Institut für Parallele und Verteilte Systeme
Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit Nr. 27

Robuste Multilevel-Lösung elliptischer partieller Differentialgleichungen mit springenden Koeffizienten

Klaudius Scheufele

Studiengang:	Informatik
Prüfer:	Prof. Dr. rer. nat. Marc Alexander Schweitzer
Betreuer:	Dr. rer. nat. Stefan Zimmer
begonnen am:	16. Juli 2012
beendet am:	16. Januar 2013
CR-Klassifikation:	G.1.0 , G.1.3, G.1.8, G.4

Kurzfassung

In dieser Arbeit wird eine robuste Multilevel-Lösung für elliptische partielle Differentialgleichungen mit springenden Koeffizientenfunktionen im Kontext der Partition of Unity Methode realisiert und analysiert. Bei dieser gitterfreien Methode müssen die Koeffizientensprünge nicht auf dem gröbsten Level geometrisch aufgelöst sein, vielmehr kann durch geeignete Anreicherungsfunktionen mittels algebraischer Verfeinerung die Approximationsqualität verbessert werden. Die Implementierung eines stabilen und robusten Multilevel-Lösers sowie die Realisierung verschiedener Anreicherungsfunktionen sind Kernbereich dieser Arbeit. Insbesondere werden verschiedene Anreicherungsfunktionen entwickelt und deren Auswirkung im Hinblick auf die Robustheit des Lösers untersucht. Mögliche Ursachen für nicht robustes Verhalten des Lösers werden in diesem Zusammenhang detailliert diskutiert und Ansätze für Verbesserungen gegeben. Der realisierte Multilevel-Löser zeigt im Vergleich zu früheren Arbeiten effizienteres und für viele Fälle durchaus robustes Verhalten. Einfache Distanzfunktionen führen dabei i. A. zu den besten Ergebnissen jedoch lässt sich durch alleinige Verbesserung der Approximationsqualität der lokalen Anreicherungsräume die Effizienz und Robustheit des Lösers aufgrund schlechterer Glättungseigenschaften nicht beliebig steigern.

Inhaltsverzeichnis

Notationen

Lateinisches Alphabet:

$a(\cdot,\cdot)$	Bilinearform (vgl. S. 7).
A_l	Repräsentation der Bilinearform auf $\mathcal{V}_l$ (vgl. S. 11).
$A_{(i,n),(j,m)}$	Blockeintrag der Steifigkeitsmatrix (vgl. S. 7).
B	Symbol für Randdifferentialoperator (vgl. S. 6).
$C_\Omega,\ C_{\Omega,k}$	Überdeckung (Cover) des Gebiets Ω bzw. Überdeckung auf Level k (S. 3, 13).
$C_\infty,\ C_\nabla$	obere Schranken an $\|\varphi_i\|$ bzw. $\|\nabla\varphi_i\|$ (vgl. S. 5).
C_i	geometrische Nachbarschaft des Patches ω_i (vgl. S. 4).
$C^G_{j,k-1,k}$	geometrische interlevel Nachbarschaft des Patches $\omega_{j,k-1}$ (vgl. S. 15).
$C^H_{j,k-1,k}$	hierarchische interlevel Nachbarschaft des Patches $\omega_{j,k-1}$ (vgl. S. 16).
$C^H_{i,k,k-1}$	hierarchische interlevel Nachbarschaft des Patches $\omega_{i,k}$ (vgl. S. 16).
$\mathcal{C}^B_{i,k}$	Baumzelle für Patch $\omega_{i,k}$ auf Level k des **k**-d-Baumes (vgl. S. 14).
c^k_M	konstanter Wert des Diffusionskoeffizients in Ω_k (vgl. S. 6).
$\mathcal{C}(A_k),\ \mathcal{C}(I^k_{k-1}),\ \mathcal{C}(I^{k-1}_k)$	nicht Null Einträge der Matrizen $A_k, I^k_{k-1}, I^{k-1}_k$ pro Zeile (vgl. S. 12).
d_k	Defekt bzw. Residuum auf Level k (vgl. S. 11).
$d(\Omega_i,\Omega_j)$	Achsenparalleler Abstand zwischen den Materialeinschlüssen Ω_i und Ω_j (S. 42).
$\mathcal{E}_i$	lokaler Raum der Anreicherungsfunktionen auf Patch ω_i (vgl. S. 5).
$f,\ \hat{f}$	kontinuierliche bzw. diskrete rechte Seite einer PDGl.
$f_{(i,n)}$	Blockeintrag des diskreten rechte Seite Vektors (vgl. S. 7).
$g,\ g_D,\ g_N$	Randdaten; Dirichlet- bzw. Neumann Randdaten eines Randwertproblems.
$I^k_{k-1},\ I^{k-1}_k$	interlevel Transferoperatoren: Prolongation bzw. Restriktion (vgl. S. 11).
$\mathbb{I}$	Einheitsmatrix.
J	Index des Levels mit der feinsten Diskretisierung.
L	elliptischer Differentialoperator 2. Ordnung (vgl. S. 6).
$\mathcal{L}^{\hat{s}_l}_i$	univariates Legendrepolynom vom Grad $\hat{s}_l$ auf dem Patch ω_i (vgl. S. 5).
M^{ν_1,ν_2}_γ	Multilevel-Verfahren mit $\nu_1,\ \nu_2$ Vor- bzw. Nachglättungsschritten (vgl. S. 11).
$M^k_k,\ M^k_{k-1}$	Massenmatrix auf Level k bzw. interlevel Massenmatrix mit globaler Basis (S. 15).
$\widetilde{M}^k_k,\ \widetilde{M}^k_{k-1}$	lokale Massenmatrix auf Level k bzw. lokale interlevel Massenmatrix (S. 16).
$\widehat{M}^k_{k-1}$	interlevel Massenmatrix mit globaler Groblevel-Basis, lokaler feinlevel Basis (S. 15).
$\overline{M}^k_k$	lumped Massenmatrix auf Level k (lokale Massenmatrix mit φ_i-Gewichtung) (S. 16).
$N,\ N_k$	Anzahl der unabh. Punkte x_i als Basis für das Cover (auf Level k).
N^{dof}_k	Anzahl der Unbekannten der Diskretisierung auf Level k (vgl. S. 12).
$\mathcal{O}(\cdot)$	Landau Symbol.
P	Punktemenge der x_i als Basis für den gitterfreien Ansatz.
$\mathcal{P}^{p_i}$	lokaler glatter Ansatzraum mit Polynomen vom Grad $\leq p_i$ (vgl. S. 5).
$\mathcal{Q}_k$	Gebiet $\Omega_k \subsetneq \mathcal{Q}_k \subsetneq \omega^{enr}_k$ auf dem die EPU-Funktion φ^{enr}_k gleich 1 ist (vgl. S. 28).
$S^{pre}_k,\ S^{post}_k$	Vor- bzw. Nachglätter für den Multilevel Löser (vgl. S. 11).
$u, \tilde{u}$	kontinuierliche Lösung der PDGl bzw. Koeffizientenvektor für die diskrete Lösung.
u^{PU}, V^{PU}	Funktion aus dem Partition of Unity Ansatzraum V^{PU}.
V^{pi}_i	lokaler Ansatzraum auf dem Patch ω_i (vgl. S. 5).
$W_i,$	lokale Gewichtsfunktion zur Konstruktion der Partition der Eins (vgl. S. 5).
$\mathcal{W}_i$	Referenzgewichtsfunktion zur Auswertung auf dem Intervall $[0,1]$ (vgl. S. 5).

Griechisches Alphabet:

α	Stretchfaktor für Baumzellen (vgl. S. 14).
γ	Anzahl der rekursiven Aufrufe beim Multilevel-Verfahren (vgl. S. 11).
γ^{blend}	Exponent der blended Enrichment Funktionen, reguliert die Glattheit des Übergangs (S. 27).
Γ, Γ_D, Γ_N	Rand des Gebiets Ω bzw. Teilrand mit Dirichlet- bzw. Neumann Randbedingungen (S. 6).
Γ_M, Γ_M^k	Materialinterface bzw. Interface mit Material k (vgl. S. 6).
Δ	Laplace Operator (vgl. S. 6).
δ^{blend}	Durchmesser des Schlauchs, innerhalb dessen η^{blend} Werte zwischen 0 und 1 annimmt (S. 27).
δ^{PU}	definiert Breite des Überlapps $\omega_k^{enr} \setminus \mathcal{Q}_k$ für die η^{PU} (S. 28).
ϵ	Wert des Diffusionskoeffizienten κ für betrachteten Einschluss (Sprunghöhe) (vgl. S. 36).
η_i^t	lokale Anreicherungsfunktion auf dem Patch ω_i (vgl. S. 5).
η_-, η_+	einfache Distanzfunktionen $\text{sdist}(x, \Gamma_M)$ als Anreicherungsfunktionen (vgl. S. 26).
η_{k^+}	einfache Distanzfunktionen $\text{sdist}(x, \Gamma_M^k)$ als Anreicherungsfunktionen (vgl. S. 26).
η_-^{blend}, $\eta_{k^+}^{blend}$	glatt auf 0 bzw. 1 fortgesetzte Anreicherungsfunktionen (vgl. S. 28).
$\eta_{j,k^\pm}^{pu}$	Partition of Unity- Anreicherungsfunktionen (vgl. S. 29).
ϑ_i^n	Basisfunktionen der lokalen Ansatzräume V_i^{pi} (vgl. S. 5).
κ	(nicht stetige) Koeffizientenfunktion (vgl. S. 6).
$\kappa(A)$	Kondition der Matrix A (vgl. S. 13).
ν_1, ν_2	Anzahl der Vor- bzw. Nachglättungsschritte des Multilevel Lösers (vgl. S. 11).
$\Pi_W^{\widetilde{W}}$	Allgemeiner L^2-Projektionsoperator (vgl. S. 14).
Π_{k-1}^k	Globale L^2-Projektion als Prolongationsoperator (vgl. S. 14).
$\widehat{\Pi}_{k-1}^k$	Global-nach-Local L^2-Projektion als Prolongationsoperator (vgl. S. 15).
$\widetilde{\Pi}_{k-1}^k$	Local-nach-Local L^2-Projektion als Prolongationsoperator (vgl. S. 16).
$\overline{\Pi}_{k-1}^k$	gewichtete Global-nach-Local L^2-Projektion als Prolongationsoperator (vgl. S. 16).
φ_i	Partition of Unity Funktionen für den PUM-Ansatzraum (vgl. S. 4).
φ_k^{enr}	Partition of Unity Funktionen (EPU) für PU-Anreicherungsfunktionen (vgl. S. 28).
ψ_i^s	Basisfunktionen der polynomiellen Ansatzräume $\mathcal{P}^{p_i}$ (vgl. S. 4).
ω_i, $\omega_{i,k}$	Patch der Überdeckung C_Ω bzw. $C_{\Omega,k}$ (vgl. S. 3, 13).
ω_i^{enr}	Patch der speziellen Überdeckung für die PU-Anreicherungsfunktionen in Kapitel 4.1.1 (S. 28).
Ω, $\overline{\Omega}$	Gebiet der Randwertaufgabe, Teilmenge des $\mathbb{R}^d$ bzw. Abschluss.
Ω_0, Ω_k	Gebiet des Matrix-Materials bzw. Gebiete der Materialeinschlüsse $k = 1, \cdots, m$ (vgl. S. 6).
Ω_+, Ω_-	Vereinigung der Materialeinschlüsse Ω_i bzw. Gebiet des Matrix-Materials (vgl. S. 26).

Abbildungsverzeichnis

Tabellenverzeichnis

KAPITEL 1

Einleitung

Die numerische Simulation ist unverzichtbarer Bestandteil in Entwicklung und Forschung und hält zunehmend Einzug in einer Vielfalt von Bereichen: Von Strömungs-, Verkehrs- und Fußgängersimulation über Materialfestigkeits- und Molekularsimulation bis hin zu Wettersimulationen, um nur einige zu nennen. Das physikalische Verhalten wird dabei in einem mathematischen Modell meist durch partielle Differentialgleichungen (PDGl) beschrieben. Zur Lösunug dieser partiellen Differentialgleichungen sind vor allem drei klassische Ansätze sehr genau untersucht und verstanden: Finite Differenzen Verfahren (FDM), Finite Volumen Verfahren (FVM) sowie Finite Elemente Verfahren (FEM). Insbesondere die Finite Elemente Methode findet in der Praxis einen sehr breiten Anwendungsbereich. Allen Verfahren gemeinsam ist jedoch die Verwendung eines zugrundeliegenden Gitters, dessen Berechnung meist einen dominierenden Faktor im Lösungsprozess der PDGl darstellt - im Besonderen bei der Finiten Elemente Methode. Dies motiviert die Entwicklung und Untersuchung sogenannter *gitterfreier Methoden*, die auf Basis einer unabhängigen Menge von Punkten arbeiten. Ein Vertreter dieser Art ist die von M. A. Schweitzer entwickelte Partikel-Partition of Unity Methode (PPUM) ([14], [15], [17], [5]), in deren Kontext die Untersuchungen dieser Arbeit stehen. Wesentlicher Vorteil dieser Methode ist, dass bereits apriori bekanntes Wissen über die Lösung gewinnbringend in Form von Anreicherungsfunktionen in den Lösungsprozess eingebracht werden kann.

Bei der Simulation von Materialien völlig unterschiedlicher Eigenschaften – beispielsweise die Wasserdurchlässigkeit von Erde im Gegensatz zu Lehm oder Gestein in der Simulation von Grundwasserströmungen – treten aufgrund dieses sog. Koeffizientensprungs am Materialübergang Schwierigkeiten auf, die die Lösung des Problems ungleich schwerer machen. In dieser Arbeit soll nun ein robuster Multilevel-Löser im Kontext der *Partition of Unity Methode* für das im Lösungsprozess entstehende große lineare Gleichungssystem entwickelt und realisiert werden. Für klassische Verfahren muss üblicherweise angenommen werden, dass die stückweise definierte Koeffizientenfunktion bereits auf dem gröbsten Level vollständig aufgelöst ist. Durch

Verwendung passender Anreicherungsfunktionen muss diese Annahme in der PUM jedoch nicht getroffen werden.
Das zentrale Problem hierbei ist die Robustheit des Lösers für beliebige, nichtstetige Koeffizientenfunktionen, d. h. die Effizienz des Lösers ist i. A. abhängig von der Koeffizientenfunktion. Kernbereich dieser Arbeit ist, neben der Implementierung des Lösers und dessen Einbettung in das neu entwickelte PUM-Framework Crass, die Analyse desselben im Hinblick auf die Robustheit für ausgewählte, kritische Problemfälle. Insbesondere soll dabei die Verwendung verschiedener Anreicherungsfunktionen und deren Auswirkungen auf die Robustheit des Lösers untersucht werden. Dabei wird nahtlos an die Untersuchungen von M. A. Schweitzer in [19] angeknüpft und die dort angestellten Überlegungen vertieft.
Insbesondere stellt sich die Frage, wie robust sich der Multilevel-Löser bezüglich einerseits durchaus komplexer Geometrie der Koeffizientensprünge und andererseits deren Höhe verhält. Weiterhin sollen Konfigurationen von mehreren Einschlüssen mit beliebigen Abständen zueinander untersucht werden.

Gliederung

Die Arbeit ist in folgender Weise gegliedert:

Kapitel 2 – Partition of Unity Methode: In diesem Kapitel werden wir die benötigten Grundlagen und eine Einführung zu der Partition of Unity Methode nach M. A. Schweitzer vorstellen.

Kapitel 3 – Multilevel-Löser: Sowohl die allgemeine Funktionsweise eines Multilevel-Lösers als auch dessen Umsetzung im Kontext der PUM werden in diesem Kapitel diskutiert. Dabei wird insbesondere auf die verwendeten und möglichen interlevel-Transferoperatoren eingegangen. In diesem Zusammenhang wird ein neuer Transferoperator vorgestellt und dessen Qualität untersucht. Ein Abschnitt zur Klärung des Begriffs der Robustheit eines Lösers, gefolgt von der Präsentation der konkreten Umsetzung und Implementierung des Löseres im Crass Code schliesen das Kapitel ab.

Kapitel 4 – Anreicherungsfunktionen: Hier werden die untersuchten Anreicherungsfunktionen definiert und deren Eigenschaften diskutiert. Die Verwendung von Anreicherungsfunktionen verlangt die Transformation des Systems in eine stabile Basis. Die Idee dieser Transformation sowie Details zur Implementierung bilden den zweiten Teil dieses Kapitels.

Kapitel 5 – Untersuchung der Robustheit des Multilevel-Lösers: In diesem Kapitel werden die Ergebnisse der Robustheitsanalyse ausgewählter Probleme präsentiert und diskutiert. Ursachen für nicht robustes Verhalten sowie mögliche Verbesserungsansätze werden erörtert und die Auswirkung verschiedener Anreicherungsfunktionen interpretiert.

Kapitel 6 – Zusammenfassung und Ausblick: Schließlich werden die gewonnenen Erkenntnisse und Resultate zusammengefasst und wesentliche Aspekte herausgearbeitet. Außerdem werden eine Reihe von Anknüpfungspunkten und Motivation für weitere Forschungsarbeiten gegeben.

KAPITEL 2

Partition of Unity Methode

Als Vertreter der gitterfreien Methoden zur Lösung partieller Differentialgleichungen verzichtet die *Partition of Unity Methode* (PUM) [14] auf den teuren und dominierenden Prozess der Gittergenerierung. Stattdessen bildet eine Menge von endlich vielen, voneinander unabhängigen Punkten $x_i,\ i = 1, \cdots N$ die Grundlage für die Diskretisierung der PDGl. Wesentlicher Bestandteil des Ansatzes ist eine endliche Überdeckung (Cover) $C_\Omega = \{\omega_i\}$ des Simulationsgebiets Ω, bestehend aus Flächen- (2D) bzw. Volumenobjekten (dD), den sog. Patches ω_i. Auf Grundlage dieser Überdeckung wird eine Partition der Eins berechnet, die voneinander unabhängige, lokal auf den Patches definierte Ansatzräume zu einem globalen Ansatzraum V^{PU} zusammenfasst. Ein wesentlicher Vorteil der Methode liegt in der unabhängigen Wahl der lokalen Ansatzräume. Insbesondere kann hier Wissen über den Character und die Regularität der Lösung durch zusätzliche Ansatzfunktionen eingepflegt werden. Damit ist die Möglichkeit gegeben, essentielle Eigenschaften der Lösung schon mit wenigen Funktionen gut aufzulösen.
Im Folgenden werden die wesentlichen Konzepte der in [14] entwickelten *Partition of Unity Methode* zusammengefasst vorgestellt. Notation und Darstellung orientieren sich an besagter Aufzeichnung. Für weitere Details verweisen wir hierauf, sowie auf ([15, 5, 17]).

2.1 Partition of Unity Ansatzraum

Ausgehend von einer Menge von Punkten $P := \left\{x_i \in \mathbb{R}^d \mid x_i \in \overline{\Omega},\ i = 1, \cdots, N\right\}$, die keine starren Abhängigkeiten untereinander besitzt, wird eine endliche Überdeckung des Gebiets Ω erzeugt, indem jedem Punkt x_i ein Patch

$$\omega_i := \bigotimes_{l=1}^{d} \left(x_i^l - h_i^l, x_i^l + h_i^l\right) \subset \mathbb{R}^d \tag{2.1}$$

zugeordnet wird. Im Allgemeinen sind auch Patches anderer Gestalt, wie beispielsweise Kreise oder Ellipsen, denkbar, diese bringen jedoch keine wesentlichen Vorteile bei schwierigerer Handhabung was die Berechnung angeht.
Der für die spätere Galerkin Diskretisierung notwendige Funktionenraum für Ansatz- und Testfunktionen wird bei der PU-Methode aus dem Produkt lokaler Ansatzräume $V_i^{pi} = \text{span}\langle\{\vartheta_i^n\}\rangle$ der Ordnung p_i mit passenden Partition of Unity Funktionen $\{\varphi_i\}$ konstruiert. Der globale Ansatzraum lässt sich damit schreiben als

$$V^{PU} := \sum_i \varphi_i V_i^{p_i} = \text{span}\langle\{\varphi_i \vartheta_i^n\}\rangle. \tag{2.2}$$

Die auf ω_i definierten lokalen Ansatzräume sind völlig unabhängig voneinander frei wählbar und bestimmen die lokale Approximationseigenschaft, d.h. die Qualität mit der die exakte Lösung lokal durch die Ansatzfunktionen $\{\vartheta_i^n\}$ approximiert werden kann. Beispielsweise werden glatte Funktionen gut durch Polynome approximiert.
Weiterhin müssen Elemente des globalen Ansatzraumes $u^{PU} \in V^{PU}$ eine gewisse globale Regularität erfüllen, d.h. die lokalen Ansatzräume müssen über ihren Definitionsbereich hinweg konform sein. Um dies zu erreichen betrachten wir die PU Funktionen $\{\varphi_i\}$, definiert durch die zugrundeliegende Überdenkung des Gebiets $\overline{\Omega} \subset \bigcup \omega_i$ mit $supp(\varphi_i) = \omega_i$, die die gewünschte globale Regularität erfüllen und fordern, dass die φ_i eine Partition der Eins bilden, d.h. $\sum_i \varphi_i \equiv 1$ auf Ω. Die letzte Forderung ist essenziell um die lokale Approximationseigenschaft nicht zu verletzen. Eine globale Approximationsfunktion u^{PU} ist also eine durch die φ Funktionen hergestellte Zusammenfassung lokaler Approximationen $u_i = \sum_n u_i^n \vartheta_i^n$,

$$u^{PU}(x) = \sum_{i=1}^{N} \varphi(x) u_i(x) \tag{2.3}$$

die globalen Regularitätsbedingungen genügt.

Für die Konstruktion der Partition of Unity Funktionen $\{\varphi_i\}$ für eine zugrundeliegende Überdeckung $\{\omega_i\}$ des Gebiets verwendet das Verfahren die Methode nach Shepard. Hierbei werden die φ_i definiert als

$$\varphi_i(x) = \frac{W_i(x)}{\sum_{j=1}^{N} W_j(x)}, \quad \text{bzw.} \quad \varphi_i(x) = \frac{W_i(x)}{\sum_{\omega_k \in C_i} W_k(x)}$$

wobei die $W_i(x)$ Gewichtsfunktionen mit $supp(W_i) = \overline{\omega_i}$ sind. Zum Beispiel können die W_i als B-Splines gewählt werden und im Fall von Achsenparalellen Patches als Produkt univariater Funktionen geschrieben werden $W_i(x) = \prod_{l=1}^{d} W_i^l(x^l)$. Durch eine Koordinatentransformation auf das Intervall $[0, 1]$ kann für alle Punkte eine Referenz-Gewichtsfunktion $\mathcal{W}$ ausgewertet werden. Da die W_i lokalen Support haben, genügt es über die Nachbarschaft $C_i := \{\omega_i \in C_\Omega \mid \omega_i \cap \omega_j \neq \emptyset\}$ des Patches zu summieren, d.h. die Komplexität einer Funktionsauswertung der φ_i ist $\mathcal{O}(1)$.
Die Nachbarschaft C_i definiert die Abhängigkeiten der Patches untereinander und damit gleichzeitig die Besetzungsstruktur der resultierenden dünnbesetzten Steifigkeitsmatrix, d.h. die Verteilung der Punktemenge P und die Konstruktion des Covers C_Ω haben einen großen Einfluss auf den benötigten Rechenaufwand.

Die PU-Funktionen φ_i erben Glattheitseigenschaften der Gewichtsfunktionen W_i. Es lassen sich also Partitionen beliebiger Regularität konstruieren. Gleichzeitig hat die Größe des Überlapps $\omega_i \cap \omega_j$ der Patches bzw. Supports der Gewichtsfunktionen direkten Einfluss auf den Betrag der Gradienten $\nabla\varphi_i$ und $\nabla\varphi_j$. Um zu garantieren, dass die Gradienten beschränkt bleiben, sollte der Durchmesser des Überlapps also ebenfalls nach unten beschränkt sein. Findet man weiterhin Konstanten, sodass $\|\varphi_i\|_{L^\infty(\mathbb{R}^d)} \leq C_\infty$ bzw. $\|\nabla\varphi_i\|_{L^\infty(\mathbb{R}^d)} \leq \frac{C_\nabla}{diam(\omega_i)}$ gilt, so kann man Fehlerabschätzungen (vgl. [14, S. 20, 21]) für die PU-Methode nachweisen. Diese Abschätzungen zeigen, dass die Approximationseigenschaft des PUM Ansatzraumes V^{PU} zum einen durch das Hinzufügen von Punkten x_i (kleinerer Patchdurchmesser, H-Verfeinerung) und zum anderen durch Verbesserung der Approximationseigenschaften der lokalen Ansatzräume (P-Verfeinerung) gesteigert werden kann.

Lokale Approximationsräume:

Unter diesem Gesichtspunkt wollen wir die Struktur der lokalen Ansatzräume genauer betrachten. Im Allgemeinen ist ein lokaler Ansatzraum aus zwei Komponenten zusammengesetzt: Einem glatten Ansatzraum $\mathcal{P}^{p_i}(\omega_i) := \text{span}\langle\psi_i^s\rangle$ und einem problemabhängigen Teil $\mathcal{E}_i(\omega_i) := \text{span}\,\langle\eta_i^t\rangle$ mit Anreicherungsfunktionen $\{\eta_i^t\}$, den sog. Enrichments:

$$V_i(\omega_i) = \text{span}\langle\vartheta_i^m\rangle = \mathcal{P}_i^{p_i}(\omega_i) + \mathcal{E}_i(\omega_i) = \text{span}\langle\psi_i^s,\ \eta_i^t\rangle \tag{2.4}$$

Der glatte Teil $\mathcal{P}^{p_i}$ des Approximationsraums besteht üblicherweise aus Polynomen vom Grad $\leq p_i$ – in dieser Arbeit verwenden wir analog zu [14, S. 18] Tensorprodukte univariater Legendre Polynome $\psi_i^s := \prod_{l=1}^d \mathcal{L}_i^{\hat{s}_l}$ mit $\sum_{i=1}^d \hat{s}_l \leq p_i$, wobei $\hat{s}_l$ der Polynomgrad des univariaten Legendre Polynoms $\mathcal{L}_i^{\hat{s}_l} : [x_i^l - h_i^l,\ x_i^l + h_i^l] \mapsto \mathbb{R}$ ist.
Da die Lösung einer partiellen DGl jedoch im Allgemeinen nicht beliebig glatt ist, können nicht alle Eigenschaften der Lösung gut durch glatte Ansatzfunktionen approximiert werden. Aufgrund der Unabhängigkeit der lokalen Ansatzräume, bietet die *Partiton of Unity Methode* die Möglichkeit zuvor bekannte Eigenschaften der Lösung wie Unstetigkeiten oder Singularitäten in Form von speziellen Anreicherungsfunktionen in die jeweiligen lokalen Approximationsräume einzupflegen. Dieser problemabhängige Teil der Ansatzräume verbessert deren Approximationsqualität signifikant und stellt einen wesentlichen Vorteil der PU-Methode im Vergleich zu der Finiten Elemente Methode dar. Da in dieser Arbeit insbesondere auch die Verwendung unterschiedlicher Anreicherungsfunktionen untersucht werden soll, werden wir diesen Aspekt in Kapitel 4 detaillierter diskutieren.

2.2 Behandlung elliptischer Gleichungen

2.2.1 Modellproblem

Die *Partition of Unity Methode* ist in erster Linie für die näherungsweise Lösung elliptischer Randwertprobleme entworfen. Randwertprobleme dieses Typs lassen sich in allgemeiner Form schreiben als

$$\begin{aligned} Lu &= f \text{ in } \Omega \subset \mathbb{R}^d \\ Bu &= g \text{ auf } \partial\Omega, \end{aligned}$$

wobei L ein symmetrischer elliptischer partieller Differentialoperator zweiter Ordnung ist und B dazu passende Randbedingungen definiert. Da in dieser Arbeit jedoch im Besonderen Probleme mit nichtstetigen Koeffizientenfunktionen - z.B. Simulation von Stoffen mit Materialeinschlüssen - untersucht werden, beschränken wir uns im Folgenden auf den einfacheren Spezialfall eines skalaren *Diffusionsproblems* (vgl. [19]):

$$\begin{aligned} -\nabla \cdot (\kappa \nabla u) &= f \text{ in } \Omega \subset \mathbb{R}^d \\ u &= g_D \text{ auf } \Gamma_D \subset \partial\Omega \\ (\kappa \nabla u) \cdot n &= g_N \text{ auf } \Gamma_N := \partial\Omega \setminus \Gamma_D, \end{aligned} \tag{2.5}$$

wobei der *Diffusionskoeffizient* κ eine im Allgemeinen nichtstetige, aber beschränkte Funktion ist. Wir beschränken uns in dieser Arbeit auf den Fall einer positiven, stückweise konstanten Koeffizientenfunktion, mit der beliebige Materialeinschlüsse modelliert werden können. Wir definieren offene Teilgebiete $\Omega_k \subset \Omega,\ k = 1, \cdots, m$ auf denen κ konstant ist, als *Materialeinschlüsse* und fordern, dass diese paarweise disjunkt sind. Weiterhin bezeichnen wir $\Omega_0 = \Omega \setminus \bigcup_{k=1}^{m} \Omega_k$ als das Gebiet des Matrix-Materials. Es gilt dann $\kappa|_{\Omega_k} = c_M^k = \text{konst.}$ für $k = 0, \cdots, m$. Den Schnitt $\Gamma_M^k := \overline{\Omega}_0 \cap \overline{\Omega}_k$ definieren wir als *Materialinterface* des k-ten Einschlusses und deren Vereinigung als $\Gamma_M := \bigcup_{k=1}^{m} \overline{\Omega}_0 \cap \overline{\Omega}_k$.
Das in 2.5 vorgestellte Modellproblem findet viele für die Praxis relevante Anwendungen wie zum Beispiel Simulation von Grundwasserströmungen, Halbleiter Modellierung oder Brennstoffzellensimulation [20]. Der Diffusionskoeffizient kann hier große Sprünge an Interfaces zwischen Materialien unterschiedlicher Eigenschaften aufweisen. Diese Koeffizientensprünge erhöhen die Schwierigkeit des Problems meist erheblich.

2.2.2 Diskretisierung

Um eine näherungsweise Lösung des Problems 2.5 zu erhalten, diskretisieren wir dieses mit der Galerkin-Methode. Die Lösung des resultierenden linearen Gleichungssystems

$$A\tilde{u} = \hat{f} \tag{2.6}$$

ist ein Koeffizientenvektor $\tilde{u}$, der die Koeffizienten der lokalen Approximationen u_i aus Gleichung 2.3 enthält.

Da wir uns auf den Fall einer stückweise konstanten Koeffizientenfunktion beschränken, können wir die erste Gleichung aus 2.5 schreiben als $-\kappa\Delta u = f$ in Ω. Der Einfachheit halber nehmen wir Dirichlet Nullrand-Bedingungen an. Mit der Variationsformulierung erhalten wir für eine geeignete Testfunktion unter Verwendung der 1. Greenschen Identität, dass

$$\int_\Omega -v \cdot (\kappa\Delta u) = \int_\Omega \kappa\nabla u\nabla v + \int_{\partial\Omega} v(\nabla u \cdot \vec{n}) = \int_\Omega f \cdot v$$

Hier bezeichnet $\vec{n}$ den äußeren Normalenvektor an das Gebiet Ω. Sei nun $a(\cdot,\cdot)$ die von $L = -\Delta$ induzierte stetige elliptische Bilinearform auf dem Sobolev-Raum $H_0^1(\Omega)$ und $l(\cdot)$ eine stetige Linearform auf $H_0^1(\Omega)$ für die rechte Seite. Anwendung der Galerkin-Methode mit Ansatz- und Testfunktionen aus dem Ansatzraum V^{PU} liefert das lineare Gleichungssystem 2.6 mit

$$A = \left(A_{(i,n),(j,m)}\right), \text{ mit } A_{(i,n),(j,m)} = a\left(\varphi_j\vartheta_j^m, \varphi_i\vartheta_i^n\right)$$
$$\hat{f} = \left(f_{(i,n)}\right), \text{ mit } f_{(i,n)} = l\left(\varphi_i\vartheta_i^n\right)$$

(vgl. [14, s. 23]), wobei

$$a\left(\varphi_j\vartheta_j^m, \varphi_i\vartheta_i^n\right) = \int_\Omega \nabla(\varphi_i\vartheta_i^n)\nabla(\varphi_j\vartheta_j^m) - \int_{\partial\Omega}(\varphi_i\vartheta_i^n)\left(\nabla(\varphi_j\vartheta_j^m)\cdot\vec{n}\right) \tag{2.7}$$
$$l\left(\varphi_i\vartheta_i^n\right) = \int_\Omega f\varphi_i\vartheta_i^n$$

Da die PU-Funktionen φ_i lokalen Support haben, müssen die Integrale in Gleichung 2.7 nur für $\omega_i \cap \omega_j \cap \Omega$ bzw. $\omega_i \cap \omega_j \cap \partial\Omega$ ausgewertet werden. A ist also eine dünnbesetzte Blockmatrix, deren Besetzungsstruktur durch die geometrischen Nachbarschaften der Patches definiert wird. Die Blockmatrizen sind dichtbesetzt und werden durch die Dimensionen der lokalen Approximationsräume bestimmt. Die Indices n und m sind Nummerierungen der Ansatzfunktionen in $V_i(\omega_i)$ bzw. $V_j(\omega_j)$.

Behandlung von Randwerten: Da die Ansatzfunktionen $\varphi_i\vartheta_i^n$ nichtinterpolatorisch sind, ist die Behandlung der Randwerte zunächst nicht klar. Während sich für Neumann Randbedingungen keine Probleme ergeben (nur Vorgaben für Normalenableitungen, nicht für die Werte der Lösung am Rand), ist die korrekte Einbindung von Dirichlet Randwerten jedoch durchaus eine Herausforderung: Die lokalen Ansatzräume problemabhängig zu wählen, sodass $\vartheta_i^n = 0$ auf Γ_D ist, ist nicht praktikabel. In der *Partition of Unity Methode* wurde hierfür der Ansatz von Nitsche [7] verwendet. Dabei werden Randterme mit einem Regularisierungsparameter β gewichtet, zu dessen Berechnung ein Eigenwertproblem gelöst werden muss. Die schwache Form des Modellproblems 2.5 mit implementierten beliebigen Randbedingungen lautet dann:

$$\int_\Omega \nabla u\nabla v + \int_{\Gamma_D} u\left(\beta v - \frac{\partial v}{\partial n}\right) - \frac{\partial u}{\partial n}v = \int_\Omega fv + \int_{\Gamma_D} g_D\left(\beta v - \frac{\partial v}{\partial n}\right) + \int_{\Gamma_N} g_N v \tag{2.8}$$

Wir werden hier nicht näher darauf eingehen, eine detaillierte Beschreibung ist in [7], [14, S. 31-37] zu finden.

Numerische Integration: Die numerische Integration von 2.7 zur Assemblierung der Steifigkeitsmatrix und des Rechte-Seite-Vektors stellt typischerweise den größten Anteil am Rechenaufwand dar. Da die lokalen Ansatzfunktionen $\varphi_i\vartheta_i^n$ keine ähnliche Struktur haben, sondern

vom zugrundeligenden Cover und den Gewichtsfunktionen W_k abhängen, ist es nicht möglich die Integration durch eine geeignete Transformation auf eine Referenzkonfiguration auszulagern. Weiterhin wird ein spezielles Integrationsverfahren benötigt, da die Ansatzfunktionen stückweise rationale Funktionen sind und die Integranden im Bereich der Überlapps $\omega_i \cap \omega_j$ Unstetigkeiten aufweisen können. Aus diesem Grund muss das Integrationsgebiet an diesen Überlapps so in Integrationszellen unterteilt werden, dass die Gewichtsfunktionen W_k in jeder dieser Zellen polynomiell sind. Die Integranden sind dann rationale Funktionen, die keine Singularitäten mehr aufweisen (diese sind jetzt genau an den Stoßstellen). Das verwendete Quadraturschema ist ein Dünngitter-Verfahren, das auf den univariaten *Gauß-Patterson* Quadraturformeln basiert. Dieses Verfahren konvergiert für glatte Integranden, wie hier lokal auf den Integrationszellen sichergestellt, exponentiell. Für eine detailliertere Beschreibung verweisen wir auf [14, S. 25-31].

KAPITEL 3

Multilevel-Löser

Der nächste Schritt besteht in der Lösung des aus der Galerkin Diskretisierung gewonnenen, großen, dünnbesetzten linearen Gleichungssystems 2.6. Die *effiziente* Lösung dieses Systems ist für die Komplexität des Verfahrens von großer Wichtigkeit, da dieser Teilprozess einen dominierenden Faktor für fast alle Lösungsverfahren partieller Differentialgleichungen darstellt. Vor allem eine schlechte Wahl des Lösers kann die Verwendung des Verfahrens in der Praxis wegen nicht akzeptabler Laufzeit bzw. zu hohem Speicherverbrauch einschränken oder unmöglich machen.
Man unterscheidet prinzipiell zwei Klassen von Lösern: Direkte Löser, wie Gauß-Elimination oder LU-Zerlegung, und sog. iterative Löser. Da direkte Löser eine Komplexität von $\mathcal{O}(N^3)$ und quadratischen Speicherverbrauch $\mathcal{O}(N^2)$ in der Anzahl der Unbekannten N haben, sind sie nur für sehr kleine Probleminstanzen $N < 200 \times 200$ praktikabel und in unserem Fall, für die Lösung eines sehr großen, dünnbesetzten LGS, unbrauchbar. Aus diesem Grund greift man hier auf Verfahren zurück, die die Lösung des Gleichungssystems approximativ in einem iterativen Prozess bestimmen. Klassische Vertreter sind zum Beispiel die Jacobi- und Gauß-Seidel Methode sowie Erweiterungen wie das SOR-Verfahren. Während diese Methoden effizient bezüglich des Speichers arbeiten, erzielen sie nicht immer gute Konvergenzraten.
Einen weiteren Ansatz der iterativen Lösung stellen die sog. *iterativen Multilevel-Löser* bzw. *Mehrgitter Verfahren* dar, die optimales Verhalten bezüglich Konvergenz und Speicherverbrauch aufweisen.
Im folgenden Kapitel wollen wir zunächst die prinzipielle Idee und Arbeitsweise von Multilevel-Lösern im Allgemeinen skizzieren, um diesen Ansatz dann auf die *Partition of Unity Methode* zu übertragen. Dabei gehen wir auch auf den Begriff der Robustheit von (Multilevel-) Lösern ein, was insbesondere Kernpunkt der Untersuchungen in dieser Arbeit ist. Am Ende dieses Kapitels werden wir noch einen Einblick in die konkrete Implementierung des PUM Multilevel-Lösers geben.

Die Ausführungen orientieren sich an [14, Kapitel 4] und [6]. Für weitere Details verweisen wir hierauf, sowie auf [16, Kapitel 5] und [5, 3, 10].

3.1 Allgemeine Multilevel-Löser

Multilevel- bzw. Mehrgitterverfahren wurden speziell zur Lösung großer, dünnbesetzter Gleichungssysteme entwickelt, die üblicherweise Ergebnis der Diskretisierung von partiellen Differentialgleichungen sind. Die Komplexität für das Lösen eines solchen Systems sollte von gleicher Ordnung sein, wie die für das Aufstellen desselben, sprich das Assemblieren der Steifigkeitsmatrix und Aufstellen des Rechte-Seite-Vektors. Jedoch können weder direkte Löser, noch klassiche Iterative Lösungsverfahren eine Komplexität aufweisen, die linear mit der Anzahl der Unbekannten skaliert. Mehrgitter- bzw. Multilevel-Verfahren erreichen diese optimale Komplexität, nutzen jedoch in großem Umfang bekannte Eigenschaften der partiellen DGl und der verwendeten Diskretisierung aus. Es handelt sich hierbei also nicht um ein rein algebraisches Verfahren zur Lösung beliebiger Gleichungssysteme.

3.1.1 Multilevel-Verfahren

Bei der Untersuchung und Analyse klassicher Iterativer Löser beobachtet man, dass zu Beginn der Iteration eine hervorragende Fehlerreduktion stattfindet, die Konvergenz des Verfahrens dann aber schlagartig einbricht und weitere Iterationen die Approximation der Lösung kaum verbessern. Eine Konvergenzanalyse zeigt, dass gerade oszillierende Komponenten des Fehlers sehr schnell und effizient reduziert werden, glatte Fehleranteile jedoch nahezu unverändert bleiben. Sind die oszillierenden Fehleranteile nach den ersten Iterationen hinreichend klein, ist der verbleibende Fehler im Wesentlichen glatt und es findet keine Reduktion mehr statt. Konzeptionelle Idee des Mehrgitterverfahrens ist es, den nun glatten Fehler auf ein gröberes Level zu transferieren, bezüglich dessen gröberer Auflösung der Fehler wieder oszillierend ist, um dann über die Residuumsgleichung $A\tilde{e} = \hat{r}$ eine Approximation an den Fehler zu berechnen. Diese Näherung wird wieder auf das feinere Level transferiert und dient als Korrektur der Feingitterlösung. Für die Lösung der Residuumsgleichung auf dem gröberen Level kann man diese Idee nun rekursiv anwenden und bis auf ein definiertes gröbstes Gitter Ω_{coarse} absteigen. Da dieses System i. d. R. aus signifikant weniger Unbekannten besteht, als das auf dem feinsten Level, kann es mit weit weniger Aufwand bis auf Genauigkeit des Diskretisierungsfehlers gelöst werden – z. B. durch direkte Löser.
Die Idee von Grobgitterkorrektur und Fehlerglättung lässt sich unter geeigneten Voraussetzungen auf den Kontext endlichdimensionaler Vektorräume abstrahieren, d.h. wir sind im Allgemeinen nicht auf eine Hierarchie klassicher Gitter angewiesen. Auf dabei auftretende Probleme und Herausforderungen werden wir später näher eingehen. Für den Moment wollen

wir annehmen, dass wir zu einer Folge endlichdimensionaler Vektorräume $\mathcal{V}_0, \cdots, \mathcal{V}_J$, wobei $\mathcal{V}_J$ der Raum der feinsten Diskretisierung ist, eine Reihe von Transferoperatoren

$$I_{k-1}^k \colon \mathcal{V}_{k-1} \to \mathcal{V}_k \ , \quad k = 1, \cdots, J \tag{3.1}$$

$$I_k^{k-1} \colon \mathcal{V}_k \to \mathcal{V}_{k-1} \ , \quad k = 1, \cdots, J \tag{3.2}$$

sowie eine Reihe diskreter Repräsentationen A_k einer symmetrisch positiv definiten Bilinearform $a(\cdot,\cdot)$ (vgl. 2.7) auf $\mathcal{V}_k$ haben. Den Transfer von einem gröberen Level auf ein feineres nennt man *Prolongation* (3.1), die umgekehrte Operation heißt *Restriktion* (3.2). Außerdem seien Iterative Löser – die sogenannten Glätter – S_k^{post} und S_k^{pre} auf jedem Level gegeben, die die oszillierenden Fehleranteile reduzieren. Ein allgemeiner Multilevel Algorithmus ist dann wie in Alg. 3.1 gegeben (vgl. [14, S. 53]): Die Parameter ν_1 bzw. ν_2 geben die Anzahl der

Algorithmus 3.1 Multilevel Algolrithmus **procedure** $M_\gamma^{\nu_1,\nu_2}(k, x_k, b_k)$

(1) *Vorglättung:* **for** $l = 1, \cdots, \nu_1$: setze $x_k = S_k^{pre}(x_k, b_k)$

(2) *Restriktion:* restringiere das Residuum auf das gröbere Level
$d_{k-1} := I_k^{k-1}(b_k - A_k x_k)$

(3) *Grobgitterkorrektur:* falls $k = 0$ löse $x_k = A^{-1} b_k$ exakt. Sonst wende γ mal den Multilevel Algorithmus an, mit Startnäherung 0
$e_{k-1} := 0$
for $i = 1, \cdots, \gamma$: $e_{k-1} = M_\gamma^{\nu_1,\nu_2}(k-1, e_{k-1}, d_{k-1})$

(4) *Prolongation:* addiere prolongierte Grobgitterlösung
$x_k = C_k(x_k, e_{k-1}) := x_k + I_{k-1}^k e_{k-1}$

(5) *Nachglättung:* **for** $l = 1, \cdots, \nu_2$: setze $x_k = S_k^{post}(x_k, b_k)$

Vor- bzw. Nachglättungsschritte an. Der Parameter γ bestimmt die Rekursionsstruktur des Algorithmus. Für $\gamma = 1$ ergibt sich der bekannte *V-Cycle*.

Überlegungen zur Konvergenz: Betrachten wir für $J = 1$, $\nu_2 = 0$, $\gamma = 1$ die Iterationsmatrix des Zweilevel-Korrekturschemas $M_1^{\nu_1,0}$ und die Fehlerentwicklung $e^{k+1} = M_1^{\nu_1,0} e^k = (M_1^{\nu_1,0})^k e^0$, so folgt, dass die Norm von $M_1^{\nu_1,0}$ das Konvergenzverhalten bestimmt. Mit der Abschätzung

$$\|\left(\mathbb{I} - I_0^1 A_0^{-1} I_1^0 A_1\right) S^{\nu_1}\| = \|\left(\mathbb{I} - I_0^1 A_0^{-1} I_1^0 A_1\right) A_1 A_1^{-1} S^{\nu_1}\| \leq \|\left(A_1^{-1} - I_0^1 A_0^{-1} I_1^0\right)\| \|A_1 S^{\nu_1}\|.$$

können wir zwei wichtige Bedingungen für die Konvergenz des Verfahrens ableiten: Zum einen muss eine gewisse Qualität des Interlevel-Transfers gewährleistet sein $\|(A_l)^{-1} - I_{l-1}^l (A_{l-1})^{-1} I_l^{l-1}\| \leq C \|A_l\|^{-1}$ (*Approximtationseigenschaft*) zum anderen wird durch die *Glättungseigenschaft* $\|A_l S^{\nu_1}\| \leq \tilde{\eta}(\nu) \|A_l\|$, $\tilde{\eta}(\nu) \xrightarrow{\nu \to \infty} 0$ eine Konvergenzanforderung an den verwendeten Glätter gestellt. Ein Konvergenznachweis über obige Glättungs- und Approximationseigenschaft verwendet die sog. Galerkin Identität $A_{l-1} = I_l^{l-1} A_l I_{l-1}^l$ und setzt damit voraus, dass die $\mathcal{V}_k$ eine Reihe geschachtelter Unterräume $\mathcal{V}_0 \subset \mathcal{V}_1 \subset \cdots \subset \mathcal{V}_J$ sind.

Obwohl diese Eigenschaft für die PU-Ansatzräume nicht efüllt ist, können wir daraus doch eine Anschauung und ein besseres Verständnis für die Wichtigkeit der beiden wesentlichen Komponenten eines Multilevel-Lösers – Interlevel-Transfer und Glätter – bekommen. Nur wenn beide Teile optimal zusammenspielen und sowohl die hochfrequenten- (Glätter) als auch die niederfrequenten Fehleranteile (Interlevel-Transfer) effektiv reduziert werden, erhalten wir einen optimalen Löser. Eine auf Teilraumzerlegung und der Theorie von Schwarz basierende Konvergenztheorie wurde von Bramble, Pasicak und Xu in [1] entwickelt und konnte auf den für uns wichtigen Fall von nicht geschachtelten Räumen verallgemeinert werden [2].

Komplexität: Wie eingangs bereits erwähnt, erhalten wir für das Multilevel-Verfahren eine optimale, d.h. lineare Komplexität bezüglich Laufzeit und Speicherverbrauch. Um dies einzusehen wollen wir über die Nich-Null-Einträge der beteiligten dünnbesetzten Matrizen argumentieren (vgl. [14, S. 55]). Seien also $\mathcal{C}(A_k), \mathcal{C}(I_k^{k-1}), \mathcal{C}(I_{k-1}^k)$ die mittlere Anzahl der Nicht-Null-Einträge der Matrizen $A_k,\ I_k^{k-1},\ I_{k-1}^k$ pro Zeile. Die Summe $\sum_{k=0}^{J}\left(\mathcal{C}(A_k)+\mathcal{C}(I_k^{k-1})+\mathcal{C}(I_{k-1}^k)\right)$ entspricht also dem Speicherverbrauch. Da die Räume $\mathcal{V}_k$ jedoch aus einer sukzessiven Verfeinerung aus $\mathcal{V}_0$ entstehen, existieren beim Übergang von Level l nach $l-1$ nur noch ein Bruchteil der vorherigen Unbekannten (typischerweise etwa die Hälfte). Obige Summe entspricht also einer geometrischen Reihe und der Speicherbedarf ist linear in der Anzahl der Unbekannten der feinsten Diskretisierung.
Für die Laufzeit zählen wir die Anzahl der Operationen pro Unbekannter für die Defektberechung (Alg. 3.1, (2)), die Grobgitterkorrektur (Alg. 3.1, (4)) und den Glättungsschritt (Alg. 3.1, (1) bzw. (5)) als $\mathcal{C}_{D,k}$, $\mathcal{C}_{C,k}$, $\mathcal{C}_{S,k}$ auf jedem Level k. Diese sind (für festes ν_1, ν_2) konstant in der Zahl der Nicht-Null-Einträge pro Zeile $\mathcal{C}(A_k)$, $\mathcal{C}(I_k^{k-1})$ und $\mathcal{C}(I_{k-1}^k)$. Bezeichnen wir mit N_k^{dof} die Anzahl der Unbekannten auf Level k, können wir die Zahl der Operationen pro Unbekannter $\mathcal{C}^k_{M_\gamma^{\nu_1,\nu_2}}$ für eine Iteration auf Level k abschätzen durch

$$N_k^{dof}\cdot\mathcal{C}^k_{M_\gamma^{\nu_1,\nu_2}} \le N_k^{dof}\left(\mathcal{C}_{D,k}+\mathcal{C}_{C,k}+\mathcal{C}_{S,k}\right)+\gamma N_{k-1}^{dof}\cdot\mathcal{C}^{k-1}_{M_\gamma^{\nu_1,\nu_2}}.$$

Wir erhalten vom Level unabhängige Konstanten, wenn wir die Zahl der Nicht-Null-Elemente pro Zeile auf jedem Level durch die des feinsten Levels abschätzen. Summation über alle Level ergibt:

$$\mathcal{C}^k_{M_\gamma^{\nu_1,\nu_2}} \le \left(\mathcal{C}_D+\mathcal{C}_C+\mathcal{C}_S\right)\left(1+\sum_{k=1}^{J-1}\gamma^k\frac{N_{J-k}^{dof}}{N_J^{dof}}\right)+\gamma^J\frac{N_0^{dof}}{N_J^{dof}}\mathcal{C}_0.$$

Die Komplexität einer Iteration ist also linear in N_J^{dof} falls obige Reihe für $J\to\infty$ konvergiert.

3.1.2 Multilevel-Verfahren als Vorkonditionierer

Das Verfahren der conjugierten Gradienten (CG):
Betrachtet man das Minimierungsproblem $\mathbf{min}\ f(\mathbf{x}) = \mathbf{x}^T A\mathbf{x} - \mathbf{x}^T b$, $\mathbf{x}\in\mathbb{R}^d$, stellt man fest, dass ein Minimierer $\mathbf{x}_*$ der Funktion f im Fall einer symmetrisch positiv definiten Matrix A gleichzeitig auch eindeutige Lösung des linearen Systems $A\mathbf{x}=b$ ist, denn es gilt: $\nabla f(\mathbf{x}) = A\mathbf{x}-b|_{\mathbf{x}=\mathbf{x}_*} = 0$. Für s.p.d. Matrizen kann also equivalent zum Lösen eines LGS

auch die Funktion $f(\mathbf{x})$ minimiert werden. Das Verfahren der conjugierten Gradienten nutzt diese Tatsache aus und sucht, beginnend bei $\mathbf{x}_0 = 0$ iterativ einen Minimierer für f. Die beste Suchrichtung ist die des steilsten Abstiegs, also $-\nabla f = b - A\mathbf{x}$. Im Punkt $\mathbf{x}_0 = 0$ ist das gerade b und somit die initiale Suchrichtung p_1. Alle weiteren Suchrichtungen p_i sind so gewählt, dass sie bezüglich des inneren Produkts $\langle A\mathbf{x}, \mathbf{x}\rangle =: \langle \mathbf{x}, \mathbf{x}\rangle_A$ orthogonal auf dem Raum stehen, der von den bisherigen Suchrichtungen aufgespannt wird, d.h. sie sind conjugiert zueinander.
Bei exakter Arithmetik stellt das CG-Verfahren sogar einen direkten Löser dar, in der Praxis ist es aber ein hervorragender iterativer Löser, da üblicherweise schon für sehr wenige Iterationen gute Approximationen erzielt werden. Die Konvergenzrate des CG-Verfahrens hängt direkt von der Kondition $\kappa(A)$ der Matrix A ab:

$$\|\mathbf{x}_k - \mathbf{x}_*\| \leq 2\left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\right)^k \|\mathbf{x}_0 - \mathbf{x}_*\|$$

Prinzip der Vorkonditionierung: Diese Erkenntnis ist Motivation für die Idee der Vorkonditionierung. Ist $\kappa(A)$ groß, so hofft man, dass für einen geeigneten symmetrisch positiv definiten Vorkonditionierer M das modifizierte System $L^{-1}AL^{-T}L^T\mathbf{x} = L^{-1}b =: \tilde{A}\tilde{\mathbf{x}} = \tilde{b}$ mit $M = LL^T$ bessere Lösungseigenschaften besitzt, d.h. dass $\kappa\left(M^{-1}A\right)$ kleiner ist als $\kappa(A)$. Da für s.p.d. Matrizen $\kappa(\tilde{A}) = \frac{\lambda_{max}(\tilde{A})}{\lambda_{min}(\tilde{A})}$ gilt, ist ein Vorkonditionierer dann gut, wenn er die Eigenwerte der Systemmatrix konzentriert.
Einen sehr guten Löser erhält man, wenn das CG-Verfahren mit einem optimalen Multilevel-Verfahren als Vorkonditionierer angewendet wird. Dieses, im Folgenden MPCG (Multilevel Preconditioned Conjugate Gradient) genannte Verfahren kommt auch in dieser Arbeit zum Lösen des Systems 2.6 zum Einsatz.

3.2 Multilevel-Löser für die Partition of Unity Methode

Um einen Multilevel-Löser für die *Partition of Unity Methode* zu konstruieren, müssen die im vorherigen Abschnitt vorgestellten Konzepte in den Kontext der PUM übertragen werden. Wichtigste Bestandteile sind eine Sequenz von adäquaten Räumen auf denen das Multilevel arbeitet, die Konstruktion von möglichst guten Interlevel-Transferoperatoren sowie effizienten Glättern. Der Löser wurde in [14, S. 56-73], [6] und [5] entwickelt; wir werden hier nur die wesentlichen Aspekte vorstellen.

3.2.1 Konstruktion einer Sequenz von Funktionsräumen

Die für das Multilevel-Verfahren benötigten Funktionsräume sind bei der *Partition of Unity Methode* nicht kanonisch gegeben. Ein räumlicher Verfeinerungsansatz führt auf eine Sequenz von Überdeckungen $C_{\Omega,k} = \{\omega_{i,k}\}$ mit zunehmender Anzahl an Patches. Zur Erzeugung der Überdeckungen $C_{\Omega,k}$ und der zugrundeliegenden Punktemengen P_k wurde in [14, Kapitel 5] ein sehr effizienter, auf **k**-d-Bäumen basierender, hierarchischer Algorithmus entwickelt, der gleichzeitig eine Datenstruktur zur schnellen Nachbarsuche darstellt und die numerische

Integration bei der Assemblierung der Steifigkeitsmatrix beschleunigt. Das Gebiet Ω wird auf Basis einer Punktemenge P_J solange mittels des **k**-d-Baums verfeinert, bis in jeder Baumzelle höchstens ein Punkt ist. Eine Überdeckung $C_{\Omega,k}$ wird im Baum durch die Gesamtheit der (aktiven) Blätter repräsentiert. Ausgehend von einer Repräsentation der feinsten Überdeckung $C_{\Omega,J} = \{\omega_{i,J} \mid i = 1, \cdots, N_J\}$ im Baum werden gröbere Überdeckungen $C_{\Omega,k},\ k = J-1, \cdots, 0$ mit $C_{\Omega,k} = \{\omega_{i,k} \mid i = 1, \cdots, N_k\}$ durch das Entfernen (Deaktivieren) aller Blätter, deren Geschwister ebenfalls ausschließlich Blätter sind, erzeugt.
Die Patches für die Überdeckungen gehen aus den entsprechenden Baumzellen $\mathcal{C}^B_{i,k}$ durch eine Skalierung mit dem Stretchparameter α hervor: $\omega_{i,k} = \alpha\mathcal{C}^B_{i,k}$. Der Parameter α bestimmt den Durchmesser des Überlapps $\omega_{i,k} \cap \omega_{j,k}$ und damit den Betrag der Gradienten $\nabla\varphi_{i,k}$ und $\nabla\varphi_{j,k}$ - er sollte aus $\alpha \in (1, 2)$ gewählt werden um zumindest Stetigkeit der Partition der Eins zu erhalten und die lineare Unabhängigkeit der Basisfunktionen $\vartheta^n_{i,k}$ zu gewährleisten. Auf Grundlage der erzeugten Überdeckungen $C_{\Omega,k}$ erhalten wir über die in Kapitel 2 vorgestellte Konstruktion eine Sequenz von PU-Funktionsräumen V^{PU}_k für das Multilevel-Verfahren. Für eine sehr detaillierte Beschreibung der hierarchischen Konstuktion von Überdeckungen und Funktionsräumen verweisen wir auf oben genannte Arbeiten.

Obwohl für die Baumzellen gilt, dass die Kinder eines inneren Knotens eine Partition dieser Baumzelle bilden, die Ränder also aufeinander liegen, gilt diese Eigenschaft für die induzierten Überdeckungen nach der Skalierung mit α nicht, d.h. $\omega_{j,k-1} \neq \bigcup_{i \in I^H_j} \omega_{i,k}$ mit $I^H_j = \{i \mid \omega_{j,k-1} \cap \omega_{i,k} \neq \emptyset\}$. Damit liegen auch die Träger der PU-Funktionen $\varphi_{i,k},\ \varphi_{j,k-1}$ nicht ausgerichtet übereinander und die resultierenden PU-Funktionsräume sind nicht ineinader geschachtelt

$$V^{PU}_{k-1} = \sum_{i=1}^{N_{k-1}} \varphi_{i,k-1} V_{i,k-1} \not\subset \sum_{i=1}^{N_k} \varphi_{i,k} V_{i,k} = V^{PU}_k,\ k = 0, \cdots, J \tag{3.3}$$

d.h. die Basisfunktionen $\varphi_{i,k-1}\vartheta^n_{i,k-1}$ können auf einem feineren Level nicht exakt repräsentiert werden. Außerdem sind die Basisfunktionen nicht interpolatorisch, die Interlevel-Transferoperatoren können also nicht, wie oft der Fall, durch einfache Interpolation und natürliche Injektion realisiert werden.

3.2.2 Konstruktion von Interlevel-Transferoperatoren

Die Realisierung der Prolongtions- und Restriktionsoperatoren $I^k_{k-1}\colon V^{PU}_{k-1} \to V^{PU}_k$ bzw. $I^{k-1}_k\colon V^{PU}_k \to V^{PU}_{k-1}$ durch geeignete L_2-Projektionen $\Pi^k_{k-1},\ \Pi^{k-1}_k$ stellt sich aufgrund der nicht geschachtelten Funktionsräume in der PUM als gute Wahl dar (vgl. [14, S. 59-64]). Eine solche Projektion $\Pi^{\widetilde{W}}_W\colon W \to \widetilde{W}$, mit $W, \widetilde{W} \subset L^2(\Omega)$ vermöge

$$\Pi^{\widetilde{W}}_W = \left(M^{\widetilde{W}}_{\widetilde{W}}\right)^{-1} \left(M^{\widetilde{W}}_W\right)$$

bildet mit Hilfe der Matrizen $\left(M^{\widetilde{W}}_{\widetilde{W}}\right)_{i,j} := \langle \phi^{\widetilde{W}}_j, \phi^{\widetilde{W}}_i \rangle_{L^2(\Omega)}$ und $\left(M^{\widetilde{W}}_W\right)_{i,j} := \langle \phi^W_j, \phi^{\widetilde{W}}_i \rangle_{L^2(\Omega)}$ Koeffizientenvektoren $\tilde{u}_W$ auf Koeffizientenvektoren $\tilde{u}_{\widetilde{W}}$ ab. Der transponierte Operator transportiert umgekehrt Information der Rechte-Seiten-Vektoren $\hat{f}$ von $\widetilde{W}$ nach W und kann

somit als Restriktionsoperator verwendet werden. Wir beschränken uns im Folgenden auf die Darstellung des Prolongationsoperators und stellen kurz die in [14, S. 59-64] entwickelten unterschiedlichen Implementierungen des L^2-Projektion-Ansatzes vor.

(1) **Global-nach-global Projektion:**
Wendet man obigen Ansatz direkt auf den Kontext der in 3.2.1 vorgestellten Funktionsräume an, erhält man mit $W = V_{k-1}^{PU}$ und der Basis $\{\phi_j^W\} = \{\varphi_{i,k-1}\vartheta_{i,k-1}^n\}$ und $\widetilde{W} = V_k^{PU}$ mit Basis $\{\phi_j^{\widetilde{W}}\} = \{\varphi_{i,k}\vartheta_{i,k}^n\}$ die globale L^2-Projektion

$$\begin{aligned} \Pi_{k-1}^k &= (M_k^k)^{-1}(M_{k-1}^k), \text{ mit} \\ (M_k^k)_{(i,n),(j,m)} &= \langle \varphi_{j,k}\vartheta_{j,k}^m, \varphi_{i,k}\vartheta_{i,k}^n \rangle_{L^2(\Omega)} \\ (M_{k-1}^k)_{(i,n),(j,m)} &= \langle \varphi_{j,k-1}\vartheta_{j,k-1}^m, \varphi_{i,k}\vartheta_{i,k}^n \rangle_{L^2(\Omega)} \end{aligned} \tag{3.4}$$

Die Verwendung dieser Prolongtion ist jedoch sehr teuer, da die vollständige Massenmatrix M_k^k invertiert werden muss und die Besetzungsstruktur der der Steifigkeitsmatrix entspricht, was den Assemblierungsprozess und Speicherverbrauch negativ beeinflusst. Weiterhin wird die Besetzungsstruktur der Interlevel- oder hierarchischen Massenmatrix M_{k-1}^k durch die *geometrischen Interlevel-Nachbarschaften* $C_{j,k-1,k}^G := \{\omega_{i,k} \in C_{\Omega,k} \mid \omega_{j,k-1} \cap \omega_{i,k} \neq \emptyset\}$, die i. A. eher groß sind, bestimmt. Die folgenden alternativen Operatoren versuchen ähnlich gutes Verhalten bei teilweise stark vereinfachten Matrizen M_k^k und M_{k-1}^k zu erreichen.

(2) **Global-nach-local Projektion:**
Bei dieser Projektion werden die globalen Groblevel-Funktionen $u_{k-1}^{PU} = \sum \varphi_{j,k-1} \sum u_{j,k-1}^m \vartheta_{j,k-1}^m$ nur durch die $\vartheta_{i,k}^n$ der lokalen Approximationsräume $V_{i,k}$ lokal auf den Feinlevel-Patches $\omega_{i,k}$ approximiert:

$$\begin{aligned} \widehat{\Pi}_{k-1}^k &= (\widetilde{M}_k^k)^{-1}(\widehat{M}_{k-1}^k), \text{ mit} \\ (\widetilde{M}_k^k)_{(i,n),(i,m)} &= \langle \vartheta_{i,k}^m, \vartheta_{i,k}^n \rangle_{L^2(\omega_{i,k}\cap\Omega)} \\ (\widehat{M}_{k-1}^k)_{(i,n),(j,m)} &= \langle \varphi_{j,k-1}\vartheta_{j,k-1}^m, \vartheta_{i,k}^n \rangle_{L^2(\omega_{i,k}\cap\Omega)} \end{aligned} \tag{3.5}$$

Die lokale Massenmatrix $\widetilde{M}_k^k$ hat jetzt Blockdiagonalgestalt und ist folglich leicht zu invertieren und mit wenig Aufwand zu assemblieren. Da jedoch nach wie vor die globalen Groblevel-Funktionen u_{k-1}^{PU} approximiert werden, ist die Besetzungsstruktur der Interlevel-Massenmatrix $\widehat{M}_{k-1}^k$ weiterhin durch $C_{j,k-1,k}^G$ definiert. Der Projektionsoperator ist jedoch nicht symmetrisch und seine Transponierte – die Restriktion – approximiert lokales Verhalten auf den feinen Patches durch globale Funktionen $\varphi_{j,k-1}\vartheta_{j,k-1}^m$, was im Fall verschiedener lokaler Ansatzfunktionen zu schlechten Resultaten für die Restriktion führen kann.

(3) **Local-nach-local Projektion:**
Für diese Variante wird angenommen, dass die Überdeckungen $C_{\Omega,k}$ von der zunehmenden Verfeinerung eines uniformen Gitters induziert werden (d.h. die Patches der Überdeckung bzgl. eines Levels k haben alle die gleiche Größe, vgl. 2.1). Die Überdeckungen $C_{\Omega,k}$, $C_{\Omega,k-1}$

haben dann die hierarchische Eigenschaft, dass es für jeden Patch $\omega_{i,k}$ der feinen Überdeckung genau einen groben Patch $\omega_{j,k-1} \in C_{\Omega,k-1}$ gibt, sodass $\omega_{i,k} \subset \omega_{j,k-1}$ gilt. Für die Interlevel-Massenmatrix genügt es dann, nur die *hierarchischen Interlevel-Nachbarschaften*

$$\begin{aligned} C^H_{j,k-1,k} &:= \{\omega_{i,k} \in C_{\Omega,k} \mid \omega_{i,k} \subseteq \omega_{j,k-1}\} \\ C^H_{i,k,k-1} &:= \{\omega_{j,k-1} \in C_{\Omega,k-1} \mid \omega_{i,k} \subseteq \omega_{j,k-1}\},\ \mathrm{card}(C^H_{i,k,k-1}) = 1 \end{aligned} \tag{3.6}$$

zu betrachten. Der Prolongationsoperator ist dann gegeben durch:

$$\begin{aligned} \widetilde{\Pi}^k_{k-1} &= (\widetilde{M}^k_k)^{-1}(\widetilde{M}^k_{k-1}),\ \text{mit} \\ (\widetilde{M}^k_k)_{(i,n),(i,m)} &= \langle \vartheta^m_{i,k}, \vartheta^n_{i,k} \rangle_{L^2(\omega_{i,k}\cap\Omega)} \\ (\widetilde{M}^k_{k-1})_{(i,n),(j,m)} &= \langle \vartheta^m_{j,k-1}, \vartheta^n_{i,k} \rangle_{L^2(\omega_{i,k}\cap\Omega)} \end{aligned} \tag{3.7}$$

Die Matrix $\widetilde{M}^k_{k-1}$ hat also nur noch einen Blockeintrag pro Feinlevel-Patch $\omega_{i,k}$. Außerdem entfällt die schwierige und aufwendige Integration der PU-Funktionen $\varphi_{i,k}$, $\varphi_{i,k-1}$ vollständig. Numerische Tests (vgl. [14, S. 73-94]) haben gezeigt, dass die lokal-nach-lokal Projektion nur geringfügig schlechter als die globale Projektion ist, jedoch signifikant bessere Laufzeiten und geringeren Speicherverbrauch aufweist.

(4) **Gewichtete global-nach-global Projektion:**
Diese Alternative des Projektionsoperators ist in jüngster Zeit entstanden und wurde in dieser Arbeit erstmals realisiert und implementiert. Kernidee ist, die Restriktionseigenschaften durch zusätzliche Gewichtung der lokalen Masse mit den PU-Funktionen $\varphi_{i,k}$ zu verbessern, bei gleichzeitig stark vereinfachter Massenmatrix. Der natürliche Zusammenhang dieser sog. *lumped mass matrix* mit Vektoren der rechten Seite $\hat{f}$ im Kontext der PUM wurde zum ersten Mal in [13] erarbeitet.
Die Einträge $f_{(i,n)}$ von $\hat{f}$ sind innere Produkte (bzgl. $L^2(\Omega)$) der Daten f mit den Basisfunktionen $\varphi_i\vartheta^n_i$ (vgl. Gleichung 2.7). Im Rahmen der PUM können diese aber auch als mit φ_i *gewichtete* innere Produkte der Daten f mit den lokalen Ansatzfunktionen φ^n_i bzgl. $L^2(\omega_i \cap \Omega)$ aufgefasst werden (vgl. [13, S. 6]):

$$f_{(i,n)} = \langle f, \varphi_i\vartheta^n_i \rangle_{L^2(\Omega)} = \int_\Omega f\varphi_i\vartheta^n_i dx = \int_{\omega_i\cap\Omega} f\varphi_i\vartheta^n_i dx = \langle f|\varphi_i|\vartheta^n_i \rangle_{L^2(\omega_i\cap\Omega)}$$

Das heißt die rechte Seite $\hat{f}$ ist auf natürliche Weise konsistent mit dem zu

$$\begin{aligned} \overline{\Pi}^k_{k-1} &= (\overline{M}^k_k)^{-1}(\widehat{M}^k_{k-1}),\ \text{mit} \\ (\overline{M}^k_k)_{(i,n),(i,m)} &= \langle \vartheta^m_{i,k}, \varphi_{i,k}\vartheta^n_{i,k} \rangle_{L^2(\omega_{i,k}\cap\Omega)} \\ (\widehat{M}^k_{k-1})_{(i,n),(j,m)} &= \langle \varphi_{j,k-1}\vartheta^m_{j,k-1}, \varphi_{i,k}\vartheta^n_{i,k} \rangle_{L^2(\Omega)} \end{aligned} \tag{3.8}$$

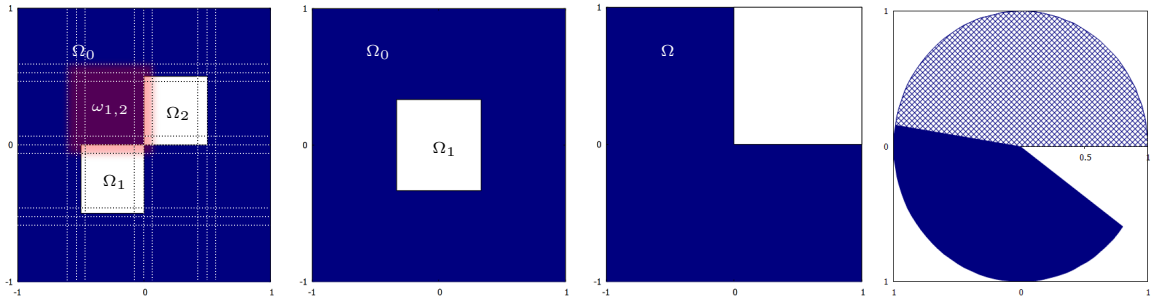
gehörenden Restriktionsoperator $\overline{\Pi}^{k-1}_k := (\overline{\Pi}^k_{k-1})^T = \widehat{M}^{k-1}_k(\overline{M}^k_k)^{-1}$.

3.2.3 Konstruktion effizienter Glätter:

Der dritte, wichtige Bestandteil eines Multilevel-Verfahrens sind die verwendeten Glätter. Diese sollten so konstruiert sein, dass sie stark oszillierende Komponenten des Fehlers effizient glätten. Da die Basisfunktionen in der PUM nicht interpolatorisch sind, können hierfür keine klassichen Glätter verwendet werden, da glatte Koeffizientenfunktionen nicht glatte Funktionen implizieren. Abhilfe schaffen hier die sog. Schwarz Methoden [12, 8], bei denen das Gebiet in überlappende Teilgebiete unterteilt wird, auf denen das Problem dann lokal gelöst wird und die Teillösungen anschließend wieder zusammengefasst werden. Man unterscheidet hierbei in additive und multiplikative Schwarz Methoden. Die sogenannte Block-Jacobi bzw. Block-Gauß-Seidel Iteration sind bekannte Vertreter der beiden Kategorien, die auch hier verwendet wurden. Für nähere Ausführungen und genaue Untersuchungen der Konvergenz- und Glättungseigenschaften der verschiedenen Glätter sowie deren Zusammenspiel mit den Interlevel-Transferoperatoren verweisen wir auf [14, Kapitel 4, S. 64-96]. In dieser Arbeit wurde ausschließlich die Block-Gauß-Seidel Iteration als Glätter verwendet.

3.3 Entwurf robuster Multilevel-Löser

Ausgemachtes Ziel beim Entwurf von Multilevel-Lösern ist es, optimale Komplexität, d.h. eine konstante, hohe Konvergenzrate, zu erhalten und dies unabhängig von der Problemgröße und insbesondere unabhängig von der Regularität der Lösung. Ein optimaler Multilevel-Löser sollte also unabhängig von der Geometrie des Gebiets und für beliebige Koeffizientenfunktionen κ im Wesentlichen gleich gute Konvergenzraten liefern. Offensichtlich stellt dies eine sehr herausfordernde Aufgabe dar, für die wir im Folgenden ausgewählte Teilprobleme und deren Behandlung im Kontext der PUM skizzieren möchten.



(a) zwei Einschlüsse **(b)** einfacher Einschluss **(c)** L-Gebiet **(d)** Kreissegment

Abbildung 3.1: Konfigurationen von Standardfällen zum Test der Robustheit des Lösers.

(1) **Konvexe und nicht konvexe Gebiete:** Durch analytische Untersuchung entsprechender Beispielprobleme kann man zeigen, dass Lösungen des Modellproblems 2.5 mit konvexem Gebiet Ω höhere Regularitätseigenschaften haben, als solche für nicht konvexe Gebiete. Sogenannte einspringende Ecken führen zu Unstetigkeiten im Gradient der Lösung und

produzieren ein singuläres Verhalten der Lösung.
Betrachtet man z. B. das Kreissegment $\Omega_\alpha = \{(r\cos\phi, r\sin\phi) \mid r \in (0,1),\ \phi \in (0, \alpha\pi)\}$ mit Öffnungswinkel $\alpha\pi$ (vgl. Abb. 3.1d). Dann löst die Funktion $u(r,\phi) = r^{\frac{1}{\alpha}} \sin\left(\frac{\phi}{\alpha}\right)$ das Randwertproblem

$$-\Delta u = 0 \text{ in } \Omega_\alpha, \quad u = r^{\frac{1}{\alpha}} \sin(\phi/\alpha) \text{ auf } \partial\Omega_\alpha$$

und man kann zeigen, dass zwar eine Lösung existiert, aber für $\alpha > 1$, also Ω_α nicht konvex, der Gradient ∇u nicht beschränkt ist, und somit $u \notin C^1(\overline{\Omega}_\alpha)$. Standardbeispiel für nicht konvexe Gebiete ist das sog. L-Gebiet, siehe Abb. 3.1c und [9, S. 10].
Gleiches Verhalten beobachtet man für die Geometrien der Materialinterfaces Γ_M, an denen die Koeffizientenfunktion einen signifikanten Sprung aufweist. Betrachten wir beispielsweise Abb. 3.1b und fixieren den Diffusionskoeffizienten $\kappa|_{\Omega_0} = 1$ während wir ihn im Inneren des quadratischen Einschlusses variieren $\kappa|_{\Omega_1} = \epsilon$, haben wir es für $\epsilon \ll 1$ mit einem konvexen Gebiet zu tun, da die Lösung ausserhalb von Ω_1 im Wesentlichen durch die Randwerte bestimmt wird. Für $\epsilon \gg 1$ findet der Diffusionsprozess hauptsächlich in Ω_0 statt - der relevante Gebietsrand hat also einspringende Ecken, was Singularitäten in der Lösung hervorruft.

(2) **Unstetigkeiten in den Koeffizienten:** Sprünge in der Koeffizientenfunktion κ verursachen Unstetigkeiten im Gradienten der Lösung ∇u und die entsprechende Lösung weist an den Sprungstellen (hier: Materialinterfaces) Knicke auf.

Die oben beschriebenen Singularitäten bzw. Knicke der Lösung u sowie Unstetigkeiten in ∇u können durch einen ausschliesslich glatten Ansatzraum (FEM) von Polynomen nicht hinreichend gut approximiert werden. Dies führt in vielen Fällen zu einer schlechten Konvergenz oder gar Divergenz des entsprechenden Multilevel-Lösers, da Grobgitterlösungen, die der tatsächlichen Lösung wenig ähneln, auf feinere Gitter transportiert werden.
Um optimale Konvergenzraten zu erhalten, fordert man deshalb bei der Finiten Elemente Methode üblicherweise, dass die Unstetigkeiten der Koeffizientenfunktion vollständig auf dem gröbsten Gitter (Level) aufgelöst sind, d.h. die Materialinterfaces liegen auf dem Gitter [4]. Dies ist jedoch mit einer Reihe von erheblichen Nachteilen verbunden: Es können unter Umständen sehr viele Verfeinerungsschritte nötig sein, um die Geometrie der Einschlüsse durch das Gitter aufzulösen. Schon die Generierung des Gitters macht einen nicht vernachlässigbaren Anteil im Gesamtrechenaufwand der FEM aus, gleichzeitig wächst jedoch auch die Zahl der Unbekannten. Die Komplexität der Geomterie bzw. der Interfaces bestimmt also die Auflösung des gröbsten Gitters und damit die Dimension des Groblevel-Systems. Für in der Praxis relevante Anwendungen dominiert also schon die Lösung des Grobgittersystems die Gesamtrechenzeit des Lösers.
Im Gegensatz dazu beschränkt man bei gitterfreien Verfahren die Komplexität des Groblevel-Systems auf eine konstante Anzahl von Unbekannten - bei der PUM etwa $\sum_{i=1}^{c_l}(p_i + e_i)$, wobei $c_l = card(C_\Omega^l)$ die Anzahl der Patches auf Level l ist, $p_i = dim(\mathcal{P}_i^{p_i})$ und $e_i = dim(\mathcal{E}_i)$. Insbesondere wird nicht angenommen, dass die Koeffizientensprünge bzw. Materialinterfaces auf dem gröbsten Level geometrisch aufgelöst sind. Um trotzdem gute Groblevel-Approximationen zu erhalten, greift man, wie schon in Kapitel 2 erwähnt, bei der PUM auf einen algebraischen Ansatz zurück, indem die lokalen Ansatzräume mit entsprechenden Anreicherungsfunktionen erweitert werden. So können durch passende Basisfunktionen mit nur wenigen Unbekannten

sehr gute Groblevel-Lösungen erreicht werden.
Dieser Ansatz ist eng mit der Robustheit des verwendeten Multilevel-Lösers verbunden. Da apriori keine Einschränkungen an die verwendenten Anreicherungsfunktionen gestellt werden, kann bei diesem Verfahren so Wissen über Singularitäten und die Regularität der Lösung eingepflegt werden. Insbesondere die oben betrachteten Probleme können so besser aufgelöst werden und führen zu robusteren Lösern, da gerade die nicht glatten Anteile der Lösung besser erfasst werden. Auf die Wahl von passenden Anreicherungsfunktionen werden wir in Kapitel 4 näher eingehen.
Doch gerade für kompliziertere Fälle wie Überlagerungen von Singularitäten o. Ä. rückt die Frage nach der Robustheit des Lösers bei dieser Methode stärker in den Vordergrund:

(3) **Mehrere Einschlüsse, Überlagerung von Singularitäten:** Betrachten wir z. B. mehr als einen Einschluss mit beliebig kleinen Abständen oder Berührpunkten (vgl. Abb. 3.1a), ist das Verhalten der Lösung nicht immer a-priori bekannt bzw. eine komplexe Überlagerung verschiedener Effekte. Die Wahl der Anreicherungsfunktionen hat hier direkten Einfluss auf die Robustheit des Lösers, insbesondere ist nicht klar, ob und in welchen Ausmaß sich lokale Anreicherungsfunktionen in kritischen Bereichen (z. B. unmittelbare Umgebung der sehr nahe beieinander liegenden Ecken der beiden Einschlüsse in Abb. 3.1a) gegenseitig stören bzw. überlagern. Vergleiche hierzu auch den in Abb. 3.1a rot unterlegten Patch $\omega_{1,2}$, der sowohl das Material Ω_1 als auch Ω_2 schneidet, d.h. der lokale Approximationsraum enthält problemabhängige Anreicherungsfunktionen für beide Interfaces, die die Approximationsgüte des jeweiligen Knicks beeinflussen. Insbesondere für den Fall verschiedener Sprunghöhen entstehen hier Probleme bezüglich der Robustheit des Lösers.

(4) **Sehr kleine Schnitte, schlechte Kondition:** Für beliebige Geometrien kann es auf einem oder mehreren Levels zu sehr kleinen Schnitten zwischen Patches und Materialeinschlüssen kommen. Teilt ein Interface einen Patch in zwei (oder mehr) Teile, deren Volumina sich signifikant voneinander unterscheiden (vgl. Abb. 3.1a, Patch $\omega_{1,2}$ hat kleinen Schnitt mit Einschluss), kann es zu Stabilitätsproblemen kommen, die die Kondition der Steifigkeitsmatrix erheblich verschlechtern. Ein robuster Löser sollte auch mit diesen Problemen weitestgehend unabhängig von Geometrie und Höhe des Sprungs umgehen können.

Viele der oben betrachteten Beispielprobleme zur Untersuchung der Robustheit eines Lösers werden wir in Kapitel 5 noch einmal genauer betrachten und die gewonnenen Messdaten einer konkreten Implementierung unter dem Gesichtspunkt der Robustheit diskutieren.

3.4 Realisierung eines Multilevel-Lösers und Einbettung in den Crass Code

Ein großer Teil dieser Arbeit bestand, neben den Untersuchungen des Multilevel-Lösers im Hinblick auf Robustheit, darin, zunächst die Funktionalität des existierenden PUM-Codes [14] in den moderneren, weitaus flexibleren und besser strukturierten Crass-Code (Crack Simualtion Software) zu übertragen. Dieser Code wurde im Rahmen eines Studentenprojekts entwickelt, mit der Intention ein flexibles und mächtiges Framework für die Lösung partieller Differentialgleichungen mit der *Partition of Unity Methode* bereit zu stellen. Besonderes Augenmerk wurde dabei auf Wiederverwendbarkeit und eine aus softwaretechnischer Sicht gute Architektur der Software gelegt.

3.4.1 Stand und Funktionalität des Codes – Zielsetzung

Im Rahmen dieser Arbeit sollte in diesen Kontext ein funktionierender und weitestgehend robuster Multilevel-Löser implementiert werden, der insbesondere über große Flexibilität für den Austausch der Prolongations- und Restriktionsoperatoren verfügt. Der bestehende Crass-Code verfügte bereits über einen Multilevel-Löser, der jedoch nicht zuverlässig und nur stark eingeschränkt für einige Spezialfälle optimales Verhalten zeigte. Der Umgang mit Anreicherungsfunktionen bzw. beliebigen und unterschiedlichen lokalen Approximationsräumen V_i^{pi} war ebensowenig möglich, wie eine zuverlässige Multilevel-Lösung bei adaptiver h- bzw. p-Verfeinerung. Einfache Probleme wie die Poisson-Gleichung mit $\kappa \equiv 1$ bei uniformer Verfeinerung konnten mit akzeptablem Aufwand gelöst werden. Kompliziertere Probleme wie Simulation von Materialeinschlüssen, also κ stückweise konstant, konnten aufgrund fehlender Anreicherungsfunktionen und Stabilitätsproblemen jedoch nur mit einigen hunderttausend Iterationen des Lösers erschlagen werden. Auf Seiten der Implementierung genügte die Umsetzung des Multilevel-Lösers, im Besonderen die Generierung von Prolongations- und Restriktionsoperatoren, nicht den Ansprüchen der Austauschbarkeit und Flexibilität dieses Frameworks. Umgesetzt war ausschließlich die lokal-nach-lokal Projektion 3.7. Die Assemblierung der Matrizen $\widetilde{M}_k^k$ und $\widetilde{M}_{k-1}^k$ war ineinander verschränkt und nicht analog zur Assemblierung der Steifigkeitsmatrix umgesetzt.

3.4.2 Implementierung des Multilevel-Lösers

Für die Realisierung eines funktionierenden und für ein breites Spektrum an Probleminstanzen robusten Multilevel-Lösers waren in dieser Arbeit also vor allem zwei Aspekte von großer Bedeutung: Zum einen sollte die funktionelle Seite von Grund auf überarbeitet werden, mit dem Ziel zunächst mindestens die Funktionalität des alten PUM-Codes sicherzustellen, später jedoch insbesondere den Umgang mit Anreicherungsfunktionen einzubinden und zu optimieren (hierzu mehr in Kapitel 4, speziell 4.3). Vor allem die Generierung der Prolongations- und Restriktionsoperatoren, sowie Teile des Lösungsverfahrens mussten hierfür neu implementiert und erweitert werden. Um ein stabiles Verhalten des Lösers zu garantieren, wurden eine Vielzahl an Tests für verschiedene Probleminstanzen gefahren und mit den in [14] erzielten

Ergebnissen des PUM-Codes verglichen.
Der zweite Hauptaugenmerk galt der Strukturierung und Generalisierung des Prozesses zur Berechnung der Transferoperatoren. Um die Idee des Crass-Codes als flexibles Framework besser umzusetzen, wurde die Gernerierung von Prolongation und Restriktion völlig entkoppelt und sinnvoll in Teilmodule unterteilt, sodass später komfortabel die verschiedenen, in 3.2.2 vorgestellten, Interlevel-Transferoperatoren realisiert werden können. Hierfür wurde zunächst die Assemblierung der benötigten Matrizen voneinander getrennt und strukturell an den Assemblierungsprozess der Steifigkeitsmatrix angepasst. Darüberhinaus kann mit der neuen Implementierung die Restriktion getrennt und unabhängig von der Prolongation aufgestellt werden und ist nicht mehr ausschließlich über Transposition derselben erreichbar. Das hat vor allem im Bezug auf eine Parallelisierung der Berechnung große Vorteile, da das Transponieren einer Matrix in paralleler Rechnung u. U. nicht effizient möglich ist. Eine schematische

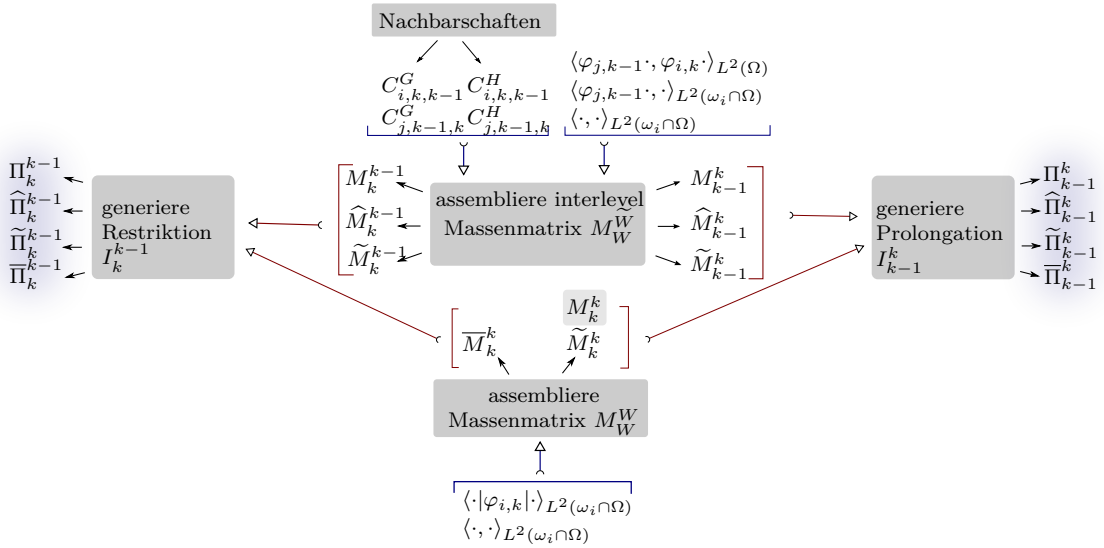


Abbildung 3.2: Schematische Darstellung zur Erzeugung der verschiedenen Prolongations- und Restriktionsoperatoren.

Darstellung zur Generierung der Prolongation und Restriktion ist in Abb. 3.2 gegeben. Anhand dieser Übersicht wollen wir im Folgenden genauer auf die einzelnen Komponenten eingehen. Die beiden wichtigsten Komponenten des Konzepts sind die Funktionen zur Assemblierung der beiden für die L^2-Projektion benötigten Matrizen: Die Interlevel-Massenmatrix $M_W^{\widetilde{W}}$ sowie die (lokale-) Massenmatrix M_W^W. Mithilfe dieser Funktionen kann jede Kombination der beiden Matrizen generiert werden um alle in Abschnitt 3.2.2 vorgestellten Projektionen zu erzeugen. Als Eingabe für die Assemblierungsfunktionen kann eine Codierung der gewünschten (gewichteten) Bilinearform übergeben werden (vgl. Abb. 3.2, blaue Pfeile). Für die Interlevel-Massenmatrix kann zusätzlich die Menge der Nachbarschaften, bezüglich der die Integration durchgeführt wird, angegeben werden: Entweder geometrische Interlevel-Nachbarschaften $C^G_{i,k,k-1}$ bzw. $C^G_{j,k-1,k}$ oder hierarchische Interlevel-Nachbarschaften $C^H_{i,k,k-1}$ bzw. $C^H_{j,k-1,k}$. Die verwendeten Nachbarschaften müssen allerdings mit der angegebenen Bilinearform konform sein.
Das Aufstellen der (vollständigen) Massenmatrix (Abb. 3.2, M_k^k grau unterlegt) ist direkt

in den Assemblierungsprozess der Steifigkeitsmatrix eingebunden, da hier die identischen Nachbarschaften und Funktionsauswertungen benötigt werden und sich lediglich die verwendete Bilinearform unterscheidet. Der Ansatz ist sogar noch allgemeiner, sodass beliebig viele Matrizen und rechte Seiten zu entsprechend definierten (verschiedenen) Bilinearformen und Linearformen innerhalb eines Assemblierungsprozesses erzeugt werden können; verwendet man die L^2-Norm erhält man also die Massenmatrix.
Die beiden Funktionen *generiereProlongation* bzw. *generiereRestriktion* multiplizieren die entsprechenden Matrizen nach vorheriger Invertierung der (lokalen) Massenmatrix. Mit der in Abb. 3.2 dargestellten Methode ist es auch möglich Transferoperatoren über mehrere Levels zu erzeugen.

3.4.3 Verifizierung des Multilevel-Lösers – Untersuchung neuer Funktionalität

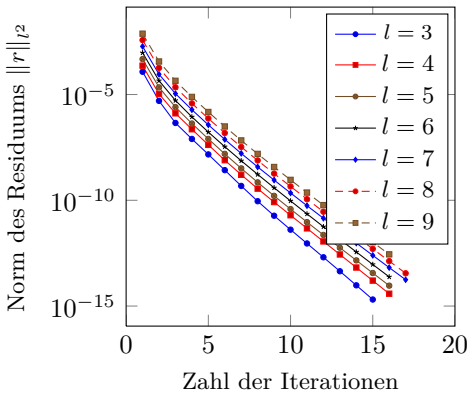
Um zu garantieren, dass das in den Crass-Code implementierte Multilevel-Verfahren korrekt funktioniert und den Anforderungen bezüglich Konvergenz und Stabilität genügt, wurden eine Reihe von Tests und Referenzproblemen durchgerechnet und mit den Ergebnissen des PUM-Codes verglichen. Exemplarisch wollen wir hier das Referenzproblem

$$\begin{aligned} -\Delta u &= 0 \text{ in } \Omega := (0,1)^2, \\ u &= 0 \text{ auf } \partial\Omega \end{aligned}$$

betrachten. Den Überlapp der Patches stellen wir auf $\alpha = 1.3$ ein. In Abbildung (3.3, links) ist die Konvergenz des Verfahrens als Norm des Residuums über die Anzahl der Iterationen für verschiedene Levels, d. h. wachsende Zahl der Unbekannten, dargestellt. Die transportierte Information hierbei ist, dass das Konvergenzverhalten, wie erwartet, unabhängig von der Problemgröße gleich gut ist. In Analogie zu [14, S. 74] wollen wir die mittleren Konvergenzraten

$$\rho_{L^2} := \left(\frac{\tilde{u}_r^T M \tilde{u}_r}{\tilde{u}_o^T M \tilde{u}_0}\right)^{\frac{1}{r}} \quad \rho_{l_2} := \left(\frac{\|\tilde{u}_r\|_{l_2}}{\|\tilde{u}_o\|_{l_2}}\right)^{\frac{1}{r}} \quad \rho_R := \left(\frac{\|A\tilde{u}_r\|_{l_2}}{\|A\tilde{u}_0\|_{l_2}}\right)^{\frac{1}{r}}$$

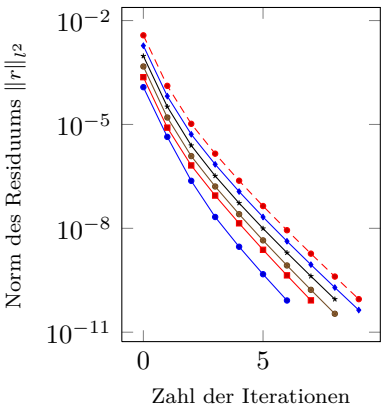
zu einem, bezüglich der L^2-Norm auf 1 normierten, Zufallsstartvektor $\tilde{u}_0$ betrachten. Für diesen zufälligen Startvektor erhalten wir zuverlässige und aussagekräftige Konvergenzaussagen. Die Iteration terminiert, wenn das Residuum unter eine definierte Toleranz gefallen ist: $\|r\|_{l_2} \leq 10^{-10}$. Die Ergebnisse sind in Abbildung (3.3, rechts) aufgetragen. Hier sind die mittleren Konvergenzraten einmal für die Richardson Iteration, also Fehlerreduktion ausschließlich durch das Multilevel, und einmal für ein PCG Verfahren mit dem Multilevelschema als Vorkonditionierer, abgebildet. Die hier erzielten Konvergenzraten sind etwas besser, wie die in [14, S. 85, Tabelle 4.8] dokumentierten Referenzwerte. Dies liegt im Wesentlichen daran, dass für die Referenzwerte über eine größere Anzahl an Iterationen gemittelt wurde und die Konvergenz für ein sehr kleines Residuum nicht mehr so groß ist. Zusammenfassend können wir aber sagen, dass die neue Implementierung zumindest nicht schlechter ist, als die bestehende Referenzimplementierung im PUM-Code.



Level	$\rho_{L^2}^{Rich}$	$\rho_{l^2}^{Rich}$	ρ_{R}^{Rich}	$\rho_{L^2}^{CG}$	$\rho_{l^2}^{CG}$	ρ_{R}^{CG}
3	0.196	0.184	0.152	0.118	0.115	0.106
4	0.221	0.209	0.175	0.148	0.141	0.124
5	0.230	0.219	0.181	0.152	0.145	0.128
6	0.239	0.227	0.186	0.164	0.156	0.133
7	0.247	0.236	0.196	0.165	0.157	0.134
8	0.252	0.242	0.202	0.170	0.162	0.135
9	0.257	0.245	0.203	0.172	0.163	0.135

Abbildung 3.3: Konvergenz eines $M_1^{1,1} =: V(1,1)$-Multilevelschemas mit Block-Gauß-Seidel Glätter und lokal-nach-lokal-Projektion. Links: Konvergenzplot für verschiedene Levels. Rechts: Mittlere Konvergenzraten für Richardson Iteration und PCG

Untersuchung der global-nach-lokal-Projektion – Vergleich: Die Transferoperatoren 3.4 - 3.7 wurden bereits ausführlich in [14, S. 73-94] untersucht. Wir wollen hier einige gewonnene Erkenntnisse über die in dieser Arbeit erstmals untersuchte *gewichtete global-nach-global*-Projektion 3.8 vorstellen. Es sei vorweggenommen, dass sich die Ergebnisse bislang nicht hinreichend mit den theoretischen Überlegungen decken.



Level	Iterationen	
1	$\widetilde{\Pi}_{k-1}^{k}$	$\overline{\Pi}_{k-1}^{k}$
1	1	1
2	6	5
3	7	6
4	8	7
5	9	8
6	9	8
7	10	9
8	11	9
9	11	9

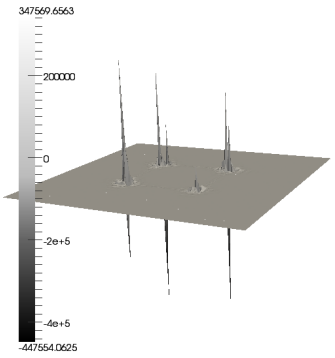


Abbildung 3.4: Untersuchung der gewichteten global-nach-lokal Prolongation bzw. Restriktion. Links: Konvergenzplot für $-\Delta u = 1$ auf Ω, $u = 0$ auf $\partial\Omega$ unter Verwendung von $\overline{\Pi}_{k-1}^{k}$. Mitte: Vergleich der Iterationszahlen für Richardsoniteration. Rechts: 3D Plot von $E(\overline{\Pi}_{k-1}^{k}, \overline{\Pi}_{k}^{k-1})\tilde{u}$

Zunächst wollen wir den einfachen Fall ohne Koeffizientensprünge untersuchen und betrachten hierfür das Problem $-\Delta u = 1$ auf $(-1,1)^2 =: \Omega$, $u = 0$ auf $\partial\Omega$. In Abbildung (3.4, links) ist ein zu Abb. (3.3, links) analoger Konvergenzplot für das Multilevelschema $V(1,1)$ mit $\overline{\Pi}_{k-1}^{k}$, und $\overline{\Pi}_{k}^{k-1}$ als Prolongation bzw. Restriktion dargestellt. In Verbindung mit dem Vergleich der Iterationszahlen (3.4, Mitte), die die Richardson Iteration benötigt, bis das Residuum hinreichend klein ist $\|r\|_{l^2} \leq 10^{-10}$, sehen wir, dass die Verwendung der gewichteten Projektion

zu besseren Ergebnissen führt. Das entspricht auch dem erwarteten Verhalten, da ja mehr Information für den Interlevel-Transfer verwendet wird.

Betrachten wir nun aber für dasselbe Problem einen quadratischen Einschluss $\Omega_1 = (-\frac{1}{4}, \frac{1}{4})^2$, $\Omega_0 := \Omega \setminus \Omega_1$ mit $\kappa|_{\Omega_1} = 10^6$ und $\kappa|_{\Omega_0} = 1$, ergibt sich ein völlig anderes Verhalten. Während die Richardson Iteration mit $\widetilde{\Pi}_{k-1}^k$, $\widetilde{\Pi}_k^{k-1}$ das Residuum in höchstens 30 Iterationen (auf Level 9) unter einen Wert von $\|r\|_{l^2} \leq 10^{-10}$ drückt, divergiert der Multilevel-Löser unter Verwendung der gewichteten Projektion $\overline{\Pi}$. Eine Untersuchung der Qualität des Interlevel-Transfers zeigt, dass prolongierte, glatte Koeffizientenvektoren $\tilde{u}$ bzw. restringierte, glatte Rechte-Seite-Vektoren $\hat{f}$ nahezu mit dem tatsächlichen Wert übereinstimmen. Insbesondere die Restriktion liefert wesentlich bessere Ergebnisse als bei der lokal-nach-lokal-Projektion. Für stark oszillierende Vektoren $\hat{f}$ liefert die gewichtete Restriktion jedoch sehr schlechte Werte. Analog zu [14, S. 90] wollen wir den linearen Operator $E_k(I_{k-1}^k, I_k^{k-1})$ mit

$$E_k(I_{k-1}^k, I_k^{k-1})\tilde{u} = \tilde{u} - I_{k-1}^k (M_{k-1}^{k-1})^{-1} I_k^{k-1} M_k^k \tilde{u}$$

betrachten. Dieser misst die Güte des Interlevel-Transfers und wäre für eine exakte Prolongation bzw. Restriktion konstant 0 für alle Koeffizientenvektoren $\tilde{u}$. Wenden wir auf $E_k(\overline{\Pi}_{k-1}^k, \overline{\Pi}_k^{k-1})$ einen Zufallsvektor $\tilde{u}_{rand}$ an, erwarten wir für gute Transferoperatoren eine gleichförmige Verteilung des Fehlers, insbesondere keine lokale Fehlerverstärkung. Die gemessenen Ergebnisse liefern jedoch gerade letzteres: dass der Fehler in der Umgebung des Interfaces enorm verstärkt wird (Abb. 3.4, rechts). Es ergibt sich also dasselbe Problem wie für die global-nach-lokal-Projektion 3.5, dass die lokalen Basisfunktionen, die im Bereich des Interfaces u. a. aus Anreicherungsfunktionen bestehen, nicht gut durch die (glatten) globalen Groblevel-Funktionen $\varphi_{j,k-1}\vartheta_{j,k-1}^m$ approximiert werden können (vgl. Qualitätsanalysen in [14, S. 90ff]).
Da diese Resultate nicht mit dem in 3.2.2(4) diskutierten vermuteten Verhalten übereinstimmen, wurden eine Vielzahl an Tests und numerischen Versuchen durchgeführt, um eine mögliche Ursache zu lokalisieren. So wurde untersucht, dass die Glättungseigenschaft des verwendeten Block-Gauß-Seidel Verfahrens nicht negativ beeinflusst wird. Leider konnten wir zu keinem befriedigenden Ergebnis gelangen und müssen die endgültige Untersuchung dieses Transferoperators weiterführenden Forschungsarbeiten überlassen.

KAPITEL 4

Anreicherungsfunktionen

In Abschnitt 3.3 haben wir bereits einige Probleme vorgestellt, bei denen unvorteilhafte Geometrien des Gebiets bzw. nicht stetige Koeffizienten zu Singularitäten, Knicken oder allgemein zu Unstetigkeiten in der Lösung führen, die durch glatte Ansatzfunktionen nicht hinreichend gut approximiert werden können. Auf Grundlage dieser Beobachtung haben wir die Verwendung problemabhängiger Anreicherungsfunktionen motiviert, die, wie in 2.1 dargestellt, als Ansatzfunktionen in die lokalen Approximationsräume eingespeist werden. Durch diese algebraische Verfeinerung der lokalen Ansatzräume sind bessere Approximatioen bei deutlich weniger Freiheitsgraden möglich, was hier vor allem für die Berechnung guter Groblevel-Approximationen für den Multilevel-Löser entscheident ist. Im Folgenden wollen wir verschiedene Typen dieser Anreicherungsfunktionen vorstellen und Eigenschaften sowie typische Anwendungsfälle diskutieren. Dieses Kapitel befasst sich ausschließlich mit der Konstruktion der lokalen Approximationsräume

$$V_i(\omega_i) = \operatorname{span}\langle \vartheta_i^m \rangle = \mathcal{P}_i^{p_i}(\omega_i) + \mathcal{E}_i(\omega_i) = \operatorname{span}\langle \psi_i^s,\ \eta_i^t \rangle, \tag{4.1}$$

insbesondere mit der Wahl des problemabhängigen Teils $\mathcal{E}_i(\omega_i)$ der Anreicherungsfunktionen. Vergleiche hierzu die Artikel [18] und [19].
Bemerkung: Für den Moment wollen wir eventuell entstehende lineare Abhängigkeiten der Anreicherungsfunktionen untereinander bzw. mit dem bestehenden glatten Ansatzraum außer Acht lassen. Wir werden uns in Abschnitt 4.2 dieses Kapitels ausführlicher damit befassen.

4.1 Typen von Anreicherungsfunktionen

4.1.1 Anreicherungsfunktionen für Unstetigkeiten in u bzw. ∇u

Für die weiteren Betrachtungen beziehen wir uns auf das Modellproblem 2.5 und die dort eingeführten Definitionen und Bezeichner. Die Koeffizientenfunktion $\kappa(x)$ wird als stückweise konstant angenommen, weist jedoch Sprünge an den Materialinterfaces Γ_M^k auf. Diese Unstetigkeiten in den Koeffizienten führen zu Unstetigkeiten in ∇u, die Lösung u weist also an den Sprungstellen (Materialinterfaces) Knicke auf. Um dieses Verhalten adäquat zu approximieren ohne jedoch alle Sprungstellen geometrisch aufzulösen, muss der Raum der Ansatzfunktionen um Funktionen mit globalem Support erweitert werden, die den Knick an den Interfaces codieren.
Die einfachste Wahl sind stückweise lineare Funktionen η in Ω mit $\eta|_{\Omega_k}$ linear, für $k = 0, \cdots, m$. Die vorzeichenbehaftete Distanzfunktion $\text{sdist}(x, \Gamma_M)$ (signed distance function), die den Abstand des Punktes x zum Materialinterface angibt, scheint also ein guter Kandidat zu sein. Die lokalen Approximationsräume V_i in einer Umgebung des Materialinterfaces, werden dann mit diesen Funktionen angereichert, genauer: Für alle ω_i mit $\omega_i \cap \Gamma_M \neq \emptyset$ definieren wir den Anreicherungsraum $\mathcal{E}_i(\omega_i) \neq \emptyset$. Wir wollen nun verschiedene, auf Distanzfunktionen basierende, Anreicherungsfunktionen vorstellen, die insbesondere auch in dieser Arbeit realisiert und untersucht worden sind.

Gekoppelte Distanzfunktion

Analog zu [19, S. 7] unterteilen wir das Gebiet Ω in zwei disjunkte Teilgebiete $\Omega_- := \Omega_0$ und $\Omega_+ := \bigcup_{i=1}^m \Omega_i$, d.h. ein Teilgebiet für das Matrix-Material und eines für die Materialeinschlüsse. Das Interface ist damit $\Gamma_M = \overline{\Omega}_- \cap \overline{\Omega}_+$, die Materialeinschlüsse werden also nicht mehr unterschieden. Als Grundlage der Anreicherungsfunktionen verwenden wir die oben eingeführte Distanz zum Interface und definieren den zweidimensionalen problemabhängigen Ansatzraum

$$\mathcal{E} = \text{span}\langle \eta_+, \eta_- \rangle \tag{4.2}$$

wobei $\eta_+ := \max(0, \text{sdist}(x, \Gamma_M))$ die Distanz zum Interface im Inneren eines Einschlusses angibt und $\eta_- := \min(0, \text{sdist}(x, \Gamma_M))$ dieselbe ausserhalb in Ω_-. Mit diesem Raum von (globalen) Anreicherungsfunktionen können wir die auftretenden Knicke der Lösung für (glatte) Interfaces hinreichend gut approximieren. Wie in 3.3 erwähnt, erzeugen weniger glatte Interfaces, wie beispielsweise Polygonzüge, zusätzliche Singularitäten in der Lösung, die mit diesen Funktionen nicht erfasst werden.

Entkoppelte Distanzfunktionen

Betrachten wir den obigen Anreicherungsraum 4.2 für Materialeinschlüsse mit verschiedenen Koeffizienten c_M^k, d.h. die Sprünge in der Koeffizientenfunktion $\kappa(x)$ entlang der Interfaces Γ_M^k sind von unterschiedlicher Höhe, treten einige Probleme auf. Koeffizientensprünge unterschiedlicher Höhe implizieren unterschiedlich charakterisierte Knicke in der Lösung u. Da bei

obigem Ansatz ausschließlich eine Distanzfunktion zugrunde liegt, werden unterschiedliche Ausprägungen der Knicke bzw. verschiedene Winkel und Steigungen nicht hinreichend gut durch die Anreicherungsfunktionen wiedergegeben und können daher nicht gut approximiert werden. Im Zusammenspiel mit dem Multilevelschema ergibt sich ein weiteres Problem: Auf groben Levels sind die Werte der Lösung innerhalb verschiedener Einschlüsse Ω_i, Ω_j aufgrund der Funktion η_+ aneinander gekoppelt. Dies spiegelt jedoch nicht die Realität wider, denn die Lösung nimmt i. A. im Inneren der Einschlüsse voneinander unabhängige Werte an. Mit der Hoffnung, dieses Verhalten besser beschreiben zu können, definieren wir einen erweiterten Anreicherungsraum

$$\mathcal{E} = span\langle \eta_-, \eta_{1^+}, \eta_{2^+}, \cdots, \eta_{m^+} \rangle, \tag{4.3}$$

wobei die $\eta_{k^+} := \text{sdist}(x, \Gamma_M^k)$, $k = 1, \cdots, m$ voneinander unabhängige Distanzfunktionen im Inneren der Einschlüsse Ω_k zum jeweiligen Interface Γ_M^k sind. Man beachte, dass die Distanzfunktion für das Komplement η_- nach wie vor gekoppelt ist. Offensichtlich sind viele der η_{k^+} aufgrund der lokalen Anreicherung für $\omega_i \cap \Gamma_M \neq \emptyset$ gleich 0. Auch dieses Problem wird in Abschnitt 4.2 behandelt.

Blended Enrichments

Die sogenannten *blended enrichments* (vgl. [11, S. 45 ff]) (glatt verschmolzene Anreicherungsfunktionen) stellen eine weitere Familie von Anreicherungsfunktionen dar, die jedoch von wesentlich komplexer Gestalt sind. Wichtigster Bestandteil ist nach wie vor die Distanzfunktion zum Interface. Im Gegensatz zu den Funktionen aus 4.2 und 4.3 wollen wir die

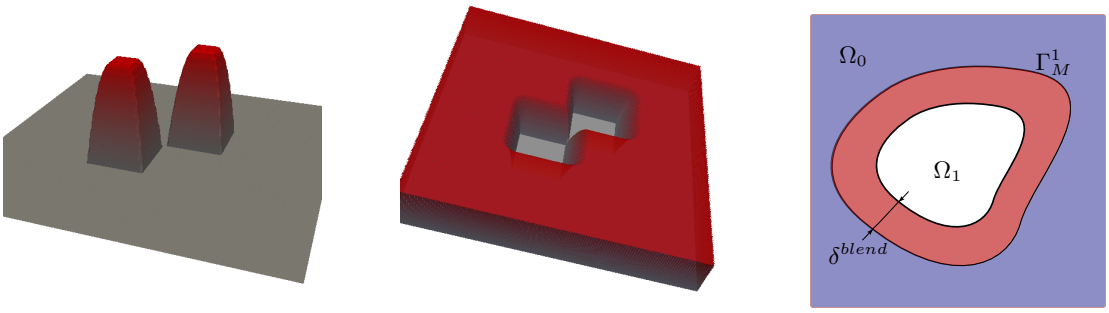


Abbildung 4.1: Beispiele für 'blended Enrichments'. Links: Anreicherungsfunktion η_+^{blend} für zwei quadratische Einschlüsse mit Parameter $\delta^{blend} = 0.12$, $\gamma^{blend} = 2$. Mitte: Entsprechende Anreicherungsfunktion η_-^{blend} für das Komplement. Rechts: Schematische Darstellung zur Konstruktion der Funktionen.

Anreicherungsfunktionen dieser Art so konstruieren, dass sie bis zu einem definierten Abstand δ^{blend} vom Interface Γ_M^k von 0 auf den Wert 1 ansteigen und außerhalb dieses δ^{blend}-Schlauchs glatt auf 0 bzw. 1 fortgesetzt werden (vgl. Abb. 4.1, links, mitte):

$$\eta^{blend} = 1 - \max\left(0, \frac{1 - \text{dist}(x, \Gamma_M^k)}{\delta^{blend}}\right)^{\gamma^{blend}},$$

Hierbei gibt der Parameter δ^{blend} die Breite des Schlauchs an, innerhalb dessen die Funktion von 0 auf 1 wächst bzw. fällt, der Parameter γ^{blend} hingegen kontrolliert die Glattheit des Übergangs. In Abbildung (4.1, rechts) ist eine schematische Darstellung zur Konstruktion dieser Funktionen gegeben. Man beachte, dass der Parameter δ^{blend} direkten Einfluss auf den Betrag der Gradienten hat und somit nicht zu klein gewählt werden sollte. Andererseits ist er durch die Krümmung des Interfaces nach oben beschränkt um Singularitäten (der Distanzfunktion) zu vermeiden. Für weitere Details sei hier auf [11, S. 46, 121 ff] verwiesen. Um nun die entsprechenden Anreicherungsräume zu definieren, können wir für diese Funktionen sowohl den gekoppelten, als auch den entkoppelten Ansatz wählen:

$$\begin{aligned} \mathcal{E} = &\ \text{span}\langle \eta_+^{blend}, \eta_-^{blend} \rangle, \\ \mathcal{E} = &\ \text{span}\langle \eta_-^{blend}, \eta_{1^+}^{blend}, \eta_{2^+}^{blend}, \cdots, \eta_{m^+}^{blend} \rangle. \end{aligned} \tag{4.4}$$

Partition of Unity Anreicherungsfunktionen

Die wohl rechenintensivste Variante der bisher vorgestellten Anreicherungsfunktionen basiert ebenfalls auf Distanzfunktionen, verwendet jedoch zusätzlich eine Partition der Eins, mit der erstere Funktionen geschickt auf den interessanten Bereich eingegrenzt werden. Die so konstruierten Funktionen sollen durch ihren hohen Grad an Unabhängigkeit und Flexibilität insbesondere die weiter oben angesprochenen Probleme bei unterschiedlichen Winkeln der Knicke, verschiedenen Sprunghöhen und schlechten Groblevel-Approximationen im Inneren der Einschlüsse angehen (näheres in Kapitel 5.2). Um die nötige Lokalität und Unabhängigkeit der Anreicherungsfunktionen gleichzeitig mit gewissen Glattheitsanforderungen zu erhalten, definieren wir zu einer Überdeckung $\{\omega_i^{enr}\}$ von Ω die Partition der Eins $\sum_{k=0}^{m} \varphi_k^{enr} \equiv 1$ mit

$$\varphi_i^{enr} = \begin{cases} 1\,, & \text{für } x \in \mathcal{Q}_i \\ 1 - \frac{x}{dist(x, \partial \mathcal{Q}_i)}\,, & \text{für } x \in \omega_i^{enr} \setminus \mathcal{Q}_i \quad , k = 1, \cdots, m \\ 0\,, & \text{sonst} \end{cases}$$

und $\varphi_0^{enr} := 1 - \sum_{k=1}^{m} \varphi_k^{enr}$ für das Matrix-Material. Hierbei sind die $\omega_i^{enr} := \text{supp}(\varphi_i^{enr})$ die Träger der ePU-Funktionen (*enrichment partition of unity*) und $\mathcal{Q}_i$ mit $\Omega_i \subsetneq \mathcal{Q}_i \subsetneq \omega_i^{enr}$ ein das Material umschließendes Gebiet, für das gilt $\varphi_i^{enr}|_{\mathcal{Q}_i} = 1$. Diese sind so gewählt, dass der Überlapp mit dem Matrix-Material $\omega_i^{enr} \cap \omega_0^{enr}$ gerade $\omega_i^{enr} \setminus \mathcal{Q}_i$ ist. In Abb. (4.2, rechts) ist exemplarisch eine Überdeckung für das Problem zweier quadratischer Einschlüsse dargestellt: Aus Gründen der Übersichtlichkeit sind nicht alle Komponenten eingezeichnet. Die rötlich schattierte Fläche ω_1^{enr} ist der Träger von φ_1^{enr}. Darin einbeschrieben ist ein zu $\mathcal{Q}_2$ (leicht grau schattiert) analoges Gebiet $\mathcal{Q}_1$ auf dem φ_1^{enr} gleich 1 ist. Sowohl der Durchmesser $d_{\mathcal{Q}_i}$ der Gebiete $\mathcal{Q}_i$ als auch der Durchmesser δ_i^{PU} des Überlapps der Träger können eingestellt werden und kontrollieren den Betrag der Gradienten $\nabla\varphi_i^{enr}$. Der Träger $\omega_0^{enr} = \Omega \setminus \bigcup_{k=1}^{m} \mathcal{Q}_k$ ist durch den rot schraffierten Bereich skizziert.

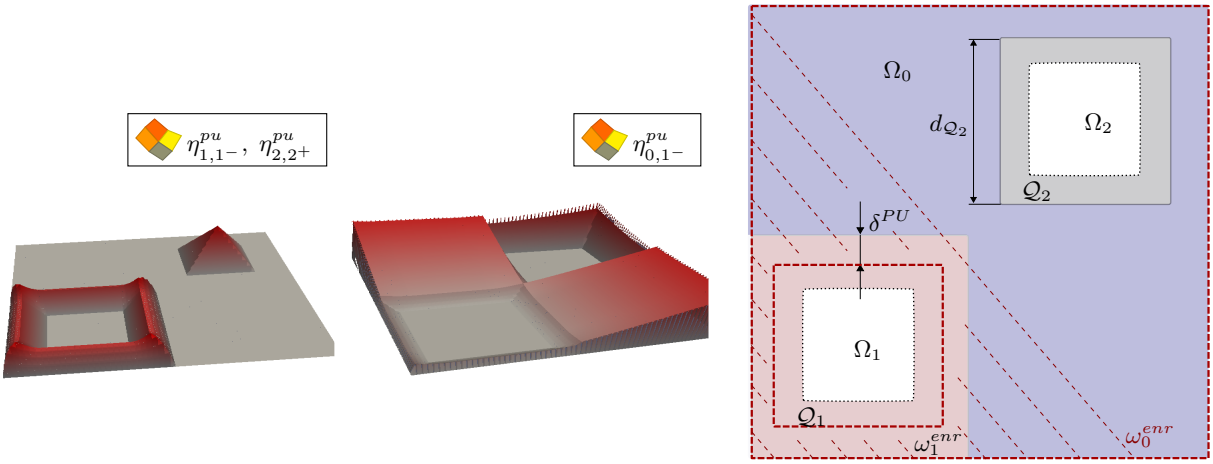


Abbildung 4.2: Links, Mitte: PU-Anreicherungsfunktionen für ein Problem mit zwei quadratischen Materialeinschlüssen. Rechts: Skizze zu Konstruktion und Definition der zugrundeliegenden Partition der Eins $\{\varphi_k^{enr}\}$ am Beispiel zweier quadratischer Einschlüsse.

Mit Hilfe dieser ePU-Funktionen definieren wir nun den Anreicherungsraum

$$\begin{aligned}\mathcal{E} = \ & \operatorname{span}\left\langle \{\operatorname{dist}^-(x,\partial\Omega_k)\} \times \left\{ \begin{array}{c} \varphi_k^{enr}, \\ \varphi_0^{enr} \end{array} \right\}, \left\{\operatorname{dist}^+(x,\partial\Omega_k)\right\} \cdot \varphi_0^{enr} \right\rangle && , \ k = 1, \cdots, m \quad (4.5) \\ =: \ & \operatorname{span}\left\langle \left\{ \eta^{pu}_{k,k^+}, \ \eta^{pu}_{k,k^-}, \ \eta^{pu}_{0,k^-} \right\} \right\rangle && , \ k = 1, \cdots, m \end{aligned}$$

Hierbei bezeichnet $\operatorname{dist}^-(x,\partial\Omega_k)$ die Distanz außerhalb des Materialeinschlusses Ω_k, $\operatorname{dist}^+(x,\partial\Omega_k)$ dieselbe innerhalb und $\eta^{pu}_{j,k^\pm} := \operatorname{dist}^\pm(x,\partial\Omega_k) \cdot \varphi_j^{enr}$.
Der problemabhängige Ansatzraum enthält also pro Einschluss 3 Anreicherungsfunktionen: Eine einfache Distanzfunktion im Inneren der Einschlüsse (vgl. Abb. 4.2, rechts: $\eta^{pu}_{2,2^+}$, links obere Ecke), eine lokale, auf 0 gedämpfte Funktion η^{pu}_{k,k^-} (vgl. Abb. 4.2, links) und eine globale Distanzfunktion η^{pu}_{0,k^-} bzgl. des jeweiligen Interfaces, multipliziert mit der Matrix-Material-ePU-Funktion φ_0^{enr} (vgl. Abb. 4.2, Mitte).
Anmerkung: Selbstverständlich ist das Erzeugendensystem aus 4.5 nicht gleichzeitig Basis des problemabhängigen Ansatzraumes, da auch hier eine Reihe linearer Abhängigkeiten existieren. In Abschnitt 4.2 werden wir in aller Kürze eine Methode vorstellen, die automatisch eine Basis der lokalen Ansatzräume generiert, deren Dimension i. A. deutlich kleiner ist, als die des Erzeugendensystems für dieses Beispiel: $\dim(V_i^{p_i}) \leq p_i + 3(m-1)$. Dies ist vor allem im Hinblick auf die Laufzeit des Verfahrens von großer Bedeutung, da die Dimension des Ansatzraumes den Rechenaufwand der Integration in die Höhe treibt. Jedoch wird die Laufzeit des Verfahrens durch diesen Ansatz nur um einen konstanten Faktor, abhängig von der Anzahl der Einschlüsse, erhöht und fällt somit asymtotisch nicht ins Gewicht.

Die oben definierten Anreicherungsräume können leicht zu Approximationsräumen höherer Ordnung erweitert werden, indem die distanzbasierten Anreicherungsfunktionen η mit Polynomen höheren Grades multipliziert werden. Dieser Ansatz bietet sich vor allem für die

in 4.2, 4.3 und 4.5 definierten Funktionen an. Ebenso können Potenzen dieser Funktinen als Anreicherungsraum betrachtet werden. Siehe hierzu [18, S. 7].

4.1.2 Anreicherungsfunktionen für Singularitäten der Lösung u

In Kapitel 3.3.(1) haben wir bereits gesehen, dass an nicht glatt berandeten Gebieten, insbesondere auch bei Gebieten mit einspringenden Ecken, Singularitäten in der Lösung u entstehen. Ebenso haben wir die Verwendung von passenden Anreicherungsfunktionen für diese und andere Singularitäten motiviert, da diese Eigenschaften der Lösung durch die bisher betrachteten distanzbasierten Anreicherungsfunktionen nicht erfasst werden. Eine Vielzahl dieser Singularitäten sind analytisch bekannt und können dem lokalen Ansatzraum additiv hinzugefügt werden. So existieren beispielsweise für beliebige Polygonzüge (als Gebietsränder) bekannte Funktionen, die das Verhalten der Lösung in den Ecken für bestimmte PDGl beschreiben (für die Poisson-Gleichung 2.5 vgl. [18, S. 8]). Eine solche Funktion für das in 3.1.(c) eingeführte L-Gebiet ist in (4.3, links) dargestellt.
Von großer Bedeutung sind diese singulären Anreicherungsfunktionen auch bei der Simulation von Rissen. Die hier entstehenden Lösungseigenschaften sind ebenfalls analytisch bekannt und können dem Verfahren in der Nähe der Risse im Vorfeld bekannt gemacht werden. Genauere Ausführungen hierzu sind in [15] und [17] zu finden.
Im Gegensatz zu obigen Distanzfunktionen haben die Singularitäten meist nicht nur lokale Auswirkungen, sondern beeinflussen die Lösung auf weiten Teilen des Gebiets. Eine ausschließlich lokale Anreicherung auf Patches, die die entsprechenden Risse oder Ecken schneiden wäre also nicht ausreichend. An dieser Stelle wollen wir an den Charakter des Multilevels und das Prinzip der sukzessiven Verfeinerung hinweisen. Auf gröberen Levels werden die Anreicherungsfunktionen also auf hinreichend großen Gebieten bekannt gemacht.

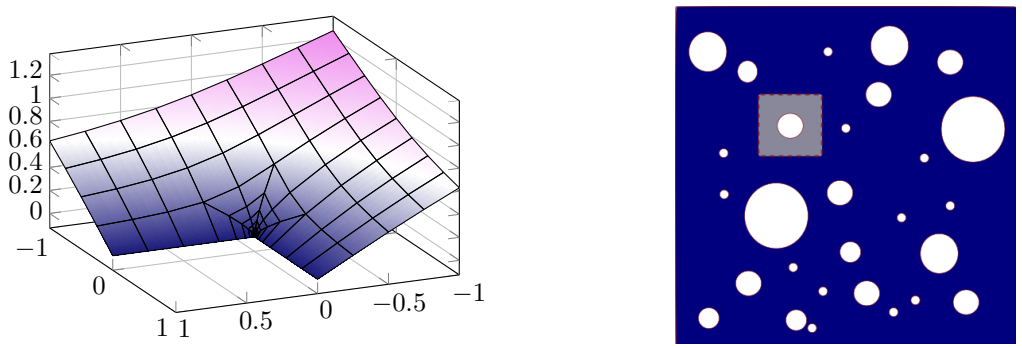


Abbildung 4.3: Links: Singularitätenfunktion für das in 3.1.(c) dargestellte L-Gebiet: $u(r,\phi)=r^{\frac{2}{3}}\sin\left(\frac{2\phi-\pi}{3}\right)$. Rechts: Probleminstanz mit vielen zufällig angeordneten, kreisförmigen Materialeinschlüssen. Vorberechnung einer lokalen Lösung (grau unterlegt) als numerische Anreicherungsfunktion.

4.1.3 Numerische Anreicherungsfunktionen

Die bisher betrachteten Anreicherungsfunktionen waren jeweils für die Codierung einer speziellen Eigenschaft der Lösung konstruiert. Treten jedoch mehrere verschiedene solcher Charakteristika, wie Knicke, Unstetigkeiten und Singularitäten, auf, so müssen die eingeführten Anreicherungsfunktionen geschickt kombiniert werden, was nicht zuletzt die Dimension der lokalen Approximationsräume in die Höhe treibt. Weiterhin ist nicht klar, in wieweit sich die Anreicherungsfunktionen gegenseitig beeinflussen oder stören. Insbesondere sollte nicht mehr als eine Funktion zur Auflösung von Singularitäten in einem lokalen Ansatzraum existieren.
Ein sehr vielversprechender Ansatz besteht hier in der Verwendung numerischer Anreicherungsfunktionen. Hierbei werden lokale Teilprobleme vorberechnet und die Lösung dieser einfacheren Probleme später als Anreicherungsfunktion für das ursprüngliche, komplexere Problem verwendet. Besonders effizient ist diese Methode für viele ähnliche Einschlüsse, wie z. B. in Abb. (4.3, rechts) dargestellt. Eine vorberechnete, lokale Lösung kann dann in Form einer Tabelle als Anreicherungsfunktion verwendet werden. Da für das Multilevel gerade eine gute Groblevel-Approximation von Interesse ist, reicht es, die Teillösungen auf einem recht grob aufgelösten Level abzuspeichern, beispielsweise Level 4. Der Speicherverbrauch kann also moderat beschränkt werden und ist kein limitierender Faktor. Diese numerischen Anreicherungsfunktionen wurden in dieser Arbeit nicht betrachtet, stellen jedoch den wohl vielversprechendsten Ansatz dar und sollten Thema weiterführender Arbeiten sein.

4.2 Stabile Transformation

Wie bereits angemerkt, ist $\text{span}\langle\{\eta_i^t\}\rangle$ aufgrund linearer Abhängigkeiten i. A. keine Basis des Raumes der problemabhängigen Anreicherungsfunktionen $\mathcal{E}_i$, sondern lediglich ein Erzeugendensystem. Damit ist auch $V_i = \text{span}\langle\psi_i^s,\ \eta_i^t\rangle$ ein Erzeugendensystem, was zu einer schlecht konditionierten bzw. singulären Steifigkeitsmatrix bzw. Massenmatrix führt. Dies macht das Lösen des LGS und insbesondere das Invertieren der (lokalen) Massenmatrix für die Interlevel-Transferoperatoren unmöglich.
Grund für die linearen Abhängigkeiten ist zum einen, dass die Anreicherungsfunktionen η^t global gesehen zwar linear unabhängig sind, die lokale Einschränkungen $\eta_i^t := \eta^t|_{\omega_i}$ aber durchaus schon in $\mathcal{P}_i^{p_i}$ enthalten sind oder gut durch Funktionen dieses Raumes approximiert werden können. Dies ist vor allem bei den in 4.2, 4.3 und 4.5 vorgestellten Ansätzen der Fall. Zum anderen ist in 4.3, 4.5 ein Großteil der Ansatzfunktionen lokal auf den $\omega_i \cap \Gamma_M \neq \emptyset$ gleich Null, was zu einem degenerierten System führt. Im Folgenden werden wir ein Verfahren skizzieren, das eine stabile Basis $\langle\tilde{\vartheta}_i^n\rangle := \langle\tilde{\psi}_i^s, \tilde{\eta}_i^t\rangle$ für die lokalen Approximationsräume liefert, sodass diese eine Partition $V_i = \mathcal{P}_i^{p_i} + \mathcal{E}_i = \mathcal{P}_i^{p_i} \oplus (\mathcal{E}_i \setminus \mathcal{P}_i^{p_i})$ darstellen. Diese stabile Transformation wurde in [15], [17] und [16, S. 65-71] entwickelt und die folgenden Erklärungen orientieren sich hieran.

Betrachten wir einen Blockeintrag der lokalen Massenmatrix $\widetilde{M}_k^k$ bezüglich des erzeugenden Systems $\langle\vartheta_i^n\rangle$, lässt sich dieser wiederum in Blöcken schreiben als

$$M_i = \begin{pmatrix} M^i_{\mathcal{P},\mathcal{P}} & M^i_{\mathcal{P},\mathcal{E}} \\ M^i_{\mathcal{E},\mathcal{P}} & M^i_{\mathcal{E},\mathcal{E}} \end{pmatrix}$$

Hierbei bezeichnet $M^i_{\mathcal{P},\mathcal{P}}$ mit $\dim(M^i_{\mathcal{P},\mathcal{P}}) = d_i^{\mathcal{P}}$ den Block der polynomiellen Ansatzfunktionen, $M^i_{\mathcal{E},\mathcal{E}}$ mit $\dim(M^i_{\mathcal{E},\mathcal{E}}) = d_i^{\mathcal{E}}$ den Block der Anreicherungsfunktionen und $M^i_{\mathcal{E},\mathcal{P}}$ bzw. $M^i_{\mathcal{P},\mathcal{E}}$ sind gemischte Blöcke mit jeweiligen Dimensionen. Lineare Abhängigkeiten der η_i^t führen dazu, dass $M^i_{\mathcal{E},\mathcal{E}}$ schlecht konditioniert ist bzw. singulär wird, da der Kern der Matrix nicht Null ist oder zumindest viele Elemente im Bild sehr nahe an Null sind (numerischer Nicht-Null-Kern). Um zu einer stabilen Basis zu gelangen betrachten wir die Eigenwertzerlegung (der Übersichtlichkeit halber unterdrücken wir den Index i hier und im Folgenden):

$$O_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{E}} O_{\mathcal{E}} = D_{\mathcal{E}} \quad \text{mit} \quad O_{\mathcal{E}}^T O_{\mathcal{E}} = \mathbb{I}_{d_i^{\mathcal{E}}} \quad \text{und} \quad D_{\mathcal{E}} = \operatorname{diag}(\lambda_1^{\mathcal{E}}, \cdots, \lambda_{d_i^{\mathcal{E}}}^{\mathcal{E}})$$

Angenommen die Eigenwerte $\lambda_i^{\mathcal{E}}$ sind absteigend sortiert, so können wir $O_{\mathcal{E}}^T$ bzw. $D_{\mathcal{E}}$ bezüglich eines Abschneide-Parameters ϵ partitionieren in

$$O_{\mathcal{E}}^T = \begin{pmatrix} \tilde{O}_{\mathcal{E}}^T \\ K_{\mathcal{E}}^T \end{pmatrix} \quad D_{\mathcal{E}}^T = \begin{pmatrix} \tilde{D}_{\mathcal{E}} & 0 \\ 0 & k_{\mathcal{E}} \end{pmatrix}$$

Hierbei sind die Zeilen von $K_{\mathcal{E}}^T$ Eigenvektoren zu den entsprechenden sehr kleinen Eigenwerten $k_{\mathcal{E}}$, die aus Stabilitätsgründen verworfen werden.

Mittels der Projektion $\Pi^* := \tilde{D}_{\mathcal{E}}^{-\frac{1}{2}} \tilde{O}_{\mathcal{E}}^T \colon \langle \eta^t \rangle \to \langle \tilde{\eta}^t \rangle$ erhalten wir dann eine stabile Basis des lokalen Anreicherungsraumes $\mathcal{E}_i = \langle \tilde{\eta}_i^t \rangle$ und es gilt:

$$M^*_{\mathcal{E},\mathcal{E}} := \Pi^* M_{\mathcal{E},\mathcal{E}} (\Pi^*)^T = \tilde{D}_{\mathcal{E}}^{-\frac{1}{2}} \tilde{O}_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{E}} \tilde{O}_{\mathcal{E}} \tilde{D}_{\mathcal{E}}^{-\frac{1}{2}} = \mathbb{I}_{\tilde{d}^{\mathcal{E}}},$$

wobei $\tilde{d}^{\mathcal{E}} \leq d^{\mathcal{E}}$ die Dimension von $\langle \tilde{\eta}^t \rangle$ ist. Über den Operator

$$T_{\mathcal{E}}^T = \begin{pmatrix} \mathbb{I}_{d^{\mathcal{P}}} & 0 \\ 0 & \Pi^* \end{pmatrix} \quad \text{und} \quad M_{\mathcal{E}} = T_{\mathcal{E}}^T M T_{\mathcal{E}} = \begin{pmatrix} M_{\mathcal{P},\mathcal{P}} & M^*_{\mathcal{P},\mathcal{E}} \\ M^*_{\mathcal{E},\mathcal{P}} & M^*_{\mathcal{E},\mathcal{E}} \end{pmatrix}$$

berechnen wir die Matrix $M_{\mathcal{E}}$ mit stabiler Basis für $\mathcal{P}$ und $\mathcal{E}$. Um nun noch die polynomiellen Komponenten aus dem Anreicherungsraum zu entfernen, berechnen wir das Schurkomplement von $M_{\mathcal{E}}$ als $M_{\mathcal{D},\mathcal{D}} = M^*_{\mathcal{E},\mathcal{E}} - M^*_{\mathcal{E},\mathcal{P}} (M_{\mathcal{P},\mathcal{P}})^{-1} M^*_{\mathcal{P},\mathcal{E}}$ und eliminieren analog zu oben dessen (numerischen) Nicht-Null-Kern durch Abschneiden kleiner Eigenwerte. Das Resultat ist eine stabile Basis für die lokalen Approximationsräume, insbesondere besteht der Anreicherungsteil $\tilde{\mathcal{E}} \approx \mathcal{E} \setminus \mathcal{P}^{p_i}$ – abhängig vom Abschneide-Prameter ϵ – (ausschließlich) aus nicht glatt approximierbaren Komponenten. Dieser Abschnitt soll nur die Idee der stabilen Transformation übermitteln, um eine Vorstellung der Arbeitsweise zu bekommen. Ausführlich wird diese Transformation in den oben genannten Arbeiten behandelt.

4.3 Realisierung der Anreicherungsfunktionen sowie der stabilen Transformation und Einbettung in den Crass Code

Mit dem Ziel einen effizienten, stabilen und weitestgehend robusten Multilevel-Löser für die PUM in den Kontext des Crass Fameworks zu implementieren, mussten die in diesem Kapitel vorgestellten Komponenten realisiert werden. Nachdem der Multilevel-Löser – ohne Unterstützung der Anreicherungsfunktionen – auf seine Funktionalität und Effizienz hin getestet und verifiziert wurde, liegt der Fokus nun auf der Implementierung und Anwendung

der verschiedenen Anreicherungsfunktionen. Wie wir gesehen haben, ist die Implementierung der stabilen Transformation, im Zusammenhang mit der Verwendung von Anreicherungsfunktionen für den Multilevel-Löser, unerlässlich, da sonst weder die (lokale) Massenmatrix invertiert, noch das Gleichungssystem gelöst werden kann.

4.3.1 Stand und Funktionalität des Codes - Zielsetzung

Bei der Planung des Crass Frameworks war die spätere Verwendung von Anreicherungsfunktionen bereits angedacht (diese stellen ja gerade einen der größten Vorteile der PUM dar) und entsprechende Schnittstellen waren in der Architektur des Codes integriert. Die Aufgabe bestand in der Realisierung unterschiedlicher Anreicherungsfunktionen, die zur Analyse der Robustheit des Lösers und dessen Reaktion auf verschiedene Funktionen leicht austauschbar organisiert werden sollten. Weitaus mehr Aufwand war es jedoch, den bestehenden Multilevel-Löser so zu erweitern, dass beliebige Arten von (sinnvollen) Anreicherungsfunktionen eingesetzt werden können, ohne, dass dieser aufgrund schlecht konditionierter Matrizen versagt. In diesem Zusammenhang ist insbesondere die Implementierung der stabilen Transformation von großer Bedeutung, die ihrerseits sensibles Zusammenspiel verscheidener Komponenten ist – so ist beispielsweise die Wahl des Abschneide-Parameters ϵ nicht unerheblich.

4.3.2 Implementierung

Die in diesem Kapitel in Abschnitt 4.1.1 vorgestellten Anreicherungsfunktionen wurden in dieser Arbeit realisiert und getestet. Während die in 4.2 vorgestellten Anreicherungsfunktionen bereits in [19] in Verbindung mit der PUM untersucht wurden, sind der entkoppelte Ansatz in 4.3 sowie die PU-Anreicherungsfunktionen in 4.5 erstmals in dieser Arbeit untersucht. Der Ansatz der 'blended enrichments' 4.4 wurde bereits in [11] für verallgemeinerte Finite Elemente Verfahren untersucht.
Da alle diese Funktionen letztendlich auf einer Distanzfunktion basieren, ist vor allem deren effiziente Implementierung von Bedeutung. Die Geometrie der Materialinterfaces wird entsprechend der Levels durch sukzessive Verfeinerung mithilfe bzgl. der x-Achse monotoner Multi-Liniensegmenten approximiert. Zur Berechnung der numerischen Distanzfunktion wird für einen Punkt die minimale Distanz zu diesen Multi-Liniensegmenten gewählt, d.h. die Komplexität ist linear in der Anzahl der Segment-Ketten. Durch geeignete Vorverarbeitung in Verbindung mit geschickten Datenstrukturen kann die Effizienz der Implementierung gesteigert werden. Da die gesamte Geometriebehandlung im Crass Framework bereits im Rahmen einer anderen Arbeit neu strukturiert und überarbeitet wird, war die Optimierung dieser Berechnung von untergeordnetem Interesse. Jedoch kann die Distanzfunktion auf gewisse Materialinterfaces beschränkt werden. So kann entweder die Distanz bezüglich Γ_M, also der Vereinigung aller Interfaces berechnet werden oder, wie insbesondere für die PU-Anreicherungsfunktionen benötigt, nur die Distanz bezüglich ausgewählter Interfaces Γ_M^k evaluiert werden.
Die Anreicherungsfunktionen wurden hier nur in additiver Form realisiert. Hierfür werden benötigte Distanzfunktionen zu einem lokalen, additiven Anreicherungsraum hinzugefügt, der die entsprechende Variante 4.2 - 4.5 der Anreicherungsfunktionen implementiert.

Um einen funktionierenden Löser zu erhalten müssen, alle Operatoren, Koeffizientenvektoren und Rechte-Seite-Vektoren in die stabile Basis transformiert werden. Insbesondere muss das lineare Gleichungssystem $A\tilde{u} = \hat{f}$ und die Interlevel-Transferoperatoren transformiert werden – die Invertierung der (lokalen) Massenmatrix ist in der nicht stabilen Basis ohnehin nicht möglich. Bezeichne im Folgenden P_k den Operator zur Transformation in die stabile Basis auf Level k. Prolongations- und Restriktionsoperator erhalten wir dann wie folgt:

$$I_{k-1}^{k} = \left(P_k M_k^k P_k^T\right)^{-1} P_k M_{k-1}^k P_{k-1}^T$$
$$I_k^{k-1} = P_{k-1} M_k^{k-1} P_k^T \left(P_k M_k^k P_k^T\right)^{-1},$$

wobei M_k^k die verwendete Variante der (lokalen) Massenmatrix auf Level k ist und M_{k-1}^k bzw. M_k^{k-1} die passende Interlevel-Matrix. Analog transformieren wir das LGS

$$P_k A_k P_k^T P_k \tilde{u} = P_k \hat{f}.$$

KAPITEL 5

Untersuchung der Robustheit des Multilevel-Lösers

Schwerpunktthema dieser Arbeit war neben der Implementierung eines stabilen Multilevel-Lösers dessen Analyse bezüglich Robustheit für ausgewählte Probleminstanzen. Insbesondere sollte dabei das Verhalten des Lösers für verschiedene Anreicherungsfunktionen untersucht werden, mit dem Ziel einen möglichst robusten, d. h. für ein weites Spektrum an durchaus problematischen Probleminstanzen nahezu gleich gut funktionierenden, Multilevel-Löser zu erhalten. Dieses Kapitel zeigt die Ergebnisse dieser Analyse auf und diskutiert mögliche Ursachen und Verbesserungsansätze für nicht robustes Verhalten des Lösers. Es stellt somit den Hauptteil dieser Arbeit dar, setzt jedoch all die zuvor erarbeiteten Konzepte und die vorgestellte Theorie samt deren Realisierung vorraus. Da ein optimaler Multilevel-Löser ein hochsensibles Zusmmenspiel vieler exakt aufeinander abgestimmter Komponenten ist, ist für die Untersuchung desselben eine genaue Kenntnis und ein tiefes Verständnis all dieser Teilbereiche nötig. Wir werden in diesem Kapitel in erster Linie auf die in Abschnitt 3.3 vorgestellten Problemfelder eingehen und dabei u. a. die Verwendung der in 4.1.1 präsentierten Anreicherungsfunktionen motivieren. Leider konnte bislang nicht jedes Verhalten des Lösers restlos erklärt werden, jedoch wurden durchaus Ergebnisse erziehlt, die die Resultate früherer Arbeiten verbessern.
Wie in den Abschnitten 3.4 und 4.3 bereits erwähnt, war die Zielsetzung bereits vorhandene Funktionalität sowie neue Erweiterungen zusammen in das Framework des Crass Codes einzubinden. Dieses befindet sich momentan stark in der Entwicklung, weshalb vor der eigentlichen Robustheitsanalyse einige wesentliche Komponenten neu entwickelt bzw. überarbeitet werden mussten. Weiterhin war dieser Entwicklungsprozess des Frameworks Grund für diverse Schwierigkeiten und Probleme, beispielsweise bei der Berechnung und Behandlung der Geometrie sowie bei der parallelen Rechnung der PUM. So war man in dieser Arbeit mit einigen limitierenden Faktoren konfrontiert: Aufgrund teilweise fehlerhafter Geometriebehandlung konnten nicht beliebige Geometrien auf allen Levels betrachtet werden – ebenso war, da man

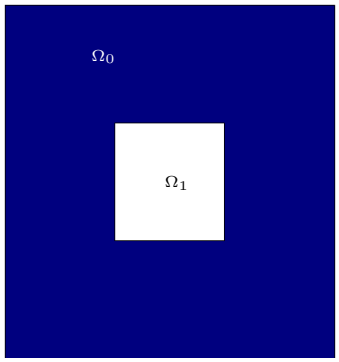
auf serielle Rechnung eingeschränkt war, eine maximale Verfeinerung nur bis Level 9 möglich. Im Weiteren werden wir zunächst, wie schon in Abschnitt 3.4.3, die bereits im PUM-Code existente Funktionalität verifizieren und die Ergebnisse der Implementierung mit den in [19] dokumentierten Werten vergleichen. Anschließend folgt eine genauere Robustheitsanalyse für ausgewählte, im Bezug auf Robustheit des Lösers kritische, Probleme.

Robustheitsanalyse für ausgewählte Probleme

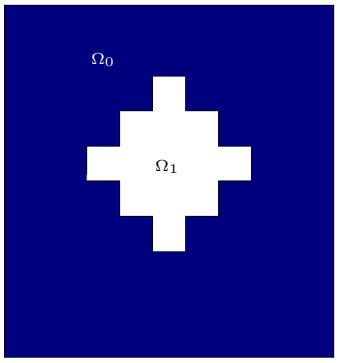
Im Folgenden stellen wir die Ergebnisse der numerischen Experimente vor. Wir beziehen uns auf das Modellproblem 2.5 und betrachten analog zu [19] Diffusionsprobleme mit einer stückweise stetigen Koeffizientenfunktion $\kappa(x)$, die die Eigenschaften eines Matrix-Materials und der darin enthaltenen Materialeinschlüsse modelliert. Üblicherweise fixieren wir den Diffusionskoeffizienten im Matrix-Material auf 1, betrachten jedoch verschiedene Werte des Koeffizienten innerhalb der Materialeinschlüsse: $\kappa|_{\Omega_0} = 1,\ \kappa|_{\Omega_k} =: \epsilon \in \{10^{-6}, 10^{-4}, 10^{-2}, 1, 10^2, 10^4, 10^6\}$. Wenn nicht anderst festgelegt, setzen wir die rechte Seite $\hat{f} = 0$ und verwenden $\Omega := (-1,\ 1)^2$ mit Dirichlet-Randbedingungen $g_D = 0$ für $x = -1$ und $g_D = 1$ für $x = 1$ sowie homogene Neumann-Randbedingungen $g_N = 0$ für die verbleibenden Ränder.
Für die Lösung des Problems mit der PUM unter Verwendung eines Multilevel-Lösers betrachten wir uniforme Verfeinerungen der Überdeckungen mit quadratischen Patches und Streckungsparameter $\alpha = 1.2$ für den Überlapp. Die glatten Approximationsräume $\mathcal{P}_i$ enthalten Polynome bis zum Grad 1. Aufgrund fehlerhafter Geometriebehandlung auf dem gröbsten Level 1, beschränken wir uns auf Level 2 als initiales Level, das aus 16 Patches besteht und mit einfachen, gekoppelten Distanzfunktionen maximal 80 Freiheitsgrade hält.
Das resultierende lineare Gleichungssystem $A\tilde{u} = \hat{f}$ wird mit dem Conjugierten-Gradienten-Verfahren mit einem Multilevel-Schema als Vorkonditionierer gelöst. Für das Multilevel verwenden wir einen Block-Gauß-Seidel Glätter sowie die lokal-nach-lokal Projektion als Interlevel-Tansfer und betrachten eine $V(1,1)$ Iteration. Den Startvektor initialisieren wir mit $\tilde{u}_0 = 0$ und verwenden nicht die prolongierte Lösung des vorherigen Levels. Die Iteration terminiert, wenn das Residuum $\|f - A\tilde{u}_j\|_{l^2} = \|r_j\|_{l^2} \leq 10^{-10}$. Kriterium für die Effizienz des Multilevel-Verfahrens ist die Anzahl der Iterationen, die das PCG-Verfahren benötigt, bis die Norm des Residuums unter den vorgeschriebenen Wert fällt. Diese Zahl steht in direktem Zusammenhang mit der Konvergenzrate des Multilevels.

5.1 Verifizierung und Vergleich einiger Ergebnisse mit Resultaten des PUM-Codes

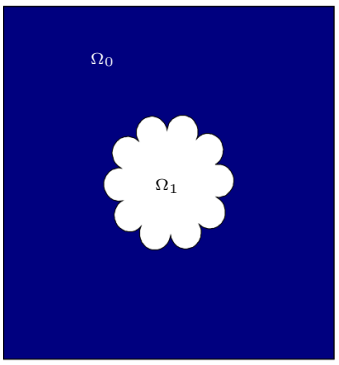
Die in [19] gerechneten Referenzprobleme sind Ausgangspunkt dieser Arbeit. Wir wollen nun unsere Resultate mit den dort präsentierten Ergebnissen vergleichen. Hierfür betrachten wir die folgenden Konfigurationen (vgl. [19, S. 3]):



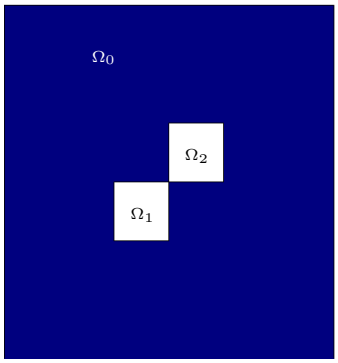
(a) ein quadratischer Einschl.



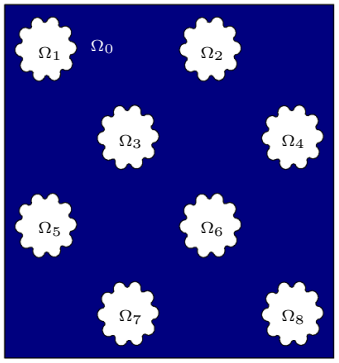
(b) ein sternähnlicher Einschl.



(c) ein zahnradähnlicher Einschl.



(d) zwei quadratische Einschl.



(e) viele zahnradähnliche Einschl.

Abbildung 5.1: Referenzprobleme ähnlich zu [19] zum Vergleich der Resultate. (a) einfacher quadratischer Einschluss, einfache Geometrie. (b) sternähnlicher Einschluss: nichtkonvexes Gebiet, Einfluss von Singularitäten. (c) zahnradähnlicher Einschluss: komplexe Geometrie, Einfluss von Singularitäten. (d) zwei quadratische Einschlüsse: Berührpunkt der Einschlüsse problematisch. (e) viele zahnradähnliche Einschlüsse: Abhängigkeit der Einschlüsse, komplexe Geometrie.

Problem 1: Quadratischer Einschluss

Zunächst betrachten wir einen einfachen quadratischen Einschluss $\Omega_1 = (-\frac{1}{3}, \frac{1}{3})^2$. Diese Konfiguration soll hauptsächlich als Referenz für spätere Experimente dienen, da die Geometrie gut auf groben Gittern aufgelöst werden kann und zumindest für $\epsilon \gg 1$ der Einfluss von Singularitäten sehr gering ist. In Tabelle 5.1 ist die Anzahl der Iterationen für verschiedenes ϵ gegeben, die der PCG-Löser Löser benötigt, bis das Residuum auf einen Wert kleiner 10^{-10} gefallen ist. Wir können vor allem zwei Dinge feststellen: Zum einen sieht man sehr deutlich, dass der Löser vollständig unabhängig von der Sprunghöhe und unabhängig vom Level arbeitet. Die Anzahl der Iterationen kann als konstant angesehen werden und erreicht einen maximalen Wert von 13. Weiterhin ist die Varianz um den Referenzfall $\epsilon = 1$ – für den praktisch kein Einschluss existiert – mit max. einer Iteration verschwindend, d. h. der Löser verhält sich für dieses Problem vollständig robust.
Vergleicht man die Werte mit den Resultaten des PUM-Codes in [19, S. 10, Tab. 1] fällt zum anderen auf, dass die Zahl der Iterationen für unser Experiment in Tabelle 5.1 signifikant

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	8	8	8	8	8	9	9
4	10	10	10	10	11	11	11
5	10	10	10	10	11	11	12
6	11	11	11	10	12	12	12
7	11	11	11	11	12	12	12
8	11	11	11	11	12	13	13
9	11	11	11	11	12	12	12

Tabelle 5.1: Ergebnisse, Problem (1): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für einen quadratischen Einschluss (vgl. Abb. 5.1d).

kleiner ist. So benötigt der Löser des PUM-Codes im schlechtesten Fall 26 Iterationen, um die gewünschte Genauigkeit zu erreichen, im Vergleich zu den 13 Iterationen hier. In beiden Experimenten wurden dieselben Anreicherungsfunktionen, nämlich einfache gekoppelte Distanzfunktionen (vgl. 4.2), verwendet. Dieser Unterschied ist Produkt mehrerer Faktoren: Bei den Messungen in [19] wurde ab Level 1 gerechnet, wohingegen wir auf Level 2 als gröbstes Level limitiert sind. Da die gröberen Levels typischerweise nicht so gut aufgelöst werden können und deren Approximationsqualität i. A. eher schlecht ist, wirkt sich dies also negativ auf das Multilevel-Verfahren aus, was die schlechteren Iterationszahlen teilweise erklären kann. Desweitern ist die Integration im Crass Code exakter und besser umgesetzt, ebenso wie die Behandlung der Geometrie. Nicht zuletzt wird auch das Regularisierungsgewicht für die Randwerte genauer ausgewertet – zusammengefasst ist also der Fehler geringer und damit die Approximationsqualität höher. Als Konsequenz arbeitet das Multilevel effizienter, da weniger lösungs-averse Komponenten von den groben auf die feinen Level transportiert werden. Vor allem letzteres ist höchstwahrscheinlich in großem Maße für die kleineren Iterationszahlen verantwortlich, was gleichzeitig einen effizienteren Löser bescheinigt.

Problem 2: Sternähnlicher Einschluss

Als zweite Problemkonfiguration betrachten wir einen sternähnlichen Einschluss innerhalb des Rahmens $(-\frac{1}{4}, \frac{1}{4})^2$ (vgl. leicht vergrößerte Darstellung in Abb. 5.1b). Die Lösung dieses Problems besitzt sowohl für sehr große, als auch für sehr kleine ϵ, einen ausgeprägteren singuläreren Charakter, da das Gebiet in beiden Fällen nicht konvex ist (vgl. Abschnitt 3.3). Wir erwarten also einen leichten Anstieg der Iterationszahlen, da diese Eigenschaften der Lösung durch die verwendeten einfachen Distanzfunktionen nicht erfasst werden.
In Tabelle 5.2 sind die Ergebnisse dieses Experiments dargestellt. Während der erwartete Anstieg der Iterationen für die beiden Grenzfälle $\epsilon = 10^{-6}$ bzw. $\epsilon = 10^{6}$ bei den Messungen in [19, S. 10, Tab. 2] deutlich – mit 32 bzw. 29 Iterationen, gegenüber 23 für $\epsilon = 1$ – zu sehen ist, ist dieser Effekt bei der Crass-Implementierung nicht zu erkennen. Die Varianz um den Gleichgewichtsfall $\epsilon = 1$ ist hier mit max. 2 Iterationen minimal. Ebenso ist kein genereller Anstieg der Iterationszahlen gegenüber den in Tabelle 5.1 aufgelisteten Zahlen für den quadratischen Einschluss zu vermerken, wie für die PUM-Implementierung der Fall. Dies ist vermutlich, wie

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	8	8	8	8	10	11	11
4	10	10	10	10	12	13	13
5	10	10	11	10	12	13	13
6	11	11	11	10	12	13	13
7	11	11	11	11	12	13	12
8	11	11	11	11	13	13	13
9	11	11	11	11	13	13	14

Tabelle 5.2: Ergebnisse, Problem (2): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für einen sternähnlichen Einschluss, der zu singulärem Charakter der Lösung führt (vgl. Abb. 5.1b).

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	8	8	8	8	9	9	9
4	10	10	10	10	10	12	13
5	10	10	10	10	11	13	13
6	11	11	11	10	11	13	13
7	11	11	11	11	12	13	17
8	11	11	11	11	12	13	17
9	12	11	11	11	13	13	19

Tabelle 5.3: Ergebnisse, Problem (3): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für einen zahnradähnlichen Einschluss sehr komplexer Geometrie (vgl. Abb. 5.1c).

oben bereits erwähnt, auf die genauere Integration und Geometriebehandlung zurückzuführen. Ebenso ist die Auswertung der Distanzfunktionen in der Crass-Implementierung sauberer gelöst, wohingegen bei der Vergleichsimplementierung des PUM-Codes analytische Distanzfunktionen für Quadrate verwendet wurden, die sich im Inneren überlagern. Der Multilevel-Löser bleibt also auch für Probleme mit wachsendem singulären Charakter und nicht konvexen Gebieten robust.

Problem 3: Zahnradähnlicher Einschluss

Um nun zu verifizieren, dass durch die algebraische Verfeinerung mittels passender Anreicherungsfunktionen tatsächlich bessere Groblevel-Approximationen erzielt werden können ohne die Koeffizientensprünge geometrisch aufzulösen, betrachten wir nun den in Abb. 5.1c dargestellten, zahnradähnlichen Einschluss. Dieser besitzt eine sehr komplexe Geometrie, die auf den groben Levels nicht hinreichend genau aufgelöst werden kann. Ebenso weist der Einschluss einige einspringende Ecken auf, was zu Singularitäten in der Lösung führt.
Die Iterationszahlen in Tabelle 5.3 belegen jedoch, dass sich der Multilevel-Löser für dieses Problem hochgradig robust, bezüglich der Höhe des Koeffizientensprungs sowie des Levels, verhält. Lediglich für extrem große Sprünge steigt die Iterationszahl leicht an, so werden für $\epsilon = 10^6$ 19 Iterationen benötigt. Dieser Anstieg ist mit einer stärkeren Gewichtung (10^6)

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	9	9	9	8	9	9	9
4	11	11	11	10	11	12	12
5	12	12	11	10	12	12	12
6	12	12	12	10	13	15	17
7	13	13	12	11	13	15	18
8	13	13	13	11	14	15	18
9	14	14	14	11	14	15	17

Tabelle 5.4: Ergebnisse, Problem (4): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für zwei quadratische Einschlüsse, die sich in einem Punkt berühren (vgl. Abb. 5.1d).

eventueller Integrationsfehler zu erklären. Für alle anderen Werte von ϵ ist das Residuum jedoch in maximal 13 Iterationen hinreichend klein. Besonders auffällig ist hierbei, dass die Gesamtzahl der Iterationen, im Vergleich zu dem sehr viel einfacheren Problem des quadratischen Einschlusses, nicht erwähnenswert ansteigt (vgl. Tab. 5.1). Im Gegensatz hierzu benötigt der Löser des PUM-Codes maximal 32 Iterationen, um das Residuum unter 10^{-10} zu drücken, ist jedoch ebenfalls robust (vgl. [19, S. 12, Tab. 5]).

Problem 4: Zwei quadratische Einschlüsse

Eines der Standardprobleme zur Untersuchung von Lösern bezüglich deren Robustheit, ist das schon in Abschnitt 3.3 behandelte Problem zweier quadratischer Einschlüsse, die sich in einem Punkt berühren (vgl. Abb. 5.1d). Dieser Berührpunkt ist aufgrund seines singulären Charakters und der Überlagerung verschiedener Anreicherungsfunktionen kritisch. Die Ergebnisse aus Tabelle 5.4 für diese Konfiguration verifizieren jedoch die Robustheit des Lösers in diesem Fall. Nach maximal 18 Iterationen terminiert der Löser, und die Abweichung vom Gleichgewichtszustand $\epsilon = 1$ ist moderat. Erneut erzielen wir bessere Ergebnisse als die PUM-Implementierung mit maximal 31 Iterationen für dieses Problem (vgl. [19, S. 11, Tab. 3].
Diese Messwerte dienen als Referenzwerte für eine spätere, detailliertere Betrachtung dieser Konfiguration (siehe Problem 6).

Problem 5: Viele gleichverteilte, zahnradähnliche Einschlüsse

Als letztes Problem dieses vergleichenden Abschnitts wollen wir eine Konfiguration vieler gleichverteilter, zahnradähnlicher Einschlüsse betrachten (vgl. Abb. 5.1e). Für alle Einschlüsse wählen wir den Diffusionskoeffizienten gleich und betrachten somit nur zwei verschiedene Materialien. Außerdem wählen wir die Abstände zwischen den Einschlüssen nicht zu klein, um keine zusätzlichen Effekte zu provozieren. Die sehr komplexe Geometrie kann nun auf groben Levels nahezu nicht aufgelöst werden und wir können so den großen Vorteil der Anreicherungsfunktionen demonstrieren. Die Ergebnisse in Tabelle 5.5 zeigen erneut ein sehr robustes Verhalten des Lösers: Die Zahl der benötigten Iterationen stimmt nahezu mit denen für einen einzelnen zahnradähnlichen Einschluss in Tabelle 5.3 überein. Auf Level 9 ist die

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	1
3	10	10	10	9	10	10	2
4	12	12	12	13	16	14	15
5	16	16	16	16	19	18	23
6	14	14	14	13	16	16	19
7	14	14	14	13	15	16	20
8	13	13	13	13	15	16	19
9	13	13	13	13	14	16	19

Tabelle 5.5: Ergebnisse, Problem (5): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für viele gleichverteilte, zahnradähnliche Einschlüsse (vgl. Abb. 5.1e).

Lösung nach max. 19 Iterationen hinreichend genau berechnet – für $\epsilon < 1$ erhalten wir sogar dieselben optimalen Iterationszahlen wie für $\epsilon = 1$.
Auf Level 5 ist jedoch für alle Werte von ϵ ein sprunghafter Anstieg der Iterationszahlen festzustellen: Von 12 auf 16 für $\epsilon = 10^{-6}$ bzw. von 15 auf 23 für $\epsilon = 10^{6}$. Während die unterschiedliche Sprunghöhe, ebenso wie die generell höheren Iterationszahlen für große ϵ, durch unterschiedliche Gewichtung der Integrationsfehler zu erklären sind, hat der Sprung an sich eine andere Ursache. Alle Einschlüsse außer Ω_3 und Ω_6 haben zu mindestens einem Rand des Gebiets einen sehr kleinen Abstand. Auf Level 5 passt erstmals ein Patch zwischen Einschluss und Gebietsrand und demzufolge liegen Randwert und Wert der Lösung innerhalb des Einschlusses auf verschiedenen Ebenen. In allen vorherigen Levels wird dieser Unterschied nicht aufgelöst und Randwert und Einschluss liegen auf einer Ebene. Diese lösungs-aversen Komponenten der Groblevel-Approximationen verursachen höhere Iterationszahlen auf feineren Levels.
Betrachten wir die Ergebnisse des Lösers im PUM-Code [19, S. 12, Tab. 6], so ist der Crass-Löser mit 13 im Vergleich zu 38 Iterationen ($\epsilon = 10^{-6}$) bzw. 19 zu 29 Iterationen ($\epsilon = 10^{6}$) nicht nur deutlich effizienter, sondern besitzt auch höhere Robustheit bezüglich der Levels (siehe die in [19, S. 13] diskutierten Iterationssprünge).

Wir sehen also, dass die einfachen, gekoppelten Distanzfunktionen 4.2 für eine große Zahl an Problemen ausreichen um hinreichend gute Groblevel-Approximationen zu erhalten und damit einen effizienten Löser ermöglichen. Beim Vergleich mit den Ergebnissen des Lösers im PUM-Code haben wir gesehen, dass eine genauere Integration und Geometriebehandlung das Ergebnis signifikant verbessern. Durch Integrationsfehler verursachtes nicht robustes Verhalten konnte so abgeschwächt werden. Weiterhin scheinen die korrekten Distanzfunktionen wesentlich besser zu funktionieren, als das Maximum bzw. Minimum einer Reihe überlagerter analytischer Distanzfunktionen, wie im PUM-Code verwendet.

5.2 Weiterführende Analyse spezieller Probleme

Im Folgenden Abschnitt werden wir nun speziell besonders kritische Konfigurationen untersuchen. Dabei betrachten wir vor allem kleine Abstände zwischen Einschlüssen, Einschlüsse

verschiedener Größe, besonders kleine Einschlüsse und Einschlüsse, die zu sehr kleinen Schnitten mit den Patches führen. Anhand dieser Konfigurationen und der erkannten Probleme bezüglich der Robustheit des Lösers werden wir die Verwendung anderer Anreicherungsfunktionen motivieren und untersuchen, ob diese zu robusterem Verhalten beitragen. Die verwendeten Anreicherungsfunktionen wurden bereits in Kapitel 4.1 vorgestellt.

Problem 6: Zwei quadratische Einschlüsse mit variablem Abstand

Für Einschlüsse mit beliebig kleinem Abstand zueinander treten ab einer bestimmten Verfeinerungsstufe (Level) zwangsläufig Probleme im Hinblick auf die Robustheit des Multilevels auf. Wir betrachten deshalb in diesem Abschnitt eine Konfiguration zweier quadratischer Einschlüsse $\Omega_1 = (-\frac{1}{3} - a, -a)^2$ und $\Omega_2 = (0, \frac{1}{3})$, deren (achsenparalleler) Abstand $a := d(\Omega_1, \Omega_2)$ variiert wird: $d(\Omega_1, \Omega_2) \in \{4 \cdot 10^{-3}, 8 \cdot 10^{-3}, 2 \cdot 10^{-2}, 2 \cdot 10^{-1}\}$, siehe Abb. 5.2, links. Da in diesem Experiment nur die Robustheit bezüglich des Abstandes der Einschlüsse untersucht werden soll, beschränken wir uns – um unerwünschte Nebeneffekte durch Singularitäten o. ä. möglichst zu vermeiden – wieder auf eine sehr einfache Geometrie, die schon auf groben Levels exakt aufgelöst werden kann. Die auftretenden Effekte lassen sich so besser differenzieren und erklären.
Die Abstände sind so gewählt, dass ein nicht robustes Verhalten des Lösers auf im Voraus bekannten Levels provoziert wird. Dabei spielt die Frage, ab welcher Verfeinerungsstufe erstmals ein Patch in den Raum zwischen den beiden Einschlüssen passt, eine wichtige Rolle, wie wir später noch genauer sehen werden. Die Seitenlänge eines Patches auf Level 9 beträgt in diesem Problem $\approx 3.9 \cdot 10^{-3}$, die auf Level 8 hingegen $\approx 7.8 \cdot 10^{-3}$ – d.h. bei einem Abstand $d(\Omega_1, \Omega_2) = 4.0 \cdot 10^{-3}$ passt auf Verfeinerungsstufe 9 erstmals ein Patch zwischen die beiden Einschlüsse. Analog gilt dies für $d(\Omega_1, \Omega_2) = 8.0 \cdot 10^{-3}$ auf Level 8, für $d(\Omega_1, \Omega_2) = 2.0 \cdot 10^{-2}$ auf Level 7 und für $d(\Omega_1, \Omega_2) = 2.0 \cdot 10^{-1}$ auf Level 4. Wir wollen nun sehen, ob sich das erwartete Verhalten einstellt und der Löser auf den entsprechenden Levels Sprünge in der Anzahl der Iterationen aufweist. Wir verwenden hierzu zunächst die einfachen gekoppelten Distanzfunktionen. Die Konfiguration aus Problem 4 soll als Referenzlösung für die folgenden Messwerte dienen.

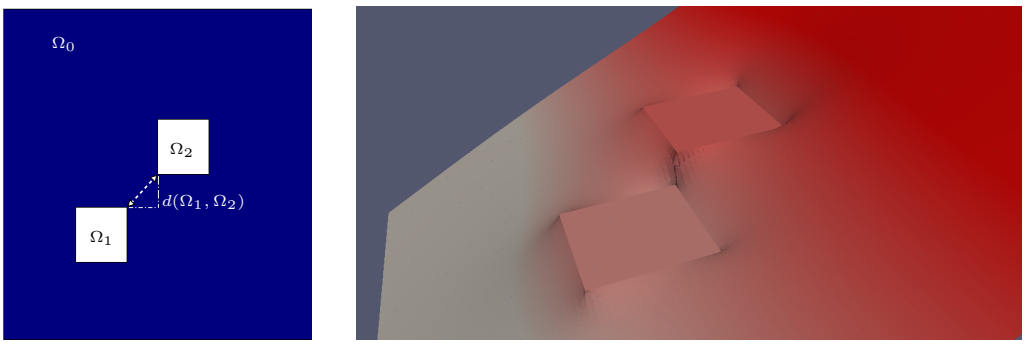


Abbildung 5.2: Zwei quadratische Einschlüsse mit variabler Distanz. Links: Skizze der betrachteten Konfiguration mit $d(\Omega_1, \Omega_2) \in \{4 \cdot 10^{-3}, 8 \cdot 10^{-3}, 2 \cdot 10^{-2}, 2 \cdot 10^{-1}\}$. Rechts: Ausschnitt der Lösung des Randwertproblems für $\epsilon = 10^6$: Die Werte der Lösung in den Einschlüssen liegen auf verschiedenen Ebenen

(a) Ergebnisse, Problem (6.1): Abstand zwischen Einschlüssen beträgt: $d(\Omega_1, \Omega_2) = 4.0 \cdot 10^{-3}$. Nicht robustes Verhalten des Lösers auf Level 9 provoziert.

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	9	9	9	8	9	9	9
4	11	11	11	10	11	12	12
5	12	12	11	10	13	14	16
6	12	12	12	10	14	16	16
7	12	12	12	11	14	16	16
8	13	13	13	11	15	17	17
9	13	13	13	11	15	20	29

(b) Ergebnisse, Problem (6.2): Abstand zwischen Einschlüssen beträgt: $d(\Omega_1, \Omega_2) = 8.0 \cdot 10^{-3}$. Nicht robustes Verhalten des Lösers auf Level 8 provoziert.

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	9	9	9	8	9	9	9
4	11	11	11	10	11	12	12
5	12	12	11	10	13	15	15
6	12	12	12	10	14	15	16
7	12	12	12	11	14	16	16
8	13	13	13	11	15	20	29
9	13	13	13	11	15	20	31

(c) Ergebnisse, Problem (6.3): Abstand zwischen Einschlüssen beträgt: $d(\Omega_1, \Omega_2) = 2.0 \cdot 10^{-2}$. Nicht robustes Verhalten des Lösers auf Level 7 provoziert.

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	9	9	9	8	9	10	10
4	11	11	11	10	12	14	19
5	11	11	11	10	13	14	15
6	12	12	12	10	13	16	17
7	12	12	12	11	14	19	32
8	13	13	12	11	15	20	30
9	13	13	13	11	15	20	31

(d) Ergebnisse, Problem (6.4): Abstand zwischen Einschlüssen beträgt: $d(\Omega_1, \Omega_2) = 2.0 \cdot 10^{-1}$. Nicht robustes Verhalten des Lösers auf Level 4 provoziert.

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	10	10	10	10	12	14	14
4	11	11	11	11	13	18	27
5	12	12	12	12	13	19	28
6	12	12	12	12	14	19	27
7	12	12	12	12	14	20	29
8	12	12	12	12	14	20	29
9	12	12	12	12	14	20	31

Tabelle 5.6: Ergebnisse, Problem (6): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für zwei quadratische Einschlüsse mit versch. Abständen $d(\Omega_1, \Omega_2) \in \{4 \cdot 10^{-3}, 8 \cdot 10^{-3}, 2 \cdot 10^{-2}, 2 \cdot 10^{-1}\}$.

Für alle betrachteten Abstände zeigt sich exakt das erwartete Verhalten: Auf dem entsprechenden Level kommt es für große Koeffizientensprünge $\epsilon \in \{10^4, 10^6\}$ zu einem sprunghaften Anstieg der Iterationszahlen des CG um 12, 13 bzw. 15 Iterationen (vgl. Tabelle 5.6). Abgesehen von diesen Anomalien ist der Löser jedoch bezüglich der Levels robust – einzig auf Level 4 ist für alle Distanzen ein minimaler Anstieg von ca. 4 Iterationen zu verzeichnen. Dieser Anstieg lässt sich durch den Effekt kleiner Schnitte zwischen Patch und Einschluss erklären. Wir wollen nun mögliche Ursachen für den sprunghaften Anstieg genauer untersuchen und erläutern:

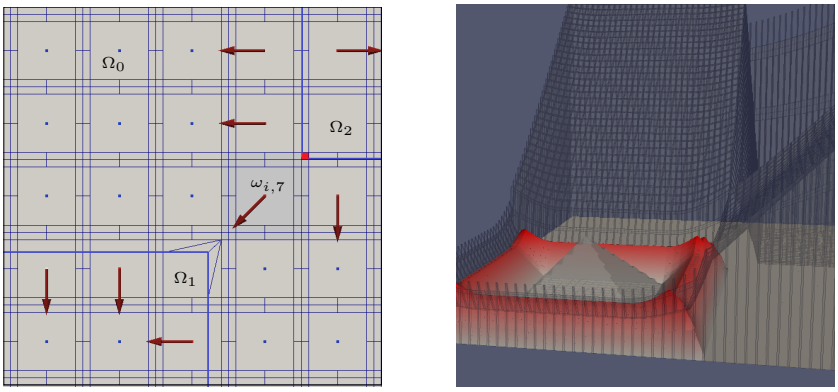
(i) **Offset der Lösung zwischen Einschlüssen:** Wie wir schon in Problem 5 gesehen haben, erzeugen kleine Abstände zu Rändern oder anderen Einschlüssen auf den gröberen Levels schlechte Approximationen der tatsächlichen Lösung, da der kleine Abstand in dieser Verfeinerungsstufe nicht sichtbar ist. Dieser Effekt ist bei den betrachteten Problemen offensichtlich die maßgebliche Ursache für das nicht robuste Verhalten: Bis zu der Verfeinerungsstufe, auf der erstmals ein Patch in den Zwischenraum passt, liegen die Werte innerhalb der beiden Einschlüsse auf einer Ebene. Die tatsächliche Lösung weist jedoch einen deutlichen Offset der Werte der beiden Einschlüsse auf, siehe Abb. (5.2, rechts). Da die Approximationen der gröberen Levels diesen Charakter der Lösung überhaupt nicht erfassen, bricht die Konvergenz des Multilevel-Lösers ein, denn fehlerhafte Komponenten werden auf feinere Levels transportiert, wo sie nur schwer durch den Glätter beseitigt werden können.

(ii) **Kleine Schnitte und Integrationsfehler:** Für alle betrachteten Konfigurationen kommt es zu sehr kleinen Schnitten zwischen Patch und Materialinterface. Die Funktionswerte werden bei der Integration üblicherweise mit der Fläche des Patches gewichtet, der Träger des lokalen Ansatzraumes ist. Korrekterweise müssten jedoch speziell die Funktionswerte der Anreicherungsfunktionen mit der Fläche des Schnittes zwischen Patch und Einschluss gewichtet werden. Diese Schnittoperation ist aber, da oft benötigt, sehr teuer und wurde bisher nicht realisiert. Als Konsequenz kann es zu Problemen wie in Abb. 5.3a dargestellt kommen: Der Patch $\omega_{i,7}$ besitzt einen extrem kleinen Schnitt (rote Fläche) mit dem Einschluss Ω_2 und ist daher mit Distanzfunktionen angereichert. Die Funktionswerte dieser Anreicherungsfunktionen werden nun nicht entsprechend des kleinen Schnitts gewichtet, sondern gehen mit derselben Gewichtung wie die Werte der glatten Ansatzfunktionen in die Integration ein, was zu einem 'Übersteuern' dieser Komponenten führt.
Ein ähnliches Problem entsteht bei kleinen Schnitten im Zusammenhang mit Integrationsfehlern und der Gewichtung mit dem Diffusionskoeffizienten: Durch kleine Schnitte verursachte Integrationsfehler werden mit dem Diffusionskoeffizienten multipliziert und fallen dementsprechend unterschiedlich stark ins Gewicht. So sehen wir bei allen Experimenten teilweise deutlich höhere Iterationszahlen für $\epsilon \gg 1$ im Vergleich zu $\epsilon \ll 1$.

(iii) **Überlagerungen verschiedener Anreicherungsfunktionen:** Auf einigen Levels existiert eine Konfiguration, in der ein Patch beide Einschlüsse schneidet, also mit den entsprechenden Distanzfunktionen zu beiden Interfaces angereichert wird. Offensichtlich stören sich diese Funktionen gegenseitig, da sie außerhalb ihres 'sinvollen Bereichs' Werte ungleich Null annehmen. In dieser Implementierung wird stets die Distanzfunktion, deren Interface näher am Integrationspunkt liegt, bevorzugt.

(iv) **Einfluss von Singularitäten:** Je näher sich die Einschlüsse aneinander annähern, desto mehr Einfluss nimmt die durch die Ecken hervorgerufene Singularität in der Lösung ein. Um dieses Verhalten zu erfassen, muss der Ansatzraum zusätzlich um passende Funktionen erweitert werden. Dass diese Singularität aber für diese Konfiguration keinen allzu großen negativen Effekt hat, zeigen die Iterationszahlen in Tabelle 5.4, für die der Einfluss der Singularität am größten ist.

Ist der Diffusionskoeffizient im Inneren der Einschlüsse sehr klein im Vergleich zum Matrix-Material ($\epsilon < 1$), lebt die Lösung vor allem im Inneren der Einschlüsse (Diffusion im Inneren, nahezu keine Diffusion im Matrix-Material). Hier tritt weder der Effekt des Offsets der Werte zwischen den Einschlüssen auf, noch werden etwaige Integrationsfehler und Probleme durch kleine Schnitte über die Maßen stark gewichtet. Das spiegelt sich direkt in den Messergebnissen aller Konfigurationen wider, siehe Tabelle 5.6: Die Zahl der Iterationen ist für diese Werte von ϵ nahezu konstant, sowohl bzgl. der Levels als auch bzgl. der Höhe des Sprungs, und ist insbesondere nicht höher als für die in 5.1 betrachteten Probleme.



(a) kleiner Schnitt **(b)** PU-Anreicherungsfunktionen

Abbildung 5.3: Links: Sehr kleiner Schnitt zwischen Patch und Einschluss führt durch ungleiche Gewichtung und Integrationsfehler zu Problemen. Rechts: Alle 3 PU-Anreicherungsfunktionen für einen Einschluss. Gut sichtbar: der Übergangsbereich zwischen $\eta^{pu}_{1,1-}$ (rot schattiert) und $\eta^{pu}_{0,1-}$ (transparentes Gitter) um Winkel optimal einstellen zu können.

Ansätze um robusteres Verhalten des Multilevel Lösers zu erhalten

Um die erkannten Probleme und Ursachen für das nicht robuste Verhalten des Lösers zu minimieren, wurden verschiedene Anreicherungsfunktionen motiviert und deren Auswirkungen auf den Löser getestet. Jedoch konnten keine wesentlichen Verbesserungen erreicht werden, da die Funktionen entweder nicht den gewünschten Effekt erzielten oder dominierende, negative Seiteneffekte verursachten.

(1) Entkoppelte Distanzfunktionen:
Da der Offset zwischen den Werten der Lösung innerhalb der Einschlüsse durch die zwei Distanzfunktionen, eine für $\Omega_+ := \Omega_1 \cup \Omega_2$ und eine für $\Omega_- := \Omega_0$, nicht aufgelöst wird und man auf die geometrische Verfeinerung angewiesen ist, war die Hoffnung, den Sprung

in den Iterationszahlen durch entkoppelte Distanzfunktionen (vgl. 4.3) zu verringern. Diese sollten insbesondere für die Lösung innerhalb der Einschlüsse voneinander unabhängige Werte zulassen, um so den Offset schon algebraisch besser aufzulösen.
Die Verwendung der entkoppelten Distanzfunktionen für Ω_+ zeigte jedoch keinerlei Verbesserung gegenüber den gekoppelten Distanzfunktionen. Ein vollständig entkoppelter Ansatz, bei dem zusätzlich die Anreicherungsfunktion für das Matrix-Material in die beiden Distanzfunktionen $\eta_{1^-} = \text{dist}^-(x, \Gamma_M^1)$ und $\eta_{2^-} = \text{dist}^-(x, \Gamma_M^2)$ entkoppelt wird, resultierte sogar in einer fehlerhaften Lösung des Randwertproblems. Dies ist durch die starke Überlagerung und gegenseitige Beeinflussung der beiden Funktionen η_{1^-} und η_{2^-} zu erklären.

(2) blended enrichments:
Die bisher betrachteten Distanzfunktionen weisen auf groben Levels im Inneren der Einschlüsse einen Knick auf (vgl. z. B. Abb. 4.2, links). Für höhere Verfeinerungsstufen ist dieser Knick (im Inneren der Einschlüsse) allerdings keine Komponente der Lösung, d. h. der Einfluss dieses Knicks auf den groben Levels sollte minimiert werden. Aus diesem Grund wurden die sog. *blended enrichments* (siehe Abschnitt 4.1.1) nach [11] implementiert. Diese Funktionen wachsen nur im Inneren eines δ-Schlauchs und werden außerhalb glatt auf 0 bzw. 1 fortgesetzt. Der Knick der Distanzfunktion ist somit schon auf groben Levels – je nach Breite des Schlauchs – im Inneren der Einschlüsse nicht sichtbar.
In Tabelle 5.7 sind Messwerte für den kritischen Fall $\epsilon = 10^6$, exemplarisch für Problem 6.3 (vgl. Tabelle 5.6c), für eine Familie dieser Anreicherungsfunktionen gegeben.
Wir sehen, dass die Iterationszahlen für geeignete Parameter leicht besser, jedoch im Wesentlichen gleich sind, wie die für die Verwendung einfacher Distanzfunktionen in Tabelle 5.6c. Die dort angesprochenen Probleme auf Level 4 bzw. Level 7 bleiben nahezu unverändert erhalten. Um den Einfluss des Knicks auf den groben Levels effizient zu minimieren, sollte der Parameter δ^{blend} sehr klein gewählt werden. Gleichzeitig hat dieser jedoch direkten Einfluss auf den Betrag der Gradienten. Eine zu kleine Wahl wie $\delta^{blend} = 0.002$, $\delta^{blend} = 0.01$ und $\delta^{blend} = 0.02$ führten bei diesem Problem zu Gradienten vom Betrag 170 und höher, was durch Übersteuern dieser Werte zu falschen Lösungen führte. Die in Tabelle 5.7 angegebenen Kombinationen der Parameter erzielten allesamt korrekte Lösungen, wobei tendenziell für glattere Anreicherungsfunktionen mit moderaten Gradienten die besten Ergebnisse erreicht werden konnten.

	$\delta = 0.12$								$\delta = 0.0875$			$\delta = 0.2$
Level	$\gamma = 0.3$	$\gamma = 0.5$	$\gamma = 0.7$	$\gamma = 1.0$	$\gamma = 1.3$	$\gamma = 1.7$	$\gamma = 2.0$	$\gamma = 4.0$	$\gamma = 1.0$	$\gamma = 1.3$	$\gamma = 2.0$	$\gamma = 2.0$
2	2	2	2	2	2	2	3	2	2	2	2	2
3	12	10	11	10	11	11	11	13	10	11	12	10
4	17	18	18	19	19	18	17	19	20	17	18	18
5	17	16	15	16	15	15	15	17	22	16	16	15
6	18	18	17	17	17	16	16	16	18	16	16	16
7	31	32	32	33	30	30	30	30	33	30	30	30
8	33	30	30	31	30	28	29	29	32	30	29	29

Tabelle 5.7: Ergebinsse, Problem (6.3) mit blended enrichments: Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für das Problem zweier quadratischer Einschlüse mit Abstand $d(\Omega_1, \Omega_2) = 2.0 \cdot 10^{-2}$ unter Verwendung der 'blended enrichments' (4.4). Betrachtet wurde der problematisch Fall $\epsilon = 10^6$ für verschiedene Parameter γ^{blend} und δ^{blend}.

Als Fazit können wir festhalten, dass die blended enrichments für sinnvolle Wahl der Parameter zwar etwas bessere Ergebnisse liefern als die einfachen Distanzfunktionen, jedoch auch in ihrer Auswertung deutlich höhere Kosten aufweisen. Weiterhin sind die Funktionen sehr anfällig, durch ungeschickte Wahl von δ^{blend} bzw. γ^{blend} falsche Lösungen zu provozieren und es ist damit nicht einfach, passende Funktionen für beliebige Probleme zu finden. Die Tatsache, dass der Löser mit diesen Funktionen in einigen Fällen um 2-3 Iterationen besser ist, das nicht robuste Verhalten desselben jedoch nicht verbessert, zeigt: Der durch den Knick der Distanzfunktionen hervorgerufene Fehler scheint nahezu keinen Einfluss auf die Robustheit des Lösers für diese Konfiguration zu haben. Trotz der minimalen Verbesserung, stellen die blended enrichments keine bessere Alternative zu den gewöhnlichen Distanzfunktionen dar, hauptsächlich wegen ihres höheren Rechenaufwands und der sensiblen Abhängigkeit von den Parametern.

(3) Partition of Unity Anreicherungsfunktionen:
Die in dieser Arbeit entwickelten *Partition of Unity-Anreicherungsfunktionen* sollten einige der angesprochenen Probleme durch einen wesentlich komplexeren Ansatz und mehr Flexibilität angehen mit der Hoffnung, einen deutlich robusteren Löser zu erhalten. Details zu Konstruktion und Aussehen dieser Anreicherungsfunktionen haben wir bereits in Abschnitt 4.1.1 vorgestellt. Die auf einer (simplen) Partition der Eins basierenden Funktionen sind so gebaut, dass sie eine automatische Wahl des optimalen Funktionswertes im Bereich des Überlapps der Träger der PU-Funktionen zulassen – vergleiche hierzu die in Abb. 5.3b dargestellten Anreicherungsfunktionen für einen Einschluss: Mithilfe des Übergangsbereichs zwischen $\eta^{pu}_{1,1^-}$ (rot schattiert) und $\eta^{pu}_{0,1^-}$ (Gitterfunktion) lassen sich beliebige Winkel einstellen. Die Anreicherungsfunktionen verschiedener Einschlüsse sind völlig unabhängig voneinander (sowohl im Inneren als auch außerhalb der Einschlüsse), was maximale Flexibilität bezüglich unterschiedlicher Offsets oder Winkel an Knicken in der Lösung ermöglichen soll – insbesondere wird hierdurch die Approximationsqualität der groben Levels erhöht.
Numerische Experimente mit diesen Anreicherungsfunktionen ergaben für nur einen Einschluss gute Werte. So konnte beispielsweise die Konfiguration aus Problem 1 und der komplizierte zahnradähnliche Einschluss aus Problem 3 für $\epsilon = 10^6$ in maximal 12 bzw. 14 Iterationen gelöst werden. Für den eigentlich interessanten Anwendungsfall von 2 oder mehr Einschlüssen, für den diese Anreicherungsfunktionen entwickelt wurden, ergeben sich jedoch andere Resultate: Zum einen sind die PU-Anreicherungsfunktionen für Einschlüsse mit sehr kleinem Abstand unbrauchbar, da die einfache Konstruktion der Partition der Eins disjunkte Träger ω_i^{enr} für die Einschlüsse $j = 1, \cdots, m$ verlangt und lediglich paarweisen Überlapp $\omega_j^{enr} \cap \omega_0^{enr} \neq \emptyset$, $j = 1, \cdots, m$ zulässt. Kleine Abstände erzwingen einerseits einen sehr kleinen Überlapp δ^{pu}, was wiederum große Gradienten provoziert, andererseits wird der 'Flat-Top-Bereich' $\mathcal{Q}_k$ der PU-Funktionen auf ein Minimum reduziert. Die Funktionen η^{pu}_{k,k^+} codieren dann kaum noch Information, da sie fast konstant Null sind. Abhilfe könnte hier jedoch eine komplexere PU-Konstruktion schaffen, die alle benachbarten Patches ω_j^{enr} berücksichtigt. Dennoch scheinen die Funktionen in der bisherigen Form i. A. für mehr als einen Einschluss zu Problemen zu führen. Experimente mit der in Abb. (4.2, rechts) dargestellten Konfiguration ergaben ebenfalls ein sehr schlechtes Lösungsverhalten: Mit bestenfalls 50, i. A. jedoch wesentlich mehr Iterationen konnte das Residuum auf einen hinreichend kleinen Wert reduziert werden. Sowohl die prolongierte Lösung als auch der restringierte Defekt, zeigen erhebliche Fehlerkomponenten

im Bereich der angereicherten Patches. Aufgrund dieser hoch oszillierenden Fehlerkomponenten mit Absolutbeträgen von einigen Zehnerpotenzen ist die Konvergenz des Multilevel-Schemas unmöglich. Als potentielle Ursache sehen wir ein Quadraturproblem im Zusammenhang mit den PU-Anreicherungsfunktionen. Bei der Integration dieser Funktionen wird das Gebiet nicht wie für die PU-Funktionen φ_i des PUM-Ansatzraumes so in Integrationszellen zerlegt, dass die zu integrierenden Funktionen stückweise polynomiell sind. So verursachte Quadraturfehler könnten Auslöser für das schlechte Verhalten der Anreicherungsfunktionen sein. Eine zusätzliche Unterteilung in Integrationszellen entsprechend der φ_k^{enr}-Funktionen würde dieses Problem entschärfen.
Eine weitere mögliche Ursache liegt in der durch die Anreicherungsfunktionen induzierte, schlechtere Fehlerreduktion des Glätters. Stärkere Kopplungen zwischen den Funktionen des Anreicherungsraums und des polynomiellen Ansatzraums mindern die Qualität des Glätters u. U erheblich. In diesem Fall müssten gezielt effizientere Glätter – angepasst auf die jeweiligen Anreicherungsfunktionen – entwickelt werden.
Um die Intensität des Einflusses der beschriebenen Effekte besser einschätzen zu können und eventuelle zusätzliche Seiteneffekte abzugrenzen, modifizieren wir die in Abb. 4.2 dargestellte Konfiguration dahingehend, dass wir den Durchmesser der Einschlüsse verkleinern $\Omega_1 = (-0.65, -0.35)^2$ bzw. $\Omega_2 = (0.35, 0.65)^2$ und so einen größeren Abstand zum Rand erreichen. Der Löser weist dann unter Verwendung der PU-Anreicherungsfunktionen mit $\delta^{PU} = 0.12$ und $d_{\mathcal{Q}} = 0.425$ ein durchaus gutes Ergebnis bzgl. der benötigten Iterationen auf, das jedoch trotz allem hinter der Performanz desselben bei Anreicherung mit einfachen Distanzfunktionen zurückbleibt (siehe Tabelle 5.8). Für die nicht aufgeführten Iterationszahlen für die Fälle mit $\epsilon < 1$ sind beide Varianten identisch und benötigen maximal 12 Iterationen. Wir sehen jetzt für die PU-Anreicherungsfunktionen ein weitaus stabileres Verhalten des Lösers. Trotz allem reagiert der Löser sehr viel sensibler auf diese Anreicherungsfunktionen und ungeschickte Kombinationen führen zu einem starken Abfall der Konvergenz. Das oben erwähnte schlechte Lösungsverhalten ist vermutlich in erster Linie auf Probleme mit fehlerhaften Funktionsauswertungen auf dem Rand und anderen Artefakten, die durch die nicht ausgereifte Geometriebehandlung entstehen, zurückzuführen. Trotzdem ist die Performanz der PU-Funktionen gegenüber den einfachen Distanzfunktionen – bei gleicher Approximationsqualität – schlechter. Hierfür machen wir oben beschriebene Integrationsprobleme und schlechtere Glättungseigenschaften verantwortlich.

	einfache Distanzfunktionen (gekoppelt)				PU-Anreicherungsfunktion			
Level	$\epsilon = 10^0$	$\epsilon = 10^2$	$\epsilon = 10^4$	$\epsilon = 10^6$	$\epsilon = 10^0$	$\epsilon = 10^2$	$\epsilon = 10^4$	$\epsilon = 10^6$
2	2	2	2	2	2	2	2	2
3	10	12	14	14	11	12	17	21
4	11	12	13	13	11	12	15	15
5	12	13	14	14	11	13	15	15
6	12	13	14	14	11	13	16	16
7	12	13	14	14	11	14	16	16
8	12	13	14	14	12	14	16	16

Tabelle 5.8: Vergleich PU-Anreicherungsfunktionen mit Distanzfunktionen für zwei quadratische Einschlüsse: Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für zwei quadratische Einschlüsse. Links unter Verwendung einfacher Distanzfunktionen als Anreicherungsfunktionen, rechts mit PU-Anreicherungsfunktionen.

Manipulation von Überlapp und Glättungsschritten für Problem 6

Nachdem wir Approximationsqualität, Verhalten und Auswirkungen verschiedener Anreicherungsfunktionen auf den Multilevel-Löser untersucht haben, ohne dessen Robustheit merklich zu verbessern, kehren wir nun zu den einfachen, gekoppelten Distanzfunktionen zurück. Im Hinblick auf die Effizienz und Robustheit des Lösers wollen wir nun zwei Parameter – den Patchüberlapp α sowie die Anzahl der Vor- bzw. Nachglättungsschritte ν^{pre}, ν^{post} – variieren. Wie bereits erwähnt, kontrolliert der Patchüberlapp die Steilheit der Gradienten der PU-Funktionen, d.h. ein größerer Überlapp sollte das Problem tendenziell gutmütiger machen. Gleichzeitig werden jedoch die Abhängigkeiten durch größere Nachbarschaften erhöht – die Steifigkeitsmatrix enthält dann eine wachsende Zahl an Nicht-Null-Einträgen, was den Rechenaufwand in die Höhe treibt. Ebenso bedeuten mehr Glättungsschritte eine höhere Laufzeit für den Multilevel-Löser.

Level	$\epsilon = 10^{-6}$			$\epsilon = 10^{-4}$			$\epsilon = 10^{-2}$			$\epsilon = 10^{0}$			$\epsilon = 10^{2}$			$\epsilon = 10^{4}$			$\epsilon = 10^{6}$		
$\alpha =$	1.2	1.5	1.8	1.2	1.5	1.8	1.2	1.5	1.8	1.2	1.5	1.8	1.2	1.5	1.8	1.2	1.5	1.8	1.2	1.5	1.8
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	9	7	13	9	7	13	9	7	13	8	6	14	9	8	17	9	9	20	9	9	24
4	11	7	12	11	7	12	11	7	12	10	6	12	11	8	15	12	11	23	12	24	35
5	12	7	13	12	7	13	11	7	13	10	6	12	13	8	14	15	9	17	15	10	33
6	12	7	13	12	7	13	12	7	13	10	6	12	14	9	15	16	13	19	16	13	24
7	12	7	13	12	7	13	12	7	13	11	7	13	14	9	16	16	12	23	16	17	27
8	13	8	13	13	8	13	13	7	13	11	7	13	15	9	17	20	16	29	20	25	43
9	13	8	14	13	8	14	13	7	14	11	7	13	15	9	17	20	17	28	20	25	41

Tabelle 5.9: Ergebinse, Problem (6.2) mit verschiedenen Werten für den Patchüberlapp α: Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für das Problem zweier quadratischer Einschlüsse mit Abstand $d(\Omega_1, \Omega_2) = 8.0 \cdot 10^{-3}$ für $\alpha \in \{1.2, 1.5, 1.8\}$.

Wir betrachten nun das Problem 6.2 zweier quadratischer Einschlüsse mit Abstand $d(\Omega_1, \Omega_2) = 8.0 \cdot 10^{-3}$ bei dem der Löser für großes ϵ nicht robustes Verhalten auf Level 8 zeigt. In Tabelle 5.9 sind die Iteratationszahlen des CG-Lösers für unterschiedlich großen Stretchfaktor $\alpha \in \{1.2, 1.5, 1.8\}$ gegeben. Wir sehen zunächst, dass sich für $\alpha = 1.5$ (mittlere Spalte) vor allem für $\epsilon \leq 1$ hervorragende Resultate ergeben. Dies ist zum einen durch die betragsmäßig kleineren Gradienten und den höheren Informationsgehalt durch stärkere Abhängigkeiten zu erklären. Zum anderen wird das Gebiet entsprechend der Schnittstellen der Patches in Integrationszellen unterteilt. Durch einen Überlapp der halben Patchbreite entstehen mehr dieser Schnittstellen und die Integrationszellen sind homogen verteilt, was vor allem auf gröberen Levels zu besseren Approximationen führt. Für sehr große ϵ treten jedoch auch für $\alpha = 1.5$ Iterationssprünge auf. Während der durch den Offset der Werte in den Einschlüssen produzierte Sprung auf Level 8 erhalten bleibt, wird der vorher kleine Integrationssprung auf Level 4 enorm verstärkt. Der Anstieg von 9 auf 24 Iterationen für $\epsilon = 10^6$ ist offensichtlich durch extrem kleine Schnitte und der entsprechenden Gewichtung der entstehenden Integrationsfehler auf diesem Level hervorgerufen. Dies sieht man sehr deutlich auch an der starken Abhängigkeit vom Betrag von ϵ: Vergleiche beispielsweise den unbedeutenden Sprung von 9 auf 11 Iterationen für $\epsilon = 10^4$.

Man kann hier sehr deutlich sehen, wie groß der Einfluss dieses Effekts auf die Robustheit des Lösers sein kann. Abgesehen von dem Fall $\epsilon = 10^6$ erhalten wir jedoch für $\alpha = 1.5$ die besten Ergebnisse.

Ganz anders verhält es sich für den sehr großen Überlapp mit Stretchfaktor $\alpha = 1.8$ (vgl. rechte Spalte). Während der Löser für kleine Werte von ϵ noch ein robustes Verhalten zeigt – wenn auch leicht schlechter als der Referenzwert $\alpha = 1.2$ – so sind die Iterationszahlen für sehr großes ϵ sprunghaft und im Allgemeinen sehr hoch. Die Effekte der kleinen Schnitte und Interferenzen von Anreicherungsfunktionen sind bei diesem Grad an Überdeckung kaum noch zu kontrollieren und verursachen unerwünschte, teils große Fehler, die zu einem drastischen Abfall der Konvergenz des Multilevels führen.
Sehr deutlich sieht man das Verhalten des Multilevels für die verschiedenen Konfigurationen, wenn man statt des PCG-Verfahrens ein Richardsonverfahren mit dem $V(1,1)$ ML-Schema als Vorkonditionierer anwendet. Die Iterationszahlen in Tabelle 5.10 bestätigen die obigen Aussagen und zeigen sehr deutlich die problematischen Stellen auf. Das Multilevel arbeitet für kleine ϵ und einem Stretchfaktor $\alpha = 1.5$ sogar als selbstständiger Löser optimal und besitzt hervorragende Konvergenzeigenschaften. Wir können nun also die Fälle für $\epsilon \ll 1$ endgültig als robust bezeichnen. Ebenso sind die Problemstellen für große ϵ auf Level 4 bzw. 8 sehr deutlich zu erkennen, was unseren Verdacht bekräftigt, dass die Gewichtung der Integrationsfehler durch den Diffusionskoeffizienten eine große Rolle spielt. Auf Level 8 erzielt errechnet die Richrdsoniteration die Lösung nicht mehr in einer akzeptablen Anzahl von Iterationsschritten. Der Eintrag $\gg$ bedeutet hier, dass das Multilevelschema nicht mehr konvergiert oder zumindest wesentlich mehr als 150 Iterationen benötigt. Für $\alpha = 1.8$ ergibt sich dieses Bild für alle Koeffizientensprünge.

Level	$\epsilon = 10^{-6}$		$\epsilon = 10^{-4}$		$\epsilon = 10^{-2}$		$\epsilon = 10^{0}$		$\epsilon = 10^{2}$		$\epsilon = 10^{4}$		$\epsilon = 10^{6}$	
$\alpha =$	1.2	1.5	1.2	1.5	1.2	1.5	1.2	1.5	1.2	1.5	1.2	1.5	1.2	1.5
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	14	8	14	8	14	8	10	7	12	19	11	11	11	11
4	17	8	17	8	17	8	14	6	19	11	21	16	21	141
5	20	9	20	9	20	9	17	7	52	18	150	24	150	24
6	22	10	22	10	22	9	19	7	50	39	147	134	150	137
7	23	10	23	10	23	9	21	7	53	39	150	150	$\gg$	$\gg$
8	23	9	23	9	23	9	22	7	77	54	$\gg$	$\gg$	$\gg$	$\gg$
9	23	9	23	9	23	9	22	7	71	48	$\gg$	$\gg$	$\gg$	$\gg$

Tabelle 5.10: Ergebinsse, Problem (6.2) mit verschiedenen Werten für den Patchüberlapp α: Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter Richardson-Iterationen (ML als standalone Löser) für Konfiguration analog zu Tabelle 5.9 für $\alpha \in \{1.2, 1.5\}$.

Um nun die Glättungseigenschaft und die Verbesserung der Konvergenz des Multilevels durch Erhöhung der Glättungsschritte zu untersuchen, betrachten wir nun die ML-Schemata $V(3,3)$ und $V(5,5)$ für obiges Problem 6.2. Zum Vergleich wird das bereits in Tabelle 5.6b untersuchte $V(1,1)$ ML-Schema nocheinmal in Tabelle 5.11 aufgeführt. Wir sehen, dass eine Erhöhung der Vor- bzw. Nachglättungsschritte von 1 auf 3 die Zahl der benötigten Iterationen in etwa halbiert. Der damit wachsende Rechenaufwand (pro Iteration etwa $2 \cdot 3 \cdot \mathcal{O}(N_k^{dof})$) ist angesichts

der erheblichen Verbesserung der Konvergenzgeschwindigkeit vertretbar. Das nicht robuste Verhalten des Lösers auf den Levels 4 und 8 ist jedoch nach wie vor zu sehen und kann durch eine bessere Fehlerkorrektur im Glättungsschritt nicht beseitigt werden. Für das $V(5,5)$ ML-Schema ergeben sich gegenüber dem $V(3,3)$ Schema kaum Verbesserungen und eine weitere Glättung ist hier nicht rentabel.
Für die Richardsoniteration ergibt sich dasselbe Bild wie bei den Experimenten mit variablem Überlapp in Tabelle 5.10: Die Konfigurationen mit $\epsilon \leq 1$ liefern sowohl für die CG-Iteration als auch für die Richardsoniteration nahezu identische Werte. Für größere Werte von ϵ verzeichnen wir jedoch analog zu Tabelle 5.10 einen starken Abfall der Konvergenz des Multilevels, der auch durch eine Erhöhung der Glättungsschritte nicht verhindert werden kann.

	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
Level	$V_{1,1}$ $V_{3,3}$ $V_{5,5}$	$V_{1,1}$ $V_{3,3}$ $V_{5,5}$	$V_{1,1}$ $V_{3,3}$ $V_{5,5}$	V_1^1 V_3^3 V_5^5	$V_{1,1}$ $V_{3,3}$ $V_{5,5}$	$V_{1,1}$ $V_{3,3}$ $V_{5,5}$	$V_{1,1}$ $V_{3,3}$ $V_{5,5}$
2	2 2 2	2 2 2	2 2 2	2 2 2	2 2 2	2 2 2	2 2 2
3	9 6 5	9 6 5	9 5 5	8 5 4	9 6 5	9 5 5	9 5 5
4	11 6 5	11 6 5	11 6 5	10 5 4	11 7 6	12 8 7	12 9 8
5	12 7 6	12 7 6	11 7 6	10 6 5	13 8 7	15 9 8	15 9 8
6	12 7 6	12 7 6	12 7 6	10 6 5	14 8 7	16 9 8	16 9 8
7	12 7 6	12 7 6	12 7 6	11 6 5	14 8 7	16 9 8	16 9 8
8	13 7 6	13 7 6	13 7 6	11 6 5	15 8 7	20 11 10	20 14 11
9	13 7 6	13 7 6	13 7 6	11 6 5	15 9 7	20 11 10	20 14 12

Tabelle 5.11: Ergebinse, Problem (6.2) mit verschiedener Anzahl an Vor- bzw. Nachglättungsschritten für das Multilevel-Schema: Anzahl der mit $V(1,1)$, $V(3,3)$ bzw $V(5,5)$ ML-Schema vorkonditionierter CG-Iterationen für das Problem zweier quadratischer Einschlüsse mit Abstand $d(\Omega_1, \Omega_2) = 8.0 \cdot 10^{-3}$.

Problem 7: Viele zahnradähnliche Einschlüsse unterschiedlicher Größe

Die bisher betrachteten Materialeinschlüsse waren jeweils von gleicher Größe und Form und hatten dieselben Eigenschaften (Diffusionskoeffizient). Wir wollen nun die Größe der Einschlüsse innerhalb einer Konfiguration variieren und die Reaktion des Multilevel-Lösers untersuchen. Wir betrachten hierzu das in Abb. 5.4a dargestellte Problem 8 zahnradähnlicher Einschlüsse unter Verwendung einfacher Distanzfunktionen als Anreicherungsfunktionen und wollen die gewonnenen Ergebnisse mit dem Referenzproblem 6.5 (vgl Abb. 5.1e) vergleichen. Die Vermutung ist, dass das Problem der Einschlüsse gleicher Größe einfacher ist und besseres Lösungsverhalten aufweist, da die lokalen Lösungseigenschaften (in einer Umgebung des Einschlusses) von gleicher Struktur sind. Die Grobgitterlösungen propagieren dann weniger fehlerhafte bzw. lösungsfremde Komponenten auf feinere Levels.

Die Lösung des betrachteten Randwertproblems auf Level 8 ist für die beiden Grenzfälle $\epsilon = 10^{-6}$ bzw. $\epsilon = 10^{6}$ in Abb. 5.4b und 5.4c gegeben. Vergleichen wir nun die Iterationszahlen in Tabelle 5.12 mit den Iterationszahlen der Konfiguration für gleichmäßig verteilte Einschlüsse gleicher Größe, siehe Tabelle 5.5, stellen wir fest, dass sich der Löser für das hier betrachtete Problem bei weitem nicht so robust verhält. Betrachten wir speziell Fälle für großes ϵ, erkennen wir einen erheblichen Anstieg der Iterationszahlen von Level 3 auf Level 4 um 23 Iterationen

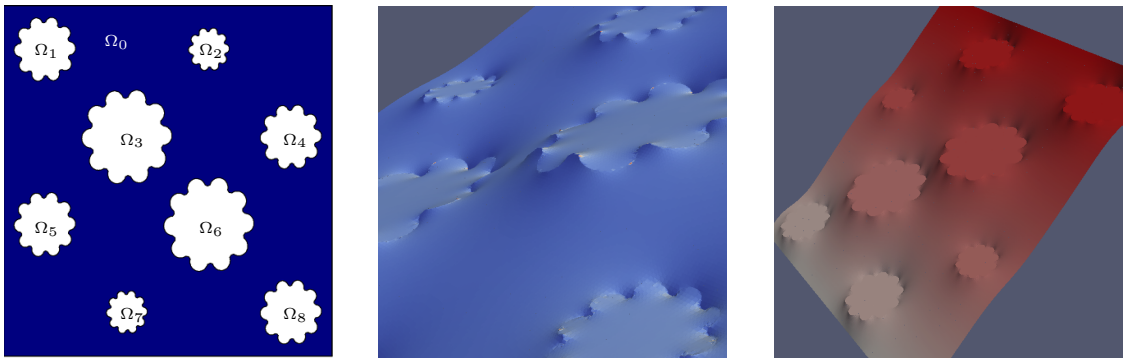


Abbildung 5.4: Viele gleichverteilte zahnradähnliche Einschlüsse unterschiedlicher Größe. Links: (a) Problemkonfiguration. Mitte: (b) Ausschnitt der Lösung des Randwertproblems auf Level 8 für $\epsilon = 10^{-6}$. Rechts: (c) Lösung des Randwertproblems auf Level 8 für $\epsilon = 10^{6}$.

auf einen Wert von 38 – zum Vergleich: Auf Level 5 wird dieses Level mit nur 15 Iterationen gelöst.
Die Ursache für dieses nicht robuste Verhalten ist ähnlich gelagert wie in Problem 6 bei der Untersuchung verschiedener Abstände. Durch die größeren Materialeinschlüsse Ω_3 und Ω_6 ist der entstehende Offset der Werte in der (exakten) Lösung nicht mehr wie in Problem 5 auf dem gröbsten Level aufgelöst sondern wird erst auf Level 4 erkannt. Der starke Anstieg der Iterationszahlen kann daher mithilfe der Erklärungen zu Problem 6 direkt anhand der Groblevel-Lösungen auf Level 3 (Abb. 5.5a) bzw. Level 4 (Abb: 5.5b) abgelesen werden.

Level	$\epsilon = 10^{-6}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-2}$	$\epsilon = 10^{0}$	$\epsilon = 10^{2}$	$\epsilon = 10^{4}$	$\epsilon = 10^{6}$
2	2	2	2	2	2	2	2
3	11	11	11	10	12	14	14
4	12	12	12	13	20	27	38
5	16	16	16	15	21	27	37
6	14	14	14	14	18	25	33
7	14	14	14	13	17	23	33
8	13	13	13	13	41	23	31
9	13	13	13	13	13	24	44

Tabelle 5.12: Ergebnisse, Problem (7): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für viele gleichverteilte, zahnradähnliche Einschlüsse unterschiedlicher Größe (vgl. Abb. 5.4a).

Ebenfalls großen Einfluss besitzen hier die schon oft erwähnten kleinen Schnitte. Die hier neu eingeführten kleineren Einschlüsse Ω_2 und Ω_7 mit Durchmesser $d = 0.2$ produzieren auf Level 4 extrem kleine Schnitte mit Patches, die fast komplett im Matrix-Material liegen, siehe Abb. 5.5c. Der Flächeninhalt des Schnitts macht jeweils nur etwa $1/400$ oder weniger des Gesamtflächeninhalts des Patches aus – jedoch wird der Patch trotz allem mit Distanzfunktionen für den Einschluss angereichert, die dieselbe Gewichtung erhalten wie die übrigen Funktionswerte. Durch den wesentlich höheren Diffusionskoeffizient $\epsilon = 10^{6}$ im Inneren der Einschlüsse werden die Integrationspunkte hier zusätzlich mít einem ungleich höheren Gewicht versehen und

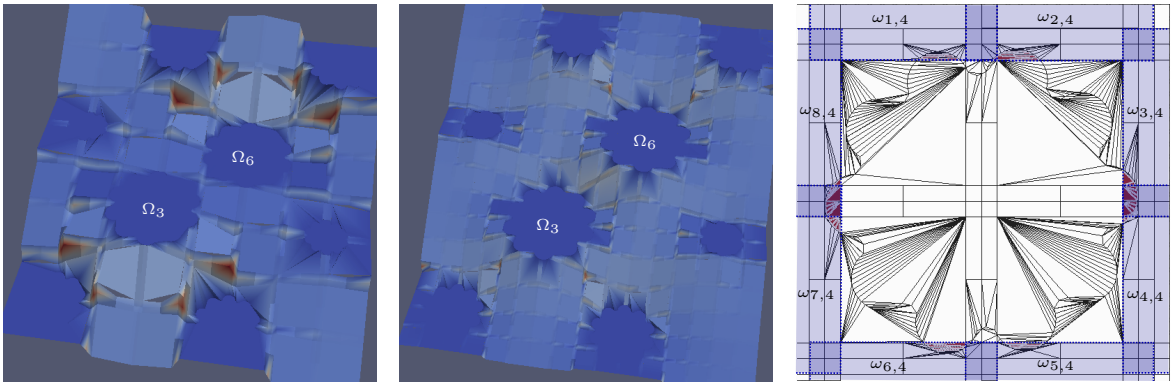


Abbildung 5.5: Lösungen von Problem 7 (vgl Abb. 5.4a) auf verschiedenen (groben) Levels. (a) Groblevel-Lösung für $\epsilon = 10^6$ auf Level 3, Farbe nach Betrag des Gradienten (b) Groblevel-Lösung für $\epsilon = 10^6$ auf Level 4, Farbe nach Betrag des Gradienten. (c) Sehr kleine Schnitte mit Einschluss auf Level 4

dominieren somit. Beim Referenzproblem 5 sind ähnliche Effekte auf Level 5 zu sehen, was den Sprung von 15 auf 23 Iterationen, neben der kleinen Distanz zum Rand, verursacht.

Um diesen Effekt etwas zu isolieren wurde Problem 7 leicht modifiziert, sodass der Abstand zwischen allen Einschlüssen schon auf Level 2 aufgelöst werden kann. Zusätzlich wurden die Positionen der Einschlüsse minimal perturbiert. Wenngleich das Resultat nicht wie erhofft bessere Iterationszahlen aufzeigt, sehen wir jedoch einmal mehr den starken Einfluss der kleinen Schnitte. Wegen der leichten Verschiebung von Ω_4 entsteht ein kleiner Schnitt mit dem Patch in der linken oberen Ecke (vgl. Abb. 5.6). Der Funktionswert der Anreicherungsfunktion wird infolgedessen zu stark gewichtet und führt zu der in 5.6 abgebildeten, sehr schlechten Groblevel-Lösung. Zum Vergleich ist die Groblevel-Lösung ohne diesen kleinen Schnitt in rot ebenfalls dargestellt. Die schlechte Groblevel-Lösung verursacht einen Iterationssprung von 30 Iterationen zwischen Level 2 und Level 3 für $\epsilon = 10^6$ und die Konvergenz des Lösers ist in der perturbierten Konfiguration insgesamt schlechter.

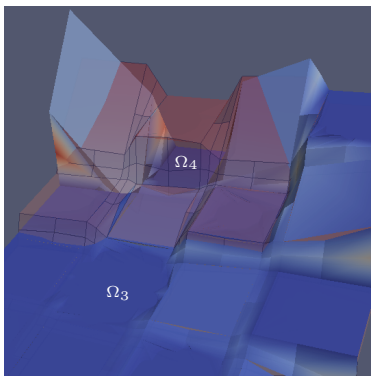


Abbildung 5.6: Schlechte Groblevel-Lösung, verursacht durch kleinen Schnitt

Aufgrund der komplizierten Geometrie und der nicht vollständig ausgereiften Geometriebehandlung in der aktuellen Version des Crass Codes kommt es vor allem auf Level 9 zu erheblichen Fehlern bei der Triangulierung der Geometrie. Der hohe Wert von 44 Iterationen ($\epsilon = 10^6$) auf dem feinsten Level ist mit großer Sicherheit durch Integrationsfehler aufgrund fehlerhafter Triangulierung verursacht und damit ein vorübergehendes technisches Problem. Versuche, ein besseres Konvergenzverhalten des Lösers durch entkoppelte Distanzfunktionen sowie blended enrichments zu erhalten, brachten nicht das gewünschte Ergebnis. Eine Erhöhung der Vor- bzw. Nachglättungsschritte auf 3 produzierte für $\epsilon = 10^6$ nahezu dieselben Werte wie für den Fall gleich großer Einschlüsse (Problem 5). Hochgradig nicht robustes Verhalten bzw. Iterationszahlen von 150 und größer zeigte der Löser in diesem Beispiel für

einen größeren Überlapp $\alpha \in \{1.5, 1.8\}$. Wir erklären diesen Konvergenzabfall durch höhere Interferenz bzw. große gegenseitige Störung der einzelnen Anreicherungsfunktionen.
Trotz der unterschiedlichen Größe der Einschlüsse erhalten wir jedoch für die Fälle der Diffusion im Inneren der Einschlüsse ($\epsilon \leq 1$) exakt dieselben Resultate, der Löser ist also für $\epsilon \leq 1$ unabhängig von Größe, Abstand und Geometrie der Einschlüsse robust. Der schon zuvor beobachtete kleine Anstieg in den Iterationszahlen von Level 4 auf Level 5 ist durch eine schlechte Grobgitterlösung auf Level 4 innerhalb der Einschlüsse zurückzuführen.

Problem 8: Unterschiedliche Sprunghöhen an Materialinterfaces

Wir wollen nun für den einfachen Fall zweier quadratischer Einschlüsse eine Koeffizientenfunktion mit unterschiedlichen Sprunghöhen an den Interfaces Γ_M^1 bzw. Γ_M^2 untersuchen, d. h. wir betrachten zwei Materialien mit unterschiedlichen Diffusionseigenschaften. Für die Konfiguration aus Problem 6.2 (vgl. Abb. 5.1d) fixieren wir $\kappa|_{\Omega_1} = 10^6$ und variieren den Diffusionskoeffizienten $\epsilon_{\Omega_2} \in \{10^{-6}, 10^{-4}, 10^{-2}, 1, 10^2, 10^4, 10^6\}$ wie zuvor für Ω_2. Die in Tabelle 5.13 dargestellten Resultate zeigen ein durchaus robustes Verhalten des Lösers – für große ϵ gewinnen jedoch wieder die in Problem 6.2 (siehe Tabelle 5.6b) diskutierten Effekte, wie der Offset zwischen den Einschlüssen ab Level 8, an Einfluss. Für $\epsilon < 1$ verzeichnen wir zwar einen Anstieg der Iterationszahlen von 16 auf 20, dieser ist jedoch unabhängig von der Sprunghöhe. Fixieren wir umgekehrt $\kappa|_{\Omega_1} = 10^{-6}$ ist die Lösung des Problems deutlich einfacher und die Resultate für $\epsilon \leq 1$ unterscheiden sich unwesentlich von denen in Tabelle 5.6b. Die Untersuchung der Konfiguration aus 4.2, in der keine Nebeneffekte durch kleine Abstände zwischen den Einschlüssen hervorgerufen werden, liefert ein robustes Verhalten des Lösers – sowohl bezüglich der Sprunghöhe ϵ als auch bezüglich der Levels: In maximal 20 Iterationen ist das Residuum auf einen hinreichend kleinen Wert gefallen. Einzig auf Level 3 zeigt der Glätter in dieser Konfiguration ein subobtimales Verhalten, sodass der Löser hier zwischen 40 und 50 Iterationen benötigt, was asymptotisch jedoch nicht ins Gewicht fällt. Die einfachen Distanzfunktionen führen demnach auch für allgemeinere Koeffizientenfunktionen mit unterschiedlichen Sprunghöhen zu guten Resultaten und einem weitestgehend robusten Löser.

Level	$\epsilon_{\Omega_2} = 10^{-6}$	$\epsilon_{\Omega_2} = 10^{-4}$	$\epsilon_{\Omega_2} = 10^{-2}$	$\epsilon_{\Omega_2} = 10^{0}$	$\epsilon_{\Omega_2} = 10^{2}$	$\epsilon_{\Omega_2} = 10^{4}$	$\epsilon_{\Omega_2} = 10^{6}$
2	2	2	2	2	2	2	2
3	16	16	16	13	10	9	9
4	14	14	14	15	14	15	12
5	16	16	16	15	17	16	15
6	16	16	16	15	17	17	16
7	16	16	16	16	17	18	16
8	20	20	20	22	22	27	29
9	19	19	19	20	21	23	31

Tabelle 5.13: Ergebnisse, Problem (8): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen für zwei quadratische Einschlüsse (vgl. Problem 6.2) mit verschiedenen Sprunghöhen der Koeffizientenfunktion an den Materialinterfaces.

Betrachten wir die Konfiguration mit $\epsilon_{\Omega_2} = 0$, ist der Löser konzeptionell mit demselben Problem wie bei dem einfachen quadratischen Einschluss (Abb. 5.1a) in Abschnitt 5.1 konfrontiert. Anders als in Problem 1 sind die Patches, die einen nichtleeren Schnitt mit dem Interface Γ_M^1 bilden jedoch angereichert, obwohl der lokale glatte Ansatzraum bereits optimale Approximationseigenschaften besitzt (in der Umgebung von Ω_1). Zwar wird durch die Anreicherungsfunktionen die Approximationsqualität der lokalen Ansatzräume verbessert, jedoch impliziert dies, wie wir gesehen haben, nicht in allen Fällen ein besseres Konvergenzverhalten des Lösers: Stärkere Kopplungen zwischen den Funktionen des problemabhängigen Anreicherungsraumes und denen des polynomiellen Ansatzraumes führen u. U. zu deutlich schlechterer Fehlerreduktion durch den Glätter bzw. im schlechtesten Fall zur Fehlerverstärkung. Ein deutliches Indiz, dass der verwendete Block-Gauß-Seidel-Glätter für die verwendeten einfachen Distanzfunktionen leicht schlechtere Glättungseigenschaften aufweist, liefert Tabelle 5.13 für den Fall $\epsilon_{\Omega_2} = 0$. Verglichen mit den entsprechenden Iterationszahlen in Tabelle 5.1 sehen wir einen deutlichen Abfall der Konvergenz des Lösers: im Durchschnitt eine Differenz von 10 Iterationen. Da der Interlevel-Transfer für dieses Beispiel sehr gute Resultate liefert und auch andere Effekte wie kleine Schnitte oder Integrationsprobleme, ausgeschlossen werden können, ist dies auf eine schlechtere Glättungseigenschaft des ML-Schemas zurückzuführen.

Da ein Multilevel-Löser sehr sensibel sowohl von der Approximationseigenschaft als auch von der Glättungseigenschaft abhängt, verlangt ein optimales Multilevel-Schema hervorragende Qualität beider Komponenten. In dieser Arbeit betrachten wir jedoch ausschließlich die Verbesserung der Approximationseigenschaft mithilfe der Anreicherungsfunktionen. Ohne Zweifel wird diese durch die Verwendung jeder der betrachteten Anreicherungsfunktionen effektiv verbessert. Gleichzeitig beobachten wir bei keinem untersuchten Referenzproblem eine herausragende Verbesserung durch die Anwendung komplexerer Anreicherungsfunktionen – oftmals sogar einen Abfall der Konvergenz des Lösers. Neben den bereits diskutierten Effekten und Ursachen kommen hier noch zwei Aspekte hinzu: Zum einen haben wir festgestellt, dass die Auswertung der Anreicherungsfunktionen auf dem Rand des Gebiets immer zu Null evaluiert – bei der Integration in randnahen Punkten auf angereicherten Patches, die den Rand schneiden, kann es somit zu erheblichen Problemen kommen. Dieser Fehler hängt mit der noch nicht ausgereiften Geometriebehandlung zusammen und liefert zusätzliche Erklärungen für schlechtes Verhalten des Lösers in Randnähe (beispelsweise Problem 5 und 7 sowie die Konfiguration für die PU-Anreicherungsfunktionen). Die übrigen Experimente wurden jedoch mit ausreichend Abstand zum Rand entworfen, sodass dieser Effekt hier kaum Einfluss nimmt.
Der zweite Aspekt wirkt sich jedoch sehr viel stärker auf das Lösungsverhalten aus und bezieht sich auf die durch die Anreicherungsfunktionen beeinflusste Glättungseigenschaft der verwendeten Glätter. Wir wollen hier festhalten, dass eine alleinige Optimierung der lokalen Ansatzräume bzgl. der Approximationsqualität nicht ausreicht. Vielmehr muss die Kopplung zwischen glatten Ansatzfunktionen und Anreicherungsfunktionen minimiert werden. Die Entwicklung effizienter Glätter für die spezifischen Anreicherungsfunktionen ist im Hinblick auf die Konvergenz des ML-Lösers sehr vielversprechend. Ebenso kann die stabile Transformation aus Abschnitt 4.2 auf eine gute Glättungseigenschaft der resultierenden Basis hin optimiert werden.

Problem 9: Enrichments für Singularitäten

Als letztes Beispiel wollen wir noch eine Konfiguration für die Verwendung singulärer Anreicherungsfunktionen untersuchen. Referenzproblem ist die in Abbildung 3.1c dargestellte Konfiguration, wobei Ω_0 dem L-Gebiet entspricht und die einspringende Ecke durch den Materialeinschluss Ω_1 implementiert ist. Der durch diese Ecke erzeugte singuläre Charakter der Lösung ist analytisch bekannt und kann dem Verfahren als Anreicherungsfunktion hinzugefügt werden. Eine Abbildung dieser Singularitätsfunktion ist in 4.3, links gegeben. Wir betrachten im Folgenden nur die Fälle $\epsilon \leq 1$, da nur hier das singuläre Verhalten der Lösung provoziert wird. Für $\epsilon < 1$ entsteht ein leichtes Problem mit konvexem Gebiet, von dem wir wissen, dass es robust und schnell gelöst werden kann.
Um ein vernünftiges Verhalten der Prolongation zu garantieren, muss hier ein bezüglich der Levels konstanter Bereich in der Umgebung der Singularität angereichert werden. Die Ergebnisse dieses Experiments sind in Tabelle 5.14 gegeben: links für einfache Distanzfunktionen, rechts unter Verwendung des singulären Enrichments. Wir sehen, dass der Löser mit dem singulären Enrichment wesentlich besseres und robusteres Verhalten zeigt – allerdings nur bis Level 7. Den starken sprunghaften Anstieg der Iterationszahlen ab Level 8 können wir nicht erklären, jedoch wird dieser mit großer Wahrscheinlichkeit durch fehlerhafte, noch nicht ausgereifte Implementierung im Bereich der Geometriebehandlung o. ä. verursacht. Für den Grenzfall $\epsilon \to \infty$ wurde eine gute Konvergenz des Multilevels für dieses Problem unter Verwendung der singulären Anreicherungsfunktion bereits in [14, S. 44 ff] nachgewiesen.

	einfache Distanzfunktionen (gekoppelt)				Singularitätsfunktion als Anreicherungsfunktion			
Level	$\epsilon = 10^0$	$\epsilon = 10^2$	$\epsilon = 10^4$	$\epsilon = 10^6$	$\epsilon = 10^0$	$\epsilon = 10^2$	$\epsilon = 10^4$	$\epsilon = 10^6$
2	2	2	2	2	2	2	2	2
3	9	10	10	9	8	9	9	9
4	9	12	11	15	9	10	10	10
5	16	14	12	12	10	10	10	10
6	11	15	17	12	10	10	10	11
7	12	15	19	15	10	11	11	11
8	10	15	27	52	10	12	20	74
9	53	18	40	24	10	14	38	233

Tabelle 5.14: Ergebnisse, Problem (9): Anzahl der mit $V(1,1)$ ML-Schema vorkonditionierter CG-Iterationen zur Lösung des L-Gebiets (vgl. Abb. 3.1c) sowohl unter Verwendung einfacher Distanzfunktionen als auch mit Anreicherung durch die spezielle Singularitätsfunktion (vgl. 4.3).

KAPITEL 6

Zusammenfassung und Ausblick

Ziel der vorliegenden Arbeit war es, einen robusten Multilevel-Löser im Kontext der Partition of Unity Methode zu realisieren und in ein Framework zur Lösung partieller Differentialgleichungen (Crass) einzubinden. Im Besonderen sollte dabei die Robustheit des Lösers untersucht werden, vor allem im Zusammenhang mit der Verwendung verschiedener Anreicherungsfunktionen. Zu diesem Zweck wurden einerseits diverse Anreicherungsfunktionen entwickelt und realisiert, andererseits wurden alternative Versionen der Interlevel-Transferoperatoren implementiert und analysiert. Wie schon in den Forschungsarbeiten von M. A. Schweitzer erzielte dabei die lokal-nach-lokal-Projektion die besten Ergebnisse, wohingegen die neu entwickelte gewichtete global-nach-global-Projektion nicht die erwarteten Resultate erbrachte und auf Grundlage des jetzigen Wissensstands nicht gewinnbringend eingesetzt werden kann.
Die Verwendung beliebiger Anreicherungsfunktionen erfordert die Transformation der Ansatzfunktionen in eine stabile Basis. Die Implementierung dieser Stabilitätstransformation war Voraussetzung für die Realisierung des Multilevel-Lösers unter Verwendung von Anreicherungsfunktionen. Implementiert und analysiert wurden in dieser Arbeit neben den einfachen Distanzfunktionen verschiedene entkoppelte Versionen derselben sowie die sogenannten blended enrichments und Partition of Unity-Anreicherungsfunktionen. Ebenfalls besteht die Möglichkeit Singularitätsfunktionen zum problemabhängigen Anreicherungsraum hinzuzufügen.
Die Robustheitsanalyse des Lösers war der Schwerpunkt dieser Arbeit. Dabei wurde nahtlos an die Forschungsarbeiten von M. A. Schweitzer angeknüpft und zunächst die Performanz des Lösers für die bereits dort untersuchten Referenzprobleme in der neuen Implementierung des Crass Codes analysiert. Die Experimente zeigten eine deutliche Verbesserung im Bezug auf die benötigten Iterationen des Lösers, was zu einer erheblich besseren Konvergenz desselben für alle betrachteten Konfigurationen führte. Die Verbesserung erklären wir maßgeblich durch eine bessere und exaktere Integration und Geometriebehandlung in der neuen Implementierung. Einige Vermutungen und Erklärungen der früheren Arbeiten konnten so bestätigt werden.

Nichtrobustes Verhalten des Lösers blieb jedoch in einigen Fällen nach wie vor erhalten und war Gegenstand weiterer Untersuchungen – insbesondere in Verbindung mit verschiedenen Anreicherungsfunktionen.

Im Speziellen wurden Probleme mit besonders komplizierter Geometrie (die auf groben Levels nicht aufgelöst werden kann) Probleme mit mehreren Einschlüssen, die teilweise sehr kleine Abstände zueinander aufweisen sowie mehrere Einschlüsse mit unterschiedlichen Eigenschaften, analysiert. Dabei ergab sich, dass der Löser für beliebig komplexe Geometrien sowohl bzgl. der Höhe des Koeffizientensprungs als auch bzgl. der Levels sehr robust ist, jedoch nur unter der folgenden Einschränkung: Die Konfiguration enthält nur einen Einschluss bzw. die Einschlüsse haben sowohl zum Rand als auch untereinander einen genügend großen Abstand, der schon auf groben Levels durch Patches aufgelöst werden kann. Für mehrere Einschlüsse mit beliebigem Abstand untereinander führen verschiedene Effekte wie kleine Schnitte, Integrationsfehler und lösungsfremde Komponenten der Grobgitterlösungen zu einem nicht robusten Verhalten des Multilevel-Lösers.
Die speziell zur Abschwächung dieser Effekte entworfenen Anreicherungsfunktionen führten in ihrer Anwendung zu gleichem bis schlechterem Konvergenzverhalten des Lösers. Obwohl die Approximationsqualität der lokalen Ansatzräume für all diese Funktionen verbessert wurde, wirken sich andere Seiteneffekte der Funktionen negativ auf die Performanz des Lösers aus, sodass die einfachste Form der Distanzfunktionen die besten Ergebnisse zeigte. Insbesondere wird die Glättungseigenschaft des verwendeten Block-Gauß-Seidel-Verfahrens durch stärkere Kopplungen der Funktionen verschlechtert. Abschließend stellen wir also fest, dass es nicht reicht lediglich die Approximationsqualität der lokalen Ansatzräume zu verbessern – eine Optimierung des Lösers muss hier sowohl die Approximations- als auch die Glättungseigenschaft des Multilevel-Lösers berücksichtigen, z. B. durch den Entwurf effizienterer Glätter.

Ausblick

In der vorliegenden Arbeit konnten einerseits Vermutungen früherer Forschungsarbeiten bestätigt werden und teilweise bessere Resultate erreicht werden. Andererseits bietet diese Arbeit durch die genaue Analyse und Herausarbeitung der Problembereiche zahlreiche Anknüpfungspunkte für weiterführende Forschungsarbeiten. Ebenso bedarf es bei einigen Bereichen noch einer eingehenderen und detaillierteren Analyse, die den Umfang dieser Arbeit sprengen. So ist zum Beispiel die gewichtete global-nach-global-Projektion noch einmal genau zu untersuchen, da die hier gewonnenen Resultate in keinster Weise befriedigend sind. Ebenso ist ein Fehler bei der Umsetzung dieses Operators nicht völlig auszuschließen. Nichtsdestotrotz stellt die Möglichkeit der Assemblierung der gewichteten lokalen Masse im neuen Crass Code einen Gewinn dar, da diese für die Simulation dynamischer Prozesse mit explizitem Zeitschrittverfahren wertvoll ist.
Die implementierten Anreicherungsfunktionen sind erst der Anfang einer Reihe von möglichen Verbesserungen für die Partition of Unity Methode. Besonders vielversprechend ist die Verwendung der numerischen Anreicherungsfunktionen, die mit großer Sicherheit einen wesentlich robusteren Löser ermöglichen und insbesondere für Konfigurationen mit sehr vielen ähnlichen Einschlüssen interessant sind. Ebenso ist durch die additive Anreicherung mit

passenden Singularitätsfunktionen ein besseres Ergebnis zu erwarten. Die Partition of Unity Anreicherungsfunktionen sollten ebenso wie die blended enrichments im Zusammenhang mit angepassten, speziell auf die Funktionen optimierten Glättern, noch einmal eingehend untersucht werden, da deren Ansatz prinzipiell vielversprechend ist.
Neben den genannten Anwendungen hat die robuste Lösung partieller Differentialgleichungen mit nichtstetigen Koeffizientenfunktionen eine wichtige Bedeutung für die Lösung stochastischer partieller Differentialgleichungen. Diese Gleichungen gewinnen durch Berücksichtigung von Unsicherheiten im Modellierungsprozess in Forschung, Industrie und Technik immer mehr an Bedeutung. Die Koeffizientenfunktion ist bei diesen Gleichungen ein Feld von Zufallsvariablen, das typischerweise in hohem Maße unstetig ist bzw. signifikante Sprünge aufweist. Ein robuster Löser im Bezug auf springende Koeffizientenfunktionen ist also unbedingte Voraussetzung zur effizienten numerischen Behandlung solcher Gleichungen.

Literaturverzeichnis

[1] J. H. Bramble, J. E. Pasciak, and J. Xu, *Parallel multilevel preconditioners*, Math. Comp., 55 (1990), pp. 1–22. (Zitiert auf Seite 12)

[2] ——, *The analysis of multigrid algorithms with nonnested spaces or noninherited quadratic forms*, Math. Comp., 56 (1991), pp. 1–34. (Zitiert auf Seite 12)

[3] W. L. Briggs, V. E. Henson, and S. F. McComrmick, *A Multigrid Tutorial*, SIAM, 2000. (Zitiert auf Seite 10)

[4] L. Chen, M. Holst, J. Xu, and Y. Zhu, *Local multilevel preconditioners for elliptic equations with jump coefficients on bisection grids*, arXiv:1006.3277v2, (2010). (Zitiert auf Seite 18)

[5] M. Griebel, P. Oswald, and M. A. Schweitzer, *A Particle-Partition of Unity Method—Part VI: A p-robust Multilevel Solver*, in Meshfree Methods for Partial Differential Equations II, M. Griebel and M. A. Schweitzer, eds., vol. 43 of Lecture Notes in Computational Science and Engineering, Springer, 2005, pp. 71–92. (Zitiert auf den Seiten 1, 3, 10 und 13)

[6] M. Griebel and M. A. Schweitzer, *A Particle-Partition of Unity Method—Part III: A Multilevel Solver*, SIAM J. Sci. Comp., 24 (2002), pp. 377–409. (Zitiert auf den Seiten 10 und 13)

[7] M. Griebel and M. A. Schweitzer, *A Particle-Partition of Unity Method—Part V: Boundary Conditions*, in Geometric Analysis and Nonlinear Partial Differential Equations, S. Hildebrandt and H. Karcher, eds., Springer, 2002, pp. 517–540. (Zitiert auf Seite 7)

[8] M. Griebel and G. W. Zumbusch, *Parallel adaptive subspace correction schemes with applications to elasticity*, Comput. Meth. Appl. Mech. Engrg., 184 (20000), pp. 303–332. (Zitiert auf Seite 17)

[9] W. Hackbusch, *Theorie und Numerik elliptischer Differentialgleichungen*, Teubner, 1986. (Zitiert auf Seite 18)

[10] ——, *Iterative Lösung großer schwachbesetzter Gleichungssysteme*, Teubner, 1991. (Zitiert auf Seite 10)

[11] K. Höllig, *Finite Element Methods with B-Splines*, Siam Verlag, 2003. (Zitiert auf den Seiten 27, 28, 33 und 46)

[12] H. A. Schwarz, *Über einige Abbildungsaufgaben*, Vierteljahrsschrift Naturforsch. Ges. Zürich, 15 (1870), pp. 272–286. (Zitiert auf Seite 17)

[13] M. A. Schweitzer, *Variational mass lumping in the partition of unitiy method*. (Zitiert auf Seite 16)

[14] M. A. Schweitzer, *A Parallel Multilevel Partition of Unity Method for Elliptic Partial Differential Equations*, vol. 29 of Lecture Notes in Computational Science and Engineering, Springer, 2003. (Zitiert auf den Seiten 1, 3, 5, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 20, 22, 23, 24 und 56)

[15] ——, *A Particle-Partition of Unity Method Part VIII: Hierarchical Enrichment*, in Lecture Notes in Computational Science and Engineering, M. Griebel and M. A. Schweitzer, eds., vol. 65, Springer, 2008, pp. 277–299. (Zitiert auf den Seiten 1, 3, 30 und 31)

[16] M. A. Schweitzer, *Meshfree and Generalized Finite Element Methods*, habilitation, Institute for Numerical Simulation, University of Bonn, 2008. (Zitiert auf den Seiten 10 und 31)

[17] M. A. Schweitzer, *Stable enrichment and local preconditioning in the particle-partition of unitiy method*, Numer. Math. 118, 1 (2011), pp. 137–170. (Zitiert auf den Seiten 1, 3, 30 und 31)

[18] ——, *Generalizations of the finite element method*, Central European Journal of Mathematics 10, 1 (2012), pp. 3–24. (Zitiert auf den Seiten 25 und 30)

[19] ——, *Multilevel partition of unity method for elliptic problems with strongly discontinuous coefficients*, (2012). (Zitiert auf den Seiten 2, 6, 25, 26, 33, 36, 37, 38, 40 und 41)

[20] Z. Wang, C. Wang, and K. Chen, *Two-phase flow and transport in the air cathode of proton exchange membrane fuel cells*, Journal of Power Sources, 94 (2001), pp. 40 – 50. (Zitiert auf Seite 6)

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschift