 unterteilt. Die Dirichlet-Randwerte sind in Tabelle 5.7 dokumentiert. Am oberen und unteren Rand gelten homogene Fluss-Randbedingungen, am rechten Rand homogene Neumann-Randbedingungen.

Tabelle 5.6: Bereich für Dirichlet-Randwerte für das Barriere-Beispiel am linken Rand.

Bezeichnung	Bereich $(x = 0)$
RW 1	$-4,25 \leq y \leq -4,00$
RW 2	$-4,75 \leq y < -4,25$
RW 3	$-5,25 \leq y < -4,75$
RW 4	$-5,75 \leq y < -5,25$
RW 5	$-6,25 \leq y < -5,75$
RW 6	$-6,75 \leq y < -6,25$
RW 7	$-7,00 \leq y < -6,75$

Tabelle 5.7: Dirichlet-Randwerte für das Barriere-Beispiel am linken Rand.

Spezies	RW 1	RW 2	RW 3	RW 4	RW 5	RW 6	RW 7
H^+	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$
CrO_4^{2-}	$5,8 \cdot 10^{-8}$	$1,9 \cdot 10^{-5}$	$3,7 \cdot 10^{-5}$	$4,8 \cdot 10^{-5}$	$9,8 \cdot 10^{-5}$	$8,9 \cdot 10^{-6}$	$2,9 \cdot 10^{-8}$
Fe^{2+}	$1,7 \cdot 10^{-15}$	$5,6 \cdot 10^{-18}$	$6,8 \cdot 10^{-17}$	$1,4 \cdot 10^{-13}$	$2,5 \cdot 10^{-13}$	$6,9 \cdot 10^{-14}$	$2,5 \cdot 10^{-14}$
Fe^{3+}	$1,2 \cdot 10^{-12}$	$1,0 \cdot 10^{-12}$	$1,0 \cdot 10^{-12}$	$3,8 \cdot 10^{-12}$	$4,7 \cdot 10^{-12}$	$3,3 \cdot 10^{-12}$	$1,7 \cdot 10^{-12}$
$Cr\,(OH)_2^+$	$2,2 \cdot 10^{-8}$	$1,6 \cdot 10^{-8}$	$1,5 \cdot 10^{-8}$	$1,1 \cdot 10^{-7}$	$1,5 \cdot 10^{-7}$	$8,9 \cdot 10^{-8}$	$3,6 \cdot 10^{-8}$
TCE	$2,0 \cdot 10^{-5}$	$3,6 \cdot 10^{-6}$	$2,8 \cdot 10^{-7}$	$2,6 \cdot 10^{-7}$	$2,0 \cdot 10^{-6}$	$2,3 \cdot 10^{-6}$	$4,7 \cdot 10^{-7}$
$cis-1,2DCE$	$5,2 \cdot 10^{-9}$	$5,2 \cdot 10^{-9}$	$5,2 \cdot 10^{-9}$	$9,3 \cdot 10^{-9}$	$1,4 \cdot 10^{-6}$	$3,0 \cdot 10^{-6}$	$5,7 \cdot 10^{-7}$
$Ethan$	$1,7 \cdot 10^{-7}$	$1,7 \cdot 10^{-7}$	$1,7 \cdot 10^{-7}$	$1,7 \cdot 10^{-7}$	$3,0 \cdot 10^{-6}$	$1,7 \cdot 10^{-7}$	$1,7 \cdot 10^{-7}$
VC	$8,0 \cdot 10^{-9}$	$8,0 \cdot 10^{-9}$	$8,0 \cdot 10^{-9}$	$8,0 \cdot 10^{-9}$	$1,7 \cdot 10^{-7}$	$1,1 \cdot 10^{-6}$	$1,2 \cdot 10^{-6}$
Cl^-	$4,2 \cdot 10^{-4}$	$5,9 \cdot 10^{-4}$	$8,4 \cdot 10^{-4}$	$2,3 \cdot 10^{-3}$	$4,0 \cdot 10^{-3}$	$2,0 \cdot 10^{-3}$	$7,4 \cdot 10^{-4}$
$O_2\,(aq)$	$9,1 \cdot 10^{-5}$	$3,7 \cdot 10^{-5}$	$6,3 \cdot 10^{-6}$	$1,6 \cdot 10^{-5}$	$8,1 \cdot 10^{-6}$	$1,1 \cdot 10^{-5}$	$2,5 \cdot 10^{-5}$
$H_2\,(aq)$	$2,6 \cdot 10^{-31}$	$1,9 \cdot 10^{-34}$	$8,6 \cdot 10^{-32}$	$2,0 \cdot 10^{-31}$	$2,0 \cdot 10^{-31}$	$1,0 \cdot 10^{-31}$	$9,9 \cdot 10^{-31}$
NO_3^-	$1,2 \cdot 10^{-5}$	$1,2 \cdot 10^{-5}$	$1,1 \cdot 10^{-5}$	$4,6 \cdot 10^{-5}$	$8,8 \cdot 10^{-5}$	$2,7 \cdot 10^{-5}$	$3,7 \cdot 10^{-6}$
NH_4^+	$3,7 \cdot 10^{-11}$	$1,3 \cdot 10^{-24}$	$2,7 \cdot 10^{-14}$	$4,4 \cdot 10^{-9}$	$1,4 \cdot 10^{-8}$	$1,2 \cdot 10^{-10}$	$1,8 \cdot 10^{-8}$
SO_4^{2-}	$1,9 \cdot 10^{-4}$	$3,3 \cdot 10^{-4}$	$5,3 \cdot 10^{-4}$	$9,4 \cdot 10^{-4}$	$1,4 \cdot 10^{-3}$	$3,2 \cdot 10^{-4}$	$8,4 \cdot 10^{-5}$
HS^-	$5,5 \cdot 10^{-88}$	$8,0 \cdot 10^{-100}$	$3,8 \cdot 10^{-90}$	$2,5 \cdot 10^{-86}$	$5,3 \cdot 10^{-86}$	$4,0 \cdot 10^{-88}$	$1,9 \cdot 10^{-85}$
CH_2O	$1,4 \cdot 10^{-4}$	$7,0 \cdot 10^{-6}$	$4,8 \cdot 10^{-5}$	$3,5 \cdot 10^{-5}$	$7,8 \cdot 10^{-5}$	$1,0 \cdot 10^{-4}$	$5,0 \cdot 10^{-5}$
$CH_4\,(aq)$	$4,8 \cdot 10^{-6}$	$1,8 \cdot 10^{-6}$	$3,9 \cdot 10^{-6}$	$3,4 \cdot 10^{-6}$	$1,4 \cdot 10^{-6}$	$9,3 \cdot 10^{-6}$	$1,4 \cdot 10^{-5}$
CO_3^{2-}	$1,1 \cdot 10^{-3}$	$1,2 \cdot 10^{-3}$	$1,2 \cdot 10^{-3}$	$1,9 \cdot 10^{-3}$	$2,0 \cdot 10^{-3}$	$3,0 \cdot 10^{-3}$	$2,4 \cdot 10^{-3}$
$Tracer$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$	$1,0 \cdot 10^{-5}$

Für die Simulationen verwenden wir ein Gitter mit

$$K \;=\; 14700$$

Knoten und 29400 Elementen, was zu einer räumlichen Schrittweite von

$$h \;\approx\; 0,0404\; m$$

führt. Für die Péclet- und Zell-Péclet-Zahl gilt

$$\begin{aligned} Pe &= 250, \\ Pe_T &\approx 2,02. \end{aligned}$$

Bemerkung 5.1. Im zugrunde liegenden Artikel [45] werden 16 weitere Mineral-Reaktionen und 19 zusätzliche Spezies eingeführt. Diese sorgen dafür, dass die Reaktionsprodukte wie z.B. $Cr\,(OH)_2^+$ und HS^- hinter der Barriere weiter abgebaut werden. Wegen fehlender Rechnerkapazitäten (vor allem Speicherplatz) wurde hier nur ein eingeschränktes Modell verwendet.

5.2.2 Simulationsergebnisse

In diesem Abschnitt werden die wesentlichen Spezies einer schnellen und einer langsamen Reaktion geplottet und diskutiert. Dabei dienen $R1$ mit $Da = 4,00 \cdot 10^{12}$ und $R7$ mit $Da = 1,26 \cdot 10^3$ als Musterbeispiele.

Abbildung 5.6 zeigt die Spezies CrO_4^{2-} (oben) und $Cr\,(OH)_2^+$ (unten), die beide nur in der ersten Reaktion vorkommen. Nach Tabelle 5.7 strömt CrO_4^{2-} mit einer maximalen Konzentration von $9,8 \cdot 10^{-5}\; mol/l$ von links in das Gebiet. Im Plot ist zu erkennen, dass es in Richtung des besonders leitfähigen mittleren Abschnitts ($Fe^0\,(s)$) reagiert. Die Konzentration fällt schlagartig auf Null ab,

Das Reaktionsprodukt $Cr\,(OH)_2^+$ fließt mit einer erheblich kleineren Konzentration von maximal $1,5 \cdot 10^{-9}\; mol/l$ ein. Es entsteht am linken Rand der Barriere ebenso plötzlich wie CrO_4^{2-} dort verschwindet. Hinter der Barriere folgt es dem Fließfeld und verlässt Ω.

Die relativ hohe Damköhler-Zahl der ersten Reaktion führt also dazu, dass CrO_4^{2-} in einem sehr schmalen räumlichen Bereich mit den Eisengranulat der Barriere und der überall vorhandenen H^+-Ionen zu $Cr\,(OH)_2^+$, Fe^{3+} und Wasser reagiert. Der Schadstoff wird dabei vollständig aufgebraucht.

Ein ganz ähnliches Bild entsteht in Abbildung 5.7 (oben) für SO_4^{2-}, das in Reaktion 7 mit $Fe^0\,(s)$ und H^+ zu Fe^{2+}, HS^- (unterer Plot der Abbildung) und Wasser reagiert. Aufgrund der sehr viel kleineren Damköhler-Zahl läuft die Reaktion jedoch nicht so schnell ab und der Stoff kann ein Stück weit in die Barriere

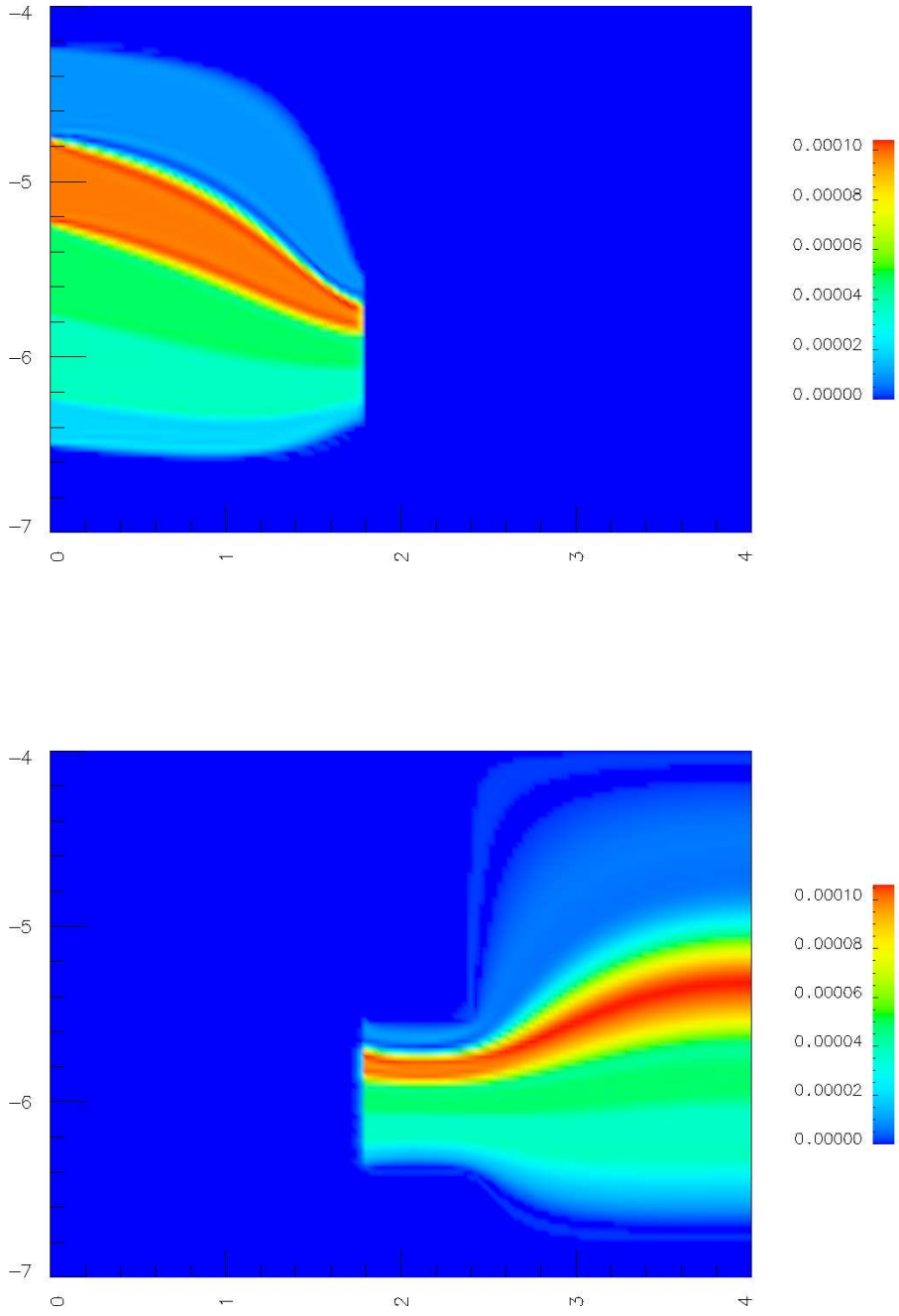

Abbildung 5.6: Plot der Spezies CrO_4^{2-} (oben) und $Cr(OH)_2^+$ (unten).

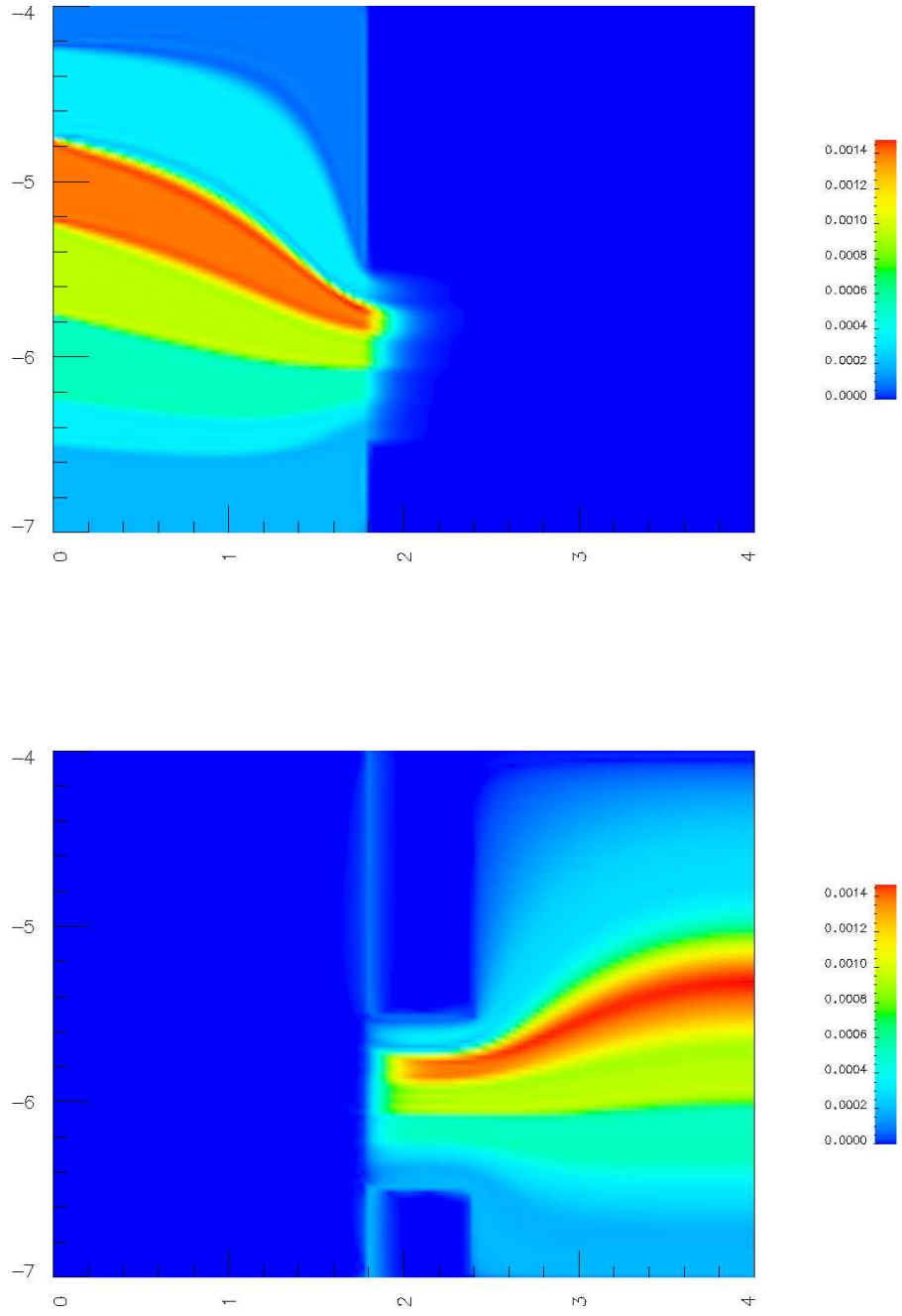

Abbildung 5.7: Plot der Spezies SO_4^{2-} (oben) und HS^- (unten).

eindringen. Die Konzentrationswerte fallen wesentlich langsamer ab als in der ersten Reaktion.

Entsprechend flach steigt die Konzentration des Produktes HS^- an. Sie erreicht ihr Maximum später als das $Cr\,(OH)_2^+$ von oben.

Die Reaktion ist jedoch so schnell, dass der Schadstoff auch hier vollständig abgebaut werden kann und kein SO_4^{2-} mehr den Bereich unter dem Fluss erreicht.

Wir verzichten auf Plots weiterer Spezies und verweisen für detailliertere Ausführungen auf die zitierten Arbeiten.

5.2.3 Performance des Verfahrens

Die Zeitschrittweite wird bei den folgenden Berechnungen fest gewählt gemäß

$$\Delta t \;=\; 2,5 \cdot 10^6 \; s$$

und adaptiv mit

$$\begin{aligned} \Delta t_{min} &:= 2,5 \cdot 10^6 \; s, \qquad & NS_{min} &:= 12, \qquad & fact_1 &:= 1,5, \\ \Delta t_{max} &:= 5,0 \cdot 10^6 \; s, \qquad & NS_{max} &:= 20, \qquad & fact_2 &:= 0,5. \end{aligned}$$

Der Endzeitpunkt ist

$$T \;=\; 1,75 \cdot 10^7 \; s \;\approx\; 202 \; d.$$

Es werden also bei fester Zeitschrittweite

$$TS \;=\; 70$$

Zeitschritte durchgeführt.
Bei T hat sich ein stationärer Endzustand eingestellt. Ein Weiterrechnen würde an den Ergebnissen nichts mehr ändern.

Für die gewählten Parameter liegt die Courant-Zahl im Intervall

$$Cr \;\in\; (7,43;\; 14,9)\,.$$

Der Einsatz des in dieser Arbeit verwendeten impliziten Verfahrens ist also berechtigt.

Bei der Simulation des Barriere-Problems soll erstmals die in Abschnitt 4.2.2 beschriebene adaptive Strategie zur Wahl der Aufteilungsschranke ϵ_B^{rel} angewandt werden. Als Startwert setzen wir

$$\epsilon_B^{rel} \;:=\; 1 \cdot 10^{-6},$$

was nach den Erfahrungen aus den bisherigen Simulationen einem nicht zu großen Wert entspricht. Die Grenzen seien

$$\begin{aligned} \epsilon^{rel}_{B,min} &= 0, \\ \epsilon^{rel}_{B,max} &= 1, \end{aligned}$$

so dass von keiner Aufteilung bis hin zu voller Aufteilung alle Möglichkeiten angenommen werden können. Als Grenzen für die Newtonschritte wählen wir

$$\begin{aligned} NS^B_{min} &= 13, \\ NS^B_{max} &= 15. \end{aligned}$$

Die untere Schranke wurde hier mit Hilfe der Ergebnisse des vollen Newton-Verfahrens gefunden. Die obere lässt drei zusätzliche Newtonschritte im Vergleich zum ursprünglichen Verfahren zu. Dieser Wert stellt sicher, dass die Aufteilung nicht so stark ist, dass die zunehmende Assemblierungszeit den Gewinn im linearen Löser kompensiert.
Für die beiden Multiplikatoren gilt

$$\begin{aligned} fact^B_1 &= 10, \\ fact^B_2 &= 0,1. \end{aligned}$$

Diese Faktoren erscheinen recht hoch, haben sich jedoch in allen Testsimulationen als zweckmäßig erwiesen.

Tabelle 5.8 dokumentiert die Performance der getesteten Verfahren. Sowohl bei fester als auch bei adaptiver Zeitschrittweite halbiert sich die Rechenzeit bei Verwendung des modifizierten Newton-Verfahrens, der lineare Löser ist etwa dreimal so schnell und die Assemblierungszeit ist geringfügig höher als beim vollen Newton-Verfahren.

Tabelle 5.8: Performance für das Barriere-Beispiel mit fester und adaptiver Zeitschrittweite und Aufteilung des linearen Gleichungssystem. Es wurden die Bezeichnungen aus Tabelle 5.3 verwendet.

Δt	Aufteilung	TS	NS/TS	Ass	$Auto$	LS	N_T	Ges
fest	ohne	70	11,9	346	–	1116	1	1530
fest	mit	70	12,8	378	5	365	13	775
adaptiv	ohne	39	13,4	213	–	842	1	1098
adaptiv	mit	39	13,7	223	3	290	13	533

Diese Erhöhung der Assemblierungszeit wird zum einen durch die zusätzlich benötigten Newtonschritte verursacht, zum anderen aber auch durch das Kopieren der

Jacobi-Matrix in die Teilmatrizen. Hier zeigt sich die in Abschnitt 3.1.4 beschriebene Überlegenheit des Kopierens im Vergleich zu einem Blocklöser bei großen n. Es benötigt nur ca. 2 % der gesamten Assemblierungszeit.

Das automatische Ermitteln der Teilsysteme mit der verwendeten Strategie (ϵ_B^{rel}) benötigt nur fünf bzw. drei Minuten und ist somit vernachlässigbar. Sowohl bei fester als auch bei adaptiver Zeitschrittweite wird das Gesamtsystem in 13 Untersysteme aufgeteilt. Es wird ein großes lineares Gleichungssystem mit den zehn Spezies

$$\begin{aligned}(H^+,\ CrO_4^{2-},\ Fe^{3+},\ Cr\,(OH)_2^+\,,\ H_2O,\ C_2HCl_3,\\ C_2H_2Cl_2,\ O2\,(aq)\,,\ H_2\,(aq)\,,\ Fe^0\,(s))\end{aligned}$$

verwendet und zwölf kleine mit je einer Spezies:

$$\begin{aligned}\left(Fe^{2+}\right),\quad &(C_2H_6),\quad &(C_2H_3Cl),\\ \left(Cl^-\right),\quad &\left(NO_3^-\right),\quad &\left(NH_4^+\right),\\ \left(SO_4^{2-}\right),\quad &\left(HS^-\right),\quad &(CH_2O),\\ (CH_4\,(aq)),\quad &\left(CO_3^{2-}\right),\quad &(Tracer).\end{aligned}$$

Durch diese Wahl verbleiben insbesondere alle Spezies der schnellen Reaktionen R_1 und R_5 in einem Teilsystem. Selbst bei einer Vergrößerung der Schranke ϵ_B^{rel} um drei Größenordnungen würde keine weitere Aufteilung erfolgen.

Die Verwendung der für den linearen Fall bewiesene Abschätzung (3.7) würde selbst mit der relativ großen Schranke

$$C_{lin}^{max} := 1 \cdot 10^{-3}$$

keine bessere Aufteilung liefern. Es würden sogar nur neun Teilsysteme verwendet werden und die Zeit für die automatische Partitionierung läge im Bereich der Assemblierungszeit. Das modifizierte Verfahren würde mehr Zeit als das volle benötigen. Diese Strategie muss deshalb auch hier als für nichtlineare Modelle ungeeignet angesehen werden.

Der Vergleich der Verfahren mit festem und adaptivem Δt zeigt, dass sich die Anzahl der Zeitschritte TS von 70 auf 39 verringert. Trotz einer Erhöhung der Anzahl der Newtonschritte NS sinkt der Aufwand zur Assemblierung und für den linearen Löser erheblich. Die Gesamtzeit verkleinert sich durch das zeitadaptive Verfahren sowohl beim vollen Newton-Verfahren als auch beim modifizierten um ca. 30 %.

Abschließend visualisiert Abbildung 5.8 die Entwicklung der Anzahl der Newtonschritte pro Zeitschritt für das volle und das modifizierte Newton-Verfahren ohne (oberer Plot) und mit Zeitadaption (unterer Plot). Der Vergleich der Kurven mit und ohne Adaption zeigt einen qualitativ ähnlichen Verlauf. Es ist auch zu erkennen, dass das modifizierte Verfahren mindestens so viele Schritte benötigt wie das volle, meist aber einen mehr.

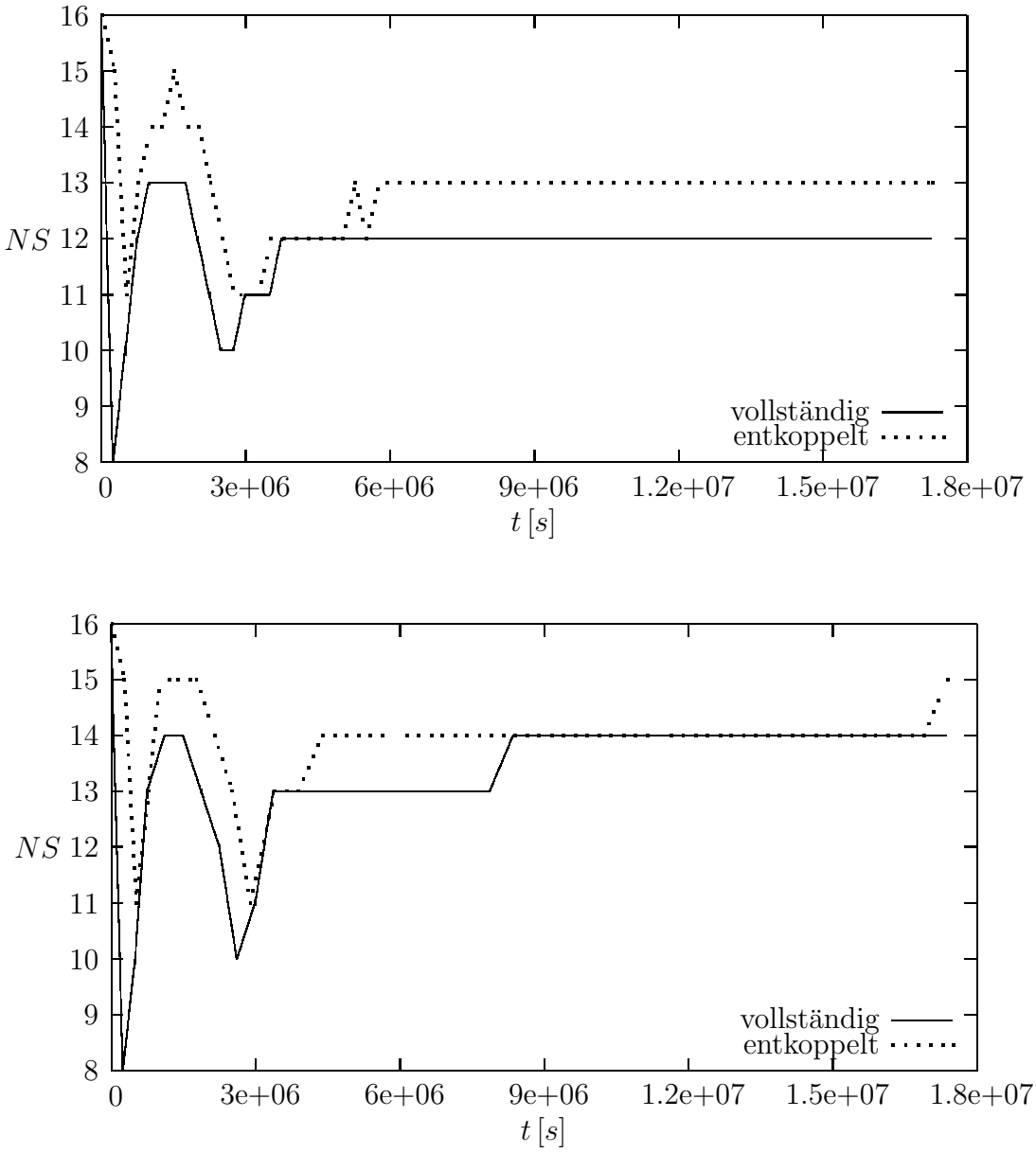

Abbildung 5.8: Anzahl der Newtonschritte pro Zeitschritt für die Simulationen aus Tabelle 5.8. Der obere Plot beinhaltet die Simulationen mit fester Zeitschrittweite (obere Hälfte der Tabelle), der untere die mit adaptiver Zeitschrittweite (untere Hälfte der Tabelle).

Anhang A

Implementierung

Die am Institut verwendete Simulationssoftware *M++* [63] sieht bereits das Lösen des durch (2.10), (2.11) und (2.12) definierte nichtlinearen Gleichungssystems mit dem Newtonverfahren vor. Als direkter linearer Löser kann unter anderem *SuperLU* verwendet werden. Hierbei handelt es sich um den derzeit wohl effizientesten frei verfügbaren direkten Löser.

M++ stellt auch zahlreiche iterative Löser zur Verfügung. In dieser Arbeit werden ausschließlich GMRES und BiCGStab mit entsprechenden Vorkonditionierern verwendet.

Als Programmiersprache dient *C++* (objektorientiert).

Nachfolgend wird zunächst eine an [33] orientierte erweiterte Anleitung zur Steuerung des modifizierten Newtonverfahrens mit Zeitadaption durch ein Skriptfile angegeben. Mit Hilfe dieses Abschnittes können insbesondere die Beispiele dieser Arbeit nachvollzogen werden.

Anschließend wird zunächst das *SuperLU*-Verfahren vorgestellt. Dann sollen nach einer Erklärung der Datenstrukturen von *M++* die wichtigsten neu implementierten Funktionen aufgelistet und kurz kommentiert werden.

Alle Erweiterungen der Software, die im Rahmen dieser Arbeit programmiert wurden, befinden sich in den Dateien *SubSystems.h*, *SubSystems.C*, *Newton.h*, *Newton.C*, *MultiSpec.C* und *Richards.h* und *Euler.C*.

A.1 Skript-File

M++ liest die benötigten Daten, Koeffizienten und Parameter aus einem Skript-File ein. Für das ursprüngliche volle Newton-Verfahren wurde dies bereits in der Diplomarbeit [33] beschrieben. Das zeitadaptive Entkopplungsverfahren erfordert einige weitergehende Angaben:

Allgemeine Parameter

Zunächst ist *M++* mitzuteilen, welche Simulation durchgeführt werden soll:

$$SimID \;=\; 0$$

(0 ist der Standardwert). Tabelle A.1 zeigt die möglichen Werte von $SimID$ für die Beispiele aus den Kapiteln 3, 4 und 5.

Tabelle A.1: $SimID$ der verschiedenen Simulationen.

$SimID$	Beschreibung
0	Eingabe aller Daten über das Skript-File (Standard).
1	Modell mit 12 Spezies und 6 Reaktionen aus Beispiel 3.7.
2	Lineares Modell mit 12 Spezies und 6 Reaktionen aus Beispiel 3.11.
4	Barriere-Problem aus Abschnitt 5.2.
5	$EDTA-$Problem aus Abschnitt 5.1.

Für diese Simulationen wurden Anfangs- und Randwerte direkt im Quelltext gesetzt. Falls $SimID$ nicht angegeben wird, sind die Werte wie in [33] beschrieben über das Skriptfile einzugeben.

Bei manchen Simulationen kann es sinnvoll sein, zur Lösung der Richards-Gleichung und der reaktiven Transportgleichungen unterschiedliche lineare Löser zu verwenden. In diesem Fall ist

$$different\_lsolver \;=\; 0$$

auf 1 zu setzen. Der Löser für die Richards-Gleichung ist über

$$NewtonSolver\_R$$

anzugeben, der für die Transportgleichungen über

$$NewtonSolver\_T.$$

Die beiden in dieser Arbeit verwendeten Verfahren sind durch die Eingabe von

$$\begin{aligned} Newton\_Solver\_X &= BiCGStab \qquad \text{bzw.} \\ Newton\_Solver\_X &= superlu \end{aligned}$$

mit $X \in \{R, T\}$ aufrufbar.

Das modifizierte Newton-Verfahren

Anschließend ist das zu verwendende Newton-Verfahren anzugeben. Dies erfolgt in der Form

$$NewtonMethod \;=\; 0,$$

Tabelle A.2: Wahl des Newton-Verfahrens mit $NewtonMethod$.

$NewtonMethod$	$Funktion$	Beschreibung
0	$NewtonSimple$	einfaches Newton-Verfahren ohne Aufteilung (Standard)
1	$NewtonBlock(A, x, 1)$	modif. Newton-Verfahren mit manueller Aufteilung
2	$NewtonBlock(A, x, 0)$	modif. Newton-Verfahren mit automatischer Aufteilung
3	$NewtonBroyden$	modif. Newton-Verfahren nach Broyden mit manueller Aufteilung

wobei die Werte aus Tabelle A.2 zulässig sind. Der Standardwert ist auch hier 0, so dass ohne Angabe von $NewtonMethod$ das einfache Newton-Verfahren ohne Aufteilung ausgeführt wird.

Für $NewtonMethod = 1$ kann die Art der Aufteilung (Blockbildung) des linearen Gleichungssystems mit dem Parameter

$$BlockType \;=\; 0$$

festgelegt werden. Die möglichen Werte sind in Tabelle A.3 verzeichnet. Standardwert ist hier $BlockType = 0$ für $NewtonMethod \neq 2$. Für $NewtonMethod = 2$ ist $BlockType = 2$ voreingestellt und kann nicht geändert werden.

Tabelle A.3: Wahl der Aufteilung des linearen Gleichungssystems (Blockbildung).

$BlockType$	N_S	N_T	Beschreibung	
0	beliebig	N_S	$(A\dots)$	voll gekoppelt (Standard)
1	beliebig	1	$(A)\dots$	voll entkoppelt
62	6	2	$(ABC)\,(DEF)$	
122	12	2	$(A\dots I)\,(JKL)$	
123	12	3	$(ABC)\,(DEF)\,(G\dots L)$	
124	12	4	$(ABC)\,(DEF)\,(GHI)\,(JKL)$	
126	12	6	$(AG)\,(BH)\,(CI)\,(DJ)\,(EK)\,(FL)$	
128	12	8	$(AG)\,(B)\,(H)\,(CI)\,(DJ)\,(E)\,(K)\,(FL)$	
2	beliebig	variabel	Automatische Aufteilung	

Mittels des Parameters

$$AssembleFull \;=\; 1$$

kann festgelegt werden, ob die Jacobi-Matrix voll assembliert werden soll (1, Standard) oder nur die benötigten Einträge (0).

Die Wahl der Entkopplungsstrategie erfolgt mit Hilfe von

$$\begin{aligned} BlockEpsilon\_rel\_min &= 1 \cdot 10^{-8} \;\hat{=}\; \epsilon_{B,1}^{rel}, \\ BlockEpsilon\_rel\_max &= 1 \;\hat{=}\; \epsilon_{B,2}^{rel}, \\ alpha\_max &= 1 \cdot 10^{-4} \;\hat{=}\; C^{max}. \end{aligned}$$

Falls nur die einfache Entkopplungsstrategie gemäß (4.4) durchgeführt werden soll, ist

$$alpha\_max = -1$$

zu wählen.

Die Zeitadaption

Für die Steuerung der Zeitadaption sind folgende Parameter zu setzen:

$$\begin{aligned} nNewtonSteps1 &= 0 \;\hat{=}\; NS_{min}, \\ nNewtonSteps2 &= 0 \;\hat{=}\; NS_{max}, \\ fact1 &= 1,5 \;\hat{=}\; fact_1, \\ fact2 &= 0,5 \;\hat{=}\; fact_2. \end{aligned}$$

Die Startschrittweite wird mit

$$dt = \ldots$$

gesetzt, der minimale und maximale Wert mit

$$\begin{aligned} dt\_min &= \ldots, \\ dt\_max &= \ldots. \end{aligned}$$

Diese drei Werte sind immer anzugeben. Es existieren deshalb keine Standardschrittweiten. Selbstverständlich muss

$$dt\_min \leq dt \leq dt\_max$$

gelten. Für

$$dt\_min = dt\_max$$

findet keine Zeitadaption statt.

Adaptive Bestimmung von ϵ_B^{rel}

Die Aufteilungsschranke ϵ_B^{rel} wird mittels

$$\begin{aligned} nNewtonSteps1\_B &= 0 \;\hat{=}\; NS_{min}^{B}, \\ nNewtonSteps2\_B &= 0 \;\hat{=}\; NS_{max}^{B}, \\ fact1\_B &= 1,0 \;\hat{=}\; fact_1^B, \\ fact2\_B &= 0,1 \;\hat{=}\; fact_2^B, \end{aligned}$$

adaptiv bestimmt. Selbstverständlich ist ein Startwert für ϵ_B^{rel} wie bereits beschrieben anzugeben. Als maximaler und minimaler Wert für ϵ_B^{rel} dienen $\epsilon_{B,1}^{rel}$ und $\epsilon_{B,2}^{rel}$ von oben.
Für

$$nNewtonSteps1\_B \;=\; nNewtonSteps2\_B$$

wird ϵ_B^{rel} nicht adaptiv verändert. Der gewählte Wert bleibt während der gesamten Simulation konstant.

A.2 Das *SuperLU*-Verfahren

Ein sehr effizientes direktes Verfahren zur Lösung großer dünnbesetzter linearer Gleichungssysteme ist *SuperLU* [40]. Es ist auf vielen Unix- und Linux-Systemen vorinstalliert und sein Quellcode kann unter der Internetadresse [41] heruntergeladen werden.

SuperLU berechnet eine Dreieckszerlegung

$$P_r D_r A D_c P_c \;=\; LU \tag{A.1}$$

mit Permutationsmatrizen P_r (Zeilen) und P_c (Spalten), Diagonalmatrizen D_r und D_c, einer normierten unteren Dreiecksmatrix L (mit $L_{ii} = 1$) und einer oberen Dreiecksmatrix U.
Das System $A\mathbf{x} = \mathbf{b}$ wird gelöst durch Berechnung von

$$\mathbf{x} \;=\; A^{-1}\mathbf{b} \;=\; D_c \left(P_c \left(U^{-1} \left(L^{-1} \left(P_r \left(D_r \mathbf{b}\right)\right)\right)\right)\right). \tag{A.2}$$

SuperLU kann sowohl auf dünn- als auch auf vollbesetzte Matrizen angewandt werden.

Der Algorithmus verläuft in drei Schritten:

1. Wähle die Permutationsmatrix P_c (Umordnung der Spalten von A), so dass möglichst viele Einträge in L und U Null sind.

2. Berechne die $LU-$Zerlegung von AP_c (siehe (A.1)).

3. Löse das Gleichungssystem (siehe (A.2)).

Wir gehen an dieser Stelle nicht auf weitere Details des Verfahrens ein und verweisen statt dessen auf die zitierte Literatur.

A.3 Datenstrukturen

Die durchaus komplizierten Datenstrukturen zur Speicherung der Matrix A in $M{+}{+}$ werden ansatzweise in [63] beschrieben. Für die Zwecke der vorliegenden Arbeit genügt es zu wissen, dass nur die Blockmatrizen ■ und □ aus Abbildung 2.2 gespeichert werden, wobei sowohl die vollbesetzten Blöcke ■ als auch die Diagonalblöcke □ je N_S^2 Speicherplätze belegen. Die Diagonalblöcke werden also wie vollbesetzte Blöcke behandelt.
Die Speicherung erfolgt in ein eindimensionales Feld a. Von besonderem Interesse ist die Zuordnung der Einträge in a zu den einzelnen Spezies. Diese ist sehr leicht möglich, da a die Struktur

i	0	1	...	N_S-1	N_S	N_S+1	...	$2N_S-1$	...
Eintrag	a_0	a_1	...	a_{N_S-1}	a_{N_S}	a_{N_S+1}	...	a_{2N_S-1}	...
Spezies	0	1	...	N_S-1	0	1	...	N_S-1	...

besitzt. Der $i-$te Eintrag in a ist somit der Spezies

$$s_i \;=\; i \mod N_S$$

zugeordnet. Hier wurde bereits die mit 0 beginnende Nummerierung der Programmiersprache C verwendet.

Vektoren $\mathbf{v}$ werden ganz analog in ein Feld v gespeichert, wobei die oben beschriebene Struktur zur Speicherung dünnbesetzter Matrizen benutzt wird.

Die Datenstrukturen sind in $M{+}{+}$ in den Klassen *Matrix* und *Vector* definiert. Sie wurden in der Headerdatei *Algebra.h* festgelegt. In *Algebra.C* befinden sich zahlreiche Funktionen und Makros zur Manipulation der Daten.

A.4 Lösen der Teilsysteme

Zu lösen sind die Teilsysteme (3.2) mit einem gewählten linearen Löser Dazu werden die Matrix und die Vektoren mit Hilfe der drei Felder

- $SpecList[i][j]$ zur Speicherung der Speziesindizes des $i-$ten Teilsystems,
- $NSpecList[i] := n_i$ (Anzahl der Spezies der Teilsysteme) und
- $SpecListOfSpecies[i]$ ($SpecList$, der die Spezies i angehört)

in die Teilmatrizen und -vektoren $AT[i]$, $xT[i]$ und $rhsT[i]$ für $i \in \{0,\ldots,N_T-1\}$ und $j \in \{0,\ldots,N_S-1\}$ kopiert.

Der Kopiervorgang wird durch die Funktion *Copy2SubSystems* ausgeführt:

```
1  void SubSystems::Copy2SubSystems(const Matrix& J, Vector& c, Vector& b)
2  {
3    double *ATPtr[MaxSubSys], *xTPtr[MaxSubSys], *rhsTPtr[MaxSubSys];
4    const double* JPtr=J.Ptr(), *cPtr=c.Ptr(), *bPtr=b.Ptr();
5    int line_in_block=0, speclist_line=SpecListOfSpecies[line_in_block];
6
7    for(int i=0; i<NT; i++)
8    {
9      ATPtr[i]  =AT[i]->Ptr();
10     xTPtr[i]  =xT[i]->Ptr();
11     rhsTPtr[i]=rhsT[i]->Ptr();
12   }
13   for(int i=0; i<J.size(); i++)
14   {
15     int species=i%NS;
16     int speclist=SpecListOfSpecies[species];
17     if(speclist==speclist_line)
18     {
19       *ATPtr[speclist]=*JPtr;
20       ATPtr[speclist]++;
21     }
22     JPtr++;
23     if(species==NS-1)
24     {
25       line_in_block++;
26       if(line_in_block==NS) line_in_block=0;
27       speclist_line=SpecListOfSpecies[line_in_block];
28     }
29   }
30   for(int i=0; i<J.dim(); i++)
31   {
32     int speclist = SpecListOfSpecies[i%NS];
33     *xTPtr[speclist]=*cPtr;
34     *rhsTPtr[speclist]=*bPtr;
35     xTPtr[speclist]++; rhsTPtr[speclist]++;
36     cPtr++; bPtr++;
37   }
38 }
```

Sie erhält aus der aufrufenden Prozedur (dem Newton-Löser) die Jacobi-Matrix J, den Lösungsvektor c und die rechte Seite b.
Zunächst werden in den Zeilen 7 bis 12 Zeiger auf die Anfangsposition der eindimensionalen Felder der Matrix (a) und der beiden Vektoren (v) der beiden Vektoren festgelegt. Die Schleife über die Nichtnullelemente von J ($13-29$) durchläuft die Matrix vom ersten bis zum letzten Eintrag und vermeidet ein zeitintensives hin- und herspringen. Der Zeiger auf das aktuelle Element $JPtr$ wird lediglich inkrementiert (22).
In den Zeilen 15 und 16 wird ermittelt, zu welcher Spezies und zu welchem Teilsystem (SpecList) das Element gehört, auf das $JPtr$ gerade zeigt. In 17 bis 21

erfolgt die Einordnung in die Teilmatrizen, wobei auch hier das Element von $AT\,[speclist]$ nicht adressiert wird. Der betreffende Zeiger $ATPtr\,[speclist]$ wird ebenfalls zeitsparend inkrementiert (20).
Der Kopiervorgang für die Vektoren c und b schließt sich in den Zeilen 30 bis 37 an. Hier wird die gleiche effiziente Strategie verwendet wie beim Kopieren der Matrix J.

Mit den oben erzeugten Teilsystemen wird im Newton-Löser der gewählte Vorkonditionierer und der lineare Löser aufgerufen. Durch das Kopieren in die neuen Strukturen sind in diesen Prozeduren keine Änderungen nötig.

A.5 Der Newtonlöser *NewtonBlock*

Der in diesem Abschnitt aufgelistete modifizierte Newtonlöser *NewtonBlock* wurde bereits in 3.1.4 beschrieben. Er ermöglicht das Lösen des nichtlinearen Gleichungssystems (2.10), (2.11) und (2.12) mit manueller oder automatischer Aufteilung des linearen Gleichungssystems (2.13) in Teilsysteme (3.2)

```
1   // Block Newton Method (auto definition of blocks)
2   void Newton::NewtonBlock(const Assemble& A, Vector& x, int manu)
3   {
4       Date Start;
5
6       A.MDirichlet(x);
7       Vector b(x), c(x);
8       Matrix J(x);
9       int nready=0, ready[MaxSubSystems];
10      int elements2Ass[MaxSpecies][MaxSpecies];
11      int nNewtonStepsSubSystems[MaxSubSystems], nNewtonStepsTS = 0;
12
13      // compute defect b
14      A.MDefect(x, b);
15
16      double d=norm(b), Eps=eps, Eps_Block[MaxSubSystems];
17
18      nTimeSteps++;
19      iter = 0;
20
21      // start Newton step only, if norm(b)>Eps
22      if( d>Eps )
23      {
24        // some definitions
25        Date start_ass1;
26        for(int i=0; i<MaxSpecies; i++)
27        {
28          for(int j=0; j<MaxSpecies; j++)
29            elements2Ass[i][j] = 1;
30        }
```

```
31        for(int i=0; i<MaxSubSystems; i++)
32        {
33          ready[i] = -1;
34          nNewtonStepsSubSystems[i] = 0;
35        }
36        ass_time += Date() - start_ass1;
37
38        if(!manu)
39        {
40          // All elements of J should be assembled
41          SetElements2Assemble();
42
43          // Assemble
44          A.MMatrix (x, J);
45          c = 0;
46          J.SetDirichlet(c, b);
47
48          if(nTimeSteps==1)
49            SetFirstNStepZero();
50        } // if(!manu)
51
52        // get the list SpecList[i] (Species belonging to SubSystem i),
53        // NSpecList[i] (number of Species in SpecList[i]) and
54        // SpecListOfSpecies[i] (SpecList, to which species i belongs to)
55        GetSubCompList(A, J, x, c, b, ready, nTimeSteps);
56
57        // compute Eps_Block
58        for(int i=0; i<NT; i++)
59          Eps_Block[i] = Eps * NSpecList[i] / NS;
60
61        // Write infos about assembled reactions to files
62        WriteReactions2AssInfo(NT, SpecList, NSpecList, SpecListOfSpecies,
63                               ready);
64
65        // create memory for SubSystems
66        CreateSubStructures(PermcType);
67
68        if(!manu)
69        {
70          // copy J, c and b to SubSystems
71          Copy2SubSystems(J, c, b);
72        } // if(!manu)
73        else
74        {
75          Copy2SubSystems(b, rhsT);
76          Copy2SubSystems(x, xT);
77        } // if(manu)
78
79        // does some of the subsystems fulfill norm(defect) <= Eps?
80        Date start_ass2;
81        for(int i=0; i<NT; i++)
```

```
82        {
83          double norm_subsys = norm(*rhsT[i]);
84          if( norm_subsys<=Eps_Block[i] )
85          {
86            ready[i] = iter;
87            nready++;
88          }
89        } // for i
90        ass_time += Date() - start_ass2;
91
92        // get array Permc (permutations for SuperLU: Solve AP_c = LU)
93        // in LinearSolver: get permutations (function get_perm_c)
94        if( (PermcType>=0) && (Permc==NULL) )
95        {
96          Permc = intMalloc(J.dim());
97          BuildPermc(J, PermcType);
98        } // Permc
99
100       // print dim and size of J
101       if(nTimeSteps==1)
102       {
103         cout << "J.dim     = " << J.dim()  << '\n';
104         cout << "J.size    = " << J.size() << '\n';
105
106         PrintInfoSubStructures();
107       } // if nTimeSteps==1
108
109       // Loop: Newton-iterations (counter iter)
110       while ( (iter<MaxIter) && (nready<NT) )
111       {
112         nNewtonSteps++;
113         nNewtonStepsTS++;
114
115         if( (!manu && iter) || manu)
116         {
117           // get elements2Ass, reactions2Ass and species2Ass
118           // according to ready
119           Date start_ass3;
120           if(AssembleFull)
121             SetElements2Assemble();
122           else
123             SetElements2Assemble(NT, SpecList, NSpecList,
124                                  SpecListOfSpecies, ready);
125           ass_time += Date() - start_ass3;
126
127           // Assemble
128           A.MMatrix (x, J);
129           c = 0;
130           J.SetDirichlet(c,b);
131           SetFirstNStepZero();
132
```

```
133           // copy J, c and b to SubSystems
134           Copy2SubSystems(J, c, b);
135         } // if iter && !manu
136
137         // reducing of bandwidth
138         if(!iter)
139           if(PermcType>=0)
140             Get_PermcT(J.dim());
141
142         // solve the SubSystems
143         for(int i=0; i<NT; i++)
144         {
145           if(ready[i] == -1)
146           {
147             nNewtonStepsSubSystems[i]++;
148             cout << "Solving SubSystem " << i+1 << " with species ";
149             for(int j=0; j<NSpecList[i]; j++)
150               cout << SpecList[i][j] << ' ';
151             cout << '\n';
152
153             if(*AT[i] != 0.0)
154             {
155               if(PermcType>=0)
156                 S(*AT[i], *xT[i], *rhsT[i], PermcT[i]);
157               else
158                 S(*AT[i], *xT[i], *rhsT[i]);
159             } // if *AT[i] != 0.0
160             else
161               *xT[i] = 0.0;
162           } // if ready[i]==-1
163         } // for i
164
165         // copy solution xT of SubSystems back to full solution c
166         Copy2Full(xT, c);
167
168         x -= c;
169         A.MDefect(x, b);
170
171         // copy b to SubSystems (rhsT)
172         Copy2SubSystems(b, rhsT);
173
174         // which subsystems fulfills norm(x)<=Eps?
175         Date start_ass4;
176         for(int i=0; i<NT; i++)
177         {
178           double norm_subsys;
179
180           norm_subsys = norm(*rhsT[i]);
181           if(ready[i]==-1)
182           {
183             if(norm_subsys<=Eps_Block[i])
```

```
184               {
185                 ready[i] = iter;
186                 nready++;
187               }
188             } // if ready[i]==-1
189             Vout(0) << "Newton Subsystem " << i+1 << " d(" << iter
190                     << ") = " << norm_subsys << '\n';
191           } // for i
192           ass_time += Date() - start_ass4;
193
194           iter++;
195         } // while iter
196
197         // Delete memory for SubStructures (created by CreateSubStructures)
198         DeleteSubStructures(PermcType);
199       } // if norm(b)>Eps
200       else
201       {
202         NT = NS;
203         for(int i=0; i<NS; i++)
204         {
205           SpecList[i][0] = i;
206           NSpecList[i] = 1;
207           SpecListOfSpecies[i] = i;
208           ready[i] = 1;
209           nNewtonStepsSubSystems[i] = 0;
210         } // for i
211       } // else (if norm(b) >Eps)
212
213       // Make counter compareable to NewtonSimple
214       nNewtonSteps++;
215       nNewtonStepsTS++;
216
217       // Write the number of Newton steps for this time step in file
218       WriteNumberOfNewtonSteps(nNewtonStepsTS);
219
220       Vout(0) << " Newton: d(" << iter << ")= " << norm(b)
221               << " rate " << rate() << " time " << Date() - Start << "\n";
222       if (iter==m) mout << "Newton: max iter = " << iter << " : "
223                         << d << "\n";
224   } // Newton::NewtonBlock
```

Abbildung 3.2 zeigt das Struktogramm dieses Unterprogramms für manuelle Aufteilung des linearen Gleichungssystems.

A.6 Aufteilung des Reaktionsgraphen

Die erforderliche automatische Partitionierung eines Reaktionsgraphen wird mit der Funktion `GetSubCompList_Auto` realisiert:

```
1  void SubSystems::GetSubCompList_Auto(Matrix& J, Vector& x, Vector& c,
2                                       Vector& b, int* ready, int nTimeSteps)
3  {
4    int X[MaxSpecGraphs][MaxSpecies], aktiv[MaxSpecGraphs], n[MaxSpecGraphs];
5    int belegt, aktuell;
6    double SpecGraph[MaxSpecies][MaxSpecies],
7           SubSpecGraph[MaxSpecies][MaxSpecies], SpecRHS[MaxSpecies];
8    double minKosten, Kosten[MaxSpecGraphs], weight;
9    double epsilon_min, epsilon_max, alpha=1.0;
10
11   for(int i=0; i<NS; i++)
12     X[0][i]=i;
13   Kosten[0]=0.0;  belegt=1;  aktuell=0;
14   aktiv[0]=1;  n[0]=NS;
15   for(int i=1; i<MaxSpecGraphs; i++)
16   {
17     aktiv[i]=0;  n[i]=0;
18   }
19
20   // compute SpecGrpah and SpecRHS (max of single graphs in spacial nodes)
21   GetSpecGraphMax(J, x, SpecGraph);
22   SetSpecGraph2UpperT(SpecGraph, NS);
23
24   weight = GetSpecGraphWeight(NS, SpecGraph);
25
26   epsilon_min = BlockEpsilon_rel_min*weight;
27   epsilon_max = BlockEpsilon_rel_max*weight;
28
29   while( n[aktuell] && (aktuell<MaxSpecGraphs) )
30   {
31     if( aktiv[aktuell] && (n[aktuell]>1) )
32     {
33       CopySpecGraph(SpecGraph, SubSpecGraph, X[aktuell], n[aktuell]);
34
35       Kosten[belegt] = MinSchnitt(X[aktuell], n[aktuell], SubSpecGraph,
36                           X[belegt], X[belegt+1], n[belegt], n[belegt+1]);
37       Kosten[belegt+1] = Kosten[belegt];
38
39       cout << "X    = ";
40       for(int i=0; i<n[belegt]; i++)
41         cout << X[belegt][i] << ' ';
42       cout << '\n';
43       cout << "Xbar = ";
44       for(int i=0; i<n[belegt+1]; i++)
45         cout << X[belegt+1][i] << ' ';
46       cout << '\n';
47       cout << "Costs =     " << Kosten[belegt] << '\n';
48
49       if( (Kosten[belegt]>epsilon_min) && (Kosten[belegt]<=epsilon_max)
50             && (alpha_max>=0.0) && (nTimeSteps>1) )
51       {
```

```
52          int aktuell_alt=aktuell, belegt_alt=belegt;
53          int a1=aktiv[aktuell], a2=aktiv[belegt], a3=aktiv[belegt+1];
54
55          aktiv[aktuell] = 0;
56          aktiv[belegt] = 1;
57          aktiv[belegt+1] = 1;
58          belegt += 2;
59          aktuell = belegt;
60
61          SetSpecList(X, aktuell, aktiv, n);
62          alpha = ComputeNewtonAlpha(A, J, x, c, b, ready, nTimeSteps);
63
64          aktuell = aktuell_alt;
65          belegt = belegt_alt;
66          aktiv[aktuell] = a1;
67          aktiv[belegt] = a2;
68          aktiv[belegt+1] = a3;
69
70          SetSpecList(X, aktuell, aktiv, n);
71        } // if
72
73        if( ((Kosten[belegt]<=epsilon_min) || (fabs(alpha)<=alpha_max))
74            && (nTimeSteps>1) )
75        {
76          aktiv[aktuell] = 0;
77          aktiv[belegt] = 1;
78          aktiv[belegt+1] = 1;
79          cout << "Aufteilung akzeptiert\n";
80        } // if Kosten<epsilon
81        else
82        {
83          aktiv[belegt] = 0;
84          aktiv[belegt+1] = 0;
85          cout << "Aufteilung verworfen\n";
86        } // if Kosten>=epsilon
87        belegt += 2;
88      } // if aktiv
89      aktuell++;
90    } // while
91
92    SetSpecList(X, aktuell, aktiv, n);
93  } // SubSystems::GetSubCompList_Auto1
```

Mit Hilfe von GetSpecGraphMax und SetSpecGraph2UpperT (Zeilen 21-22) wird die Adjazenzmatrix *SpecGraph* des Reaktionsgraphen $\mathcal{R}$ ermittelt. In der ersten Funktion werden alle Graphen $\mathcal{R}_K$, $K \in \mathcal{K}$, durchlaufen, das betragsmäßige Maximum der einzelnen Kanten ermittelt und in *SpecGraph* eingetragen, die zweite bildet daraus eine obere Dreiecksmatrix.
In der *while*-Schleife in den Zeilen 29 bis 90 erfolgt die Zerlegung des Graphen in seine Zusammenhangskomponenten mittels des in 4.2.1 beschriebenen Algo-

rithmus von Stoer-Wagner [60]. Dazu werden mit *CopySpecGraph* (33) die in X angegebenen n Spezies aus der Matrix $SpecGraph$ in $SubCompSpecGraph$ kopiert.
Die eigentliche Zerlegung erledigt die Funktion *MinSchnitt* (35-36), deren Algorithmus direkt aus [60] übernommen wurde. Die Schleife wird solange durchlaufen, bis entweder die kleinsten Teilgraphen nur noch ein Element enthalten oder die Kosten für eine weitere Zerlegung die zu tolerierende Schranke $epsilon_{min}$ bzw. $alpha_{max}$ überschreiten.
Die so gewonnenen Teilsysteme werden mit SetSpecList (92) in die drei Felder $SpecList$, $NSpecList$ und $SpecListOfSpecies$ übertragen.

Anhang B

Symbolverzeichnis

c Konzentration, 6
c_0 Anfangswert, 6
$\mathbf{c}$ Vektor mit den Reaktionswerten aller mobilen Spezies, 7
$\bar{\mathbf{c}}$ Vektor mit den Reaktionswerten aller immobilen Spezies, 7
C_{lin} Kontraktionszahl des modifizierten Newton-Verfahrens in linearen Fall, 32
Cr Courant-Zahl, 15

D Diffusions/Dispersions-Tensor, 6, 7
d Dimension von Ω, 5
Da Damkohler-Zahl, 15
Δt Zeitschrittweite, 10
Δt_{min} untere Schranke fuer Δt, 48
Δt_{max} obere Schranke fuer Δt, 48
$d_{\mathcal{G}}$ Grad (Valenz) eines Knotens, 13
$d_{\mathcal{G}}^{max}$ maximaler Grad eines Knotens, 13
$\partial\Omega$ Rand von Ω, 5
$\partial\Omega_C$ Randteil mit Konzentrationswerten, 6
$\partial\Omega_N$ Randteil mit Neumannwerten, 6

ϵ_{abs} absolute Fehlerschranke beim Newtonverfahren, 11
ϵ_B^{rel} Schranke zur Aufteilung des Reaktionsgraphen, 55
$\epsilon_{B,max}^{rel}$ obere Schranke fuer ϵ_B^{rel}, 55
$\epsilon_{B,min}^{rel}$ untere Schranke fuer ϵ_B^{rel}, 55
ϵ_{rel} relative Fehlerschranke beim Newtonverfahren, 12

$fact_1$ Vergroesserungsfaktor bei Zeitadaption, 47
$fact_1^B$ Vergroesserungsfaktor fuer ϵ_B^{rel}, 56
$fact_2$ Verkleinerungsfaktor bei Zeitadaption, 48
$fact_2^B$ Verkleinerungsfaktor fuer ϵ_B^{rel}, 56
f_C Konzentrations-Randwert, 6

$\mathcal{G}$ Gesamtgraph, 52
γ Anzahl der Nichtnullelemente einer Matrix, 13

$H_{0,D}^1$ kontinuierlicher Ansatzraum, 9

I Einheitsmatrix, 7

$\mathcal{J}$ Zeitintervall, 5
J Jacobi-Matrix, 11, 12, 32
J_d ausgeduennte Jacobi-Matrix, 32
J_R Rest-Jacobi-Matrix, 32

$\mathcal{K}$ Menge der Knoten, 10
κ (spektrale) Konditionszahl, 20
$\mathcal{K}_D$ Menge der Knoten auf $\partial\Omega_D$, 10
$\mathcal{K}_I$ Menge der inneren Knoten, 10
$\mathcal{K}_N$ Menge der Knoten auf $\partial\Omega_N$, 10

M Anzahl der Knoten, 10
$\mathcal{M}$ raumliches Gitter, 13

m Bandbreite einer Matrix, 25

N Anzahl der Zeitintervalle, 10
n Normalenvektor, 5
N_R Anzahl der Reaktionen, 5
N_S Anzahl der Spezies, 5
$N_{S_{imob}}$ Anzahl der immobilen Spezies, 7
NS^B_{max} obere Schranke zur Bestimmung von ϵ^{rel}_B, 56
NS^B_{min} untere Schranke zur Bestimmung von ϵ^{rel}_B, 56
$N_{S_{mob}}$ Anzahl der mobilen Spezies, 7
NS_{min} untere Schranke fuer Zeitadaption, 47
NS_{max} obere Schranke fuer Zeitadaption, 47
N_T Anzahl voneinander unabhaengiger Teilsysteme, 18

Ω Gebiet, 5

Pe Péclet-Zahl, 15
Pe_T Zell-Péclet-Zahl, 15
$\mathcal{P}_k$ Raum der Polynome vom Hoechstgrad k, 10

q (Wasser-)Fluss, 6

R Reaktionsterm, 6, 8
$\mathcal{R}$ Reaktionsgraph, 52, 53
ρ Spektralradius, 32
ρ_B Masse Boden pro Gesamtvolumen (Bulk Density), 8

T Endzeitpunkt, 5
$\mathcal{T}$ konforme Triangulierung, 10
Θ Wassergehalt, 6, 7

Zusammenfassung

Inhalt der vorliegenden Dissertation ist die Entwicklung eines modifizierten Newton-Verfahrens zur Lösung (nichtlinearer) Gleichungssysteme der Form

$$\mathbf{f}(\mathbf{x}) \;=\; 0,$$

die aus der numerischen Simulation zahlreicher Vorgänge resultieren. Dabei wird eine spezielle Klasse von Problemen, die sogenannten Mehrkomponentenmodelle, betrachtet.
Die numerische Lösung der beschriebenen Modelle führt auf ein lineares Gleichungssystem

$$A\mathbf{x} \;=\; \mathbf{b}, \qquad A \in \mathbb{R}^{n,n},\ \mathbf{x}, \mathbf{b} \in \mathbb{R}^{n},$$

das mit einem linearen Löser berechnet werden kann.
Bei Mehrkomponentenmodellen kommt es häufig vor, dass das lineare Gleichungssystem (näherungsweise) in N_T Teilsysteme

$$A_i\mathbf{x}_i \;=\; \mathbf{b}_i, \qquad A_i \in \mathbb{R}^{n_i,n_i},\ \mathbf{x}_i, \mathbf{b}_i \in \mathbb{R}^{n_i},\ \sum_{i=1}^{N_T} n_i \;=\; n,$$

zerfällt. Diese können unter Umständen mit einem geringeren Aufwand gelöst werden als das ursprüngliche Gleichungssystem.
Das in dieser Arbeit vorgestellte modifizierte Newton-Verfahren teilt das Gesamtsystem mit Hilfe geeigneter Regeln in die Teilsysteme auf und löst diese. Soweit möglich werden dabei Regeln verwendet, die theoretisch abgesichert sind und somit eine Konvergenz des Verfahrens sicherstellen. Wo dies nicht möglich oder aufgrund der höheren Rechenzeit nicht sinnvoll erscheint, wurde das neue Verfahren anhand zahlreicher Testsimulationen auf seine Praxistauglichkeit getestet.

Zu Beginn der Dissertation wird die Bedeutung von Mehrkomponentenmodellen in der Anwendung erläutert, bereits bekannte modifizierte Newton-Verfahren zu deren Lösung angesprochen und der Inhalt dieser Arbeit skizziert.

Es folgen im zweiten Kapitel die Gleichungen des als Testmodell verwendeten Mehrkomponentenproblems. Dabei handelt es sich um ein Modell zur Simulation von reaktivem Stofftransport in porösen Medien. In der vorliegenden Arbeit

wird das modifizierte Newton-Verfahren nur auf das hier vorgestellte Modell angewandt. Die Überlegungen sind jedoch für jedes Modell dieser Klasse gültig. Das neu entwickelte Verfahren ist also auf beliebige Mehrkomponentenmodelle anwendbar.
Die Modellgleichungen werden mit konformen Finiten Elementen unter Angabe der Variationsformulierungen und des zu lösenden (nichtlinearen) Gleichungssystems diskretisiert. Im nichtlinearen Fall erfolgt eine Linearisierung mit dem Newton-Verfahren und das resultierende lineare Gleichungssystem wird angegeben. Eine Diskussion der Struktur der Jacobi-Matrix und die Angabe einiger wichtiger Kenngrößen beschließen dieses zweite Kapitel.

Das dritte Kapitel beinhaltet die Vorstellung des modifizierten Newton-Verfahrens. Hier wird die Entkopplung im linearen Löser analysiert, das Konvergenzverhalten untersucht und mit Hilfe von Testsimulationen verifiziert, Vergleichssimulationen zu einem ähnlichen bereits vorhandenen Verfahren durchgeführt und ein weiteres modifiziertes Newton-Verfahren, das Broyden-Verfahren, beschrieben und getestet.
Zu Beginn werden die entkoppelten linearen Gleichungssysteme (die Teilsysteme) mathematisch definiert. Anschließend wird analysiert, ob iterative oder direkte Verfahren zu deren Lösung einen höheren Gewinn im Vergleich zur Lösung des Gesamtsystems versprechen.
Bei beiden Verfahren entsteht ein Gewinn durch eine Verringerung der Anzahl der Rechenoperationen, der aus der mit der Entkopplung verbundenen Ausdünnung der Jacobi-Matrix resultiert. Dieser fällt bei direkten Verfahren natürlich deutlich höher aus, da sie $O(n^3)$ Operationen zur Lösung des linearen Gleichungssystems benötigen, während das CG-Verfahren als Repräsentant der iterativen Verfahren nur $O(n^2)$ Rechenoperationen benötigt.
Es wird auch untersucht, ob iterative Verfahren zur Lösung der Teilsysteme weniger Iterationsschritte benötigen als zur Lösung des Gesamtsystems. Für das BiCGStab-Verfahren mit einem symmetrischen Gauß-Seidel-Verfahren als Vorkonditionierer (ein Schritt), den für die verwendeten Modelle derzeit besten verfügbaren iterativen Löser, kann diese Frage verneint werden. Eine Analyse der Konditionszahlen der beteiligten Matrizen ergibt, dass die der Teilmatrizen im Allgemeinen nicht kleiner sind als die der Gesamtmatrix. Dies ist unabhängig von den verwendeten Damköhler-Zahlen und der räumlichen und zeitlichen Schrittweiten.
Es folgt eine Betrachtung des Assemblierungsgewinns, der durch Nichtassemblierung der Kopplungsterme in der Jacobi-Matrix entsteht. Hier wird gezeigt, dass die Anzahl der zu assemblierenden Einträge unter Umständen beträchtlich sinkt. Allerdings ist bei den später erfolgenden Testsimulationen zu erkennen, dass dies keinen großen Einfluss auf die zur Assemblierung benötigte Zeit hat. Die nichtassemblierten Einträge stammen ausschließlich aus Reaktionstermen. Die Transportterme verschlingen aber den größten Anteil der Assemblierungszeit.

Der Abschnitt wird durch das Struktogramm des neu entwickelten modifizierten Newton-Verfahrens *NewtonBlock* abgeschlossen. Es erfolgen einige Erläuterungen zum Verfahren und eine Bemerkung zu den Abbruchkriterien des klassischen und des modifizierten Verfahrens. Es wird auch auf die Unterschiede des vorgestellten Verfahrens mit einem bereits bestehenden eingegangen und die Vor- und Nachteile der beiden analysiert.
Im zweiten Abschnitt des dritten Kapitels wird das Konvergenzverhalten des modifizierten Newton-Verfahrens betrachtet. Im linearen Fall kann unter Angabe der Kontraktionszahl gezeigt werden, dass bei nicht zu starker Entkopplung mindestens lineare Konvergenz vorliegt. Es wird auch ein hinreichendes Konvergenzkriterium hergeleitet und anhand einiger Testsimulationen verifiziert.
Für den nichtlinearen Fall kann ein Satz aus der Literatur zitiert werden, der die Konvergenz bei mäßiger Entkopplung sicherstellt. Er macht jedoch keine Aussage zur Konvergenzrate, so dass ein Konvergenzkriterium wie im linearen Fall hier nicht verfügbar ist. Es folgen einige nichtlineare Testsimulationen, mit deren Hilfe geprüft werden soll, ob das Konvergenzkriterium aus dem linearen Fall auf den nichtlinearen angewandt werden kann. Im Ergebnis zeigt sich, dass dies im Allgemeinen nicht zulässig ist.
Im dritten Abschnitt werden zahlreiche Vergleichssimulationen zur Dissertation [47] durchgeführt. Nach der Beschreibung des Testbeispiels wird zunächst mittels dreier Beispiele gezeigt, warum der Assemblierungsgewinn wie bereits beschrieben sehr gering ist. Dann erfolgen Simulationen mit unterschiedlichen Damköhler-Zahlen und Blockbildungen (Aufteilungen in Teilsysteme). Es werden jeweils direkte und iterative lineare Löser verwendet.
Bei den direkten Verfahren entsteht in allen Fällen ein deutlicher Rechenzeitgewinn. Allerdings führt die Verwendung eines iterativen Lösers im vollen Verfahren meist bereits zu niedrigeren Zeiten als beim direkten Löser im modifizierten. Darüber hinaus ist der Gewinn durch Entkopplung mit dem iterativen Löser sehr gering. Oft benötigt das modifizierte Verfahren sogar länger als das volle.
Dieses unerfreuliche Ergebnis wird durch veränderte Schrittweiten verbessert. Für größere Δt und kleiner h entsteht schon unter Verwendung des iterativen Lösers ein beträchtlicher Gewinn. Der direkte Löser ist teilweise noch schneller.
Das größere Einsparpotential des direkten linearen Lösers soll im letzten Abschnitt des dritten Kapitels durch Einführung des Broyden-Verfahrens noch verbessert werden. Hier wird das lineare Gleichungssystem mit einem direkten Löser berechnet. Im ersten Newtonschritt sind hierfür $O\left(n^3\right)$ Rechenoperationen nötig, in den restlichen Schritten nur $O\left(n^2\right)$.
Die angeschlossenen Beispielsimulationen zeigen allerdings, dass dieses Verfahren bei den verwendeten Modellen nur unter Verwendung sehr kleiner Zeitschrittweiten angewandt werden kann. In diesem Fall ist es zwar deutlich schneller als das klassische Newton-Verfahren mit direktem Löser, im Vergleich mit einem iterativen aber immer noch unterlegen. Der Einsatz des Broyden-Verfahrens wird deshalb nicht weiter verfolgt.

Im vierten Kapitel werden die bisher manuell vorgenommenen Einstellungen - Wahl der Zeitschrittweite und der Teilsysteme - automatisiert. Kriterien für das verwendete zeitadaptive Verfahren ist die Anzahl der Newton-Schritte. Ist sie klein, wird Δt vergrößert. Falls sie zu groß wird, erfolgt eine Verkleinerung der Zeitschrittweite. Die Funktionsweise wird wieder an den bereits bekannten Beispielen getestet.
Der zweite Abschnitt beinhaltet die Entwicklung einer automatischen Entkopplungsstrategie. Dabei wird der Begriff des Reaktionsgraphen erläutert und ein Algorithmus zu dessen Aufteilung vorgestellt. Mit Hilfe dieser Werkzeuge kann die automatisierte Entkopplung durchgeführt werden:
Zunächst sind die lokalen Reaktionsgraphen, die an den einzelnen Knoten platziert sind, auf einen globalen Reaktionsgraphen abzubilden. Dieser wird mit Hilfe des oben beschriebenen Algorithmus in zwei Teilgraphen partitioniert, so dass das Gewicht der aufgetrennten Kanten minimal ist. Falls dieses Gewicht eine vorgegebene Schranke nicht überschreitet, werden die beiden Teilgraphen weiter aufgeteilt, sonst wird die Partitionierung verworfen.
Im linearen Fall kann als Aufteilungskriterium die Kontraktionszahl aus der Konvergenzabschätzung verwendet werden. Es ist auch möglich, diese beiden Kriterien zu kombinieren.
Natürlich schließen sich Beispielsimulationen zur Demonstration der Wirksamkeit dieser Strategien an. Hier wird bereits deutlich, dass die Verwendung der Strategie mit den aufgetrennten Kantengewichten als Kriterium für die Partitionierung trotz ihrer Einfachheit sehr gut geeignet ist. Der Einsatz der Kontraktionszahl bringt kaum bessere Ergebnisse, verursacht aber deutlich höhere Rechenzeiten.

Die vorliegende Arbeit wird durch zwei praxisorientierte Simulationen abgeschlossen. Bei der ersten wird der Abbau von Cobalt-EDTA-Komplexen berechnet, beim zweiten die Verringerung einer Bodenverunreinigung mit Hilfe einer reaktiven Barriere aus Eisengranulat.
Beim EDTA-Problem findet kein Fließen statt. Hier können durch die Zeitadaption große Rechenzeitgewinne erzielt werden, während der Gewinn durch die Entkopplung nahezu Null ist. Grund hierfür ist die Dominanz der Assemblierungszeit. Generell scheint das modifizierte Newton-Verfahren für Probleme ohne Fließen weniger geeignet zu sein.
Das Barriere-Problem weist dem gegenüber ein sehr komplexes Fließgeschehen auf. Darüber hinaus entstehen in den Stoffkonzentrationen sehr große Gradienten. Der Gewinn durch das modifizierte Verfahren ist dementsprechend hoch. Auch hier zeigt sich die Wirksamkeit der oben entwickelten automatischen Entkopplungsstrategie.

Summary

The scope of this thesis is the development of a modified Newton-method for solving (nonlinear) systems of equations which are common in the solution of reactive solute transport in porous media [47, 48]

$$\partial_t \left(\Theta c_i\right) - \nabla \cdot \left(D_i \nabla c_i - \mathbf{q} c_i\right) \;=\; R_i \left(c_1, \ldots, c_{N_S}\right).$$

We consider cases where one can divide the linear system (approximately) in N_T subsystems

$$A_i \mathbf{x}_i \;=\; \mathbf{b}_i, \qquad A_i \in \mathbb{R}^{n_i, n_i},\; \mathbf{x}_i, \mathbf{b}_i \in \mathbb{R}^{n_i},\; \sum_{i=1}^{N_T} n_i \;=\; n,$$

that can be solved faster than the original system of equations.
The modified Newton method introduced in this thesis leads to smaller subsystems of equations. As far as possible convergence of the new method is ensured by convergence monitoring. To identify situations where this is not possible or due to the computational costs not useful the developed rules are sufficiently tested.

In the introduction the importance of the above multicomponent models in applied sciences is explained, informations about existing modified Newton methods are given and the content of this work is commented.

The second chapter starts with a presentation of the model equations. They describe multicomponent transport problems in porous media. This is the test model for the modified Newton method, which can be used for arbitrary multicomponent models.
A discretization with conforming finite elements follows. This section includes the variational formulations and the resulting (nonlinear) system of equations. In case of nonlinearity it is linearized with Newton's method and the linear system of equations is given. A discussion of the structure of the Jacobian and definition of some important indicators (Péclet-, Damköhler- and Courant-number) conclude this chapter.

In the beginning of the third chapter the decoupled linear systems of equations are defined. Then it is analyzed whether iterative or direct solvers promise a higher acceleration of the computation compared to the original model.

A faster performance arises in both iterative and direct solvers because of a reduction of computations due to the underlying neglection of elements in the Jacobian. While direct solvers need $O(n^3)$ operations to solve the system, CG-solvers need only $O(n^2)$. Therefore one gets a bigger acceleration by using the first ones.
It is also considered whether iterative solvers need less steps after decoupling than before. This is not the case for a BiCGStab-solver with symmetric Gauß-Seidel pre-conditioning (one step), which is the best iterative solver available. Analysis of the condition numbers of the single matrices shows that the sub-matrices do not have smaller ones than the original Jacobian. This fact does not depend on the Damköhler-numbers or the step-sizes.
A consideration of the gain in assembling time follows. It arises because entries in the Jacobian coupling the subsystems can be neglected. Although the number of neglected entries may be quiet high the assembling time remains quite the same as before decoupling. This is due to only reaction-terms can be neglected, but unfortunately the transport terms dominate assembling time.
The section ends with a listing of the modified Newton-method's structure chart. This subsection includes some explanations and a note on the breaking criteria of original and modified Newton methods. The modified method uses

$$\|\mathbf{b}_i\| \;\leq\; \epsilon_{abs}\frac{n_i}{N_S} \;=:\; \epsilon_i, \qquad i \in \{1,\ldots,N_S\},$$

so that the breaking criteria of the original method

$$\|\mathbf{b}\| \;\leq\; \epsilon_{abs}$$

is still fulfilled.
The second section of this chapter includes some discussions about the convergence behavior of the modified method. In the linear case it can be shown that it converges at least linear, if decoupling is not too strong. The contraction number and an adequate criterion for convergence are given. These results are verified in some test-simulations.
In the case of nonlinearity one can cite a well-known lemma from literature which ensures convergence.Here the convergence rate is not known. Test-simulations indicate that the convergence result from the linear case can't be used for the nonlinear one.
The numerical simulations of the third section show that the acceleration of direct solvers is much higher. On the other hand iterative solvers are often able to solve the full system faster than direct solvers the reduced one. Moreover the gain in computation time by decoupling with iterative solvers is very small. In many cases the full method is even faster than the modified one because of the increased number of iterations.
To deal with this problem the step-sizes are changed. With bigger Δt and smaller h the acceleration of the iterative solver is much higher than before. Sometimes direct solvers are even faster.

The bigger potential of direct solvers shall still be improved in the last section of the third chapter by using Broyden's method where the Jacobian is assembled only once per time step. In the following steps it is approximated by a rank-1-update. Here the unknowns of the linear system of equations are computed with direct solvers. For the first Newton step one needs $O(n^3)$ operations, for the others only $O(n^2)$.
The following examples show that this method can only be applied in combination with very small Δt. It is considerably faster than the full method with a direct solver. But iterative solvers are still faster. Therefore Broyden's method will not be used in the following chapters.

So far the choice of Δt and the subsystems had to be defined prior to the simulation. Now this should be done automatically. The number of Newton steps (NS) is used as criterion for time adaption. If NS is sufficiently small, Δt can be increased. Should it become to big, Δt has to be decreased. The method is tested with the well-known examples.
The second section contains the development of an automated decoupling strategy. The term reaction graph is defined and an algorithm for its partitioning is presented.
As a first step local graphs connected with nodes of the spatial mesh have to be mapped to a global one. The above algorithm is used to divide this graph in two subgraphs, such that the weight of neglected edges is minimal. If this weight is smaller than a given upper limit both subgraphs are divided further. Otherwise the division is aborted.
In the case of linear models the contraction number of the convergence rate can be used as criterion for decoupling. Combination of both criteria is also possible. The strategies are tested with numerous simulations which demonstrate that the first strategy is despite its simplicity very efficient. The use of the contraction number leads to slightly better performances but needs significant higher computation times.

The dissertation ends with two applied simulations. The first one deals with the degradation of Cobalt-EDTA-complexes, the second one with the reduction of contaminations with the help of a reactive barrier composed of granular iron.
In the EDTA-problem no flow takes place. Time adaptation gives great acceleration of computation time (it reduces up to factor 35), while decoupling is nearly ineffective. This behavior is an outcome of the small dynamic of the reactions that leads to very big time steps and a dominant assembling time. In cases like this one can not expect a considerable gain in computation time by the modified method.
In contrast to this, the barrier-problem has a quiet complicated flow regime. Additionally the gradients of the concentrations have very big values. Therefore the solution of this problem is more demanding than the EDTA case. As a

consequence performance of modified Newton method is much better. The linear subsystems can be solved about 3 times as fast as the original one, the total computation time speeds up by a factor 2.

Literaturverzeichnis

[1] Alexander, M., Socow, K.M.: *Kinetics of biodegradation in soil.* In Sawhney, B.L., Brown, K. (editors), Reactions and Movement of Organic Chemicals in Soils, Soil Science Society of America (1989), 243-269.

[2] Aschenbrenner, L., Dinkler, D.: *Time dependent behaviour of asphalt modelled by application of fractional calculus,* In Proceedings 1st IFAC Workshop on Fractional Differentiation and its Applications, ENSIRB Bordeaux (2004).

[3] Bannett, T.A.: *An in-situ reactive barrier for the treatment of hexavalent chromium and trichloroethylene in groundwater,* M.Sc. thesis, Univ. of Waterloo, Waterloo, Ont., Canada (1997).

[4] Bause, M., Knabner, P.: *Numerical simulation of contaminant biodegradation by higher order methods and adaptive time stepping,* Computing and Visualization in Science 78 (2004), 39-61.

[5] Bear, J.: *Dynamics of Fluids in Porous Media,* Dover, New York (1972).

[6] Bethke, C. M.: *Geochemical Reaction Modeling,* Oxford University Press, New York (1996).

[7] Blowes, D.W., Mayer, K.U.: *An in-situ permeable reactive barrier for the treatment of hexavalent chromium and trichloroethylene in groundwater, vol. 3, Reactive transport modelling,* Tech. Rep. EPA/600/R-99/095c, U.S. Environ. Prot. Agency, Washington, D. C. (1999).

[8] Borden, R.C., Bedient, P.B.: *Transport of dissolvend hydrocarbons influenced by oxygen-limited biodegradation, 1. Theoretical development,* Water Resources Research 22(13) (1986), 1973-1982.

[9] Borrmann, T.: *Ein hydrodynamisches 3D-Mehrkomponentenmodell der Heliosphäre und ihrer Wechselwirkung mit kosmischer Strahlung,* Ruhr-Universität Bochum, Fakultät für Physik und Astronomie (2005).

[10] Braess, D.: *Finite Elemente,* 2. Auflage, Springer, Berlin (1997).

[11] Brezzi, F., Fortin, M.: *Mixed and Hybrid Finite Element Methods,* volume 15 of Springer Series in Computational Mathematics, Springer, New York (1991).

[12] Broyden, C.G.: *A class of methods for solving nonlinear simultaneous equations,* Math. Comp. 19 (1965), 577-583.

[13] Brun, A., Engesgaard, P.: *Modelling of transport and biogeochemical processes in pollution plumes: literature review and model development,* Journal of Hydrology 256 (2002), 211-227.

[14] Chapelle, F.H.: *Ground-Water Microbiology and Geochemistry,* second edition, John Wiley and Sons, New York (2001)

[15] Chilakapati, A., Ginn, T., Szecsody, J.: *An analysis of complex reaction networks in groundwater modelling,* Water Resources Research 34(7) (1998), 1767-1780.

[16] Curtis, G.P.: *Comparison of approaches for simulating reactive solute transport involving organic degradation reactions by multiple terminal electron acceptors,* Computers and Geosciences 29 (2003), 319-329.

[17] Cuthill E., McKee J.: *Reducing the bandwidth of sparse symmetric matrices,* Proc. 24th Nat. Conf. ACM (1969), 157-172.

[18] de Marsily, G.: *Quantitative Hydrogeology,* Academic Press, San Diego (1986).

[19] Demmel, J.W.: *Applied numerical linear algebra,* SIAM (1997).

[20] Deuflhard, P.: *Newton Methods for Nonlinear Problems,* Springer, Berlin, Heidelberg (2004).

[21] Diestel, R.: *Graphentheorie,* Springer, Berlin (1996).

[22] Domenico, P.A., Schwartz, F.W.: *Physical and Chemical Hydrogeology,* John Wiley & Sons, second edition, New York (1998).

[23] Duff I., Erisman A.M., Reid J.: *Direct Methods for Sparse Matrices,* Oxford University Press (Calderon Press), Oxford (1986).

[24] Evans, L.C.: *Partial Differential Equations,* American Mathematical Society, Providence (1998).

[25] Fang, Y., Yeh, G.T., Burgos, W.D.: *A general paradigm to model reaction-based biogeochemical processes in batch systems,* Water Resources Research 39(4) (2003), 1083.

[26] Findeiß, R.E.: *Ein orts- und zeitadaptives Finite-Elemente-Verfahren zur Traglastanalyse wassergesättigter Böden,* Dissertation, Lehrstuhl für Statistik der TU München (2001).

[27] Friedly, J.C., Rubin, J.: *Solute transport with multiple equilibrium-controlled or kinetically controlled reactions,* Water Resources Research 28(6) (1992), 1935-1953.

[28] Golub, G.H., Van Loan, C.F.: *Matrix Computations,* 3rd edition, Johns Hopkins University Press, Baltimore (1996).

[29] Hackbusch, W.: *Iterative Lösung großer schwachbesetzter Gleichungssysteme,* 2. Auflage, B.G. Teubner, Stuttgart (1993).

[30] Hackbusch, W.: *Multi-Grid Methods and Applications,* 2. Auflage, Springer, Berlin (1985).

[31] Häfner, F., Dietrich, S., Voigt, H.D.: *Wärme- und Stofftransport,* Springer, Berlin (1992).

[32] Helmit, Rainer: Institut für Wasserbau, www.co2sink.org.

[33] Hoffmann, J.: *Ein Entkopplungsverfahren für Systeme von Transportreaktionsgleichungen in porösen Medien: Algorithmische Umsetzung und Simulation realistischer 2D-Szenarien* Diplomarbeit, Erlangen (2005).

[34] Holstad, A.: *A mathematical and numerical model for reactive fluid flow systems,* Computational Geosciences 4 (2000), 103-139.

[35] Hundsdorfer, W., Verwer, J.: *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations,* Springer, Berlin (2003).

[36] Kelley, C.T.: *Iterative Methods for Linear and Nonlinear Equations,* SIAM, Philadelphia (1995).

[37] Knabner, P., Angermann, L.: *Numerical Methods for Elliptic and Parabolic Partial Differential Equations,* Springer, New York (2003).

[38] Knabner, P.: *Mathematische Modelle für Transport und Sorption gelöster Stoffe in porösen Medien,* Peter Lang, Frankfurt/Main (1991).

[39] Knabner, P.: *The modeling of reactive solute transport with sorption to mobile and immobile sorbents; 1. Experimental evidence and model development,* Water Resources Research 32(6) (1996), 1611-1622.

[40] Li, X.S., Demmel, J.W.: *SuperLU_DIST: A Scalable Distributed-Memory Sparse Direct Solver for Unsymmetric Linear Systems,* ACM Trans. Mathematical Software Vol 29, No 2 (June 2003), 110-140.

[41] Li, X.S., Demmel, J.W., Gilbert J.: *http://crd.lbl.gov/∼xiaoye/SuperLU/,* 2007.

[42] Lichtner, P.C.: *Reactive Transport in Porous Media,* Reviews in Mineraloty 34. Mineralogical Society of America, Washington (1996), 1-81.

[43] Lichtner, P.C., Steefel, C.I., Oelkers, E.H.: *Reactive Transport in Porous Media,* Reviews in Mineralogy 34, Mineralogical Society of America, Washington (1996).

[44] Mayer, K.U.: *A numerical model for multicomponent reactive transport in variably saturated poroous media,* Ph.D. thesis, Dep. of Earth Sci., Univ. of Waterloo, Waterloo, Ont., Canada (1999).

[45] Mayer, K.U., Blowes, D.W., Frind, E.O.: *Reactive transport modeling of an in situ reactive barrier for the treatment of hexavalent chromium and trichloroethylene in groundwater,* Water Resources Research 37(12) (2001), 3091-3103.

[46] Oßmann, S.: *Mathematische Modellierung und numerische Simulation von präferenziellem Fließen mit Stofftransport in strukturierten porösen Medien,* Diplomarbeit, Erlangen (2004).

[47] Prechtel, A.: *Modelling and Efficient Numerical Solution of Hydrogeochemical Multicomponent Transport Problems by Process-Preserving Decoupling Techniques,* Dissertation, Erlangen (2005).

[48] Radu, F.: *Mixed finite element discretization of Richards' equation: error analysis and application to realistic infiltration problems,* Dissertation, Erlangen (2004).

[49] Rittmann, B.E., Van Briesen, J.M.: *Reactive Transport in Porous Media,* Reviews in Mineralogy 34. Mineralogical Society of America, Washington (1996), 311-334.

[50] Redl, C.: *Analysis and Application of Multi-Dimensional Debris Flow Models,* Diplomarbeit, Montanuniversität Leoben (1999).

[51] Robinson, B.A., Viswanathan, H.S., Valocchi, A.J.: *Efficient numerical techniques for modelling multicomponent ground-water transport based upon simultaneous solution of strongly coupled subsets of chemical components,* Advances in Water Resources Vol 23 (2000), 307-324.

[52] Rubin, J.: *Transport of reacting solutes in porous media: Relation between mathematical nature of problem formulation and chemical nature of reactions,* Water Resources Research 19(5) (1983), 1231-1252.

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

[54] Schwarz, H.R.: *Numerische Mathematik,* 4. Auflage, B.G. Teubner, Stuttgart (1997).

[55] Scheidegger, A.E.: *General theory of dispersion in porous media,* Journal of Geophysical Research, 66 (1961), 3273-3278.

[56] Schirmer, M., Molson, J. W., Frind, E. O., Barker, J. F.: *Biodegradation modelling of a dissolved gasoline plume applying independent laboratory and field parameters,* Journal of Contaminant Hydrology, Vol 46 (2000), 339-374.

[57] Schneid, E.: *Hybrid-Gemischte Finite-Elemente-Diskretisierung der Richards-Gleichung,* Dissertation, Erlangen (2000).

[58] Steefel, C.I., MacQuarrie, K.T.B.: *Reactive Transport in Porous Media,* Reviews in Mineralogy 34. Mineralogical Society of America, Washington (1996), 83-129.

[59] Szecsody, J.E., Zachara, J.M., Chilakapati, A., Jardine, P.M., Ferrency, A.S.: *Importance of flow and particle-scale heterogeneity on co(ii/iii)edta reactive transport,* Journal of Hydrology 209 (1998), 112-136.

[60] Turao, V.: *Algorithmische Graphentheorie,* Addison-Wesley, Bonn (1996).

[61] Weigand, H., Totsche, K.U.: *Flow and reactivity effects on dissolved organic matter transport in soil columns,* Soil Sci. Soc. Am. J. 62 (1998), 1269-1274.

[62] Widdowson, M. A., Molz, F. J., Benefield, L. D.: *A numerical transport model for oxygen- and nitrate-based respiration linked to substrate and nutrient availability in porous media,* Water Resources Research, Vol 24, No 9 (1988), 1553-1565.

[63] Wieners, C.: *Parallele Finite Elemente,* http://www.mathematik.uni-karlsruhe.de/user/∼ipm/Lehre/06/pfem.pdf (2000).

[64] Yeh, G.G., Tripathi, V.S.: *A model for simulating transport of reactive multispecies components: Model development and demonstration,* Water Resources Research 27(12) (1991), 3075-3094.

[65] Zysset, A., Stauffer, F., Dracos, T.: *Modeling of chemically reactive groundwater transport,* Water Resources Research 30(7) (1994), 2217-2228.

Curriculum Vitae

Name:	Stephan Oßmann
Geburtsdatum:	18. März 1974
Geburtsort:	Lichtenfels
Familienstand:	ledig
Staatsangehörigkeit:	deutsch

Schulausbildung:

09/1980 - 07/1986	Grund- und Hauptschule Unnersdorf/Grundfeld
09/1986 - 07/1990	Staatliche Realschule Staffelstein (kaufmännischer Zweig)
Juni 1990	Abschluss mit Mittlerer Reife
02/1994 - 07/1994	Vorbereitungskurs Fachoberschule Bamberg
09/1994 - 06/1995	Fachoberschule Bamberg (technischer Zweig)
Juni 1995	Abschluss mit Fachhochschulreife

Berufsausbildung:

09/1990 - 02/1994	Ausbildung zum Kommunikationselektroniker beim Fernmeldeamt Bamberg
Februar 1994	Abschluss mit Facharbeiterprüfung

Zivildienst:

07/1995 - 07/1996	Werkstätten für Behinderte Lichtenfels

Studium:

10/1996 - 08/2000	Studium der Mathematik an der Fachhochschule Regensburg (Schwerpunkt Technik)
August 2000	Abschluss als Diplom-Mathematiker (FH)

10/2000 - 03/2001	Studium der Mathematik an der Universität Kassel
04/2001 - 10/2004	Studium der Mathematik an der Friedrich-Alexander-Universität Erlangen-Nürnberg (Schwerpunkt Numerik partieller Differentialgleichungen)
Oktober 2004	Abschluss als Diplom-Mathematiker Univ.
seit 11/2004	Promotionsstudium am Institut für Angewandte Mathematik der Friedrich-Alexander-Universität Erlangen-Nürnberg

Praktika:

10/1997 - 02/1998	Prototyperstellung eines firmeneigenen INTRANETS Loewe Opta GmbH, Kronach
03/1999 - 07/1999	Erstellen von Analysetools zur Auswertung von Messdateien aus Vermittlungsstellen in C++ Deutsche Telekom AG, Niederlassung Bamberg

Berufliche Tätigkeiten:

08/2003 - 09/2004	Studentische Hilfskraft am Institut für Angewandte Mathematik der Friedrich-Alexander-Universität Erlangen-Nürnberg
seit 11/2004	Wissenschaftlicher Mitarbeiter am Institut für Angewandte Mathematik der Friedrich-Alexander-Universität Erlangen-Nürnberg