



Fakultät für Mathematik und Physik
Institut für Angewandte Mathematik

Diplomarbeit

Ein hierarchischer Fehlerschätzer für Hindernisprobleme

von Cornelius Rüther
Matr.-Nr.: 2517350

4. Dezember 2014

Erstprüfer: Prof. Dr. Gerhard Starke
Zweitprüfer: Prof. Dr. Peter Wriggers

Inhaltsverzeichnis

Abbildungsverzeichnis	vii
Tabellenverzeichnis	vii
Algorithmenverzeichnis	viii
Quellcodeverzeichnis	ix
1 Einleitung	10
2 Grundlagen	13
2.1 Hilberträume	13
2.2 Variationsformulierung	15
2.3 Finite Elemente Methode	24
2.3.1 A priori Fehlerabschätzung	31
2.4 Adaptive Verfeinerungsstrategien	34
2.4.1 A posteriori Fehlerschätzer	34
2.4.2 Verfeinerung des Netzes	35
2.5 Einführung in die Strukturmechanik	37
2.5.1 Kinematik	37
2.5.2 Kinetik	40
2.5.3 Konstitutive Gleichungen und Prinzipien	41
3 Variationsungleichungen	43
3.1 Ein Hindernisproblem	43
3.1.1 Variationsformulierung für das Hindernisproblem	44
3.1.2 Existenz und Eindeutigkeit der Lösung	47
3.1.3 Lösung des Hindernisproblems mittels FEM	49
3.2 Kontaktprobleme	56
3.2.1 Mathematische Modellierung eines Kontaktproblems	56
3.2.2 Variationsformulierung des Signorini-Kontaktproblems	60
3.2.3 Lösung des Kontaktproblems mittels FEM	65

4 Ein hierarchischer Fehlerschätzer für Hindernisprobleme	67
4.1 Herleitung von einem hierarchischen a posteriori Fehlerschätzer	68
4.1.1 Diskretisierung des Defektproblems	68
4.1.2 Lokaler Anteil des Fehlerschätzers	75
4.1.3 Oszillationsterme	80
4.1.4 Zuverlässigkeit des Fehlerschätzers	83
4.1.5 Effizienz des Fehlerschätzers	98
4.2 Ein adaptiver Algorithmus	100
4.3 Erfüllung einer Saturationseigenschaft	101
4.4 Übertragung des Fehlerschätzers auf Kontaktprobleme	104
5 Implementierung des Fehlerschätzers in Matlab	109
5.1 Implementierung eines Hindernisproblems	109
5.2 Implementierung eines Kontaktproblems	117
6 Validierung	122
6.1 Numerische Beispiele zum Hindernisproblem	122
6.2 Numerisches Beispiel zum Kontaktproblem	127
7 Zusammenfassung und Ausblick	130
Literaturverzeichnis	131
A Funktionalanalysis	134
A.1 Sobolev-Räume	134
A.2 Optimalitätskriterien	136
A.3 Konvergenzbegriffe	137
B Optimierung	139
B.1 Quadratische Programmierung	139
B.2 Active Set-Methode für konvexe QPs	140
B.3 Algorithmus	144
C Tensorrechnung	145
C.1 Tensoralgebra	145
C.2 Tensoranalysis	147
D Quellcode	149
D.1 Implementierung des Fehlerschätzers für das Hindernisproblem	149
D.2 Implementierung des Fehlerschätzers für das Kontaktproblem	173
Index	188
Selbstständigkeitserklärung	188

Abbildungsverzeichnis

1.1	Auslenkung einer eingespannten Membran unter Einwirkung einer vertikalen Lastfunktion f	11
2.1	Membran Ω mit Flächenlast f und Auslenkung $u(x)$	15
2.2	Zulässige und unzulässige Triangulierung (mit hängendem Knoten)	27
2.3	Dreiecke für nodale Basen (linear, quadratisch, kubisch)	28
2.4	Triangulierung von $\Omega = [-1, 1]^2$ in 8 Courant-Elemente	29
2.5	Referenzelement \tilde{T} für ein allgemeines Dreieck $T \in \mathcal{T}_h$	31
2.6	Verfeinerungen von Dreiecken	36
2.7	Konfiguration eines Kontinuums \mathcal{B}	37
3.1	Ein Hindernisproblem mit Hindernis ψ , konstanter Streckenlast f und Lösung u	44
3.2	Körper \mathcal{B}^1 und \mathcal{B}^2 mit Randbezeichnungen	57
3.3	Kontaktformulierung zwischen zwei Körpern	57
4.1	Beispiel eines affinen Hindernisses ψ mit $v \in \mathcal{A}_{\mathcal{Q}}$ in \mathbb{R}	72
4.2	Dreiecke T_1 und T_2 mit Einheitsnormalen \mathbf{n}	77
4.3	Darstellung von ω_p (grau) und \mathcal{E}_p (abgehende Kanten von p) für ein beliebiges ϕ_p	78
5.1	Ungefährre Lage der Stützstellen in einem allgemeinen Dreieck für die Gauß-Quadratur	114
6.1	Exakte und numerische Lösung des Beispiels 6.1	123
6.2	Gitterverfeinerung für das Hindernisproblem im Beispiel 6.1 mit $\theta_1 = 0,4$ und $\theta_2 = 0,3$	124
6.3	Diagramm mit dem Fehler und der Oszillation für Beispiel 6.1	126
6.4	Numerische Lösung des Beispiels 6.2 in der Verfeinerungsstufe 7 mit $\theta_1 = \theta_2 = 0,3$	127
6.5	Netzverfeinerungen für das Beispiel 6.2 mit $\theta_1 = \theta_2 = 0,3$	128
6.6	Numerische Lösung des Beispiels 6.3 mit $\theta_1 = \theta_2 = 0,3$	128
6.7	Netzverfeinerungen für das Beispiel 6.3 mit $\theta_1 = \theta_2 = 0,3$	129

Tabellenverzeichnis

2.1	Ableitungen der nodalen Basisfunktion ϕ_5	29
5.1	Stützstellen (ξ_k, η_k) und Gewichte w_k für die Gauß-Quadratur über das Referenzdreieck \tilde{T}	113
6.1	Vergleich von adaptiver und uniformer Verfeinerung für Bei- spiel 6.1	125

Algorithmenverzeichnis

4.1	Adaptive Verfeinerungsstrategie für ein Hindernisproblem . . .	101
B.1	Active-Set-Methode für konvexe quadratische Probleme . . .	144

Quellcodeverzeichnis

D.1	Assemblierung von Matrix und Vektor	149
D.2	Berechnung der lokalen Anteile von ρ_S	150
D.3	Lösung des lokalen Defektproblems (4.9)	151
D.4	Dreiecksindizes zur Verfeinerung	153
D.5	Gradientenberechnung	154
D.6	Bestimmung der inneren Knoten $\mathcal{N} \cap \Omega$	155
D.7	Bestimmung der inneren Kontaktknoten \mathcal{N}^0	155
D.8	Bestimmung der inneren Nicht-Kontaktknoten \mathcal{N}^+	155
D.9	Bestimmung der Menge \mathcal{N}^{++}	155
D.10	Berechnung der Menge an Knoten aus \mathcal{N}^{0+} und \mathcal{N}^{0-}	156
D.11	Bestimmung der lokalen Steifigkeitsmatrix	157
D.12	Berechnung der Indizes anliegender Dreiecke	158
D.13	Bestimmung der Mittelpunkte und Zuordnung zu den Dreiecken	159
D.14	Berechnung der Normalflüsse j_E für alle $E \in \mathcal{E}_p$	160
D.15	Bestimmung der Oszillation $\text{osc}_1(u_S, \psi)$	161
D.16	Berechnung der Oszillationsterme $\text{osc}_2(u_S, \psi, f)$	162
D.17	Gewichte und Stützstellen für die Quadratur	163
D.18	Adaptive Verfeinerung des Gitters und Lösung des Hindernisproblems	164
D.19	Startdatei (Beispiel 6.1)	166
D.20	Startdatei (Beispiel 6.2)	168
D.21	Startdatei (Beispiel 6.3)	171
D.22	Assemblierung von Matrix und Vektor für das Kontaktproblem	173
D.23	Berechnung der lokalen Anteile von ρ_S	176
D.24	Berechnung der lokalen Steifigkeitsmatrix für das Kontaktproblem	178
D.25	Bestimmung des Normalenspannungsflusses über eine Kante E	179
D.26	Gewichte und Stützstellen für die Quadratur über eine Kante	180
D.27	Adaptive Verfeinerung des Gitters und Lösung des Kontaktproblems	181
D.28	Startdatei (Beispiel ??)	185

Kapitel 1

Einleitung

Partielle Differentialgleichungen sind in vielen Bereichen der Natur- und Ingenieurwissenschaften wiederzufinden. Da die exakte Lösung nicht immer existiert bzw. die Bestimmung solch einer Lösung beliebig kompliziert werden kann, haben numerische Löser für partielle Differentialgleichungen in den letzten Jahrzehnten insbesondere durch die Weiterentwicklung der Computertechnologie immer mehr an Bedeutung gewonnen. Die Lösung kann daher z.B. mittels Galerkin-Verfahren (bzw. Finiter-Elemente-Methode) ermittelt werden. Hierbei wird die eigentliche Differentialgleichung nur noch approximativ auf einem Gitter gelöst, wobei wir durch Verfeinerung des Gitters die Genauigkeit der Lösung erhöhen, was allerdings auch zu höherem Rechenaufwand führt. Daher ist es sinnvoll, die Verfeinerung des Gitters gerade so zu steuern, dass möglichst wenig, aber für eine hinreichend genaue Lösung noch genügend Elemente verfeinert werden. Diese Verfahren werden als *adaptive Verfeinerungsstrategien* bezeichnet und sind im Allgemeinen von der Form:

solve → estimate → mark → refine.

Da die exakte Lösung in den meisten Fällen nicht bekannt ist, ist der Schritt „estimate“ von entscheidender Bedeutung. Hierbei wird der Fehler zur exakten Lösung auf dem nächsten Gitter mittels eines a posteriori Fehlerschätzer bzgl. des aktuellen Gitters abgeschätzt, was Zuverlässigkeit und Effizienz des Schätzers voraussetzt. Dadurch wird sichergestellt, dass durch Verringerung des Fehlerschätzers wird auch der exakte Fehler kleiner. Im Schritt „mark“ werden dann genau diejenigen Elemente ausgewählt, die an dem gewählten Schätzer einen hohen lokalen Anteil haben.

Diese Verfeinerungsstrategien wollen wir in der vorliegenden Arbeit zunächst für einen hierarchischen Fehlerschätzer zu einem Hindernisproblem untersuchen. Solche Schätzer basieren auf einer Hierarchie von Ansatzräumen, d.h. man untersucht das gegebene Problem in einem „besseren“ Finite-Element-Raum und schätzt damit den Fehler bzgl. der „schlechteren“ Galerkin-Approximation ab. Ein Modellproblem eines Hindernisproblems stellt

z.B. eine eingespannte Membran dar, die mit einer Last f vertikal belastet und deren Auslenkung $u : \Omega \rightarrow \mathbb{R}$ durch ein Hindernis ψ behindert wird. Da die in der Membran gespeicherte Energie minimal ist, kann man die Auslenkung dieser durch ein Optimierungsproblem von der Form

$$u = \arg \min_{v \in K} J(v) = \frac{1}{2} \int_{\Omega} \nabla v \cdot \nabla v \, dx - \int_{\Omega} f v \, dx \quad (1.1)$$

modellieren, wobei K die Menge der Auslenkungen u ist, die oberhalb des Hindernisses ψ liegen, d.h. $u \geq \psi$ erfüllen. In Abbildung 1.1 sind im Vergleich die Auslenkungen einer auf einem Kreisgebiet belasteten Membran ohne und mit (ebenem) Hindernis dargestellt. Für das Problem (1.1) werden wir in Kapitel 4.1 einen wie oben beschriebenen hierarchischen a posteriori Fehlerschätzer herleiten. Das Hauptresultat dieser Arbeit befindet sich in Theorem 4.22, welches die Verwendung des Schätzers ermöglicht.

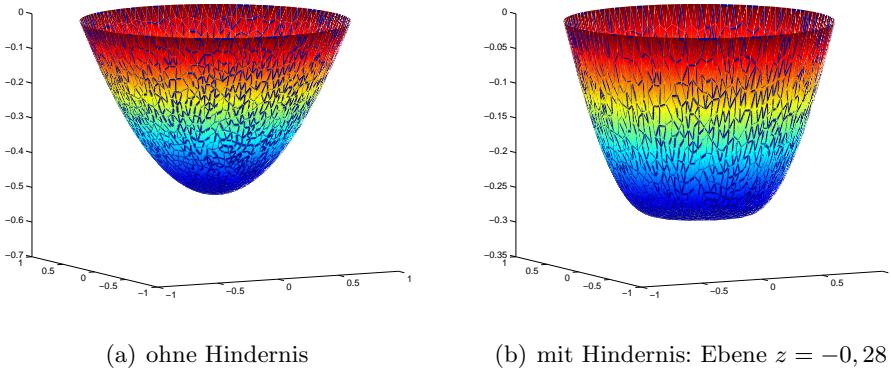


Abbildung 1.1: Auslenkung einer eingespannten Membran unter Einwirkung einer vertikalen Lastfunktion f

Nachdem wir in Kapitel 4 einen hierarchischen a posteriori Fehlerschätzer für das Modellproblem (1.1) untersucht haben, wollen wir diesen auf Kontaktprobleme übertragen. Kontaktprobleme sind Probleme aus der Strukturmechanik und stellen eine Anwendung von partiellen Differentialgleichungen unter Nebenbedingung (nämlich der Kontaktbedingung) dar. Wir werden uns in dieser Arbeit mit einem vereinfachten Kontaktproblem, dem *Signorini-Kontaktproblem* beschäftigen, d.h. es gibt keine Reibungskräfte auf der Kontaktfläche. Außerdem werden wir von einem linear-elastischen Fall ausgehen. Die Lösung eines Kontaktproblems kann wie oben eindeutig (vgl. Kapitel 3.2) durch die Minimierung des Energiefunktional

$$\mathcal{J}(\mathbf{v}) = \frac{1}{2} \int_{\Omega} \boldsymbol{\sigma}(\mathbf{v}) : \boldsymbol{\varepsilon}(\mathbf{v}) \, d\Omega - \int_{\Omega} \mathbf{b} \cdot \mathbf{v} \, d\Omega - \int_{\Gamma} \mathbf{t} \cdot \mathbf{v} \, d\Gamma \quad (1.2)$$

1. Einleitung

über \mathcal{K} berechnet werden, wobei die Menge \mathcal{K} diejenigen Verschiebungsfelder \mathbf{v} enthält, die die Kontaktbedingung $\mathbf{v} \cdot \mathbf{n} - g \leq 0$ erfüllen. Die Funktion g gibt hierbei die Lücke zwischen den beiden Körpern an und wird daher auch als *Gap-Funktion* bezeichnet. Diese stellt für Problem (1.2) also das Hindernis dar, wie zuvor ψ für (1.1). In Kapitel 4.4 werden dann die Übertragungen der Konzepte des Fehlerschätzers für Problem (1.1) beschrieben.

Zunächst werden wir jedoch in Kapitel 3 zeigen, dass die beiden Probleme (1.1) und (1.2) grundlegend auf dieselben Variationsungleichungen führen und die Existenz einer eindeutigen Lösung für solch eine Variationsungleichung, und damit auch für die jeweiligen Probleme, zeigen.

In Kapitel 2 werden grundlegende Konzepte zu Hilberträumen, Variationsrechnung, adaptive Verfeinerungsstrategien und der Strukturmechanik eingeführt. Weitere Resultate, die in der vorliegenden Arbeit verwendet werden und nicht in Kapitel 2 aufgeführt sind, können in Anhang A (Funktionalanalysis), B (Optimierung) und C (Tensorrechnung) nachgeschlagen werden.

Die Lösungsverfahren für die adaptive Verfeinerung der Hindernis- bzw. Kontaktprobleme werden in Matlab implementiert und die wichtigsten Schritte hierfür in Kapitel 5 beschrieben. Der Quellcode ist im Detail im Anhang D einzusehen.

In Kapitel 6 werden wir abschließend mehrere Beispiele für Hindernis- bzw. Kontaktprobleme präsentieren, welche die Resultate aus Kapitel 4 untermauern.

Kapitel 2

Grundlagen

In diesem Kapitel wollen wir uns mit grundlegender Theorie beschäftigen, die nicht im Anhang aufgeführt, zum Verständnis der darauffolgenden Kapitel jedoch notwendig ist.

Dieses Kapitel basiert auf [Bra13], [Sta08], [Ste12b], [Wal11], [Alt12].

2.1 Hilberträume

Wir benötigen für die Variationsrechnung Hilberträume und wollen uns daher in diesem Kapitel mit wichtigen Eigenschaften solcher Räume im Allgemeinen beschäftigen. Zunächst führen wir ein, was wir unter einem Hilbertraum verstehen.

Definition 2.1. Ein *Hilbertraum* ist ein reeller oder komplexer Vektorraum H mit Skalarprodukt $(\cdot, \cdot)_H$, der vollständig bzgl. der durch das Skalarprodukt induzierten Norm, $\|v\|_H^2 := (v, v)_H$ für alle $v \in H$, ist, d.h. in dem jede Cauchy-Folge konvergiert.

Beispiel. Es sei $H = \mathbb{R}^n$ und $(\cdot, \cdot)_H : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ definiert durch das Standardskalarprodukt. Dann konvergiert jede Cauchy-Folge in H bzgl. der durch $(\cdot, \cdot)_H$ induzierten (euklidischen) Norm (vgl. [Kö00], metrische Räume) und damit ist H ein Hilbertraum.

Wir wollen die im Folgenden aufgeführten Eigenschaften später auf weniger triviale Räume anwenden, vor allem den Funktionenraum $H_0^1(\Omega)$ (s. Anhang A, Sobolev-Räume). Es sei in diesem Kapitel H ein reeller Hilbertraum mit Skalarprodukt $(\cdot, \cdot)_H$ und der dazu induzierten Norm $\|v\|_H^2 = (v, v)_H$ für alle $v \in H$.

Bemerkung. Für alle $v, w \in H$ gilt die Cauchy-Schwarz'sche Ungleichung

$$(v, w)_H \leq \|v\|_H \|w\|_H .$$

2. Grundlagen

Da wir uns in dieser Arbeit mit Variationsproblemen über konvexen abgeschlossenen Mengen beschäftigen werden, sammeln wir zunächst einige Aussagen bzgl. dieser Mengen.

Satz 2.2 (Approximationssatz). *Es sei $\emptyset \neq M \subset H$ konvex und abgeschlossen. Dann existiert für alle $v \in H$ ein $m_v \in M$ mit*

$$\|v - m_v\|_H = \text{dist}(v, M) := \inf_{w \in M} \|v - w\|_H.$$

Wir nennen $P_M : H \rightarrow M$ mit $v \mapsto m_v$ die Projektionen auf M .

Beweis. Der Beweis ist in [Wal11] Kapitel 7.1 Satz 7.2 zu finden. \square

Satz 2.3 (Charakterisierung der Projektionen). *$\emptyset \neq M \subset H$ sei konvex, abgeschlossen und $v \in H$. Dann gilt:*

$$m_0 = P_M(v) \iff (m - m_0, v - m_0)_H \leq 0$$

für alle $m \in M$.

Beweis. Es sei o.B.d.A. $0 \in M$ und $m_0 = 0$.

„ \Rightarrow “ Wegen $0 = P_M(x)$ muss $\|v - tm\|_H \geq \|v\|_H$ für $m \in M$ und $0 \leq t \leq 1$ sein. Dann ist

$$\|v\|_H^2 \leq \|v\|_H^2 - 2t(v, m)_H + t^2\|m\|_H^2 \implies 0 \leq -2t(v, m)_H + \underbrace{t^2\|m\|_H^2}_{\geq 0}.$$

Damit ist $2(v, m)_H \leq 0$.

„ \Leftarrow “ Für alle $m \in M$ ist $(v, m)_H \leq 0$. Es folgt

$$\|v\|_H^2 \leq \|v\|_H^2 + \|m\|^2 - 2(v, m)_H = \|v - m\|_H^2.$$

Wegen $0 \in M$ ist $\text{dist}(v, M) = \|v\|_H$ und damit $0 = P_M(v)$. \square

Satz 2.4. *Es sei $\emptyset \neq M \subset H$ konvex und abgeschlossen. Dann gilt:*

$$\|P_M(v) - P_M(w)\|_H \leq \|v - w\|_H \quad \forall v, w \in H.$$

Beweis. Da $P_M(v), P_M(w) \in M$ für alle $v, w \in H$ ist, folgt aus Satz 2.3

$$(P_M(w) - P_M(v), v - P_M(v))_H \leq 0, \tag{2.1}$$

$$(P_M(v) - P_M(w), w - P_M(w))_H \leq 0. \tag{2.2}$$

Addieren wir (2.1) und (2.2), so erhalten wir

$$\begin{aligned} 0 &\geq (P_M(w) - P_M(v), v - P_M(v))_H + (P_M(v) - P_M(w), w - P_M(w))_H \\ &= (P_M(w) - P_M(v), v - w + P_M(w) - P_M(v))_H \\ &= \|P_M(w) - P_M(v)\|_H^2 - (P_M(w) - P_M(v), w - v)_H \\ &\stackrel{\text{C.S.}}{\geq} \|P_M(w) - P_M(v)\|_H^2 - \|P_M(w) - P_M(v)\|_H \|w - v\|_H. \end{aligned}$$

Nach Umstellen der Ungleichung folgt die Behauptung. \square

Definition 2.5. Es sei $\emptyset \neq M \subset H$ und wir definieren das *orthogonale Komplement* von M durch

$$M^\perp := \{v \in H \mid v \perp M\} := \{v \in H \mid (v, m)_H = 0 \forall m \in M\}.$$

Satz 2.6. Es sei M ein abgeschlossener Untervektorraum von H . Dann ist

$$H = M \oplus M^\perp,$$

d.h. jedes $v \in M$ hat eine eindeutige Zerlegung $v = v_M + v_{M^\perp}$ mit $v_M \in M$ und $v_{M^\perp} \in M^\perp$.

Beweis. Der Beweis findet sich in [Wal11] Kapitel 7.1 Theorem 7.6. \square

Korollar 2.7. Es sei $\emptyset \neq M \subset H$ ein Untervektorraum. Dann ist der Abschluss $\bar{M} = H$ genau dann, wenn $M^\perp = \{0\}$ ist.

Beweis. Man kann zeigen, dass $\overline{\text{span } M} = (M^\perp)^\perp =: M^{\perp\perp}$ ist und dann folgt unter Verwendung von Satz 2.6 die Behauptung. Den kompletten Beweis können wir in [Wal11] Kapitel 7.1 Korollar 7.7 (iii) einsehen. \square

2.2 Variationsformulierung

Bevor wir uns mit Variationsproblemen auf konvexen Teilmengen eines Hilbertraumes beschäftigen, wollen wir die Variationsrechnung an einem einfachen Modellproblem ohne Nebenbedingung beschreiben.

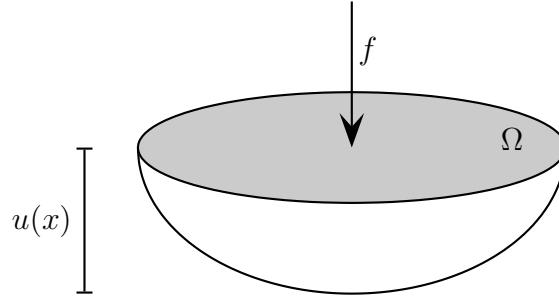


Abbildung 2.1: Membran Ω mit Flächenlast f und Auslenkung $u(x)$

Wir betrachten als Modellproblem die Auslenkung $u : \Omega \rightarrow \mathbb{R}$ einer in $\Omega \subset \mathbb{R}^d$ eingespannten Membran unter Krafteinwirkung f . Mathematisch beschrieben wird dies durch das *Dirichlet-Problem*

$$-\Delta u = f \text{ in } \Omega, \tag{2.3a}$$

$$u = g \text{ auf } \partial\Omega, \tag{2.3b}$$

dabei ist $g : \partial\Omega \rightarrow \mathbb{R}$ eine für die Randwerte von u gegebene Funktion.

Notation. In der Praxis übliche Dimensionen sind $d = 2, 3$. Der Einfachheit halber sei im Folgenden $d = 2$ und $\Omega \subset \mathbb{R}^2$ ein durch ein Polygonzug berandetes Gebiet, wobei wir den Rand $\partial\Omega$ auch mit Γ bezeichnen.

Bemerkung. Sollte Ω ein allgemein berandetes Gebiet sein, so können wir dieses beliebig genau durch ein polygonales Gebiet approximieren; hierbei entsteht schon bei der Gebietszerlegung ein Fehler.

Diesen Fehler kann man durch Verwendung von *isoparametrischen Elementen* (vgl. [Bra13] Kapitel III, §2, Isoparametrische Elemente) verringern. Dies soll in dieser Arbeit aber nicht weiter vertieft werden.

Es sei $u_0 : \Omega \rightarrow \mathbb{R}$ eine für das Dirichlet-Problem zulässige Funktion, d.h. die für (2.3) hinreichend regulär ist und für die $u_0 = g$ auf Γ gilt. Dann gilt für $\tilde{u} = u - u_0$

$$-\Delta \tilde{u} = \tilde{f} \text{ in } \Omega, \quad (2.4a)$$

$$\tilde{u} = 0 \text{ auf } \Gamma \quad (2.4b)$$

mit $\tilde{f} = f - \Delta u_0$. Also reicht es aus, sich auf das *homogene Dirichlet-Problem* (2.4) zu beschränken. Im Folgenden betrachten wir somit (2.3) mit $g \equiv 0$.

Mit $H_0^1(\Omega)$ bezeichnen wir, wie in Bemerkung A.8 im Anhang beschrieben, den Raum der in Ω einmal schwach differenzierbaren Funktionen, die am Rand Γ verschwinden im Sinne der Spur. Wählen wir nun ein beliebiges $v \in H_0^1(\Omega)$, dann folgt durch Multiplikation von (2.3a) mit v und Integration über Ω die Beziehung

$$\int_{\Omega} -\Delta u \cdot v \, dx = \int_{\Omega} fv \, dx.$$

Wir betrachten nun (2.3a) also nicht mehr punktweise (lokal), sondern im gewichteten Mittel über ganz Ω (global). Durch Anwenden der 1. Green'schen Formel (C.4) (bzw. dem Satz von Gauß) ergibt sich

$$\begin{aligned} \int_{\Omega} \nabla u \cdot \nabla v \, dx - \underbrace{\int_{\Gamma} v \partial_{\nu} u \, ds}_{=0, \text{ da } v|_{\Gamma}=0} &= \int_{\Omega} fv \, dx \\ \iff \quad \int_{\Omega} \nabla u \cdot \nabla v \, dx &= \int_{\Omega} fv \, dx. \end{aligned} \quad (2.5)$$

Die Gleichung (2.5) wird als *Variationsgleichung* bezeichnet. Wenn wir die Notationen aus Satz A.5 (b) verwenden, so können wir (2.5) kurz schreiben als

$$(\nabla u, \nabla v)_0 = (f, v)_0,$$

damit definieren wir die Bilinearform $a : (H_0^1(\Omega))^2 \rightarrow \mathbb{R}$, $a(u, v) := (\nabla u, \nabla v)_0$ und $(f, v) := (f, v)_0$.

Bemerkung. Wir werden in dieser Arbeit oftmals auch $a(\cdot, \cdot)$ für eine beliebige Bilinearform $a : H \times H \rightarrow \mathbb{R}$ verwenden.

Definition 2.8. Eine Funktion $u \in H_0^1(\Omega)$ heißt *schwache Lösung* vom homogenen Dirichlet-Problem

$$\begin{aligned} -\Delta u &= f \text{ in } \Omega, \\ u &= 0 \text{ auf } \Gamma, \end{aligned} \tag{DP}$$

wenn die Gleichung

$$a(u, v) = (f, v) \quad \forall v \in H_0^1(\Omega) \tag{2.6}$$

gilt.

Anschaulich nennen wir eine solche Lösung schwach, da sie das Problem (DP) nur im gewichteten Mittel löst. Eine schwache Lösung muss das *starke Problem* nicht lösen, da sie beispielsweise die Regularitätsanforderungen an das Problem nicht erfüllen muss.

Wir wollen uns nun die Frage nach der Eindeutigkeit und Existenz einer Lösung für die Variationsgleichung (2.6) stellen. Diese Frage wollen wir zunächst allgemein für einen beliebigen reellen Hilbertraum H beantworten. Wie wir nachher im Beweis des zentralen Satzes von Lax-Milgram sehen werden, ist hierfür explizit ein Hilbertraum notwendig.

Zuvor benötigen wir allerdings noch ein paar Definitionen und Eigenschaften für Bilinearformen.

Definition 2.9. Sei H ein Hilbertraum. Die Bilinearform $a : H \times H \rightarrow \mathbb{R}$ heißt *stetig*, falls mit einem $c > 0$

$$|a(u, v)| \leq c \|u\|_H \|v\|_H \quad \forall u, v \in H$$

gilt. Sie heißt *H -elliptisch* (oder kurz *elliptisch* oder *koerziv*), falls es ein $\alpha > 0$ gibt, so dass

$$a(v, v) \geq \alpha \|v\|_H^2 \quad \forall v \in H$$

gilt.

Da man die Variationsgleichung (2.6) auch aus der Minimierung eines quadratischen Energiefunktionalen $J : (H_0^1(\Omega))^2 \rightarrow \mathbb{R}$, $J(v) := \frac{1}{2}a(v, v) - (f, v)$ herleiten kann, wollen wir für ein solches Funktional zuvor einige Eigenschaften sammeln.

Lemma 2.10. *Es sei H ein Hilbertraum. Das Funktional*

$$J : H \rightarrow \mathbb{R}, \quad J(v) := \frac{1}{2}a(v, v) - F(v),$$

ist konvex, wobei $a : H \times H \rightarrow \mathbb{R}$ eine stetige bilineare koerzive und $F : H \rightarrow \mathbb{R}$ eine lineare Abbildung ist.

Beweis. Es seien $u, v \in H$, dann gilt $u + t(v - u) = (1-t)u + tv \in H$ (dies gilt auch, wenn wir den Satz auf eine konvexe Teilmenge $M \subset H$ beschränken). Damit folgt mit $t \in [0, 1]$:

$$\begin{aligned}
J((1-t)u + tv) &= \frac{1}{2}a((1-t)u + tv, (1-t)u + tv) - F((1-t)u + tv) \\
&= (1-t)J(u) + tJ(v) + \frac{1}{2}a((1-t)u + tv, (1-t)u + tv) \\
&\quad - \frac{1}{2}(1-t)a(u, u) - \frac{1}{2}ta(v, v) \\
&= (1-t)J(u) + tJ(v) + \frac{1}{2}a(u, u) + t a(u, v - u) \\
&\quad + \frac{t^2}{2}a(v - u, v - u) - \frac{1}{2}(1-t)a(u, u) - \frac{1}{2}ta(v, v) \\
&= (1-t)J(u) + tJ(v) + \frac{t^2}{2}a(v - u, v - u) \\
&\quad + t a(u, v) - \frac{1}{2}t a(u, u) - \frac{1}{2}t a(v, v) \\
&\quad \underbrace{- \frac{1}{2}ta(v - u, v - u)}_{=-\frac{1}{2}ta(v-u,v-u)} \\
&= (1-t)J(u) + tJ(v) - \underbrace{\frac{1}{2}t(1-t)}_{\geq 0} \underbrace{a(v - u, v - u)}_{\geq \alpha \|v - u\|_H^2 \geq 0} \\
&\leq (1-t)J(u) + tJ(v).
\end{aligned}$$

Daraus folgt die Behauptung. \square

Lemma 2.11. *Sei H ein Hilbertraum. Das Funktional $J : H \rightarrow \mathbb{R}$, $J(v) = \frac{1}{2}a(v, v) - F(v)$ aus Lemma 2.10 ist Gâteaux-differenzierbar (s. Definition A.10).*

Beweis. Wir rechnen einfach nach, dass der Grenzwert des Differenzenquotienten existiert und verwenden dabei die Bilinearität von a und die Linearität von F . Seien $u, v \in H$, dann gilt

$$\begin{aligned}
\mathcal{D}_v J(u) &= \lim_{t \rightarrow 0} \frac{J(u + tv) - J(u)}{t} \\
&= \lim_{t \rightarrow 0} \frac{J(u) + t(a(u, v) - F(v)) + \frac{t^2}{2}a(v, v) - J(u)}{t} \\
&= \lim_{t \rightarrow 0} (a(u, v) - F(v)) + \frac{t}{2}a(v, v) \\
&= a(u, v) - F(v) < \infty,
\end{aligned}$$

da a und F jeweils stetig sind und daher durch $\|u\|_H, \|v\|_H$ beschränkt sind. Damit folgt die Behauptung. \square

Nun können wir die Existenz und Eindeutigkeit einer Lösung durch das folgende Theorem zeigen.

Theorem 2.12 (Lax-Milgram). *Es sei H ein Hilbertraum und $a : H \times H \rightarrow \mathbb{R}$ eine symmetrische, in H stetige, koerzive Bilinearform. Weiter sei $F : H \rightarrow \mathbb{R}$ ein stetiges lineares Funktional, d.h.*

$$|F(v)| \leq c \|v\|_H \quad \forall v \in H$$

mit einer Konstante $c > 0$. Dann gibt es eine eindeutige Lösung $u \in H$, für die

$$a(u, v) = F(v) \quad \forall v \in H.$$

gilt. Diese minimiert den Ausdruck

$$J(v) = \frac{1}{2}a(v, v) - F(v)$$

unter allen $v \in H$.

Beweis. (i) Zunächst zeigen wir die Äquivalenz der beiden oberen Probleme.

„ \Rightarrow “ Es sei $u \in H$, so dass $a(u, v) = F(v) \forall v \in H$. Für $t > 0$ und $v \in H$ gilt dann

$$\begin{aligned} J(u + tv) &= \frac{1}{2}a(u + tv, u + tv) - F(u + tv) \\ &= \frac{1}{2}a(u, u) + t a(u, v) + \frac{t^2}{2}a(v, v) - F(u) - t F(v) \\ &= \frac{1}{2}a(u, u) - F(u) + t \underbrace{(a(u, v) - F(v))}_{=0} + \frac{t^2}{2} \underbrace{a(v, v)}_{\geq 0, \text{ da } a \text{ koerziv}} \\ &> \frac{1}{2}a(u, u) - F(u) = J(u), \end{aligned}$$

also ist $u = \arg \min_{v \in H} J(v)$.

„ \Leftarrow “ Es sei $u \in H$ das Minimum von dem Problem

$$\min_{v \in H} J(v) = \frac{1}{2}a(v, v) - F(v).$$

Da $J : H \rightarrow \mathbb{R}$ nach Lemma 2.10 ein konvexes Funktional ist und J nach Lemma 2.11 Gâteaux-differenzierbar, gilt mit Satz A.11 für alle $v \in H$

$$\begin{aligned} 0 &= \mathcal{D}_v J(u) = \frac{d}{dt} J(u + tv) \Big|_{t=0} \\ &= \frac{d}{dt} (J(u) + t(a(u, v) - F(v)) + \frac{t^2}{2}a(v, v)) \Big|_{t=0} \\ &= a(u, v) - F(v) + t a(v, v) \Big|_{t=0} = a(u, v) - F(v) \end{aligned}$$

2. Grundlagen

(ii) Eindeutigkeit: Es seien $u, \tilde{u} \in H$ Lösungen der Variationsgleichung, d.h.

$$a(u, v) = F(v) \wedge a(\tilde{u}, v) = F(v) \quad \forall v \in H.$$

Damit folgt durch Subtraktion der beiden Gleichungen für alle $v \in H$

$$a(u, v) = a(\tilde{u}, v) \iff a(u - \tilde{u}, v) = 0. \quad (2.7)$$

Da H ein Vektorraum ist, gilt auch $u - \tilde{u} \in H$. Ersetzen wir also in (2.7) $v = u - \tilde{u}$, dann ergibt sich

$$0 = a(u - \tilde{u}, u - \tilde{u}) \stackrel{\substack{a \text{ koerziv} \\ F \text{ linear}}}{\geq} \underbrace{\alpha}_{>0} \|u - \tilde{u}\|_H^2 \geq 0 \implies \|u - \tilde{u}\|_H^2 = 0,$$

also folgt $u = \tilde{u}$.

(iii) Existenz: Die Existenz einer Lösung weisen wir über das Funktional nach.

$$\begin{aligned} J(v) &= \frac{1}{2}a(v, v) - F(v) \stackrel{\substack{a \text{ koerziv} \\ F \text{ linear}}}{\geq} \frac{1}{2}\alpha\|v\|_H^2 - c\|v\|_H \\ &= \frac{1}{2}\alpha \left(\|v\|_H^2 - \frac{2c}{\alpha}\|v\|_H \right) = \frac{1}{2}\alpha \left(\|v\|_H - \frac{c}{\alpha} \right)^2 - \frac{c^2}{2\alpha} \\ &\geq -\frac{c^2}{2\alpha} \end{aligned}$$

Folglich ist J nach unten beschränkt. Sei $\eta := \inf\{J(v) \mid v \in H\}$ und $(v_n)_{n \in \mathbb{N}}$ eine Folge mit $J(v_n) \rightarrow \eta$ für $n \rightarrow \infty$, dann folgt mit der Koerzivität von a

$$\begin{aligned} \alpha\|v_n - v_m\|_H^2 &\leq a(v_n - v_m, v_n - v_m) \\ &= a(v_n, v_n) + a(v_m, v_m) - a(v_n, v_m) - a(v_m, v_n) \\ &= 2a(v_n, v_n) + 2a(v_m, v_m) - \underbrace{a(v_n, v_n + v_m) + a(v_m, v_n + v_m)}_{=-a(v_n+v_m, v_n+v_m)} \\ &= 2a(v_n, v_n) - 4F(v_n) + 2a(v_m, v_m) - 4F(v_m) \\ &\quad - a(v_n + v_m, v_n + v_m) + 4F(v_n + v_m) \\ &= 4J(v_n) + 4J(v_m) - 4a\left(\frac{v_n + v_m}{2}, \frac{v_n + v_m}{2}\right) + 8F\left(\frac{v_n + v_m}{2}\right) \\ &= 4J(v_n) + 4J(v_m) - 8J\left(\frac{v_n + v_m}{2}\right) \\ &\leq 4J(v_n) + 4J(v_m) - 8\eta \xrightarrow{n, m \rightarrow \infty} 4\eta + 4\eta - 8\eta = 0, \end{aligned}$$

d.h. $(v_n)_{n \in \mathbb{N}}$ ist eine Cauchy-Folge. Da H ein Hilbertraum ist, gilt somit: $\exists u \in H : v_n \xrightarrow{n \rightarrow \infty} u$ mit $J(u) = \eta$. \square

Um die allgemeine Aussage aus dem Theorem von Lax-Milgram auf unser Modellproblem (2.6) übertragen zu können, benötigen wir die *Poincaré-Friedrich-Ungleichung*, die auch später noch eine zentrale Rolle für den hierarchischen Fehlerschätzer spielen wird.

Satz 2.13 (Poincaré-Friedrich-Ungleichung). *Es sei Ω in einem d -dimensionalen Würfel der Kantenlänge $s > 0$ enthalten. Dann gilt*

$$\|v\|_0 \leq s\|\nabla v\|_0 \quad \forall v \in H_0^1(\Omega),$$

wobei $\|\cdot\|_0$ die durch das Skalarprodukt $(\cdot, \cdot)_0$ induzierte Norm ist.

Beweis. Der Beweis ist in [Bra13] Kapitel II, §1 Sobolev-Räume, Satz 1.5 oder [Sta08] Satz 1.5 zu finden. \square

Bemerkung 2.14. Für die Gültigkeit der Poincaré-Friedrich-Ungleichung muss v nicht auf ganz Γ gleich Null sein, sondern es reicht aus, dass

$$v \in H_{\Gamma_u}^1(\Omega) := \{v \in H^1(\Omega) \mid v = 0 \text{ auf } \Gamma_u\}$$

mit $\Gamma_u \subset \Gamma$ ist, wobei mit einem Maß μ gilt: $\mu(\Gamma_u) \neq 0$, d.h. Γ_u ist keine Nullmenge (vgl. [Bra13] Kapitel II, §1, Bemerkung 1.6).

Jetzt können wir mittels Theorem 2.12 überprüfen, ob Problem (2.6) mit $a : (H_0^1(\Omega))^2 \rightarrow \mathbb{R}, a(u, v) = (\nabla u, \nabla v)_0$ und $F : H_0^1(\Omega) \rightarrow \mathbb{R}, F(v) := (f, v)$ eine eindeutige Lösung hat. Es seien $u, v \in H_0^1(\Omega)$, dann gilt

$$\begin{aligned} a(v, v) &= \int_{\Omega} \nabla v \nabla v \, dx = \|\nabla v\|_0^2 \\ &\geq \frac{s^2 + 1}{(1+s)^2} \|\nabla v\|_0^2 \stackrel{\text{Satz 2.13}}{\geq} \frac{1}{(1+s)^2} (\|v\|_0^2 + \|\nabla v\|_0^2) \\ &= \frac{1}{(1+s)^2} \|v\|_1^2. \end{aligned}$$

Damit ist a mit $\alpha := \frac{1}{(1+s)^2}$ koerativ. Weiter rechnen wir unter Verwendung der Cauchy-Schwarz-Ungleichung nach:

$$\begin{aligned} |a(u, v)| &= \left| \int_{\Omega} \nabla u \nabla v \, dx \right| \leq \sum_{i=1}^d \int_{\Omega} |\partial_i u| |\partial_i v| \, dx \\ &\stackrel{\text{C.S.}}{\leq} \sum_{i=1}^d \left(\int_{\Omega} |\partial_i u|^2 \, dx \right)^{\frac{1}{2}} \left(\int_{\Omega} |\partial_i v|^2 \, dx \right)^{\frac{1}{2}} \\ &\leq \left(\sum_{i=1}^d \int_{\Omega} |\partial_i u|^2 \, dx \right)^{\frac{1}{2}} \left(\sum_{i=1}^d \int_{\Omega} |\partial_i v|^2 \, dx \right)^{\frac{1}{2}} \\ &\leq \left(\int_{\Omega} |\nabla u|^2 \, dx + \int_{\Omega} u^2 \, dx \right)^{\frac{1}{2}} \left(\int_{\Omega} |\nabla v|^2 \, dx + \int_{\Omega} v^2 \, dx \right)^{\frac{1}{2}} \\ &= \|u\|_1 \|v\|_1, \end{aligned}$$

d.h. a ist stetig mit $c := 1$. Die Symmetrie von a ist trivial, also bleibt nur noch die Stetigkeit von F zu zeigen. Es sei $v \in H_0^1(\Omega)$, dann gilt

$$\begin{aligned} |F(v)| &= |(f, v)| = \left| \int_{\Omega} f v \, dx \right| \stackrel{\text{C.S.}}{\leq} \left(\int_{\Omega} |f|^2 \, dx \right)^{\frac{1}{2}} \left(\int_{\Omega} |v|^2 \, dx \right)^{\frac{1}{2}} \\ &\leq c \left(\int_{\Omega} \underbrace{|\nabla v|^2}_{\geq 0} + |v|^2 \, dx \right)^{\frac{1}{2}} = c \|v\|_1 \end{aligned}$$

mit $0 < c := (\int_{\Omega} |f|^2 \, dx)^{\frac{1}{2}} < \infty$, wenn $f \in L_2(\Omega)$ ist. Damit ist F ein stetiges lineares Funktional und somit existiert nach Theorem 2.12 eine eindeutige Lösung $u \in H_0^1(\Omega)$ für die schwache Formulierung des homogenen Dirichlet-Problems.

Weiter minimiert die Lösung $u \in H_0^1(\Omega)$ auch das Funktional

$$J(v) = \frac{1}{2} \int_{\Omega} \nabla v \nabla v \, dx - \int_{\Omega} f v \, dx,$$

welches die gespeicherte Energie der durch die Kraft f belasteten, in Ω eingespannten Membran beschreibt.

Bemerkung. Die Stetigkeit der Linearform F zeigt, dass die Kraft f aus dem Dirichlet-Problem wenigstens quadratisch integrierbar, also in $L^2(\Omega)$, sein muss, damit es eine schwache Lösung geben kann.

Notation. (a) Mit H' bezeichnen wir den Dualraum zum Hilbertraum H .

(b) Den Dualraum zu $H^1(\Omega)$ bezeichnen wir mit $H^{-1}(\Omega)$.

Als Folgerung aus dem Theorem von Lax-Milgram betrachten wir den nächsten Satz.

Satz 2.15 (Riesz'scher Darstellungssatz). *Es sei H ein Hilbertraum mit einem Skalarprodukt $(\cdot, \cdot)_H$. Es sei $F \in H'$, dann existiert genau ein $u \in H$, so dass*

$$(u, v)_H = F(v) \quad \forall v \in H.$$

Beweis. Dies ist eine direkte Folgerung aus dem Theorem 2.12. Die Abbildung $(\cdot, \cdot)_H : H \times H \rightarrow \mathbb{R}$ ist als Skalarprodukt bilinear, symmetrisch und positiv definit, damit auch bzgl. der auf H durch das Skalarprodukt induzierten Norm $\|v\|_H := \sqrt{(v, v)_H}$, koerziv. F ist als Element des Dualraumes H' eine lineare, stetige Abbildung $F : H \rightarrow \mathbb{R}$ und damit folgt mit $a(\cdot, \cdot) := (\cdot, \cdot)_H$ aus dem Theorem von Lax-Milgram die Behauptung. \square

Lemma 2.16. *Es sei H ein Hilbertraum mit Skalarprodukt $(\cdot, \cdot)_H$ und $a : H \times H \rightarrow \mathbb{R}$ eine stetige koerzive Bilinearform. Dann existiert genau ein linearer Operator $A : H \rightarrow H$, so dass gilt:*

$$a(u, v) = (Au, v)_H \quad \forall u, v \in H.$$

Beweis. Es sei $u \in H$ fest, dann ist $L : H \rightarrow \mathbb{R}$, $L(v) := a(u, v)$ eine lineare Abbildung, die stetig ist, da

$$|L(v)| = |a(u, v)| \stackrel{\text{stetig}}{\leq} c \|u\|_H \|v\|_H = \tilde{c} \|v\|_H$$

mit $0 < \tilde{c} := c \|u\|_H$ gilt. Damit folgt nach dem Darstellungssatz von Riesz, dass es ein eindeutiges $l \in H$ gibt, so dass

$$a(u, v) = L(v) = (l, v)_H \quad \forall v \in H$$

gilt. Da $u \in H$ jedoch beliebig ist, bleibt zu zeigen, dass es ein eindeutiges $A : H \rightarrow H$ gibt, so dass $Au = l$ ist.

Wir zeigen zunächst mithilfe der Bilinearform a , dass A linear ist. Es gilt für $\lambda, \mu \in \mathbb{R}$ und $u, v \in H$

$$\begin{aligned} (A(\lambda u + \mu v), w)_H &= a(\lambda u + \mu v, w) = \lambda a(u, w) + \mu a(v, w) \\ &= \lambda(Au, w)_H + \mu(Av, w)_H \\ &= (\lambda Au + \mu Av, w)_H \end{aligned}$$

für alle $w \in H$. Weiter gilt

$$\|Au\|_H^2 = (Au, Au)_H = a(u, Au) \stackrel{\text{stetig}}{\leq} c \|u\|_H \|Au\|_H,$$

d.h. $\|Au\|_H \leq c \|u\|_H$ und damit ist nach [Wer11] Satz II.1.2 (oder vgl. auch [Kö00] Kapitel 1.3) der Operator A stetig.

Betrachten wir den Kern von A , so ergibt sich

$$\ker A := \{v \in H \mid Av = 0\} = \{0\}, \quad (2.8)$$

denn

$$\alpha \|v\|_H^2 \stackrel{\text{koerziv}}{\leq} a(v, v) = (Av, v)_H \stackrel{\text{CS}}{\leq} \|Av\|_H \|v\|_H$$

und damit gilt $\|Av\|_H \geq \alpha \|v\|_H$, d.h. $Av = 0 \Leftrightarrow v = 0$. Dies impliziert, dass A injektiv ist, denn mit $v_1, v_2 \in H$, $Av_1 = Av_2$ folgt

$$0 = Av_1 - Av_2 = A(v_1 - v_2) \xrightarrow{(2.8)} v_1 = v_2.$$

Weiter betrachten wir das Bild von A , d.h.

$$\text{im } A := \{v \in H \mid \exists u \in H : Au = v\} \subset H.$$

Sei $(v_n)_{n \in \mathbb{N}}$ eine Folge mit $v_k \in \text{im}(A)$ für alle $k \in \mathbb{N}$. Dann folgt, dass für jedes v_k ein $u_k \in H$ existiert mit $Au_k = v_k$. Es gelte, dass $Au_k = v_k \rightarrow v \in H$ geht, dann folgt

$$\begin{aligned} \alpha \|u_n - u_m\|_H &\leq \|A(u_n - u_m)\|_H = \|Au_n - Au_m\|_H \\ &= \|v_n - v_m\|_H \xrightarrow{n, m \rightarrow \infty} 0, \end{aligned}$$

d.h. $(u_n)_{n \in \mathbb{N}} \subset H$ ist eine Cauchy-Folge und konvergiert daher in H . Also existiert ein $u \in H$ mit $u_n \rightarrow u$. Mit der Stetigkeit von A folgt dann

$$v_n = Au_n \xrightarrow{n \rightarrow \infty} Au = v,$$

d.h. $v \in \text{im}(A)$ und damit ist $\text{im}(A)$ abgeschlossen. Wir betrachten nun ein $v \in H$ mit $v \perp \text{im } A \subset H$, dann gilt

$$(Au, v)_H = 0 \quad \forall u \in H.$$

Damit folgt mit $u = v \in H$ oben eingesetzt

$$0 = (Av, v)_H = a(v, v) \geq \alpha \|v\|_H^2 \implies v = 0.$$

Also besteht der zu $\text{im}(A)$ orthogonale Raum nur aus dem Nullelement und mit Korollar 2.7 gilt dann $\text{im } A = \overline{\text{im } A} = H$. Damit ist A bijektiv.

Es seien nun $0 \neq l \in H$ sowie $A_1, A_2 \in \mathcal{L}(H, H)$ zwei lineare Operatoren mit $A_1 u = l$ und $A_2 u = l$, die nach der obigen Weise konstruiert sind. Dann gilt

$$0 = A_1 u - A_2 u = (A_1 - A_2)u \implies A_1 = A_2,$$

da $u \neq 0$ und die Summe zweier bijektiver linearer Operatoren wieder bijektiv ist, also ist ein so konstruierter Operator eindeutig. \square

2.3 Finite Elemente Methode

Für unser Modellproblem kann man zeigen, dass es für bestimmte Gebiete Ω eine exakte, d.h. analytische, Lösung gibt (vgl. [Wal11] Kapitel 5). Solch eine Lösung muss im Allgemeinen nicht für jedes Problem bekannt oder gar berechenbar sein. Daher wollen wir nicht mehr die exakte Lösung von unserer Variationsgleichung (2.6) berechnen, sondern eine Approximation davon, die sogenannte *Galerkin-Approximation*.

Unter dem *Galerkin-Verfahren* verstehen wir, dass wir die Variationsgleichung

$$a(u, v) = F(v) \quad \forall v \in H \tag{2.9}$$

nur noch auf einem endlich dimensionalen Unterraum $V_h \subset H$ lösen wollen, d.h. finde $u_h \in V_h$, so dass

$$a(u_h, v_h) = F(v_h) \quad \forall v_h \in V_h. \tag{2.10}$$

Satz 2.17. *Das Galerkin-Verfahren (2.10) hat eine eindeutige Lösung.*

Beweis. Da V_h als Unterraum von H auch ein Hilbertraum ist und die Eigenschaften von a, F weiterhin erfüllt sind, gilt auch hier der Satz von Lax-Milgram, was die Eindeutigkeit und Existenz einer Lösung sichert. \square

Da V_h ein endlich dimensionaler Unterraum von H ist, wird jener von einer endlichen Basis $\mathcal{B}_h := \{\phi_1, \dots, \phi_N\}$ aufgespannt, d.h. für $u_h \in V_h$ gilt:

$$\exists! \boldsymbol{\mu} \in \mathbb{R}^N : u_h(x) = \sum_{i=1}^N \mu_i \phi_i(x). \quad (2.11)$$

Da $F(\cdot), a(u, \cdot)$ linear sind und alle $v_h \in V_h$ wie in (2.11) darstellbar sind, ist das Galerkin-Verfahren (2.10) äquivalent zum Problem

$$a(u_h, \phi_i) = F(\phi_i) \quad \forall i = 1, \dots, N,$$

mit u_h wie in (2.11) beschrieben. Ersetzen wir u_h in der Bilinearform durch (2.11), so ergibt sich

$$a(u_h, \phi_i) = a\left(\sum_{j=1}^N \mu_j \phi_j, \phi_i\right) = \sum_{j=1}^N \mu_j a(\phi_j, \phi_i),$$

also

$$\sum_{j=1}^N \mu_j a(\phi_j, \phi_i) = F(\phi_i) \quad \forall i = 1, \dots, N.$$

Damit erhalten wir ein lineares Gleichungssystem

$$A\boldsymbol{\mu} = \mathbf{f}$$

mit $A = [a(\phi_j, \phi_i)]_{i,j=1}^N$, $\boldsymbol{\mu} = [\mu_i]_{i=1}^N$ und $\mathbf{f} = [F(\phi_i)]_{i=1}^N$. Dieses Gleichungssystem gilt es zu lösen, um die gesuchten Koordinaten $\mu_i, i = 1, \dots, N$, bzgl. der Basis \mathcal{B}_h für die approximierte Lösung u_h zu finden.

In den Ingenieurwissenschaften, insbesondere bei kontinuumsmechanischen Problemen, wird A als *Steifigkeitsmatrix* bezeichnet.

Bemerkung 2.18. Ist die Bilinearform a symmetrisch, so ist es auch die Matrix A , denn

$$a_{ij} = a(\phi_i, \phi_j) = a(\phi_j, \phi_i) = a_{ji}.$$

Außerdem folgt aus der Koeffizienten von a , dass mit $\mathbf{0} \neq \boldsymbol{\nu} \in \mathbb{R}^N$ gilt

$$\begin{aligned} \boldsymbol{\nu}^T A \boldsymbol{\nu} &= \sum_{i,j=1}^N \nu_i a_{ij} \nu_j = \sum_{i=1}^N \nu_i \sum_{j=1}^N a(\phi_i, \phi_j) \nu_j \\ &= \sum_{i=1}^N \nu_i a\left(\phi_i, \sum_{j=1}^N \nu_j \phi_j\right) = a\left(\sum_{i=1}^N \nu_i \phi_i, \sum_{j=1}^N \nu_j \phi_j\right) \\ &= a(v_h, v_h) \geq \alpha \|v_h\|_H^2 > 0, \end{aligned}$$

da $\sum \nu_i \phi_i = v_h \neq 0$ wegen $\boldsymbol{\nu} \neq \mathbf{0}$. Damit ist A also positiv definit und es folgt nochmals, dass $A\boldsymbol{\mu} = \mathbf{f}$ eine eindeutige Lösung hat.

Um eine Basis \mathcal{B}_h bzgl. V_h beschreiben zu können, muss das Gebiet $\Omega \subset \mathbb{R}^2$ in endliche Elemente zerlegt werden. Die Basis \mathcal{B}_h und damit der Raum V_h wird dann bzgl. einer Zerlegung \mathcal{T}_h beschrieben. Eine gebräuchliche Zerlegung \mathcal{T}_h kann durch Dreiecke oder auch Vierecke geschehen. Wir wollen in der vorliegenden Arbeit nur Zerlegungen durch Dreiecke betrachten, hierfür führen wir den folgenden Begriff ein (vgl. [Bra13] Seite 58 oder [Sta08] Seite 19).

Definition 2.19 (Triangulierung). Es sei $\Omega \subset \mathbb{R}^2$ ein durch einen Polygonzug berandetes Gebiet. Dann heißt eine Zerlegung aus Dreiecken

$$\mathcal{T} = \{T_1, T_2, \dots, T_M\}$$

Triangulierung, wenn gilt:

- (a) Für alle Dreiecke $T \in \mathcal{T}$ gilt: T ist abgeschlossen.
- (b) Ganz Ω wird durch alle Dreiecke aus \mathcal{T} überdeckt, d.h. $\bar{\Omega} = \bigcup_{T \in \mathcal{T}} T$.
- (c) Der Schnitt zweier Dreiecke $T_i \cap T_j$ mit $i \neq j$ überlappt sich nicht, d.h. $\text{int}(T_i) \cap \text{int}(T_j) = \emptyset$.

Wir nennen eine Triangulierung *konform* oder *zulässig*, wenn zusätzlich gilt:

- (d) Für jede Kante k eines Dreiecks $T \in \mathcal{T}$ gilt entweder $k \subset \partial\Omega$ oder $k = \tilde{k}$ für eine weitere Kante \tilde{k} eines weiteren Dreiecks $\tilde{T} \in \mathcal{T}$.

Der Radius des Umkreises eines Dreiecks T wird mit h bezeichnet und beschreibt die Größe eines Dreiecks. Wenn jedes Dreieck $T \in \mathcal{T}$ höchstens einen Radius von h hat, so schreiben wir \mathcal{T}_h statt \mathcal{T} .

Bemerkung 2.20. (a) Ein Dreieck $T \in \mathcal{T}_h$ bezeichnen wir auch als (*finites*) *Element*.

- (b) Sollte $\Omega \subset \mathbb{R}^3$ sein, so können wir analog zu Definition 2.19 eine Zerlegung mit Tetraedern definieren.

Für die Netzverfeinerung führen wir zwei unterschiedliche Familien von Zerlegungen ein (vgl. [Bra13] Seite 58).

Definition 2.21 ((quasi-) uniforme Zerlegung). Eine Familie von Zerlegungen $\{\mathcal{T}_h\}$ heißt *quasi-uniform*, wenn es eine Zahl $\kappa > 0$ gibt, so dass jedes $T \in \mathcal{T}_h$ einen Kreis vom Radius

$$\rho_T \geq \frac{h_T}{\kappa}$$

enthält, wobei h_T der Radius des Dreiecks T ist.

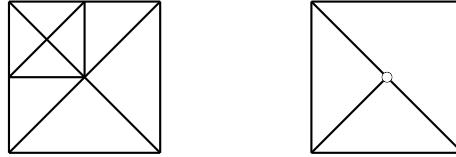


Abbildung 2.2: Zulässige und unzulässige Triangulierung (mit hängendem Knoten)

Eine Familie von Zerlegungen $\{\mathcal{T}_h\}$ heißt *uniform*, wenn es eine Zahl $\kappa > 0$ gibt, so dass jedes $T \in \mathcal{T}_h$ einen Kreis vom Radius

$$\rho_T \geq \frac{h}{\kappa}$$

enthält, wobei $h := \max_{T \in \mathcal{T}_h} h_T$ ist.

In der Abbildung 2.2 ist eine quasi-uniforme zulässige Zerlegung und eine unzulässige Zerlegung zu sehen; zweitere ist daher unzulässig, da es Kanten gibt, die weder am Rand liegen, noch mit einer Kante eines anderen Dreiecks übereinstimmen. Die Knoten, die zu diesem Phänomen führen, nennen wir *hängende Knoten*. Dies sind Knoten, die nicht Eckpunkt jedes angrenzenden Dreiecks sind.

Bemerkung 2.22. Wie man leicht sehen kann, ist jede uniforme Zerlegung auch quasi-uniform. Umgekehrt gilt dies nicht (s. Abbildung 2.2).

Allerdings lassen uniforme Zerlegungen keine lokalen Verfeinerungen zu. Da dies für adaptive Verfeinerungsstrategien allerdings ausschlaggebend ist, gehen wir im Folgenden immer von einer quasi-uniformen Zerlegung \mathcal{T}_h aus.

Nun wollen wir uns Gedanken über unseren Ansatzraum V_h machen. Hierfür gibt es, abhängig von der Konstruktion des verwendeten Elements, viele Möglichkeiten – vgl. hierzu auch [Bra13] Kapitel II, §5, Tabelle 2. Wir wollen uns weitestgehend aber nur auf ein Element konzentrieren. Zuvor betrachten wir hierfür ein wichtiges Resultat, wobei noch bemerkt sei, dass eine Funktion u auf Ω bei gegebener Zerlegung \mathcal{T}_h eine Eigenschaft stückweise hat, wenn sie auf jedem Element diese Eigenschaft besitzt.

Satz 2.23. Es sei $k \geq 1$ und $\Omega \subset \mathbb{R}^2$ ein polygonales Gebiet. Eine stückweise beliebig oft differenzierbare Funktion $v : \bar{\Omega} \rightarrow \mathbb{R}$ liegt in $H^k(\Omega)$ genau dann, wenn $v \in C^{k-1}(\bar{\Omega})$ ist.

Beweis. Der Beweis ist in [Bra13] Kapitel II, §5, Satz 5.2 zu finden. □

Der Satz 2.23 rechtfertigt, dass wir für das Modellproblem (2.6) auf einer Triangulierung \mathcal{T}_h einen Ansatzraum V_h mit stetigen Funktionen $v \in C^0(\Omega)$ verwenden, da dann auch $v \in H^1(\Omega)$ gilt. Daher wählen wir

$$V_h := \{v \in C^0(\Omega) \mid v|_T \in \mathcal{P}_m \text{ für } T \in \mathcal{T}_h, v|_{\partial\Omega} = 0\},$$

wobei \mathcal{P}_m der Raum der Polynome vom Grad m ist. Es stellt sich nun die Frage, wie wir geschickt eine Basis wählen können, um V_h aufzuspannen. Die einfachste Möglichkeit stellen *nodale Basisfunktionen* dar.

Definition 2.24 (nodale Basisfunktion). Zu einem Finiten-Element-Raum V_h und einer gegebenen Zerlegung \mathcal{T}_h sei eine Menge von Punkten P bekannt mit $|P| = N$. Die Menge $\mathcal{B}_h = \{\phi_1, \dots, \phi_N\}$ mit $\phi_i \in \mathcal{P}_m, i = 1, \dots, N$, heißt *nodale Basis* (oder *Lagrange-Basis*), wenn

$$\phi_i(x_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases},$$

für alle $\phi_i \in \mathcal{B}_h$ und $x_j \in P$ gilt.

Mit der Menge der vorgegebenen Punkte P kann durch die Anzahl N der Grad des zur Interpolation verwendeten Polynoms gesteuert werden.

Bemerkung 2.25. Sei $m \geq 0$. In einem Dreieck T seien auf $m+1$ Linien $l = 1+2+\dots+(m+1)$ Punkte z_1, \dots, z_l angeordnet (s. Abb. 2.3). Dann gibt es zu jedem $C^0(T)$ genau ein Polynom p vom Grad m mit der Eigenschaft

$$p(z_i) = f(z_i) \quad \forall i = 1, \dots, m.$$

Beweis. Der Beweis ist in [Bra13] Kapitel II, §5, Bemerkung 5.4 zu finden. \square

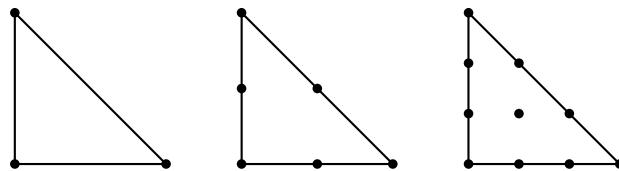


Abbildung 2.3: Dreiecke für nodale Basen (linear, quadratisch, kubisch)

Damit lässt sich für V_h mit einem beliebigen Polynomgrad m eine eindeutige nodale Basis finden, die den Raum aufspannt. Im Weiteren wollen wir lineare Ansatzfunktionen verwenden. Sofern nicht anders beschrieben bezeichnen wir im Folgenden \mathcal{S}_h mit

$$\mathcal{S}_h := \{v \in C^0(\Omega) \mid v|_T \in \mathcal{P}_1 \text{ für } T \in \mathcal{T}_h, v|_{\partial\Omega} = 0\},$$

d.h. in diesem Raum sind die Eckpunkte der Dreiecke vorgegeben bzw. später im LGS gesucht. Das Galerkin-Verfahren mit dem Ansatzraum \mathcal{S}_h wird auch Finite-Elemente-Methode (kurz: FEM) genannt.

Folgendes Beispiel soll zur Erläuterung der Berechnung von den Matrixeinträgen der Steifigkeitsmatrix A dienen.

Beispiel 2.26. Wir betrachten das Variationsproblem (2.10) auf $\Omega = [-1, 1]^2$ mit S_h wie oben eingeführt als den Raum der linearen Ansatzfunktionen auf einer Zerlegung \mathcal{T}_h aus 8 *Courant-Elementen*, wie in Abbildung 2.4 zu sehen ist, wobei wir auf die rechte Seite $F(v_h)$ zunächst noch nicht genauer eingehen möchten.

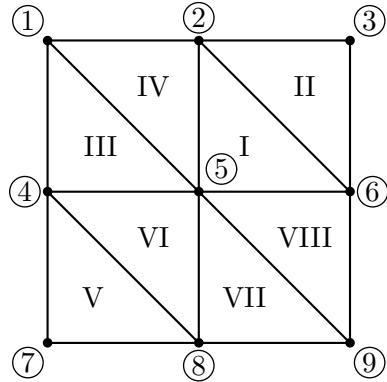


Abbildung 2.4: Triangulierung von $\Omega = [-1, 1]^2$ in 8 Courant-Elemente

Wir stellen für die nodale Basisfunktion ϕ_5 die Einträge in der Steifigkeitsmatrix A auf. Man rechnet leicht nach, dass

$$\phi_5(x, y) = \begin{cases} 1 - x - y, & \text{auf I} \\ 1 + x, & \text{auf III} \\ 1 - y, & \text{auf IV} \\ 1 + x + y, & \text{auf VI} \\ 1 + y, & \text{auf VII} \\ 1 - x, & \text{auf VIII} \\ 0, & \text{sonst} \end{cases}$$

	I	II	III	IV	V	VI	VII	VIII
$\partial_x \phi_5$	-1	0	1	0	0	1	0	-1
$\partial_y \phi_5$	-1	0	0	-1	0	1	1	0

Tabelle 2.1: Ableitungen der nodalen Basisfunktion ϕ_5 .

ist und es ergeben sich die Ableitungen aus Tabelle 2.1. Dann gilt

$$\begin{aligned}
 a(\phi_5, \phi_5) &= \int_{\Omega} \nabla \phi_5 \nabla \phi_5 \, dx dy = \int_{I \cup \dots \cup VIII} \underbrace{(\partial_x \phi_5)^2}_{\geq 0} + \underbrace{(\partial_y \phi_5)^2}_{\geq 0} \, dx dy \\
 &= 2 \int_{I \cup III \cup IV} (\partial_x \phi_5)^2 + (\partial_y \phi_5)^2 \, dx dy \\
 &= 2 \left(\int_{I \cup III} \underbrace{(\partial_x \phi_5)^2}_{=1} \, dx dy + \int_{I \cup IV} \underbrace{(\partial_y \phi_5)^2}_{=1} \, dx dy \right) \\
 &= 2(\mathcal{A}(I) + \mathcal{A}(III) + \mathcal{A}(I) + \mathcal{A}(IV)) \\
 &= 8 \cdot \mathcal{A}(I) = 8 \cdot \frac{1}{2} = 4,
 \end{aligned}$$

wobei verwendet wurde, dass die Dreiecke kongruent zueinander sind und $\mathcal{A}(\cdot)$ den Flächeninhalt eines Dreiecks berechnet. Analog können wir auch die übrigen acht nodalen Basisfunktionen aufstellen und damit die Einträge der Steifigkeitsmatrix

$$\begin{aligned}
 a(\phi_5, \phi_2) &= a(\phi_5, \phi_4) = a(\phi_5, \phi_6) = a(\phi_5, \phi_8) = -1, \\
 a(\phi_5, \phi_1) &= a(\phi_5, \phi_3) = a(\phi_5, \phi_7) = a(\phi_5, \phi_9) = 0
 \end{aligned}$$

berechnen. Damit ist der Anteil der Basisfunktion ϕ_5 an der Steifigkeitsmatrix A von der Form

$$\tilde{A} = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}.$$

Hierbei müssen die Einträge aus \tilde{A} in die Matrix $A \in \mathbb{R}^{9 \times 9}$ an die richtige Stelle zugeordnet werden, wie durch die Formel $a_{ij} = a(\phi_i, \phi_j)$ beschrieben wird. Daher nennen wir \tilde{A} *lokale Steifigkeitsmatrix* bzgl. des Knoten 5.

Dieses Vorgehen müssten wir noch für die übrigen Basisfunktionen analog durchführen, um die vollständige Steifigkeitsmatrix A zu erhalten. Dies soll hier aber nicht weiter ausgeführt werden.

Wie man sieht, ist das Vorgehen aus Beispiel 2.26 sehr aufwendig. Außerdem ist es schwer jenes auf diese Weise zu verallgemeinern, damit man es gut implementieren kann, da die Ansatzfunktionen auf das Gitter bezogen von individueller Form sind.

Um die Berechnung der Einträge der Steifigkeitsmatrix A zu verallgemeinern, betrachten wir das Referenzelement

$$\tilde{T} := \{(\xi, \eta) \in \mathbb{R}^2 \mid 0 \leq \xi \leq 1, 0 \leq \eta \leq 1 - \xi\}.$$

Auf diesem können wir dann die Integrale für die Ansatzfunktionen lokal berechnen, um dann die berechneten Werte affin auf ein beliebiges Dreieck

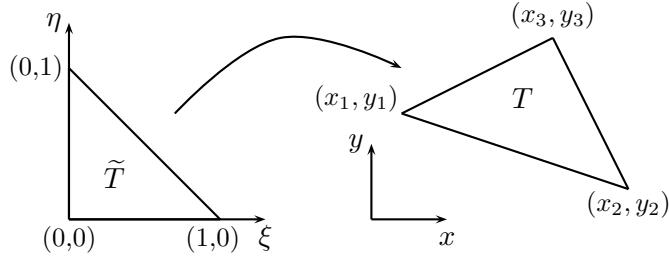


Abbildung 2.5: Referenzelement \tilde{T} für ein allgemeines Dreieck $T \in \mathcal{T}_h$

T zu transformieren (s. Abbildung 2.5). Diese auf das Element T bezogene lokale Steifigkeitsmatrix müssen wir dann durch *global-local node ordering* in die globale Steifigkeitsmatrix A assemblieren. Die genaue Berechnungsvorschrift für das oben beschriebene Vorgehen wird in Kapitel 5 noch mal genauer hergeleitet.

2.3.1 A priori Fehlerabschätzung

Wir wollen nun zeigen, dass durch Netzverfeinerung, d.h. Verkleinern von h , der Fehler zwischen der exakten Lösung u und der Galerkin-Approximation u_h kleiner wird.

Lemma 2.27. Durch $\|\cdot\|_E : H_0^1(\Omega) \rightarrow \mathbb{R}$, $\|v\|_E := (a(v, v))^{\frac{1}{2}}$ mit einer stetigen koerziven Bilinearform a wird eine Norm auf $H_0^1(\Omega)$ definiert.

Beweis. Aus der Stetigkeit und Koerzivität von a folgt direkt

$$\alpha \|v\|_1^2 \leq \underbrace{a(v, v)}_{=\|v\|_E^2} \leq c \|v\|_1^2. \quad (2.12)$$

Damit ist $\|\cdot\|_E$ nach oben und unten durch die Norm auf $H_0^1(\Omega)$ beschränkt und somit eine zu dieser äquivalente Norm. \square

Bemerkung. (a) Die Norm $\|\cdot\|_E$ bezeichnen wir als *Energie-Norm*. Sie gibt für die von uns später in der Strukturmechanik betrachtete Bilinearform die Verzerrungsenergie eines Kontinuums an.

(b) Für die Bilinearform

$$a(u, v) = \int_{\Omega} \nabla u \nabla v \, dx$$

mit $u, v \in H_0^1(\Omega)$ gilt dann $\|\cdot\|_E = |\cdot|_1$ (s. Bemerkung A.6).

(c) Auf $H^1(\Omega)$ wäre $\|\cdot\|_E$ also nur eine Halbnorm, da konstante Funktionen $v = c$ auch die Norm $\|v\|_E = 0$ hätten.

Satz 2.28. Die Galerkin-Approximation u_h ist die beste Approximation von u bzgl. der Energie-Norm, also

$$\|u - u_h\|_E = \inf_{v \in V_h} \|u - v\|_E.$$

Beweis. Zunächst betrachten wir die exakte und approximierte Variationsgleichung (2.9) und (2.10), d.h. finde $u \in H$ und $u_h \in V_h$, so dass

$$a(u, v) = F(v) \quad \forall v \in H, \quad (2.13)$$

$$a(u_h, v_h) = F(v_h) \quad \forall v_h \in V_h. \quad (2.14)$$

Da $V_h \subset H$ ist, gilt (2.13) auch für alle $v_h \in V_h$. Ersetzen wir dies in (2.13) und subtrahieren (2.13) und (2.14) voneinander, so erhalten wir

$$a(u - u_h, v_h) = 0 \quad \forall v_h \in V_h. \quad (2.15)$$

Damit rechnen wir für ein beliebiges $v \in V_h$ einfach nach:

$$\begin{aligned} \|u - u_h\|_E^2 &= a(u - u_h, u - u_h) \\ &= a(u - u_h, u - v + v - u_h) \\ &= a(u - u_h, u - v) + a(u - u_h, \underbrace{v - u_h}_{\in V_h}) \\ &\quad \underbrace{= 0 \text{ wegen (2.14)}}_{\text{C.S.}} \\ &= a(u - u_h, u - v) \\ &\leq \|u - u_h\|_E \|u - v\|_E \end{aligned}$$

und damit folgt nach Division $\|u - u_h\|_E \leq \|u - v\|_E$, was zu zeigen war. \square

Bemerkung. Die Gleichung (2.15) drückt aus, dass die Verbindung $u - u_h$ orthogonal zum Raum V_h steht und wird daher auch *Galerkin-Orthogonalität* genannt. Diese wird bei Hindernisproblemen im Allgemeinen nicht mehr erfüllt, daher führen Hindernisprobleme nicht mehr auf Variationsgleichungen, wie wir später sehen werden.

Satz 2.29 (Céa). Der Fehler der Galerkin-Approximation u_h zur exakten Lösung u hat in der H^1 -Norm die Eigenschaft

$$\|u - u_h\|_1 \leq \tilde{c} \inf_{v \in V_h} \|u - v\|_1.$$

Beweis. Aus (2.12) und Satz 2.28 folgt

$$\|u - u_h\|_1 \leq \left(\frac{1}{\alpha}\right)^{\frac{1}{2}} \|u - u_h\|_E \leq \left(\frac{1}{\alpha}\right)^{\frac{1}{2}} \|u - v\|_E \leq \left(\frac{c}{\alpha}\right)^{\frac{1}{2}} \|u - v\|_1.$$

Damit folgt die Behauptung mit $\tilde{c} := \sqrt{\frac{c}{\alpha}}$. \square

Nun kommen wir zum zentralen Satz dieses Unterkapitels, mit dem man direkt eine a priori Fehlerschätzung folgern kann. Vergleiche hierzu auch [Bra13] Kapitel II, §6, Satz 6.4.

Theorem 2.30 (Approximationssatz für Interpolationen). *Es sei $k \geq 2$ und \mathcal{T}_h eine quasi-uniforme Triangulierung von Ω . Dann gilt für die Interpolation I_h auf die stetigen, stückweise durch Polynome vom Grad $k-1$ gegebenen Funktionen mit einer von Ω, κ und k abhängigen Konstanten c die a priori Fehlerabschätzung*

$$\|u - I_h u\|_m \leq ch^{k-m}|u|_k$$

für $u \in H^k(\Omega)$ und $0 \leq m \leq k$.

Beweis. Für den Beweis würden wir noch weitere Ausführungen über affine Transformationen benötigen, die wir hier nicht weiter aufführen wollen. Der komplette Beweis ist in [Bra13] auf Seite 75ff einzusehen. \square

Für $k = 2$, d.h. lineare Polynome, und $m = 1$ (die Norm in H^1) gilt dann

$$\|u - I_h u\|_1 \leq ch|u|_2 \quad (2.16)$$

für $u \in H^2(\Omega)$.

Korollar 2.31. *Für lineare C^0 -Elemente gilt bzgl. der Galerkin-Approximation u_h die a priori Fehlerschätzung für unser Modellproblem (DP)*

$$\|u - u_h\|_1 \leq \tilde{c}h|u|_2 .$$

Beweis. Mit Theorem 2.30 und Satz 2.29 folgt

$$\begin{aligned} \|u - u_h\|_1 &\leq \left(\frac{c_1}{\alpha}\right)^{\frac{1}{2}} \inf_{v \in V_h} \|u - v\|_1 \leq \left(\frac{c_1}{\alpha}\right)^{\frac{1}{2}} \|u - I_h u\|_1 \\ &\stackrel{(2.16)}{\leq} \left(\frac{c_1}{\alpha}\right)^{\frac{1}{2}} c_2 h|u|_2 . \end{aligned}$$

Mit $u \in H^2(\Omega)$ und $\tilde{c} := \left(\frac{c_1}{\alpha}\right)^{\frac{1}{2}} c_2$ folgt dann die Behauptung. \square

Mit Korollar 2.31 gilt also, dass für $h \rightarrow 0$ die Galerkin-Approximation u_h gegen die exakte Lösung u konvergiert. Es ist also sinnvoll das Netz zu verfeinern, allerdings bringt jede Verfeinerung auch mehr Knoten und damit ein größeres Gleichungssystem mit sich. Daher stellt sich die Frage, ob es sinnvoll ist, nur einzelne Teile des Gitters zu verfeinern, womit wir uns im Kapitel 2.4 beschäftigen wollen.

2.4 Adaptive Verfeinerungsstrategien

Wir wollen nun betrachten, wie sich das Gitter geschickt verfeinern lässt, so dass die Anzahl der Knoten im Vergleich zur Verringerung des Fehlers hinreichend groß ist. Diese Verfeinerung im n -ten Schritt geschieht adaptiv in Abhängigkeit des aktuellen Gitters \mathcal{T}_{h_n} bzw. der aktuellen Lösung u_{h_n} . Hierbei wird *a posteriori* der Fehler im nächsten Schritt $\|u - u_{h_{n+1}}\|$ mit dem Fehler im aktuellen $\|u - u_{h_n}\|$ verglichen und daraus die notwendige Größe der Verfeinerung abgeschätzt. Da die Lösung u für ein Problem, wie schon beschrieben, nicht bekannt oder berechenbar sein muss, gilt es den oben beschriebenen Fehler zu schätzen. Für solche *a posteriori* Fehlerschätzer gibt es mehrere Ansätze.

2.4.1 A posteriori Fehlerschätzer

Wie auch in [Bra13] Kapitel III, §8, Seite 176 genauer beschrieben, gibt es verschiedene Arten von *a posteriori* Fehlerschätzern:

- (a) Residuale Schätzer
- (b) Schätzung über ein lokales Neumann-Problem
- (c) Schätzung über ein lokales Dirichlet-Problem
- (d) Schätzung durch Mitteilung
- (e) Hierarchische Schätzer

Als Fehlerschätzer für Hindernis- oder auch Kontaktprobleme sind häufig residuale Schätzer zu finden. Wir wollen uns in dieser Arbeit mit hierarchischen Fehlerschätzern beschäftigen.

Die Idee dabei ist, dass wir den Fehler durch eine genauere Lösung aus einem „besseren“ Ansatzraum abschätzen, in dem Sinne, dass für den Ansatzraum V_h , in dem die berechnete Lösung u_h liegt, gilt:

$$V_h \subset W_h \text{ mit } W_h = V_h \oplus Z_h, \quad (2.17)$$

wobei bzgl. der Obermenge W_h die Lösung u_h^W genauer ist als u_h , d.h. die *Saturationseigenschaft*

$$a(u - u_h^W, u - u_h^W) \leq \beta^2 a(u - u_h, u - u_h) \quad (2.18)$$

mit einer Konstanten $0 \leq \beta < 1$ erfüllt.

Bemerkung 2.32. Da wir zur Berechnung unserer Galerkin-Approximation den Raum $V_h = \mathcal{S}_h$ der linearen Ansatzfunktionen verwenden werden, wählen wir später eine Hierarchie

$$W_h = \mathcal{Q}_h := \{v \in C^0(\Omega) \mid v|_T \in \mathcal{P}_2 \text{ für } T \in \mathcal{T}_h, v|_{\partial\Omega} = 0\},$$

den Raum der quadratischen Ansatzfunktionen. Daraus lässt sich dann der Raum Z_h ableiten.

Wir wollen nun aber nicht die bessere Approximation $u_h^W \in W_h$ exakt berechnen, da dies ein zu großer Aufwand wäre, sondern schränken das sogenannte *Defektproblem* lokal auf die Erweiterung Z_h ein, d.h.

$$a(u_h^W, z_h) = F(z_h) \quad \forall z_h \in Z_h, \quad (2.19)$$

wobei in (2.19) u_h^W aufgeteilt werden kann, da W_h aus direkter Summe von V_h, Z_h entsteht, d.h. $u_h^W = u_h + e_h$ mit $u_h \in V_h, e_h \in Z_h$. Da die Lösung u_h in jedem Schritt bekannt ist, lässt sich das lokale Defektproblem (2.19) schreiben als

$$e_h \in Z_h : \quad a(e_h, z_h) = \underbrace{F(z_h) - a(u_h, z_h)}_{= \tilde{F}(z_h)} \quad \forall z_h \in Z_h. \quad (2.20)$$

Man kann zeigen, dass für die Lösung e_h aus (2.20) unter der Bedingung (2.18) der Term $a(e_h, e_h)$ beschränkt ist und daher gibt der auf ein Dreieck T bezogene lokale Anteil $\eta_T := a_T(e_h, e_h)^{\frac{1}{2}}$ den hierarchischen Fehlerschätzer an.

Damit lässt sich $a(e_h, e_h)$ in die lokalen Anteile η_T aufteilen, so dass

$$a(e_h, e_h) = \sum_{T \in \mathcal{T}_h} a_T(e_h, e_h).$$

Daher verwenden wir als adaptive Strategie: Verfeinere alle Dreiecke $T \in \mathcal{T}_h$, deren lokaler Fehleranteil größer gleich dem skalierten Gesamtfehler ist, also

$$\eta_T \geq \sigma \left(\sum_{T \in \mathcal{T}_h} \eta_T^2 \right)^{\frac{1}{2}},$$

wobei $\sigma \in (0, 1)$ ist. Wählen wir σ sehr klein, so werden viele Dreiecke verfeinert, da auch kleinere lokale Fehleranteile die Ungleichung erfüllen. Umgekehrt gilt für ein großes σ , dass man viele Adaptionsschritte benötigt, um einen hinreichend kleinen Fehler zu erhalten, da nur wenige Dreiecke pro Verfeinerungsschritt ausgewählt werden.

Wie die Hierarchie bzgl. der Räume geschickt gewählt werden kann, wollen wir in Kapitel 4 genauer beschreiben und auf Variationsprobleme unter Nebenbedingung anwenden, um einen a posteriori Schätzer herzuleiten.

2.4.2 Verfeinerung des Netzes

Auf der Grundlage des a posteriori Schätzers müssen die ausgewählten Dreiecke verfeinert werden. Dabei ist es essentiell, dass eine konforme Triangulierung auch konform bleibt, was durch Verwendung verschiedenener Verfeinerungsmethoden möglich ist (s. Abbildung 2.6).

- (i) *Rote* (reguläre) Verfeinerung,
- (ii) *Grüne* Verfeinerung,
- (iii) *Blaue* Verfeinerung.

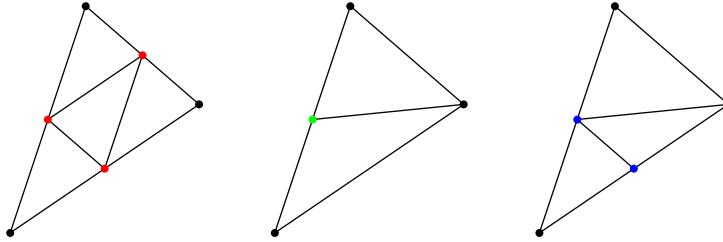


Abbildung 2.6: Verfeinerungen von Dreiecken

Bei der regulären Verfeinerung werden die drei Mittelpunkte der Kanten eines Dreiecks miteinander verbunden. Der Vorteil dabei ist, dass die Winkel der entstandenen Dreiecke identisch zu den vorherigen Winkeln sind. Damit bleibt das Verhältnis zwischen dem Radius des Um- zum Innenkreises $\frac{h_T}{\rho_T}$, was den Parameter für die Quasi-Uniformität angibt, gleich.

Beim regulären Verfeinern können allerdings in anliegenden Dreiecken, die bzgl. des a posteriori Fehlerschätzers nicht verfeinert werden müssen, hängende Knoten (vgl. Abbildung 2.2) entstehen, wodurch eine nichtzulässige Triangulierung entstehen würde. Um die hängenden Knoten zu eliminieren, kann die Verfeinerungsmethode (ii) und (iii) verwendet werden.

Bei der grünen Verfeinerung wird der Mittelpunkt genau einer Kante mit dem gegenüberliegenden Eckpunkt verbunden.

Bei der blauen Verfeinerung verbindet man den Mittelpunkt der längsten Kante im Dreieck mit dem gegenüberliegenden Eckpunkt und einem weiteren Mittelpunkt einer anderen Kante.

Matlab verwendet diese Verfeinerungsstrategien in der Methode *refine-mesh*. Diese Methode werden wir bei der Implementierung der numerischen Beispiele auch verwenden und daher wollen wir das Vorgehen von Matlab kurz vorstellen. Ist $\widetilde{\mathcal{T}}_h$ die Menge der zu verfeinernden Dreiecke, dann werden folgende Schritte durchgeführt:

- (i) Halbiere alle Kanten der ausgewählten Dreiecke $T \in \widetilde{\mathcal{T}}_h$.
- (ii) Halbiere jeweils die längste Kante aller Dreiecke, die schon eine geteilte Kante haben.
- (iii) Bilde die neuen Dreiecke nach folgender Strategie:
 - (a) Wenn alle drei Kanten eines Dreiecks geteilt sind, verwende die reguläre (rote) Verfeinerung.

- (b) Sind genau zwei Seiten halbiert, so verwende die blaue Verfeinerung.
- (c) Ist genau eine Kante eines Dreiecks halbiert, dann benutze die grüne Verfeinerung.

2.5 Einführung in die Strukturmechanik

Um in Kapitel 3.2 Kontaktprobleme bzgl. der Mechanik beschreiben zu können, wollen wir in diesem Kapitel kurz die in der vorliegenden Arbeit verwendeten Konzepte der Kontinuumsmechanik einführen.

Definition 2.33 (Kontinuum). Ein *Kontinuum* ist eine Teilmenge $\mathcal{B} \subset \mathbb{R}^3$, dessen Punkte stetig verteilt sind. Den Punkten werden gewisse Materialeigenschaften zugewiesen.

Notation. Wir wollen Vektoren oder vektorwertige (bzw. tensorwertige) Funktionen durch fettgedruckte Buchstaben darstellen (s. auch Anhang C).

2.5.1 Kinematik

Um die Kinematik für ein Kontinuum zu beschreiben, betrachten wir dieses in zeitlich voneinander abhängigen Zuständen. Dabei unterscheiden wir zwischen den *materiellen Punkten* des Kontinuums, die mit $\mathbf{X} = (X_1, X_2, X_3)$ beschrieben werden, und den *räumlichen Punkten* $\mathbf{x} = (x_1, x_2, x_3)$.

Definition 2.34 (Konfiguration). Eine zu jedem Zeitpunkt t differenzierbare und stetige Zuordnung $\mathbf{x} = \varphi(\mathbf{X}, t)$ nennen wir *Konfiguration*. Die Konfiguration zum Zeitpunkt $t = t_0$ nennen wir *Ausgangs- oder Referenzkonfiguration*, die Konfiguration zum aktuellen Zeitpunkt t *aktuelle Konfiguration* oder *Momentankonfiguration*.

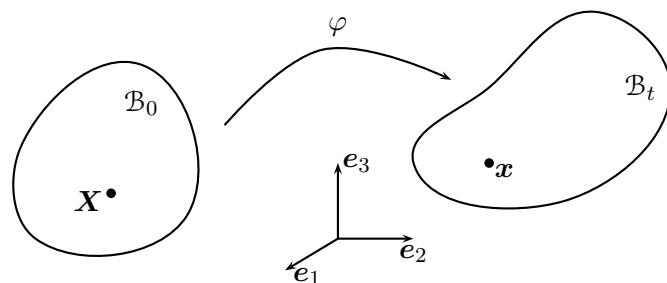


Abbildung 2.7: Konfiguration eines Kontinuums \mathcal{B}

Bemerkung 2.35. Wir können der Einfachheit halber zu einem Startzeitpunkt $t = t_0$ die materiellen Punkte durch die räumlichen Koordinaten der Ausgangskonfiguration $\mathbf{X} = \mathbf{x}(t_0)$ beschreiben. So liegt der Körper \mathcal{B} aus Abbildung 2.7 im selben Koordinatensystem wie die aktuelle Konfiguration \mathcal{B}_t des Körpers.

Damit ist die Bewegung (oder auch Deformation) eines Kontinuums als zeitlich stetige Folge von Konfigurationen $\mathbf{x} = \mathbf{x}(\mathbf{X}, t)$ zu verstehen. Hierfür gibt es zwei grundlegende Betrachtungsweisen, die *Lagrange'sche* und die *Euler'sche Betrachtungsweise*. Bei der Lagrange'schen Betrachtung ist der Beobachter mit dem materiellen Punkt \mathbf{X} verbunden und misst alle Änderungen des Kontinuums an diesem Punkt; jene wird auch *materielle Betrachtungsweise* genannt. Bei der Euler'schen Betrachtungsweise befindet sich der Beobachter am räumlichen Punkt \mathbf{x} und misst zu jedem Zeitpunkt t die Änderungen am Punkt \mathbf{x} , die sich durch das Passieren von Teilchen \mathbf{X} ergeben; jene wird auch *räumliche Betrachtungsweise* genannt.

Um die Deformation eines Körpers \mathcal{B} nun beschreiben zu können, betrachten wir die Veränderung von Linienelementen $d\mathbf{x}$ bzgl. der Ausgangs- und Momentankonfiguration. Wir beschreiben den sogenannten *Deformationsgradienten* \mathbf{F} durch den Faktor, der zwischen der Deformation dieser Linienelemente liegt, d.h.

$$d\mathbf{x} = \mathbf{F} d\mathbf{X} . \quad (2.21)$$

Damit ergibt sich der Deformationsgradient als

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} = \text{Grad } \mathbf{x} , \quad (2.22)$$

dies ist der Gradient bzgl. der materiellen Betrachtung und wird daher auch *materieller Deformationsgradient* genannt. Analog können wir den *räumlichen Deformationsgradienten* erhalten als

$$\mathbf{F}^{-1} = \frac{\partial \mathbf{X}}{\partial \mathbf{x}} = \text{grad } \mathbf{X} . \quad (2.23)$$

Die Deformationsgradienten sind zweistufige Tensoren, die sich jeweils auf die Referenz- und Momentankonfiguration bzgl. der Basen beziehen.

Damit können wir nun die Verzerrung eines Kontinuums mittels des Deformationsgradienten ausdrücken. Um jene zu beschreiben, betrachten wir die Längenänderung zwischen den Linienelementen $d\mathbf{X}$ und $d\mathbf{x}$.

$$\begin{aligned} \|d\mathbf{x}\|^2 - \|d\mathbf{X}\|^2 &= (d\mathbf{x})^T d\mathbf{x} - (d\mathbf{X})^T d\mathbf{X} \\ &\stackrel{(2.21)}{=} (d\mathbf{X})^T \mathbf{F}^T \mathbf{F} d\mathbf{X} - (d\mathbf{X})^T d\mathbf{X} \\ &= (d\mathbf{X})^T (\mathbf{F}^T \mathbf{F} - \mathbf{1}) d\mathbf{X} , \end{aligned}$$

wobei $\mathbf{1}$ den zweistufigen Einheitstensor beschreibt. Also lässt sich die Verzerrung bzgl. der Ausgangskonfiguration durch den *Green-Lagrange'schen Verzerrungstensor*

$$\mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{1}) \quad (2.24)$$

mit dem *rechten Cauchy-Green-Tensor* $\mathbf{C} = \mathbf{F}^T \mathbf{F}$ beschreiben.

Bemerkung 2.36. Dies ist natürlich nur eine Möglichkeit. Wichtig für die Wahl eines Verzerrungsmaßes ist jedoch, dass für eine reine Translation oder Rotation dieses Null wird.

Daher ist beispielsweise $\mathbf{F} - \mathbf{1}$ als Verzerrungsmaß unbrauchbar. Der Deformationsgradient lässt sich in Rotation \mathbf{R} und Streckung \mathbf{U} durch

$$\mathbf{F} = \mathbf{R} \cdot \mathbf{U}$$

aufteilen. Sollten wir nun eine reine Rotation betrachten, so ist $\mathbf{U} = \mathbf{1}$ und damit $\mathbf{F} = \mathbf{R}$, was nicht zwangsläufig der Einheitstensor sein muss.

Wenn wir einen Punkt bzgl. seiner Ausgangs- und Momentankonfiguration vergleichen, so erhalten wir die *Verschiebung*

$$\mathbf{u}(\mathbf{X}, t) = \mathbf{x}(\mathbf{X}, t) - \mathbf{X}. \quad (2.25)$$

Es ergibt sich analog zum materiellen und räumlichen Deformationsgradienten (2.22) und (2.23) mit (2.25) der materielle und räumliche *Verschiebungsgradient*:

$$\begin{aligned} \mathbf{H} &= \text{Grad } \mathbf{u} = \frac{\partial(\mathbf{x} - \mathbf{X})}{\partial \mathbf{X}} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} - \frac{\partial \mathbf{X}}{\partial \mathbf{X}} = \mathbf{F} - \mathbf{1}, \\ \mathbf{h} &= \text{grad } \mathbf{u} = \frac{\partial(\mathbf{x} - \mathbf{X})}{\partial \mathbf{x}} = \frac{\partial \mathbf{x}}{\partial \mathbf{x}} - \frac{\partial \mathbf{X}}{\partial \mathbf{x}} = \mathbf{1} - \mathbf{F}^{-1}. \end{aligned}$$

Damit ergibt sich beispielsweise für den Green-Lagrange'schen Verzerrungstensor (2.24) bzgl. des materiellen Verschiebungsgradienten die Beziehung

$$\begin{aligned} \mathbf{E} &= \frac{1}{2}(\mathbf{C} - \mathbf{1}) = \frac{1}{2}(\mathbf{F}^T \mathbf{F} - \mathbf{1}) = \frac{1}{2}((\mathbf{1} + \mathbf{H})^T (\mathbf{1} + \mathbf{H}) - \mathbf{1}) \\ &= \frac{1}{2}(\mathbf{1} + \mathbf{H}^T + \mathbf{H} + \mathbf{H}^T \mathbf{H} - \mathbf{1}) = \frac{1}{2}(\mathbf{H}^T + \mathbf{H} + \mathbf{H}^T \mathbf{H}). \end{aligned} \quad (2.26)$$

Wenn wir von kleinen Deformationen ausgehen, so ist es sinnvoll für die spätere numerische Berechnung die nichtlinearen Verzerrungsgrößen (wie in (2.26)) zu linearisieren. Dies können wir mithilfe von Taylor und der Gâteaux-Ableitung (A.1) bestimmen. Für eine vektor- oder tensorwertige Funktion \mathbf{A} gilt dann

$$\text{lin}(\mathbf{A})_{\mathbf{x}, \mathbf{u}} = \mathbf{A}(\mathbf{x}) + \frac{d}{d\varepsilon}(\mathbf{A}(\mathbf{x} + \varepsilon \mathbf{u})) \Big|_{\varepsilon=0}. \quad (2.27)$$

Wenden wir also (2.27) auf den Green-Lagrange'schen Verzerrungstensor (2.26) an, so erhalten wir die Linearisierung:

$$\begin{aligned}
 \text{lin}(\mathbf{E})_{\mathbf{X}, \mathbf{u}} &= \underbrace{\mathbf{E}(\mathbf{X})}_{=0} + \frac{d}{d\varepsilon} (\mathbf{E}(\mathbf{X} + \varepsilon\mathbf{u})) \Big|_{\varepsilon=0} \\
 &= \frac{1}{2} \frac{d}{d\varepsilon} \left(\left(\frac{\partial(\mathbf{X} + \varepsilon\mathbf{u})}{\partial\mathbf{X}} \right)^T \frac{\partial(\mathbf{X} + \varepsilon\mathbf{u})}{\partial\mathbf{X}} - \mathbf{1} \right) \Big|_{\varepsilon=0} \\
 &= \frac{1}{2} \frac{d}{d\varepsilon} ((\mathbf{1} + \varepsilon\mathbf{H})^T (\mathbf{1} + \varepsilon\mathbf{H}) - \mathbf{1}) \Big|_{\varepsilon=0} \\
 &= \frac{1}{2} \frac{d}{d\varepsilon} (\mathbf{1} + \varepsilon\mathbf{H}^T + \varepsilon\mathbf{H} + \varepsilon^2 \mathbf{H}^T \mathbf{H} - \mathbf{1}) \Big|_{\varepsilon=0} \\
 &= \frac{1}{2} (\mathbf{H}^T + \mathbf{H} + 2\varepsilon\mathbf{H}^T \mathbf{H}) \Big|_{\varepsilon=0} = \frac{1}{2} (\mathbf{H}^T + \mathbf{H}) =: \boldsymbol{\varepsilon}.
 \end{aligned}$$

Wir wollen der Einfachheit halber in dieser Arbeit von kleinen Deformationen ausgehen. Dann gilt $\mathbf{H} \approx \mathbf{h} = \nabla\mathbf{u}$, somit werden wir im Weiteren den linearisierten Verzerrungstensor

$$\boldsymbol{\varepsilon} = \frac{1}{2} (\nabla^T \mathbf{u} + \nabla \mathbf{u}) \quad (2.28)$$

verwenden.

2.5.2 Kinetik

Da wir von kleinen Deformationen ausgehen, betrachten wir den *Cauchy-Spannungstensor* $\boldsymbol{\sigma}$, der die aktuelle Kraft bzgl. der Querschnittsfläche in der Momentankonfiguration setzt. Wie in [Wri06] Kapitel 3.2.2 beschrieben, gilt das *Cauchy-Theorem*, das besagt, dass die Spannung \mathbf{t} auf einer Schnittfläche eines beliebigen Schnittes im Körper \mathcal{B} gleich der Spannung $\boldsymbol{\sigma}$ in Normalenrichtung \mathbf{n} ist, d.h.

$$\mathbf{t} = \boldsymbol{\sigma} \cdot \mathbf{n}. \quad (2.29)$$

Mit den Bilanzgleichungen für das Momentengleichgewicht kann man herleiten, dass der Cauchy-Spannungstensor symmetrisch ist, also $\boldsymbol{\sigma}^T = \boldsymbol{\sigma}$ gilt.

Betrachten wir nun eine Volumenkraft $\bar{\mathbf{b}}$ und eine Oberflächenlast $\bar{\mathbf{t}}$, die auf den Körper \mathcal{B} wirken, so erhalten wir das (*globale*) Kräftegleichgewicht

$$\int_{\Omega} \bar{\mathbf{b}} \, dv + \int_{\partial\Omega} \bar{\mathbf{t}} \, da = \mathbf{0}, \quad (2.30)$$

wobei $\Omega \subset \mathcal{B} \subset \mathbb{R}^3$ eine Teilmenge des Kontinuums \mathcal{B} beschreibt. Mit dem Cauchy-Theorem (2.29) und dem Satz von Gauß lässt sich (2.30) als Integral über Ω schreiben durch

$$\int_{\Omega} \bar{\mathbf{b}} + \text{div } \boldsymbol{\sigma} \, dv = \mathbf{0}. \quad (2.31)$$

Die Gleichung (2.31) muss nach dem Schnittprinzip auf jeder Teilmenge Ω gelten, was nur erfüllt werden kann, wenn der Integrand Null ist. Damit erhalten wir die sogenannte *starke* oder auch *lokale* Form des Gleichgewichts

$$\operatorname{div} \boldsymbol{\sigma} + \bar{\mathbf{b}} = \mathbf{0} \text{ auf } \Omega. \quad (2.32)$$

Wir wollen in dieser Arbeit von einem konstanten Wärmefeld ausgehen, sodass wir thermodynamische Prozesse vernachlässigen können.

2.5.3 Konstitutive Gleichungen und Prinzipien

Auch für die konstitutiven Gleichungen (Materialannahmen etc.) machen wir uns zu Nutze, dass wir von kleinen Deformationen ausgehen. Wir gehen daher von einem linear elastischen Material aus, d.h. dass Spannung und Verzerrung in einem linearen Zusammenhang stehen. Hierfür werden wir in der Arbeit das *Hooke'sche Materialmodell* verwenden:

$$\boldsymbol{\sigma} = \mathcal{C} : \boldsymbol{\varepsilon} = 2\mu\boldsymbol{\varepsilon} + \lambda(\operatorname{tr} \boldsymbol{\varepsilon})\mathbf{I}, \quad (2.33)$$

wobei λ, μ die *Lamé-Konstanten* darstellen. Dabei handelt es sich um materialabhängige Parameter, die im Zusammenhang mit dem Elastizitätsmodul E und der Querkontraktionszahl ν stehen.

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)}.$$

Der Materialtensor \mathcal{C} ist 4-stufig und mit „:“ ist das doppelt verjüngende Skalarprodukt (s. Definition C.8) gemeint. Es sei bemerkt, dass ν gleich dem Schubmodul G ist.

Da wir in dieser Arbeit immer $\Omega \subset \mathbb{R}^2$ annehmen wollen, wollen wir an dieser Stelle noch zwei Prinzipien zur Behandlung von dreidimensionalen strukturmechanischen Problemen im \mathbb{R}^2 vorstellen (vgl. [Wri09] Kapitel 5.4.1 und 5.4.2).

Ist beispielsweise die dritte Richtung dünn gegenüber den anderen zwei, so können wir das Prinzip des *ebenen Spannungszustand* verwenden. Hierbei ist die Spannung in der dritten Richtung $\sigma_{33} = 0$. Dies impliziert jedoch nicht, dass es in dieser Richtung keine Verzerrung geben muss. Aus dem Hooke'schen Materialgesetz (2.33) errechnen wir nämlich, dass die Verzerrung ε_{33} abhängig von ε_{11} und ε_{22} ist.

$$\varepsilon_{33} = -\frac{\nu}{1-\nu}(\varepsilon_{11} + \varepsilon_{22})$$

Damit lässt sich der Materialtensor \mathcal{C} als Matrix schreiben, wenn wir die *Voigt-Notation* bzgl. des Spannungs- und Verzerrungstensors verwenden.

Analog erhalten wir den *ebenen Verzerrungszustand*, indem wir die Verzerrung in die dritte Richtung $\varepsilon_{33} = 0$ setzen. Anschaulich heißt das, dass

2. Grundlagen

die dritte Richtung unendlich ausgedehnt ist. Auch hier ergeben sich durch Einsetzen von $\varepsilon_{33} = 0$ in (2.33) Abhangigkeiten zwischen den Spannungen:

$$\sigma_{33} = \nu(\sigma_{11} + \sigma_{22}),$$

was einerseits bedeutet, dass die Spannung in der dritten Richtung nicht zwangslaufig Null sein muss und andererseits uns wieder Eintrage aus dem Materialtensor \mathcal{C} eliminieren lasst.

Kapitel 3

Variationsungleichungen

Hindernis- und Kontaktprobleme basieren auf der Minimierung von Energiefunktionalen, wie dies im Theorem von Lax-Milgram (Theorem 2.12) auch schon verwendet wurde. Allerdings minimieren wir diese jetzt nur auf einer Teilmenge der Funktionen aus dem betrachteten Hilbertraum, d.h. wir werden eine Nebenbedingung einführen. Wir wollen in diesem Kapitel die theoretische Grundlage für die Lösung solcher Probleme bilden und eine a priori Fehlerschätzung zwischen exakter und mit der Finiten-Elemente-Methode approximierter Lösung durchführen.

Dieses Kapitel basiert auf [KO88], [Sta11], [Ste12b], [Ste12a], [Wri01], [Wri06], [HHNL80], [Glo08], [Fal74].

3.1 Ein Hindernisproblem

Als Modellproblem wollen wir wieder die Auslenkung $u : \Omega \rightarrow \mathbb{R}$ einer in Ω eingespannten Membran betrachten, die mit der Flächenlast f belastet wird (vgl. auch Abbildung 2.1). Nun wollen wir jedoch die Auslenkung u durch ein Hindernis ψ in Ω behindern (s. Abbildung 3.1 – die Auslenkung u der Membran liegt bzgl. der angedeuteten Kurve in der Skizze rotationssymmetrisch zur z -Achse). Dies führt auf die Minimierung des quadratischen Energiefunktionalen

$$\min_{v \in K} J(v) = \frac{1}{2}a(v, v) - (f, v) \quad (3.1)$$

mit $K := \{v \in H_0^1(\Omega) \mid v \geq \psi \text{ fast überall in } \Omega\}$, wobei $a(\cdot, \cdot), (f, \cdot)$ wie in Kapitel 2.2 definiert sind. Das Funktional J gibt dabei wie zuvor die in der Membran gespeicherte Energie an.

Der Unterschied zur Minimierung von J aus dem Theorem von Lax-Milgram liegt nun also darin, dass wir J nicht über ganz $H_0^1(\Omega)$ minimieren, sondern nur über eine Teilmenge $K \subset H_0^1(\Omega)$ (anschaulich alle Auslenkungen u , die oberhalb des Hindernisses ψ liegen).

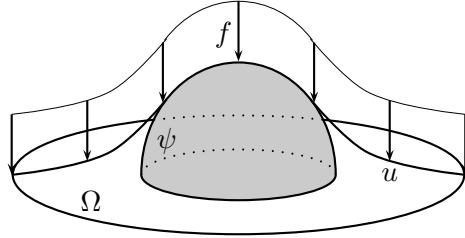


Abbildung 3.1: Ein Hindernisproblem mit Hindernis ψ , konstanter Streckenlast f und Lösung u

Wir können nun analog zum Kapitel 2.2 auch für diese Art von Problemen eine Variationsformulierung herleiten.

3.1.1 Variationsformulierung für das Hindernisproblem

Um für das Hindernisproblem (3.1) eine äquivalente Variationsformulierung zu erhalten, wollen wir die in Anhang A.2 aufgeführten Optimalitätskriterien verwenden. Hierfür müssen wir zunächst zeigen, dass die oben aufgeführte Menge K konvex und abgeschlossen in $H_0^1(\Omega)$ ist.

Lemma 3.1. *Die Menge $K = \{v \in H_0^1(\Omega) \mid v \geq \psi \text{ fast überall in } \Omega\}$ ist eine konvexe abgeschlossene Teilmenge von $H_0^1(\Omega)$.*

Beweis. (i) Es seien $u, v \in K$, d.h. $u \geq \psi$ und $v \geq \psi$ fast überall in Ω . Dann gilt für $t \in [0, 1]$

$$(1-t)u + tv \geq (1-t)\psi + t\psi = \psi,$$

somit ist $(1-t)u + tv \in K$, also K konvex.

(ii) Es sei $(v_n)_{n \in \mathbb{N}} \subset K$ eine konvergente Folge mit $v_n \rightarrow v$ für $n \rightarrow \infty$. Da $H_0^1(\Omega)$ laut Bemerkung A.8 ein abgeschlossener Unterraum von $H^1(\Omega)$ ist, folgt direkt $v \in H_0^1(\Omega)$. Da weiter $v_n \geq \psi$ für alle $n \in \mathbb{N}$ gilt, folgt aus dem Spursatz (vgl. [Bra13] Kapitel II, §3, Satz 3.1), dass auch $v \geq \psi$ fast überall in Ω gilt und damit ist $v \in K$, d.h. K ist abgeschlossen. \square

Satz 3.2. *Es sei $K = \{v \in H_0^1(\Omega) \mid v \geq \psi \text{ fast überall in } \Omega\}$. Das Minimierungsproblem*

$$\min_{v \in K} J(v) = \frac{1}{2}a(v, v) - (f, v) \quad (3.2)$$

ist äquivalent zur Variationsungleichung: Finde $u \in K$, so dass

$$a(u, v - u) \geq (f, v - u) \quad \forall v \in K. \quad (3.3)$$

Beweis. Aus Lemma 2.10 folgt, dass J konvex ist und damit gilt mit Satz A.11, dass $u \in K$ genau dann eine Lösung von (3.2) ist, wenn

$$\mathcal{D}_{v-u}J(u) \geq 0 \quad \forall v \in K \quad (3.4)$$

gilt. Analog zu der berechneten Gâteaux-Ableitung von J in Lemma 2.11, gilt

$$\mathcal{D}_{v-u}J(u) = \frac{d}{dt}J(u + t(v - u))\Big|_{t=0} = a(u, v - u) - (f, v - u)$$

und damit folgt mit (3.4) die Behauptung. \square

Bemerkung 3.3. Wie man mit Satz A.11 nachrechnen kann, gilt analog zu Satz 3.2 auch allgemeiner: Es sei $K \subset H$ eine konvexe Teilmenge eines Hilbertraumes H . Dann ist

$$\min_{v \in K} J(v) = \frac{1}{2}a(v, v) - F(v)$$

äquivalent zur Variationsungleichung: Finde $u \in K$, so dass

$$a(u, v - u) \geq F(v - u) \quad \forall v \in K,$$

wobei $F : H \rightarrow \mathbb{R}$ eine lineare stetige Abbildung ist.

Wie wir sehen, erhalten wir durch die Einführung einer Nebenbedingung keine Variationsgleichung mehr, wie in Kapitel 2.2, sondern eine Variationsungleichung. Anschaulich liegt dies daran, dass die Veränderung der auf die Membran wirkenden Flächenlast f nicht zwangsläufig eine Änderung in der Auslenkung u hervorrufen muss, da diese durch ψ behindert werden könnte.

Diese Ungleichungsrelation überträgt sich auch auf die äquivalente, starke Formulierung des Problems (3.1), die wir auch für das Hindernisproblem, analog zum homogenen Dirichlet-Problem, finden können.

Satz 3.4 (Starke Formulierung des Hindernisproblems). *Jede Lösung $u \in H^2(\Omega) \cap H_0^1(\Omega)$ des Problems*

$$\begin{aligned} -\Delta u - f &\geq 0, \\ u - \psi &\geq 0, \\ (u - \psi)(-\Delta u - f) &= 0 \end{aligned} \quad (3.5)$$

mit $\psi \in H^1(\Omega)$ erfüllt die Variationsungleichung (3.3). Umgekehrt ist jede Lösung $u \in H^2(\Omega) \cap K$ von (3.3) auch eine Lösung von (3.5).

3. Variationsungleichungen

Beweis. „ \Rightarrow “ Sei $u \in H^2(\Omega) \cap H_0^1(\Omega)$ eine Lösung von (3.5), dann gilt für ein beliebiges $v \in K$

$$\begin{aligned} \int_{\Omega} (-\Delta u - f)(v - u) dx &= \underbrace{- \int_{\Omega} \Delta u(v - u) dx}_{\stackrel{\text{Green}}{=} \int_{\Omega} \nabla u \nabla(v - u) dx} - \int_{\Omega} f(v - u) dx \\ &\quad - \underbrace{\int_{\Gamma} (v - u) \partial_{\nu} u ds}_{=0} \\ &= \int_{\Omega} \nabla u \nabla(v - u) dx - \int_{\Omega} f(v - u) dx \\ &= a(u, v - u) - (f, v - u). \end{aligned}$$

Mit $\Omega_0 := \{x \in \Omega \mid u = \psi\}$ folgt, dass $-\Delta u = f$ auf $\Omega_1 := \Omega \setminus \bar{\Omega}_0$ gelten muss. Hieraus ergibt sich

$$\int_{\Omega = \Omega_0 \cup \Omega_1} \underbrace{(-\Delta u - f)(v - u)}_{=0 \text{ auf } \Omega_1} dx = \int_{\Omega_0} \underbrace{(-\Delta u - f)}_{\geq 0} \underbrace{(v - \psi)}_{\geq 0} dx \geq 0.$$

Damit ist u eine Lösung von (3.3)

$$a(u, v - u) \geq (f, v - u) \quad \forall v \in K.$$

„ \Leftarrow “ Es sei $u \in H^2(\Omega) \cap K$ Lösung von (3.3). Weiter sei $v \in K$ beliebig, dann gilt

$$\begin{aligned} 0 &\leq a(u, v - u) - (f, v - u) \\ &= \int_{\Omega} \nabla u \nabla(v - u) dx - \int_{\Omega} f(v - u) dx \\ &\stackrel{\text{Green}}{=} \int_{\Omega} -\Delta u(v - u) dx - \int_{\Omega} f(v - u) dx \\ &= \int_{\Omega} (-\Delta u - f)(v - u) dx. \end{aligned} \tag{3.6}$$

Wir nehmen an, dass $-\Delta u - f < 0$ in einem Ball $B_{r_0} := B_{r_0}(x_0) \subset \Omega$ mit Radius r_0 um $x_0 \in \Omega$ gilt. Sei weiter $\chi \in C^{\infty}(\Omega)$ mit $\chi = 0$ auf $\Omega \setminus \bar{B}_{r_0}$, $\rho(r) := \left(1 - \frac{r}{r_0}\right)^2 \chi > 0$ und $v := u + \rho(r) \in K$, da $u \in K$ und $\rho(r) > 0$. Dann gilt

$$\int_{\Omega} (-\Delta u - f)(v - u) dx = \int_{B_{r_0}} \underbrace{(-\Delta u - f)}_{<0} \underbrace{\rho(r)}_{>0} dx < 0,$$

was im Widerspruch zu (3.6) steht. Also muss $-\Delta u - f \geq 0$ gelten.

Nun nehmen wir an, dass $-\Delta u - f > 0$ und $u > \psi$ fast überall in einem Ball B_{r_0} gilt. Wir betrachten $v := u + \varepsilon \rho(r)(\psi - u) \in K$ mit $0 < \varepsilon \leq 1$, dann folgt

$$\int_{\Omega} (-\Delta u - f)(v - u) dx = \varepsilon \int_{B_{r_0}} \underbrace{(-\Delta u - f)}_{>0} \underbrace{\rho(r)}_{>0} \underbrace{(\psi - u)}_{<0} dx < 0,$$

was wiederum im Widerspruch zu (3.6) steht. Damit muss $u = \psi$ gelten, wenn $-\Delta u = f$ ist. Es folgt, dass $u \in H^2(\Omega) \cap K$ eine Lösung von (3.5) ist. \square

3.1.2 Existenz und Eindeutigkeit der Lösung

Wir wollen nun die Existenz und Eindeutigkeit der Lösung des Hindernisproblems (3.2) bzw. der Variationsungleichung (3.3) überprüfen. Hierzu betrachten wir zunächst wieder allgemein das reelle, quadratische Funktional

$$J : H \rightarrow \mathbb{R}, \quad J(v) = \frac{1}{2}a(v, v) - F(v).$$

Folgende Voraussetzungen wollen wir für den Beweis der Existenz und Eindeutigkeit der Lösung annehmen.

Voraussetzung 3.5. Sei H ein reeller Hilbertraum mit Skalarprodukt $(\cdot, \cdot)_H$ und der damit induzierten Norm $\|\cdot\|_H$. Mit H' bezeichnen wir den Dualraum zu H . Weiter sei vorausgesetzt:

- (a) $a : H \times H \rightarrow \mathbb{R}$ ist eine stetige, koerzive Bilinearform,
- (b) $F : H \rightarrow \mathbb{R}$ ist ein stetiges, lineares Funktional,
- (c) $K \neq \emptyset$ ist eine abgeschlossene, konvexe Teilmenge von H .

Theorem 3.6 (Existenz und Eindeutigkeit). *Unter den obigen Voraussetzungen 3.5 hat die Variationsungleichung: Finde $u \in K$, so dass*

$$a(u, v - u) \geq F(v - u) \quad \forall v \in K \tag{3.7}$$

ist, genau eine Lösung.

Beweis. (i) Eindeutigkeit: Es seien $u_1, u_2 \in K$ zwei Lösungen der Variationsungleichung (3.7), d.h.

$$a(u_1, v - u_1) \geq F(v - u_1) \quad \forall v \in K, \tag{3.8}$$

$$a(u_2, v - u_2) \geq F(v - u_2) \quad \forall v \in K. \tag{3.9}$$

Addieren wir (3.8) und (3.9) miteinander und setzen zuvor $v = u_2$ in (3.8) und $v = u_1$ in (3.9), so erhalten wir

$$\begin{aligned} 0 &\leq a(u_1, u_2 - u_1) - F(u_2 - u_1) + a(u_2, u_1 - u_2) - \underbrace{F(u_1 - u_2)}_{=F(u_2 - u_1)} \\ &= a(u_1, u_2 - u_1) - a(u_2, u_2 - u_1) = -a(u_2 - u_1, u_2 - u_1) \\ &\leq -\alpha \|u_2 - u_1\|_H^2. \end{aligned}$$

Also gilt $\|u_2 - u_1\|_H^2 \leq 0 \Rightarrow \|u_2 - u_1\|_H^2 = 0$ und damit folgt $u_1 = u_2$.

3. Variationsungleichungen

(ii) Existenz: Aus dem Darstellungssatz von Riesz bzw. das Lemma 2.16 folgt, dass ein $A \in \mathcal{L}(H, H), l \in H$ existiert, so dass

$$\begin{aligned} a(u, v) &= (Au, v)_H \quad \forall u, v \in H, \\ F(v) &= (l, v)_H \quad \forall v \in H. \end{aligned}$$

Damit gilt

$$\begin{aligned} F(v - u) - a(u, v - u) &= (l, v - u)_H - (Au, v - u)_H \\ &= (l - Au, v - u)_H \leq 0. \end{aligned}$$

Durch Multiplikation mit $\varrho > 0$ und Addition der Null erhalten wir das äquivalente Problem: Finde $u \in K$, so dass

$$(u - \varrho(Au - l) - u, v - u)_H \leq 0 \quad \forall v \in K. \quad (3.10)$$

Nach Satz 2.3 ist u damit das Bild der Projektion von $u - \varrho(Au - l)$ auf K , d.h.

$$u = P_K(u - \varrho(Au - l)).$$

Es bleibt daher zu zeigen, dass $W_\varrho : H \rightarrow K, W_\varrho(v) := P_K(v - \varrho(Av - l))$ einen Fixpunkt besitzt. Mit Anwendung von Satz 2.4 und der Koerzivität von a rechnen wir nach, dass

$$\begin{aligned} \|W_\varrho(v_1) - W_\varrho(v_2)\|_H^2 &= \|P_K(v_1 - \varrho(Av_1 - l)) - P_K(v_2 - \varrho(Av_2 - l))\|_H^2 \\ &\leq \|v_1 - \varrho(Av_1 - l) - (v_2 - \varrho(Av_2 - l))\|_H^2 \\ &= \|(v_1 - v_2) - \varrho A(v_1 - v_2)\|_H^2 \\ &= \|v_1 - v_2\|_H^2 + \varrho^2 \|A(v_1 - v_2)\|_H^2 \\ &\quad - \underbrace{\varrho (A(v_1 - v_2), v_1 - v_2)_H - \varrho (v_1 - v_2, A(v_1 - v_2))_H}_{=2\varrho (A(v_1 - v_2), v_1 - v_2)_H = 2\varrho a(v_1 - v_2, v_1 - v_2)} \\ &\leq \|v_1 - v_2\|_H^2 + \varrho^2 \|A\|^2 \|v_1 - v_2\|_H^2 - 2\varrho \alpha \|v_1 - v_2\|_H^2 \\ &= (1 - 2\varrho \alpha + \varrho^2 \|A\|^2) \|v_1 - v_2\|_H^2 \end{aligned}$$

mit $\|A\| := \sup_{v \in H} \frac{\|Av\|_H}{\|v\|_H}$ gilt. Also ist die Abbildung W_ϱ eine Kontraktion, wenn gilt

$$1 - 2\varrho \alpha + \varrho^2 \|A\|^2 < 1 \implies 0 < \varrho < \frac{2\alpha}{\|A\|^2}.$$

Nach dem Banach'schen Fixpunktsatz (vgl. [Sto99] Satz 5.2.3) existiert für solch ein ϱ ein $u \in H$ mit $u = W_\varrho(u) = P_K(u - \varrho(Au - l))$.

Insgesamt gibt es also für das Problem (3.7) genau eine Lösung. \square

Korollar 3.7. Das Problem (3.1) hat eine eindeutige Lösung.

Beweis. Da laut Lemma 3.1 die Menge

$$K = \{v \in H_0^1(\Omega) \mid v \geq \psi \text{ fast überall in } \Omega\}$$

abgeschlossen und konvex ist, $F(v) = (f, v)$ ein stetiges, lineares Funktional und

$$a(u, v) = \int_{\Omega} \nabla u \nabla v \, dx$$

stetig, bilinear und koerziv ist, sind die Voraussetzungen für Theorem 3.6 erfüllt. Damit hat das Problem, finde $u \in K$, so dass

$$a(u, v - u) \geq (f, v - u) \quad \forall v \in K, \quad (3.11)$$

genau eine Lösung. Nach Satz 3.2 ist die Minimierung des Funktionals (3.1) äquivalent zur Variationsungleichung (3.11) und damit folgt die Behauptung. \square

Bemerkung 3.8. Insbesondere hat auch das Problem (3.5) nach Satz 3.4 und Theorem 3.6 eine eindeutige Lösung, wenn $u \in H^2(\Omega) \cap H_0^1(\Omega)$ ist.

3.1.3 Lösung des Hindernisproblems mittels FEM

Analog zu Kapitel 2.3 können wir nun auch die Variationsungleichung (3.11) mittels FEM lösen. Hierzu betrachten wir (3.11) bzgl. eines endlich dimensionalen Unterraum

$$K_h := \{v_h \in V_h \mid v_h(p) \geq \psi(p) \forall p \in \mathcal{N} \cap \Omega\},$$

wobei \mathcal{N} die Knotenmenge bzgl. der Triangulierung \mathcal{T}_h bezeichnet und V_h wie oben ein endlich dimensionaler Unterraum eines Hilbertraumes H ist. Damit schreibt sich (3.11) in diskreter Form: Finde $u_h \in K_h$, so dass

$$a(u_h, v_h - u_h) \geq (f, v_h - u_h) \quad \forall v_h \in K_h. \quad (3.12)$$

Auch für das diskrete Problem existiert eine eindeutige Lösung, wie sich mithilfe des nächsten Satzes zeigen lässt.

Satz 3.9 (Fixpunktsatz von Brouwer). *Es sei $K \neq \emptyset$ eine kompakte, konvexe Teilmenge eines endlich dimensionalen normierten Raumes H und $F : K \rightarrow K$ sei stetig. Dann besitzt F einen Fixpunkt $v \in K$.*

Beweis. Der Beweis ist in [Wer11] Kapitel 4 Satz 7.15 zu finden. \square

Theorem 3.10 (Existenz und Eindeutigkeit). *Es gelten die Voraussetzungen 3.5. Das Problem (3.12) hat eine eindeutige Lösung $u_h \in K_h$.*

3. Variationsungleichungen

Beweis. Der Beweis ist analog zu Theorem 3.6 zu führen. Wir ersetzen lediglich H durch V_h und K durch K_h und verwenden im endlich dimensionalen Raum V_h den Fixpunktsatz von Brouwer. \square

Bemerkung. In Kapitel 2.2 von [Sta08] sind die Argumente bzgl. der Existenz und Eindeutigkeit einer Lösung von (3.11) für den endlich dimensionalen Fall K_h auch noch einmal im Einzelnen aufgeführt.

Es sei $\mathcal{B}_h = \{\phi_1, \dots, \phi_N\}$ eine nodale Basis von V_h , d.h. analog zu (2.11) können wir u_h und v_h mit Koordinaten $\mu_i, \nu_i, i = 1, \dots, N$ bzgl. \mathcal{B}_h ausdrücken. Dann schreiben wir (3.12) als

$$\begin{aligned} & \sum_{i=1}^N \sum_{j=1}^N \mu_i a(\phi_i, \phi_j) (\nu_j - \mu_j) \geq \sum_{j=1}^N (f, \phi_j) (\nu_j - \mu_j) \\ \iff & \boldsymbol{\mu}^T A(\boldsymbol{\nu} - \boldsymbol{\mu}) \geq \mathbf{f}^T (\boldsymbol{\nu} - \boldsymbol{\mu}) \end{aligned}$$

mit $A = [a(\phi_j, \phi_i)]_{i,j=1}^N$, $\boldsymbol{\mu} = [\mu_i]_{i=1}^N$, $\boldsymbol{\nu} = [\nu_i]_{i=1}^N$ und $\mathbf{f} = [(f, \phi_i)]_{i=1}^N$. Damit lässt sich die Menge K_h auch eindeutig durch die Koordinatenvektoren bzgl. \mathcal{B}_h ausdrücken. Die Menge K_h ist bzgl. V_h äquivalent zu

$$K_S := \{\boldsymbol{\nu} \in \mathbb{R}^N \mid \nu_i \geq \psi(p_i), p_i \in \mathcal{N} \cap \Omega, i = 1, \dots, N\}. \quad (3.13)$$

Im Folgenden schreiben wir $\boldsymbol{\psi} := [\psi(p_i)]_{i=1}^N$ mit $p_i \in \mathcal{N} \cap \Omega$.

Bemerkung. K_h bzw. K_S sind analog zu K konvex und abgeschlossen.

Damit erhalten wir aus (3.12) die diskrete Variationsungleichung: Finde $\boldsymbol{\mu} \in K_S$, so dass

$$(A\boldsymbol{\mu} - \mathbf{f})^T (\boldsymbol{\nu} - \boldsymbol{\mu}) \geq 0 \quad \forall \boldsymbol{\nu} \in K_S. \quad (3.14)$$

Es fehlt uns nun jedoch die Möglichkeit solch eine Ungleichung, die offensichtlich nicht mehr linear ist, zu lösen. Wir wollen also (3.14) mathematisch äquivalent formulieren, um das erzeugte Problem lösen zu können.

Satz 3.11. Das Problem (3.14) ist äquivalent zum linearen Komplementaritätsproblem: Bestimme $\boldsymbol{\mu} \in K_S$, so dass

$$A\boldsymbol{\mu} - \mathbf{f} \geq \mathbf{0} \quad \text{und} \quad (A\boldsymbol{\mu} - \mathbf{f})^T (\boldsymbol{\mu} - \boldsymbol{\psi}) = 0 \quad (3.15)$$

gilt.

Beweis. „ \Rightarrow “ Sei $\boldsymbol{\mu} \in K_S$ Lösung von (3.14). Wir setzen $\boldsymbol{\nu} = \boldsymbol{\mu} + \mathbf{e}_i \geq \boldsymbol{\psi}$ mit einem beliebigen $i \in \{1, \dots, N\}$, wobei \mathbf{e}_i den i -te Einheitsvektor bezeichnet. Dann gilt

$$0 \leq (A\boldsymbol{\mu} - \mathbf{f})^T (\boldsymbol{\nu} - \boldsymbol{\mu}) = (A\boldsymbol{\mu} - \mathbf{f})^T \mathbf{e}_i = (A\boldsymbol{\mu} - \mathbf{f})_i.$$

Da i beliebig war, folgt $A\boldsymbol{\mu} - \mathbf{f} \geq \mathbf{0}$.

Wir nehmen nun an, dass ein $i \in \{1, \dots, N\}$ existiert, so dass

$$(A\boldsymbol{\mu} - \mathbf{f})_i(\boldsymbol{\mu} - \boldsymbol{\psi})_i > 0$$

ist. Weiter wählen wir

$$\boldsymbol{\nu} = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_{i-1} \\ 0 \\ \mu_{i+1} \\ \vdots \\ \mu_N \end{pmatrix} + \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \psi_i \\ 0 \\ \vdots \\ 0 \end{pmatrix} \geq \boldsymbol{\psi}$$

und damit folgt

$$0 > (A\boldsymbol{\mu} - \mathbf{f})_i(\boldsymbol{\psi} - \boldsymbol{\mu})_i = (A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\nu} - \boldsymbol{\mu}) \geq 0,$$

was im Widerspruch zu (3.14) steht. Daraus folgt die Behauptung.

„ \Leftarrow “ Es sei $\boldsymbol{\mu} \in K_S$ Lösung von (3.15). Dann rechnen wir nach, dass für ein beliebiges $\boldsymbol{\nu} \in K_S$ gilt

$$\begin{aligned} (A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\nu} - \boldsymbol{\mu}) &= (A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\nu} - \boldsymbol{\psi} + \boldsymbol{\psi} - \boldsymbol{\mu}) \\ &= \underbrace{(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\nu} - \boldsymbol{\psi})}_{\geq 0} + \underbrace{(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\psi} - \boldsymbol{\mu})}_{\geq 0} - \underbrace{(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\mu} - \boldsymbol{\psi})}_{=0} \\ &\geq 0. \quad \square \end{aligned}$$

Satz 3.12 (Äquivalenz zu quadratischem Programm). *Das Problem (3.14) ist äquivalent zum quadratischen Programm*

$$\min_{\boldsymbol{\nu} \in \mathbb{R}^N} J(\boldsymbol{\nu}) = \frac{1}{2} \boldsymbol{\nu}^T A \boldsymbol{\nu} - \mathbf{f}^T \boldsymbol{\nu} \quad s.t. \quad \boldsymbol{\nu} \geq \boldsymbol{\psi}. \quad (3.16)$$

Beweis. Wir zeigen zunächst die Äquivalenz von (3.15) zu (3.16) und dann folgt mit Satz 3.11 die Behauptung.

„ \Rightarrow “ Es sei $\boldsymbol{\mu} \in \mathbb{R}^N$ Lösung vom Problem (3.15). Dann folgt mit einem beliebigen $\boldsymbol{\nu} \in K_S$

$$\begin{aligned} J(\boldsymbol{\nu}) - J(\boldsymbol{\mu}) &= \frac{1}{2} \boldsymbol{\nu}^T A \boldsymbol{\nu} - \mathbf{f}^T \boldsymbol{\nu} - \frac{1}{2} \boldsymbol{\mu}^T A \boldsymbol{\mu} + \mathbf{f}^T \boldsymbol{\mu} \\ &= \frac{1}{2} \underbrace{(\boldsymbol{\nu} - \boldsymbol{\mu})^T A (\boldsymbol{\nu} - \boldsymbol{\mu})}_{\geq 0 \text{ wegen Bem. 2.18}} + \boldsymbol{\mu}^T A \boldsymbol{\nu} - \boldsymbol{\mu}^T A \boldsymbol{\mu} - \mathbf{f}^T (\boldsymbol{\nu} - \boldsymbol{\mu}) \\ &\geq (A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\nu} - \boldsymbol{\psi} + \boldsymbol{\psi} - \boldsymbol{\mu}) \\ &= \underbrace{(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\nu} - \boldsymbol{\psi})}_{\geq 0} + \underbrace{(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\psi} - \boldsymbol{\mu})}_{\geq 0} - \underbrace{(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\mu} - \boldsymbol{\psi})}_{=0} \\ &\geq 0. \end{aligned}$$

3. Variationsungleichungen

Somit ist $\boldsymbol{\mu} \in K_S$ auch Lösung des quadratischen Programms (3.16).

„ \Leftarrow “ Sei $\boldsymbol{\mu} \in K_S$ Lösung von (3.16), dann gelten nach [NW06] Kapitel 12, Theorem 12.1 für die Lagrange-Funktion

$$\mathcal{L}(\boldsymbol{\nu}, \boldsymbol{\lambda}) = J(\boldsymbol{\nu}) - \boldsymbol{\lambda}^T(\boldsymbol{\nu} - \boldsymbol{\psi})$$

die Karush-Kuhn-Tucker Bedingungen für den Optimalpunkt $(\boldsymbol{\mu}, \boldsymbol{\lambda}^*)$

$$\nabla_{\boldsymbol{\nu}} \mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\lambda}^*) = \nabla J(\boldsymbol{\mu}) - \boldsymbol{\lambda}^* = A\boldsymbol{\mu} - \mathbf{f} - \boldsymbol{\lambda}^* \stackrel{!}{=} \mathbf{0}, \quad (3.17a)$$

$$\boldsymbol{\mu} - \boldsymbol{\psi} \geq \mathbf{0}, \quad (3.17b)$$

$$\boldsymbol{\lambda}^* \geq \mathbf{0}, \quad (3.17c)$$

$$\lambda_i^*(\mu_i - \psi_i) = 0 \quad \forall i = 1, \dots, N. \quad (3.17d)$$

Mit (3.17a) gilt also $\boldsymbol{\lambda}^* = A\boldsymbol{\mu} - \mathbf{f}$ und daher folgt aus (3.17c)

$$A\boldsymbol{\mu} - \mathbf{f} \geq \mathbf{0}.$$

Aus (3.17d) folgt wegen $(A\boldsymbol{\mu} - \mathbf{f})_i(\mu_i - \psi_i) = 0$ für alle $i = 1, \dots, N$ direkt

$$(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\mu} - \boldsymbol{\psi}) = 0.$$

Also ist $\boldsymbol{\mu} \in K_S$ auch Lösung von (3.15). \square

Bemerkung 3.13. Analog zu Satz 3.11 und 3.12 können wir durch leichte Abwandlung der Beweise zeigen, dass das quadratische Programm

$$\min_{\boldsymbol{\nu} \in \mathbb{R}^N} J(\boldsymbol{\nu}) = \frac{1}{2} \boldsymbol{\nu}^T A \boldsymbol{\nu} - \mathbf{f}^T \boldsymbol{\nu} \quad \text{s.t.} \quad B\boldsymbol{\nu} \geq \boldsymbol{\psi} \quad (3.18)$$

mit $B \in \mathbb{R}^{M \times N}$ äquivalent ist zur Variationsungleichung: Finde $\boldsymbol{\mu} \in \mathbb{R}^N$ mit $B\boldsymbol{\mu} \geq \boldsymbol{\psi}$, so dass

$$(A\boldsymbol{\mu} - \mathbf{f})^T(\boldsymbol{\nu} - \boldsymbol{\mu}) \geq 0 \quad \forall \boldsymbol{\nu} \in \mathbb{R}^N \text{ mit } B\boldsymbol{\nu} \geq \boldsymbol{\psi}. \quad (3.19)$$

Diese Formulierung werden wir später in Kapitel 3.2.3 wiederfinden, wenn wir Kontaktprobleme mittels der Finiten-Elemente-Methode lösen werden.

Bemerkung 3.14. Die quadratischen Programme (3.16) und (3.18) haben mit den Voraussetzungen aus Bemerkung 2.18 eine globale Lösung $\boldsymbol{\mu}$, wenn diese die KKT-Bedingungen (B.3) erfüllt (vgl. Theorem B.1).

Da wir wegen Bemerkung 2.18 konvexe, quadratische Programme vorliegen haben, können wir diese mit dem in Anhang B.2 vorgestellten Active-Set-Algorithmus lösen. Wie wir später sehen werden, ist es sinnvoll bei der Implementierung auf eine Innere-Punkte-Methode (kurz: IPM) zurückzutreifen.

Es bleibt noch zu prüfen, ob die Lösung der Variationsungleichung für Netzverfeinerung an die exakte Lösung konvergiert. Folgende Voraussetzungen werden hierfür für die verwendeten Ansatzräume angenommen.

Voraussetzung 3.15. Gegeben sei ein Parameter $h \rightarrow 0$. Weiter seien $(V_h)_h$ eine Familie aus abgeschlossenen Teilmengen eines Hilbertraums H , $K \subset H$ eine nichtleere, konvexe, abgeschlossene Teilmenge und $(K_h)_h$ eine Familie von abgeschlossenen, konvexen, nichtleeren Teilmengen von H , so dass $K_h \subset V_h$ für alle h .

Dabei sei K_h eine Approximation von K im folgenden Sinne:

- (i) wenn $(v_h)_h$ eine in H beschränkte Folge mit $v_h \in K_h$ ist, dann folgt $v_h \rightarrow v \in K$,
- (ii) es existiert ein $W \subset H$ mit $\overline{W} = K$ und ein $I_h : W \rightarrow K_h$, so dass

$$\lim_{h \rightarrow 0} I_h v = v$$

stark in H für alle $v \in W$ konvergiert.

Theorem 3.16 (a priori Konvergenz). *Mit den obigen Voraussetzungen für K und $(K_h)_h$ gilt für die Lösungen u von (3.7) und u_h vom approximierten Problem: Finde $u_h \in K_h$, so dass*

$$a(u_h, v_h - u_h) \geq F(v_h - u_h) \quad \forall v_h \in K_h, \quad (3.20)$$

der Zusammenhang

$$\lim_{h \rightarrow 0} \|u_h - u\|_H = 0.$$

Beweis. (i) Abschätzung von u_h : Es sei u_h Lösung von (3.20), dann gilt nach einer Umformung für alle $v_h \in K_h$

$$\begin{aligned} a(u_h, u_h) &\leq a(u_h, v_h) - F(v_h - u_h) \\ &= (Au_h, v_h)_H - (f, v_h - u_h)_H \\ &\stackrel{\text{CS}}{\leq} \underbrace{\|Au_h\|_H}_{\leq \|A\| \|u_h\|_H} \|v_h\|_H + \|f\|_H \underbrace{\|v_h - u_h\|_H}_{\leq \|v_h\|_H + \|u_h\|_H} \\ &\leq \|A\| \|u_h\|_H \|v_h\|_H + \|f\|_H (\|v_h\|_H + \|u_h\|_H). \end{aligned}$$

Zusammen mit der Koerzivität folgt dann

$$\alpha \|u_h\|_H^2 \leq \|A\| \|u_h\|_H \|v_h\|_H + \|f\|_H (\|v_h\|_H + \|u_h\|_H). \quad (3.21)$$

Wähle ein festes $v_0 \in W$, sodass $I_h v_0 = v_h \in K_h$ gilt. Aus Voraussetzung 3.15 (ii) folgt dann

$$\lim_{h \rightarrow 0} I_h v_0 = v_0$$

3. Variationsungleichungen

und daher muss v_h beschränkt sein, d.h. es existiert ein $m \in \mathbb{R} : \|v_h\|_H \leq m$ für alle h . Zusammen mit (3.21) gilt dann

$$\begin{aligned}\|u_h\|_H^2 &\leq \frac{1}{\alpha}(m\|A\|\|u_h\|_H + \|f\|_H(m + \|u_h\|_H)) \\ &= \underbrace{\left(\frac{m}{\alpha}\|A\| + \|f\|_H\right)}_{=:c_1}\|u_h\|_H + \underbrace{\frac{m}{\alpha}\|f\|_H}_{=:c_2} \\ &= c_1\|u_h\|_H + c_2\end{aligned}$$

und damit können wir anhand quadratischer Ergänzung folgern, dass es ein $c \in \mathbb{R}$ gibt mit $\|u_h\| \leq c$ für alle h , d.h. $(u_h)_h$ ist gleichmäßig beschränkt.

(ii) schwache Konvergenz: Da $(u_h)_h$ in H gleichmäßig beschränkt ist, folgt mit Bemerkung A.14 (b), dass es eine schwach konvergente Teilfolge $(u_{h_j})_{h_j} \in K_{h_j}$ mit einem Grenzwert $u^* \in H$ gibt, d.h.

$$u_{h_j} \rightharpoonup u^* \in H.$$

Mit den Voraussetzung 3.15 (i) für $(K_h)_h$ folgt direkt $u^* \in K$, außerdem ist u^* nach Bemerkung A.14 (e) eindeutig.

Wir zeigen nun, dass u^* eine Lösung von (3.7) ist. Für die oben betrachtete Teilfolge gilt

$$a(u_{h_j}, v_{h_j} - u_{h_j}) \geq F(v_{h_j} - u_{h_j}) \quad \forall v_{h_j} \in K_{h_j}. \quad (3.22)$$

Sei $v \in W$ mit $v_{h_j} = I_{h_j}v$. Dann gilt $v_{h_j} = I_{h_j}v \rightarrow v \in W$ für $h_j \rightarrow 0$. Mit (3.22) folgt

$$\begin{aligned}a(u_{h_j}, u_{h_j}) &\leq a(u_{h_j}, v_{h_j}) - F(v_{h_j} - u_{h_j}) \\ &= a(u_{h_j}, I_{h_j}v) - F(v_{h_j} - u_{h_j}) \\ \implies \liminf_{h_j \rightarrow 0} a(u_{h_j}, u_{h_j}) &\leq a(u^*, v) - F(v - u^*).\end{aligned} \quad (3.23)$$

Weiter schätzen wir durch Bemerkung A.14 (f) nach unten ab

$$\liminf_{h_j \rightarrow 0} a(u_{h_j}, u_{h_j}) = \liminf_{h_j \rightarrow 0} \|u_{h_j}\|_H^2 \geq \|u^*\|_H^2 = a(u^*, u^*). \quad (3.24)$$

Insgesamt folgt also mit (3.23) und (3.24)

$$a(u^*, u^*) \leq \liminf_{h_j \rightarrow 0} a(u_{h_j}, u_{h_j}) \leq a(u^*, v) - F(v - u^*)$$

und damit nach Umformung

$$a(u^*, v - u^*) \geq F(v - u^*) \quad \forall v \in W.$$

Da W dicht in K liegt, d.h. $\overline{W} = K$, und a, F stetig sind, erhalten wir

$$a(u^*, v - u^*) \geq F(v - u^*) \quad \forall v \in K$$

mit $u^* \in K$, also ist $u^* =: u$ Lösung von (3.7). Da u ein Häufungspunkt von $(u_h)_h$ bzgl. der schwachen Topologie von H ist, konvergiert auch die Folge $(u_h)_h$ schwach gegen u .

(iii) starke Konvergenz: Aus der Koerzivität von a folgt

$$\begin{aligned} 0 &\leq \alpha \|u_h - u\|_H^2 \leq a(u_h - u, u_h - u) \\ &= a(u_h, u_h) - a(u_h, u) - a(u, u_h) + a(u, u), \end{aligned} \quad (3.25)$$

wobei u_h Lösung des approximierten Problems (3.20) und u Lösung des exakten Problems (3.7) ist. Es sei $v \in W$ mit $I_h v = v_h \in K_h$, dann folgt aus (3.20)

$$a(u_h, u_h) \leq a(u_h, I_h v) - F(I_h v - u_h) \quad \forall v \in W. \quad (3.26)$$

Da $u_h \rightharpoonup u$ in H und $I_h v \rightarrow v$ in H für $h \rightarrow 0$, folgt aus (3.25) und (3.26) unter Verwendung von Voraussetzung (ii)

$$0 \leq \alpha \lim_{h \rightarrow 0} \|u_h - u\|_H^2 \leq a(u, v - u) - F(v - u) \quad \forall v \in W. \quad (3.27)$$

Da a und F stetig sind und W dicht in K liegt, gilt (3.27) auch für alle $v \in K$. Setzen wir dann also $v = u$ in (3.27), dann folgt die Behauptung $\lim_{h \rightarrow 0} \|u_h - u\|_H^2 = 0$. \square

Es sei $V_h \subset H_0^1(\Omega)$ ein beliebiger endlich dimensionaler Unterraum von $H_0^1(\Omega)$ und K_h wie zu Beginn dieses Kapitels definiert. Dann ist $K_h \subset V_h$ für alle Parameter h . Außerdem folgt Voraussetzung 3.15 (i) wegen der Abgeschlossenheit von K_h . Betrachten wir

$$W := \{v \in H_0^1(\Omega) \mid v > \psi \text{ f.ü. in } \Omega\},$$

so gilt $\overline{W} = K$ und mit der bekannten Spline-Interpolation folgt dann auch Voraussetzung 3.15 (ii). Damit gilt

$$\lim_{h \rightarrow 0} \|u_h - u\|_1 = 0.$$

Es lässt sich folglich auch eine a priori Abschätzung für den Fehler von u und u_h herleiten. Die Herleitung ist detailliert unter anderem in [Fal74] wiederzufinden.

Theorem 3.17 (a priori Fehlerabschätzung). *Seien u und u_h die Lösungen von (3.7) und (3.20). Dann existiert eine Konstante $C := C(\Omega, f, \psi)$ unabhängig von u , so dass*

$$\|u_h - u\|_1 \leq Ch.$$

Beweis. Vgl. [Fal74] Theorem 2 bzw. [Ste12b] Theorem 6.4. \square

Damit führt die Netzverfeinerung also zur exakten Lösung der Variationsungleichung (3.11). Inwiefern adaptive Netzverfeinerung hier sinnvoll ist, wollen wir in Kapitel 4 betrachten.

3.2 Kontaktprobleme

Das in Kapitel 3.1 vorgestellte Hindernisproblem ist ein Modellproblem zur Minimierung eines Energiefunktionalen unter Nebenbedingung. Kontaktprobleme sind Probleme aus der Strukturmechanik, die auch auf eine Variationsungleichung führen. Daher können wir die Ideen zur Lösung eines Hindernisproblems auf diese Problemstellung übertragen.

3.2.1 Mathematische Modellierung eines Kontaktproblems

Zunächst wollen wir mithilfe der in Kapitel 2.5 eingeführten mechanischen Gleichungen ein Kontaktproblem mathematisch modellieren.

Voraussetzung 3.18. Wir treffen folgende Annahmen für unser Kontaktmodell:

- (a) Die in Kontakt stehenden Körper sind beschränkt.
- (b) Es liegen linear elastische Materialien mit kleinen Deformationen vor.
- (c) Wir betrachten ein konstantes Temperaturfeld, d.h. thermodynamische Prozesse werden ausgeschlossen.
- (d) Zu Beginn, also in der Ausgangskonfiguration, gilt für die Spannung und Verzerrung

$$\boldsymbol{\sigma} = \mathbf{0}, \quad \boldsymbol{\varepsilon} = \mathbf{0}.$$

- (e) Wir gehen von einem reibunglosen Kontakt aus. Diese Art von Kontaktproblem wird als *Signorini-Kontakt-Problem* bezeichnet.
- (f) Wir gehen von ebenen Problemen aus, d.h. $\Omega \subset \mathbb{R}^2$. Im \mathbb{R}^3 sind alle Resultate analog.

Zur Herleitung der starken Kontaktformulierung wollen wir zwei Körper $\mathcal{B}^1, \mathcal{B}^2$ betrachten, welche wegen Voraussetzung 3.18 (a) durch zwei beschränkte Gebiete Ω^1, Ω^2 mathematisch beschrieben werden können. Diese Voraussetzung lässt sich auch auf beliebig viele Körper verallgemeinern (vgl. hierfür [CSW99]).

Weiter lassen sich die Ränder $\Gamma^i = \partial\Omega^i$, $i = 1, 2$, in drei disjunkte Teile unterteilen (s. Abbildung 3.2):

Γ_u^i : Der *Dirichlet-Rand*, oder auch *Verschiebungsrand*, auf dem die Werte von der Verschiebung \mathbf{u} vorgegeben sind.

Γ_σ^i : Der *Neumann-Rand*, oder auch *Spannungsrand*, auf dem die Oberflächenlast bzw. -spannung $\bar{\mathbf{t}}$ vorgegeben ist.

Γ_c^i : Der *Kontaktrand*, auf dem die Kontaktbedingungen definiert sind.

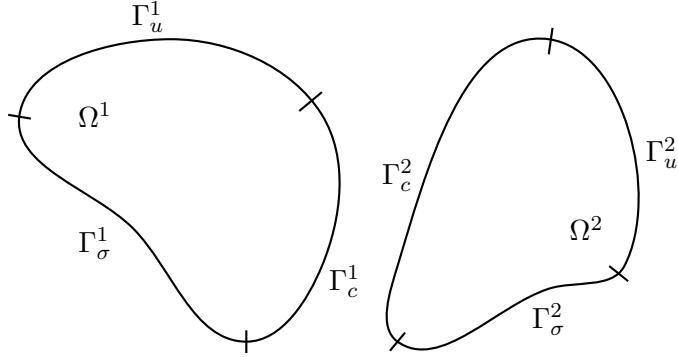


Abbildung 3.2: Körper \mathcal{B}^1 und \mathcal{B}^2 mit Randbezeichnungen

Kontaktkinematik

Wir betrachten zunächst die Kontaktkinematik (wie in [CSW99]) etwas allgemeiner, als in [Wri01] oder [Wri06] beschrieben. Für die Formulierung der Kontaktbedingungen werden den Körpern $\mathcal{B}^1, \mathcal{B}^2$ die Bezeichnungen *Master* und *Slave* zugeordnet. Mit Slave bezeichnen wir dabei die Menge an Punkten, für die überprüft wird, ob sie in die Master-Fläche eindringen. Die Zuordnung von Master und Slave ist jedoch für das Ergebnis der Kontaktbedingung vollkommen irrelevant. Es sei daher o.B.d.A. \mathcal{B}^1 der Slave.

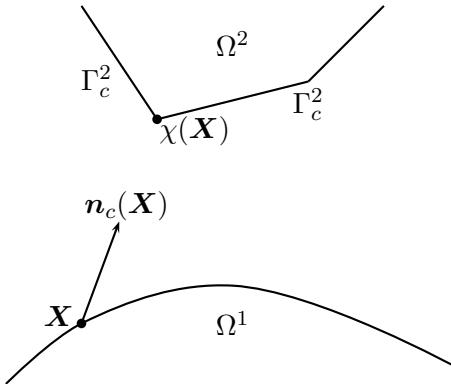


Abbildung 3.3: Kontaktformulierung zwischen zwei Körpern

Für einen gegebenen Punkt $\mathbf{X} \in \Omega^1$, bzw. $\mathbf{X} \in \Gamma_c^1$, in der Ausgangskonfiguration ist $\bar{\mathbf{X}} := \chi(\mathbf{X})$ derjenige Punkt aus Γ^2 , der minimalsten Abstand zu \mathbf{X} hat, d.h.

$$\|\mathbf{X} - \bar{\mathbf{X}}\| = \min\{\|\mathbf{X} - \mathbf{Y}\| \mid \mathbf{Y} \in \Gamma^2\},$$

also ist $\chi : \Gamma_c^1 \cup \Gamma_c^2 \rightarrow \Gamma^1 \cup \Gamma^2$ eine Abbildung der kleinsten Distanz (s. Abbildung 3.3). Damit definieren wir entsprechend die *kritische Richtung*

mit Länge 1 als

$$\mathbf{n}_c(\mathbf{X}) := \frac{\chi(\mathbf{X}) - \mathbf{X}}{\|\chi(\mathbf{X}) - \mathbf{X}\|}, \quad (3.28)$$

wobei im Falle $\mathbf{X} = \chi(\mathbf{X})$, d.h. im Falle des Kontaktes, für $\mathbf{n}_c(\mathbf{X})$ eine beliebige normierte Richtung gesetzt wird. Den Punkt $\bar{\mathbf{X}}$ nennen wir analog *kritischen Punkt*.

Bemerkung 3.19. Der Punkt $\bar{\mathbf{X}}$ heißt deshalb kritischer Punkt, da er wegen des kleinsten Abstandes zu \mathbf{X} der wohlmöglich nächste Punkt ist, der mit \mathbf{X} in Kontakt tritt. Der Vorteil dieser Formulierung ist, dass die kritische Richtung $\mathbf{n}_c(\mathbf{X})$ auch existiert, wenn der Rand eines Körpers nicht hinreichend glatt ist.

In den Koordinaten der Momentankonfiguration gilt $\mathbf{x} = \mathbf{X} + \mathbf{u}(\mathbf{X})$ für das Verschiebungsfeld \mathbf{u} und damit erhalten wir die *Nichtdurchdringungsbedingung*

$$(\bar{\mathbf{x}} - \mathbf{x}) \mathbf{n}_c(\mathbf{X}) \geq 0, \quad (3.29)$$

wobei $\bar{\mathbf{x}} := \bar{\mathbf{X}} + \mathbf{u}(\bar{\mathbf{X}})$ ist. Dies bedeutet, dass die Verbindung der Punkte in der Momentankonfiguration mit der kritischen Richtung einen Winkel $\alpha \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ einschließen muss, andernfalls läge $\bar{\mathbf{x}}$ „hinter“ \mathbf{x} , d.h. \mathcal{B}^1 wäre in \mathcal{B}^2 eingedrungen. Aus (3.29) folgt

$$\begin{aligned} 0 &\leq (\bar{\mathbf{x}} - \mathbf{x}) \mathbf{n}_c(\mathbf{X}) = (\bar{\mathbf{X}} + \mathbf{u}(\bar{\mathbf{X}}) - \mathbf{X} - \mathbf{u}(\mathbf{X})) \mathbf{n}_c(\mathbf{X}) \\ &= (\bar{\mathbf{X}} - \mathbf{X}) \underbrace{\frac{\bar{\mathbf{X}} - \mathbf{X}}{\|\bar{\mathbf{X}} - \mathbf{X}\|}}_{=\|\bar{\mathbf{X}} - \mathbf{X}\|=:g} + (\mathbf{u}(\bar{\mathbf{X}}) - \mathbf{u}(\mathbf{X})) \mathbf{n}_c(\mathbf{X}) \\ &= g + (\mathbf{u}(\chi(\mathbf{X})) - \mathbf{u}(\mathbf{X})) \mathbf{n}_c(\mathbf{X}), \end{aligned}$$

was uns die Nichtdurchdringungsbedingung bzgl. der Ausgangskonfiguration liefert, wobei wir die Funktion g auch als *Gap-Funktion* bezeichnen, da sie die Lücke zwischen den Körpern beschreibt.

Aufgrund von Voraussetzung 3.18 (b) gehen wir von kleinen Deformationen aus. Damit gilt unter anderem $\mathbf{X} \approx \mathbf{x}$, $\nabla_{\mathbf{X}} = \text{Grad}(\cdot) \approx \text{grad}(\cdot) = \nabla_{\mathbf{x}}$ (vgl. [Alt12] S. 122f). Daher schreiben wir im folgenden immer \mathbf{x} statt \mathbf{X} . Insgesamt lässt sich also die Nichtdurchdringungsbedingung schreiben als

$$(\mathbf{u} \circ \chi - \mathbf{u}) \cdot \mathbf{n}_c + g \geq 0 \quad \forall \mathbf{x} \in \Gamma_c, \quad (3.30)$$

wobei $\Gamma_c := \Gamma_c^1 \cup \Gamma_c^2$ ist.

Der Einfachheit halber wollen wir in dieser Arbeit noch weitere Forderungen an die beiden Körper \mathcal{B}^1 und \mathcal{B}^2 stellen. Wir fordern, dass die Ränder Γ^i hinreichend glatt sind. Daraus folgt, dass (3.30) auch für \mathbf{n}_c als

Einheitsnormale von \mathcal{B}^1 gilt. Weiter soll $\mathbf{u}(\bar{x}) \equiv \mathbf{0}$ gelten, d.h. falls \mathcal{B}^2 ein Verschiebungsfeld ungleich Null hat, können wir \mathbf{u} bzgl. $\Omega = \Omega^1 \cup \Omega^2$ auch als Relativverschiebung interpretieren.

Im numerischen Beispiel wollen wir später \mathcal{B}^1 als feste Ebene verwenden. Damit reduziert sich (3.30) auf

$$\mathbf{u} \cdot \mathbf{n} - g \leq 0 \quad \forall x \in \Gamma_c, \quad (3.31)$$

wobei wir auch $u_n := \mathbf{u} \cdot \mathbf{n}$ im Folgenden schreiben werden. Weiter muss auf dem Kontaktrand Γ_c die Normalkraft eine Druckkraft sein oder es herrscht Kräftegleichgewicht, d.h. für $\sigma_n := \mathbf{n} \cdot (\boldsymbol{\sigma} \cdot \mathbf{n})$, also die Spannung in Normalenrichtung, gilt

$$\sigma_n \leq 0 \quad \text{auf } \Gamma_c. \quad (3.32)$$

Wenn die Kontaktbedingung nicht aktiv ist, d.h. wenn keine Gleichheit gilt, muss Kräftegleichgewicht herrschen, d.h. in (3.32) gilt die Gleichheit. Zusammen erhält man die *Komplementaritätsbedingung*

$$(u_n - g) \sigma_n = 0 \quad \text{auf } \Gamma_c. \quad (3.33)$$

Laut der Voraussetzung (e) betrachten wir den Signorini-Kontakt (d.h. keine Reibung) und damit muss die Tangentialkraft auf dem Kontaktrand gleich Null sein, d.h.

$$\boldsymbol{\sigma}_t := \boldsymbol{\sigma} \cdot \mathbf{n} - \sigma_n \mathbf{n} = 0 \quad \text{auf } \Gamma_c. \quad (3.34)$$

Bilanzgleichungen, materialunabhängige Gleichungen

Wie in Kapitel 2.5 eingeführt, gelten auch hier die Gleichung (2.32) des Kräftegleichgewichts. Da wir laut Voraussetzung (b) von kleinen Deformationen ausgehen, gilt

$$\operatorname{div} \boldsymbol{\sigma} + \bar{\mathbf{b}} = \mathbf{0} \quad \text{in } \Omega. \quad (3.35)$$

Weiter gilt nach dem Cauchy-Theorem (2.29), dass die Spannung in Normalenrichtung auf der Rand Γ (speziell auf dem Neumann-Rand Γ_σ) von Ω gleich der von außen angebrachten Spannung $\bar{\mathbf{t}}$ ist, d.h.

$$\boldsymbol{\sigma} \cdot \mathbf{n} = \bar{\mathbf{t}} \quad \text{auf } \Gamma_\sigma. \quad (3.36)$$

Konstitutive Gleichungen

Da wir laut Voraussetzung 3.18 (b) von einem linear elastischen Material und kleinen Deformationen ausgehen, gilt ein linearer Zusammenhang bzgl.

3. Variationsungleichungen

der Spannung σ und Verzerrung ϵ , deswegen verwenden wir das Hooke'sche Gesetz (2.33). Die kleinen Deformationen erlauben die Verwendung des linearisierten Verzerrungstensors ϵ (vgl. (2.28)), d.h. mit einem 4stufigem Materialtensor $\mathcal{C} = (c_{ijkl})$ gilt

$$\sigma - \mathcal{C} : \epsilon = \mathbf{0} \quad \text{in } \Omega. \quad (3.37)$$

Zusammenfassend lässt sich das Signorini-Kontakt-Problem mit (3.31) bis (3.37) in der starken Formulierung beschreiben durch:

$$\operatorname{div} \sigma + \bar{b} = \mathbf{0} \quad \text{in } \Omega \quad (3.38a)$$

$$\sigma - \mathcal{C} : \epsilon = \mathbf{0} \quad \text{in } \Omega \quad (3.38b)$$

$$\sigma \cdot \mathbf{n} = \bar{t} \quad \text{auf } \Gamma_\sigma \quad (3.38c)$$

$$\mathbf{u} = \mathbf{0} \quad \text{auf } \Gamma_u \quad (3.38d)$$

$$\left. \begin{array}{l} u_n - g \leq 0 \\ \sigma_n \leq 0 \\ (u_n - g) \sigma_n = 0 \\ \sigma_t = \mathbf{0} \end{array} \right\} \text{auf } \Gamma_c \quad (3.38e)$$

An dieser Stelle sei kurz angemerkt, wie sich die Problemstellung (3.38) ändern würde, wenn wir ein Modell mit Reibung betrachten.

Bemerkung 3.20. Ein Kontaktmodell mit $\sigma_t \neq \mathbf{0}$ ist beispielsweise das Modell mit *Tresca-Reibung*. Für dieses Problem wird die letzte Bedingung aus (3.38e) durch die Bedingungen

$$\|\sigma_t\| \leq \mathcal{F}, \quad \sigma_t \mathbf{u}_t + \mathcal{F} \|\mathbf{u}_t\| = 0 \quad (3.39)$$

mit $\mathbf{u}_t := \mathbf{u} - u_n \mathbf{n}$, dem tangentialen Anteil des Verschiebungsfeldes \mathbf{u} , ersetzt. Hierbei ist $\mathcal{F} \geq 0$ eine Schranke für die Reibung. Gilt $\|\sigma_t\| < \mathcal{F}$, so folgt aus der zweiten Gleichung von (3.39), dass $\mathbf{u}_t = \mathbf{0}$ ist. Also kann $\mathbf{u}_t \neq \mathbf{0}$ nur gelten, wenn $\|\sigma_t\| = \mathcal{F}$ ist.

Mit $\mathcal{F} := \mu \sigma_n$ erhalten wir das *Reibungsgesetz von Coulomb*, wobei μ den aus der Mechanik bekannten Reibungskoeffizienten darstellt.

Da die Herleitung der zu diesem Problem äquivalenten Variationsungleichung zusätzliche mathematische Resultate erfordert, werden wir uns in der weiteren Herleitung auf das Signorini-Kontakt-Problem beziehen. Außerdem führt solch ein Problem auf eine sogenannte Variationsungleichung 2. Art, deren Lösung wir in dieser Arbeit nicht ausführen wollen.

3.2.2 Variationsformulierung des Signorini-Kontaktproblems

Erinnerung. Wegen Voraussetzung 3.18 (f) sei $\Omega \subset \mathbb{R}^2$ ein zweidimensionales Gebiet.

Wir betrachten das Signorini-Kontakt-Problem (3.38). Um hierfür eine Variationsformulierung herzuleiten, führen wir analog zu $H_0^1(\Omega)$ den Raum

$$H_{\Gamma_u}^1(\Omega) := \{\mathbf{v} \in (H^1(\Omega))^2 \mid \mathbf{v} = \mathbf{0} \text{ auf } \Gamma_u\} \quad (3.40)$$

der Funktionen, die auf dem Dirichlet-Rand Γ_u gleich Null sind. Weiter sei analog zur Teilmenge $K \subset H_0^1(\Omega)$ die zu betrachtende Teilmenge von $H_{\Gamma_u}^1(\Omega)$ definiert durch

$$\mathcal{K} := \{\mathbf{v} \in H_{\Gamma_u}^1(\Omega) \mid v_n - g \leq 0 \text{ auf } \Gamma_c\}, \quad (3.41)$$

welche die Funktionen enthält, die die Dirichlet-Rand- und KontaktRandbedingungen erfüllen. Die Menge \mathcal{K} ist analog zu Lemma 3.1 konvex und abgeschlossen (vgl. auch [KO88] Kapitel 6.2 und [CSW99] Proposition 3.2).

Weiter seien $\mathbf{u}, \mathbf{v} \in \mathcal{K}$, wobei \mathbf{u} die Lösung des Signorini-Kontakt-Problems darstellt und \mathbf{v} (häufig in den Ingenieurswissenschaften als *virtuelle Verschiebung* bezeichnet) eine beliebige Testfunktion ist. Dann gilt, dass

$$\mathbf{w} = \mathbf{v} - \mathbf{u} \in (H_{\Gamma_u}^1(\Omega))^2$$

auch eine Testfunktion ist, die wir mit (3.38a) multiplizieren und über Ω integrieren. Daraus folgt

$$\begin{aligned} 0 &= \int_{\Omega} (\operatorname{div} \boldsymbol{\sigma} + \bar{\mathbf{b}}) \cdot \mathbf{w} d\Omega = \int_{\Omega} \operatorname{div} \boldsymbol{\sigma} \cdot \mathbf{w} + \bar{\mathbf{b}} \cdot \mathbf{w} d\Omega \\ &= \int_{\Omega} \operatorname{div}(\mathbf{w} \cdot \boldsymbol{\sigma}) - \operatorname{grad} \mathbf{w} : \boldsymbol{\sigma} + \bar{\mathbf{b}} \cdot \mathbf{w} d\Omega \\ &= \int_{\Gamma} \underbrace{\mathbf{w} \cdot \boldsymbol{\sigma} \cdot \mathbf{n}}_{=\mathbf{w} \cdot \|\mathbf{n}\|^2 \cdot (\boldsymbol{\sigma} \cdot \mathbf{n})} d\Gamma - \int_{\Omega} \underbrace{\frac{1}{2}(\operatorname{grad} \mathbf{w} + \operatorname{grad}^T \mathbf{w}) : \boldsymbol{\sigma}}_{=\boldsymbol{\epsilon}(\mathbf{w})} d\Omega + \int_{\Omega} \bar{\mathbf{b}} \cdot \mathbf{w} d\Omega \\ &= \int_{\Gamma_c} w_n \sigma_n d\Gamma + \int_{\Gamma_{\sigma}} \bar{\mathbf{t}} \cdot \mathbf{w} d\Gamma - \int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{\epsilon}(\mathbf{w}) d\Omega + \int_{\Omega} \bar{\mathbf{b}} \cdot \mathbf{w} d\Omega, \end{aligned} \quad (3.42)$$

wobei im zweiten Schritt die Produktregel für die Divergenz (C.2) und darauf folgend der Integralsatz von Gauß verwendet wurde. Weiter wurde die disjunkte Aufteilung vom Rand $\Gamma = \Gamma_u \cup \Gamma_{\sigma} \cup \Gamma_c$ benutzt, wobei $\mathbf{w} = \mathbf{0}$ auf Γ_u und $\boldsymbol{\sigma} \mathbf{n} = \bar{\mathbf{t}}$ auf Γ_{σ} gilt.

Um weiter zu einer Variationsungleichung zu gelangen, betrachten wir das Integral über Γ_c . Es gilt für den Integranden

$$\begin{aligned} w_n \sigma_n &= (v_n - u_n) \sigma_n \stackrel{"+0"}{=} (v_n - g + g - u_n) \sigma_n \\ &= (v_n - g) \sigma_n - \underbrace{(u_n - g) \sigma_n}_{=0 \text{ auf } \Gamma_c} \\ &= \underbrace{(v_n - g)}_{\leq 0} \underbrace{\sigma_n}_{\leq 0} \geq 0 \end{aligned}$$

Damit ist das Integral größer gleich Null und mit (3.42) folgt

$$\begin{aligned} 0 &\geq \int_{\Gamma_\sigma} \bar{\mathbf{t}} \cdot \mathbf{w} d\Gamma - \int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{\varepsilon}(\mathbf{w}) d\Omega + \int_{\Omega} \bar{\mathbf{b}} \cdot \mathbf{w} d\Omega \\ \iff \int_{\Omega} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{v} - \mathbf{u}) d\Omega &\geq \int_{\Omega} \bar{\mathbf{b}} \cdot (\mathbf{v} - \mathbf{u}) d\Omega + \int_{\Gamma_\sigma} \bar{\mathbf{t}} \cdot (\mathbf{v} - \mathbf{u}) d\Gamma \quad (3.43) \end{aligned}$$

Wegen (3.38b) steht die Spannung aus (3.43) im linearen Zusammenhang mit der Verzerrung und kann durch $\boldsymbol{\sigma} = \mathcal{C} : \boldsymbol{\varepsilon}$ ausgedrückt werden. Mit der Bilinearform $a : H_{\Gamma_u}^1 \times H_{\Gamma_u}^1 \rightarrow \mathbb{R}$ und der Linearform $F : H_{\Gamma_u}^1 \rightarrow \mathbb{R}$ mit

$$a(\mathbf{u}, \mathbf{v}) := \int_{\Omega} \boldsymbol{\varepsilon}(\mathbf{u}) : \mathcal{C} : \boldsymbol{\varepsilon}(\mathbf{v}) d\Omega, \quad (3.44a)$$

$$F(\mathbf{v}) := \int_{\Omega} \bar{\mathbf{b}} \cdot \mathbf{v} d\Omega + \int_{\Gamma_\sigma} \bar{\mathbf{t}} \cdot \mathbf{v} d\Gamma \quad (3.44b)$$

lässt sich (3.43) in der altbekannten Form schreiben: Finde $\mathbf{u} \in \mathcal{K}$, so dass

$$a(\mathbf{u}, \mathbf{v} - \mathbf{u}) \geq F(\mathbf{v} - \mathbf{u}) \quad \forall \mathbf{v} \in \mathcal{K} \quad (3.45)$$

gilt. Wir stellen folgende Voraussetzungen an den Materialtensor $\mathcal{C} = (C_{ijkl})$ (vgl. [KO88] Bedingung (5.5)).

Voraussetzung 3.21. (a) Es sei $C_{ijkl} \in L^\infty(\Omega)$. Damit gilt, dass eine Konstante M existiert, so dass

$$\max_{1 \leq i,j,k,l \leq n} \|C_{ijkl}\|_{0,\infty} \leq M \quad (3.46)$$

gilt. \mathcal{C} ist also nach oben beschränkt.

(b) Der Materialtensor \mathcal{C} ist in folgendem Sinne symmetrisch:

$$C_{ijkl}(\mathbf{x}) = C_{klji}(\mathbf{x}) = C_{jikl}(\mathbf{x}) \text{ fast überall in } \Omega. \quad (3.47)$$

(c) Es existiert eine Konstante $m > 0$, so dass fast überall in Ω

$$C_{ijkl}(\mathbf{x}) \varepsilon_{ij} \varepsilon_{kl} \geq m \varepsilon_{ij} \varepsilon_{ij} \quad (3.48)$$

für jedes $\boldsymbol{\varepsilon} \in \mathbb{R}^{n \times n}$ mit $\varepsilon_{ij} = \varepsilon_{ji}$ gilt, d.h. der Materialtensor ist nach unten beschränkt.

Notation. (a) Der Parameter n aus Voraussetzung 3.21 gibt die Dimension des Verzerrungstensors bzw. Spannungstensors an, in unserem Fall also $n = 2$.

(b) In Voraussetzung 3.21 (c) wird die *Einstein'sche Summenkonvention* verwendet. Hierbei wird über doppelt vorkommende Indizes summiert, d.h.

$$\varepsilon_{ij} \varepsilon_{ij} = \sum_{i=1}^n \sum_{j=1}^n \varepsilon_{ij}^2.$$

Unter den Voraussetzungen 3.21 lässt sich für die Bilinearform $a(\cdot, \cdot)$ aus (3.44a) die Stetigkeit und Koerzivität zeigen, denn:

$$\begin{aligned}
 a(\mathbf{u}, \mathbf{v}) &= \int_{\Omega} \underbrace{\boldsymbol{\varepsilon}(\mathbf{u}) : \mathcal{C} : \boldsymbol{\varepsilon}(\mathbf{v})}_{=\varepsilon_{ij}(\mathbf{u}) C_{ijkl} \varepsilon_{kl}(\mathbf{v})} d\Omega \\
 &\stackrel{(3.46)}{\leq} M \int_{\Omega} \sum_{i,j=1}^2 \varepsilon_{ij}(\mathbf{u}) \cdot \sum_{k,l=1}^2 \varepsilon_{kl}(\mathbf{v}) d\Omega \\
 &\stackrel{\text{C.S.}}{\leq} M \left(\int_{\Omega} \left(\sum_{i,j=1}^2 \varepsilon_{ij}(\mathbf{u}) \right)^2 d\Omega \right)^{\frac{1}{2}} \left(\int_{\Omega} \left(\sum_{k,l=1}^2 \varepsilon_{kl}(\mathbf{v}) \right)^2 d\Omega \right)^{\frac{1}{2}} \\
 &= M \left(\int_{\Omega} |\operatorname{grad} \mathbf{u}|^2 d\Omega \right)^{\frac{1}{2}} \left(\int_{\Omega} |\operatorname{grad} \mathbf{v}|^2 d\Omega \right)^{\frac{1}{2}} \\
 &\leq M \|\mathbf{u}\|_1 \|\mathbf{v}\|_1,
 \end{aligned}$$

wobei wir im vorletzten Schritt verwendet haben, dass

$$\sum_{i,j=1}^2 \varepsilon_{ij}(\mathbf{u}) = \sum_{i,j=1}^2 \frac{1}{2} (u_{i,j} + \underbrace{u_{j,i}}_{=\partial_i u_j}) = \sum_{i,j=1}^2 u_{i,j}$$

ist. Für die Koerzivität rechnen wir unter Verwendung von Voraussetzung 3.21 (c) nach:

$$\begin{aligned}
 a(\mathbf{v}, \mathbf{v}) &= \int_{\Omega} \underbrace{\boldsymbol{\varepsilon}(\mathbf{v}) : \mathcal{C} : \boldsymbol{\varepsilon}(\mathbf{v})}_{=\varepsilon_{ij}(\mathbf{v}) C_{ijkl} \varepsilon_{kl}(\mathbf{v})} d\Omega \geq m \int_{\Omega} \underbrace{\varepsilon_{ij}(\mathbf{v}) \varepsilon_{ij}(\mathbf{v})}_{=\sum_{i,j=1}^2 v_{i,j}^2} d\Omega \\
 &= m \int_{\Omega} |\operatorname{grad} \mathbf{v}|^2 d\Omega \stackrel{\text{Satz 2.13}}{\geq} c \|\mathbf{v}\|_1^2
 \end{aligned}$$

mit einem $c > 0$. Die letzte Ungleichung folgt aus der zweiten *Korn'schen Ungleichung* (s. [Bra13] Kapitel VI, §3, Satz 3.3). Als letztes berechnen wir für die Linearform $F(\cdot)$ aus (3.44b) die Stetigkeit nach.

$$\begin{aligned}
 |F(\mathbf{v})| &= \left| \int_{\Omega} \bar{\mathbf{b}} \cdot \mathbf{v} d\Omega + \int_{\Gamma_\sigma} \bar{\mathbf{t}} \cdot \mathbf{v} d\Gamma \right| \\
 &\stackrel{\text{C.S.}}{\leq} \|\bar{\mathbf{b}}\|_0 \|\mathbf{v}\|_0 + \|\bar{\mathbf{t}}\|_{0,\Gamma_\sigma} \|\mathbf{v}\|_{0,\Gamma_\sigma} \\
 &\stackrel{\text{Theorem A.9}}{\leq} c (\|\bar{\mathbf{b}}\|_0 + \|\bar{\mathbf{t}}\|_{0,\Gamma_\sigma}) \|\mathbf{v}\|_1
 \end{aligned}$$

Also ist F unter der Voraussetzung, dass $\bar{\mathbf{b}} \in (L^2(\Omega))^2$ und $\bar{\mathbf{t}} \in (L^2(\Gamma_c))^2$ gilt (analog zum Modellproblem aus Kapitel 2.2), stetig. Damit folgt direkt die Aussage des nächsten Theorems.

Theorem 3.22. Es sei Voraussetzung 3.21 erfüllt sowie $\bar{\mathbf{b}} \in (L^2(\Omega))^2$ und $\bar{\mathbf{t}} \in (L^2(\Gamma_c))^2$. Dann hat die Variationsungleichung (3.43) bzw. (3.45) eine eindeutige Lösung.

Beweis. Unter den Voraussetzungen ist a eine stetige koerzive Bilinearform, sowie F stetig. Weiter ist \mathcal{K} abgeschlossen und konvex. Dann folgt aus Theorem 3.6 die Behauptung. \square

Wie schon erwähnt, gilt auch bei dieser Variationsungleichung die Äquivalenz zu einem Minimierungsproblem eines Energiefunktionalen.

Theorem 3.23. Es sei $\mathbf{v} : \Omega \rightarrow \mathbb{R}^2$ und \mathcal{K} wie oben definiert. Die Variationsungleichung (3.45) ist äquivalent zum Minimierungsproblem:

$$\min_{\mathbf{v} \in \mathcal{K}} J(\mathbf{v}) = \frac{1}{2}a(\mathbf{v}, \mathbf{v}) - F(\mathbf{v}) \quad (3.49)$$

mit der Bilinearform a aus (3.44a) und der Linearform F aus (3.44b).

Beweis. Es sei zu Beginn noch einmal bemerkt, dass \mathcal{K} konvex und abgeschlossen ist. Da a stetig und koerziv sowie F stetig ist, folgt direkt aus Lemma 2.10, dass J ein konkaves Funktional ist. Außerdem folgt analog zum Beweis von Lemma 2.11, dass J auch Gâteaux-differenzierbar ist und damit gilt wegen Satz A.11, dass das Minimierungsproblem (3.49) und die Variationsungleichung (3.45) äquivalent sind. \square

Bemerkung 3.24. Für das Kontaktproblem mit Tresca-Reibung (s. Bemerkung 3.20) gibt es ein zu (3.49) analoges Energiefunktional. Da durch die Reibung weitere Energie entsteht, bzw. verloren geht, ist dieses Funktional auch von der Reibung abhängig. Es ergibt sich dann:

$$J(\mathbf{v}) = \frac{1}{2}a(\mathbf{v}, \mathbf{v}) + j(\mathbf{v}) - F(\mathbf{v}) \quad (3.50)$$

mit dem *Reibungsfunktional*

$$j(\mathbf{v}) := \int_{\Gamma_c} \mathcal{F} \|\mathbf{v}_t\| d\Gamma. \quad (3.51)$$

Wegen des Reibungsfunktionalen j ist (3.50) nicht mehr Gâteaux-differenzierbar. Dennoch kann man eine zur Minimierungsaufgabe über (3.50) äquivalente Variationsungleichung herleiten (vgl. [Ste12a]). Eine solche Ungleichung nennen wir Variationsungleichung 2. Art und ist von der Form

$$a(\mathbf{u}, \mathbf{v} - \mathbf{u}) + j(\mathbf{v}) - j(\mathbf{u}) \geq F(\mathbf{v} - \mathbf{u}). \quad (3.52)$$

Allerdings ist der Beweis der Existenz einer Lösung für (3.52) mit den in dieser Arbeit aufgeführten Mitteln nicht möglich, weshalb wir vom reibungsfreien Fall ausgehen.

3.2.3 Lösung des Kontaktproblems mittels FEM

Um die Variationsungleichung (3.45) mittels der Finiten-Elemente-Methode zu lösen, betrachten wir analog zu \mathcal{S}_h den Raum der mehrdimensionalen linearen Ansatzfunktionen bzgl. einer quasi-uniformen Triangulierung \mathcal{T}_h

$$\mathcal{S}_h := \{\mathbf{v} \in (C^0(\Omega))^2 \mid \mathbf{v}|_T \in \mathcal{P}_1^2 \text{ für } T \in \mathcal{T}_h, \mathbf{v}|_{\Gamma_u} = \mathbf{0}\} \subset H_{\Gamma_u}^1(\Omega). \quad (3.53)$$

Mit einer Basis $\mathcal{B}_h := \{\psi_1, \dots, \psi_{2N}\}$ von \mathcal{S}_h lässt sich dann jedes Element $\mathbf{v}_h \in \mathcal{S}_h$ als Linearkombination schreiben

$$\mathbf{v}_h(\bar{x}) = \sum_{i=1}^{2N} x_i \psi_i(\bar{x}) \quad \forall \bar{x} \in \Omega \quad (3.54)$$

für genau ein $(x_1, \dots, x_{2N})^T =: \mathbf{x} \in \mathbb{R}^{2N}$. Betrachten wir analog zu Kapitel 3.1.3 die Variationsgleichung (3.45) diskret, so erhalten wir das Problem: Finde $\mathbf{u}_h \in \mathcal{S}_h$, so dass

$$a(\mathbf{u}_h, \mathbf{v}_h - \mathbf{u}_h) \geq F(\mathbf{v}_h - \mathbf{u}_h) \quad \forall \mathbf{v}_h \in \mathcal{K}_h. \quad (3.55)$$

mit $\mathcal{K}_h := \{\mathbf{v}_h \in \mathcal{S}_h \mid \mathbf{v}_h(\bar{x}_i) \cdot \mathbf{n} - g(\bar{x}_i) \leq 0 \text{ mit } \bar{x}_i \in \mathcal{N} \cap \Gamma_c\}$, d.h. die punktuelle Form (der Nebenbedingung) von \mathcal{K} . Auch hier stellt \mathcal{N} (mit $|\mathcal{N}| = N$) wieder die Menge der Knoten dar.

Analog zu $K_{\mathcal{S}}$ aus (3.13) können wir auch hier bzgl. einer Basis \mathcal{B}_h die Menge \mathcal{K}_h äquivalent durch den Koordinatenvektor $\mathbf{x} \in \mathbb{R}^{2N}$ ausdrücken, d.h.

$$\begin{aligned} \mathcal{K}_{\mathcal{S}} &:= \left\{ \mathbf{x} \in \mathbb{R}^{2N} \mid \sum_{j=1}^{2N} x_j \psi_j(\bar{x}_i) \cdot \mathbf{n} - g(\bar{x}_i) \leq 0 \text{ für } \bar{x}_i \in \mathcal{N} \cap \Gamma_c \right\} \\ &= \{\mathbf{x} \in \mathbb{R}^{2N} \mid B\mathbf{x} \geq \mathbf{c}\} \end{aligned}$$

mit $B = [-\psi_j(\bar{x}_i) \cdot \mathbf{n}(\bar{x}_i)]_{\bar{x}_i \in \mathcal{N} \cap \Gamma_c, 1 \leq j \leq 2N}$, $\mathbf{c} = [-g(\bar{x}_i)]_{\bar{x}_i \in \mathcal{N} \cap \Gamma_c}$.

Damit können wir das diskrete Problem in Matrixschreibweise wieder durch Einsetzen der Linearkombination (3.54) in die Variationsungleichung (3.55) erhalten. Das Problem liest sich dann als: Finde $\mathbf{x}^* \in \mathcal{K}_{\mathcal{S}}$, so dass

$$(A\mathbf{x}^* - \mathbf{b})^T(\mathbf{x} - \mathbf{x}^*) \geq 0 \quad \forall \mathbf{x} \in \mathcal{K}_{\mathcal{S}}, \quad (3.56)$$

wobei

$$\begin{aligned} A &= \left[\int_{\Omega} \boldsymbol{\varepsilon}(\psi_j) : \mathcal{C} : \boldsymbol{\varepsilon}(\psi_i) d\Omega \right]_{1 \leq i,j \leq 2N}, \\ \mathbf{b} &= \left[\int_{\Omega} \bar{\mathbf{b}} \cdot \psi_i d\Omega + \int_{\Gamma_{\sigma}} \bar{\mathbf{t}} \cdot \psi_i d\Gamma \right]_{1 \leq i \leq 2N} \end{aligned}$$

3. Variationsungleichungen

ist. Aus Bemerkung 3.13 folgt, dass die Variationsungleichung (3.56) äquivalent zu folgendem quadratischen Programm ist:

$$\min_{\boldsymbol{x} \in \mathbb{R}^{2N}} \frac{1}{2} \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{x} \quad \text{s.t. } \boldsymbol{B} \boldsymbol{x} \geq \boldsymbol{c}, \quad (3.57)$$

d.h. da wegen der Koerzivität der Bilinearform $a(\cdot, \cdot)$ das Problem (3.57) konvex ist, kann dieses wieder mithilfe des in Anhang B.2 vorgestellten gelöst werden.

Bemerkung 3.25. Wir erhalten beim Kontaktproblem im Vergleich zum Hindernisproblem doppelt so große Vektoren $\boldsymbol{x}, \boldsymbol{b}$ bzw. eine doppelt so große Matrix \boldsymbol{A} , da wir an jedem der N Knoten ein Verschiebungsfeld, d.h. eine Verschiebung in x - und y -Richtung, berechnen.

Kapitel 4

Ein hierarchischer Fehlerschätzer für Hindernisprobleme

Die adaptive Netzverfeinerung (s. Kapitel 2.4) ist gerade auch bei Hindernis- und Kontaktproblemen von großem Vorteil. Wie wir in Kapitel 3 gesehen haben, führt die Lösung von solchen Problemen auf quadratische Programme, die wir mit zeitaufwändigen Verfahren lösen müssen. An dieser Stelle sei bemerkt, dass man Variationsungleichungen auch mit anderen Verfahren – häufig üblich ist z.B. die *Penalty-Methode* – lösen kann, die jedoch nicht weniger aufwendig sind.

Um nun eine adaptive Verfeinerungsstrategie herleiten zu können, müssen wir zunächst einen geeigneten Fehlerschätzer für unser Modellproblem (3.1) erhalten. Dabei ist „geeignet“ in dem Sinne zu verstehen, dass der Schätzer unseren echten Fehler nach oben und unten beschränkt. Weiter wollen wir mit dem Fehler a posteriori nach jedem Verfeinerungsschritt den Fehler in der nächsten Verfeinerung abschätzen. Mit diesen Fragestellungen werden wir uns in diesem Kapitel beschäftigen und einen ersten adaptiven Algorithmus sowie die Übertragung auf Kontaktprobleme zeigen.

Es sei bemerkt, dass man als a posteriori Fehlerschätzer für Hindernisprobleme in der Literatur häufig residuale Schätzer wiederfinden kann. Wir wollen uns jedoch hier mit einem hierarchischen Fehlerschätzer beschäftigen, dessen grundlegende Idee auch schon in Kapitel 2.4.1 dargestellt wurde.

Dieses Kapitel basiert größtenteils auf [ZVKG11].

4.1 Herleitung von einem hierarchischen a posteriori Fehlerschätzer

Bevor wir uns detailliert der Herleitung eines hierarchischen a posteriori Fehlerschätzers widmen, wollen wir eine Generalvoraussetzung für das gesamte Kapitel 4 stellen.

Voraussetzung 4.1. Das Hindernis wird durch eine stückweise lineare steigige Funktion ψ beschrieben.

Für nichtstetige oder auch glatte Hindernisse sind analoge Aussagen beweisbar, die jedoch schwerer zu zeigen sind. Für einen residualen Schätzer kann man das Vorgehen für glatte Hindernisse auch in [Pag10] finden.

4.1.1 Diskretisierung des Defektproblems

Es sei \mathcal{B}_h eine nodale Basis bzgl. einer quasi-uniformen Triangulierung \mathcal{T}_h für \mathcal{S}_h (s. Kapitel 2.3), dem Raum der stückweise linearen Funktionen über \mathcal{T}_h . Weiter sei K_h wie in Kapitel 3.1.3 definiert

$$K_h = \{v_h \in \mathcal{S}_h \mid v_h(p) \geq \psi(p) \forall p \in \mathcal{N} \cap \Omega\},$$

wobei \mathcal{N} wieder die Menge der Knoten von \mathcal{T}_h darstellt. Wir betrachten nun wieder die diskrete Variationsungleichung (3.12): Finde $u_h \in K_h$ mit

$$a(u_h, v_h - u_h) \geq (f, v_h - u_h) \quad \forall v_h \in K_h$$

oder äquivalent die Minimierung des Funktionalen $J(v) = \frac{1}{2}a(v, v) - (f, v)$ über K_h , d.h.

$$u_h \in K_h : \quad J(u_h) \leq J(v_h) \quad \forall v_h \in K_h. \quad (4.1)$$

Wegen Voraussetzung 4.1, dass ψ stückweise linear ist, gilt $K_h \subset K$, da die linearen Ansatzfunktionen nicht nur punktuell, sondern auch kontinuierlich die Nebenbedingung erfüllen. Damit ist (3.12) eine konforme Finite-Elemente-Methode; nichtkonforme Finite-Elemente würden z.B. durch ein nichtstetiges Hindernis erzeugt werden. Diese sollen jedoch nicht Teil dieser Arbeit sein.

Wir werden einen a posteriori Fehlerschätzer für den Fehler bzgl. der Funktionswerte der Funktionale $J(u)$, $J(u_h)$ herleiten. Hierfür gilt

$$J(u_h) - J(u) \geq 0,$$

denn aus den beiden Minimierungsproblemen über K und K_h folgt

$$J(u) \leq J(v) \forall v \in K, \quad J(u_h) \leq J(v_h) \forall v_h \in K_h.$$

Da $K_h \subset K$ gilt, gilt insbesondere $J(u) \leq J(v_h)$ für alle $v_h \in K_h$. Setzen wir $v_h = u_h$, so folgt

$$J(u) \leq J(u_h) \iff J(u_h) - J(u) \geq 0.$$

Bemerkung 4.2. Gilt $\psi = -\infty$, d.h. ist kein Hindernis vorhanden, so folgt

$$\begin{aligned}
 J(u_h) - J(u) &= \frac{1}{2}a(u_h, u_h) - (f, u_h) - \left(\frac{1}{2}a(u, u) - (f, u) \right) \\
 &= \frac{1}{2}a(u_h, u_h) - (f, u_h) - \frac{1}{2}a(u, u) + (f, u) \\
 &\quad + \overbrace{(a(u, u - u_h) - (f, u - u_h))}^{=0} \\
 &= \frac{1}{2}a(u_h, u_h) - \frac{1}{2}a(u, u) + a(u, u - u_h) \\
 &= \frac{1}{2}a(u_h, u_h) - \frac{1}{2}a(u, u) + a(u, u) - a(u, u_h) \\
 &= \frac{1}{2}(a(u_h, u_h) + a(u, u) - 2a(u, u_h)) \\
 &= \frac{1}{2}a(u_h - u, u_h - u) = \frac{1}{2}\|u_h - u\|_E^2.
 \end{aligned}$$

Ist nun ein $\psi > -\infty$ gegeben, dann addieren wir im zweiten Schritt nicht mehr Null, sondern es gilt für den Term

$$a(u, u - u_h) - (f, u - u_h) \leq 0$$

und damit gilt $J(u_h) - J(u) \geq \frac{1}{2}\|u_h - u\|_E^2$, d.h. eine obere Schranke des Fehlers im Funktional schätzt auch den Fehler zwischen exakter und approximierter Lösung in der Energienorm ab.

Notation. Um im Folgenden den hierarchischen Split leichter beschreiben zu können, schreiben wir für die Galerkin-Lösung u_h die Notation $u_{\mathcal{S}}$, um auszudrücken, dass diese im linearen Ansatzraum \mathcal{S}_h liegt. Analog sind die im Weiteren übrigen verwendeten Indizes zu verstehen.

Zudem werden wir die für die Energienorm die Notation

$$a(v, v)^{\frac{1}{2}} = \|v\|$$

für alle $v \in H^1(\Omega)$, also ohne Index, verwenden. Auch in diesem Kapitel werden wir wieder $(u, v) = (u, v)_0$ für alle $u, v \in H^1(\Omega)$ schreiben.

Zur Herleitung des hierarchischen Fehlerschätzers führen wir die Fehlerfunktion $e = u - u_{\mathcal{S}}$ ein, die den exakten Fehler angibt. Weiter sei

$$\mathcal{I}(v) = \frac{1}{2}a(v, v) - \rho_{\mathcal{S}}(v) \text{ mit } \rho_{\mathcal{S}}(v) = (f, v) - a(u_{\mathcal{S}}, v), \quad v \in H_0^1(\Omega).$$

Bemerkung 4.3. (a) Die Linearform $\rho_{\mathcal{S}}$ stellt das Residuum der Variationsgleichung (d.h. ohne Hindernis) dar.

(b) Nach dem Darstellungssatz von Riesz existiert ein $v^* \in H_0^1(\Omega)$, so dass

$$(v^*, v) = \rho_{\mathcal{S}}(v) \quad \forall v \in H_0^1(\Omega)$$

ist. Wir können also v^* als Lagrange-Multiplikator bzgl. der Nebenbedingung $v \geq \psi$ interpretieren.

Damit führen wir das Defektproblem für das Hindernisproblem ein, welches der exakte Fehler e löst.

Satz 4.4 (Lösung des Defektproblems). *Mit den obigen Bezeichnungen löst die Fehlerfunktion e folgendes Defektproblem:*

$$e \in \mathcal{A} : \quad \mathcal{I}(e) \leq \mathcal{I}(v) \quad \forall v \in \mathcal{A}, \quad (4.2)$$

wobei $\mathcal{A} := \{v \in H_0^1(\Omega) \mid v \geq \psi - u_{\mathcal{S}}\} = -u_{\mathcal{S}} + K$.

Beweis. Es sei u die Lösung von (3.2) und $u_{\mathcal{S}}$ die Lösung von (4.1). Dann gilt

$$\begin{aligned} u \in K : \quad & J(u) \leq J(\tilde{v}) \quad \forall \tilde{v} \in K \\ \iff u \in K : \quad & J(u) - J(u_{\mathcal{S}}) \leq J(\tilde{v}) - J(u_{\mathcal{S}}) \quad \forall \tilde{v} \in K. \end{aligned} \quad (*)$$

Wir rechnen für die linke Seite nach, dass gilt

$$\begin{aligned} J(u) - J(u_{\mathcal{S}}) &= \frac{1}{2}a(u, u) - (f, u) - \left(\frac{1}{2}a(u_{\mathcal{S}}, u_{\mathcal{S}}) - (f, u_{\mathcal{S}}) \right) \\ &= \frac{1}{2}a(u, u) + \frac{1}{2}a(u_{\mathcal{S}}, u_{\mathcal{S}}) - a(u_{\mathcal{S}}, u_{\mathcal{S}}) - (f, u - u_{\mathcal{S}}) \\ &= \frac{1}{2}a(u, u) + \frac{1}{2}a(u_{\mathcal{S}}, u_{\mathcal{S}}) - a(u_{\mathcal{S}}, u) \\ &\quad - ((f, u - u_{\mathcal{S}}) - a(u_{\mathcal{S}}, u - u_{\mathcal{S}})) \\ &= \frac{1}{2}a(u - u_{\mathcal{S}}, u - u_{\mathcal{S}}) - \rho_{\mathcal{S}}(u - u_{\mathcal{S}}) \\ &= \frac{1}{2}a(e, e) - \rho_{\mathcal{S}}(e) = \mathcal{I}(e). \end{aligned}$$

Analog gilt für die rechte Seite $J(\tilde{v}) - J(u_{\mathcal{S}}) = \mathcal{I}(\tilde{v} - u_{\mathcal{S}})$. Mit $v := \tilde{v} - u_{\mathcal{S}}$ gilt $v \in \mathcal{A}$ und damit ist $(*)$ äquivalent zu: Finde $e \in \mathcal{A}$, so dass

$$\mathcal{I}(e) \leq \mathcal{I}(v) \quad \forall v \in \mathcal{A}. \quad \square$$

Korollar 4.5. *Das Problem (4.2) ist äquivalent zur Variationsungleichung: Finde $e \in \mathcal{A}$ mit*

$$a(e, v - e) \geq \rho_{\mathcal{S}}(v - e) \quad \forall v \in \mathcal{A}. \quad (4.3)$$

Beweis. Analog zu Lemma 3.1 lässt sich zeigen, dass \mathcal{A} abgeschlossen und konvex ist. Aus Lemma 2.10 und 2.11 folgt, dass \mathcal{I} konvex und Gâteaux-differenzierbar ist und mit Satz A.11 folgt dann die Behauptung. \square

Bemerkung. (a) Da ψ stückweise linear ist und $\psi - u_{\mathcal{S}} \leq 0$ gilt, folgt $0 \in \mathcal{A}$, d.h. das „gewünschte“ Ergebnis für den Fehler e liegt in der für das Defektproblem betrachteten Menge.

(b) Wir werden noch zeigen, dass $\rho_{\mathcal{S}}$ eine Schlüsselgröße für die a posteriori Abschätzung darstellt.

Die Herleitung des a posteriori Schätzers vollzieht sich jetzt analog zu Kapitel 2.4.1 in zwei Schritten.

- (i) Diskretisiere (4.3) bzgl. einer Erweiterung von \mathcal{S}_h (hier mit quadratischen Funktionen), so dass e hinreichend genau approximiert wird.
- (ii) Teile den neuen Raum so auf, dass (4.3) lokal in der Erweiterung exakt gelöst werden kann.

Als Erweiterung von \mathcal{S}_h betrachten wir einen Raum \mathcal{Q}_h mit $\mathcal{S}_h \subset \mathcal{Q}_h$, wobei

$$\mathcal{Q}_h := \{v \in C^0(\Omega) \mid v|_T \in \mathcal{P}_2 \text{ für } T \in \mathcal{T}_h, v|_{\partial\Omega} = 0\}$$

ist, d.h. der Raum der stückweise quadratischen Funktionen über einer quasi-uniformen Zerlegung \mathcal{T}_h . Damit definieren wir $\mathcal{N}_{\mathcal{Q}} := \mathcal{N} \cup \{x_E \mid E \in \mathcal{E}\}$, wobei x_E den Mittelpunkt der Kante E bezeichne und \mathcal{E} somit die Menge aller Kanten ist. Daraus ergibt sich \mathcal{A} über \mathcal{Q}_h diskret als

$$\mathcal{A}_{\mathcal{Q}} := \{v \in \mathcal{Q}_h \mid v(p) \geq \psi(p) - u_{\mathcal{S}}(p) \forall p \in \mathcal{N}_{\mathcal{Q}} \cap \Omega\} \quad (4.4)$$

und im Bezug zu (4.4) erhalten wir somit das diskrete Defektproblem

$$e_{\mathcal{Q}} \in \mathcal{A}_{\mathcal{Q}} : \quad a(e_{\mathcal{Q}}, v - e_{\mathcal{Q}}) \geq \rho_{\mathcal{S}}(v - e_{\mathcal{Q}}) \quad \forall v \in \mathcal{A}_{\mathcal{Q}}. \quad (4.5)$$

Bemerkung 4.6. Im Allgemeinen gilt hierbei nicht $\mathcal{A}_{\mathcal{Q}} \subset \mathcal{A}$. So kann man sich anschaulich eine quadratische Funktion $v_{\mathcal{Q}} \in \mathcal{A}_{\mathcal{Q}}$ vorstellen, die allerdings zwischen den übereinstimmenden Werten aufgrund ihrer Krümmung das lineare Hindernis aus \mathcal{A} durchdringt (vgl. das eindimensionale Beispiel aus Abbildung 4.1).

Als hierarchischen Split des Raumes \mathcal{Q}_h verwenden wir $\mathcal{Q}_h = \mathcal{S}_h + \mathcal{V}_h$, wobei $\mathcal{V}_h := \{\phi_E \mid E \in \mathcal{E}\}$ der Raum der quadratischen *Bubble-Funktionen* ist, wobei solch eine Bubble-Funktion ϕ_E bzgl. der Eck- und Kantenmittelpunkte eines Dreiecks definiert ist durch

$$\phi_E(p) = \delta_{x_E, p} = \begin{cases} 1, & p = x_E \\ 0, & \text{sonst} \end{cases}.$$

Es stellt sich nun die Frage, ob \mathcal{Q}_h als direkte Summe der beiden Räume \mathcal{S}_h und \mathcal{V}_h geschrieben werden kann.

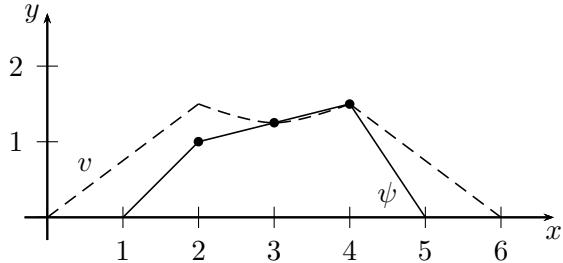


Abbildung 4.1: Beispiel eines affinen Hindernisses ψ mit $v \in \mathcal{A}_Q$ in \mathbb{R}

Satz 4.7. Mit den oben verwendeten Notationen gilt $\mathcal{Q}_h = \mathcal{S}_h \oplus \mathcal{V}_h$.

Beweis. Wir zeigen, dass $\mathcal{Q}_h = \mathcal{S}_h \oplus \mathcal{V}_h$ auf dem Referenzdreieck gilt und damit gilt es auch für beliebige Dreiecke $T \in \mathcal{T}_h$, da ein allgemeines Dreieck T aus dem Referenzelement \tilde{T} (vgl. Kapitel 2.3) durch affine Transformation hervorgeht.

Auf dem Referenzelement \tilde{T} ist $\{\phi_1, \phi_2, \phi_3\}$ eine Basis von \mathcal{S}_h mit

$$\phi_1(\xi, \eta) = 1 - \xi - \eta, \quad \phi_2(\xi, \eta) = \xi, \quad \phi_3(\xi, \eta) = \eta$$

und $\{\phi_4, \phi_5, \phi_6\}$ eine Basis von \mathcal{V}_h mit

$$\phi_4(\xi, \eta) = 4\xi(1 - \xi - \eta), \quad \phi_5(\xi, \eta) = 4\xi\eta, \quad \phi_6(\xi, \eta) = 4\eta(1 - \xi - \eta).$$

Damit ist $\{\phi_1, \dots, \phi_6\}$ ein Erzeugendensystem von \mathcal{Q}_h , da jedes Element

$$a_0 + a_1\xi + a_2\eta + a_3\xi^2 + a_4\xi\eta + a_5\eta^2 \in \mathcal{Q}_h$$

als Linearkombination aus den Funktionen beschrieben werden kann (ϕ_1 bis ϕ_6 enthalten alle vorkommenden Summanden eines Polynom 2. Grades). Außerdem ist leicht nachzurechnen, dass die Funktionen $\phi_i, i = 1, \dots, 6$, linear unabhängig sind und damit gilt

$$\mathcal{Q}_h = \text{span}\{\phi_1, \dots, \phi_6\}.$$

Aus der linearen Unabhängigkeit folgt damit auch $\mathcal{S}_h \cap \mathcal{V}_h = \{0\}$ gilt und damit die Behauptung. \square

Satz 4.7 erlaubt es uns also, jedes Element $v_Q \in \mathcal{Q}_h$ als $v_Q = v_S + v_V$ mit $v_S \in \mathcal{S}_h, v_V \in \mathcal{V}_h$ schreiben zu können. Aus diesem Grund führen wir folgende Bilinearform ein:

$$a_Q(v, w) := a(v_S, w_S) + \sum_{E \in \mathcal{E}} u_V(x_E) w_V(x_E) a(\phi_E, \phi_E) \quad \forall v, w \in \mathcal{Q}_h,$$

welche aufgrund der Eigenschaften der direkten Summe von \mathcal{S}_h und \mathcal{V}_h wohldefiniert ist. Dabei erhält man a_Q durch Entkopplung von a bzgl. der Räume

\mathcal{S}_h und \mathcal{V}_h und anschließender „Diagonalisierung“ auf \mathcal{V}_h in dem Sinne, dass wir nur noch die Koordinaten der Elemente v, w auf \mathcal{V}_h betrachten, deren Basisfunktionen ϕ_E gleich sind. Wenn wir mittels $a_{\mathcal{Q}}$ eine Matrix bzgl. der Basisfunktionen ϕ_E mit dem bekannten Muster aus dem Galerkinverfahren aufstellen, ergibt sich eine Diagonalmatrix.

Es ist legitim und sinnvoll mit der im Vergleich zur Bilinearform a einfacheren Form $a_{\mathcal{Q}}$ weiterzuarbeiten, was der folgende Satz besagt.

Satz 4.8. *Die zu $a_{\mathcal{Q}}$ assoziierte Energienorm*

$$\|v\|_{\mathcal{Q}} := a_{\mathcal{Q}}(v, v)^{\frac{1}{2}}, \quad v \in \mathcal{Q}_h$$

ist äquivalent zur Energienorm $a(\cdot, \cdot)^{\frac{1}{2}} = \|\cdot\|$, d.h. es gibt Konstanten c_1, c_2 (die insbesondere nur von der Quasi-Uniformität von \mathcal{T}_h abhängen), so dass

$$c_1 \|v\| \leq \|v\|_{\mathcal{Q}} \leq c_2 \|v\|, \quad \forall v \in \mathcal{Q}_h.$$

Beweis. Die Aussage folgt aus Theorem 4.1 bzw. Bemerkung 4.3 in [HK92] zusammen mit dem Lemma auf Seite 14 in [DLY89]. \square

Deshalb führen wir die approximierte Energie

$$\mathcal{I}_{\mathcal{Q}}(v) := \frac{1}{2}a_{\mathcal{Q}}(v, v) - \rho_{\mathcal{S}}(v), \quad v \in \mathcal{Q}_h \quad (4.6)$$

ein, das damit verbundene Defektproblem ist allerdings noch durch die Nebenbedingung aus $\mathcal{A}_{\mathcal{Q}}$ mit \mathcal{S}_h gekoppelt und daher noch nicht alleine auf die Raumerweiterung \mathcal{V}_h bezogen. Als Abhilfe ignorieren wir einfach die linearen Beiträge in $\mathcal{A}_{\mathcal{Q}}$ und führen eine echte Teilmenge

$$\mathcal{A}_{\mathcal{V}} := \{v \in \mathcal{V}_h \mid v(x_E) \geq \psi(x_E) - u_{\mathcal{S}}(x_E) \forall E \in \mathcal{E}\} \quad (4.7)$$

von $\mathcal{A}_{\mathcal{Q}}$ ein. Zusammen mit (4.6) und (4.7) erhalten wir dann das lokale diskrete Defektproblem

$$\varepsilon_{\mathcal{V}} \in \mathcal{A}_{\mathcal{V}} : \quad \mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}}) \leq \mathcal{I}_{\mathcal{Q}}(v) \quad \forall v \in \mathcal{A}_{\mathcal{V}} \quad (4.8)$$

bzw. die dazu äquivalente Variationsungleichung

$$\varepsilon_{\mathcal{V}} \in \mathcal{A}_{\mathcal{V}} : \quad a_{\mathcal{Q}}(\varepsilon_{\mathcal{V}}, v - \varepsilon_{\mathcal{V}}) \geq \rho_{\mathcal{S}}(v - \varepsilon_{\mathcal{V}}) \quad \forall v \in \mathcal{A}_{\mathcal{V}}. \quad (4.9)$$

Bemerkung 4.9. (a) Da ψ stetig stückweise linear ist und somit $u_{\mathcal{S}} \geq \psi$ gilt, folgt $0 \in \mathcal{A}_{\mathcal{V}}$. Damit ist auch hier die gewünschte Lösung für $\varepsilon_{\mathcal{V}}$ in $\mathcal{A}_{\mathcal{V}}$ enthalten

(b) Auch für $\mathcal{A}_{\mathcal{V}}$ lässt sich mit analogem Vorgehen zu Lemma 3.1 die Konvexität zeigen.

Lemma 4.10. Das Energiefunktional \mathcal{I}_Q ist konvex.

Beweis. Da a eine stetige koerzive Bilinearform, werden aufgrund der Konstruktion von a_Q diese Eigenschaften auch auf a_Q übertragen. Weiterhin ist leicht zu überprüfen, dass ρ_S eine stetige Linearform ist. Dann folgt aus Lemma 2.10 direkt die Behauptung. \square

Das lokale diskrete Defektproblem (4.9) ist sogar exakt lösbar. Die Lösung des Problems gibt uns der nächste Satz.

Satz 4.11. Die Lösung von (4.8) bzw. (4.9) ist explizit gegeben durch

$$\varepsilon_V(x_E) = \frac{\max\{-d_E, \rho_E\}}{\|\phi_E\|} \quad (4.10)$$

wobei

$$d_E = (u_S(x_E) - \psi(x_E))\|\phi_E\| \geq 0, \quad \rho_E = \frac{\rho_S(\phi_E)}{\|\phi_E\|}. \quad (4.11)$$

Beweis. Es sei $M = |\mathcal{E}|$ die Anzahl der Kanten. Zunächst berechnen wir zur besseren Übersicht $\varepsilon_V(x_E)$ konkret, d.h.

$$\begin{aligned} \varepsilon_V(x_E) &= \frac{\max\{-d_E, \rho_E\}}{\|\phi_E\|} \\ &= \frac{\max\left\{(\psi(x_E) - u_S(x_E))\|\phi_E\|, \frac{\rho_S(\phi_E)}{\|\phi_E\|}\right\}}{\|\phi_E\|} \\ &= \max\left\{\psi(x_E) - u_S(x_E), \frac{\rho_S(\phi_E)}{\|\phi_E\|^2}\right\} \\ &= \max\left\{\psi(x_E) - u_S(x_E), \frac{1}{\|\phi_E\|^2}((f, \phi_E) - a(u_S, \phi_E))\right\}. \end{aligned} \quad (4.12)$$

Da $\varepsilon_V = \sum_{E \in \mathcal{E}} \varepsilon_V(x_E) \phi_E$ ist, können wir (4.8) bzgl. der Basis $\{\phi_E \mid E \in \mathcal{E}\}$ von \mathcal{V}_h diskret schreiben als

$$\min_{\mathbf{v} \in \mathbb{R}^M} \frac{1}{2} \mathbf{v}^T D \mathbf{v} - \mathbf{g}^T \mathbf{v} \quad \text{s.t.} \quad \mathbf{v} \geq \boldsymbol{\psi} - \mathbf{u}_S,$$

wobei $\mathbf{v} = [\varepsilon_V(x_{E_i})]_{1 \leq i \leq M}$, $D = \text{diag}(a(\phi_{E_1}, \phi_{E_1}), \dots, a(\phi_{E_M}, \phi_{E_M}))$, $\mathbf{g} = [(f, \phi_{E_i}) - a(u_S, \phi_{E_i})]_{1 \leq i \leq M}$, $\boldsymbol{\psi} = [\psi(x_{E_i})]_{1 \leq i \leq M}$ und $\mathbf{u}_S = [u_S(x_{E_i})]_{1 \leq i \leq M}$. Da \mathcal{A}_V und \mathcal{I}_Q konvex sind, existiert ein Minimum $\mathbf{v}^* \in \mathcal{A}_V$ von \mathcal{I}_Q , das die KKT-Bedingungen erfüllt. Damit gilt

$$D \mathbf{v} - \mathbf{g} - \boldsymbol{\lambda} = \mathbf{0}, \quad (4.13a)$$

$$\boldsymbol{\lambda} \geq 0, \quad (4.13b)$$

$$\mathbf{v} \geq \boldsymbol{\psi} - \mathbf{u}_S, \quad (4.13c)$$

$$\lambda_i (\mathbf{v} - \boldsymbol{\psi} + \mathbf{u}_S)_i = 0 \quad \forall i = 1, \dots, M. \quad (4.13d)$$

Es sei $k \in \{1, \dots, M\}$ beliebig.

Fall 1: Gilt $\lambda_k = 0$, so folgt aus (4.13a)

$$\varepsilon_{\mathcal{V}}(x_{E_k}) = v_k = \frac{g_k}{a(\phi_{E_k}, \phi_{E_k})} = \frac{1}{\|\phi_{E_k}\|^2} ((f, \phi_{E_k}) - a(u_{\mathcal{S}}, \phi_{E_k})).$$

Fall 2: Gilt $\lambda_k \neq 0$, dann folgt wegen (4.13d)

$$\varepsilon_{\mathcal{V}}(x_{E_k}) = v_k = (\psi - u_{\mathcal{S}})_k = \psi(x_{E_k}) - u_{\mathcal{S}}(x_{E_k}).$$

Insgesamt folgt mit (4.13c) und (4.12) die Behauptung. \square

Als a posteriori Fehlerschätzer werden wir im Folgenden

$$-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}}) = -\frac{1}{2}a_{\mathcal{Q}}(\varepsilon_{\mathcal{V}}, \varepsilon_{\mathcal{V}}) + \rho_{\mathcal{S}}(\varepsilon_{\mathcal{V}})$$

betrachten und zeigen, dass dieser äquivalent zum Fehler im Funktional $J(u_{\mathcal{S}}) - J(u)$ ist (vgl. Kapitel 4.1.4 und 4.1.5). Zunächst wollen wir aber eine Einführung der lokalen Anteile des Fehlerschätzers $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}})$ bieten.

4.1.2 Lokaler Anteil des Fehlerschätzers

Notation. (a) Wir schreiben im Folgenden „ \lesssim “ statt „ $\leq C$ “, wenn die Konstante C nur von der Quasi-Uniformität von \mathcal{T}_h abhängt.

(b) Weiter schreiben wir „ $A \approx B$ “ für „ $A \lesssim B$ “ und „ $B \lesssim A$ “.

Um die lokalen Anteile des Fehlerschätzers herzuleiten, zeigen wir zunächst ein paar Eigenschaften der Fehlerfunktion $e = u - u_{\mathcal{S}}$.

Lemma 4.12. *Die Fehlerfunktion $e = u - u_{\mathcal{S}}$ erfüllt die Ungleichungen*

$$\frac{1}{2}\|e\|^2 \leq \frac{1}{2}\rho_{\mathcal{S}}(e) \leq -\mathcal{I}(e) \leq \rho_{\mathcal{S}}(e). \quad (4.14)$$

Beweis. Wir erinnern uns, dass

$$-\mathcal{I}(e) := -\frac{1}{2} \underbrace{a(e, e)}_{\geq 0} + \rho_{\mathcal{S}}(e) \leq \rho_{\mathcal{S}}(e),$$

da a koerativ ist. Dann gilt weiter

$$\begin{aligned} -\mathcal{I}(e) &= -\frac{1}{2}a(e, e) + \rho_{\mathcal{S}}(e) \\ &= -\frac{1}{2}a(u - u_{\mathcal{S}}, e) + \rho_{\mathcal{S}}(e) \\ &= -\frac{1}{2}a(u, e) \underbrace{\frac{1}{2}a(u_{\mathcal{S}}, e) - \frac{1}{2}(f, e)}_{=-\frac{1}{2}\rho_{\mathcal{S}}(e)} + \frac{1}{2}(f, e) + \rho_{\mathcal{S}}(e) \\ &= -\frac{1}{2} \underbrace{(a(u, u - u_{\mathcal{S}}) - (f, u - u_{\mathcal{S}}))}_{\leq 0} + \frac{1}{2}\rho_{\mathcal{S}}(e) \geq \frac{1}{2}\rho_{\mathcal{S}}(e). \end{aligned}$$

Es bleibt also die erste Ungleichung von (4.14) zu zeigen. Wir rechnen nach, dass

$$\begin{aligned}\frac{1}{2}\rho_S(e) &= \frac{1}{2}(f, e) - \frac{1}{2}a(u_S, e) \\ &= \frac{1}{2}\underbrace{((f, u - u_S) - a(u, u - u_S))}_{\geq 0} + a(u - u_S, e) \\ &\geq \frac{1}{2}a(u - u_S, e) = \frac{1}{2}a(e, e) = \frac{1}{2}\|e\|^2\end{aligned}$$

gilt, womit (4.14) insgesamt bewiesen ist. \square

Korollar 4.13. *Für die Lösungen e_Q, ε_V von (4.5) und (4.9) gilt*

$$\frac{1}{2}\|e_Q\|^2 \leq \frac{1}{2}\rho_S(e_Q) \leq -\mathcal{I}(e_Q) \leq \rho_S(e_Q), \quad (4.15)$$

$$\frac{1}{2}\|\varepsilon_V\|_Q^2 \leq \frac{1}{2}\rho_S(\varepsilon_V) \leq -\mathcal{I}_Q(\varepsilon_V) \leq \rho_S(\varepsilon_V). \quad (4.16)$$

Beweis. Da e_Q und ε_V Lösungen der Variationsungleichungen (4.5) und (4.9) sind, folgt die Behauptung analog zum Beweis von Lemma 4.12. \square

Wegen (4.16) ist also $\rho_S(\varepsilon_V)$ äquivalent zum Fehlerschätzer $-\mathcal{I}_Q(\varepsilon_V)$ und kann daher als Fehlerindikator für $-\mathcal{I}_Q(\varepsilon_V)$ verwendet werden, denn wenn wir ρ_S verkleinern, so wird auch $-\mathcal{I}_Q$ kleiner. In Kapitel 4.1.4 und 4.1.5 werden wir dann die Äquivalenz von $-\mathcal{I}_Q(\varepsilon_V)$ zum exakten Fehler in den Funktionalen $J(u_S) - J(u) = -\mathcal{I}(e)$ zeigen.

Aus Lemma 4.12 folgt analog, dass der Fehler $J(u_S) - J(u)$ äquivalent zu $\rho_S(e)$ ist. Daher werden wir die lokalen Anteile des Fehlerschätzers mittels $\rho_S(\varepsilon_V)$ berechnen. Es sei u_S die Lösung von (3.12), dann gilt für alle $T \in \mathcal{T}_h$ die Gleichung $\Delta u_S = 0$, da u_S auf jedem T linear ist. Mit $\Omega = \bigcup_{T \in \mathcal{T}_h} T$ berechnen wir daher für alle $v \in H^1(\Omega)$

$$\begin{aligned}\rho_S(v) &= (f, v) - a(u_S, v) = \int_{\Omega} fv dx - \int_{\Omega} \nabla u_S \nabla v dx \\ &= \int_{\Omega} fv dx - \sum_{T \in \mathcal{T}_h} \int_T \nabla u_S \nabla v dx \\ &= \int_{\Omega} fv dx - \sum_{T \in \mathcal{T}_h} \left(\int_{\partial T} v \partial_{\mathbf{n}} u_S ds - \int_T \underbrace{\Delta u_S}_= v dx \right) \\ &= \int_{\Omega} fv dx - \sum_{T \in \mathcal{T}_h} \int_{\partial T} v \partial_{\mathbf{n}} u_S ds,\end{aligned} \quad (4.17)$$

wobei im vorletzten Schritt die 1. Green'sche Formel angewendet wurde und \mathbf{n} die äußere Einheitsnormale von einem Dreieck T bezeichnet.

Betrachten wir zwei beliebige Dreiecke T_1, T_2 wie in Abbildung 4.2, wobei \mathbf{n} hierbei die Einheitsnormale, die von T_1 nach T_2 zeigt, bezeichnet, so können wir die Summe aus (4.17) bzgl. der Menge der Kanten \mathcal{E} darstellen, da der Rand $\partial T = E_1 \cup E_2 \cup E_3$ für jedes T disjunkt in seine Kantenstücke aufgeteilt werden kann.

Dabei sei E nun die Kante, die T_1 und T_2 zugleich enthalten, d.h. \mathbf{n} steht rechtwinklig auf E . Dann gilt, dass die Richtungsableitung $\partial_{\mathbf{n}} u_S|_{T_2}$ negativ ist bzgl. (4.17) wegen der negativen Orientierung von \mathbf{n} bzgl. T_2 .

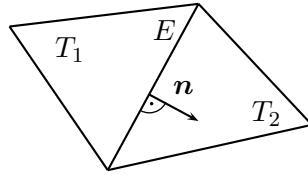


Abbildung 4.2: Dreiecke T_1 und T_2 mit Einheitsnormalen \mathbf{n}

Dies in (4.17) eingesetzt ergibt

$$\begin{aligned} \rho_S(v) &= \int_{\Omega} fv dx - \sum_{T \in \mathcal{T}_h} \int_{\partial T} v \partial_{\mathbf{n}} u_S ds \\ &= \int_{\Omega} fv dx - \sum_{E \in \mathcal{E}} \int_E v (\underbrace{\partial_{\mathbf{n}} u_S|_{T_1} - \partial_{\mathbf{n}} u_S|_{T_2}}_{=: j_E}) ds \\ &= \int_{\Omega} fv dx + \sum_{E \in \mathcal{E}} \int_E j_E v ds. \end{aligned} \quad (4.18)$$

Da für die nodalen Basisfunktionen $\{\phi_p \mid p \in \mathcal{N} \cap \Omega\}$ von \mathcal{S}_h die Partition der Eins gilt, also

$$\sum_{p \in \mathcal{N}} \phi_p = 1 \text{ auf ganz } \Omega,$$

lässt sich ρ_S wie folgt in lokale Anteile aufteilen können:

$$\rho_p(v) := \rho_S(v\phi_p), \quad v \in H^1(\Omega), p \in \mathcal{N}. \quad (4.19)$$

In Lemma 4.14 werden wir zeigen, dass wir den in (4.19) definierten lokalen Anteil von ρ_S leicht berechnen können, und mit der Aussage aus Korollar 4.15, dass sich der Fehlerindikator ρ_S auch eindeutig in die lokalen Anteile ρ_p zerlegen lässt.

Lemma 4.14. Für ρ_p gilt

$$\rho_p(v) = \int_{\omega_p} fv \phi_p dx + \sum_{E \in \mathcal{E}_p} \int_E j_E v \phi_p ds, \quad v \in H^1(\Omega)$$

mit $\omega_p := \text{supp } \phi_p$ und $\mathcal{E}_p := \{E \in \mathcal{E} \mid E \ni p\}$, d.h. die Menge der Kanten, in denen p enthalten ist.

Beweis. Wir rechnen einfach mit der Definition (4.19) und (4.18) nach, dass für ein beliebiges $v \in H^1(\Omega)$ gilt

$$\begin{aligned}\rho_p(v) &= \rho_{\mathcal{S}}(v\phi_p) = \int_{\Omega} fv\phi_p dx + \sum_{E \in \mathcal{E}} \int_E j_E v\phi_p ds \\ &= \int_{\omega_p} fv\phi_p dx + \sum_{E \in \mathcal{E}_p} \int_E j_E v\phi_p ds,\end{aligned}$$

da $\phi_p \equiv 0$ auf $\mathcal{O} := \overline{\Omega \setminus \omega_p}$ und damit auch auf $\mathcal{F} := \mathcal{E} \setminus \mathcal{E}_p$, da $\mathcal{F} \subset \mathcal{O}$ (vgl. Abbildung 4.3). \square

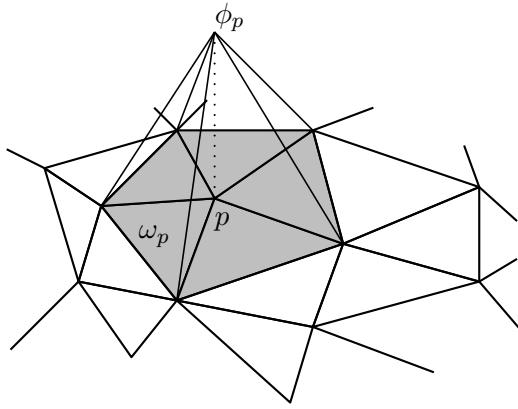


Abbildung 4.3: Darstellung von ω_p (grau) und \mathcal{E}_p (abgehende Kanten von p) für ein beliebiges ϕ_p

Korollar 4.15. Der Indikator $\rho_{\mathcal{S}}$ lässt sich schreiben als

$$\rho_{\mathcal{S}} = \sum_{p \in \mathcal{N}} \rho_p. \quad (4.20)$$

Beweis. Die Behauptung folgt direkt aus Lemma 4.14 zusammen mit

$$\Omega = \bigcup_{p \in \mathcal{N}} \omega_p, \quad \mathcal{E} = \bigcup_{p \in \mathcal{N}} \mathcal{E}_p \quad \text{und} \quad \sum_{p \in \mathcal{N}} \phi_p = 1$$

durch einfaches Nachrechnen. \square

Im unbeschränkten Fall gilt $\rho_{\mathcal{S}} = 0 \Leftrightarrow e = 0$, denn zu $\rho_{\mathcal{S}} = 0$ ist

$$a(e, v) = \rho_{\mathcal{S}}(v) = 0 \quad \forall v \in \mathcal{V}$$

äquivalent. Da $e \in \mathcal{V}$ ist, folgt wegen der Galerkin-Orthogonalität, dass $e = 0$ sein muss. Die Umkehrung gilt offensichtlich auch.

Wie wir schon weiter oben beschrieben haben, gilt die Galerkin-Orthogonalität bei Variationsungleichungen nicht. Aber aus Lemma 4.12 folgt allgemeiner, falls $\rho_S(v) \leq 0$ für alle $v \in \mathcal{A}$ gilt,

$$\frac{1}{2}\|e\|^2 \leq \rho_S(e) \leq 0 \implies \|e\| = 0 \implies e = 0,$$

d.h. aus $\rho_S = 0$ folgt, dass $e = 0$ ist. Man kann nun diese globale Eigenschaft auch auf die lokalen Anteile übertragen. Mit u_S als Lösung von (3.12) gilt für alle $p \in \mathcal{N} \cap \Omega$, dass $v = u_S + \phi_p \geq \psi$, d.h. $v \in K_h$. Daraus folgt mit Einsetzen von v in (3.12)

$$\begin{aligned} a(u_S, u_S + \phi_p - u_S) &\geq (f, u_S + \phi_p - u_S) \\ \iff 0 &\geq (f, \phi_p) - a(u_S, \phi_p) = \rho_S(\phi_p). \end{aligned} \quad (4.21)$$

Dies bedeutet, dass die lineare Approximation des Fehlers e gleich Null ist. Falls an einem Punkt p kein Kontakt zwischen u_S und ψ vorliegt, also $u_S(p) > \psi(p)$ ist, dann können wir ein $\alpha > 0$ hinreichend klein wählen, sodass $v = u_S - \alpha\phi_p \in K_h$ liegt. Dann folgt analog durch Einsetzen von v in (3.12)

$$\begin{aligned} 0 &\geq (f, -\alpha\phi_p) - a(u_S, -\alpha\phi_p) \\ \iff 0 &\leq (f, \phi_p) - a(u_S, \phi_p) = \rho_S(\phi_p) \stackrel{(4.21)}{\leq} 0 \end{aligned}$$

und damit gilt $\rho_S(\phi_p) = 0$. Zusammen ergeben sich die (KKT-)Bedingungen

$$\rho_S(\phi_p) \leq 0, \quad \psi(p) - u_S(p) \leq 0, \quad \rho_S(\phi_p)(\psi(p) - u_S(p)) = 0. \quad (4.22)$$

Dies berechtigt zur Definition von Kontakt- und Nichtkontaktpunkten.

Definition 4.16. Wir definieren die Mengen von *inneren Kontaktknoten* \mathcal{N}^0 und *Nichtkontaktknoten* \mathcal{N}^+ durch

$$\mathcal{N}^0 := \{p \in \mathcal{N} \cap \Omega \mid u_S(p) = \psi(p)\}, \quad \mathcal{N}^+ := \{p \in \mathcal{N} \cap \Omega \mid u_S(p) > \psi(p)\}.$$

Bemerkung 4.17. Die Bedingungen (4.22) können wir auch auf den lokalen Anteil ρ_p übertragen, damit ergibt sich für alle $p \in \mathcal{N} \cap \Omega$

$$\rho_p(1) \leq 0, \quad (4.23a)$$

$$u_S(p) > \psi(p) \implies \rho_p(1) = 0, \quad (4.23b)$$

denn $\rho_p(1) = \rho_S(\phi_p)$.

Damit ist also die Approximation von e über S_h gleich Null, wenn die lokalen Anteile von ρ_S kleiner gleich Null sind.

4.1.3 Oszillationsterme

In Kapitel 4.1.4 werden wir zeigen, dass $-\mathcal{I}_Q(\varepsilon\mathcal{V})$ eine obere Schranke von $-\mathcal{I}(e)$ bis auf Terme höherer Ordnung bereitstellen, d.h. Terme, die nicht in \mathcal{V} enthalten sind. Diese sogenannten *Oszillationsterme* wollen wir daher in diesem Kapitel anschaulich einführen (ohne präzise Beweise).

Wie wir später auch in den numerischen Beispielen sehen werden, bringt eine Verkleinerung der Oszillation auch eine Verringerung des Fehlers mit sich. Daher ist es von Interesse die Oszillationsterme einfach berechnen zu können und auch mit in die adaptive Verfeinerung einzubeziehen.

Wir werden zwei Arten von Oszillationen betrachten und daher die Gesamtoszillation in diese Anteile aufteilen.

$$\text{osc}(u_S, \psi, f) := (\text{osc}_1(u_S, \psi)^2 + \text{osc}_2(u_S, \psi, f)^2)^{\frac{1}{2}} \quad (4.24)$$

Bemerkung 4.18. In [ZVKG11] werden die Oszillationsterme (4.24) durch

$$\text{osc}(u_S, \psi, f) = \text{osc}_1(u_S, \psi) + \text{osc}_2(u_S, \psi, f)$$

eingeführt. Wir wählen hier absichtlich eine leicht veränderte Darstellung, da diese für die spätere Implementierung bzgl. der lokalen Anteile der Oszillationen von Vorteil ist. Außerdem ist in vielen Fällen die Menge \mathcal{N}^{0+} (s.u.) nach wenigen Verfeinerungsschritten leer, sodass $\text{osc}(u_S, \psi, f) = \text{osc}_2(u_S, \psi, f)$ in beiden beschriebenen Fällen gilt.

Im unbeschränkten Fall ist die Oszillation nur von f abhängig (vgl. [MNS00]) und daher wird dabei von „Daten-Oszillation“ gesprochen.

Der Oszillationsterm $\text{osc}_1(u_S, \psi)$ ist ein Maß für die Oszillation zwischen dem Hindernis ψ und der Galerkin-Lösung u_S , d.h.

$$\text{osc}_1(u_S, \psi) := \left(\sum_{p \in \mathcal{N}^{0+}} \|\nabla(\psi - u_S)\|_{0, \omega_p}^2 \right)^{\frac{1}{2}},$$

wobei $\mathcal{N}^{0+} := \{p \in \mathcal{N}^0 \mid u_S > \psi \text{ in } \omega_p \setminus \{p\}\}$ ist, also die Menge der *isolierten Kontaktknoten*, d.h. u_S ist auf der Menge ω_p nur im Punkt p mit ψ in Kontakt. Anschaulich stellt $\text{osc}_1(u_S, \psi)$ deshalb ein Maß für die Hindernisoszillation dar, weil wegen der Linearität von u_S, ψ folgt, dass eine größere Differenz zwischen ψ und u_S auch einen größeren Gradienten $\nabla(\psi - u_S)$ impliziert und damit auch eine größere Oszillation $\text{osc}_1(u_S, \psi)$.

Das kontinuierliche Gegenstück zu \mathcal{N}^{0+} ist die Menge der *isolierten Kontaktpunkte* x_c , die aufgrund von $u - \psi > 0$ für alle $x \in \mathcal{U}(x_c, \varepsilon) \subset \Omega$ mit $u(x_c) = \psi(x_c)$ alle strikten Minima $x_c \in \Omega$ von $u - \psi$ enthält. Daraus folgt, dass $(\nabla u - \nabla \psi) = 0$ ist für alle isolierten Kontaktpunkte, wenn u, ψ hinreichend glatt sind.

Stellt nun ψ nur eine lineare Approximation an ein hinreichend glattes Hindernis dar, so folgt mit den Aussagen aus Theorem 3.16 und Theorem 2 in [Zha07], dass wegen der Konvergenz der diskretisierten Ränder für einen isolierter Kontaktknoten $p \in \mathcal{N}^{0+}$, der bei Verfeinerung bestehen bleibt, ein für die exakte Lösung u korrespondierender Kontaktspunkt \tilde{p} existiert und es gilt

$$\bigcup_{p \in \mathcal{N}^{0+}} \omega_p \xrightarrow[h \rightarrow 0]{} \tilde{p}.$$

Sind u und ψ also hinreichend glatt, so verschwindet der Oszillationsterm $\text{osc}_1(u_S, \psi)$ für $h \rightarrow 0$.

Der Oszillationsterm $\text{osc}_2(u_S, \psi, f)$ ist auf zwei Mengen definiert:

$$\mathcal{N}^{++} := \{p \in \mathcal{N}^+ \mid \rho_E \geq -d_E \forall E \in \mathcal{E}_p\}, \quad (4.25)$$

d.h. alle Knoten ohne Kontakt, in denen der Fehler ε_V nicht in Kontakt mit dem Hindernis aus \mathcal{A}_V steht (wie in Beweis von Satz 4.11 ersichtlich) und

$$\mathcal{N}^{0-} := \{p \in \mathcal{N}^0 \mid u_S = \psi, f \leq 0 \text{ auf } \omega_p, j_E \leq 0 \forall E \in \mathcal{E}_p\}, \quad (4.26)$$

d.h. die Kontaktknoten mit vollem Kontakt, sodass die Kraft f eine Drucklast ist und negativer Normalenfluss j_E herrscht (vgl. auch [SV07] Gleichung (2.11)).

Aus der Nebenbedingung von \mathcal{N}^{0-} folgt

$$0 \geq f + \sum_{E \in \mathcal{E}_p} j_E$$

Durch Multiplikation mit geeigneten Testfunktionen v und Integration über ω_p ergibt

$$\begin{aligned} 0 &\geq \int_{\omega_p} f v \, dx + \sum_{E \in \mathcal{E}_p} \int_E j_E v \, ds \\ &= \int_{\omega_p} f v \, dx - \int_{\omega_p} \underbrace{\nabla u_S}_{=\nabla \psi} \nabla v \, dx \end{aligned}$$

und damit gilt

$$\int_{\omega_p} \nabla \psi \nabla v \, dx \geq \int_{\omega_p} f v \, dx. \quad (4.27)$$

Es gilt also mit (4.27) laut Satz 3.4, dass $-\Delta \psi - f \geq 0$ auf ω_p im distributionellem Sinne (vgl. auch [Wal11] Kapitel 3) ist. Dies ist laut Satz 3.4 auch notwendig, damit $u = \psi$ auf ω_p gilt.

Mit den Mengen (4.25) und (4.26) ist der Oszillationsterm $\text{osc}_2(u_S, \psi, f)$ definiert als

$$\text{osc}_2(u_S, \psi, f) := \left(\sum_{p \in \mathcal{N}^{++}} h_p^2 \|f - \bar{f}_p\|_{0, \omega_p}^2 + \sum_{p \in \mathcal{N} \setminus (\mathcal{N}^0 \cup \mathcal{N}^{++})} h_p^2 \|f\|_{0, \omega_p}^2 \right)^{\frac{1}{2}},$$

wobei $h_p := \max_{E \in \mathcal{E}_p} |E|$ für jedes $p \in \mathcal{N}$ ist, h_p ist ein Maß für den Durchmesser von ω_p , und \bar{f}_p den Mittelwert von f über ω_p bezeichne, also

$$\bar{f}_p = \frac{1}{|\omega_p|} \int_{\omega_p} f \, dx.$$

Anschaulich kann man die Summanden der ersten Summe als Varianz der Last f auf ω_p interpretieren. Da wir in der Variationsrechnung ein gemitteltes Problem betrachten, ist die erste Summe also ein Maß dafür, auf welchen Teilen von Ω die wirkliche Last f um einen gemittelten Wert oszilliert. Dies impliziert bei starker Oszillation anschaulich, dass eine Verfeinerung in solchen Gebieten, die Lösung genauer machen würde.

In der Menge $\mathcal{N} \setminus (\mathcal{N}^0 \cup \mathcal{N}^{++})$ sind die inneren Knoten, die keinen vollen Kontakt $u_S = \psi$ auf ω_p haben und für die die Lösung ε_V des lokalen Defektproblems keinen Kontakt mit der Nebenbedingung von \mathcal{A}_V hat. Man beachte also, dass in $\mathcal{N} \setminus (\mathcal{N}^0 \cup \mathcal{N}^{++})$ nur die Kontaktknoten fehlen, die vollen Kontakt haben, und die Nichtkontaktknoten, für die ε_V in Kontakt mit dem Hindernis aus \mathcal{A}_V stehen. Diese Menge beschreibt gerade die Randpunkte am Hindernis und $\text{osc}_2(u_S, \psi, f)$ enthält nur Anteile aus Knoten ohne vollen Kontakt, da für diese die Varianz bzgl. des Lastvektors f irrelevant wäre, denn hier würde nach Verfeinerung immer noch voller Kontakt herrschen.

Bemerkung. Die Oszillationsterme können leicht berechnet bzw. implementiert werden, wie man in Kapitel 5 oder auch Anhang D nachvollziehen kann.

Im unbeschränkten Fall, also $\psi = -\infty$, ist ε_V nicht im Kontakt mit dem Hindernis für alle Knoten aus \mathcal{N} und damit gilt $\mathcal{N}^{++} = \mathcal{N} \cap \Omega$. Wir rechnen nach, dass dann für die Menge der Randpunkte gilt:

$$\mathcal{N} \setminus (\mathcal{N}^0 \cup \mathcal{N}^{++}) = \mathcal{N} \setminus (\mathcal{N} \cap \Omega) = \mathcal{N} \cap \partial\Omega.$$

Also erhalten wir für den Oszillationsterm $\text{osc}_2(u_S, \psi, f)$ die Form

$$\text{osc}_2(u_S, \psi, f) = \left(\sum_{p \in \mathcal{N} \cap \Omega} h_p^2 \|f - \bar{f}_p\|_{0, \omega_p}^2 + \sum_{p \in \mathcal{N} \cap \partial\Omega} h_p^2 \|f\|_{0, \omega_p}^2 \right)^{\frac{1}{2}},$$

was der Daten-Oszillation aus dem unbeschränkten Fall entspricht. Damit ist der zweite Oszillationsterm aus (4.24) also eine Verallgemeinerung des

unbeschränkten Falles. Wenn also der Teil ohne Kontakt bekannt wäre, dann wäre der beschränkte Fall auf dieser Menge von Knoten äquivalent zu einem unbeschränkten Dirichlet-Problem.

4.1.4 Zuverlässigkeit des Fehlerschätzers

Wir werden in diesem Kapitel eine obere Schranke für den Fehler im Energiefunktional $-\mathcal{I}(e)$, die vom hierarchischen Fehlerschätzer $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}})$ abhängt, herleiten. Hierfür sind zunächst Eigenschaften für die Interpolation von e auf $\varepsilon_{\mathcal{V}}$ notwendig.

Die Reduktion des Fehlers $e = u - u_{\mathcal{S}} \in H_0^1(\Omega)$ auf den approximierten Fehler $\varepsilon_{\mathcal{V}} \in \mathcal{V}$ erhalten wir durch lokale Projektionen für jedes $p \in \mathcal{N}$ mit

$$\pi_p : H^1(\Omega) \rightarrow \mathcal{Q}_p = \text{span}\{\phi_p\} \cup \mathcal{V}_p, \quad \mathcal{V}_p = \text{span}\{\phi_E \mid E \in \mathcal{E}_p\}.$$

Die lokalen Projektionen π_p sind für jedes $v \in H^1(\Omega)$ aus Dimensionsgründen ($\dim(\mathcal{Q}_p) = p + 1$) eindeutig bestimmt durch

$$\int_E \pi_p v \, ds = \int_E v \, ds \quad \forall E \in \mathcal{E}_p \text{ und } \begin{cases} \int_{\omega_p} \pi_p v \, dx = \int_{\omega_p} v \, dx & , p \in \mathcal{N}^{++} \\ 0 & , \text{sonst} \end{cases}. \quad (4.28)$$

Zunächst zeigen wir ein Resultat, das uns die Koordinaten für das Bild der lokalen Projektion π_p in einer Basis von \mathcal{Q}_p liefert.

Lemma 4.19. *Es sei π_p die oben beschriebene Projektion. Dann gelten für die Koordinaten bzgl. der Basis $\{\phi_p\} \cup \{\phi_E \mid E \in \mathcal{E}_p\}$ von*

$$\pi_p v = \alpha_p(v) \phi_p + \sum_{E \in \mathcal{E}_p} \alpha_E(v) \phi_E$$

die Beziehungen

$$\alpha_p(v) = \begin{cases} \frac{c_p(v)}{c_p(\phi_p)} & , p \in \mathcal{N}^{++} \\ 0 & , \text{sonst} \end{cases}, \quad (4.29a)$$

$$\alpha_E(v) = \frac{\int_E v \, ds - \alpha_p(v) \int_E \phi_p \, ds}{\int_E \phi_E \, ds}, \quad (4.29b)$$

wobei

$$c_p(v) = \int_{\omega_p} v \, dx - \sum_{E \in \mathcal{E}_p} \left(\int_E v \, ds \right) \left(\int_{\omega_p} \phi_E \, dx \right) \left(\int_E \phi_E \, ds \right)^{-1}$$

Insbesondere gilt $c_p(\phi_p) = -\frac{1}{6} |\omega_p|$.

Beweis. Für eine bessere Übersicht im Beweis werden wir die Differentialformen dx und ds weg. Es sei $v \in H^1(\Omega)$ beliebig. Dann gilt für jede Kante $E \in \mathcal{E}_p$ mit

$$\pi_p v = \alpha_p(v) \phi_p + \alpha_E(v) \phi_E \in \mathcal{Q}_p$$

nach (4.28), dass

$$\begin{aligned} \int_E v &= \int_E \pi_p v = \int_E \alpha_p(v) \phi_p + \alpha_E(v) \phi_E \\ \implies \alpha_E(v) &= \left(\int_E v - \alpha_p(v) \int_E \phi_p \right) \left(\int_E \phi_E \right)^{-1}. \end{aligned} \quad (4.30)$$

Wenn $p \notin \mathcal{N}^{++}$ ist, so gilt $\pi_p v \in \mathcal{V}_p = \text{span}\{\phi_E \mid E \in \mathcal{E}_p\}$, d.h. $\alpha_p(v) = 0$.

Es sei nun also $p \in \mathcal{N}^{++}$. Dann folgt aus der zweiten Eigenschaft von (4.28) und (4.30) für $\pi_p v = \alpha_p(v) \phi_p + \sum_{E \in \mathcal{E}_p} \alpha_E(v) \phi_E \in \mathcal{Q}_p$

$$\begin{aligned} \int_{\omega_p} v &= \int_{\omega_p} \pi_p v = \int_{\omega_p} \alpha_p(v) \phi_p + \sum_{E \in \mathcal{E}_p} \alpha_E(v) \phi_E \\ &= \alpha_p(v) \int_{\omega_p} \phi_p + \sum_{E \in \mathcal{E}_p} \alpha_E(v) \int_{\omega_p} \phi_E \\ &= \alpha_p(v) \int_{\omega_p} \phi_p + \sum_{E \in \mathcal{E}_p} \left(\int_E v - \alpha_p(v) \int_E \phi_p \right) \left(\int_E \phi_E \right)^{-1} \left(\int_{\omega_p} \phi_E \right) \\ &= \alpha_p(v) \underbrace{\left(\int_{\omega_p} \phi_p - \sum_{E \in \mathcal{E}_p} \left(\int_E \phi_p \right) \left(\int_{\omega_p} \phi_E \right) \left(\int_E \phi_E \right)^{-1} \right)}_{=c_p(\phi_p)} \\ &\quad + \sum_{E \in \mathcal{E}_p} \left(\int_E v \right) \left(\int_{\omega_p} \phi_E \right) \left(\int_E \phi_E \right)^{-1}. \end{aligned}$$

Nach dem Umformen nach $\alpha_p(v)$ ergibt sich dann

$$\alpha_p(v) = \frac{c_p(v)}{c_p(\phi_p)}$$

mit dem oben definierten $c_p(\cdot)$.

Es bleibt also zu zeigen, dass $c_p(\phi_p) = -\frac{1}{6} |\omega_p|$. Hierfür betrachten wir die einzelnen Summanden von $c_p(\phi_p)$. Zunächst berechnet das Integral von ϕ_p über ω_p das Volumen der von ϕ_p erzeugten Pyramide mit Grundfläche $|\omega_p|$, d.h.

$$\int_{\omega_p} \phi_p = \frac{1}{3} |\omega_p|. \quad (4.31)$$

Weiter ist ϕ_p auf jeder Kante $E \in \mathcal{E}_p$ eine von 1 zu 0 abfallende Gerade und damit ist das Integral über E gerade der Flächeninhalt vom darüber liegenden Dreieck, also

$$\int_E \phi_p = \frac{1}{2} |E|. \quad (4.32)$$

Die letzten beiden Integrale berechnen wir über die Referenzelemente in \mathbb{R} oder \mathbb{R}^2 für das Kurven- bzw. Flächenintegral. Die Funktion ϕ_E über einer Kante E ist eine nach unten geöffnete Parabel. Auf dem Referenzelement $[-1, 1] \subset \mathbb{R}$ ist dies die Funktion

$$\hat{\phi}_E = 1 - \xi^2$$

und mit einer affinen Transformation $s : [-1, 1] \rightarrow [a, b] = E, s(\xi) = \frac{b-a}{2}\xi + \frac{b+a}{2}$ lässt sich das Referenzelement auf das Element E abbilden. Damit ergibt sich mit dem Transformationssatz der Integration

$$\int_E \phi_E = \frac{b-a}{2} \int_{-1}^1 \hat{\phi}_E = \frac{1}{2} |E| \cdot \frac{4}{3} = \frac{2}{3} |E|. \quad (4.33)$$

Der letzte Fall ist komplizierter zu beschreiben. Zunächst sei erwähnt, dass $\text{supp}(\phi_E) = T_i \cup T_j, T_i, T_j \subset \omega_p, i \neq j$ gilt, ϕ_E also nur auf zwei Dreiecken, die in ω_p enthalten sind, ungleich Null ist. Damit wird für jede Kante $E \in \mathcal{E}_p$ über jedes Dreieck $T \subset \omega_p$ genau zweimal integriert.

Auf dem Referenzelement

$$\hat{T} := \{(\xi, \eta) \in \mathbb{R}^2 \mid 0 \leq \xi \leq 1, 0 \leq \eta \leq 1 - \xi\}$$

haben wir die drei Bubble-Funktionen

$$\hat{\phi}_{E_1} = 4\xi(1 - \xi - \eta), \quad \hat{\phi}_{E_2} = 4\xi\eta, \quad \hat{\phi}_{E_3} = 4\eta(1 - \xi - \eta),$$

für die man leicht nachrechnen kann, dass

$$\int_{\hat{T}} \hat{\phi}_{E_1} = \int_{\hat{T}} \hat{\phi}_{E_2} = \int_{\hat{T}} \hat{\phi}_{E_3} = \frac{1}{6}$$

gilt. Es sei nun J_T die Jacobi-Determinante bzgl. einer affinen Transformation $r : \hat{T} \rightarrow T$, dann gilt nach Transformationssatz mit einem $T \subset \text{supp}(\phi_E)$

$$\int_T \phi_E = |J_T| \int_{\hat{T}} \hat{\phi}_E = \frac{1}{6} |J_T|.$$

Weiter rechnen wir nach, dass

$$|T| = \int_T dx = |J_T| \int_{\hat{T}} dx = \frac{1}{2} |J_T| \implies |J_T| = 2 |T|$$

gilt und damit folgt insgesamt zusammen mit (4.31) bis (4.33)

$$\begin{aligned}
 c_p(\phi_p) &= \int_{\omega_p} \phi_p - \sum_{E \in \mathcal{E}_p} \left(\int_E \phi_p \right) \left(\int_{\omega_p} \phi_E \right) \left(\int_E \phi_E \right)^{-1} \\
 &= \frac{1}{3} |\omega_p| - 2 \sum_{T \subset \omega_p} \frac{1}{2} |E| \cdot \frac{1}{6} |J_T| \cdot \frac{3}{2} |E|^{-1} \\
 &= \frac{1}{3} |\omega_p| - \sum_{T \subset \omega_p} \frac{1}{2} |T| = \left(\frac{1}{3} - \frac{1}{2} \right) |\omega_p| \\
 &= -\frac{1}{6} |\omega_p|. \quad \square
 \end{aligned}$$

Für das zentrale Lemma in diesem Kapitel benötigen wir noch eine Abschätzung für die Koordinaten der lokalen Projektionen.

Lemma 4.20. *Die Koeffizienten in (4.29) erfüllen die Eigenschaft*

$$\max_{Q \in \{p\} \cup \mathcal{E}_p} |\alpha_Q(v)| \lesssim h_p^{-1} (\|v\|_{0,\omega_p} + h_p \|\nabla v\|_{0,\omega_p}) \quad (4.34)$$

und π_p ist stabil im Sinne von

$$\|\pi_p v\|_{0,\omega_p} \lesssim \|v\|_{0,\omega_p} + h_p \|\nabla v\|_{0,\omega_p}. \quad (4.35)$$

Insbesondere gilt, wenn $p \notin \mathcal{N}^{++}$ ist, dass für $\alpha_E(v) = (\int_E v ds) (\int_E \phi_E)^{-1}$ die Eigenschaft gilt:

$$\int_E v ds \geq \int_E (\psi - u_S) ds \implies \alpha_E(v) \gtrsim \psi(x_E) - u_S(x_E) \quad \forall E \in \mathcal{E}_p. \quad (4.36)$$

Beweis. Unter Verwendung der Cauchy-Schwarz'schen Ungleichung und einer Anwendung vom Spursatz (s. Theorem A.9) erhalten wir zunächst

$$\begin{aligned}
 \left| \int_{\omega_p} v dx \right| &\stackrel{\text{C.S.}}{\leq} \underbrace{\left(\int_{\omega_p} dx \right)^{\frac{1}{2}}}_{=|\omega_p|^{\frac{1}{2}}} \left(\int_{\omega_p} v^2 dx \right)^{\frac{1}{2}} \lesssim h_p \|v\|_{0,\omega_p}, \\
 \left| \int_E v ds \right| &\stackrel{\text{C.S.}}{\leq} \underbrace{\left(\int_E ds \right)^{\frac{1}{2}}}_{=|E|^{\frac{1}{2}} \leq h_p^{\frac{1}{2}}} \underbrace{\left(\int_E v^2 ds \right)^{\frac{1}{2}}}_{=\|v\|_{0,E}} \stackrel{\text{Spursatz}}{\lesssim} h_p (h_p^{-1} \|v\|_{0,\omega_p} + \|\nabla v\|_{0,\omega_p}).
 \end{aligned}$$

Analog zum Beweis von Lemma 4.19 können wir für ϕ_p und ϕ_E berechnen, dass die Integrale jeweils über E sowie ω_p von den Gebieten und einer weiteren Konstanten abhängen und damit also „ $\approx h_p$ “ sind. Durch Anwendung

der Dreiecksungleichung sowie den eben erwähnten Integralen lässt sich damit berechnen, dass

$$|\alpha_p(v)| \lesssim \|v\|_{0,\omega_p} + h_p \|\nabla v\|_{0,\omega_p}$$

gilt. Dies in (4.29b) eingesetzt, ergibt dann wieder unter Verwendung der Dreiecksungleichung und der endlichen Integrale von ϕ_p, ϕ_E sowie der obigen Abschätzungen die Schranke

$$|\alpha_E(v)| \lesssim h_p^{-1} (\|v\|_{0,\omega_p} + h_p \|\nabla v\|_{0,\omega_p})$$

und damit folgt (4.34).

Die Abschätzung (4.35) folgt direkt aus den Abschätzungen von (4.34) zusammen mit der Koordinatendarstellung

$$\pi_p v = \alpha_p(v) \phi_p + \sum_{E \in \mathcal{E}_p} \alpha_E(v) \phi_E$$

der lokalen Projektion und vorheriger Anwendung der Dreiecksungleichung auf die Norm $\|\pi_p v\|_{0,\omega_p}$.

Es sei $P \notin \mathcal{N}^{++}$. Dann gilt wegen (4.29a) $\alpha_p(v) = 0$ und damit mit (4.29b)

$$\alpha_E(v) = \left(\int_E v \, ds \right) \left(\int_E \phi_E \, ds \right)^{-1}.$$

Da ψ wegen Voraussetzung 4.1 stückweise linear ist, gilt dies auch für $\psi - u_S$, d.h. die Funktion stellt eine Gerade über der Kante E dar. Mit der Mittelpunktsregel der Quadratur (vgl. [Sto99] Kapitel 3) ergibt sich dann also

$$\int_E (\psi - u_S) \, ds = |E| (\psi(x_E) - u_S(x_E)).$$

Wie wir im Beweis von Lemma 4.19 nachgerechnet haben, gilt $\int_E \phi_E \, ds = \frac{2}{3}|E|$ und dann ist unter der Voraussetzung von (4.36)

$$\begin{aligned} \psi(x_E) - u_S(x_E) &= \frac{3}{2}|E|^{-1} \int_E (\psi - u_S) \, ds \\ &\lesssim \left(\int_E \phi_E \, ds \right)^{-1} \left(\int_E v \, ds \right) \\ &= \alpha_E(v). \end{aligned} \quad \square$$

Jetzt haben wir das komplette Werkzeug, um das zentrale Lemma zum Beweis einer oberen Schranke für $J(u_S) - J(u)$ bzgl. $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_V)$ zu zeigen. Da wir gesehen haben, dass $-\mathcal{I}(e)$ äquivalent zum Fehlerindikator $\rho_S(e)$ ist, leiten wir für diesen im folgenden Lemma eine obere Schranke her.

Lemma 4.21. *Es sei Voraussetzung 4.1 erfüllt. Dann gilt*

$$\rho_{\mathcal{S}}(e) \lesssim \sum_{E \in \mathcal{E}} \eta_E |\rho_E| + \text{osc}(u_{\mathcal{S}}, \psi, f)^2 \quad (4.37)$$

mit ρ_E wie in (4.11), $\text{osc}(u_{\mathcal{S}}, \psi, f)$ wie in (4.24) und $\eta_E = |\varepsilon_{\mathcal{V}}(x_E)| \|\phi_E\|$.

Beweis. Die Idee des Beweises beruht auf Gleichung (4.20); damit können wir den Indikator in die lokalen Anteile bzgl. der Punkte $p \in \mathcal{N}$ aufteilen, d.h.

$$\rho_{\mathcal{S}}(e) = \sum_{p \in \mathcal{N}} \rho_p(e). \quad (4.38)$$

Hierbei ist die Abschätzung der lokalen Anteile $\rho_p(e)$ abhängig von einer Anwendung der Poincaré-Friedrichungleichung (Satz 2.13), die auf die *verallgemeinerte Poincaré-Friedrich-Ungleichung* (folgt direkt aus Satz 2.13 mit $\tilde{v} = v - c$ eingesetzt)

$$\|v - c\|_{0, \omega_p} \lesssim h_p \|\nabla v\|_{0, \omega_p} \quad (4.39)$$

mit einer Konstanten c und $v \in H^1(\Omega)$, so dass $v = c$ auf einer Menge $\Gamma \subset \partial\omega_p$ mit einem Maß $\mu(\Gamma) \neq 0$ gilt, führt. Da (4.39) von $p \in \mathcal{N}$ abhängt, werden wir sehen, dass die Anwendung von der Poincaré-Friedrich-Ungleichung vom Typ des Knotens p abhängt. Deshalb betrachten wir die disjunkte Vereinigung

$$\begin{aligned} & \overbrace{\mathcal{N}^{++} \cup (\mathcal{N}^+ \setminus \mathcal{N}^{++})}^{=\mathcal{N}^+} \cup (\mathcal{N} \cap \partial\Omega) \cup \overbrace{(\mathcal{N}^0 \setminus (\mathcal{N}^{0+} \cup \mathcal{N}^{0-})) \cup \mathcal{N}^{0+} \cup \mathcal{N}^{0-}}^{=\mathcal{N}^0} \\ &= \mathcal{N}^+ \cup \mathcal{N}^0 \cup (\mathcal{N} \cap \partial\Omega) = (\mathcal{N} \cap \Omega) \cup (\mathcal{N} \cap \partial\Omega) = \mathcal{N}. \end{aligned} \quad (4.40)$$

Wir wollen im Folgenden die in (4.40) aufgeführten Fälle chronologisch abarbeiten.

Fall 1: Es sei $p \in \mathcal{N}^{++}$. Wir behaupten, dass

$$\rho_p(e) \lesssim \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f - \bar{f}_p\|_{0, \omega_p} \right) \|\nabla e\|_{0, \omega_p} \quad (4.41)$$

gilt, wobei $\mathcal{E}_p^+ = \{E \in \mathcal{E}_p \mid \rho_E \geq -d_E\}$. Da $p \in \mathcal{N}^{++}$ ist, gilt für alle $E \in \mathcal{E}_p$ die Ungleichung $\rho_E \geq -d_E$ ist, d.h. $\mathcal{E}_p^+ = \mathcal{E}_p$. Wir setzen

$$w = (e - c)\phi_p, \quad c = \frac{1}{|\omega_p|} \int_{\omega_p} e \, dx.$$

Da $\mathcal{N}^{++} \subset \mathcal{N}^+ \cap \Omega$ ist, d.h. $u_{\mathcal{S}}(p) > \psi(p)$ ist, gilt

$$\rho_p(c) = c \rho_p(1) \stackrel{(4.23b)}{=} c \cdot 0 = 0.$$

4. Ein hierarchischer Fehlerschätzer für Hindernisprobleme

Damit erhalten wir

$$\begin{aligned}
\rho_p(e) &= \rho_p(e) - \rho_p(c) = \rho_p(e - c) \stackrel{\text{Lem. 4.14}}{=} \int_{\omega_p} f w \, dx + \sum_{E \in \mathcal{E}_p} \int_E j_E w \, ds \\
&\stackrel{"+0"}{=} \int_{\omega_p} f \pi_p w \, dx + \sum_{E \in \mathcal{E}_p} \underbrace{\int_E j_E \pi_p w \, ds}_{\stackrel{(4.28)}{=} \int_E j_E w \, ds} + \int_{\omega_p} f(w - \pi_p w) \, dx \\
&= \rho_S(\pi_p w) + \int_{\omega_p} f(w - \pi_p w) \, dx - \bar{f}_p \underbrace{\int_{\omega_p} (w - \pi_p w) \, dx}_{=0 \text{ wegen (4.28)}} \\
&= \rho_S(\pi_p w) + \int_{\omega_p} (f - \bar{f}_p)(w - \pi_p w) \, dx \\
&\stackrel{\text{C.S.}}{\leq} \sum_{E \in \mathcal{E}_p} \alpha_E(w) \rho_E \|\phi_E\| + \|f - \bar{f}_p\|_{0,\omega_p} \|w - \pi_p w\|_{0,\omega_p}, \tag{4.42}
\end{aligned}$$

wobei im letzten Schritt zusätzlich zur Cauchy-Schwarz-Ungleichung im zweiten Summanden noch angewendet wurde, dass

$$\begin{aligned}
\rho_S(\pi_p w) &= \rho_S \left(\alpha_p(w) \phi_p + \sum_{E \in \mathcal{E}_p} \alpha_E(w) \phi_E \right) \\
&= \alpha_p(w) \underbrace{\rho_S(\phi_p)}_{=\rho_p(1)=0} \sum_{E \in \mathcal{E}_p} \alpha_E(w) \underbrace{\rho_S(\phi_E)}_{\stackrel{(4.11)}{=} \rho_E \|\phi_E\|} \\
&= \sum_{E \in \mathcal{E}_p} \alpha_E(w) \rho_E \|\phi_E\|
\end{aligned}$$

ist. Da $\|\phi_p\|_{\infty,\omega_p} \leq 1$, gilt mit der Poincaré-Friedrichs-Ungleichung (4.39)

$$\|w\|_{0,\omega_p} = \|(e - c)\phi_p\|_{0,\omega_p} \leq \|e - c\|_{0,\omega_p} \lesssim h_p \|\nabla e\|_{0,\omega_p}, \tag{4.43}$$

wobei die erste Ungleichung aus dem Mittelwertsatz der Integralrechnung folgt. Es sei weiterhin darauf hingewiesen, dass $\|\nabla \phi_p\|_{\infty,\omega_p} \lesssim h_p^{-1}$, da die Steigung der Hutfunktion nur von der Form von ω_p abhängt. Damit erhalten wir dann durch Anwenden von (4.34) für alle $E \in \mathcal{E}_p$

$$\begin{aligned}
|\alpha_E(w)| &\lesssim h_p^{-1} (\|w\|_{0,\omega_p} + h_p \|\nabla w\|_{0,\omega_p}) \\
&= h_p^{-1} (\|(e - c)\phi_p\|_{0,\omega_p} + h_p \|\nabla((e - c)\phi_p)\|_{0,\omega_p}) \\
&\leq h_p^{-1} (h_p \|\nabla e\|_{0,\omega_p} + h_p \underbrace{\|\nabla(e - c)\phi_p + (e - c)\nabla \phi_p\|_{0,\omega_p}}_{=\nabla e}) \\
&\stackrel{\Delta \neq}{\leq} \|\nabla e\|_{0,\omega_p} + \|\nabla e \phi_p\|_{0,\omega_p} + \|(e - c)\nabla \phi_p\|_{0,\omega_p} \\
&\lesssim 2 \|\nabla e\|_{0,\omega_p} + h_p^{-1} \|e - c\|_{0,\omega_p} \stackrel{(4.43)}{\lesssim} \|\nabla e\|_{0,\omega_p}.
\end{aligned} \tag{4.44}$$

Über das Referenzelement \hat{T} kann man zeigen, dass

$$\|\phi_E\| = a(\phi_E, \phi_E)^{\frac{1}{2}} = \left(\int_{\Omega} \nabla \phi_E \nabla \phi_E \, dx \right)^{\frac{1}{2}} = \left(\frac{8}{3} (|J_{T_1}| + |J_{T_2}|) \right)^{\frac{1}{2}} \lesssim \tilde{c},$$

da die Jacobi-Determinanten J_{T_1}, J_{T_2} (wobei $E \subset T_i, i = 1, 2$ gilt) endlich sind, jedoch von der Form von ω_p abhängen. Damit gilt

$$\|\nabla e\|_{0,\omega_p} \approx \|\phi_E\|^{-1} \|\nabla e\|_{0,\omega_p}. \quad (4.45)$$

Analog folgt mit Anwendung von (4.35) und (4.43), dass gilt:

$$\begin{aligned} \|w - \pi_p w\|_{0,\omega_p} &\stackrel{\triangle \neq}{\leq} \|w\|_{0,\omega_p} + \|\pi_p w\|_{0,\omega_p} \\ &\lesssim h_p \|\nabla e\|_{0,\omega_p} + \|w\|_{0,\omega_p} + h_p \|\nabla e\|_{0,\omega_p} \lesssim h_p \|\nabla e\|_{0,\omega_p} \end{aligned} \quad (4.46)$$

Setzen wir (4.44) bis (4.46) in (4.42), so folgt nach Ausklammern von $\|\nabla e\|_{0,\omega_p}$ die Behauptung (4.41) mit $\mathcal{E}_p = \mathcal{E}_p^+$.

Fall 2: Es sei $p \in \mathcal{N}^+ \setminus \mathcal{N}^{++}$. Wir behaupten, dass gilt:

$$\rho_p(e) \lesssim \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) \|\nabla e\|_{0,\omega_p}. \quad (4.47)$$

Auch hier können wir analog zu (4.42) eine Ungleichung herleiten, wobei die zweite Addition der Null nicht gilt, da $p \notin \mathcal{N}^{++}$. Damit erhalten wir die Aussage

$$\rho_p(e) \leq \sum_{E \in \mathcal{E}_p} \alpha_E(w) \rho_E \|\phi_E\| + \|f\|_{0,\omega_p} \|w - \pi_p w\|_{0,\omega_p}, \quad (4.48)$$

wobei wir hier

$$w = (e - c)\phi_p, \quad c = \min \left\{ \left(\int_E e \phi_p \, ds \right) \left(\int_E \phi_p \, ds \right)^{-1} \mid E \in \mathcal{E}_p \right\} \quad (4.49)$$

setzen. Damit gilt

$$\begin{aligned} \alpha_E(w) &= \left(\int_E w \, ds \right) \left(\int_E \phi_E \, ds \right)^{-1} = \left(\int_E (e - c)\phi_p \, ds \right) \left(\int_E \phi_E \, ds \right)^{-1} \\ &= \left(\int_E e \phi_p \, ds - c \int_E \phi_p \, ds \right) \left(\int_E \phi_E \, ds \right)^{-1} \\ &\stackrel{(4.49)}{\geq} \frac{\int_E e \phi_p \, ds - (\int_E e \phi_p \, ds) (\int_E \phi_p \, ds)^{-1} (\int_E \phi_p \, ds)}{\int_E \phi_E \, ds} \\ &= 0. \end{aligned}$$

Daraus können wir folgern, dass für die Kanten $E \in \mathcal{E}_p$ mit $\rho_E < -d_E \leq 0$

$$\alpha_E(w)\rho_E \leq 0$$

gilt. Ersetzen wir dies in (4.48), so folgt (4.47) insgesamt mit denselben Abschätzungen aus (4.44), (4.45) und (4.46).

Fall 3: Sei $p \in \mathcal{N} \cap \partial\Omega$. Wir behaupten für diesen Fall, dass

$$\rho_p(e) \lesssim \sum_{E \in \mathcal{E}_p^0} d_E |\rho_E| + \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) \|\nabla e\|_{0,\omega_p} \quad (4.50)$$

mit $\mathcal{E}_p^0 = \mathcal{E}_p \setminus \mathcal{E}_p^+ = \{E \in \mathcal{E}_p \mid \rho_E < -d_E\}$. Auch hier betrachten wir die Ungleichung (4.48), wobei wir dieses Mal $w = e\phi_p$ setzen, d.h. mit der obigen Wahl $c = 0$ setzen. In diesem Fall kann kein anderes c gewählt werden, da wir wegen $p \notin \Omega$ nicht direkt $\rho_p(c) = 0$ aus (4.23b) folgern können. Da $p \in \partial\Omega$ ist, gilt mindestens auf einer Kante $E \in \mathcal{E}_p$ von $\partial\omega_p$

$$e = u - u_S = 0.$$

Damit ist e auf einer Teilmenge, vom Maße ungleich Null, des Randes gleich Null und wir können daher die allgemeine Poincaré-Friedrich-Ungleichung (4.39) anwenden. Damit erhalten wir die Anteile von $E \in \mathcal{E}_p^+$ durch die Abschätzungen (4.44), (4.45) und (4.46). Um die Anteile für die Kanten $E \in \mathcal{E}_p^0$ zu erhalten, betrachten wir

$$\begin{aligned} u_S + w &= u_S + e\phi_p = u_S + (u - u_S)\phi_p \\ &= \underbrace{(1 - \phi_p)u_S}_{\in [0,1]} + \underbrace{\phi_p u}_{\in [0,1]} \\ &\geq (1 - \phi_p)\psi + \phi_p\psi = \psi. \end{aligned}$$

Da die Ungleichung punktweise gilt, folgt auch

$$\int_E w ds \geq \int_E \psi - u_S ds \stackrel{(4.36)}{\implies} \alpha_E(w) \gtrsim \psi(x_E) - u_S(x_E) \stackrel{(4.11)}{=} -d_E \|\phi_E\|^{-1}$$

und daher gilt $\alpha_E(w) \lesssim d_E \|\phi_E\|^{-1}$. Alle Aussagen ersetzt in (4.48) ergeben dann die Behauptung.

Fall 4: Sei $p \in \mathcal{N}^0 \setminus (\mathcal{N}^{0-} \cup \mathcal{N}^{0+})$. Wir behaupten, dass

$$\rho_p(e) \lesssim \sum_{E \in \mathcal{E}_p^0} d_E |\rho_E| + \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) \|\nabla e\|_{0,\omega_p} \quad (4.51)$$

gilt. Um dies zu zeigen, spalten wir den Fehler $e = e^+ + e^-$ auf mit $e^+ := \max\{e, 0\}$ und $e^- := \min\{e, 0\}$. Damit lässt sich der lokale Anteil des Indikators ρ_S schreiben als

$$\rho_p(e) = \rho_p(e^+ + e^-) = \rho_p(e^+) + \rho_p(e^-). \quad (4.52)$$

Wir betrachten zunächst $\rho_p(e^+)$ und setzen analog zum Fall 2

$$w = (e^+ - c)\phi_p, \quad c = \min \left\{ \left(\int_E e^+ \phi_p ds \right) \left(\int_E \phi_p ds \right)^{-1} \mid E \in \mathcal{E}_p \right\}. \quad (4.53)$$

Da $e^+ \geq 0$ ist, gilt auch $c \geq 0$. Wegen $\mathcal{N}^0 \setminus (\mathcal{N}^{0-} \cup \mathcal{N}^{0+}) \subset \mathcal{N} \cap \Omega$ gilt nach (4.23a) $\rho_p(1) \leq 0$, also analog zu Fall 2

$$\begin{aligned} \rho_p(e^+) &\leq \rho_p(e^+) - \underbrace{c \rho_p(1)}_{\leq 0} = \rho_p(e^+ - c) \\ &\leq \sum_{E \in \mathcal{E}_p} \alpha_E(w) \rho_E \|\phi_E\| + \|f\|_{0,\omega_p} \|w - \pi_p w\|_{0,\omega_p}. \end{aligned} \quad (4.54)$$

Die Aussagen (4.53), (4.54) sind identisch zu denen aus Fall 2, wenn wir $e = e^+$ setzen. Damit folgt mit $\|\nabla e^+\|_{0,\omega_p} \leq \|\nabla e\|_{0,\omega_p}$ und denselben Argumenten wie in Fall 2, dass

$$\rho_p(e^+) \lesssim \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) \|\nabla e\|_{0,\omega_p}. \quad (4.55)$$

Wir betrachten nun $\rho_p(e^-)$. Analog zu Fall 3 setzen wir $w = (e^- - c)\phi_p$, $c = 0$ und leiten damit wieder die obere Schranke

$$\rho_p(e^-) \leq \sum_{E \in \mathcal{E}_p} \alpha_E(w) \rho_E \|\phi_E\| + \|f\|_{0,\omega_p} \|w - \pi_p w\|_{0,\omega_p} \quad (4.56)$$

her. Weiter gilt punktweise

$$\begin{aligned} w &= e^- \phi_p \geq e^- = \min\{e, 0\} = \min\{u - u_S, 0\} \\ &\geq \underbrace{\min\{\psi - u_S, 0\}}_{\leq 0} = \psi - u_S \end{aligned}$$

und damit auch die Aussage über $E \in \mathcal{E}_p$ integriert; also folgt aus (4.36)

$$0 \geq \alpha_E(w) \gtrsim \psi(x_E) - u_S(x_E) = -d_E \|\phi_E\|^{-1} \quad \forall E \in \mathcal{E}_p. \quad (4.57)$$

Damit folgt speziell für alle $E \in \mathcal{E}_p^0$ die Abschätzung

$$\begin{aligned} |\alpha_E(w) \rho_E \|\phi_E\|| &= |\alpha_E(w)| |\rho_E| \|\phi_E\| \\ &\stackrel{(4.57)}{\lesssim} d_E \|\phi_E\|^{-1} |\rho_E| \|\phi_E\| = d_E |\rho_E|. \end{aligned} \quad (4.58)$$

Nun bleiben noch die oberen Schranken von $|\alpha_E(w)|$ für $E \in \mathcal{E}_p^+$ und $\|w - \pi_p w\|_{0,\omega_p}$ zu zeigen. Da $p \notin \mathcal{N}^{0+}$, also kein isolierter Knoten ist, gibt es mindestens eine Kante $E \in \mathcal{E}_p$, so dass $u_S = \psi$ gilt. Damit folgt

$$0 = \psi - u_S \leq e^- = \min\{e, 0\} \leq 0 \implies e^- = 0 \text{ auf } E.$$

Wie in Fall 3 ist damit die allgemeine Poincaré-Friedrich-Ungleichung anwendbar und wir können die Aussagen (4.44), (4.45) und (4.46) mit Anwendung von $\|\nabla e^-\|_{0,\omega_p} \leq \|\nabla e\|_{0,\omega_p}$ zeigen. Insgesamt folgt dann mit (4.58)

$$\rho_p(e^-) \lesssim \sum_{E \in \mathcal{E}_p^0} d_E |\rho_E| + \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) \|\nabla e\|_{0,\omega_p}. \quad (4.59)$$

Zusammen mit (4.52), (4.55) und (4.59) folgt dann die Behauptung (4.51).

Fall 5: Es sei nun $p \in \mathcal{N}^{0+}$. Wir behaupten, dass

$$\begin{aligned} \rho_p(e) &\lesssim \sum_{E \in \mathcal{E}_p^0} d_E |\rho_E| \\ &+ \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) (\|\nabla e\|_{0,\omega_p} + \|\nabla(\psi - u_S)\|_{0,\omega_p}) \end{aligned} \quad (4.60)$$

gilt. Wie in Fall 4 verwenden wir die Aufteilung des Indikators nach Gleichung (4.52). Mit genau demselben Vorgehen wie in Fall 4 können wir für $\rho_p(e^+)$ zeigen, dass die Abschätzung (4.55) gilt. Für $\rho_p(e^-)$ können die Aussagen (4.56) bis (4.58) wie in Fall 4 gezeigt werden. Es bleiben also auch hier noch die oberen Schranken von $|\alpha_E(w)|$ für $E \in \mathcal{E}_p^+$ und $\|w - \pi_p w\|_{0,\omega_p}$ zu zeigen.

Wir erinnern uns, dass $\psi - u_S \leq e^- \leq w \leq 0$ gilt und damit folgt

$$\begin{aligned} 0 \geq \alpha_E(w) &= \left(\int_E w \, ds \right) \left(\int_E \phi_E \, ds \right)^{-1} \\ &\geq \left(\int_E \psi - u_S \, ds \right) \left(\int_E \phi_E \, ds \right)^{-1} \\ &= \alpha_E(\psi - u_S). \end{aligned} \quad (4.61)$$

Aus (4.61) folgt

$$\begin{aligned} |\alpha_E(w)| \|\phi_E\| &\leq |\alpha_E(\psi - u_S)| \|\phi_E\| \\ &= \left| \left(\int_E \psi - u_S \, ds \right) \underbrace{\left(\int_E \phi_E \, ds \right)^{-1}}_{\lesssim h_p^{-1}} \right| \cdot \underbrace{\left(\int_{\omega_p} \nabla \phi_E \nabla \phi_E \, dx \right)^{\frac{1}{2}}}_{\lesssim h_p^{\frac{1}{2}}} \\ &\lesssim h_p^{-\frac{1}{2}} \left| \int_E \psi - u_S \, ds \right| \stackrel{\text{C.S.}}{\lesssim} h_p^{-\frac{1}{2}} \|\psi - u_S\|_{0,E} \\ &\lesssim \|\nabla(\psi - u_S)\|_{0,\omega_p}, \end{aligned} \quad (4.62)$$

wobei im letzten Schritt eine *skalierte* Version der Poincaré-Friedrich-Ungleichung¹ verwendet wurde, die darauf basiert, dass $(\psi - u_{\mathcal{S}})(p) = 0$ und $\psi - u_{\mathcal{S}}$ linear ist wegen Voraussetzung 4.1. Wegen $\psi - u_{\mathcal{S}} \leq w$ folgt auch, dass

$$\|w\|_{0,\omega_p} \leq \|\psi - u_{\mathcal{S}}\|_{0,\omega_p} \lesssim h_p \|\nabla(\psi - u_{\mathcal{S}})\|_{0,\omega_p}, \quad (4.63)$$

wobei in (4.63) im letzten Schritt wegen $(\psi - u_{\mathcal{S}})(p) = 0$ wieder die skalierte Version der Poincaré-Friedrich-Ungleichung verwendet wurde. Aus (4.62) folgern wir unter Verwendung der Dreiecksungleichung

$$\begin{aligned} \|\pi_p w\|_{0,\omega_p} &\leq \sum_{E \in \mathcal{E}_p} |\alpha_E(w)| \|\phi_E\|_{0,\omega_p} \\ &\lesssim \sum_{E \in \mathcal{E}_p} |\alpha_E(w)| h_p \underbrace{\|\nabla \phi_E\|_{0,\omega_p}}_{=\|\phi_E\|} \\ &\lesssim h_p \|\nabla(\psi - u_{\mathcal{S}})\|_{0,\omega_p}. \end{aligned} \quad (4.64)$$

Also folgt aus (4.63) und (4.64) insgesamt

$$\|w - \pi_p w\|_{0,\omega_p} \leq \|w\|_{0,\omega_p} + \|\pi_p w\|_{0,\omega_p} \lesssim h_p \|\nabla(\psi - u_{\mathcal{S}})\|_{0,\omega_p}. \quad (4.65)$$

Setzen wir nun für die Kanten $E \in \mathcal{E}_p^0$ die Abschätzung (4.58) und für die Kanten $E \in \mathcal{E}_p^+$ die Ungleichungen (4.62) und (4.65) in die Bedingung (4.56) ein, so erhalten wir die Aussage

$$\rho_p(e^-) \lesssim \sum_{E \in \mathcal{E}_p^0} d_E |\rho_E| + \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) \|\nabla(\psi - u_{\mathcal{S}})\|_{0,\omega_p}. \quad (4.66)$$

Damit folgt mit der Aufteilung (4.52) und den Abschätzungen (4.55) und (4.66) die Behauptung.

Fall 6: Es sei $p \in \mathcal{N}^{0-}$. In diesem Fall haben wir vollen Kontakt, also $u_{\mathcal{S}} = \psi$ auf ω_p und daher gilt

$$e = u - u_{\mathcal{S}} = u - \psi \geq 0 \text{ auf ganz } \omega_p.$$

Weiter gelten für alle $p \in \mathcal{N}^{0-}$ die Eigenschaften $f \leq 0$ auf ω_p und $j_E \leq 0$ für alle $E \in \mathcal{E}_p$. Daher rechnen wir leicht nach, dass gilt:

$$\rho_p(e) = \rho_{\mathcal{S}}(e\phi_p) = \int_{\omega_p} \underbrace{fe\phi_p}_{\leq 0} dx + \sum_{E \in \mathcal{E}_p} \int_E \underbrace{j_E e\phi_p}_{\leq 0} ds \leq 0. \quad (4.67)$$

¹Anschaulich können wir uns dies folgendermaßen vorstellen: Da $\psi - u_{\mathcal{S}}$ linear und am Punkt p gleich ist, ist der Gradient $\nabla(\psi - u_{\mathcal{S}}) = \text{const.}$ Damit kann man mit dieser Konstanten als Höhe mit dem Mittelwertsatz der Integralrechnung die beiden Normen gegenseitig abschätzen, wobei diese Ungleichung abhängig von einer Konstanten c ist, die wiederum nur von der Form von ω_p bzw. E abhängt.

4. Ein hierarchischer Fehlerschätzer für Hindernisprobleme

Bevor wir die sechs Fälle zusammenführen, machen wir uns klar, dass

$$\begin{aligned} & (\mathcal{N}^+ \setminus \mathcal{N}^{++}) \cup (\mathcal{N} \cap \partial\Omega) \cup \overbrace{(\mathcal{N}^0 \setminus (\mathcal{N}^{0-} \cup \mathcal{N}^{0+})) \cup \mathcal{N}^{0+}}^{=\mathcal{N}^0 \setminus \mathcal{N}^{0-}} \\ &= (\mathcal{N} \cap \Omega) \setminus (\mathcal{N}^{0-} \cup \mathcal{N}^{++}) \cup (\mathcal{N} \cap \partial\Omega) = \mathcal{N} \setminus (\mathcal{N}^{0-} \cup \mathcal{N}^{++}) \end{aligned}$$

gilt. Verwenden wir nun die sechs gezeigten Fälle, so ergibt sich:

$$\begin{aligned} \rho_{\mathcal{S}}(e) &= \sum_{p \in \mathcal{N}} \rho_p(e) \\ &= \sum_{E \in \mathcal{E}^0} d_E |\rho_E| + \sum_{p \in \mathcal{N} \setminus \mathcal{N}^{0-}} \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| \right) \|\nabla e\|_{0,\omega_p} \\ &\quad + \sum_{p \in \mathcal{N} \setminus (\mathcal{N}^{0-} \cup \mathcal{N}^{++})} h_p \|f\|_{0,\omega_p} \|\nabla e\|_{0,\omega_p} + \sum_{p \in \mathcal{N}^{++}} h_p \|f - \bar{f}_p\|_{0,\omega_p} \|\nabla e\|_{0,\omega_p} \\ &\quad + \sum_{p \in \mathcal{N}^{0+}} \left(\sum_{E \in \mathcal{E}_p^+} |\rho_E| + h_p \|f\|_{0,\omega_p} \right) \|\nabla(\psi - u_{\mathcal{S}})\|_{0,\omega_p}. \end{aligned}$$

Damit folgt nach der Cauchy-Schwarz-Ungleichung, dass mit einer Konstante $C > 0$, die nur von der Quasi-Uniformität von \mathcal{T}_h abhängt, gilt:

$$\begin{aligned} C \rho_{\mathcal{S}}(e) &\leq \sum_{E \in \mathcal{E}^0} d_E |\rho_E| + \left(\sum_{E \in \mathcal{E}^+} |\rho_E|^2 + \sum_{p \in \mathcal{N} \setminus (\mathcal{N}^{0-} \cup \mathcal{N}^{++})} h_p^2 \|f\|_{0,\omega_p}^2 \right. \\ &\quad \left. + \sum_{p \in \mathcal{N}^{++}} h_p^2 \|f - \bar{f}_p\|_{0,\omega_p}^2 \right)^{\frac{1}{2}} \left(\|\nabla e\|_{0,\Omega}^2 + \sum_{p \in \mathcal{N}^{0+}} \|\nabla(\psi - u_{\mathcal{S}})\|_{0,\omega_p}^2 \right)^{\frac{1}{2}} \\ &= \sum_{E \in \mathcal{E}^0} d_E |\rho_E| \\ &\quad + \left(\sum_{E \in \mathcal{E}^+} |\rho_E|^2 + \text{osc}_2(u_{\mathcal{S}}, \psi, f)^2 \right)^{\frac{1}{2}} \left(\|\nabla e\|_{0,\Omega}^2 + \text{osc}_1(u_{\mathcal{S}}, \psi)^2 \right)^{\frac{1}{2}} \\ &\leq \sum_{E \in \mathcal{E}^0} d_E |\rho_E| \\ &\quad + \frac{\varepsilon}{2} \left(\sum_{E \in \mathcal{E}^+} |\rho_E|^2 + \text{osc}_2(u_{\mathcal{S}}, \psi, f)^2 \right) + \frac{1}{2\varepsilon} \left(\|\nabla e\|_{0,\Omega}^2 + \text{osc}_1(u_{\mathcal{S}}, \psi)^2 \right), \end{aligned}$$

wobei wir als letztes die Ungleichung von Young mit einem $\varepsilon > 0$ verwendet haben und

$$\mathcal{E}^0 = \bigcup_{p \in \mathcal{N}} \mathcal{E}_p^0, \quad \mathcal{E}^+ = \bigcup_{p \in \mathcal{N}} \mathcal{E}_p^+.$$

Wählen wir $\varepsilon \leq C$, so ergibt sich nach leichtem Umstellen der Ungleichung

$$\begin{aligned} c(\varepsilon)\rho_S(e) &\leq \frac{\varepsilon}{2} \left(\sum_{E \in \mathcal{E}^+} |\rho_E|^2 + \text{osc}_2(u_S, \psi, f)^2 \right) \\ &\quad + \frac{1}{2\varepsilon} \text{osc}_1(u_S, f)^2 + \sum_{E \in \mathcal{E}^0} d_E |\rho_E| \\ &\leq \left(1 + \frac{1}{2\varepsilon} \right) \sum_{E \in \mathcal{E}_p} \eta_E |\rho_E| \\ &\quad + \left(\frac{\varepsilon}{2} + \frac{1}{2\varepsilon} \right) (\text{osc}_2(u_S, \psi, f)^2 + \text{osc}_1(u_S, f)^2) \\ &\lesssim \sum_{E \in \mathcal{E}_p} \eta_E |\rho_E| + \text{osc}(u_S, \psi, f)^2 \end{aligned}$$

mit $c(\varepsilon) = \varepsilon - \frac{1}{2\varepsilon}$ und η_E wie in (4.11) definiert. Damit folgt die Behauptung. \square

Mit dem Lemma 4.21 und der Äquivalenz des a posteriori Fehlerschätzers $-\mathcal{I}_Q(\varepsilon_V)$ zum Fehlerindikator $\rho_S(\varepsilon_V)$ folgt das gewünschte Resultat für unseren Fehlerschätzer im nächsten Theorem.

Theorem 4.22. *Es sei Voraussetzung 4.1 für ψ erfüllt. Dann ist der hierarchische Fehlerschätzer $-\mathcal{I}_Q(\varepsilon_V)$ eine obere Schranke für den Fehler im Energiefunktional bis auf Addition von Oszillationstermen und einer Konstante C , die nur von der Quasi-Uniformität von \mathcal{T}_h abhängt, d.h.*

$$J(u_S) - J(u) \lesssim -\mathcal{I}_Q(\varepsilon_V) + \text{osc}(u_S, \psi, f)^2. \quad (4.68)$$

Beweis. Die Aussage folgt direkt durch Lemma 4.12 und 4.21, denn

$$\begin{aligned} J(u_S) - J(u) &= -\mathcal{I}(e) \leq \rho_S(e) \\ &\lesssim \underbrace{\sum_{E \in \mathcal{E}} \eta_E |\rho_E|}_{=\rho_S(\varepsilon_V)} + \text{osc}(u_S, \psi, f)^2 \\ &\leq 2 \cdot (-\mathcal{I}_Q(\varepsilon_V)) + \text{osc}(u_S, \psi, f)^2 \\ &\leq 2 \cdot (-\mathcal{I}_Q(\varepsilon_V) + \text{osc}(u_S, \psi, f)^2) \end{aligned}$$

und damit folgt die Behauptung. \square

An der Abschätzung (4.68) können wir sehen, dass es sinnvoll ist, nicht nur den hierarchischen Fehlerschätzer $-\mathcal{I}_Q(\varepsilon_V)$ von einem Verfeinerungsschritt zum nächsten zu verringern, sondern auch zu fordern, dass die Oszillationsterme $\text{osc}(u_S, \psi, f)$ kleiner werden. Daher wollen wir in unserem Algorithmus später für einen adaptiven Verfeinerungsschritt diese Forderung mit verwenden, dass die Oszillationsterme aus (4.24) verringert werden.

Wie wir in Bemerkung 4.2 gesehen haben, gibt eine obere Schranke vom Fehler der Funktionswerte des Energiefunktionalen J auch eine obere Schranke für den exakten Fehler bzgl. der Energienorm an.

Theorem 4.23. *Es sei die Voraussetzung 4.1 für ψ erfüllt. Dann liefert die Lösung vom lokalisierten Defektpunktproblem (4.9) eine obere Schranke für den exakten Fehler*

$$\|u - u_S\| \lesssim \left(\sum_{E \in \mathcal{E}} \eta_E |\rho_E| \right)^{\frac{1}{2}} + \text{osc}(u_S, \psi, f) \quad (4.69)$$

bis auf Oszillationsterme, wie in (4.24) definiert, und einer Konstante, die nur von der Quasi-Uniformität von \mathcal{T}_h abhängt.

Beweis. Mit Lemma 4.12 und 4.21 folgt direkt

$$\|u - u_S\|^2 \leq 2\rho_S(e) \lesssim \sum_{E \in \mathcal{E}} \eta_E |\rho_E| + \text{osc}(u_S, \psi, f)^2.$$

Nach Wurzel ziehen und Verwendung der Dreiecksungleichung folgt die Behauptung. \square

Wir wollen noch ein Resultat liefern, was die Forderung an die Verringerung der Oszillationsterme in jedem Verfeinerungsschritt begründet. Hierbei handelt es sich um ein analoges Resultat zu Lemma 3.8 aus [MNS00].

Lemma 4.24. *Es sei $0 < \gamma < 1$ ein Parameter, der die Reduktion der Größe des Dreiecks bei Verfeinerung wiedergibt. Weiter sei $0 < \hat{\theta} < 1$ gegeben und eine Menge an Punkten $\hat{\mathcal{N}} \subset \mathcal{N}$, die die zu verfeinernden Dreiecke anzeigen, gegeben, so dass*

$$\text{osc}(u_S, \psi, f, \hat{\mathcal{N}}) \geq \hat{\theta} \text{osc}(u_S, \psi, f, \mathcal{N}).$$

Dann existiert ein $\hat{\alpha} \in (0, 1)$, so dass

$$\text{osc}(u_S, \psi, f, \tilde{\mathcal{N}}) \leq \hat{\alpha} \text{osc}(u_S, \psi, f, \mathcal{N}), \quad (4.70)$$

wobei $\tilde{\mathcal{N}}$ die Menge an Punkten nach Verfeinerung der Triangulierung \mathcal{T}_h bzgl. der Punkte $\hat{\mathcal{N}}$ ist.

Beweisskizze. Es sei $T \in \mathcal{T}_h$ ein Element, das Teilmenge von einem ω_p ist. Da

$$\bar{f}_p = \frac{1}{|\omega_p|} \int_{\omega_p} f dx \text{ bzw. } \bar{f}_T := \frac{1}{|T|} \int_T f dx$$

L^2 -Projektionen von f auf den Raum der stückweise konstanten Funktionen über ω_p bzw. T sind, gilt

$$\|f - \bar{f}_p\|_{0, \omega_p} \leq \|f - \bar{f}_T\|_{0, T}.$$

Weiter können wir auch mit einem Kontraktionsparameter $\beta > 0$ zeigen, dass gilt:

$$\|f - \bar{f}_p\|_{0,\omega_p} \geq \beta \|f - \bar{f}_T\|_{0,T}.$$

Analoge Aussagen ergeben sich für $\|f\|_{0,\omega_p}$ und $\|f\|_{0,T}$. Damit lässt sich der Oszillationsterm $\text{osc}_2(u_S, \psi, f)$ äquivalent beschreiben durch

$$\widetilde{\text{osc}}_2(u_S, \psi, f, \mathcal{T}_h) = \left(\sum_{T \in \mathcal{T}_h^1} h_T^2 \|f - \bar{f}_T\|_{0,T}^2 + \sum_{T \in \mathcal{T}_h^2} h_T^2 \|f\|_{0,T}^2 \right)^{\frac{1}{2}},$$

wobei $\mathcal{T}_h^i, i = 1, 2$ die Menge der Dreiecke ist, über die die Summanden des Oszillationsterms berechnet werden sollen, und h_T wie gewohnt der Radius des Dreiecks ist.

Es sei nun $\hat{\mathcal{T}}_H^i, i = 1, 2$ die zu verfeinernde Menge an größeren Elementen. Da $h_T \leq \gamma h_{\hat{T}}$, ergibt sich

$$\begin{aligned} \widetilde{\text{osc}}_2(u_S, \psi, f, \mathcal{T}_h)^2 &= \sum_{T \in \mathcal{T}_h^1} h_T^2 \|f - \bar{f}_T\|_{0,T}^2 + \sum_{T \in \mathcal{T}_h^2} h_T^2 \|f\|_{0,T}^2 \\ &\leq \gamma^2 \left(\sum_{\hat{T} \in \hat{\mathcal{T}}_H^1} h_{\hat{T}}^2 \|f - \bar{f}_{\hat{T}}\|_{0,\hat{T}}^2 + \sum_{\hat{T} \in \hat{\mathcal{T}}_H^2} h_{\hat{T}}^2 \|f\|_{0,\hat{T}}^2 \right) \\ &\quad + \left(\sum_{T \in \mathcal{T}_H \setminus \hat{\mathcal{T}}_H^1} h_T^2 \|f - \bar{f}_T\|_{0,T}^2 + \sum_{T \in \mathcal{T}_H \setminus \hat{\mathcal{T}}_H^2} h_T^2 \|f\|_{0,T}^2 \right) \\ &\stackrel{+0^*}{=} \underbrace{(\gamma^2 - 1) \widetilde{\text{osc}}_2(u_S, \psi, f, \hat{\mathcal{T}}_H)^2}_{\leq 0} + \widetilde{\text{osc}}_2(u_S, \psi, f, \mathcal{T}_H)^2 \\ &\leq \tilde{\alpha} \widetilde{\text{osc}}_2(u_S, \psi, f, \mathcal{T}_H)^2. \end{aligned}$$

Eine analoge Aussage ergibt sich auch für $\widetilde{\text{osc}}_1(u_S, \psi, \mathcal{T}_h)$. Wegen der Äquivalenz der Darstellung von $\widetilde{\text{osc}}_1, \widetilde{\text{osc}}_2$ zu den Oszillationstermen (4.24), folgt die Behauptung (4.70). \square

4.1.5 Effizienz des Fehlerschätzers

Der Fehlerschätzer $-\mathcal{I}_Q(\varepsilon_V)$ ist für den exakten Fehler des Energiefunktionalen auch effektiv, d.h. wir werden zeigen, dass der hierarchische Fehlerschätzer $-\mathcal{I}_Q(\varepsilon_V)$ auch eine untere Schranke für $-\mathcal{I}(e) = J(u_S) - J(u)$ ist.

Theorem 4.25. *Das Hindernis ψ sei stückweise linear und stetig. Dann ist der hierarchische a posteriori Fehlerschätzer $\mathcal{I}_Q(\varepsilon_V)$ auch eine untere Schranke für den Fehler im Energiefunktional im Sinne von*

$$-\mathcal{I}_Q(\varepsilon_V) \leq 6(J(u_S) - J(u)). \quad (4.71)$$

4. Ein hierarchischer Fehlerschätzer für Hindernisprobleme

Beweis. Zunächst folgt mit (4.16)

$$\begin{aligned} -\mathcal{I}_Q(\varepsilon_V) &\leq \rho_S(\varepsilon_V) = \rho_S \left(\sum_{E \in \mathcal{E}} \varepsilon_V(x_E) \phi_E \right) \\ &= \sum_{E \in \mathcal{E}} \varepsilon_V(x_E) \rho_S(\phi_E) = \sum_{E \in \mathcal{E}} \eta_E |\rho_E| \end{aligned} \quad (4.72)$$

mit $\eta_E = |\varepsilon_V(x_E)| \cdot \|\phi_E\|$ und $\rho_E = \frac{\rho_S(\phi_E)}{\|\phi_E\|}$, wobei man zeigen kann, dass $\text{sign}(\varepsilon_V(x_E)) = \text{sign}(\rho_S(\phi_E))$ gilt. Weiter sollte man erwähnen, dass (4.72) äquivalent ist zu [SV07] Gleichung (2.16).

Das weitere Vorgehen ist ähnlich zum Beweis von Theorem 3.2 aus [SV07]. Es sei

$$\varphi = \frac{1}{3} \sum_{E \in \mathcal{E}} \beta_E \phi_E$$

eine Linearkombination aus Bubble-Funktionen. Dann lässt sich $u_S + \varphi$ auf jedem $T \in \mathcal{T}_h$ durch eine Konvexitätskombination aus $v_E := u_S + \beta_E \phi_E, E \in \mathcal{E}$ schreiben, d.h.

$$(u_S + \varphi) \Big|_T = \frac{1}{3} \sum_{E \in \mathcal{E}, E \subset T} v_E \Big|_T.$$

Da $\mathbb{R}^2 \ni x \mapsto \frac{1}{2}|x|^2$ konvex ist, rechnen wir mit den obigen Bezeichnungen schnell nach, dass gilt

$$\begin{aligned} J(u_S + \varphi) &= \int_{\Omega} \frac{1}{2} |\nabla(u_S + \varphi)|^2 - f(u_S + \varphi) dx \\ &= \sum_{T \in \mathcal{T}_h} \int_T \frac{1}{2} \left| \nabla(u_S + \varphi) \Big|_T \right|^2 - f(u_S + \varphi) \Big|_T dx \\ &= \sum_{T \in \mathcal{T}_h} \int_T \frac{1}{2} \left| \left(\frac{1}{3} \sum_{E \in \mathcal{E}, E \subset T} \nabla v_E \Big|_T \right) \right|^2 - f \left(\frac{1}{3} \sum_{E \in \mathcal{E}, E \subset T} v_E \Big|_T \right) dx \\ &\leq \frac{1}{3} \sum_{E \in \mathcal{E}, E \subset T} \sum_{T \in \mathcal{T}_h} \int_T \frac{1}{2} \left| \nabla v_E \Big|_T \right|^2 - f v_E \Big|_T dx. \end{aligned}$$

Da wir drei Kanten pro Dreieck T haben, gilt analog die Gleichung

$$J(u_S) = \frac{1}{3} \sum_{E \in \mathcal{E}, E \subset T} \sum_{T \in \mathcal{T}_h} \int_T \frac{1}{2} |\nabla u_S|^2 - f u_S dx.$$

Durch Subtraktion der letzten beiden Terme und einigen Umformungen ergibt sich dann

$$J(u_S) - J(u_S + \varphi) \geq \frac{1}{3} \sum_{E \in \mathcal{E}} (J(u_S) - J(u_S + \beta_E \phi_E)). \quad (4.73)$$

Wir rechnen nach, dass für alle $E \in \mathcal{E}$

$$\begin{aligned} J(u_{\mathcal{S}} + \beta_E \phi_E) &= \frac{1}{2} a(u_{\mathcal{S}} + \beta_E \phi_E, u_{\mathcal{S}} + \beta_E \phi_E) - (f, u_{\mathcal{S}} + \beta_E \phi_E) \\ &= J(u_{\mathcal{S}}) + \frac{1}{2} a(\beta_E \phi_E, \beta_E \phi_E) - ((f, \beta_E \phi_E) - a(u_{\mathcal{S}}, \beta_E \phi_E)) \\ &= J(u_{\mathcal{S}}) + \mathcal{I}(\beta_E \phi_E) \end{aligned}$$

gilt. Damit ist das Minimieren von $J(u_{\mathcal{S}} + \beta_E \phi_E)$, so dass $\beta_E \geq -d_E$, mit d_E wie oben definiert, äquivalent ist zum Problem:

$$\min_{\beta_E \geq -d_E} \mathcal{I}(\beta_E \phi_E),$$

was den nächsten Schritt legitimiert. Wir setzen nun $\beta_E = \varepsilon_{\mathcal{V}}(x_E)$, dann gilt, dass $u_{\mathcal{S}} + \beta_E \phi_E \in K$ ist für alle $E \in \mathcal{E}$ und damit aufgrund der Konvexität von K auch $u_{\mathcal{S}} + \varphi \in \mathcal{K}$. Damit folgt insgesamt

$$\begin{aligned} J(u_{\mathcal{S}}) - J(u) &\geq J(u_{\mathcal{S}}) - J(u_{\mathcal{S}} + \varphi) \\ &\geq \frac{1}{3} \sum_{E \in \mathcal{E}} (J(u_{\mathcal{S}}) - J(u_{\mathcal{S}} + \beta_E \phi_E)) = \frac{1}{3} \sum_{E \in \mathcal{E}} -\mathcal{I}(\beta_E \phi_E) \\ &= \frac{1}{3} \sum_{E \in \mathcal{E}} \left(\rho_{\mathcal{S}}(\beta_E \phi_E) - \frac{1}{2} a(\beta_E \phi_E, \beta_E \phi_E) \right) \\ &= \frac{1}{3} \sum_{E \in \mathcal{E}} \left(\beta_E \rho_{\mathcal{S}}(\phi_E) - \frac{1}{2} \beta_E^2 a(\phi_E, \phi_E) \right) \\ &\geq \frac{1}{3} \sum_{E \in \mathcal{E}} \left(\underbrace{\frac{\max\{-d_E, \rho_E\}}{\|\phi_E\|} \rho_{\mathcal{S}}(\phi_E)}_{=\eta_E |\rho_E|} - \frac{1}{2} \underbrace{\frac{\max\{-d_E, \rho_E\}^2}{\|\phi_E\|^2} \|\phi_E\|^2}_{\geq \eta_E |\rho_E|} \right) \\ &= \frac{1}{3} \sum_{E \in \mathcal{E}} \left(\underbrace{\max\{-d_E, \rho_E\} \rho_E}_{=\eta_E |\rho_E|} - \frac{1}{2} \underbrace{\max\{-d_E, \rho_E\}^2}_{\geq \eta_E |\rho_E|} \right) \\ &\geq \frac{1}{3} \sum_{E \in \mathcal{E}} \frac{1}{2} \eta_E |\rho_E| = \frac{1}{6} \sum_{E \in \mathcal{E}} \eta_E |\rho_E|. \end{aligned}$$

Zusammen mit (4.72) folgt dann die Behauptung. \square

4.2 Ein adaptiver Algorithmus

Mit den Resultaten aus Kapitel 4.1 können wir nun einen implementierbaren Algorithmus für eine adaptive Verfeinerung angeben.

Mittels Theorem 4.22 bildet der hierarchische a posteriori Fehlerschätzer $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}})$ eine obere Schranke für den exakten Fehler $J(u_{\mathcal{S}}) - J(u)$ (und wegen Theorem 4.25 auch eine untere), d.h. dass die Verringerung unseres Schätzers auch eine Verkleinerung des Fehlers mit sich führt. Aufgrund von

Korollar 4.13 ist außerdem der Fehlerschätzer $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}})$ äquivalent zum Fehlerindikator $\rho_{\mathcal{S}}(\varepsilon_{\mathcal{V}})$, den wir durch Lemma 4.14 in seine lokalen Anteile bzgl. der einzelnen Knoten aufteilen können.

Zuletzt bleibt noch aus, dass auch die Oszillationsterme in der oberen Schranke von Theorem 4.22 enthalten sind. Diese sollten sich also in einem Verfeinerungsschritt nicht vergrößern. Diese Forderung können wir mittels Lemma 4.24 (durch möglicherweise weitere Verfeinerung) erfüllen.

Algorithm 4.1 Adaptive Verfeinerungsstrategie für ein Hindernisproblem

Gegeben sei eine Fehlerschranke $\varepsilon > 0$, Parameter $\theta_1, \theta_2 \in (0, 1)$ und ein initiales Gitter \mathcal{T}_0 . Wir setzen $\mathcal{T}_n := \mathcal{T}_0$.

- 1: Berechne die Galerkin-Lösung $u_{\mathcal{S}}$ von (3.12) über der Zerlegung \mathcal{T}_n .
- 2: Berechne die Lösung $\varepsilon_{\mathcal{V}}$ vom lokalen Defektproblem (4.9) mittels (4.10) und berechne damit $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}})$.
- 3: Berechne die lokalen Anteile ρ_p des Fehlerindikators $\rho_{\mathcal{S}}$ mit Lemma 4.14 für alle $p \in \mathcal{N}$ und damit auch $\rho_{\mathcal{S}}$ mit (4.20).
- 4: Weiter berechne die Mengen $\mathcal{N}^{0+}, \mathcal{N}^{++}$ und \mathcal{N}^{0-} um die lokalen Anteile (also die einzelnen Summanden) von $\text{osc}_1(u_{\mathcal{S}}, \psi)$ und $\text{osc}_2(u_{\mathcal{S}}, \psi, f)$ sowie die globalen Oszillationsterme $\text{osc}(u_{\mathcal{S}}, \psi, f)$ zu bestimmen.
- 5: **if** $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}}) < \varepsilon$ **then**
- 6: **break**
- 7: **end if**
- 8: Finde die kleinste Menge an Punkten $\hat{\mathcal{N}} \subset \mathcal{N}$, so dass gilt

$$\sum_{p \in \hat{\mathcal{N}}} \rho_p(\varepsilon_{\mathcal{V}}) \geq \theta_1 \rho_{\mathcal{S}}(\varepsilon_{\mathcal{V}}).$$

- 9: Erweitere, falls notwendig, $\hat{\mathcal{N}}$ zu einer Menge an Punkten $\tilde{\mathcal{N}} \supset \hat{\mathcal{N}}$, so dass gilt

$$\text{osc}(u_{\mathcal{S}}, \psi, f, \tilde{\mathcal{N}}) \geq \theta_2 \text{osc}(u_{\mathcal{S}}, \psi, f).$$
 - 10: Bestimme aus $\tilde{\mathcal{N}}$ die angrenzenden Dreiecke $\tilde{\mathcal{T}}_n \subset \mathcal{T}_n$.
 - 11: Erhalte die neue Triangulierung \mathcal{T}_{n+1} durch die Verfeinerung des Gitters \mathcal{T}_n über die zu verfeinernden Dreiecke $\tilde{\mathcal{T}}_n$.
 - 12: Setze $n \leftarrow n + 1$ und Springe zu Schritt 1.
-

4.3 Erfüllung einer Saturationseigenschaft

In Kapitel 2.4, in dem wir adaptive Verfeinerungsstrategien für Variationsgleichungen beschrieben haben, musste für den hierarchischen Fehlerschätzer eine Saturationseigenschaft erfüllt werden. Dies ist hier nicht benötigt worden, dennoch wird eine solche äquivalente Eigenschaft erfüllt. Wir wollen daher in diesem Kapitel zeigen, dass das für die quadratische Finite-Element-

Approximation u_Q mit

$$u_Q \in K_Q : \quad a(u_Q, v - u_Q) \geq (f, v - u_Q) \quad \forall v \in K_Q, \quad (4.74)$$

wobei $K_Q = \{v \in Q \mid v(p) \geq \psi(p) \forall p \in N_Q \cap \Omega\}$ ist, die Saturationseigenschaft

$$J(u_Q) - J(u) \leq \alpha(J(u_S) - J(u)) \quad (4.75)$$

mit $\alpha \in (0, 1)$ erfüllt, wenn die Oszillationsterme $\text{osc}(u_S, \psi, f)$ relativ klein sind. Hierfür werden wir zunächst die Äquivalenz zu einer Ungleichung abhängig von $-\mathcal{I}(e_Q)$ zeigen, um dann aufgrund der Äquivalenz von $-\mathcal{I}(e_Q)$ zum hierarchischen Fehlerschätzer das gewünschte Resultat zu erhalten.

Lemma 4.26. *Es sei $\alpha \in (0, 1)$. Dann gilt die Saturationseigenschaft (4.75) genau dann, wenn gilt:*

$$J(u_S) - J(u) \leq \frac{-\mathcal{I}(e_Q)}{1 - \alpha}. \quad (4.76)$$

Beweis. Es sei e_Q Lösung von Problem (4.5) bzw. u_Q Lösung von (4.74), damit gilt $u_Q = u_S + e_Q$. Es gilt

$$\begin{aligned} & -\mathcal{I}(e_Q) + (J(u_Q) - J(u)) \\ &= \rho_S(e_Q) - \frac{1}{2}a(e_Q, e_Q) + \frac{1}{2}a(u_Q, u_Q) - (f, u_Q) - J(u) \\ &= (f, e_Q) - a(u_S, e_Q) + a(u_S, u_Q) - \frac{1}{2}a(u_S, u_S) - (f, u_Q) - J(u) \\ &= \underbrace{\frac{1}{2}a(u_S, u_S) - (f, u_S)}_{=J(u_S)} - J(u) = J(u_S) - J(u). \end{aligned}$$

Damit folgt, wenn (4.75) gilt,

$$\begin{aligned} J(u_S) - J(u) &= -\mathcal{I}(e_Q) + (J(u_Q) - J(u)) \\ &\leq -\mathcal{I}(e_Q) + \alpha(J(u_S) - J(u)) \end{aligned}$$

und mit Umformung nach $J(u_S) - J(u)$ folgt (4.76). Analog folgt, wenn (4.76) gilt:

$$(1 - \alpha)(J(u_S) - J(u)) \leq -\mathcal{I}(e_Q) = J(u_S) - J(u_Q)$$

und damit folgt nach leichter Umformung (4.75). \square

Lemma 4.27. *Es gilt*

$$\mathcal{I}(e_Q) \lesssim \mathcal{I}_Q(\varepsilon_Q) \leq \mathcal{I}_Q(\varepsilon_V) \leq 0. \quad (4.77)$$

4. Ein hierarchischer Fehlerschätzer für Hindernisprobleme

Beweis. Es bezeichne ε_Q die Lösung des Minimierungsproblems: Finde $\varepsilon_Q \in \mathcal{A}_Q$, so dass

$$\mathcal{I}_Q(\varepsilon_Q) \leq \mathcal{I}_Q(v) \quad \forall v \in \mathcal{A}_Q. \quad (4.78)$$

Weiter sei $\varepsilon_V \in \mathcal{A}_V$ die Lösung von (4.8). Da $0 \in \mathcal{A}_V \subset \mathcal{A}_Q$ liegt, folgt durch Einsetzen von ε_V in die obige Gleichung bzw. $v = 0$ in (4.8) die Behauptung

$$\mathcal{I}_Q(\varepsilon_Q) \leq \mathcal{I}_Q(\varepsilon_V) \leq 0.$$

Analog zu Lemma 4.12 bzw. Korollar 4.13 gilt auch für ε_Q die Ungleichung $-\mathcal{I}_Q(\varepsilon_Q) \leq \rho_S(\varepsilon_Q)$. Gilt nun $\rho_S(\varepsilon_Q) \lesssim \rho_S(e_Q)$, so folgt mit (4.16)

$$\mathcal{I}(e_Q) \leq -\frac{1}{2}\rho_S(e_Q) \lesssim -\rho_S(\varepsilon_Q) \leq \mathcal{I}_Q(\varepsilon_Q).$$

Es bleibt also $\rho_S(\varepsilon_Q) \lesssim \rho_S(e_Q)$ zu zeigen. Setzen wir $v = \varepsilon_Q \in \mathcal{A}_Q$ in Problem (4.5) ein und verwenden die Cauchy-Schwarz-Ungleichung, sowie die für ε_Q zu (4.15) und (4.16) analogen Aussagen, dann ergibt sich

$$\begin{aligned} \rho_S(\varepsilon_Q - e_Q) &\leq a(e_Q, \varepsilon_Q - e_Q) \stackrel{\text{C.S.}}{\leq} \|e_Q\| \|\varepsilon_Q - e_Q\| \\ &\leq \rho_S(e_Q)^{\frac{1}{2}} \|\varepsilon_Q - e_Q\|. \end{aligned} \quad (4.79)$$

Da $e_Q, \varepsilon_Q \in \mathcal{A}_Q$ Lösungen von (4.5) bzw. (4.78) sind, folgt aus den dazugehörigen Variationsungleichungen

$$-a(e_Q, \varepsilon_Q - e_Q) \leq \rho_S(e_Q - \varepsilon_Q) \leq a_Q(\varepsilon_Q, e_Q - \varepsilon_Q). \quad (4.80)$$

Dann folgt wegen der Äquivalenz von $\|\cdot\|_Q$ und $\|\cdot\|$ (vgl. Satz 4.8), der Cauchy-Schwarz-Ungleichung und (4.80) die obere Schranke

$$\begin{aligned} \|\varepsilon_Q - e_Q\|^2 &= a(\varepsilon_Q - e_Q, \varepsilon_Q - e_Q) \\ &= a(\varepsilon_Q, \varepsilon_Q - e_Q) - a(e_Q, \varepsilon_Q - e_Q) \\ &\stackrel{(4.80)}{\leq} a(\varepsilon_Q, \varepsilon_Q - e_Q) + a_Q(\varepsilon_Q, e_Q - \varepsilon_Q) \\ &\stackrel{\text{C.S.}}{\leq} \|\varepsilon_Q\| \|\varepsilon_Q - e_Q\| + \|\varepsilon_Q\|_Q \|\varepsilon_Q - e_Q\|_Q \\ &\stackrel{\text{Satz 4.8}}{\lesssim} \|\varepsilon_Q\|_Q \|\varepsilon_Q - e_Q\|. \end{aligned}$$

Mit Division durch $\|\varepsilon_Q - e_Q\|$ erhalten wir dann

$$\|\varepsilon_Q - e_Q\| \lesssim \|\varepsilon_Q\|_Q \leq \rho_S(\varepsilon_Q)^{\frac{1}{2}},$$

wobei wir im letzten Schritt wieder für ε_Q die zu (4.15) äquivalente Aussage verwendet haben. Setzen wir dies in (4.79) ein, so erhalten wir

$$\rho_S(\varepsilon_Q - e_Q) \leq c \rho_S(e_Q)^{\frac{1}{2}} \rho_S(\varepsilon_Q)^{\frac{1}{2}} \stackrel{\text{Young}}{\leq} \frac{c^2}{2} \rho_S(e_Q) + \frac{1}{2} \rho_S(\varepsilon_Q)$$

mit einer Konstanten $C > 0$, die nur von der Quasi-Uniformität von \mathcal{T}_h abhängt. Nach Anwendung der Linearität von ρ_S auf der linken Seite und Umformung nach $\rho_S(\varepsilon_Q)$ folgt die Behauptung durch

$$\rho_S(\varepsilon_Q) \leq (2 + c^2) \rho_S(e_Q) \lesssim \rho_S(e_Q). \quad \square$$

Theorem 4.28. *Es gibt Konstanten $c > 0$ und $\alpha \in (0, 1)$, die nur von der Quasi-Uniformität von \mathcal{T}_h abhängt, so dass kleine Oszillation im Sinne von*

$$\text{osc}(u_S, \psi, f)^2 \leq c(J(u_S) - J(u)) \quad (4.81)$$

die Saturationseigenschaft (4.75)

$$J(u_Q) - J(u) \leq \alpha(J(u_S) - J(u))$$

impliziert.

Beweis. Mit Theorem 4.22 und Lemma 4.27 folgt, dass es zwei Konstanten c_1, c_2 gibt, die nur von der Quasi-Uniformität von \mathcal{T}_h abhängen, so dass

$$\begin{aligned} J(u_S) - J(u) &\leq -c_1 \mathcal{I}_Q(\varepsilon_V) + c_2 \text{osc}(u_S, \psi, f)^2 \\ &\stackrel{(4.83)}{\lesssim} -c_1 \mathcal{I}(e_Q) + c_2 c (J(u_S) - J(u)). \end{aligned}$$

Wir formen die letzte Aussage nach dem Fehler im Energiefunktional um, so dass

$$J(u_S) - J(u) \lesssim \frac{-\mathcal{I}(e_Q)}{\frac{1-c_2c}{c_1}}$$

ist und damit gilt für $1 - \alpha = \frac{1-c_2c}{c_1}$, dass

$$c = \frac{1 - c_1(1 - \alpha)}{c_2}$$

sein muss. Setzen wir nun $c_1 \geq 1$, so gilt für $\alpha \in \left(\frac{c_1-1}{c_1}, 1\right) \subset (0, 1)$, damit $c > 0$ ist. Damit sind die Voraussetzungen aus Lemma 4.26 erfüllt und somit folgt die Behauptung (4.75). \square

4.4 Übertragung des Fehlerschätzers auf Kontaktprobleme

Nun wollen wir das in Kapitel 4.1 vorgestellte Konzept auf das Kontaktproblem (3.38) übertragen. Hierfür haben wir in Kapitel 3.2 schon gesehen, dass dieses äquivalent zur Variationsungleichung:

$$\mathbf{u} \in \mathcal{K} : \quad a(\mathbf{u}, \mathbf{v} - \mathbf{u}) \geq F(\mathbf{v} - \mathbf{u}) \quad \forall \mathbf{v} \in \mathcal{K}, \quad (4.82)$$

wobei \mathcal{K} die konvexe, abgeschlossene Menge aus (3.41) ist, a die stetige, koerzitive Bilinearform (3.44a) und F die stetige Linearform (3.44b).

Diskretisierung des lokalen Defektproblems

Die Diskretisierung des Defektproblems ist analog zu Kapitel 4.1.1 durchzuführen. Wir betrachten hierfür die diskretisierte Variationsungleichung (3.55):

$$\mathbf{u}_h \in \mathcal{K}_h : \quad a(\mathbf{u}_h, \mathbf{v}_h - \mathbf{u}_h) \geq F(\mathbf{v}_h - \mathbf{u}_h) \quad \forall \mathbf{v}_h \in \mathcal{K}_h. \quad (4.83)$$

mit $\mathcal{K}_h := \{\mathbf{v}_h \in \mathcal{S}_h \mid \mathbf{v}_h(\bar{x}_i) \cdot \mathbf{n} - g(\bar{x}_i) \leq 0 \text{ mit } \bar{x}_i \in \mathcal{N} \cap \Gamma_c\}$. Wie wir schon gesehen haben, existiert auch hierfür eine äquivalente Minimierungsproblem eines Energiefunktionalen:

$$\mathbf{u}_h \in \mathcal{K}_h : \quad J(\mathbf{u}_h) \leq J(\mathbf{v}_h) \quad \forall \mathbf{v}_h \in \mathcal{K}_h, \quad (4.84)$$

wobei $J(\mathbf{v}) = \frac{1}{2}a(\mathbf{v}, \mathbf{v}) - F(\mathbf{v})$ ist.

Notation. Wir wollen der Einfachheit halber auch wieder statt \mathbf{u}_h die Notation $\mathbf{u}_{\mathcal{S}}$ für $\mathbf{u}_h \in \mathcal{S}_h$ verwenden. Analog sind zudem die übrigen Indizes zu verstehen (vgl. obige Notation).

Betrachten wir wieder die exakte Fehlerfunktion $\mathbf{e} = \mathbf{u} - \mathbf{u}_{\mathcal{S}}$ und das Funktional

$$\mathcal{I}(\mathbf{v}) = \frac{1}{2}a(\mathbf{v}, \mathbf{v}) - \rho_{\mathcal{S}}(\mathbf{v}) \text{ mit } \rho_{\mathcal{S}}(\mathbf{v}) = a(\mathbf{u}_{\mathcal{S}}, \mathbf{v}) - F(\mathbf{v}),$$

so gelten für das Funktional \mathcal{I} wegen der Linearität von F die Aussagen von Satz 4.4 und Korollar 4.5 bzgl. der Menge

$$\mathcal{A} = \{\mathbf{v} \in (H_{\Gamma_u}^1(\Omega))^2 \mid v_n - (g - \mathbf{u}_{\mathcal{S}} \cdot \mathbf{n}) \leq 0 \text{ auf } \Gamma_c\} = -\mathbf{u}_{\mathcal{S}} + \mathcal{K}.$$

Betrachten wir nun die Hierarchie $\mathcal{Q}_h \subset \mathcal{S}_h$ mit

$$\mathcal{Q}_h := \{\mathbf{v} \in (C^0(\Omega))^2 \mid \mathbf{v}|_T \in (\mathcal{P}_2)^2 \text{ für } T \in \mathcal{T}_h, \mathbf{v}_{\Gamma_u} = \mathbf{0}\},$$

so erhalten wir analog zu (4.4) und (4.5) das diskrete Defektproblem:

$$\mathbf{e}_{\mathcal{Q}} \in \mathcal{A}_{\mathcal{Q}} : \quad a(\mathbf{e}_{\mathcal{Q}}, \mathbf{v} - \mathbf{e}_{\mathcal{Q}}) \geq F(\mathbf{v} - \mathbf{e}_{\mathcal{Q}}) \quad \forall \mathbf{v} \in \mathcal{A}_{\mathcal{Q}} \quad (4.85)$$

mit $\mathcal{A}_{\mathcal{Q}} = \{\mathbf{v} \in \mathcal{Q}_h \mid v_n(p) - (g(p) - (\mathbf{u}_{\mathcal{S}})_n(p)) \leq 0 \forall p \in \mathcal{N}_{\mathcal{Q}} \cap \Gamma_c\}$, wobei $\mathcal{N}_{\mathcal{Q}} = \mathcal{N}_{\mathcal{Q}}$ wieder die Menge der Mittelpunkte auf den Kanten $E \in \mathcal{E}$ bezeichnet. Da \mathcal{S}_h bzw. \mathcal{Q}_h komponentenweise mit den Räumen \mathcal{S}_h und \mathcal{Q}_h übereinstimmt, können wir analog die Aussage von Satz 4.7 im Zweidimensionalen zeigen und somit den hierarchischen Split $\mathcal{Q}_h = \mathcal{S}_h \oplus \mathcal{V}_h$ übertragen, wobei \mathcal{V}_h der Raum der zweidimensionalen, quadratischen Bubble-Funktionen ist.

Damit lässt sich jede Funktion $\mathbf{v}_{\mathcal{Q}} \in \mathcal{Q}_h$ schreiben als $\mathbf{v}_{\mathcal{Q}} = \mathbf{v}_{\mathcal{S}} + \mathbf{v}_{\mathcal{V}}$ und somit auch die zu $\|\cdot\|^2 = a(\cdot, \cdot)$ äquivalente Norm

$$\begin{aligned} a_{\mathcal{Q}}(\mathbf{v}, \mathbf{w}) &= a(\mathbf{v}_{\mathcal{S}}, \mathbf{w}_{\mathcal{S}}) + \sum_{E \in \mathcal{E}} ((\mathbf{u}_{\mathcal{V}}(x_E))_x (\mathbf{v}_{\mathcal{V}}(x_E))_x a(\psi_{E_x}, \psi_{E_x}) \\ &\quad + (\mathbf{u}_{\mathcal{V}}(x_E))_y (\mathbf{v}_{\mathcal{V}}(x_E))_y a(\psi_{E_y}, \psi_{E_y})) \quad \forall \mathbf{v}, \mathbf{w} \in \mathcal{Q}_h \end{aligned}$$

analog zu a_Q einführen, wobei ψ_{E_x}, ψ_{E_y} die Bubble-Funktionen für die Kante $E \in \mathcal{E}$ bzgl. der x- und y-Verschiebungen, bzw. $(\cdot)_x, (\cdot)_y$ die x- und y-Komponenten der dort betrachteten Vektoren sind. Damit können wir analog zu (4.6) bis (4.9) das diskrete lokale Defektproblem aufstellen:

$$\boldsymbol{\varepsilon}_V \in \mathcal{A}_V : \quad \mathcal{J}_Q(\boldsymbol{\varepsilon}_V) \leq \mathcal{J}_Q(\mathbf{v}) \quad \forall \mathbf{v} \in \mathcal{A}_V \quad (4.86)$$

mit $\mathcal{A}_V := \{\mathbf{v} \in \mathcal{V}_h \mid v_n(x_E) - (g(x_E) - (u_S)_n(x_E)) \leq 0 \forall E \subset \Gamma_c\}$. Weiter erhalten wir die äquivalente Variationsungleichung zu (4.86):

$$\boldsymbol{\varepsilon}_V \in \mathcal{A}_V : \quad a_Q(\boldsymbol{\varepsilon}_V, \mathbf{v} - \boldsymbol{\varepsilon}_V) \geq \rho_S(\mathbf{v} - \boldsymbol{\varepsilon}_V) \quad \forall \mathbf{v} \in \mathcal{A}_V, \quad (4.87)$$

da auch hier \mathcal{J}_Q die notwendigen Bedingungen zum Anwenden von Satz A.11 aufgrund der zugrunde liegenden Eigenschaften von a und F erfüllt. Die Variationsungleichung (4.87) können wir dann wieder mit Methoden aus der Nichtlinearen Optimierung wie schon mehrmals angesprochen lösen.

Es ist also sinnvoll auch für Kontaktprobleme sich den hierarchischen a posteriori Fehlerschätzer

$$-\mathcal{J}_Q(\boldsymbol{\varepsilon}_V) = -\frac{1}{2}a_Q(\boldsymbol{\varepsilon}_V, \boldsymbol{\varepsilon}_V) + \rho_S(\boldsymbol{\varepsilon}_V)$$

näher anzusehen.

Lokaler Anteil des Fehlerschätzers

Da wir schon gezeigt haben, dass a unter bestimmten Bedingungen eine koerzitive, stetige Bilinearform und F eine stetige Bilinearform ist, gelten die Aussagen aus dem Lemma 4.12 auch für \mathcal{J} bzw. ρ_S . Da sich die Eigenschaften von a auch auf a_Q übertragen gilt auch das Korollar 4.13 bezogen auf die Bilinearform a_Q und Linearform $\rho_S(\cdot) = F(\cdot) - a_Q(\mathbf{u}_S, \cdot)$. Daher ist ρ_S auch für Kontaktprobleme äquivalent zum Fehlerschätzer $-\mathcal{J}_Q$ und kann als Fehlerindikator verwendet werden.

Bei der lokalen Aufteilung gibt es jetzt rechnerische Unterschiede zum skalaren Fall aus Kapitel 4.1.2, die hier erörtert werden sollen; das grundlegende Vorgehen bleibt jedoch gleich. Da \mathbf{u}_S in jeder Komponente linear ist über einem Dreieck $T \in \mathcal{T}_h$, gilt die Identität

$$\operatorname{div}(\boldsymbol{\sigma}(\mathbf{u}_S)) = \operatorname{div}(\mathcal{C} : \boldsymbol{\varepsilon}(\mathbf{u}_S)) = \mathbf{0}, \quad (4.88)$$

wenn \mathcal{C} ein konstanter Tensor ist. Damit lässt sich der Fehlerindikator für

ein beliebiges $\mathbf{v} \in (H^1(\Omega))^2$ schreiben als

$$\begin{aligned}
 \rho_S(\mathbf{v}) &= F(\mathbf{v}) - a(\mathbf{u}_S, \mathbf{v}) = F(\mathbf{v}) - \int_{\Omega} \boldsymbol{\varepsilon}(\mathbf{u}_S) : \mathcal{C} : \boldsymbol{\varepsilon}(\mathbf{v}) d\Omega \\
 &= F(\mathbf{v}) - \sum_{T \in \mathcal{T}_h} \int_T \underbrace{\boldsymbol{\varepsilon}(\mathbf{u}_S)}_{=\boldsymbol{\sigma}(\mathbf{u}_S)} : \mathcal{C} : \boldsymbol{\varepsilon}(\mathbf{v}) d\Omega \\
 &\stackrel{\sigma=\sigma^T}{=} F(\mathbf{v}) - \sum_{T \in \mathcal{T}_h} \int_T \underbrace{\boldsymbol{\sigma}(\mathbf{u}_S) : \text{grad}(\mathbf{v})}_{\stackrel{(C.2)}{=} \text{div}(\boldsymbol{\sigma}(\mathbf{v} \cdot \mathbf{u}_S)) - \mathbf{v} \cdot \text{div}(\boldsymbol{\sigma}(\mathbf{u}_S))} d\Omega \\
 &\stackrel{(C.3a)}{=} F(\mathbf{v}) - \sum_{T \in \mathcal{T}_h} \int_{\partial T} \mathbf{v} \cdot \boldsymbol{\sigma}(\mathbf{u}_S) \cdot \mathbf{n} d\Gamma. \tag{4.89}
 \end{aligned}$$

Jetzt lässt sich (4.89) analog zu der Gleichung (4.18) auch als Summe über die Kanten $E \in \mathcal{E}$ beschreiben. Wenn wir \mathbf{n} wieder als die Einheitsnormale bezeichnen, die von einem gegebenen Dreieck T_1 in das Dreieck T_2 zeigt, dann lässt sich der *Nomalenspannungsfluss* über E berechnen als

$$\mathbf{j}_E := \boldsymbol{\sigma}(\mathbf{u}_S|_{T_2}) \cdot \mathbf{n} - \boldsymbol{\sigma}(\mathbf{u}_S|_{T_1}) \cdot \mathbf{n}.$$

Damit lässt sich (4.89) umformen zu

$$\rho_S(\mathbf{v}) = \int_{\Omega} \mathbf{b} \cdot \mathbf{v} d\Omega + \int_{\Gamma_\sigma} \mathbf{t} \cdot \mathbf{v} d\Gamma + \sum_{E \in \mathcal{E}} \int_E \mathbf{v} \cdot \mathbf{j}_E d\Gamma. \tag{4.90}$$

Um nun den lokalen Anteil von ρ_S bzgl. eines Punktes $p \in \mathcal{N}$ zu erhalten, gewichten wir genau wie in (4.19) die Funktion \mathbf{v} mit der Hutfunktion ϕ_p , d.h.

$$\rho_p(\mathbf{v}) := \rho_S(\mathbf{v} \phi_p) \quad \text{mit } \mathbf{v} \in (H^1(\Omega))^2, p \in \mathcal{N}. \tag{4.91}$$

Wegen der Zerlegung der Eins durch ϕ_p können wir die Aussagen von Lemma 4.14 und Korollar 4.15 direkt übertragen, wir erhalten also

$$\rho_p(\mathbf{v}) = \int_{\omega_p} \mathbf{b} \cdot \mathbf{v} \phi_p d\Omega + \int_{\substack{E \subset \Gamma_\sigma \\ E \in \mathcal{E}_p}} \mathbf{t} \cdot \mathbf{v} \phi_p d\Gamma + \sum_{E \in \mathcal{E}_p} \int_E \mathbf{v} \cdot \mathbf{j}_E \phi_p d\Gamma, \tag{4.92}$$

wobei ω_p und \mathcal{E}_p wie oben definiert sind.

HIER FEHLT NOCH DIE IDEE DER LOKALEN ANTEILE

Für die Übertragung der Oszillationsterme betrachten wir wieder zwei Teilmengen der Knotenmenge \mathcal{N} .

Definition 4.29. Wir definieren die Menge der *Kontaktknoten* \mathcal{N}^0 und die Menge der *Nichtkontaktknoten* \mathcal{N}^+ analog zu Definition 4.16 durch

$$\mathcal{N}^0 := \{p \in \mathcal{N} \mid (u_S)_n(p) = g(p)\}, \quad \mathcal{N}^+ := \{p \in \mathcal{N} \mid (u_S)_n(p) < g(p)\},$$

wobei $(u_S)_n = \mathbf{u}_S \cdot \mathbf{n}$ ist.

Es sei hier angemerkt, dass nun nicht mehr die inneren Knoten betrachtet werden wie in Definition 4.16, sondern alle Knoten aus \mathcal{N} , da diejenigen Knoten, die in Kontakt stehen können, in den von uns betrachteten Problemen immer auf dem Rand des Gebietes Ω liegen.

Oszillationsterme

Wir definieren die Oszillationen analog zu Kapitel 4.1.3 durch

$$\text{osc}(\mathbf{u}_S, g, \mathbf{b}, \mathbf{t}) := (\text{osc}_1(\mathbf{u}_S, g)^2 + \text{osc}_2(\mathbf{u}_S, g, \mathbf{b})^2)^{\frac{1}{2}}.$$

Dabei ist osc_1 wieder ein Maß für die Hindernisoszillation, d.h.

$$\text{osc}_1(\mathbf{u}_S, g) := \left(\sum_{p \in \mathcal{N}^{0+}} \|\nabla(g - (u_S)_n)\|_{0, \omega_p}^2 \right)^{\frac{1}{2}},$$

wobei $\mathcal{N}^{0+} := \{p \in \mathcal{N}^0 \mid (u_S)_n - g < 0 \text{ in } \omega_p \setminus \{p\}\}$ wieder die isolierten Kontaktknoten enthält. Weiter definieren wir analog zu (4.25) und (4.26) die Mengen

$$\begin{aligned} \mathcal{N}^{++} &:= \{p \in \mathcal{N}^+ \mid (\varepsilon_V)_n(x_E) < g(x_E) - (u_S)_n \forall E \in \mathcal{E}_p, E \subset \Gamma_c\}, \\ \mathcal{N}^{0-} &:= \{p \in \mathcal{N}^0 \mid (u_S)_n = g \forall E \subset \Gamma_c \subset \mathcal{E}_p, \mathbf{b} \leq \mathbf{0} \text{ auf } \omega_p, \mathbf{j}_E \leq \mathbf{0} \forall E \in \mathcal{E}_p\}. \end{aligned}$$

Dann lässt sich der zweite Oszillationsterm beschreiben durch

$$\text{osc}_2(\mathbf{u}_S, g, \mathbf{b}) := \left(\sum_{p \in \mathcal{N}^{++}} h_p^2 \|\mathbf{b} - \bar{\mathbf{b}}_p\|_{0, \omega_p}^2 + \sum_{p \in \mathcal{N} \setminus (\mathcal{N}^{++} \cup \mathcal{N}^{0-})} h_p^2 \|\mathbf{b}\|_{0, \omega_p}^2 \right)^{\frac{1}{2}},$$

wobei $\|\mathbf{f}\|_{0, \omega_p}^2 = \int_{\omega_p} |\mathbf{f}|^2 d\Omega$ und $\bar{\mathbf{b}}_p$ analog zu \bar{f}_p der Mittelwert von \mathbf{b} über ω_p ist (komponentenweise integriert).

Zuverlässigkeit des Fehlerschätzers

Als letztes wollen wir uns noch mit der Übertragung der oberen Schranke auf den Fehlerschätzer $-\mathcal{I}_{\mathcal{Q}}(\varepsilon_V)$ beschäftigen, in der die hierfür definierten Oszillationen wieder eine Rolle spielen werden.

Zunächst lassen sich die Resultate für die lokalen Projektionen aus Lemma 4.19 und 4.20 vollständig übernehmen, da wir hier die komponentenweise Betrachtung der Basis aus dem Raum \mathcal{Q}_h verwenden können und somit für die Resultate der einzelnen Komponenten auf den neuen Raum \mathcal{Q}_h übertragen können.

Kapitel 5

Implementierung des Fehlerschätzers in Matlab

In diesem Kapitel wollen wir uns einen kurzen Überblick über den Quellcode für das programmierte Hindernis- bzw. Kontaktproblem verschaffen. Während wir an dieser Stelle einige wichtige Funktionen detailliert betrachten werden, können wir im Anhang D den Matlab-Quellcode komplett einsehen.

In die Implementierung sind neben [ZVKG11] unter anderem auch Resultate aus [MNS00], [BCH05], [Bra13] und [Ste12b] eingeflossen.

5.1 Implementierung eines Hindernisproblems

Grundlegender Aufbau des Programms

In der Datei `start_example*.m` werden die grundlegenden Daten für das betrachtete Beispiel festgelegt, wie beispielsweise

- die Geometriedaten für das Gitter und die Dirichlet-Randbedingungen,
- die exakte Lösung des Funktionals $J(u)$,
- die Lastfunktion f und
- die initiale Triangulierung \mathcal{T}_0 .

Weiter werden nach Ausführung des adaptiven Algorithmus die Galerkin-Lösung und ein Fehlerdiagramm geplottet.

Die Funktion `adaptive_refinement_solution.m` ist das Herzstück des Programms. Diese Datei beinhaltet den Ablauf des Algorithmus 4.1, wobei hier die Abbruchbedingung aus Zeile 5 erweitert wurde. Alle weiteren Programmteile werden in `adaptive_refinement_solution.m` aufgerufen.

Lokale Steifigkeitsmatrix und Assemblierung

Wie schon in Kapitel 2.3, Beispiel 2.26 beschrieben ist es für die Implementierung notwendig eine Verallgemeinerung zur Berechnung der Steifigkeitsmatrix herzuleiten. Die Idee ist an selber Stelle kurz vorgestellt worden; wir wollen nun angelehnt an Abbildung 2.5 die Formeln für die lokale Steifigkeitsmatrix eines beliebigen Elementes T über das Referenzelement

$$\tilde{T} = \{(\xi, \eta) \in \mathbb{R}^2 \mid 0 \leq \xi \leq 1, 0 \leq \eta \leq 1 - \xi\}$$

herleiten. Für die affine Transformation vom Referenzelement auf ein beliebiges Dreieck T gelten die Bedingungen

$$x = x_1 + (x_2 - x_1)\xi + (x_3 - x_1)\eta, \quad (5.1a)$$

$$y = y_1 + (y_2 - y_1)\xi + (y_3 - y_1)\eta, \quad (5.1b)$$

wobei $(x_i, y_i), i = 1, 2, 3$, die Eckpunkte des Dreiecks T sind (vgl. auch Abbildung 2.5). Mit den Bedingungen (5.1) erhalten wir dann die Funktionaldeterminante

$$\begin{aligned} J &= \det \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix} \\ &= (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1). \end{aligned} \quad (5.2)$$

Dabei gilt $J > 0$, da die Orientierung der Eckpunkte von \tilde{T} zu T erhalten bleibt. Wir bezeichnen mit $\tilde{u} : \tilde{T} \rightarrow \mathbb{R}$ eine Funktion über dem Referenzdreieck und mit $u : T \rightarrow \mathbb{R}$ die dazugehörige Funktion über dem allgemeinen Element T . Dann gilt zwischen \tilde{u} und u der Zusammenhang

$$\begin{aligned} u(x, y) &= u(x_1 + (x_2 - x_1)\xi + (x_3 - x_1)\eta, y_1 + (y_2 - y_1)\xi + (y_3 - y_1)\eta) \\ &= \tilde{u}(\xi, \eta). \end{aligned}$$

Wir rechnen also unter Verwendung der Kettenregel nach, dass

$$\begin{aligned} \begin{pmatrix} \tilde{u}_\xi \\ \tilde{u}_\eta \end{pmatrix} &= \begin{pmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{pmatrix} \begin{pmatrix} u_x \\ u_y \end{pmatrix} \\ \iff \begin{pmatrix} u_x \\ u_y \end{pmatrix} &= \frac{1}{J} \begin{pmatrix} y_3 - y_1 & y_1 - y_2 \\ x_1 - x_3 & x_2 - x_1 \end{pmatrix} \begin{pmatrix} \tilde{u}_\xi \\ \tilde{u}_\eta \end{pmatrix} \end{aligned} \quad (5.3)$$

gilt. Damit können wir $\nabla u \nabla v$ auf dem Referenzelement \tilde{T} ausdrücken durch

$$\begin{aligned} \begin{pmatrix} u_x \\ u_y \end{pmatrix}^T \begin{pmatrix} v_x \\ v_y \end{pmatrix} &= \frac{1}{J^2} \begin{pmatrix} \tilde{u}_\xi \\ \tilde{u}_\eta \end{pmatrix}^T \begin{pmatrix} y_3 - y_1 & x_1 - x_3 \\ y_1 - y_2 & x_2 - x_1 \end{pmatrix} \begin{pmatrix} y_3 - y_1 & y_1 - y_2 \\ x_1 - x_3 & x_2 - x_1 \end{pmatrix} \begin{pmatrix} \tilde{v}_\xi \\ \tilde{v}_\eta \end{pmatrix} \\ &= \frac{1}{J^2} \begin{pmatrix} \tilde{u}_\xi \\ \tilde{u}_\eta \end{pmatrix}^T \begin{pmatrix} a & b \\ b & c \end{pmatrix} \begin{pmatrix} \tilde{v}_\xi \\ \tilde{v}_\eta \end{pmatrix} \\ \text{mit } &\left\{ \begin{array}{l} a = (y_3 - y_1)^2 + (x_3 - x_1)^2 \\ b = -((y_3 - y_1)(y_2 - y_1) + (x_3 - x_1)(x_2 - x_1)) \\ c = (y_2 - y_1)^2 + (x_2 - x_1)^2 \end{array} \right. . \end{aligned}$$

Insgesamt können wir nun die lokale Bilinearform $a_T(u, v)$ von einem allgemeinen Element T auf das Referenzelement \tilde{T} transformieren.

$$\begin{aligned}
 a_T(u, v) &:= \int_T \nabla u \nabla v \, dx dy = \int_T u_x v_x + u_y v_y \, dx dy \\
 &= \int_{\tilde{T}} \frac{1}{J^2} (a \tilde{u}_\xi \tilde{v}_\xi + b (\tilde{u}_\xi \tilde{v}_\eta + \tilde{u}_\eta \tilde{v}_\xi) + c \tilde{u}_\eta \tilde{v}_\eta) J \, d\xi d\eta \\
 &= \frac{1}{J} \int_{\tilde{T}} a \tilde{u}_\xi \tilde{v}_\xi + b (\tilde{u}_\xi \tilde{v}_\eta + \tilde{u}_\eta \tilde{v}_\xi) + c \tilde{u}_\eta \tilde{v}_\eta \, d\xi d\eta \\
 &= \frac{1}{J} (a S_1 + b S_2 + c S_3)
 \end{aligned} \tag{5.4}$$

Die Matrizen $S_k, k = 1, 2, 3$, beinhalten dann die Anteile der einzelnen Summanden des Integranden aus dem oberen Integral von den jeweiligen lokalen Ansatzfunktionen. Eine Basis der linearen Ansatzfunktionen ist auf dem Referenzelement \tilde{T} beispielsweise von der Form

$$\varphi_1(\xi, \eta) = 1 - \xi - \eta, \quad \varphi_2(\xi, \eta) = \xi, \quad \varphi_3(\xi, \eta) = \eta. \tag{5.5}$$

Damit lassen sich die Einträge von $S_1 =: S = (s_{ij})_{i,j=1,2,3}$ berechnen durch

$$s_{ij} = \int_{\tilde{T}} \varphi_{i,\xi} \varphi_{j,\xi} \, d\xi d\eta, \tag{5.6}$$

d.h. mit (5.5) berechnen wir die Gradienten

$$\nabla \varphi_1(\xi, \eta) = (-1, -1), \quad \nabla \varphi_2(\xi, \eta) = (1, 0), \quad \nabla \varphi_3(\xi, \eta) = (0, 1)$$

und damit ergibt sich beispielsweise

$$s_{11} = \int_0^1 \int_0^{1-\xi} \varphi_{1,\xi} \varphi_{1,\xi} \, d\eta d\xi = \int_0^1 \int_0^{1-\xi} d\eta d\xi = \frac{1}{2}.$$

Analog lassen sich mit (5.6) die weiteren Matrixeinträge aus S_1 berechnen bzw. mit (5.4) und den dazugehörigen Formeln auch S_2 und S_3 :

$$S_1 = \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} & 0 \\ -\frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad S_2 = \begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}, \quad S_3 = \begin{pmatrix} \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & 0 & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}.$$

Insgesamt ist dann mit (5.4) die lokale Steifigkeitsmatrix für die linearen Ansatzfunktionen eines beliebigen Elementes gegeben durch

$$S = \frac{1}{2J} \begin{pmatrix} a + 2b + c & -a - b & -b - c \\ -a - b & a & b \\ -b - c & b & c \end{pmatrix}.$$

Betrachten wir nun eine Basis von quadratischen Ansatzfunktionen auf \tilde{T} , d.h.

$$\varphi_4(\xi, \eta) = 4\xi(1 - \xi - \eta), \quad \varphi_5(\xi, \eta) = 4\xi\eta, \quad \varphi_6(\xi, \eta) = 4\eta(1 - \xi - \eta),$$

dann können wir auch für diese nach Berechnung der Gradienten mit (5.4), (5.6) und den analog resultierenden Formeln eine lokale Steifigkeitsmatrix \bar{S} aufstellen. Diese hat dann die Form

$$\bar{S} = \frac{4}{3J} \begin{pmatrix} a+b+c & -b-c & b \\ -b-c & a+b+c & -a-b \\ b & -a-b & a+b+c \end{pmatrix}.$$

Um das lokale Defektproblem (4.9) zu lösen, ist nicht nur das Aufstellen von Steifigkeitsmatrizen über quadratische Ansatzfunktionen notwendig, sondern auch das Berechnen der rechten Seite $\rho_S(v) = (f, v) - a(u_S, v)$. Auch hier können wir für zumindest einen Teil der Summe einen lokalen Vektor bestimmen. Analog zu (5.4) rechnen wir nach, dass

$$\begin{aligned} a_T(u_S, v) &= \int_T \nabla u_S \nabla v \, dx dy \\ &= \int_{\tilde{T}} (a \tilde{u}_{S,\xi} + b \tilde{u}_{S,\eta}) \tilde{v}_\xi + (b \tilde{u}_{S,\xi} + c \tilde{u}_{S,\eta}) \tilde{v}_\eta \, d\xi d\eta \\ &= \lambda \mathbf{w}_1 + \mu \mathbf{w}_2 \end{aligned} \tag{5.7}$$

gilt, wobei $\mathbf{w}_i, i = 1, 2$ Vektoren sind, deren Einträge gerade die lokalen Anteile an dem i -ten Summanden des Integranden bzgl. der quadratischen Ansatzfunktionen enthalten, analog zu den Matrizen S_k . Wenn wir also eine Lösung $u(x, y) = \alpha x + \beta y + \gamma$ auf T gegeben haben, der Gradient $\nabla u(x, y) = (\alpha, \beta)$ noch auf \tilde{T} zu transformieren. Mit der affine Transformation (5.1) lässt sich leicht nachrechnen, dass dann

$$\nabla \tilde{u}_S(\xi, \eta) = (\alpha(x_2 - x_1) + \beta(y_2 - y_1), \alpha(x_3 - x_1) + \beta(y_3 - y_1)) =: (\tilde{\alpha}, \tilde{\beta})$$

ist. Damit ist $\lambda = a\tilde{\alpha} + b\tilde{\beta}$ und $\mu = b\tilde{\alpha} + c\tilde{\beta}$. Die Werte α, β des Gradienten auf T lassen sich durch die gegebenen Funktionswerte von u_S an den Eckpunkten des Elementes T berechnen, da u_S auf T linear ist. Dies wird in der Datei `grad_u.m` verwendet. Die Vektoren $\mathbf{w}_1, \mathbf{w}_2$ ergeben sich nun aus

$$\mathbf{w}_1 = \int_0^1 \int_0^{1-\xi} \begin{pmatrix} \varphi_{4,\xi} \\ \varphi_{5,\xi} \\ \varphi_{6,\xi} \end{pmatrix} d\eta d\xi = \begin{pmatrix} 0 \\ \frac{2}{3} \\ -\frac{2}{3} \end{pmatrix}, \quad \mathbf{w}_2 = \begin{pmatrix} -\frac{2}{3} \\ \frac{2}{3} \\ 0 \end{pmatrix}.$$

Damit ist der lokale Vektor aus (5.7) bzgl. eines beliebigen Dreiecks T gegeben durch

$$\bar{\rho} = \lambda \begin{pmatrix} 0 \\ \frac{2}{3} \\ -\frac{2}{3} \end{pmatrix} + \mu \begin{pmatrix} -\frac{2}{3} \\ \frac{2}{3} \\ 0 \end{pmatrix} = \begin{pmatrix} -\frac{2}{3}(b\tilde{\alpha} + c\tilde{\beta}) \\ \frac{2}{3}(a\tilde{\alpha} + b(\tilde{\beta} + \tilde{\alpha}) + c\tilde{\beta}) \\ -\frac{2}{3}(a\tilde{\alpha} + b\tilde{\beta}) \end{pmatrix}.$$

Die hier hergeleiteten Formeln für die lokalen Steifigkeitsmatrizen bzw. Vektoren werden in `local_mat.m` verwendet und ermöglichen einerseits eine leichtere Implementierung, da wir nur über ein Element integrieren müssen, andererseits erzeugen sie aber auch einen schnelleren Programmcode, weil wir keine Integrationen mehr durchführen müssen, um die Steifigkeitsmatrizen aufzustellen.

Die globalen Steifigkeitsmatrizen erzeugen wir dann durch Assemblierung, d.h. wir berechnen für alle Elemente die lokalen Steifigkeitsmatrizen und ordnen mit dem global-local node ordering jedem globalen Knoten in einem Element den jeweils lokalen zu. Damit addieren wir in der globalen Steifigkeitsmatrix im zugehörigen Eintrag den jeweiligen lokalen Eintrag auf. Dieses Vorgehen wird im Programmcode `assemble.m` verwendet. Natürlich ist die Assemblierung für den globalen Vektor aus ρ_S analog.

Quadratur auf einem Dreieck

Das Skalarprodukt $(f, v)_T = \int_T f v dx$ muss allerdings auf den einzelnen Dreiecken T ausgewertet werden. Dies kann leider nur durch direkte Integration geschehen, weil die Last f auf den Elementen variabel sein kann und daher nicht allgemein auf das Referenzdreieck transformierbar ist, so dass wir einen lokalen Vektor erstellen können, der nur noch assembliert werden muss.

Für die zweidimensionale Integration sind in Matlab beispielsweise die Funktionen `integral2` und `quad2d` implementiert. Um jedoch die Laufzeit des Codes zu verbessern, werden wir direkt eine Gauß-Quadratur über dem Referenzdreieck \tilde{T} verwenden. Eine mögliche Wahl (mit dem dazugehörigen Exaktheitsgrad) ist beispielsweise in [Sch] wiederzufinden. Implementiert sind davon die *Seitenmitten-Regel* und die *7-Punkte-Formel*, wobei wir aufgrund der hohen Genauigkeit beim Aufrufen des Programmcodes vorwiegend letztere verwenden. Die lokalen Stützstellen (ξ_k, η_k) und Gewichte w_k für die 7-Punkte-Formel sind auch in [Bra13] Kapitel II, Tabelle 5 zu finden und lauten wie folgt (s. Abbildung 5.1 für die ungefähre Lage der Stützstellen).

k	ξ_k	η_k	w_k
1	1/3	1/3	9/80
2	$(6 + \sqrt{15})/21$	$(6 + \sqrt{15})/21$	$(155 + \sqrt{15})/2400$
3	$(9 - 2\sqrt{15})/21$	$(6 + \sqrt{15})/21$	$(155 + \sqrt{15})/2400$
4	$(6 + \sqrt{15})/21$	$(9 - 2\sqrt{15})/21$	$(155 + \sqrt{15})/2400$
5	$(6 - \sqrt{15})/21$	$(6 - \sqrt{15})/21$	$(155 - \sqrt{15})/2400$
6	$(9 + 2\sqrt{15})/21$	$(6 - \sqrt{15})/21$	$(155 - \sqrt{15})/2400$
7	$(6 - \sqrt{15})/21$	$(9 + 2\sqrt{15})/21$	$(155 - \sqrt{15})/2400$

Tabelle 5.1: Stützstellen (ξ_k, η_k) und Gewichte w_k für die Gauß-Quadratur über das Referenzdreieck \tilde{T}

Um nun das Integral über ein beliebiges Element T zu ziehen, benötigen wir noch die affine Transformation von T auf das Referenzelement \tilde{T} , welche schon weiter oben bei der Berechnung der Steifigkeitsmatrizen genauer beschrieben wurde. Dadurch folgt mit $f(x_1 + (x_2 - x_1)\xi + (x_3 - x_1)\eta, y_1 + (y_2 - y_1)\xi + (y_3 - y_1)\eta) = \tilde{f}(\xi, \eta)$ die Formel

$$\int_T f(x, y) dx dy = \int_{\tilde{T}} \tilde{f}(\xi, \eta) J d\xi d\eta \approx J \sum_{k=1}^7 w_k \tilde{f}(\xi_k, \eta_k). \quad (5.8)$$

Die benötigten Gewichte und Stützstellen für die Quadraturformel über einem Dreieck (5.8) werden durch `quad_tris.m` übergeben. Weiterhin übergibt diese Funktion auch die Funktionswerte der mitgegebenen Ansatzfunktionen. Die Integration kann dann durch Anwendung der Formel (5.8) geschehen.

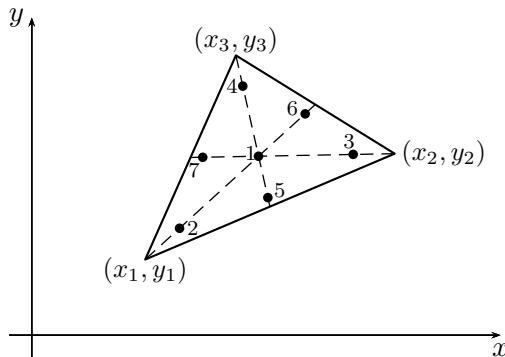


Abbildung 5.1: Ungefähr Lage der Stützstellen in einem allgemeinen Dreieck für die Gauß-Quadratur

Lösung der quadratischen Programme

Wie wir in Anhang B beschrieben können wir die konvexen quadratischen Programme, die sich durch Hindernis- oder Kontaktproblem erzeugen, mit dem Active-Set-Algorithmus lösen. Dieser ist unter anderem in der Matlab-Funktion `quadprog` hinterlegt. Allerdings ist der Active-Set-Algorithmus mit steigender Dimension schnell sehr langsam. Das Funktionen-Paket `quadprog` beinhaltet jedoch auch die *Innere-Punkte-Methode* (kurz: IPM), welche für große Systeme wesentlich schneller ist als die Active-Set-Methode. Daher verwenden wir die IPM zur Lösung der quadratischen Programme. Außerdem verwenden wir dazu *Sparse*-Matrizen und die Option *large-scale*, um die Auswertung zu beschleunigen.

Berechnung des Normalenflusses

Mit der Datei `normal_flux.m` berechnen wir für jede Kante aus der übergebenen Kantenmenge E_p den Normalenfluß j_E . Hierfür werden zunächst für jede Kante $E \in E_p$ die Indizes der anliegenden Dreiecke berechnet (dies geschieht mit der Funktion `neighbourhood.m`). Dabei gibt der Parameter `flag` an, ob eine Kante nur einen oder zwei Nachbarn besitzt, d.h. ob Randkante ist oder innen liegt. Durch die Indizes der anliegenden Dreiecke lassen sich die Eckpunkte und damit die Gradienten von u_S auf T_1 und T_2 sowie der Normalenvektor \mathbf{n} , der immer von T_1 nach T_2 zeigen soll, berechnen. Da diese Eigenschaft durch die gewählte Erzeugung des Normalenvektors im Code nicht immer erfüllt sein muss, wird überprüft, ob der berechnete Vektor \mathbf{n} mit einer der übrigen Kanten aus T_1 einen Winkel $\alpha \in (0, \frac{\pi}{2})$ einschließt. Sollte dies der Fall sein, so drehen wir den Vektor \mathbf{n} um, ansonsten beschreibt er schon die gewünschte Richtung. Danach werden die durch das Skalarprodukt $\nabla u_S|_{T_i} \cdot \mathbf{n}$ berechneten Richtungsableitungen noch voneinander abgezogen, um j_E zu erhalten.

Berechnung der einzelnen Knotenmengen

Um die Knotenmengen \mathcal{N}^0 und \mathcal{N}^+ zu berechnen, müssen wir zunächst die inneren Knoten $\mathcal{N} \cap \Omega$ bestimmen. Dies kann man sehr leicht durch die Adjazenzmatrix $H \in \{0, 1\}^{m \times n}$, die mit der Funktion `assemb` berechnet werden kann und durch den Matrixeintrag 1 angibt welcher der n Knoten einer der m Randknoten ist, bestimmen. Dafür berechnen wir eine Vergleichsmatrix, die überprüft, welcher der Einträge in H gleich Null ist, und bilden dann spaltenweise das Produkt. Die Einträge, die in dem erzeugten Vektor, ungleich Null sind, können keine Randpunkte sein.

Die inneren Kontakt- und Nichtkontaktknoten lassen sich dann leicht durch Überprüfen der Bedingungen berechnen, bzw. die Nichtkontaktknoten lassen sich auch einfach als Komplement zu den Kontaktknoten bilden. Bei der Berechnung der Kontaktknotenindizes ist zu beachten, dass wegen der Approximationsfehler eine obere Fehlerschranke $\varepsilon \approx 0$ verwendet wird. Dieses Vorgehen ist in `N0.m` und `Nplus.m` verwendet.

Analog kann man auch die Menge der Knoten \mathcal{N}^{++} (s. `Nplusplus.m`) leicht implementieren, indem wir einfach die Bedingung aus der Menge mit den mitgegebenen Daten überprüfen.

Dies ist für der Menge der isolierten Kontaktknoten \mathcal{N}^{0+} und der Kontaktknoten mit vollständigem Kontakt \mathcal{N}^{0-} nicht mehr so einfach möglich. Die nun beschriebenen Ideen sind gemeinsam in der Datei `N0plusminus.m` angewendet. Da wir in Matlab keine kompletten Funktionen miteinander vergleichen können (s. Bedingung $u_S = \psi$ auf ganz ω_p), transformieren wir die Dreiecke T wieder auf das Referenzdreieck und vergleichen einzelne Funktionswerte bzgl. eines möglichst engmaschigem Gitter auf diesem. Hierbei

laufen wir in einer Schleife über alle Dreiecke $T \subset \omega_p$ und berechnen auf jedem Dreieck $u_S - \psi$ auf dem betrachteten Gitter. Um nun zu überprüfen, ob p ein isolierter Kontaktknoten ist, müssen wir aus den berechneten Funktionswerten die der Eckpunkte eliminieren, was in Zeile 52-60 geschieht. Sollte nun ein Funktionswert kleiner oder gleich Null sein, so kann p kein isolierter Kontaktknoten mehr sein und wir setzen daher `flag_plus=1`, womit wir später diese Entscheidung fällen, wenn wir aus der Schleife über alle Dreiecke T kommen. Sollten nun aber alle Funktionswerte an den Eckpunkten von ψ und u_S übereinstimmen, so gilt wegen der Linearität der beiden Funktionen die Gleichheit und weiter $f \leq 0$ an den betrachteten Gitterpunkten, so erhöhen wir den Parameter `flag_minus` um Eins, mit dem wir damit später entscheiden können, ob die Bedingung auch auf allen Dreiecken von ω_p gilt. Es fehlt also nur noch die Überprüfung, ob `flag_plus = 0` ($\Rightarrow p \in \mathcal{N}^{0+}$) bzw. `flag_minus = |omega_p|` und $j_E \leq 0$ für alle $E \in \mathcal{E}_p$ ($\Rightarrow p \in \mathcal{N}^{0-}$) ist.

Oszillationsterme

Die Berechnung der Oszillationsterme geschieht in `osc1.m` und `osc2.m`. Die größte Schwierigkeit zur Implementierung dieser, liegt in der Bestimmung der einzelnen Punktemengen, die oben beschrieben wurde. Daher werden hier größtenteils nur die Integrale aus $\text{osc}_1(u_S, \psi)$ und $\text{osc}_2(u_S, \psi, f)$ eingearbeitet.

Bestimmung der lokalen Anteile des Fehlerindikators $\rho_S(\varepsilon_V)$

In der Datei `eval_rho_p.m` werden die lokalen Anteile $\rho_p(\varepsilon_V) = \rho_S(\varepsilon_V \phi_p)$ berechnet. Dabei werden die einzelnen Integrale mit der oben beschriebenen Quadratur berechnet. Es bleibt nur noch anzumerken, dass die Kurvenintegrale

$$\int_E j_E \varepsilon_V(x_E) \phi_E \phi_p ds = j_E \varepsilon_V(x_E) \int_E \phi_E \phi_p ds$$

über alle Kanten $E \in \mathcal{E}$ mittels eindimensionaler bestimmt werden können, wobei wir auch hier das Integral auf das Referenzintervall $[-1, 1]$ transformieren. Die Quadratur selbst ist danach leicht durchzuführen, weil ϕ_p auf einer Kante E identisch zu einer ansteigenden oder abfallenden Gerade und ϕ_E gleich einer nach unten geöffneten Parabel ist, die auf dem Anfangs- und Endpunkt ihre Nullstellen hat. Daher lassen sich die Werte

$$\int_{-1}^1 \phi_E(x(\xi), y(\xi)) \phi_p(x(\xi), y(\xi)) d\xi$$

unabhängig von E bestimmen und müssen später nur noch mit der Funktionaldeterminante

$$J = \frac{|E|}{2}$$

multipliziert werden.

Berechnung des zu verfeinernden Dreiecksindizes

Im Quellcode `find_triangle_refinement.m` werden die Indizes der zu verfeinernden Dreiecke berechnet. Dies geschieht dadurch, dass wir so viele der größten Anteile vom Fehlerindikator ρ_S heraussuchen, bis jene größer als ein skalierter Wert des Indikators ist. Dasselbe Vorgehen verwenden wir dann noch für die Oszillationsterme. Für die damit erhaltenen Punktindizes berechnen wir mit `neighbourhood` die anliegenden Dreiecke und übergeben diese für die Verfeinerung.

Sollten wir ein symmetrisches Problem vorliegen haben, dann ist es von Vorteil dieses auch symmetrisch bzgl. der Netzverfeinerung bzw. der Lösung zu halten. In diesem Fall sind auch die Fehleranteile im hierarchischen Fehlerschätzer $-\mathcal{I}_Q(\varepsilon_V)$ bzw. Fehlerindikator ρ_S symmetrisch angeordnet. Hierfür können wir den Wert `option` auf '`'symmetric'`' setzen. Dies bewirkt, dass bei der Auswahl eines Fehleranteiles alle weiteren Punkte mit demselben Anteil direkt mit gewählt werden. Damit kann es zu einer stärkeren Netzverfeinerung kommen, das erzeugte Gitter bleibt jedoch symmetrisch und damit auch die dadurch berechnete Galerkin-Lösung.

5.2 Implementierung eines Kontaktproblems

Für die Untersuchung eines Kontaktproblems ist das *Hertz'sche Kontaktproblem* implementiert worden. Hierbei handelt es sich um einen Zylinder, der mit einer Oberflächenlast \mathbf{t} belastet auf eine Ebene gedrückt wird.

Grundlegende Änderungen in der Implementierung bzgl. des Hindernisproblems

Der Aufbau der Startdatei bleibt grundlegend gleich; es ändern sich nur die Übergabe der Lastfunktionen (Volumenlast \mathbf{b} und Flächenlast \mathbf{t}) und es müssen neue Parameter μ und λ , die Lamé-Konstanten, hinzugefügt werden. Diese werden als Vektor übergeben, damit wir für den Zylinder und die Ebene verschiedene Elastizitäten erhalten.

Da wir den Zylinder nur vertikal, d.h. normal zur Ebene, belasten, wird für die Gap-Funktion der Einfachheit halber nicht der Normalenvektor an Ω verwendet, sondern der Vektor $(0, \pm 1)^T$.

Da wir nun zwei verschiedene Lasten erhalten, was den Lastvektor der rechten Seite in der Variationsungleichung ändert, und auch eine andere Bilinearform $a(\cdot, \cdot)$ verwenden, sowie verschiedene Randbedingungen haben, müssen in folgenden Dateien Änderungen vorgenommen werden:

- `assemble.m`,
- `local_mat.m`,

- `eval_rho_p.m`,
- `normal_flux.m`,
- `adaptive_refinement_solution.m`.

Zu den Änderungen gibt es noch zwei neue Dateien: `quad_edge.m` und `boundary_disjunction.m`. Erstere arbeitet analog zu `quad_tri.m` und gibt einfach die Stützstellen und Gewichte für die Gauß-Quadratur über eine Kante mit den dazu gegebenen Eckpunkten wieder zurück. Die Funktion `boundary_disjunction.m` berechnet für die übergebenen Randpunkte drei disjunkte Teilmengen von Randpunkten, die mit den Dirichlet-, Neumann und Kontaktrandpunkten übereinstimmen.

Lokale Steifigkeitsmatrizen und Assemblierung

Die Assemblierung der lokalen Matrizen bzw. Vektoren ist nur leicht verändert. Da wir in dem Kontaktproblem ein Verschiebungsfeld berechnen, werden die Matrizen und Vektoren doppelt so groß als im Hindernisproblem. Um weiterhin eine Struktur zu behalten, werden x- und y-Verschiebung blockweise in den Verschiebungsvektor hintereinander geschrieben. Dies führt dazu, dass bei der Assemblierung der Steifigkeitsmatrix A auch (2×2) -Matrixblöcke assembliert werden. Analog ist das Vorgehen für den Lastvektor blockweise durchzuführen. Als Änderung zum Hindernisproblem erhalten wir hier jedoch auch eine Last, die auf bestimmten Randpunkten (Neumann-Rand) existiert. Diese wird mithilfe von `quad_edge.m` durch direkte Integration über die jeweiligen Randkanten in den Lastvektor in die passenden Komponenten eingepflegt.

Die Berechnung der lokalen Steifigkeitsmatrizen wird analog zu (5.4) hergeleitet. Wir rechnen nach, dass bzgl. des Referenzdreiecks \tilde{T} gilt:

$$\begin{aligned}
 a_T(\mathbf{u}, \mathbf{v}) &= \int_T \boldsymbol{\varepsilon}(\mathbf{u}) : \mathcal{C} : \boldsymbol{\varepsilon}(\mathbf{v}) \, dx dy \\
 &= \frac{1}{J} \int_{\tilde{T}} a \tilde{u}_{1,1} \tilde{v}_{1,1} + b (\tilde{u}_{1,2} \tilde{v}_{1,1} + \tilde{u}_{1,1} \tilde{v}_{1,2}) + c (\tilde{u}_{2,1} \tilde{v}_{2,2} + \tilde{u}_{2,2} \tilde{v}_{2,1}) \\
 &\quad + d (\tilde{u}_{1,1} \tilde{v}_{2,2} + \tilde{u}_{2,2} \tilde{v}_{1,1}) + e (\tilde{u}_{1,1} \tilde{v}_{2,1} + \tilde{u}_{2,1} \tilde{v}_{1,1}) + f \tilde{u}_{1,2} \tilde{v}_{1,2} \\
 &\quad + g \tilde{u}_{2,1} \tilde{v}_{2,1} + h (\tilde{u}_{1,2} \tilde{v}_{2,1} + \tilde{u}_{2,1} \tilde{v}_{1,2}) + i (\tilde{u}_{1,2} \tilde{v}_{2,2} + \tilde{u}_{2,2} \tilde{v}_{1,2}) \\
 &\quad + j \tilde{u}_{2,2} \tilde{v}_{2,2} \, d\xi d\eta, \tag{5.9}
 \end{aligned}$$

wobei für beide Komponenten von \mathbf{u} und \mathbf{v} jeweils die Kettenregel (5.3) angewendet angewendet wurde. Wir lesen $(\cdot)_{i,j}$ so, dass die i -te Komponente von (\cdot) nach der j -ten Variable abgeleitet wird. Die Koeffizienten aus (5.9)

berechnen sich als

$$\begin{aligned}
 a &= \mu(x_3 - x_1)^2 + (2\mu + \lambda)(y_3 - y_1)^2, \\
 b &= -(\mu(x_3 - x_1)(x_2 - x_1) + (2\mu + \lambda)(y_3 - y_1)(y_2 - y_1)), \\
 c &= -(\mu(y_3 - y_1)(y_2 - y_1) + (2\mu + \lambda)(x_3 - x_1)(x_2 - x_1)), \\
 d &= \mu(x_3 - x_1)(y_2 - y_1) + \lambda(y_3 - y_1)(x_2 - x_1), \\
 e &= -(\mu + \lambda)(x_3 - x_1)(y_3 - y_1), \\
 f &= \mu(x_2 - x_1)^2 + (2\mu + \lambda)(y_2 - y_1)^2, \\
 g &= \mu(y_3 - y_1)^2 + (2\mu + \lambda)(x_3 - x_1)^2, \\
 h &= \mu(x_2 - x_1)(y_3 - y_1) + \lambda(x_3 - x_1)(y_2 - y_1), \\
 i &= -(\mu + \lambda)(x_2 - x_1)(y_2 - y_1), \\
 j &= \mu(y_2 - y_1)^2 + (2\mu + \lambda)(x_2 - x_1)^2.
 \end{aligned}$$

Als Basis für \mathcal{S}_h über dem Referenzelement \tilde{T} verwenden wir

$$\begin{aligned}
 \psi_1(\xi, \eta) &= \begin{pmatrix} 1 - \xi - \eta \\ 0 \end{pmatrix}, & \psi_2(\xi, \eta) &= \begin{pmatrix} 0 \\ 1 - \xi - \eta \end{pmatrix}, & \psi_3(\xi, \eta) &= \begin{pmatrix} \xi \\ 0 \end{pmatrix}, \\
 \psi_4(\xi, \eta) &= \begin{pmatrix} 0 \\ \xi \end{pmatrix}, & \psi_5(\xi, \eta) &= \begin{pmatrix} \eta \\ 0 \end{pmatrix}, & \psi_6(\xi, \eta) &= \begin{pmatrix} 0 \\ \eta \end{pmatrix}.
 \end{aligned}$$

Mit dieser Basis und den übrigen Resultaten lässt sich aus (5.9) die lokale Steifigkeitsmatrix mit umfangreicher Rechnung zu

$$S = \frac{1}{2J} \begin{pmatrix} a + 2b & d + e + h + i & -a - b & -e - h & -b - f & -d - i \\ d + e + h + i & 2c + j + g & -d - e & -c & -h - i & -c - j \\ -a - b & -d - e & a & e & b & d \\ -e - h & -c & e & g & h & c + d \\ -b - f & -h - i & b & h & f & i \\ -d - i & -c - j & d & c + d & i & j \end{pmatrix}$$

bestimmen. Analog zu der linearen Basis können wir auch eine Basis aus quadratischen Bubble-Funktionen erstellen. Da die Berechnung der lokalen Steifigkeitsmatrix in diesem Fall sehr aufwendig ist und wir für die Bilinearform a_Q nur die Hauptdiagonale der Matrix benötigen, bestimmen wir diese analog zu S als

$$\bar{S} = \frac{4}{3J} \operatorname{diag}(a + b + f, c + g + j, a + b + f, c + g + j, a + b + f, c + g + j).$$

Wie im eindimensionalen Fall können wir auch hier wieder den lokalen Anteil von $a(\mathbf{u}_S, \mathbf{v})$ für ein Dreieck T berechnen. Wir berechnen auch hier unter

Verwendung der Kettenregel (5.3), dass auf einem Dreieck T

$$\begin{aligned} a_T(\mathbf{u}_S, \mathbf{v}) &= \frac{1}{J} \int_{\tilde{T}} (a \tilde{u}_{S_{1,1}} + b \tilde{u}_{S_{1,2}} + e \tilde{u}_{S_{2,1}} + d \tilde{u}_{S_{2,2}}) \tilde{v}_{1,1} \\ &\quad + (b \tilde{u}_{S_{1,1}} + f \tilde{u}_{S_{1,2}} + h \tilde{u}_{S_{2,1}} + i \tilde{u}_{S_{2,2}}) \tilde{v}_{1,2} \\ &\quad + (e \tilde{u}_{S_{1,1}} + h \tilde{u}_{S_{1,2}} + g \tilde{u}_{S_{2,1}} + c \tilde{u}_{S_{2,2}}) \tilde{v}_{2,1} \\ &\quad + (d \tilde{u}_{S_{1,1}} + i \tilde{u}_{S_{1,2}} + c \tilde{u}_{S_{2,1}} + j \tilde{u}_{S_{2,2}}) \tilde{v}_{2,2} d\xi d\eta \\ &= \alpha \bar{\mathbf{w}}_1 + \beta \bar{\mathbf{w}}_2 + \gamma \bar{\mathbf{w}}_3 + \delta \bar{\mathbf{w}}_4 \end{aligned}$$

gilt. Durch Einsetzen der Basisfunktionen in $a_T(\mathbf{u}_S, \mathbf{v})$ ergeben sich analog zu $\mathbf{w}_1, \mathbf{w}_2$ die einzelnen lokalen Anteile von ρ_S mit

$$\bar{\mathbf{w}}_1 = \begin{pmatrix} 0 \\ 0 \\ \frac{2}{3} \\ 0 \\ -\frac{2}{3} \\ 0 \end{pmatrix}, \quad \bar{\mathbf{w}}_2 = \begin{pmatrix} -\frac{2}{3} \\ 0 \\ \frac{2}{3} \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \bar{\mathbf{w}}_3 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{2}{3} \\ 0 \\ -\frac{2}{3} \end{pmatrix}, \quad \bar{\mathbf{w}}_4 = \begin{pmatrix} 0 \\ -\frac{2}{3} \\ 0 \\ \frac{2}{3} \\ 0 \\ 0 \end{pmatrix}.$$

Berechnung der lokalen Anteile von ρ_S

Für die Berechnung der lokalen Anteile von $\rho_S(\boldsymbol{\varepsilon}_V)$ verwenden wir die Gleichung (4.91). Diese ist analog zum Hindernisproblem definiert, nur dass in den Integralen Skalarprodukte, also mehrere Summanden, vorhanden sind über die integriert werden müssen. Der grundlegende Aufbau des Quellcodes bleibt also gleich.

In Punkten, die auf dem Neumann-Rand liegen, müssen wir noch die Anteile aus dem Randintegral im Vektor der lokalen Anteile von $\rho_S(\boldsymbol{\varepsilon}_V)$ hinzufügen.

Berechnung des Normalenspannungsflußes

Der Normalenspannungsfluß \mathbf{j}_E , der zwischen den Gleichungen (4.89) und (4.90) definiert ist, ist analog zum Normalenfluss definiert. Da allerdings hier die Spannung für \mathbf{u}_S berechnet werden muss, werden der Funktion `normal_flux.m` zusätzlich die Lamé-Konstanten λ und μ übergeben. Diese fließen im Quellcode in die Beziehung $\boldsymbol{\sigma} = \mathcal{C} : \boldsymbol{\varepsilon} = 2\mu\boldsymbol{\varepsilon} + \lambda(\text{tr } \boldsymbol{\varepsilon})\mathbf{I}$ ein. Der sonstige Aufbau der Datei bleibt gleich und wir nutzen aus, dass wir mit `gradu.m` die Gradienten für die erste und zweite Komponente des Verschiebungsfeldes \mathbf{u} getrennt berechnen können.

Besonderheiten in `adaptive_refinement_solution.m`

Auch hier muss der grundlegende Aufbau der Quellcodes nicht verändert werden, allerdings müssen ein paar Zusätze eingefügt werden. Diese beziehen sich vor allem auf die Tatsache, dass die Nebenbedingung bzgl. des

Hindernisses nicht mehr auf ganz Ω gilt, sondern noch auf einem Teil des Randes, dem Kontaktrand Γ_c . Daher berechnen wir uns zunächst die Indizes für die Randpunkte, die auf dem Kontaktrand liegen und bestimmen alle notwendigen Größen zur Lösung der Variationsungleichung bzgl. dieser Punkte. Also wird das Hindernis (hier die Gap-Funktion) beispielsweise nur an den Kontaktrandpunkten ausgewertet. Weiter müssen wir für diese Punkte auch die Matrix B aus Kapitel 3.2.3 aufstellen. Da wir der Einfachheit halber für \mathbf{n} die Vektoren $(0, \pm 1)^T$ verwendet haben (je nach Lage des Knotens), erhalten wir für B eine $(m \times n)$ -Matrix ($m = |\mathcal{N}_{\Gamma_c}|, n = |\mathcal{W}|$), die in jeder Zeile nur einen Eintrag ungleich Null, nämlich gleich ± 1 hat. Dieser befindet sich gerade in der $2k$ -ten Spalte, wenn der dazugehörige Randpunkt den Index k hat. (Analogen Vorgehen ist für die Nebenbedingung des lokalen Defektproblems notwendig.)

Erweiterung der vorhandenen Implementierung

In der Implementierung fehlt die Verwendung der Oszillationsterme. Wenn diese noch eingearbeitet werden, kann man analog zum Algorithmus 4.1 den Schritt 9 auf das Kontaktproblem übertragen und damit erreichen, dass sich mit Sicherheit in jedem Verfeinerungsschritt die Oszillationen verringern.

Kapitel 6

Validierung

In diesem Kapitel wollen wir numerische Beispiele für das Hindernis- bzw. Kontaktproblem diskutieren. Dabei werden Auswertungen, die aus dem Quellcode aus Anhang D resultieren, benutzt.

Mögliche numerische Beispiele wurden aus den Veröffentlichungen [SV07], [BCH07], [BCH09] und [CSW99] entnommen.

6.1 Numerische Beispiele zum Hindernisproblem

Beispiel 6.1 (glatte, rotationssymmetrische Lösung). Wir betrachten das Hindernisproblem mit dem Hindernis $\psi \equiv 0$ und der Lastfunktion $f \equiv -2$ auf $\Omega = [-\frac{3}{2}, \frac{3}{2}]^2$. Die Dirichlet-Randbedingungen werden durch die Funktion

$$\tilde{u}(x, y) = \frac{x^2 + y^2}{2} - \frac{1}{2} \ln(x^2 + y^2) - \frac{1}{2}$$

gegeben. Für dieses Problem existiert eine exakte (glatte) Lösung, die in Polarkoordinaten wie folgt lautet:

$$u(r, \varphi) = \begin{cases} \frac{r^2}{2} - \ln(r) - \frac{1}{2} & , r \geq 1 \\ 0 & , \text{ sonst} \end{cases}. \quad (6.1)$$

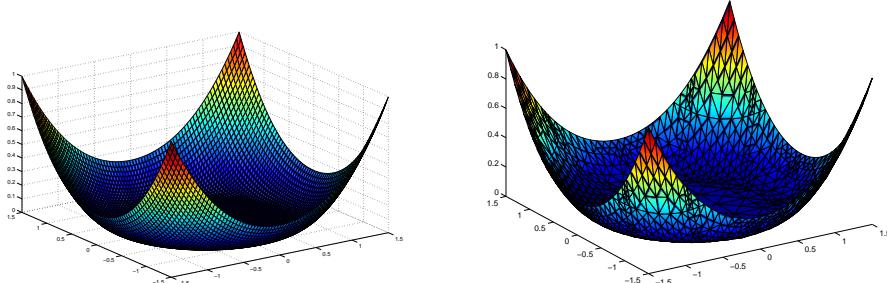
Damit lässt sich der exakte Wert des Energiefunktionalen $J(u)$ mit (6.1) ermitteln. Hierfür nutzen wir die Rotationssymmetrie von u aus, da u unabhängig vom Drehwinkel φ ist. Dadurch reicht es aus, die zu berechnenden Integrale nur im ersten Quadranten von Ω zu berechnen. Um jetzt $J(u) = \frac{1}{2}a(u, u) - (f, u)$ zu bestimmen, müssen wir mit der Kettenregel den Zusammenhang zwischen u_x, u_y und u_r, u_φ ermitteln. Dieser ergibt sich zu

$$\begin{pmatrix} u_x \\ u_y \end{pmatrix} = \frac{1}{r} \begin{pmatrix} r \cos \varphi & -\sin \varphi \\ r \sin \varphi & \cos \varphi \end{pmatrix} \begin{pmatrix} u_r \\ u_\varphi \end{pmatrix}. \quad (6.2)$$

Da $u_\varphi = 0$ und $u_r = r - \frac{1}{r}$ gilt, erhalten wir für den exakten Wert des Energiefunktionalen unter Verwendung von (6.2)

$$\begin{aligned}
 J(u) &= \frac{1}{2} \int_{\Omega} \underbrace{\nabla u \nabla u}_{=u_x^2+u_y^2} dx dy - \int_{\Omega} f u dx dy \\
 &= 4 \left(\frac{1}{2} \int_0^{\frac{\pi}{4}} \int_1^{\frac{3}{2 \cos \varphi}} (\cos^2 \varphi + \sin^2 \varphi) \left(r - \frac{1}{r}\right)^2 r dr d\varphi \right. \\
 &\quad + \frac{1}{2} \int_{\frac{\pi}{4}}^{\frac{3\pi}{4}} \int_1^{\frac{3}{2 \sin \varphi}} (\cos^2 \varphi + \sin^2 \varphi) \left(r - \frac{1}{r}\right)^2 r dr d\varphi \\
 &\quad + 2 \int_0^{\frac{\pi}{4}} \int_1^{\frac{3}{2 \cos \varphi}} \left(\frac{r^2}{2} - \ln(r) - \frac{1}{2}\right) r dr d\varphi \\
 &\quad \left. + 2 \int_{\frac{\pi}{4}}^{\frac{3\pi}{4}} \int_1^{\frac{3}{2 \sin \varphi}} \left(\frac{r^2}{2} - \ln(r) - \frac{1}{2}\right) r dr d\varphi \right) \\
 &\approx 3,980995748730258 .
 \end{aligned}$$

Die oberen Grenzen $\frac{3}{2 \cos \varphi}$ und $\frac{3}{2 \sin \varphi}$ kommen dabei durch Beschreibung der Randpunkte von Ω in Polarkoordinaten für $\varphi \in [0, \frac{\pi}{4}]$ zustande. Die Rotationssymmetrie kann auf das zweite Integral nur deshalb angewendet werden, weil f eine konstante (und damit auch rotationssymmetrische) Funktion ist.



(a) Exakte Lösung für das Hindernisproblem (b) Numerische Lösung in der Verfeinerungsstufe 6 mit $\theta_1 = 0,4$ und $\theta_2 = 0,3$

Abbildung 6.1: Exakte und numerische Lösung des Beispiels 6.1

In Abbildung 6.1 ist die exakte Lösung die Galerkin-Lösung auf der Verfeinerungsstufe 6 der implementierten, adaptiven Verfeinerungsstrategie im Vergleich dargestellt. In der Tabelle 6.1 sind die Ergebnisse für das Beispiel für $\theta_1 = 0,3$ und $\theta_2 = 0,2$, sowie für eine uniforme Verfeinerung dargestellt. Die maximale Anzahl der Knoten $|\mathcal{N}|$ ist jeweils mittels der Implementierung auf 150.000 begrenzt. Man kann gut erkennen, dass die Anzahl der Knoten

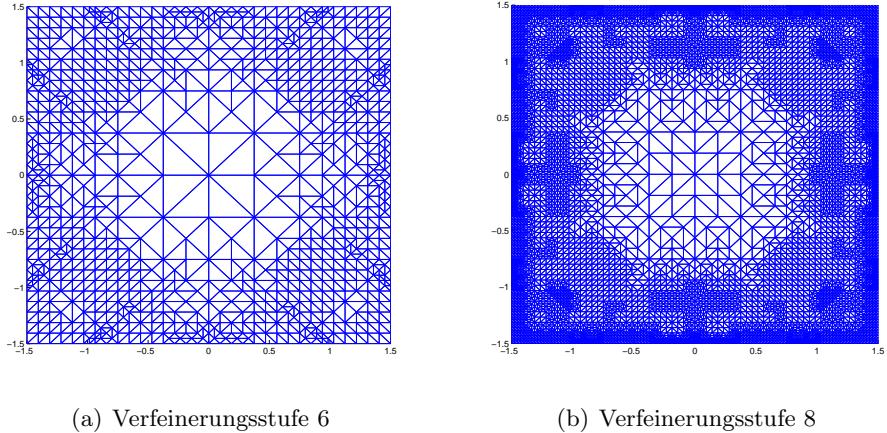


Abbildung 6.2: Gitterverfeinerung für das Hindernisproblem im Beispiel 6.1 mit $\theta_1 = 0,4$ und $\theta_2 = 0,3$

im uniformen Fall wesentlich schneller ansteigt. Da in den ersten Verfeinerungsstufen sehr wenig Knoten vorhanden sind, gleichen sich zunächst die adaptive und uniforme Verfeinerung. Der starke Anstieg des Freiheitsgrades in den letzten beiden Schritten liegt daran, dass die lokalen Anteile vom Fehlerindikator ρ_S sehr klein werden (wegen hoher Knotenanzahl) und könnte man durch eine variable Wahl der Parameter θ_1 und θ_2 verringern.

Weiter stellen wir in Abbildung 6.2 eine adaptive Gitterverfeinerung für das Beispiel 6.1, die mit `start_example_1.m` für $\theta_1 = 0,4$ und $\theta_2 = 0,3$ erzeugt werden können, in den Stufen 6 und 8 dar und können gut erkennen, dass im Bereich des vollen Kontaktes weniger stark verfeinert wird. Weiter kann man schön sehen, dass durch die Option '`'symmetric'`' das Gitter auch symmetrisch angeordnet bleibt von Verfeinerungsschritt zu Verfeinerungsschritt.

In Abbildung 6.3 sind als letztes auch noch die Oszillationen in einer Verfeinerungsstufe k dargestellt. Dabei kann man erkennen, dass es in der Stufe $k = 1$ noch einen isolierten Knoten geben muss, der jedoch durch Verfeinerung verschwindet, weil es keinen äquivalenten, isolierten Kontaktspunkt für die exakte Lösung gibt. Außerdem können wir sehen, dass durch die Bedingung aus Lemma 4.24 der Wert der Oszillationsterme von Verfeinerung zu Verfeinerung sinkt.

Stufe	Adaptive Verfeinerung				Uniforme Verfeinerung	
	$ \mathcal{N} $	$\rho_{\mathcal{S}}(\varepsilon_{\mathcal{V}})$	$-\mathcal{I}_{\mathcal{Q}}(\varepsilon_{\mathcal{V}})$	$-\mathcal{I}(e)$	$ \mathcal{N} $	$-\mathcal{I}(e)$
1	9	6.3512	3.1756	4.6323	9	4.6323
2	25	1.8284	0.9142	1.0978	25	1.0978
3	81	0.8642	0.4349	0.2667	81	0.2667
4	197	0.5422	0.2713	0.1113	289	0.0670
5	357	0.3896	0.1949	0.0564	1089	0.0167
6	821	0.2564	0.1282	0.0231	4225	0.0046
7	1133	0.1896	0.0948	0.0141	16641	0.0014
8	1957	0.0993	0.0497	0.0079	66049	0.0010
9	7569	0.0476	0.0238	0.0020		
10	29761	0.0223	0.0112	0.0018		
11	118017	0.0075	0.0037	0.0007		

Tabelle 6.1: Vergleich von adaptiver und uniformer Verfeinerung für Beispiel 6.1 mit Angabe des Freiheitsgrades, a posteriori Fehlerschätzer und exaktem Fehler

Beispiel 6.2 (Singularität und L-förmiges Gebiet). Wir wollen nun ein Hindernisproblem mit Singularität im Nullpunkt betrachten, was zu einer starken Verfeinerung an diesem Punkt führen muss. Hierfür sei das Gebiet L-förmig, also von der Form

$$\Omega := (-2, 2)^2 \setminus ([0, 2] \times [-2, 0]).$$

Das Hindernis sei wieder durch $\psi \equiv 0$ gegeben und die Lastfunktion in Polarkoordinaten (r, φ) durch

$$f(r, \varphi) := -r^{\frac{2}{3}} \sin\left(\frac{2\varphi}{3}\right) \left(\frac{\gamma_1'(r)}{r} + \gamma_1''(r)\right) - \frac{4}{3} r^{\frac{1}{3}} \gamma_1'(r) \sin\left(\frac{2\varphi}{3}\right) - \gamma_2(r),$$

wobei mit $\bar{r} = 2(r - \frac{1}{4})$ gilt

$$\begin{aligned} \gamma_1(r) &= \begin{cases} 1 & , \bar{r} < 0 \\ -6\bar{r}^5 + 15\bar{r}^4 - 10\bar{r}^3 + 1 & , 0 \leq \bar{r} < 1 \\ 0 & , \bar{r} \geq 1 \end{cases}, \\ \gamma_2(r) &= \begin{cases} 0 & , r < \frac{5}{4} \\ 1 & , \text{sonst} \end{cases}. \end{aligned}$$

Für dieses Problem gibt es eine exakte Lösung, die Polarkoordinaten

$$u(r, \varphi) = r^{\frac{2}{3}} \gamma_1(r) \sin\left(\frac{2\varphi}{3}\right)$$

lautet. Die Funktion hat im Ursprung eine Singularität. Damit können wir auch für dieses Beispiel mit den Umformungen aus (6.2) den exakten Wert

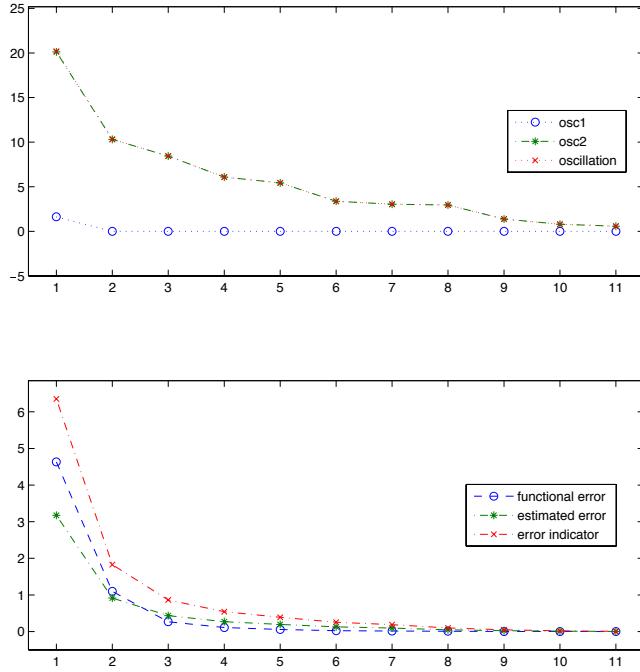


Abbildung 6.3: Oszillationsterme, Fehlerindikator ρ_S , Fehlerschätzer $-\mathcal{I}_Q$ und exakten Fehler für Beispiel 6.1 mit $\theta_1 = 0,3$ und $\theta_2 = 0,2$

des Energiefunktional J berechnen. Durch Anwendung der Polarkoordinaten und Transformationssatz der Integration erhalten wir

$$J(u) \approx 0,5.$$

Hier kommen noch die Auswertungen der Daten rein!

Beispiel 6.3. Als letztes Beispiel für ein affines Hindernisproblem können wir das Beispiel 6.2 noch erweitern, indem wir als affines Hindernis nicht die Nullfunktion, sondern eine umgedrehte Pyramide verwenden, die gegeben wird durch

$$\psi(x) := 0,5 (\text{dist}(x, \partial[-2, 2]^2) - 2,01), \quad x \in \bar{\Omega},$$

während alle übrigen Eingabedaten gleich bleiben. In Abbildung 6.6 können wir das Hindernis gut an der Ecke des Gebietes Ω erkennen, wenn wir diese mit der Lösung von Beispiel 6.2 aus Abbildung 6.4 vergleichen.

Hier kommen noch die Auswertungen der Daten rein!

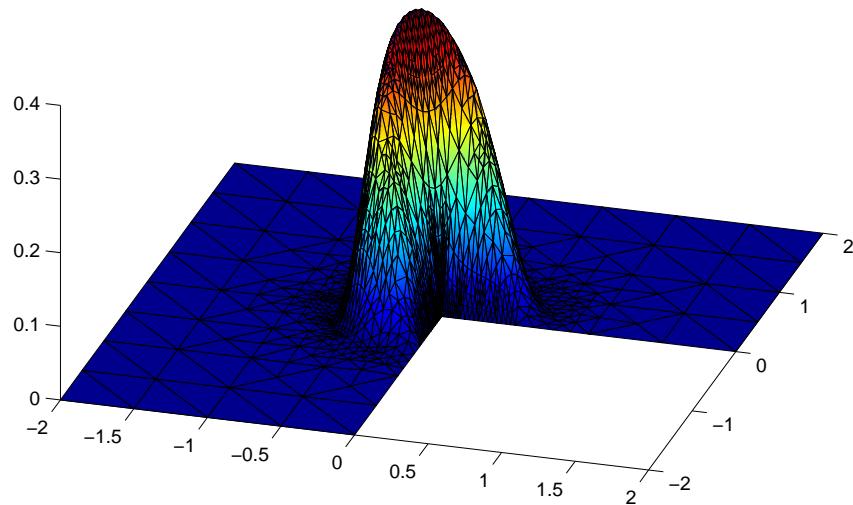


Abbildung 6.4: Numerische Lösung des Beispiels 6.2 in der Verfeinerungsstufe 7 mit $\theta_1 = \theta_2 = 0,3$

6.2 Numerisches Beispiel zum Kontaktproblem

meine Stichpunkte

- numerisches Beispiel (Problemstellung) → vielleicht mit Kontakt und nur Hindernis
- Vergleich mit Analytischer Lösung?! (Tabelle mit Ergebnissen) → Ergebnisse diskutieren

6. Validierung

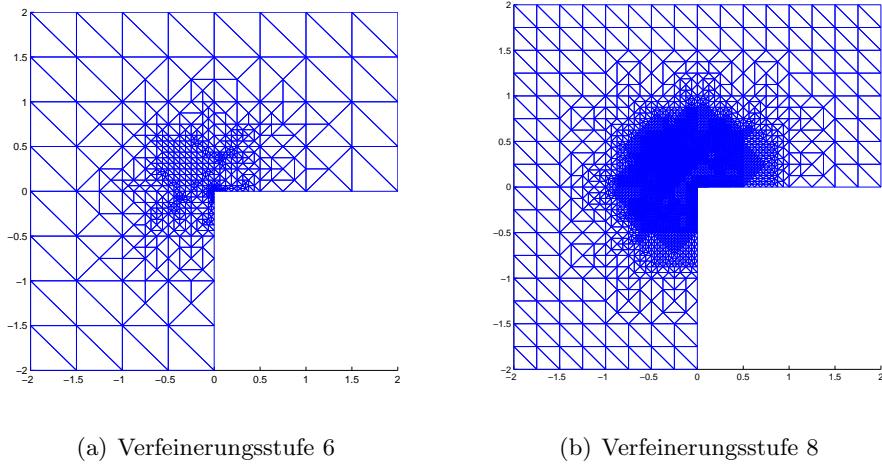


Abbildung 6.5: Netzverfeinerungen für das Beispiel 6.2 mit $\theta_1 = \theta_2 = 0,3$

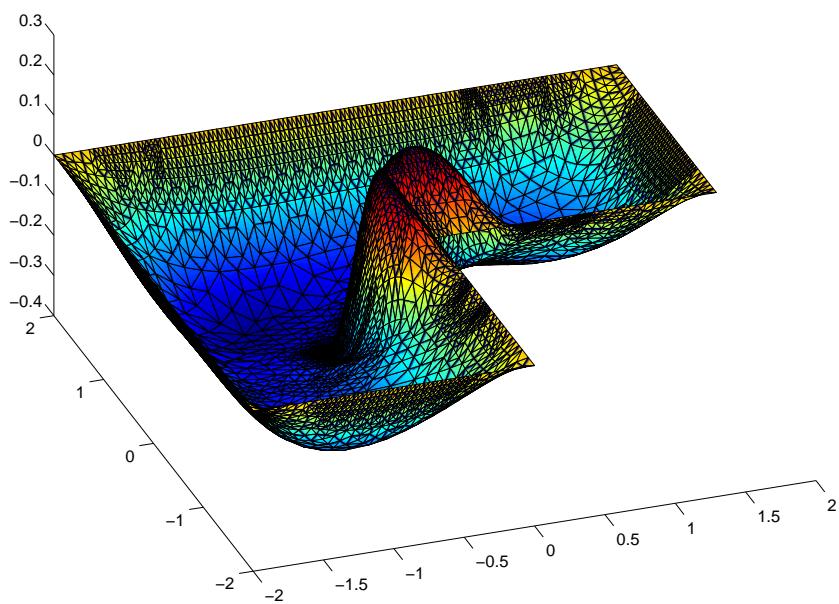
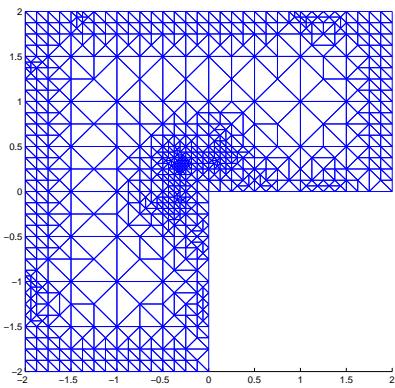
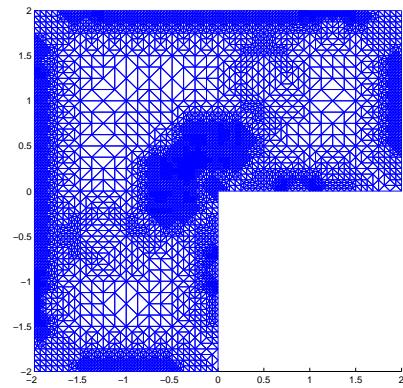


Abbildung 6.6: Numerische Lösung des Beispiels 6.3 mit $\theta_1 = \theta_2 = 0,3$



(a) Verfeinerungsstufe 6



(b) Verfeinerungsstufe 8

Abbildung 6.7: Netzverfeinerungen für das Beispiel 6.3 mit $\theta_1 = \theta_2 = 0,3$

Kapitel 7

Zusammenfassung und Ausblick

- kurz einleiten, worum es ging (Einleitung in einem Absatz zusammenfassen)
- Was ist rausgekommen?!
- Ausblick: Was ist noch offen geblieben, was kann man noch machen...
In dieser Arbeit linearisierte Verzerrung verwendet; kann verallgemeinert werden durch allgemeine Verzerrungstensoren (bzgl. der jeweiligen Konfiguration).
Vielleicht statt refinemesh ein Bissectionsverfahren implementieren ⇒ weniger Verfeinerung pro Schritt. Nachteil: der Regulatittsparameer bleibt in keinem Dreieck gleich

Literaturverzeichnis

- [Alt12] ALTBACH, Holm: *Kontinuumsmechanik*. 2. Auflage. Springer, 2012
- [BCH05] BARTELS, S. ; CARSTENSEN, C. ; HECHT, A.: 2D isoparametric FEM in MATLAB / Humboldt-Universität, Berlin. 2005. – Forschungsbericht
- [BCH07] BRAESS, D. ; CARSTENSEN, C. ; HOPPE, R.: Convergence analysis of a conforming adaptive finite element method for an obstacle problem. In: *Numerische Mathematik* 107 (2007), S. 455–471
- [BCH09] BRAESS, D. ; CARSTENSEN, C. ; HOPPE, R.: Error reduction in adaptive finite element approximation of elliptic obstacle problems. In: *Journal of Computational Mathematics* 27 (2009), Nr. 2-3, S. 148–169
- [Bra05] BRAESS, Dietrich: A Posteriori Error Estimators for Obstacle Problems – Another Look / Faculty of Mathematics, Ruhr-University. 2005. – Forschungsbericht
- [Bra13] BRAESS, Dietrich: *Finite Elemente – Theorie, schnelle Löser und Anwendungen in der Elastizitätstheorie*. 5. Auflage. Springer-Verlag, 2013
- [CSW99] CARSTENSEN, C. ; SCHERF, O. ; WRIGGERS, P.: Adaptive finite elements for elastic bodies in contact. In: *SIAM J. Sci. Comput.* 20 (1999), Nr. 5, S. 1605–1626
- [DLY89] DEUFLHARD, P. ; LEINEN, P. ; YSERENTANT, H.: Concepts of an Adaptive Hierarchical Finite Element Code. In: *Impact of Computing in Science and Engineering* 1 (1989), S. 3–35
- [Fal74] FALK, Richard S.: Error estimates for the approximation of a class of variational inequalities. In: *Math. Comp.* 28 (1974), S. 963–971
- [Fis05] FISCHER, Gerd: *Lineare Algebra*. 15. Auflage. 2005

- [Glo08] GLOWINSKI, Roland: *Numerical methods for nonlinear variational problems*. Reprint. Springer, 2008
- [GRT09] GÖPFERT, A. ; RIEDRICH, T. ; TAMMER, C.: *Angewandte Funktionalanalysis*. Vieweg und Teubner, 2009
- [HH80] HASLINGER, J. ; HLAVÁCEK, I.: Contact between elastic bodies. I. Continuous problems. In: *Apl. Mat.* 25 (1980), S. 324–347
- [HHNL80] HLAVÁCEK, I. ; HASLINGER, J. ; NECAS, J. ; LOVÍSEK, J.: *Solution of Variational Inequalities in Mechanics*. 9. Auflage. Springer, 1980
- [HK92] HOPPE, R. ; KORNHUBER, R.: Adaptive Multilevel-Methods for Obstacle Problems. In: *Preprint SC 91-16* (1992), April
- [Joh92] JOHNSON, Claes: Adaptive finite element methods for the obstacle problem. In: *Math. Models Methods Appl. Sci.* 2 (1992), Nr. 4, S. 483–487
- [KO88] KIKUCHI, N. ; ODEN, J.T.: *Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods*. SIAM, 1988
- [KZ11] KORNHUBER, Ralf ; ZOU, Qingsong: Efficient and reliable hierarchical error estimates for the discretization error of elliptic obstacle problems. In: *Mathematics of Computation* 80 (2011), Nr. 273, S. 69–88
- [Kö00] KÖNIGSBERGER, Konrad: *Analysis 2*. 5. Auflage. Springer, 2000
- [MNS00] MORIN, P. ; NOCHETTO, R.H. ; SIEBERT, K.G.: Data Oscillation and convergence of adaptive FEM. In: *SIAM J. Numer. Anal.* 38 (2000), Nr. 2, S. 466–488
- [NW06] NOCEDAL, Jorge ; WRIGHT, Stephen J.: *Numerical Optimization*. 2. ed. New York, NY : Springer, 2006
- [Pag10] PAGE, M.: Schätzerreduktion und Konvergenz adaptiver FEM für Hindernisprobleme / Universität Wien. 2010. – Masterarbeit
- [QSS02] QUARTERONI, A. ; SACCO, R. ; SALERI, F.: *Numerische Mathematik 2*. 1. Auflage. Springer, 2002
- [Rud91] RUDIN, Walter: *Functional Analysis*. 2. Auflage. McGraw-Hill, 1991
- [Sch] SCHNEIDER, Rene: *Beispiele für Gauß-Integration im Mehrdimensionalen*. – https://www-user.tu-chemnitz.de/~rens/lehre/archiv/numerik1_11SS/folien/gaussxd.pdf

- [Sta08] STARKE, Gerhard: Numerik partieller Differentialgleichungen / IFAM - Universität Hannover. 2008. – Vorlesungsskript
- [Sta11] STARKE, Gerhard: Variationsungleichungen / IFAM - Universität Hannover. 2011. – Vorlesungsskript
- [Ste12a] STEPHAN, Ernst P.: Contact Problems – Numerical Analysis and Implementation / IFAM - Universität Hannover. 2012. – Vorlesungsskript
- [Ste12b] STEPHAN, Ernst P.: Numerik partieller Differentialgleichungen I / IFAM - Universität Hannover. 2012. – Vorlesungsskript
- [Sto99] STOER, Josef: *Numerische Mathematik I*. 8. Auflage. Springer, 1999
- [SV07] SIEBERT, K.G. ; VEESER, A.: A Unilaterally Constrained Quadratic Minimization with Adaptive Finite Elements. In: *SIAM J. Optim.* 18 (2007), Nr. 1, S. 260–289
- [Wal11] WALKER, Christoph: Partielle Differentialgleichungen / IFAM - Universität Hannover. 2011. – Vorlesungsskript
- [Wer11] WERNER, Dirk: *Funktionalanalysis*. 7. Auflage. Springer, 2011
- [Wri01] WRIGGERS, Peter: *Nichtlineare Finite-Element-Methoden*. 5. Auflage. Springer, 2001
- [Wri06] WRIGGERS, Peter: *Computational Contact Mechanics*. 2. Auflage. Springer, 2006
- [Wri09] WRIGGERS, Peter: Finite-Elemente-Methode / IKM - Universität Hannover. 2009. – Vorlesungsskript
- [Zha07] ZHANG, Yongmin: Convergence of free boundaries in discrete obstacle problems. In: *Numerische Mathematik* (2007), Nr. 106, S. 157–164
- [Zou11] ZOU, Qingsong: Efficient and reliable hierarchical error estimates for an elliptic obstacle problem. In: *Applied Numerical Mathematics* 61 (2011), S. 344–355
- [ZVKG11] ZOU, Q. ; VEESER, A. ; KORNHUBER, R. ; GRÄSER, C.: Hierarchical error estimates for the energy functional in obstacle problems. In: *Numerische Mathematik* (2011), Nr. 117, S. 653–677

Anhang A

Funktionalanalysis

A.1 Sobolev-Räume

Sei im Weiteren $\emptyset \neq \Omega \subset \mathbb{R}^n$. Wir definieren den Sobolev-Raum allgemein wie folgt (vgl. [Bra13] Kapitel II, §2 und [Wal11] Kapitel 6).

Definition A.1. Seien $1 \leq p \leq \infty$ und $m \in \mathbb{N}$. Die Menge

$$W_p^m(\Omega) := \left(\{u \in L_p(\Omega) \mid \partial^\alpha u \in L_p(\Omega) \forall |\alpha| \leq m\}, \|\cdot\|_{W_p^m} \right)$$

heißt *Sobolev-Raum* der Ordnung m . Dabei ist

$$\|u\|_{W_p^m} := \|u\|_{W_p^m(\Omega)} := \left(\sum_{|\alpha| \leq m} \|\partial^\alpha u\|_{L_p}^p \right)^{\frac{1}{p}},$$

wenn $1 \leq p < \infty$. Im Fall $p = \infty$ ist $\|u\|_{W_p^m} := \max_{|\alpha| \leq m} \|\partial^\alpha u\|_\infty$.

Weiterhin bezeichne $L_p(\Omega)$ den *Lebesgue-Raum*, d.h. den Raum der messbaren Funktionen, deren p -te Potenz Lebesgue-integrierbar über Ω ist, d.h.

$$L_p(\Omega) := \left(\{u : \Omega \rightarrow \mathbb{R} \mid f \text{ messbar}, \|\cdot\|_{L_p} < \infty\}, \|\cdot\|_{L_p} \right),$$

wobei $\|u\|_{L_p} := \|u\|_{L_p(\Omega)} = \|u\|_{W_p^0}$.

Definition A.2. Der Raum

$$\mathcal{D}(\Omega) := C_c^\infty(\Omega) = \{\varphi \in C^\infty(\Omega) \mid \text{supp}(\varphi) \subset \subset \Omega\}$$

heißt der *Raum der Testfunktionen*, wobei $K \subset \subset \Omega \Leftrightarrow \bar{K} \subset \Omega$ kompakt.

Bemerkung A.3. Seien $u \in W_p^m(\Omega)$, $\varphi \in \mathcal{D}(\Omega)$ und $\alpha \in \mathbb{N}^n$ mit $|\alpha| \leq m$. Dann bezeichnen wir $v = \partial^\alpha u$ als *schwache Ableitung* von u , wenn gilt

$$\int_{\Omega} v \cdot \varphi \, dx = (-1)^{|\alpha|} \int_{\Omega} u \cdot \partial^\alpha \varphi \, dx.$$

Beispiel A.4. Es sei $\Omega = (-1, 1) \subset \mathbb{R}$ und $u(x) = |x| \in L_2(\Omega)$. Betrachten wir $v(x) = \text{sign}(x)$, so ergibt sich für $\varphi \in \mathcal{D}(\Omega)$

$$\begin{aligned}\int_{\Omega} v \cdot \varphi \, dx &= \int_{-1}^0 -1 \cdot \varphi(x) \, dx + \int_0^1 1 \cdot \varphi(x) \, dx \\ &= -x\varphi(x) \Big|_{-1}^0 - \int_{-1}^0 -x\varphi'(x) \, dx + x\varphi(x) \Big|_0^1 - \int_0^1 x\varphi'(x) \, dx \\ &= - \int_{-1}^1 |x|\varphi'(x) \, dx = (-1)^1 \int_{\Omega} u \cdot \varphi' \, dx,\end{aligned}$$

da $\varphi(-1) = \varphi(1) = 0$. Also ist $v = \partial u$ und somit $u \in W_2^1(\Omega)$. Analog kann man nachrechnen, dass

$$\int_{\Omega} v \cdot \varphi' \, dx = -2\varphi(0)$$

ist und somit u nicht zweimal schwach ableitbar ist, d.h. $u \notin W_2^2(\Omega)$.

Wir wollen in der Theorie der Finiten Elemente Methode vor allem Sobolev-Räume über dem Raum $L_2(\Omega)$ betrachten, daher ist folgender Satz essentiell.

Satz A.5. Es seien $1 \leq p \leq \infty$ und $m \in \mathbb{N}$. Dann gilt:

- (a) $W_p^m(\Omega)$ ist ein Banachraum.
- (b) $H^m(\Omega) := W_2^m(\Omega)$ ist ein Hilbertraum mit Skalarprodukt

$$(u, v)_m := (u, v)_{H^m(\Omega)} := \sum_{|\alpha| \leq m} (\partial^\alpha u, \partial^\alpha v)_0 \quad \forall u, v \in H^m(\Omega),$$

wobei

$$(u, v)_0 := (u, v)_{L_2(\Omega)} := \int_{\Omega} uv \, dx.$$

Bemerkung A.6. (a) Die Norm auf $H^m(\Omega)$ ergibt sich analog zur Norm des allgemeinen Sobolev-Raumes durch das Skalarprodukt, d.h. $\|u\|_m := \|u\|_{H^m(\Omega)} := \|u\|_{W_2^m}$.

(b) Analog dazu definieren wir die Halbnorm $|\cdot|_m$ auf H^m wie folgt:

$$|u|_m := |u|_{H^m(\Omega)} := \left(\sum_{|\alpha|=m} \|\partial^\alpha u\|_{L_2}^2 \right)^{\frac{1}{2}}.$$

Definition A.7. Der Raum $H_0^m(\Omega)$ ist die Vervollständigung von $\mathcal{D}(\Omega)$ bzgl. der Norm $\|\cdot\|_m$.

Bemerkung A.8. Die Funktionen $u \in H_0^m(\Omega)$ können als die Funktionen $u \in H^m(\Omega)$ mit $u = 0$ auf $\partial\Omega$ aufgefasst werden. Weiter ist $H_0^m(\Omega)$ ein abgeschlossener Unterraum von $H^m(\Omega)$ (vgl. auch [Wal11] Bemerkung 6.7).

Theorem A.9 (Spursatz). *Es sei $\Omega \subset \mathbb{R}^n$ und C^2 . Dann gilt für $1 < p < \infty$, dass ein $c > 0$ existiert mit*

$$\|u|_{\partial\Omega}\|_{L_p(\partial\Omega)} \leq c \|u\|_{W_p^1(\Omega)}$$

für alle $u \in C^\infty(\bar{\Omega})$.

Die Restriktion $[u \mapsto u|_{\partial\Omega}]$ lässt sich also eindeutig stetig zum Spuroperator $\gamma_0 \in \mathcal{L}(W_p^1(\Omega), L_p(\partial\Omega))$ fortsetzen.

A.2 Optimalitätskriterien

Zunächst definieren wir einen verallgemeinerten Begriff der Richtungsableitung, der auch auf unendlich dimensionalen Vektorräumen existiert.

Definition A.10. Es seien V ein Vektorraum, $M \subset V$ und W ein normierter Raum, sowie $F : M \rightarrow W$ eine Abbildung, $x_0 \in M$ und $v \in V$. Dann heißt F Gâteaux-differenzierbar (bzw. in Richtung v an der Stelle x_0 differenzierbar), falls es ein $\varepsilon > 0$ mit $[x_0 - \varepsilon v, x_0 + \varepsilon v] \subset M$ gibt und der Grenzwert

$$\mathcal{D}_v F(x_0) := \frac{d}{dt} F(x_0 + tv) \Big|_{t=0} := \lim_{t \rightarrow 0} \frac{F(x_0 + tv) - F(x_0)}{t} \quad (\text{A.1})$$

in W existiert. $\mathcal{D}_v F(x_0)$ heißt dann Gâteaux-Ableitung von F an der Stelle x_0 in Richtung v .

Falls wir nur $[x_0, x_0 + \varepsilon v] \subset M$ voraussetzen, so können wir in (A.1) $\lim_{t \rightarrow 0}$ durch $\lim_{t \rightarrow +0}$ ersetzen. Dann nennen wir (A.1) die rechtsseitige Gâteaux-Ableitung und bezeichnen diese mit $\mathcal{D}_v^+ F(x_0)$.

Für die Variationsrechnung sind folgende zwei Sätze für uns von besonderer Bedeutung.

Satz A.11. (Charakterisierungssatz der konvexen Optimierung) *Es seien $M \subset V$ eine konvexe Menge, V ein Vektorraum und $F : M \rightarrow \mathbb{R}$ ein konvexes Funktional. Dann gilt für $x_0, x \in M$:*

x_0 ist Lösung von $\min_{x \in M} F(x)$ genau dann, wenn für alle $x \in M$ gilt

$$\mathcal{D}_{x-x_0}^+ F(x_0) \geq 0.$$

Beweis. Siehe [GRT09], Kapitel 3.3.3, Satz 3.34. □

Satz A.12. Es sei $U \subset V$ ein (Unter-)Vektorraum, V ein Vektorraum und $F : U \rightarrow \mathbb{R}$ eine Gâteaux-differenzierbare konvexe Funktion. Dann ist $x_0 \in U$ genau dann Lösung von $\min_{x \in U} F(x)$, wenn für alle $u \in U$ gilt

$$\mathcal{D}_u F(x_0) = 0.$$

Beweis. Siehe [GRT09], Kapitel 3.3.3, Satz 3.35. \square

A.3 Konvergenzbegriffe

Definition A.13. Es sei $m \in \mathbb{N}, 1 \leq p < \infty, 1 = \frac{1}{p} + \frac{1}{p'}$.

- (a) Eine Folge (u_j) in L_p konvergiert schwach gegen $u \in L_p(\Omega)$

$$\begin{aligned} &\iff u_j \rightharpoonup u \text{ in } L_p(\Omega) \\ &\iff \forall v \in L_{p'}(\Omega) : \int_{\Omega} u_j v \, dx \longrightarrow \int_{\Omega} u v \, dx \text{ in } \mathbb{K}. \end{aligned}$$

- (b) Eine Folge $(u_j) \in W_p^m(\Omega)$ konvergiert schwach gegen $u \in W_p^m(\Omega)$

$$\begin{aligned} &\iff u_j \rightharpoonup u \text{ in } W_p^m(\Omega) \\ &\iff \partial^\alpha u_j \rightharpoonup \partial^\alpha u \text{ in } L_p(\Omega) \forall |\alpha| \leq m. \end{aligned}$$

Bemerkung A.14. Sei $1 \leq p < \infty, m \in \mathbb{N}$, dann ist:

- (a) Ist $u_j \rightarrow u$ in $W_p^m(\Omega)$, dann folgt $u_j \rightharpoonup u$ in $W_p^m(\Omega)$, d.h. „starke Konvergenz ist stärker als schwache Konvergenz“.

Beweis. $\forall v \in L_{p'}(\Omega), |\alpha| \leq m$ gilt

$$\left| \int_{\Omega} (\partial^\alpha u_j - \partial^\alpha u) v \, dx \right| \underset{\text{Hölder}}{\leq} \|v\|_{L_{p'}(\Omega)} \|\partial^\alpha u_j - \partial^\alpha u\|_{L_p(\Omega)} \longrightarrow 0. \quad \square$$

- (b) Sei $1 < p < \infty, (u_j) \subset W_p^m(\Omega)$ beschränkt (bzgl. $\|\cdot\|_{W_p^m}$), dann folgt, dass eine Teilfolge $(u_{j'})$ und ein $u \in W_p^m(\Omega)$ existiert, so dass $u_{j'} \rightharpoonup u$ in $W_p^m(\Omega)$, d.h. „beschränkte Folgen sind relativ schwach kompakt“.

Beweis. Vgl. [Rud91]. \square

- (c) Es sei $M \subset W_p^m(\Omega)$ konvex und abgeschlossen (bzgl. $\|\cdot\|_{W_p^m}$), sowie $(u_j) \subset M$ mit $u_j \rightharpoonup u$ in $W_p^m(\Omega)$, dann ist $u \in M$, d.h. „abgeschlossene konvexe Mengen sind schwach abgeschlossen“ (Theorem von Mazur; ohne Beweis, vgl. [Rud91]).

- (d) Es sei $u_j \rightharpoonup u$ in $W_p^m(\Omega)$, dann folgt (u_j) ist beschränkt in $W_p^m(\Omega)$ (bzgl. $\|\cdot\|_{W_p^m}$), d.h. „schwach konvergente Folgen sind beschränkt“.

Beweis. Theorem von Mackey, vgl. [Rud91]. □

- (e) $u_j \rightharpoonup u$ in $W_p^m(\Omega)$, $u_j \rightharpoonup v$ in $W_p^m(\Omega)$, dann gilt $u = v$, d.h. „Grenzwerte von schwach konvergenten Folgen sind eindeutig“.

Beweis. Aus dem Hauptsatz der Variationsrechnung folgt die Behauptung. □

- (f) Sei $u_j \rightharpoonup u$ in $W_p^m(\Omega)$, dann folgt $\|u\|_{W_p^m(\Omega)} \leq \liminf \|u_j\|_{W_p^m(\Omega)}$.

Theorem A.15. *In einem reflexiven Raum V , d.h. der Bidualraum V'' ist isomorph zu V , besitzt jede beschränkte Folge $(v_n)_{n \in \mathbb{N}}$ eine schwach konvergente Teilfolge (v_{n_j}) .*

Beweis. Der Beweis befindet sich in [Wer11] Kapitel III, Theorem 3.7. □

Bemerkung A.16. Jeder Hilbertraum H ist reflexiv.

Beweis. Dies folgt aus dem Darstellungssatz von Riesz (Satz 2.15). □

Anhang B

Optimierung

B.1 Quadratische Programmierung

Um im folgenden die Idee des Algorithmus zu verstehen, führen wir zunächst grundlegende Begriffe ein. Ein quadratisches Problem mit Gleichungs- und Ungleichungsnebenbedingungen ist von der Form

$$\begin{aligned} \min_{\mathbf{x}} \quad & q(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T G \mathbf{x} + \mathbf{x}^T \mathbf{c} \\ \text{s.t.} \quad & \mathbf{a}_i^T \mathbf{x} = b_i, \quad i \in \mathcal{E}, \\ & \mathbf{a}_i^T \mathbf{x} \geq b_i, \quad i \in \mathcal{I}, \end{aligned} \tag{B.1}$$

wobei \mathcal{E} und \mathcal{I} die Indexmengen der Gleichungs- und Ungleichungsnebenbedingungen darstellen und $\mathbf{c}, \mathbf{x}, \mathbf{a}_i \in \mathbb{R}^n, b_i \in \mathbb{R}, i \in \mathcal{E} \cup \mathcal{I}$, sowie G eine symmetrische $(n \times n)$ -Matrix ist, welche die Hesse-Matrix des Problems darstellt. Damit ist die Hesse-Matrix konstant und daher das Problem konvex, wenn G positiv semidefinit ist. (Ist G positiv definit, so nennen wir das Problem strikt konvex. Wenn G indefinit ist, ist (B.1) „nicht konvex“.)

Da sonst das quadratische Problem (und damit der Active-Set Algorithmus) zu kompliziert wird, betrachten wir hier nur den konvexen Fall. Für diesen Fall können wir leicht zeigen, dass eine Lösung \mathbf{x}^* , die die Bedingungen 1. Ordnung erfüllt, auch globale Lösung des Problems ist (s. Theorem ??). Anschaulich kann es im indefiniten Fall mehrere optimale Punkte geben, die voneinander getrennt liegen, d.h. die Menge der optimalen Punkte ist nicht zusammenhängend, wodurch das Auffinden des globalen Minimums erschwert wird.

Die notwendigen Bedingungen 1. Ordnung sind die KKT-Bedingungen und können hier angewendet werden, da die Restriktionen und die Zielfunktion stetig differenzierbar sind. Die Lagrangefunktion \mathcal{L} für das quadratische Problem ist

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \frac{1}{2} \mathbf{x}^T G \mathbf{x} + \mathbf{x}^T \mathbf{c} - \sum_{i \in \mathcal{I} \cup \mathcal{E}} \lambda_i (\mathbf{a}_i^T \mathbf{x} - b_i). \tag{B.2}$$

Damit ergeben sich – vgl. [NW06], Theorem 12.1 – mit der Menge der aktiven Nebenbedingungen $\mathcal{A}(\mathbf{x}^*) = \{i \in \mathcal{E} \cup \mathcal{I} : \mathbf{a}_i^T \mathbf{x}^* = b_i\}$ die KKT-Bedingungen

$$\begin{aligned}\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) &= G\mathbf{x}^* + \mathbf{c} - \sum_{i \in \mathcal{A}(\mathbf{x}^*)} \lambda_i^* \mathbf{a}_i = 0, \\ \mathbf{a}_i^T \mathbf{x}^* &= b_i, \quad \forall i \in \mathcal{A}(\mathbf{x}^*), \\ \mathbf{a}_i^T \mathbf{x}^* &\geq b_i, \quad \forall i \in \mathcal{I} \setminus \mathcal{A}(\mathbf{x}^*), \\ \lambda_i^* &\geq 0, \quad \forall i \in \mathcal{I} \cap \mathcal{A}(\mathbf{x}^*).\end{aligned}\tag{B.3}$$

Hierbei ist \mathbf{x}^* Lösung von (B.1) und erfüllt die LICQ-Bedingung; $\boldsymbol{\lambda}^*$ ist dazugehöriger optimaler Lagrange-Multiplikator. In (B.3) wird die Komplementaritätsbedingung $\lambda_i^* c_i(\mathbf{x}^*) = 0$ impliziert durch $\lambda_i^* = 0 \forall i \notin \mathcal{A}(\mathbf{x}^*)$.

Theorem B.1. Wenn \mathbf{x}^* die Bedingungen (B.3) erfüllt mit $\lambda_i^*, i \in \mathcal{A}(\mathbf{x}^*)$ und G ist positiv semidefinit, dann ist \mathbf{x}^* eine globale Lösung von (B.1).

Beweis. Wenn \mathbf{x} ein beliebiger weiterer zulässiger Punkt für (1.1) ist, gelten die Restriktionen $\mathbf{a}_i^T \mathbf{x} = b_i, i \in \mathcal{E}$, sowie $\mathbf{a}_i^T \mathbf{x} \geq b_i, i \in \mathcal{I} \cap \mathcal{A}(\mathbf{x}^*)$ für \mathbf{x} und damit gilt zusammen mit der ersten Bedingung von (B.3), dass

$$(\mathbf{x} - \mathbf{x}^*)^T (G\mathbf{x}^* + \mathbf{c}) = \sum_{i \in \mathcal{E}} \underbrace{\lambda_i^* \mathbf{a}_i^T (\mathbf{x} - \mathbf{x}^*)}_{\geq 0} + \sum_{i \in \mathcal{A}(\mathbf{x}^*) \cap \mathcal{I}} \underbrace{\lambda_i^* \mathbf{a}_i^T (\mathbf{x} - \mathbf{x}^*)}_{\geq 0} \geq 0.$$

Dann drücken wir $q(\mathbf{x})$ durch $q(\mathbf{x}^*)$ aus und wenden die obere Ungleichung sowie die positive Semidefinitheit für G an, um zu zeigen, dass $q(\mathbf{x}) \geq q(\mathbf{x}^*)$ ist. Damit ist \mathbf{x}^* globale Lösung des quadratischen Problems. \square

Daher ist im positiv semidefiniten Fall gesichert, dass ein optimaler Punkt auch gleichzeitig globale Lösung ist.

B.2 Active Set-Methode für konvexe QPs

Wenn wir eine Lösung \mathbf{x}^* für das Problem (B.1) kennen, so ist auch die Menge der aktiven Nebenbedingungen $\mathcal{A}(\mathbf{x}^*)$ bekannt und wir können (B.1) vereinfachen zum Optimierungsproblem

$$\min_{\mathbf{x}} q(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T G \mathbf{x} + \mathbf{x}^T \mathbf{c}, \quad \text{s.t. } \mathbf{a}_i^T \mathbf{x} = b_i, \quad i \in \mathcal{A}(\mathbf{x}^*). \tag{B.4}$$

Dieses könnten wir dann beispielsweise mit direkten Verfahren wie der Schur-Komplement-Methode oder der Nullraum-Methode lösen. Natürlich ist die optimale Lösung zu Beginn noch nicht bekannt und damit auch nicht die aktiven Restriktionen. Jedoch können wir diese Idee für die Active-Set-Methode verwenden.

Das Hauptziel der Active-Set-Methode ist, die Menge der aktiven Restriktionen bzgl. der optimalen Lösung zu finden, wobei wir hier die primale

Variante betrachten wollen, in der die Approximierte \mathbf{x}_k zulässig bzgl. des primalen Problems ist.

Die Grundidee ist, ein quadratisches Teilproblem zu lösen, bei dem wir bestimmte Nebenbedingungen aus Problem (B.1) bzgl. \mathcal{I} als aktiv annehmen. Die dadurch beschriebene Indexmenge der aktiven Restriktionen für \mathbf{x}_k im k -ten Schritt heißt *working set* und kann wie folgt beschrieben werden

$$\mathcal{W}_k = \{i \mid \mathbf{a}_i^T \mathbf{x}_k = b_i, i \in \mathcal{E} \cup \mathcal{J}, \mathcal{J} \subset \mathcal{I}\}.$$

Hierbei muss vorausgesetzt werden, dass die Nebenbedingungen in \mathcal{W}_k die LICQ-Bedingung erfüllen, selbst wenn diese bezogen auf alle Nebenbedingungen an der Stelle x_k nicht erfüllt wird.

Wir betrachten nun den k -ten Schritt mit der Approximierten \mathbf{x}_k und dem working set \mathcal{W}_k . Wir berechnen die neue Iterierte \mathbf{x}_{k+1} , indem wir eine Richtung \mathbf{p} finden, in der wir unter den Nebenbedingungen \mathcal{W}_k die Funktion q minimieren. Hierfür betrachten wir $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}$ und setzen \mathbf{x}_{k+1} in q ein:

$$\begin{aligned} q(\mathbf{x}_{k+1}) &= q(\mathbf{x}_k + \mathbf{p}) = \frac{1}{2}(\mathbf{x}_k + \mathbf{p})^T G(\mathbf{x}_k + \mathbf{p}) + (\mathbf{x}_k + \mathbf{p})^T \mathbf{c} \\ &= \frac{1}{2} \mathbf{x}_k^T G \mathbf{x}_k + \underbrace{\mathbf{x}_k^T G \mathbf{p}}_{\text{da } G \text{ symm.}} + \frac{1}{2} \mathbf{p}^T G \mathbf{p} + \mathbf{x}_k^T \mathbf{c} + \mathbf{p}^T \mathbf{c} \\ &= \frac{1}{2} \mathbf{p}^T G \mathbf{p} + \mathbf{g}_k^T \mathbf{p} + \rho_k, \end{aligned}$$

wobei $\mathbf{g}_k = G\mathbf{x}_k + \mathbf{c}$ und $\rho_k = \frac{1}{2} \mathbf{x}_k^T G \mathbf{x}_k + \mathbf{x}_k^T \mathbf{c}$. Da wir den Parameter \mathbf{p} so wählen wollen, so dass $q(\mathbf{x}_{k+1})$ minimal wird, ist der Term ρ_k bzgl. des Problems konstant und kann somit für die Lösung jenes weggelassen werden. Da weiterhin auch \mathbf{x}_{k+1} die aktiven Nebenbedingungen \mathcal{W}_k erfüllen soll, gilt

$$\mathbf{a}_i^T \mathbf{p} = \mathbf{a}_i^T (\mathbf{x}_{k+1} - \mathbf{x}_k) = \underbrace{\mathbf{a}_i^T \mathbf{x}_{k+1}}_{=b_i} - \underbrace{\mathbf{a}_i^T \mathbf{x}_k}_{=b_i} = 0 \quad \forall i \in \mathcal{W}_k.$$

Zusammengefasst müssen wir also im k -ten Schritt das Teilproblem

$$\begin{aligned} \min_{\mathbf{p}} \quad & \frac{1}{2} \mathbf{p}^T G \mathbf{p} + \mathbf{g}_k^T \mathbf{p}, \\ \text{s.t.} \quad & \mathbf{a}_i^T \mathbf{p} = 0, \quad \forall i \in \mathcal{W}_k \end{aligned} \tag{B.5}$$

lösen. Die Lösung im k -ten Schritt von (B.5) bezeichnen wir mit \mathbf{p}_k . Umgekehrt gilt damit, analog zur obigen Rechnung, natürlich auch, dass für alle $i \in \mathcal{W}_k$ die Restriktion aktiv bleibt für $\mathbf{x}_k + \alpha \mathbf{p}_k$ mit beliebigem α . Da G positiv definit ist, kann (B.5) nun – wie schon bei (B.4) erwähnt – mit Schur-Komplement-Methode oder Nullraum-Methode gelöst werden.

Wie wir schon wissen, ist die neue Iterierte $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$ bzgl. \mathcal{W}_k immer noch zulässig. Nun müssen wir jedoch feststellen, ob diese Iterierte

auch alle übrigen Restriktionen mit $i \notin \mathcal{W}_k$ erfüllt. Ist dies der Fall, so setzen wir $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$, ansonsten suchen wir das größtmögliche $\alpha_k \in [0, 1]$, so dass

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

zulässig bleibt. Hierfür betrachten wir zwei Fälle.

Fall 1: Gilt für ein $i \notin \mathcal{W}_k$, dass $\mathbf{a}_i^T \mathbf{p}_k \geq 0$ ist, so folgt

$$\mathbf{a}_i^T (\mathbf{x}_k + \alpha_k \mathbf{p}_k) = \mathbf{a}_i^T \mathbf{x}_k + \underbrace{\alpha_k \mathbf{a}_i^T \mathbf{p}_k}_{\geq 0} \geq \mathbf{a}_i^T \mathbf{x}_k \geq b_i,$$

da $\alpha_k \geq 0$, d.h. für diese Nebenbedingungen müssen wir für die Wahl von α_k nichts beachten.

Fall 2: Existiert ein $i \notin \mathcal{W}_k$, für das $\mathbf{a}_i^T \mathbf{p}_k < 0$ ist, so gilt

$$\begin{aligned} & \mathbf{a}_i^T (\mathbf{x}_k + \alpha_k \mathbf{p}_k) \geq b_i \\ \iff & \mathbf{a}_i^T \mathbf{x}_k + \alpha_k \mathbf{a}_i^T \mathbf{p}_k \geq b_i \\ \iff & \alpha_k \underbrace{\mathbf{a}_i^T \mathbf{p}_k}_{< 0} \geq b_i - \mathbf{a}_i^T \mathbf{x}_k \\ \iff & \alpha_k \leq \frac{b_i - \mathbf{a}_i^T \mathbf{x}_k}{\mathbf{a}_i^T \mathbf{p}_k}. \end{aligned} \quad (\text{B.6})$$

Damit folgt mit (B.6) und den vorherigen Überlegungen, dass zusammengefasst

$$\alpha_k = \min \left\{ 1, \min_{i \notin \mathcal{W}_k, \mathbf{a}_i^T \mathbf{p}_k < 0} \frac{b_i - \mathbf{a}_i^T \mathbf{x}_k}{\mathbf{a}_i^T \mathbf{p}_k} \right\} \quad (\text{B.7})$$

gilt. Eine Restriktion $i \notin \mathcal{W}_k$, für die das Minimum für α_k angenommen wird, nennen wir *blocking constraint*; diese muss nicht eindeutig sein, da wir beispielsweise anschaulich auch von einer Ecke geblockt werden können. Ist $\alpha_k = 1$, so werden alle Restriktion außerhalb vom working set mit dem Schritt $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$ erfüllt, d.h. es gibt keine blocking constraint. Gibt es eine Nebenbedingung $j \notin \mathcal{W}_k$, die aktiv ist, obwohl sie nicht zum working set gehört, so gilt

$$\begin{aligned} \alpha_k &= \min \left\{ 1, \min_{i \notin \mathcal{W}_k, \mathbf{a}_i^T \mathbf{p}_k < 0} \frac{b_i - \mathbf{a}_i^T \mathbf{x}_k}{\mathbf{a}_i^T \mathbf{p}_k} \right\} \\ &= \min \left\{ 1, \frac{b_j - \mathbf{a}_j^T \mathbf{x}_k}{\mathbf{a}_j^T \mathbf{p}_k} \right\} \\ &= \min \left\{ 1, \frac{b_j - b_j}{\mathbf{a}_j^T \mathbf{p}_k} \right\} = 0. \end{aligned}$$

Es sei $j \notin \mathcal{W}_k$ nun ein Index einer blocking constraint. Dann ist

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k = \mathbf{x}_k + \frac{b_j - \mathbf{a}_j^T \mathbf{x}_k}{\mathbf{a}_j^T \mathbf{p}_k} \mathbf{p}_k.$$

Setzen wir \mathbf{x}_{k+1} in die j -te Restriktion ein, so erhalten wir

$$\begin{aligned} \mathbf{a}_j^T \mathbf{x}_{k+1} &= \mathbf{a}_j^T \left(\mathbf{x}_k + \frac{b_j - \mathbf{a}_j^T \mathbf{x}_k}{\mathbf{a}_j^T \mathbf{p}_k} \mathbf{p}_k \right) = \mathbf{a}_j^T \mathbf{x}_k + \frac{b_j - \mathbf{a}_j^T \mathbf{x}_k}{\mathbf{a}_j^T \mathbf{p}_k} \cdot \cancel{\mathbf{a}_j^T \mathbf{p}_k} \\ &= \mathbf{a}_j^T \mathbf{x}_k + b_j - \mathbf{a}_j^T \mathbf{x}_k = b_j, \end{aligned}$$

d.h. die blocking constraint ist für die neue Iterierte \mathbf{x}_{k+1} nach Konstruktion aktiv. Daher setzen wir als neues working set $\mathcal{W}_{k+1} = \mathcal{W}_k \cup \{j\}$.

Das oben beschriebene Vorgehen wiederholen wir so lange, bis wir das working set $\hat{\mathcal{W}}$ mit dem Minimum des quadratischen Problems $\hat{\mathbf{x}}$ gefunden haben. Dies ist leicht zu erkennen, da wir (B.1) auf \mathcal{W}_k nicht weiter minimieren können, sobald es keinen Schritt p gibt, in dessen Richtung wir q verringern können, d.h. wenn $\mathbf{p} = \mathbf{0}$ die Lösung für das Teilproblem (B.5) ist. Dann ist der optimale Punkt $\hat{\mathbf{x}}$ bzgl. des working sets $\hat{\mathcal{W}} \subset \mathcal{A}(\hat{\mathbf{x}})$ gefunden.

Wir müssen jetzt überprüfen, ob $\hat{\mathbf{x}}$ die KKT-Bedingungen erfüllt. Wir wissen, dass für $\mathbf{p} = \mathbf{0}$ die KKT-Bedingungen für (B.5)

$$\begin{pmatrix} G & A^T \\ A & 0 \end{pmatrix} \cdot \begin{pmatrix} -\mathbf{p} \\ \hat{\lambda} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{g}} \\ \hat{\mathbf{h}} \end{pmatrix}$$

mit $\hat{\mathbf{g}} = \mathbf{c} + G\hat{\mathbf{x}}$, $\hat{\mathbf{h}} = A\hat{\mathbf{x}} + \mathbf{b}$ und $\mathbf{p} = \mathbf{0}$ erfüllt. Daraus folgt

$$\begin{aligned} A^T \hat{\lambda} &= \hat{\mathbf{g}} \iff \sum_{i \in \hat{\mathcal{W}}} \mathbf{a}_i \hat{\lambda}_i = G\hat{\mathbf{x}} + \mathbf{c}, \\ \mathbf{0} &= \hat{\mathbf{h}} \iff A\hat{\mathbf{x}} = \mathbf{b}, \end{aligned}$$

wobei A die Gradienten \mathbf{a}_i^T der aktiven Restriktionen $\hat{\mathcal{W}}$ zeilenweise enthält. Damit werden die ersten beiden KKT-Bedingungen aus (B.3) erfüllt. Da die Schrittlänge α_k mit (B.6) so gewählt ist, dass die übrigen Restriktionen erfüllt bleiben, gilt auch die dritte Bedingung aus (B.3). Es bleibt zu überprüfen, ob die Lagrange-Multiplikatoren $\hat{\lambda}_i \geq 0$ sind.

Gilt $\hat{\lambda}_i \geq 0$ für alle $i \in \hat{\mathcal{W}} \cap \mathcal{I}$, so sind alle KKT-Bedingungen erfüllt und damit $\mathbf{x}^* = \hat{\mathbf{x}}$. Existiert allerdings ein $j \in \hat{\mathcal{W}} \cap \mathcal{I}$, so dass $\hat{\lambda}_j < 0$ ist, so können wir den Wert von q noch weiter verringern, indem wir die j -te Restriktion wegfällen lassen (vlg. [NW06], Kapitel 12.3). Dies zeigt das folgende Theorem.

Theorem B.2. Der Punkt $\hat{\mathbf{x}}$ erfülle die notwendigen Bedingungen 1. Ordnung für das Teilproblem (B.5) auf $\hat{\mathcal{W}}$. Weiter seien die Gradienten $\mathbf{a}_i, i \in$

$\hat{\mathcal{W}}$, linear unabhängig (LICQ) und es gebe einen Index $j \in \mathcal{W}$ mit $\hat{\lambda}_j < 0$. Es sei \mathbf{p} die Lösung vom Teilproblem (B.5) ohne die Restriktion j , d.h.

$$\begin{aligned} \min_{\mathbf{p}} \quad & \frac{1}{2} \mathbf{p}^T G \mathbf{p} + (G \hat{\mathbf{x}} + \mathbf{c})^T \mathbf{p}, \\ \text{s.t.} \quad & \mathbf{a}_i^T \mathbf{p} = 0, \quad \forall i \in \hat{\mathcal{W}} \setminus \{j\}. \end{aligned}$$

Dann ist \mathbf{p} eine zulässige Richtung für die Nebenbedingung j , d.h. $\mathbf{a}_j^T \mathbf{p} \geq 0$. Weiterhin gilt sogar $\mathbf{a}_j^T \mathbf{p} > 0$ und \mathbf{p} ist eine Abstiegsrichtung für q , wenn \mathbf{p} die hinreichenden Bedingungen 2. Ordnung erfüllt.

Da wir zeigen können, dass der erzielte Abstieg für q durch das Weglassen einer Nebenbedingung mit negativem Lagrange-Multiplikator λ_i proportional zu $|\lambda_i|$ ist, eliminieren wir gerade die Restriktion mit kleinstem Langrange-Multiplikator. Es kann allerdings sein, dass der folgende zu berechnende Schritt \mathbf{p} aufgrund einer blocking constraint kurz ist, wodurch nicht garantiert ist, dass q den größtmöglichen Abstieg erfährt.

B.3 Algorithmus

Algorithm B.1 Active-Set-Methode für konvexe quadratische Probleme
Gegeben sei ein zulässiger Startpunkt \mathbf{x}_0 für (B.1) und definiere \mathcal{W}_0 z.B. mit allen aktiven Restriktionen bzgl. \mathbf{x}_0 .

```

for  $k = 0, 1, 2, \dots$  do
    Löse (B.5) zur Berechnung von  $\mathbf{p}_k$ ;
    if  $\mathbf{p}_k = \mathbf{0}$  then
        Berechne die Lagrange-Multiplikatoren mittels (2.5a)
        und setze  $\hat{\mathcal{W}} = \mathcal{W}_k$ ;
        if  $\hat{\lambda}_i \geq 0 \forall i \in \hat{\mathcal{W}} \cap \mathcal{I}$  then
            stop mit der Lösung  $\mathbf{x}^* = \hat{\mathbf{x}}$ ;
        else
             $j \leftarrow \arg \min_{j \in \mathcal{W}_k \cap \mathcal{I}} \hat{\lambda}_j$ ;
             $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k$ ,  $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{j\}$ ;
        end if
    else ( $\mathbf{p}_k \neq \mathbf{0}$ )
        Berechne  $\alpha_k$  mit (B.7);
         $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{p}_k$ ;
        if  $\alpha_k < 1$  (blocking constraint existiert) then
            Bestimme blocking constraint  $j$  und setze  $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \cup \{j\}$ 
        else
             $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k$ 
        end if
    end if
end for

```

Anhang C

Tensorrechnung

C.1 Tensoralgebra

Definition C.1. Es sei V ein endlicher Vektorraum und \mathbb{K} ein beliebiger Körper. Ein p -Tensor (oder auch p -stufiger Tensor) $\varphi : V \times \dots \times V \rightarrow \mathbb{K}$ ist eine multilinear Abbildung, d.h. linear in jeder Komponente. Die Menge aller p -Tensoren über V wird mit $T^p(V)$ bezeichnet.

Beispiel C.2. Der Dualraum V^* eines Vektorraums V ist der Raum der Linearformen, also der Raum der 1-Tensoren. Also können wir schreiben $V^* = T^1(V)$.

Bemerkung C.3. Da V endlich ist, können wir $T^p(V)$ durch eine endliche Anzahl von Basisvektoren $\varphi_k \in T^p(V)$ aufspannen. Solche Basisvektoren lassen sich durch die Basis des Dualraumes V^* und der folgenden Definition bilden.

Definition C.4 (Tensorprodukt). Ist $\varphi \in T^p(V)$ und $\psi \in T^q(V)$, so definieren wir das *Tensorprodukt* (oder auch *dyadische Produkt*) $\varphi \otimes \psi \in T^{p+q}(V)$ durch

$$(\varphi \otimes \psi)(v_1, \dots, v_p, v_{p+1}, \dots, v_{p+q}) = \varphi(v_1, \dots, v_p) \cdot \psi(v_{p+1}, \dots, v_{p+q}).$$

Bemerkung. Wir sollten beachten, dass $\varphi \otimes \psi$ nicht zwangsläufig kommutativ ist.

Bemerkung C.5. Da $T^1(V)$ der Raum der Linear- und $T^2(V)$ der Raum der Bilinearformen ist, können wir die Elemente als Vektoren bzw. Matrizen identifizieren. Hierbei ist allerdings zu beachten, dass diese nur die Koordinaten bzgl. der jeweils betrachteten Basen angeben.

Wir notieren einstufige Tensoren, d.h. Elemente aus $T^1(V)$, in fetten Kleinbuchstaben, also $\mathbf{a}, \mathbf{b}, \mathbf{c}$ usw., und schreiben diese bzgl. der Standardbasis

$$\mathbf{a} = a_i \mathbf{e}_i := \sum_{i=1}^n a_i \mathbf{e}_i.$$

Hierbei ist die *Einstein'sche Summenkonvention* benutzt worden, in der über doppelt vorkommende Indizes summiert wird.

Analog beschreiben wir zweistufige Tensoren, also Elemente aus $T^2(V)$, mit fetten Großbuchstaben, d.h. $\mathbf{A}, \mathbf{B}, \mathbf{C}$ etc., und schreiben diese bzgl. der Standardbasis als Tensorprodukt von zwei einstufigen Tensoren:

$$\mathbf{A} = A_{ij} \mathbf{e}_i \otimes \mathbf{e}_j := \sum_{i,j=1}^n A_{ij} \mathbf{e}_i \otimes \mathbf{e}_j .$$

Bemerkung. Die Tensordarstellung aus Bemerkung C.5 kann natürlich auch bzgl. allgemeiner Basen geschehen.

Definition C.6 (Kroneckerdelta, Einheitstensor). Das *Kroneckerdelta* ist definiert durch

$$\delta_{ij} := \begin{cases} 1, & i = j \\ 0, & \text{sonst} \end{cases} .$$

Damit können wir den zweistufigen *Einheitstensor* definieren durch

$$\mathbf{1} := \delta_{ij} \mathbf{e}_i \otimes \mathbf{e}_j .$$

Bemerkung C.7. Wir können leicht nachrechnen, dass das Kroneckerdelta folgende Eigenschaften hat:

$$\delta_{ij} \delta_{jk} = \delta_{ik} , \quad \delta_{ij} \delta_{jk} \delta_{kl} = \delta_{il} .$$

Mit diesen Eigenschaften können wir beispielsweise 4- oder 6-stufige Einheitstensoren/Einheitstensor definieren.

Definition C.8 (Skalarprodukte). Das *einfach verjüngende Skalarprodukt* ist für zwei Vektoren (1-Tensoren) \mathbf{a}, \mathbf{b} definiert als

$$\mathbf{a} \cdot \mathbf{b} := (a_i \mathbf{e}_i) \cdot (b_j \mathbf{e}_j) = a_i b_j \underbrace{\mathbf{e}_i \cdot \mathbf{e}_j}_{=\delta_{ij}} = a_i b_i .$$

Das *doppelt verjüngende Skalarprodukt* ist für zwei Matrizen (2-Tensoren) \mathbf{T}, \mathbf{S} definiert als

$$\mathbf{T} : \mathbf{S} = (T_{ij} \mathbf{e}_i \otimes \mathbf{e}_j) : (S_{kl} \mathbf{e}_k \otimes \mathbf{e}_l) := T_{ij} S_{kl} \delta_{ik} \cdot \delta_{jl} = T_{ij} S_{ij} .$$

Bemerkung C.9. Analog können wir die Skalarprodukte aus Definition C.8 auch auf höher stufige Tensoren übertragen. Die Skalarprodukte heißen einfach und doppelt verjüngend, da sie im Endergebnis die Stufe der Tensoren um eins bzw. zwei verringern.

C.2 Tensoranalysis

In dieser Arbeit werden skalar-, vektor- oder auch tensorwertige Funktionen in einem mehrdimensionalen Raum verwendet. Hierfür wollen wir Gradient und Divergenz einführen.

Definition C.10 (Gradient,Divergenz). Der *Gradient* einer skalar-, vektor- oder tensorwertige Funktionen ist definiert als

$$\text{grad}(\dots) := \frac{\partial(\dots)}{\partial \mathbf{x}} := \frac{\partial(\dots)}{\partial x_i} \otimes \mathbf{e}_i := (\dots)_{,i} \otimes \mathbf{e}_i.$$

Für den Gradienten schreiben wir auch häufig den Nabla-Operator ∇ .

Die *Divergenz* einer skalar-, vektor- oder tensorwertige Funktionen ist definiert durch

$$\text{div}(\dots) := \nabla \cdot (\dots) = \frac{\partial(\dots)}{\partial x_i} \cdot \mathbf{e}_i = (\dots)_{,i} \cdot \mathbf{e}_i.$$

Bemerkung. Der Gradient vergrößert also die Stufe eines Tensors, während die Divergenz diese verringert.

Für den Gradienten und die Divergenz gelten einige Produktregeln bzgl. tensorwertigen Funktionen (von 0. bis 2. Stufe). Folgende werden wir davon benötigen.

Satz C.11 (Produktregeln). *Es seien $\phi(\mathbf{x})$ eine skalarwertiges, $\mathbf{a}(\mathbf{x})$ eine vektorwertiges und $\mathbf{T}(\mathbf{x})$ ein tensorwertiges (2. Stufe) Tensorfeld. Dann gelten für die Divergenz folgende Produktregeln.*

$$\text{div}(\phi \cdot \mathbf{a}) = \text{grad } \phi \cdot \mathbf{a} + \phi \cdot \text{div } \mathbf{a} \quad (\text{C.1})$$

$$\text{div}(\mathbf{a} \cdot \mathbf{T}) = \text{grad } \mathbf{a} : \mathbf{T} + \mathbf{a} \cdot \text{div } \mathbf{T} \quad (\text{C.2})$$

Beweis. Einfaches Nachrechnen. \square

Weiter gelten in der Tensoranalysis verschiedene Integralsätze von Gauß, Stokes und Green, von denen wir folgende aufführen wollen, wobei wir uns wieder auf $\Omega \subset \mathbb{R}^2, \partial\Omega = \Gamma$ beschränken.

Satz C.12 (Integralsatz von Gauß). *Es seien die Tensorfelder $\mathbf{a}(\mathbf{x}), \mathbf{T}(\mathbf{x})$ wie in Satz C.11 gegeben. Dann gelten die Sätze von Gauß*

$$\int_{\Omega} \text{div } \mathbf{a} d\Omega = \int_{\Gamma} \mathbf{a} \cdot \mathbf{n} d\Gamma, \quad (\text{C.3a})$$

$$\int_{\Omega} \text{div } \mathbf{T} d\Omega = \int_{\Gamma} \mathbf{T} \cdot \mathbf{n} d\Gamma, \quad (\text{C.3b})$$

wobei \mathbf{n} die äußere Einheitsnormale auf Γ bezeichne.

Beweis. Standardresultat aus der Analysis 2. \square

Korollar C.13 (1. Green'sche Formel). *Es seien $u, v : \Omega \rightarrow \mathbb{R}$ skalarwertige Funktionen. Dann folgt*

$$\int_{\Omega} v \Delta u \, dx + \int_{\Omega} \nabla u \cdot \nabla v \, dx = \int_{\Gamma} v \partial_{\mathbf{n}} u \, ds \quad (\text{C.4})$$

mit $\partial_{\mathbf{n}} u = \nabla u \cdot \mathbf{n}$.

Beweis. Es seien u, v wie vorausgesetzt. Dann ist ∇u vektorwertig und nach (C.1) gilt dann

$$\begin{aligned} \operatorname{div}(v \cdot \nabla u) &= \nabla v \cdot \nabla u + v \cdot \operatorname{div}(\nabla u) \\ &= \nabla v \cdot \nabla u + v \cdot \Delta u. \end{aligned}$$

Weiter gilt wegen (C.3a)

$$\int_{\Omega} \operatorname{div}(v \cdot \nabla u) \, d\Omega = \int_{\Gamma} v \cdot \underbrace{\nabla u \cdot \mathbf{n}}_{=\partial_{\mathbf{n}} u} \, d\Gamma.$$

Zusammen mit dem oberen Resultat folgt dann die Behauptung (C.4). \square

Anhang D

Quellcode

D.1 Implementierung des Fehlerschätzers für das Hindernisproblem

Im Folgenden ist der Quellcode angegeben, der für die Implementierung für das in Kapitel 4 beschriebene affine Hindernisproblem geschrieben wurde. Als letztes sind auch die Startdateien für die in Kapitel 6.1 dargestellten numerischen Beispiele angegeben.

```
1 function [A,f] = assemble(points,triangle,fun,num_of_nodes,option,u_S)
2 %ASSEMBLE evaluates the global matrix A and load vector f out of the
3 % given nodes, triangles, loadfunction, number of nodes and the
4 % Galerkin approximation u_S.
5
6 % ordering: triangles ↔ midpoints:
7 [midpoints,midtriangle] = midpoints_of_triangle(triangle,points);
8
9 % initialising the dimension and the solution:
10 np = size(points,2);
11 nt = size(triangle,2);
12 nmp = size(midpoints,2);
13
14 if nargin == 4
15     option = 'linear';
16     u_S = zeros(np,1);
17 end
18
19 if nargin == 5
20     u_S = zeros(np,1);
21 end
22
23 % beginning of the assembling:
24 switch lower(option)
25     case {'linear'}
26         % linear hatfunctionen on the reference-element:
27         hat = @(xi,eta) [1 - xi - eta; xi; eta];
28
29         % initialising of the global values:
30         A = sparse(np,np);
31         f = sparse(np,1);
32         my_tri = triangle(1:3,:);
```

D. Quellcode

```

31 case {'bubble','quadratic'}
32 % bubblefunctionen on the reference-element:
33 hat = @(xi,eta) [4*xi.*(1-xi-eta); 4*xi.*eta; 4*eta.*(1-xi-eta)];
34
35 % initialising of the global values:
36 A = sparse(nmp,nmp);
37 f = sparse(nmp,1);
38 my_tri = midtriangle;
39
40 end
41
42 % loop over the triangles for the assembling:
43 for i = 1:nt
44     poi = points(:,triangle(1:3,i));
45     u_S_loc = u_S(triangle(1:3,i));
46     tri = my_tri(:,i);
47
48     % evaluation of the local linear stiffness matrix and the
49     % Jacobian:
50     [S,fl,J] = local_mat(poi,u_S_loc,option);
51
52     % Evaluating the local vector fl: (here is Gauss-quadrature over
53     % triangles used)
54     [wi,gauss,ansatz_values] = quad_tri(poi,hat,num_of_nodes);
55
56     % quadrature over a triangle:
57     for k = 1:3
58         fl(k) = fl(k)+J*sum(wi.*fun(gauss(1,:),gauss(2,:)).*...
59             ansatz_values(k,:));
60     end
61
62     % assembling into the global stiffness matrix A and the load
63     % vector f:
64     for k = 1:3
65         for l = 1:3
66             A(tri(k),tri(l)) = A(tri(k),tri(l))+S(k,l);
67         end
68
69         f(tri(k)) = f(tri(k))+fl(k);
70     end
71 end
72
73 end

```

Code D.1: Assemblierung der Steifigkeitsmatrix A und des Lastvektor f

```

1 function rho_p =
2     eval_rho_p(nodes, triangles, midpoints, midtri, u_S, eps_V, fun)
3 %EVAL_RHO_P evaluates the local contribution rho_P of rho_S, given
4 % the nodes, triangles, midpoints of the edges (of the triangles)
5 % and die midpoints-triangle-ordering (midtri). Also it is given
6 % the solution auf die local defect problem eps_V and the function
7 % fun, which is part of the integral in rho_S. u_S are, as always,
8 % the Galerkin solution.
9
10 % Initializing:
11 np = size(nodes,2);
12 rho_p = zeros(np,1);
13
14 % Hat- and Bubble- functions:
15 phi_P = @(xi,eta) [1-xi-eta; xi; eta];
16 phi_E = @(xi,eta) [4*xi.*(1-xi-eta); 4*xi.*eta; 4*eta.*(1-xi-eta)];

```

```

11 % Evaluation of the weights and values of the function for the
12 % surface integral:
13 [wi,~,phi_E_values] = quad_tri([0,1,0;0,0,1],phi_E,7);
14 [~,~,phi_P_values] = quad_tri([0,1,0;0,0,1],phi_P,7);
15
16 for k = 1:np
17 % Ordering triangle ↔ reference function and support of phi_P:
18 [phi_p_local,w_p] = find(triangles(1:3,:)==k);
19 % indices of the edges in the considered triangle:
20 E_index = midtri(:,w_p);
21
22 % Evaluation of the surface integral of rho_p over w_p:
23 for j = 1:length(w_p)
24 % points of the considered triangle with Jacobi determinant:
25 mypoi = nodes(:,triangles(1:3,w_p(j)));
26 x = mypoi(1,:);
27 y = mypoi(2,:);
28 J = (x(2)-x(1))*(y(3)-y(1))-(x(3)-x(1))*(y(2)-y(1));
29 % Gauss-points and local contributions of eps_V:
30 eps_V_local = eps_V(E_index(:,j));
31 [~,gauss,~] = quad_tri(mypoi,@(x,y) 0,7);
32 % the functionvalues of the considered local hatfunction
33 phi_P:
34 phi_Pl_values = phi_P_values(phi_p_local(j),:);
35 % evaluation of the first integral of rho_p:
36 rho_p(k) = rho_p(k) + J *
37 sum(wi.*fun(gauss(1,:),gauss(2,:)).*...
38 (eps_V_local'*phi_E_values).*(phi_Pl_values)));
39 end
40
41 % Evaluation of the line integral over the edges E\in E_p:
42 % Evaluaton of the set E_p of the edges, which contain the
43 % point P:
44 [~,E_p] = find(midpoints(3:4,:)==k);
45 % Evaluation of the normal-fluxes j_E for all edges in E_p:
46 j_E = normal_flux(E_p,nodes,triangles,midpoints,midtri,u_S);
47
48 for i = 1:length(E_p)
49 % evaluating the points of the edges:
50 edge_poi_ind = midpoints(3:4,E_p(i));
51 edge_poi = nodes(:,edge_poi_ind);
52
53 % affin transformation of the local integral on the
54 % global edge by multiplication of the jacobian:
55 laenge = norm(edge_poi(:,1)-edge_poi(:,2));
56 global_phiPE_int = 1/3*laenge;
57
58 % determination of the functionvalues of eps_V(x_E):
59 epsV_loc = eps_V(E_p(i));
60
61 % adding the line integral to the local contribution
62 rho_p(k) = rho_p(k)+j_E(i)*epsV_loc*global_phiPE_int;
63 end
64 end
65 end

```

Code D.2: Berechnung der lokalen Anteile von ρ_S

D. Quellcode

```

1  function [eps_V_exact ,rho_E ,d_E ,a_phi] =
2    defect_problem_solution (points ,triangle ,mid_triangle ,rho_s ,u_S_mid ,
3    z_obs_midpoints)
4    %DEFECT_PROBLEM SOLUTION evaluates the solution of the local defect
5    % problem (2.9), given the nodes, triangles,
6    % triangle-midpoints-ordering, the functionvalues of the Galerkin
7    % solution u_S at the midpoints and the same for the solution.
8
9    % initializing:
10   ntri = size(triangle ,2);
11   nmp = size(u_S_mid ,1);
12
13   dxi_bubble = @(xi ,eta) [4 - 8*xi - 4*eta; 4*eta ; - 4*eta];
14   deta_bubble = @(xi ,eta) [-4*xi; 4*xi; 4 - 4*xi - 8*eta];
15   mymidtri = [mid_triangle; zeros(1 ,ntri)];
16   a_phi = zeros(nmp,1);
17
18   % evaluation of the value a(phi_E ,phi_E) to determine the norm of
19   % phi_E:
20   for i = 1 : nmp
21     index = find(mymidtri == i);
22     bubble_ind = mod(index ,4);
23     triangle_ind = ceil(index/4);
24
25     poi = points (: ,triangle (1:3 ,triangle_ind));
26     x = poi (1 ,:);
27     y = poi (2 ,:);
28     J = (x(2) - x(1)) * (y(3) - y(1)) - (x(3) - x(1)) * (y(2) - y(1));
29     a = ((x(3) - x(1))^2 + (y(3) - y(1))^2);
30     b = -((y(2) - y(1)) * (y(3) - y(1)) + (x(2) - x(1)) * (x(3) - x(1)));
31     c = ((x(2) - x(1))^2 + (y(2) - y(1))^2);
32
33     [wi ,~, val_dxi_bubble] = quad_tri(poi ,dxi_bubble ,7);
34     [~, ~, val_deta_bubble] = quad_tri(poi ,deta_bubble ,7);
35
36     for j = 1 : length(bubble_ind)
37       a_phi(i) = a_phi(i) + 1/J* sum(wi.* (a*val_dxi_bubble ...
38         (bubble_ind(j) ,:).^2 +
39         b*val_dxi_bubble(bubble_ind(j) ,:).*...
40         val_deta_bubble(bubble_ind(j) ,:)* 2 ...
41         +c*val_deta_bubble(bubble_ind(j) ,:).^2));
42     end
43
44   end
45
46   % evaluating the values of (2.11) and at least (2.10):
47   a_phi = a_phi.^ (1/2);
48   rho_E = rho_s ./ a_phi;
49   d_E = (u_S_mid - z_obs_midpoints).* a_phi;
50   eps_V_exact = max(-d_E ,rho_E) ./ a_phi;
51
52 end

```

Code D.3: Lösung des lokalen Defektpproblems (4.9)

```

1 function triangle_index =
2     find_triangle_refinement(rho_p, rho_global, osc_local, osc_global,
3         triangles, theta_rho, theta_osc, option)
4 %FIND_TRIANGLE_REFINEMENT evaluates the possible indices of
5     % triangles, which will be refined, given the local contribution
6     % rho_p of the error indicator rho_global, the local contributions
7     % osc_local of the oscillation terms osc_global, the triangles and
8     % two parameter to provide a boundary.
9
10    % Initializing the indices of the evaluated points or triangleindices:
11    np = length(rho_p);
12    point_index = zeros(np,1);
13    bound = zeros(np,1);
14    triangle_index = [];
15    counter = 1;
16
17    if nargin == 7
18        option = ',';
19    end
20
21    switch lower(option)
22        case 'symmetric'
23            % evaluating the points, which have got a large contribution
24            % to the error indicator:
25            while 1
26                new_rho = max(rho_p);
27                new_points = find(abs(rho_p - new_rho) < 0.001);
28                rho_p(new_points) = 0;
29                number_new_points = length(new_points);
30                bound(counter) = number_new_points * new_rho;
31                point_index(counter:counter+number_new_points-1) =
32                    new_points;
33                counter = counter + number_new_points;
34
35                % termination criterion for the search of the points:
36                if sum(bound) >= theta_rho*rho_global
37                    break;
38                end
39            end
40
41            % evaluation of the points, which oscillation contribution is
42            % high:
43            while 1
44                % termination criterion for the search of the points:
45                if sum(bound) >= theta_osc*osc_global
46                    break;
47                end
48
49                new_osc = max(osc_local);
50                new_point = find(abs(osc_local - new_osc) < 0.001);
51                osc_local(new_point) = 0;
52                number_new_points = length(new_point);
53                bound(counter) = number_new_points * new_osc;
54                point_index(counter:counter+number_new_points-1) =
55                    new_point;
56                counter = counter + number_new_points;
57            end
58
59        otherwise
60            % evaluating the points, which have got a large contribution
61            % to the error indicator:

```

D. Quellcode

```

51     while 1
52         [new_rho, index] = max(rho_p);
53         rho_p(index) = 0;
54         bound(counter) = new_rho;
55         point_index(counter) = index;
56         counter = counter + 1;
57
58         % termination criterion for the search of the points:
59         if sum(bound) >= theta_rho*rho_global
60             break;
61         end
62     end
63
64     % evaluation of the points, which oscillation contribution is
65     % high:
66     while 1
67         % termination criterion for the search of the points:
68         if sum(bound) >= theta_osc*osc_global
69             break;
70         end
71
72         [new_osc, index] = max(osc_local);
73         osc_local(index) = 0;
74         bound(counter) = new_osc;
75         point_index(counter) = index;
76         counter = counter + 1;
77     end
78
79     % determination of the triangles indices for the refinement:
80     point_index = setdiff(point_index, 0);
81
82     for k = 1:length(point_index)
83         help_index = neighbourhood(point_index(k), triangles, 'point');
84         triangle_index = union(triangle_index, help_index);
85     end
86
87 end

```

Code D.4: Berechnung der zu verfeinernden Dreiecksindizes

```

1 function grad_u = gradu(points, zvalues)
2 %GRADU determines for given (x,y,z)-values the gradient of the
3 % function u=a*x+b*y+c*z, that is grad u=(a,b).
4
5 % x- and y-values of the points:
6 x = points(1,:);
7 y = points(2,:);
8
9 % the Jacobian:
10 J = (x(2)-x(1))*(y(3)-y(1)) - (x(3)-x(1))*(y(2)-y(1));
11
12 % evaluating grad_u:
13 grad_u = 1/J * ([y(3)-y(1), y(1)-y(2); x(1)-x(3), x(2)-x(1)]*...
14 [zvalues(2)-zvalues(1); zvalues(3)-zvalues(1)]);
15
16 end

```

Code D.5: Berechnung des Gradienten von der Galerkin-Approximation auf einem Dreieck

```

1 function non_bound_index = inner_points(adjacency)
2 %DIRICHLET_BOUNDARY assembles the inner points, that is the non
3 % boundary points, by using the adjacency matrix given by assemb.
4 % evaluating the inner points out of the adjacency matrix:
5 [m,n] = size(adjacency);
6 non_bound_index = find(prod(adjacency == zeros(m,n)));
7
8 end

```

Code D.6: Bestimmung der inneren Knoten $\mathcal{N} \cap \Omega$

```

1 function N0_set = N0(uS_values, inner_points, obstacle_values)
2 %N0 determines the set  $N^0$  of the inner points, which are in contact
3 %with the obstacle. It will be done by simply compare the function
4 %values of u_S and the obstacle.
5 index = find(abs(uS_values - obstacle_values) < 0.00001);
6 N0_set = intersect(index, inner_points);
7
8 end

```

Code D.7: Bestimmung der inneren Kontaktknoten \mathcal{N}^0

```

1 function Nplus_set = Nplus(contact_set, inner_points, nodes)
2 %NPLUS computes the set  $N^+$  of the inner non-contact nodes, given the
3 %set of contact nodes  $N^0$ , inner points and the set of all nodes.
4 np = length(nodes);
5 index = setdiff(1:np, contact_set);
6 Nplus_set = intersect(index, inner_points);
7
8 end

```

Code D.8: Bestimmung der inneren Nicht-Kontaktknoten \mathcal{N}^+

```

1 function Nplusplus_set = Nplusplus(Nplus_set, edges, rho_E, d_E)
2 %NPLUSPLUS calculates the set  $N^{++}$  of non-contact nodes, where the
3 %approximate error eps_V is not in contact.
4
5 % Initialization:
6 Nplusplus_set = zeros(size(Nplus_set));
7
8 % evaluation of the entries of  $N^{++}$ :
9 for i = 1:length(Nplus_set)
10    [~, E_p] = find(edges == Nplus_set(i));
11
12    % verify if for all  $E \in E_p$  is valid:  $\rho_E \geq -d_E$ 
13    if all(rho_E(E_p) + d_E(E_p) >= 0)
14        Nplusplus_set(i) = Nplus_set(i);
15    end
16
17 % elimination of the zeros:
18 Nplusplus_set = setdiff(Nplusplus_set, 0);
19
20 end

```

Code D.9: Bestimmung der Menge \mathcal{N}^{++}

D. Quellcode

```

1  function [N0plus_set,N0minus_set] = N0plusminus(N0_set, nodes,
2      triangles, edges, edge_triangles, f_load, obstacle, uS_values)
3  %N0PLUSMINUS evaluates the sets  $N^{\{0\}+}$  and  $N^{\{0\}-}$  for the calculation
4  % of the oscillationterms, given the inner contact nodes N^0_set,
5  % the nodes of the mesh, the triangle ordering, the edges and the
6  % midpoints-triangle ordering, the loadfunction, the obstacle and
7  % vector of u_S values.
8
9  % parameter for the approximate of zero:
10 zero = 0.0001;
11
12 % initializing of the sets  $N^{\{0\}+}$  and  $N^{\{0\}-}$ :
13 N0plus_set = zeros(size(N0_set));
14 N0minus_set = zeros(size(N0_set));
15
16 % calculating the points of the reference element to compare the
17 % functionvalues later:
18 [xi, eta] = meshgrid(0:0.01:1, 0:0.01:1);
19 index_out_of_bounds = find(xi+eta>1);
20 xi(index_out_of_bounds) = 0;
21 eta(index_out_of_bounds) = 0;
22 [m_mesh, n_mesh] = size(xi);
23 np = m_mesh * n_mesh;
24 index_within = setdiff(1:np, index_out_of_bounds);
25
26 % hat functions with functionvalues:
27 phi_P = @(xi, eta) [1 - xi - eta; xi; eta];
28 phi_P_values = phi_P(xi, eta);
29
30 % evaluation of the indices of the  $N^{\{0\}+}$  nodes:
31 for i = 1:length(N0_set)
32     % support of the shape function and index of the local shape
33     % function:
34     [phi_p_local, w_p] = find(triangles(1:3,:)==N0_set(i));
35     flag_plus = 0;           % verification value for the condition of N0+
36     flag_minus = 0;          % verification value for the condition of N0-
37
38     for j = 1:length(w_p)
39         % vertices of the triangle and functionvalues of u_S:
40         p_index = triangles(1:3,w_p(j));
41         uS_local = uS_values(p_index);
42         mypoi = nodes(:, p_index);
43         x = mypoi(1,:);
44         y = mypoi(2,:);
45
46         % transformation from the local onto the global triangle:
47         xval = x(1)+(x(2)-x(1)).*xi+(x(3)-x(1)).*eta;
48         yval = y(1)+(y(2)-y(1)).*xi+(y(3)-y(1)).*eta;
49
50         % evaluating the functionvalues of u_S-psi:
51         z = uS_local(1)*phi_P_values(1:m_mesh,:)+uS_local(2)*...
52             phi_P_values(1+m_mesh:2*m_mesh,:)+uS_local(3)*...
53             phi_P_values(1+2*m_mesh:3*m_mesh,:)-obstacle(xval, yval);
54
55         % computing the functionvalues of the load f_load:
56         f = f_load(xval, yval);
57
58         % verifying, if u_S-psi>0 and if u_S=psi & f<=0:
59         switch phi_p_local(j)
60             case 1
61                 new_within = setdiff(index_within, 1);
62
63             case 2
64                 if z > 0
65                     new_within = setdiff(index_within, 1);
66
67                 else
68                     if f <= 0
69                         new_within = setdiff(index_within, 1);
70
71                     end
72
73                 end
74
75             case 3
76                 if z > 0
77                     new_within = setdiff(index_within, 1);
78
79                 else
80                     if f <= 0
81                         new_within = setdiff(index_within, 1);
82
83                     end
84
85                 end
86
87             otherwise
88                 if z > 0
89                     new_within = setdiff(index_within, 1);
90
91                 else
92                     if f <= 0
93                         new_within = setdiff(index_within, 1);
94
95                     end
96
97                 end
98
99             end
100         end
101
102         if abs(z) < zero
103             if abs(f) <= zero
104                 new_within = setdiff(index_within, 1);
105
106             end
107
108         end
109
110     end
111
112 end
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154

```

```

55     case 2
56         new_within = setdiff(index_within, sub2ind([m_mesh, ...
57             n_mesh], 1, n_mesh));
58     case 3
59         new_within = setdiff(index_within, m_mesh);
60     end
61
62     % If any z-value is less than zero, the point cannot be an
63     % element of N^{0+}, that is, why flag_plus = 1. If all
64     % f_load-values and all z-values are less than zero, p
65     % could be in N^{0-}:
66     if any(z(new_within) <= zero)
67         flag_plus = 1;
68
69     z_nodes = uS_local-obstacle(x,y)';
70     if (all(abs(z_nodes) <= zero) && all(f(index_within)<=0))
71         flag_minus = flag_minus + 1;
72     end
73 end
74
75 % verification if there is no contact except of the point p:
76 if flag_plus == 0
77     N0plus_set(i) = N0_set(i);
78     continue;
79 end
80
81 % verification of j_E <= 0 for all E in E_p: evaluating of the
82 % set of edges E_p, which contains the point p:
83 [~,E_p] = find(edges(3:4,:)==N0_set(i));
84 % evaluating the normal fluxes of all edges of E_p:
85 j_E =
86     normal_flux(E_p,nodes,triangles,edges,edge_triangles,uS_values)
87 % verificating, if there is full contact and f<=0 and j_E<=0 for
88 % all E in E_p:
89 if (flag_minus == length(w_p) && all(j_E <= zero))
90     N0minus_set(i) = N0_set(i);
91 end
92
93 end
94
95 % elimination of the zeros:
96 N0plus_set = setdiff(N0plus_set,0);
97 N0minus_set = setdiff(N0minus_set,0);
98
99 end

```

Code D.10: Berechnung der Menge an Knoten aus \mathcal{N}^{0+} und \mathcal{N}^{0-}

```

1 function [S,f_local,J] = local_mat(points,uS_local,option)
2 %LOCALMAT computes the Jacobian J, a local stiffness matrix S and if
3 %option='bubble' a local vector f_local, which will be needed for
4 %a(u_S,*) in rho_S. LOCALMAT expects the nodes (points) of the
5 %local triangle, the z-values uS_local of this nodes and an
6 %option, which can be 'linear' or 'bubble.
7
8 % initialization of f_local:
9 f_local = zeros(3,1);
10
11 % x- and y-values of the nodes:
12 x = points(1,:);
13 y = points(2,:);

```

D. Quellcode

```

11 % the Jacobian J and other factors for the affine transformation from
12 % the reference element onto a arbitrary triangle:
13 J = (x(2)-x(1))*(y(3)-y(1)) - (x(3)-x(1))*(y(2)-y(1));
14 a = 1/J * ((x(3)-x(1))^2 + (y(3)-y(1))^2);
15 b = -1/J * ((y(2)-y(1))*(y(3)-y(1)) + (x(2)-x(1))*(x(3)-x(1)));
16 c = 1/J * ((x(2)-x(1))^2 + (y(2)-y(1))^2);
17
18 switch lower(option)
19   case {'linear'}
20     % local stiffness matrix for linear shape functions:
21     S = [ a/2+b+c/2, -a/2-b/2, -b/2-c/2;
22           -a/2-b/2, a/2, b/2;
23           -b/2-c/2, b/2, c/2 ];
24   case {'bubble'}
25     % local stiffness matrix for quadratic shape functions:
26     S_quad = 4/3* [ a+b+c, -b-c, b;
27                      -b-c, a+b+c, -a-b;
28                      b, -a-b, a+b+c ];
29     S = diag(diag(S_quad));
30
31   % gradient of u_S:
32   gradu_S = gradu(points, uS_local);
33   % transformation onto the reference element:
34   gradu_S = [x(2)-x(1), y(2)-y(1); x(3)-x(1), y(3)-y(1)]*gradu_S';
35   % f_local:
36   f_local = -(a*gradu_S(1)+b*gradu_S(2))*[0; 2/3; -2/3]...
37   -(b*gradu_S(1)+c*gradu_S(2))*[-2/3; 2/3; 0];
38   case {'quadratic'}
39     % local stiffness matrix for quadratic shape functions:
40     S = 4/3* [ a+b+c, -b-c, b;
41                 -b-c, a+b+c, -a-b;
42                 b, -a-b, a+b+c ];
43   otherwise
44     error('Unknown option');
45 end
46

```

Code D.11: Bestimmung der lokalen Steifigkeitsmatrix im linearen und quadratischen Fall

```

1 function [neighbours, flag] =
2   neighbourhood(edges_or_point, triangles, option)
%EDGE_NEIGHBOURHOOD computes the indices of the triangles, which are
%the neighbours for the given edges or point. The vector flag
%shows, if an edge has got only one neighbour (flag = 0) or two
%(flag = 1).
3
4 n = length(edges_or_point);
5 flag = zeros(1,n);
6
7 switch lower(option)
8   case {'edge', 'edges'}
9     neighbours = zeros(2,n);
10
11   for i = 1:n
12     [~,neighbours(:,i)] = find(triangles==edges_or_point(i));
13
14     if neighbours(1,i) ~= neighbours(2,i)
15       flag(i) = 1;
16     end

```

```

17     end
18
19     case { 'point'}
20         [~,neighbours] = find(triangles(1:3,:)==edges_or_point);
21 end

```

Code D.12: Berechnung der Indizes anliegender Dreiecke

```

1 function [midpoints,mid_triangle] =
2     midpoints_of_triangle(triangle,points)
%MIDPOINTS_OF_TRIANGLE evaluates the midpoints of the edges of the
% triangle T. The first and second row stores the x- and y-values
% of the midpoints and the third and fourth row stores the indices
% of the nodes, that are the start and end of the edge. The matrix
% mid_triangle restores the indices of the midpoints in the rows of
% every triangle by the columns (by positive mathematical
% orientation).
3
4 [nt] = size(triangle,2);
5 mid_triangle = zeros(3,nt);
6 midpoints = zeros(4,1);
7 ind_counter = 1;
8
9 for i = 1:nt
% determination of the nodes of a triangle:
10    tri = triangle(1:3,i);
11    poi = points(:,tri);
12
13    % evaluation of the midpoints to this triangle:
14    mid_poi = [1/2*(poi(:,1)+poi(:,2)), 1/2*(poi(:,2)+poi(:,3)), ...
15               1/2*(poi(:,1)+poi(:,3)); tri([1,2]), tri([2,3]), tri([1,3])];
16
17    % verification, if the midpoints have been already calculated:
18    [glob,loc] = ismember(midpoints(1:2,:),mid_poi(1:2,:),'rows');
19    global_ind = find(glob);
20    local_ind = loc(global_ind);
21
22    % case distinction and determination of the vector of midpoints:
23    switch length(global_ind)
24        case 1
25            ind = setdiff([1,2,3],local_ind);
26            mid_triangle(local_ind,i) = global_ind;
27            mid_triangle(ind,i) = [ind_counter,ind_counter+1];
28            midpoints(:,[ind_counter,ind_counter+1]) = mid_poi(:,ind);
29            ind_counter = ind_counter + 2;
30        case 2
31            ind = setdiff([1,2,3],local_ind);
32            mid_triangle(local_ind,i) = global_ind;
33            mid_triangle(ind,i) = ind_counter;
34            midpoints(:,ind_counter) = mid_poi(:,ind);
35            ind_counter = ind_counter + 1;
36        case 3
37            mid_triangle(local_ind,i) = global_ind;
38            otherwise
39                mid_triangle(:,i) = [ind_counter,ind_counter+1,ind_counter+2];
40                midpoints(:,ind_counter:ind_counter+2) = mid_poi;
41                ind_counter = ind_counter + 3;
42        end
43    end
44 end
45 end

```

Code D.13: Bestimmung der Mittelpunkte und Zuordnung zu den Dreiecken

D. Quellcode

```

1  function j_E =
2      normal_flux(E_p, nodes, triangles, edges, edge_triangles, uS_values)
3      %NORMALFLUX computes the normal flux for all given edges E \in E_p.
4      % The other given objects are: The nodes of the triangulation, the
5      % indices of the triangles-nodes ordering, the
6      % midpoints/edges-matrix edges, the edges-triangle-ordering
7      % edge_triangle and the functionvalues of the Galerkin solution
8      % uS_values.
9
10     % Initializing:
11     j_E = zeros(size(E_p));
12
13     % Evaluate the neighbours of all edges in the edges-set E_p:
14     [neighbours, flag] = neighbourhood(E_p, edge_triangles, 'edges');
15
16     % Computation of the single normal fluxes:
17     for k = 1:length(E_p)
18         % Evaluation of the edge-points:
19         edge_poi_ind = edges(3:4,E_p(k));
20         edge_poi = nodes(:,edge_poi_ind);
21
22         % Calculation of the gradient of u_S on T_1 and T_2:
23         neigh_tri.ind = neighbours(:,k);
24         neigh_tri = triangles(1:3,neigh_tri.ind);
25         p_T = nodes(:,neigh_tri);
26         uS_T = uS_values(neigh_tri);
27
28         p_T1 = p_T(:,1:3);
29         p_T2 = p_T(:,4:6);
30         uS_T1 = uS_T(:,1);
31
32         if flag(k) == 0
33             uS_T2 = zeros(3,1);
34         else
35             uS_T2 = uS_T(:,2);
36         end
37
38         graduS_T = [gradu(p_T1,uS_T1);gradu(p_T2,uS_T2)];
39
40         % the normalvector from T_1 to T_2:
41         connect = edge_poi(:,2)-edge_poi(:,1);
42         orth_connect = [-connect(2);connect(1)];
43         n = 1/norm(orth_connect)*orth_connect;
44
45         % Verification, if n shows from T_1 to T_2:
46         p3_T1 = setdiff(p_T1',edge_poi', 'rows')';
47         edge_test = (p3_T1-edge_poi(:,1))';
48
49         if edge_test*n > 0
50             n = -n;
51         end
52
53         normal = graduS_T*n;
54         j_E(k) = normal(2)-normal(1);
55     end
56
57

```

Code D.14: Berechnung der Normalflüsse j_E für alle $E \in \mathcal{E}_p$

```

1 function [ osc1_val , osc1_vec ] =
2     osc1( N0plus_set , obstacle_values , nodes , triangles , uS_values )
%OSCI evaluates the obstacle oscillation osc_1(u_S,phi). The general
condition in this case is that psi is affine.
3
4 % Initializing:
5 osc1_vec = zeros( length(nodes) , 1 );
6
7 % Evaluation of the sum over the points of N^{0+}:
8 for i = 1:length(N0plus_set)
    % the help value int_h for the integral of the gradient:en:
    int_h = 0;
    % evaluating the support of the shape funktions and the index of
    the local shape function:
12    [~,w_p] = find( triangles(1:3,:)==N0plus_set(i));
13
14    % computation of the single norms (integrals):
15    for j = 1:length(w_p)
        % nodes of the triangle, Jacobian and factors for the affine
        trafo:
17        p_index = triangles(1:3,w_p(j));
18        mypoi = nodes(:,p_index);
19        x = mypoi(1,:);
20        y = mypoi(2,:);
21        J = (x(2)-x(1))*(y(3)-y(1)) - (x(3)-x(1))*(y(2)-y(1));
22        a = 1/J * ((x(3)-x(1))^2 + (y(3)-y(1))^2);
23        b = -1/J * ((y(2)-y(1))*(y(3)-y(1)) +
                    (x(2)-x(1))*(x(3)-x(1)));
24        c = 1/J * ((x(2)-x(1))^2 + (y(2)-y(1))^2);
25
26        % evaluate the gradient of psi and u_S:
27        grad_psi = gradu(mypoi,obstacle_values(p_index));
28        grad_u = gradu(mypoi,uS_values(p_index));
29
30        % transformation of the gradients onto the reference element:
31        grad_psi = [x(2)-x(1), y(2)-y(1); x(3)-x(1),
                     y(3)-y(1)]*grad_psi';
32        grad_u = [x(2)-x(1), y(2)-y(1); x(3)-x(1), y(3)-y(1)]*grad_u';
33
34        % to integrate the gradients, we just multiplicate the
        area of the reference element with the high (the
        functionvalue), because the gradients are constant:
35        int_h = int_h + 1/2*(a*grad_psi(1)^2+2*b*grad_psi(1)*...
36                           *grad_psi(2)+c*grad_psi(2)^2-2*(a*grad_psi(1)*grad_u(1)*...
37                           *grad_psi(2)+b*(grad_psi(1)*grad_u(2)+grad_psi(2)*grad_u(1))*...
38                           +c*grad_psi(2)*grad_u(1))+a*grad_u(1)^2+2*b*(grad_u(1)*...
39                           *grad_u(2))+c*grad_u(2)^2);
40    end
41
42    osc1_vec( N0plus_set(i) ) = int_h ;
43 end
44
45 % square root of the sum of the integrals:
46 osc1_val = sqrt(sum(osc1_vec));
47
48 end

```

Code D.15: Bestimmung der Oszillation $\text{osc}_1(u_S, \psi)$

D. Quellcode

```

1  function [osc2_val,osc2_vec] = osc2(Nplusplus_set,N0minus_set,
2      nodes,triangles,edges,f_load)
3  %OSC2 evaluates the oscillation osc_2(u_S,psi,f).
4
5  % initialization:
6  np = length(nodes);
7  osc2_vec = zeros(np,1);
8  N_set = 1:np;
9  N_sum2 = setdiff(N_set,union(Nplusplus_set,N0minus_set));
10
11 % evaluations of h_P for the two sums of osc_2:
12     function h_P = eval_hP(set,global_nodes,global_edges)
13         % initialization:
14         h_P = zeros(size(set));
15
16         for l = 1:length(h_P)
17             % computing of the edges, which contain p:
18             [~,E_p] = find(global_edges(3:4,:)==set(l));
19
20             % calculation of the lengths of E in E_p:
21             points_1 = global_nodes(:,global_edges(3,E_p));
22             points_2 = global_nodes(:,global_edges(4,E_p));
23             points_diff = points_1 - points_2;
24             h_P(l) = max(sqrt(points_diff(1,:).^2 + points_diff(2,:).^2));
25         end
26     h_P1 = eval_hP(Nplusplus_set,nodes,edges);
27     h_P2 = eval_hP(N_sum2,nodes,edges);
28
29 % evaluating of the sums:
30 for i = 1:length(Nplusplus_set)
31     int_h = 0;
32     % support of phi_p, that is the indices of the triangles:
33     [~,w_p] = find(triangles(1:3,:)==Nplusplus_set(i));
34
35     % computing of the mean-values f_p:
36     area_wp = 0;
37     intfp_h = 0;
38     for k = 1:length(w_p)
39         % local points of the considered triangle and the Jacobian:
40         mypoi = nodes(:,triangles(1:3,w_p(k)));
41         x = mypoi(1,:);
42         y = mypoi(2,:);
43         J = (x(2)-x(1))*(y(3)-y(1))-(x(3)-x(1))*(y(2)-y(1));
44         % evaluating the Gauss-points for the quadrature:
45         [wi,gauss,~] = quad_tri(mypoi,@(x,y) 0,7);
46         % quadrature:
47         intfp_h = intfp_h + J *
48             sum(wi.*f_load(gauss(1,:),gauss(2,:)).^2);
49         % computing of |w_p|:
50         area_wp = area_wp + J/2;
51     end
52     f_P = 1/area_wp*intfp_h;
53
54     % calculation of the norm:
55     for j = 1:length(w_p)
56         % local points of the considered triangle and the Jacobian:
57         mypoi = nodes(:,triangles(1:3,w_p(j)));
58         x = mypoi(1,:);
59         y = mypoi(2,:);
60         J = (x(2)-x(1))*(y(3)-y(1))-(x(3)-x(1))*(y(2)-y(1));

```

```

60 % evaluating the Gauss-points for the quadrature:
61 [wi,gauss,~] = quad_tri(mypoi,@(x,y) 0,7);
62 % quadrature:
63 int_h = int_h + J * sum(wi.*f_load(gauss(1,:),gauss(2,:))...
64 - f_P).^2;
65 end
66
67 osc2_vec(Nplusplus_set(i)) =
68 osc2_vec(Nplusplus_set(i))+h_P1(i)^2*int_h;
69 end
70
71 for i = 1:length(N_sum2)
72 int_h = 0;
73 % support of phi_p, that is the indices of the triangles:
74 [~,w_p] = find(triangles(1:3,:)==N_sum2(i));
75
76 % calculation of the norm:
77 for j = 1:length(w_p)
78 % local points of the considered triangle and the Jacobian:
79 mypoi = nodes(:,triangles(1:3,w_p(j)));
80 x = mypoi(1,:);
81 y = mypoi(2,:);
82 J = (x(2)-x(1))*(y(3)-y(1))-(x(3)-x(1))*(y(2)-y(1));
83 % evaluating the Gauss-points for the quadrature:
84 [wi,gauss,~] = quad_tri(mypoi,@(x,y) 0,7);
85 % quadrature:
86 int_h = int_h + J * sum(wi.*f_load(gauss(1,:),gauss(2,:))).^2;
87 end
88 osc2_vec(N_sum2(i)) = osc2_vec(N_sum2(i))+h_P2(i)^2*int_h;
89 end
90
91 osc2_val = sqrt(sum(osc2_vec));
92 end

```

Code D.16: Berechnung der Oszillationsterme $\text{osc}_2(u_{\mathcal{S}}, \psi, f)$

```

1 function [wi,gauss,ansatz_values] =
2     quad_tri(points,ansatz_fun,num_of_nodes)
%QUAD_TRI computes the weights, Gauss-points and functionvalues of
the shape functions, which are needed for the quadrature. The
matrix points stores the x-,y-values of the nodes of the
triangle, ansatz_fun a vector of shape functions and num_of_nodes
the number of nodes for the quadrature formula.
3
4 % x- and y-values of the points:
5 x = points(1,:);
6 y = points(2,:);
7
8 switch num_of_nodes
9     % determination of the weights and the local points xi,eta:
10    case 3
11        wi = [1/6,1/6,1/6];
12        local_poi = [1/2,1/2,0;0,1/2,1/2];
13        xi = local_poi(1,:);
14        eta = local_poi(2,:);
15
16    case 7
17        wh = [(155-sqrt(15))/2400,(155+sqrt(15))/2400];
18
19        wi = [9/80,wh(1),wh(1),wh(1),wh(2),wh(2),wh(2)];
20        local_poi = [1/3,(6-sqrt(15))/21,(9+2*sqrt(15))/21,...
```

D. Quellcode

```

21      (6-sqrt(15))/21,(6+sqrt(15))/21,(9-2*sqrt(15))/21, ...
22      (6+sqrt(15))/21;1/3,(6-sqrt(15))/21,(6-sqrt(15))/21, ...
23      (9+2*sqrt(15))/21,(6+sqrt(15))/21,(6+sqrt(15))/21, ...
24      (9-2*sqrt(15))/21];
25      xi = local_poi(1,:);
26      eta = local_poi(2,:);
27
28      otherwise
29          error('keine Quadraturformel bekannt');
30
31 end
32
33 % evaluation of the Gauss-points by using xi and eta with the points
34 % x and y:
35 gauss(1,:) = x(1)+(x(2)-x(1))*xi+(x(3)-x(1))*eta;
36 gauss(2,:) = y(1)+(y(2)-y(1))*xi+(y(3)-y(1))*eta;
37
38 % computing the functionvalues of the shape functions in the local
39 % coordinates:
40 ansatz_values = ansatz_fun(xi,eta);
41
42 end

```

Code D.17: Gewichte und Stützstellen für die Quadratur

```

1 function
2     [u_S,points,edges,triangles,midtri,midpoints,rhoS_plot,IQ_plot,
3      J_error,osc_term,osc1_term,osc2_term,recursion_depth,
4      degree_of_freedom,time_vec] = adaptive_refinement_solution
5      (points,edges,triangles,solution,load_fun,obstacle,geo_data,J_exact
6      max_error,para_rho,para_osc,max_points,max_recursion)
7 %ADAPTIVE_REFINEMENT SOLUTION uses the adaptive refinement strategy
8 %shown
9 %in chapter 4 and evaluates the solution on a adaptive refined mesh.
10
11 % initializing all local values:
12 u_S = solution;
13 J_u = J_exact;
14 rhoS_plot = zeros(max_recursion,1);
15 IQ_plot = zeros(max_recursion,1);
16 J_error = zeros(max_recursion,1);
17 osc_term = zeros(max_recursion,1);
18 osc1_term = zeros(max_recursion,1);
19 osc2_term = zeros(max_recursion,1);
20 recursion_depth = 1;
21 time_vec = zeros(max_recursion,1);
22 degree_of_freedom = zeros(max_recursion,1);
23
24 while 1
25     tic
26     % further initializations:
27     np = size(points,2);
28     degree_of_freedom(recursion_depth) = np;
29
30     % evaluation of the midpoints:
31     [midpoints,midtri] = midpoints_of_triangle(triangles,points);
32     nmp = size(midpoints,2);
33
34     % computation of the functionvalues of the obstacle onto the mesh
35     % nodes and midpoints:
36     z_obs_prob = obstacle(points(1,:),points(2,:));
37     z_obs_midpoints = obstacle(midpoints(1,:),midpoints(2,:));

```

```

31     if size(z_obs_prob,1) == 1
32         z_obs_prob = z_obs_prob';
33         z_obs_midpoints = z_obs_midpoints';
34     end
35
36 % assembling of the matrix/vector data and the evaluation of the
37 % Dirichlet boundary data:
38 [A,f] = assemble(points, triangles, load_fun, 7, 'linear');
39 [~,~,H,R]=assemb(geo_data.mygeom, points, edges);
40
41 % solution of the variational inequality with
42 % active-set/inner-points-method:
43 u_S = sparse(u_S);
44 z_obs_prob = sparse(z_obs_prob);
45 opts = optimset('Algorithm','interior-point-convex','LargeScale',
46 % on', 'Display', 'off');
47 [u_S, J_uS] = quadprog(A, -f, [], [], H, R, z_obs_prob, [], u_S, opts);
48
49 % computing the functionvalues of u_S onto the midpoints:
50 u_S_mid = zeros(nmp,1);
51
52 for j = 1 : nmp
53     index = midpoints([3,4],j);
54     u_S_mid(j) = (u_S(index(1))+u_S(index(2)))/2;
55 end
56
57 % the solution of the local defect problem by assembling the
58 % matrix with the bubble functions and using the equations in
59 % (4.10) and (4.11):
60 [A_Q, rhoS_phiE] =
61     assemble(points, triangles, load_fun, 7, 'bubble', u_S);
62 [eps_V, rho_E, d_E, ~] = defect_problem_solution(points, triangles,
63 midtri, rhoS_phiE, u_S_mid, z_obs_midpoints);
64
65 % the hierarchical error estimator: evaluation of rho_S(eps_V)
66 % with the equation beyond (4.68):
67 rhoS_glob = eps_V * rhoS_phiE;
68 rhoS_plot(recursion_depth) = rhoS_glob;
69
70 % computation of -I_Q(eps_V) with (4.6):
71 IQ_plot(recursion_depth) = -1/2*(eps_V.^2) * diag(A_Q) + rhoS_glob;
72
73 % calculation of the error of -I(e):
74 J_error(recursion_depth) = J_uS - J_u;
75
76 % evaluating the local contributions of rho_S with Lemma 4.14:
77 rho_p = eval_rho_p(points, triangles, midpoints, midtri, u_S, eps_V,
78 load_fun);
79
80 % determination of the sets N0, N0+, N+, N++, N0- for the
81 % oscillationterms:
82 inner_points_omega = inner_points(H);
83 N0_set = N0(u_S, inner_points_omega, z_obs_prob);
84 Nplus_set = Nplus(N0_set, inner_points_omega, points);
85 [N0plus_set, N0minus_set] = N0plusminus(N0_set, points,
86 triangles, midpoints, midtri, load_fun, obstacle, u_S);
87 Nplusplus_set = Nplusplus(Nplus_set, midpoints, rho_E, d_E);
88
89 % evaluation of the oscillationterms:

```

D. Quellcode

```

81 [osc1_term(recursion_depth), osc1_local] =
82     osc1(N0plus_set, z_obs_prob, points, triangles, u_S);
83 [osc2_term(recursion_depth), osc2_local] =
84     osc2(Nplusplus_set, N0minus_set, points,
85         triangles, midpoints, load_fun);
86 osc_local = osc1_local + osc2_local;
87 osc_term(recursion_depth) = sqrt(osc1_term(recursion_depth)^2 +
88     osc2_term(recursion_depth)^2);

89 % calculating the indices of the triangles, which have to be
90 % refined:
91 refine_triangle = find_triangle_refinement(rho_p, rho_S_glob,
92     osc_local, osc_term(recursion_depth), triangles, para_rho,
93     para_osc, 'symmetric');

94 % refinement of the mesh:
95 [p_h, e_h, t_h, uS_h] = refinemesh(geo_data.mygeomg, points, edges,
96     triangles, u_S, refine_triangle);

97 % termination criterion: if the number of the nodes is too large,
98 % no triangle will be refined or the hierarchical error
99 % estimator is small enough:
100 if isempty(refine_triangle) || rho_S_glob < max_error ||
101     recursion_depth == max_recursion || length(p_h) > max_points)
102     length(p_h)
103     fprintf('%s %f.\n', 'Die Rekursionstiefe ist ',
104             recursion_depth);
105     break;
106 else
107     points = p_h;
108     edges = e_h;
109     triangles = t_h;
110     u_S = uS_h;
111     recursion_depth = recursion_depth + 1;
112 end
113 time_vec(recursion_depth) = toc;
114
115
116
117 end
118
119 % elimination of the zeros in the vectors of the error/-estimator:
120 rho_S_plot = rho_S_plot(1:recursion_depth);
121 IQ_plot = IQ_plot(1:recursion_depth);
122 J_error = J_error(1:recursion_depth);
123 osc_term = osc_term(1:recursion_depth);
124 osc1_term = osc1_term(1:recursion_depth);
125 osc2_term = osc2_term(1:recursion_depth);
126 degree_of_freedom = degree_of_freedom(1:recursion_depth);
127
128
129 end

```

Code D.18: Adaptive Verfeinerung des Gitters und Lösung des Hindernisproblems

```

1 clear
2 clear all
3 clc
4
5 % loading of the geometry data (non-constant boundary):
6 data = load('square_with_unconst_dirichlet.mat');
7 % loadfunction f (here contant):
8 fun = @(x,y) -2*ones(size(x));

```

```

9 % loading the exakt data for the given problem:
10 data_exact = load('fval_log_u.mat');
11 J_u = data_exact.fval;
12 % obstacle function:
13 my_obstacle = @(x,y) zeros(size(x));
14
15 % initializing the mesh:
16 h = 2;
17 [p,e,t] = initmesh(data.mygeomg, 'Hmax',h);      %square: [-1,1]^2
18
19 % initialization of the global values:
20 u_S = [];
21 itermax = 5;           % maximum iteration depth
22 nmax = 1000000;        % maximum number of nodes
23 eps = 0.001;          % upper bound for the hierarchical error
24 estimate
25 theta_rho = 0.4;       % contraction parameter for local contributions
26             % of the error estimate
27 theta_osc = 0.3;       % contraction parameter for local contributions
28             % of the oscillations
29
30 tic
31 % adaptive algorithm:
32 [u_S,p,e,t,midtri,midpoints,rhoS_plot,IQ_plot,J_error,osc_term,
33   osc1_term,osc2_term,recursion_depth,degree_of_freedom] =
34   adaptive_refinement_solution(p,e,t,u_S,fun,my_obstacle,
35   data,J_u,eps,theta_rho,theta_osc,nmax,itermax);
36 toc
37
38 % plot of the mesh with the nodes and midpoints/edges:
39 figure(1)
40 subplot(2,1,1);pdemesh(p,e,t);
41 title('numbering of the nodes and triangles','FontSize',12);
42
43 ntri = size(t,2);
44 np = size(p,2);
45
46 for j = 1 : ntri
47     tri = t(1:3,j);
48     poi = p(:,tri);
49     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;
50     text(adv(1),adv(2), num2str(j), 'FontSize',9);
51 end
52
53 for i = 1 : np
54     text(p(1,i),p(2,i), num2str(i), 'FontSize',9);
55 end
56
57 subplot(2,1,2);pdemesh(p,e,t);
58 title('numbering of the edges/midpoints','FontSize',12);
59
60 for j = 1 : ntri
61     tri = midtri(1:3,j);
62     poi = midpoints(:,tri);
63     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;
64     text(adv(1),adv(2), num2str(j), 'FontSize',9);
65 end
66
67 for i = 1 : length(midpoints)
68     text(midpoints(1,i),midpoints(2,i), num2str(i), 'FontSize',9);
69 end

```

D. Quellcode

```

65 % plot of the error and the error estimator:
66 figure(2);
67 subplot(2,1,1);
68 plot(1:recursion_depth,osc1_term,'o',1:recursion_depth,osc2_term,
69      '.*',1:recursion_depth,osc_term,'x');
70 ymin = min([min(osc_term),min(osc1_term),min(osc2_term)])-5;
71 ymax = max([max(osc_term),max(osc1_term),max(osc2_term)])+5;
72 axis([0.5,recursion_depth+0.5,ymin,ymax]);
73 legend('osc1','osc2','oscillation','location','best');

74 subplot(2,1,2);
75 plot(1:recursion_depth,J_error,'-o',...
76      1:recursion_depth,IQ_plot,'.*',1:recursion_depth,rhoS_plot,'-x');
77 ymin = min([min(J_error),min(IQ_plot),min(rhoS_plot)])-0.5;
78 ymax = max([max(J_error),max(IQ_plot),max(rhoS_plot)])+0.5;
79 axis([0.5,recursion_depth+0.5,ymin,ymax]);
80 legend('functional error','estimated error','error
81      indicator','location',...
82      'best');

83 % plot of the solution:
84 u_S = full(u_S);
85
86 figure(3);
87 pdeplot(p,e,t,'xydata',u_S,'zdata',u_S,'mesh','on','colormap',
88         'jet','colorbar','off');
89 title('solution of the obstacle problem','FontSize',15)
90
91 figure(4);
92 pdeplot(p,e,t,'zdata',u_S);
93 title('solution of the obstacle problem','FontSize',15)

```

Code D.19: Startdatei des Beispiels 6.1 für das Hindernisproblem

```

1 function [J_error,rhoS_plot,IQ_plot,degree_of_freedom,time] =
2     start_example2
3
4 clear
5 clear all
6 clc
7
8 % loading of the geometry data (non-constant boundary):
9 data = load('myshape.mat');
10 % loadfunction f (here constant):
11     function z = my_fun(x,y)
12         [m,n] = size(x);
13         z = zeros(m,n);
14         r = sqrt(x.^2+y.^2);
15         r_new = 2*r-1/2;
16
17         index_gamma1 = find(sqrt(x.^2+y.^2)>=1/4 &
18                             sqrt(x.^2+y.^2)<3/4 & y>=0);
19
20         z(index_gamma1) = -(r(index_gamma1)).^(2/3)...
21             .* sin(2/3*atan2(y(index_gamma1),x(index_gamma1)))...
22             .*(1./r(index_gamma1).*(-60*r_new(index_gamma1)).^4+120*...
23                 r_new(index_gamma1).^3-60*r_new(index_gamma1).^2)+(-480*...
24                 r_new(index_gamma1).^3+720*r_new(index_gamma1).^2-240*...
25                 r_new(index_gamma1))-4/3*r(index_gamma1).^(1/3).*(-60*...
26                 r_new(index_gamma1)).^4+120*r_new(index_gamma1).^3-60*...
27                 r_new(index_gamma1).^2).* sin(2/3*atan2(y(index_gamma1),...
28                     x(index_gamma1)));

```

```

26     index_gamma1 = find(sqrt(x.^2+y.^2)>=1/4 &
27                     sqrt(x.^2+y.^2)<3/4 & y<0);
28
29     z(index_gamma1) = -(r(index_gamma1)).^(2/3) ...
30             .* sin(2/3*(atan2(y(index_gamma1),x(index_gamma1))+2*pi)) ...
31             .* (1./r(index_gamma1).*(-60*r_new(index_gamma1).^4+120*...
32             r_new(index_gamma1).^3-60*r_new(index_gamma1).^2)+(-480*...
33             r_new(index_gamma1).^3+720*r_new(index_gamma1).^2-240*...
34             r_new(index_gamma1))-4/3*r(index_gamma1).^(1/3).*(-60*...
35             r_new(index_gamma1).^4+120*r_new(index_gamma1).^3-60*...
36             r_new(index_gamma1).^2).*sin(2/3*(atan2(y(index_gamma1),...
37             x(index_gamma1))+2*pi));
38
39     index_gamma2 = find(sqrt(x.^2+y.^2)>5/4);
40     z(index_gamma2) = -1;
41 end
42 fun @(x,y) my_fun(x,y);
43 % loading the exakt data for the given problem
44 J_u = -0.5;%-0.592278926839300071734;
45 % obstacle function:
46 my_obstacle = @(x,y) zeros(size(x));
47
48 % initializing the mesh:
49 h = 2;
50 [p,e,t]=initmesh(data.mygeomg,'Hmax',h,'jiggle','on','MesherVersion',...
51 'R2013a');
52 [p,e,t] = refinemesh(data.mygeomg,p,e,t);
53 [p,e,t] = refinemesh(data.mygeomg,p,e,t);
54
55 % initialization of the global values:
56 u_S = [];
57 itermax = 5;           % maximum iteration depth
58 nmax = 25000;          % maximum number of nodes
59 eps = 0.01;            % upper bound for the hierarchical error
60 theta_rho = 0.3;       % contraction parameter for local contributions
61 theta_rho = 0.3;       % of the error estimate
62 theta_osc = 0.3;       % contraction parameter for local contributions
63 theta_osc = 0.3;       % of the oscillations
64
65 tic
66 % adaptive algorithm:
67 [u_S,p,e,t,midtri,midpoints,rhoS_plot,IQ_plot,J_error,osc_term,
68 osc1_term,osc2_term,recursion_depth,degree_of_freedom,time] =
69 adaptive_refinement_solution(p,e,t,u_S,fun,my_obstacle,
70 data,J_u,eps,theta_rho,theta_osc,nmax,itermax);
71 toc
72
73 % plot of the mesh with the nodes and midpoints/edges:
74 figure(1)
75 subplot(2,1,1);pdemesh(p,e,t);
76 title('numbering of the nodes and triangles','FontSize',12);
77
78 ntri = size(t,2);
79 np = size(p,2);
80
81 for j = 1 : ntri
82     tri = t(1:3,j);
83     poi = p(:,tri);
84     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;

```

D. Quellcode

```

79     text (adv(1),adv(2), num2str(j), 'FontSize',9);
80 end
81
82 for i = 1 : np
83     text(p(1,i),p(2,i), num2str(i), 'FontSize',9);
84 end
85
86 subplot(2,1,2);pdemesh(p,e,t);
87 title('numbering of the edges/midpoints', 'FontSize',12);
88
89 for j = 1 : ntri
90     tri = midtri(1:3,j);
91     poi = midpoints(:,tri);
92     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;
93     text(adv(1),adv(2), num2str(j), 'FontSize',9);
94 end
95
96 for i = 1 : length(midpoints)
97     text(midpoints(1,i),midpoints(2,i), num2str(i), 'FontSize',9);
98 end
99
100
101 % plot of the error and the error estimator:
102 figure(2);
103 subplot(2,1,1);
104 plot(1:recursion_depth,osc1_term,'o',1:recursion_depth,osc2_term,
105      '-.*',1:recursion_depth,osc_term,'x');
106 ymin = min([min(osc_term),min(osc1_term),min(osc2_term)])-5;
107 ymax = max([max(osc_term),max(osc1_term),max(osc2_term)])+5;
108 axis([0.5,recursion_depth+0.5,ymin,ymax]);
109 legend('osc1','osc2','oscillation','location','best');
110
111 subplot(2,1,2);
112 plot(1:recursion_depth,J_error,'--o',...
113      1:recursion_depth,IQ_plot,'-.*',1:recursion_depth,rhoS_plot,'-x');
114 ymin = min([min(J_error),min(IQ_plot),min(rhoS_plot)])-0.02;
115 ymax = max([max(J_error),max(IQ_plot),max(rhoS_plot)])+0.02;
116 axis([0.5,recursion_depth+0.5,ymin,ymax]);
117 legend('functional error','estimated error','error
118 indicator','location',...
119 'best');
120
121 % plot of the solution:
122 u_S = full(u_S);
123
124 figure(3);
125 pdeplot(p,e,t,'xydata',u_S,'zdata',u_S,'mesh','on','colormap',
126         'jet','colorbar','off');
127 title('solution of the obstacle problem', 'FontSize',15)
128
129 figure(4);
130 pdeplot(p,e,t,'zdata',u_S);
131 title('solution of the obstacle problem', 'FontSize',15)
132
133 end

```

Code D.20: Startdatei des Beispiels 6.2 für das Hindernisproblem

```

1 function [ J_error , rhoS_plot , IQ_plot , degree_of_freedom , time ] =
2     start_example3
3
4 clear
5 clear all
6 clc
7
8 % loading of the geometry data (non-constant boundary):
9 data = load( 'mylshape.mat' );
10 % loadfunction f (here contant):
11     function z = my_fun(x,y)
12         [m,n] = size(x);
13         z = zeros(m,n);
14         r = sqrt(x.^2+y.^2);
15         r_new = 2*r - 1/2;
16
17         index_gamma1 = find(sqrt(x.^2+y.^2)>=1/4 &
18             sqrt(x.^2+y.^2)<3/4 & y>=0);
19
20         z(index_gamma1) = - (r(index_gamma1)).^(2/3) ...
21             .* sin(2/3*atan2(y(index_gamma1),x(index_gamma1))) ...
22             .* (1./r(index_gamma1).*(-60*r_new(index_gamma1)).^4+120*...
23                 r_new(index_gamma1).^3-60*r_new(index_gamma1).^2)+(-480*...
24                 r_new(index_gamma1).^3+720*r_new(index_gamma1).^2-240*...
25                 r_new(index_gamma1))-4/3*r(index_gamma1).^( -1/3).*(-60*...
26                 r_new(index_gamma1).^4+120*r_new(index_gamma1).^3-60*...
27                 r_new(index_gamma1).^2).*sin(2/3*atan2(y(index_gamma1),...
28                     x(index_gamma1)));
29
30         index_gamma1 = find(sqrt(x.^2+y.^2)>=1/4 &
31             sqrt(x.^2+y.^2)<3/4 & y<0);
32
33         z(index_gamma1) = - (r(index_gamma1)).^(2/3) ...
34             .* sin(2/3*(atan2(y(index_gamma1),x(index_gamma1))+2*pi)) ...
35             .* (1./r(index_gamma1).*(-60*r_new(index_gamma1)).^4+120*...
36                 r_new(index_gamma1).^3-60*r_new(index_gamma1).^2)+(-480*...
37                 r_new(index_gamma1).^3+720*r_new(index_gamma1).^2-240*...
38                 r_new(index_gamma1))-4/3*r(index_gamma1).^( -1/3).*(-60*...
39                 r_new(index_gamma1).^4+120*r_new(index_gamma1).^3-60*...
40                 r_new(index_gamma1).^2).*sin(2/3*(atan2(y(index_gamma1),...
41                     x(index_gamma1))+2*pi));
42
43         index_gamma2 = find(sqrt(x.^2+y.^2)>5/4);
44         z(index_gamma2) = - 1;
45
46         z = -z;
47     end
48 fun = @(x,y) my_fun(x,y);
49 % loading the exakt data for the given problem
50 J_u = -0.5;
51 % obstacle function:
52     function z = obs_fun(x,y)
53         [m,n] = size(x);
54         z = zeros(m,n);
55
56         for k = 1:m
57             for l = 1:n
58                 z(k,l) = - 0.5*(2.01-min([2-x(k,l);2+x(k,l);...
59                     2+y(k,l);2-y(k,l)]));
60             end
61         end
62     end

```

D. Quellcode

```

56      end
57 my_obstacle = @(x,y) obs_fun(x,y);
58
59 % initializing the mesh:
60 h = 2;
61 [p,e,t]=initmesh(data.mygeomg,'Hmax',h,'jiggle','on','MesherVersion',
62                   'R2013a');
63 [p,e,t] = refinemesh(data.mygeomg,p,e,t);
64 [p,e,t] = refinemesh(data.mygeomg,p,e,t);
65
66 % initialization of the global values:
67 u_S = [];
68 itermax = 7;           % maximum iteration depth
69 nmax = 1000;          % maximum number of nodes
70 eps = 0.001;          % upper bound for the hierarchical error
71           estimate
72 theta_rho = 0.3;       % contraction parameter for local contributions
73           of the error estimate
74 theta_osc = 0.3;       % contraction parameter for local contributions
75           of the oscillations
76
77 tic
78
79 % adaptive algorithm:
80 [u_S,p,e,t,midtri,midpoints,rhoS_plot,IQ_plot,J_error,osc_term,
81 osc1_term,osc2_term,recursion_depth,degree_of_freedom,time] =
82 adaptive_refinement_solution(p,e,t,u_S,fun,my_obstacle,
83 data,J_u,eps,theta_rho,theta_osc,nmax,itermax);
84 toc
85
86
87 % plot of the mesh with the nodes and midpoints/edges:
88 figure(1)
89 subplot(2,1,1);pdemesh(p,e,t);
90 title('numbering of the nodes and triangles','FontSize',12);
91
92 ntri = size(t,2);
93 np = size(p,2);
94
95 for j = 1 : ntri
96     tri = t(1:3,j);
97     poi = p(:,tri);
98     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;
99     text(adv(1),adv(2), num2str(j), 'FontSize',9);
100 end
101
102 for i = 1 : np
103     text(p(1,i),p(2,i), num2str(i), 'FontSize',9);
104 end
105
106 subplot(2,1,2);pdemesh(p,e,t);
107 title('numbering of the edges/midpoints','FontSize',12);
108
109 for j = 1 : ntri
110     tri = midtri(1:3,j);
111     poi = midpoints(:,tri);
112     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;
113     text(adv(1),adv(2), num2str(j), 'FontSize',9);
114 end
115
116 for i = 1 : length(midpoints)
117     text(midpoints(1,i),midpoints(2,i), num2str(i), 'FontSize',9);
118 end

```

```

111
112 % plot of the error and the error estimator:
113 figure(2);
114 subplot(2,1,1);
115 plot(1:recursion_depth,osc1_term,:o',1:recursion_depth,osc2_term,
116      '-.*',1:recursion_depth,osc_term,:x');
117 ymin = min([min(osc_term),min(osc1_term),min(osc2_term)]) - 5;
118 ymax = max([max(osc_term),max(osc1_term),max(osc2_term)]) + 5;
119 axis([0.5, recursion_depth+0.5,ymin,ymax]);
120 legend('osc1','osc2','oscillation','location','best');
121
122 subplot(2,1,2);
123 plot(1:recursion_depth,J_error,'--o',...
124      1:recursion_depth,IQ_plot,'-.*',1:recursion_depth,rhoS_plot,'-x');
125 ymin = min([min(J_error),min(IQ_plot),min(rhoS_plot)]) - 0.02;
126 ymax = max([max(J_error),max(IQ_plot),max(rhoS_plot)]) + 0.02;
127 axis([0.5, recursion_depth+0.5,ymin,ymax]);
128 legend('functional error','estimated error','error
129         indicator','location',...
130         'best');
131
132 % plot of the solution:
133 u_S = full(-u_S);
134
135 figure(8);
136 pdeplot(p,e,t,'xydata',u_S,'zdata',u_S,'mesh','on','colormap',
137          'jet','colorbar','off');
138 title('solution of the obstacle problem','FontSize',15)
139
140 figure(9);
141 pdeplot(p,e,t,'zdata',u_S);
142 title('solution of the obstacle problem','FontSize',15)
143
144 end

```

Code D.21: Startdatei des Beispiels 6.3 für das Hindernisproblem

D.2 Implementierung des Fehlerschätzers für das Kontaktproblem

Für den Quellcode des Kontaktproblems fügen wir nur die Dateien an, die verändert werden mussten für die Übertragung vom Hindernis- zum Kontaktproblem. Die detaillierte Beschreibung ist in Kapitel 5.2 zu finden.

```

1 function [A,f] = assemble(points,triangle,vol_load_x,vol_load_y,
2                           surf_load_x,surf_load_y,neumann_points,lambda,mu,num_of_nodes,
3                           option,u_S)
%ASSEMBLE evaluates the global matrix A and load vector f out of the
4 % given nodes, triangles, loadfunction, number of nodes and the
5 % Galerkin approximation u_S.
6
7 % ordering: triangles ↔ midpoints:
8 [midpoints,midtriangle] = midpoints_of_triangle(triangle,points);
9
10 % initialising the dimension and the solution:
11 np = size(points,2);
12 nt = size(triangle,2);

```

D. Quellcode

```

10 nmp = size(midpoints,2);
11
12 if nargin == 10
13     option = 'linear';
14     u_S = zeros(2*np,1);
15 end
16
17 if nargin == 11
18     u_S = zeros(2*np,1);
19 end
20
21 hat_edge = @(xi) [1/2.*(1-xi); 1/2.*(1+xi)];
22
23 % beginning of the assembling:
24 switch lower(option)
25     case {'linear'}
26         % linear hatfunctionen on the reference-element:
27         hat = @(xi,eta) [1-xi-eta; xi; eta];
28
29         % initialising of the global values:
30         A = sparse(2*np,2*np);
31         f = sparse(2*np,1);
32         my_tri = triangle(1:3,:);
33
34     case {'bubble', 'quadratic'}
35         % bubblefunctionen on the reference-element:
36         hat = @(xi,eta) [4*xi.*(1-xi-eta); 4*xi.*eta; 4*eta.*(1-xi-eta)];
37
38         % initialising of the global values:
39         A = sparse(2*nmp,2*nmp);
40         f = sparse(2*nmp,1);
41         my_tri = midtriangle;
42 end
43
44 % loop over the triangles for the assembling:
45 for i = 1:nt
46     poi = points(:,triangle(1:3,i));
47     u_S_loc_x = u_S(2*triangle(1:3,i)-1);
48     u_S_loc_y = u_S(2*triangle(1:3,i));
49     tri = my_tri(:,i);
50     common_boundary = intersect(neumann_points,triangle(1:3,i));
51     non_common_boundary = setdiff(triangle(1:3,i),common_boundary);
52
53     if any(poi(2,:)>0)
54         my_mu = mu(1);
55         my_lambda = lambda(1);
56     else
57         my_mu = mu(2);
58         my_lambda = lambda(2);
59     end
60
61     % evaluation of the local linear stiffness matrix and the
62     % Jacobian:
63     [S,f1,J] =
64         local_mat(poi,my_lambda,my_mu,u_S_loc_x,u_S_loc_y,option);
65
66     % Evaluating the local vector fl: (here is Gauss-quadrature over
67     % triangles used)
68     [wi,gauss,ansatz_values] = quad_tri(poi,hat,num_of_nodes);

```

```

69      f1(2*k-1) =
70          f1(2*k-1)+J*sum(wi.*vol_load_x(gauss(1,:),gauss(2,:)).*...
71              ansatz_values(k,:));
72      f1(2*k) =
73          f1(2*k)+J*sum(wi.*vol_load_y(gauss(1,:),gauss(2,:)).*...
74              ansatz_values(k,:));
75  end
76
77  if length(common_boundary) == 2
78      boundary_points = points(:,common_boundary);
79      [wi_edge, gauss_edge, J_edge, ansatz_val_edge] =
80          quad_edge(boundary_points, hat_edge,3);
81
82  switch find(triangle(1:3,i)==non_common_boundary)
83      case {1,3}
84          counter = 2;
85          for k = 1:3
86              if ismember(triangle(k,i),common_boundary)
87                  f1(2*k-1) = f1(2*k-1)+J_edge*sum(wi_edge.*...
88                      surf_load_x(gauss_edge(1,:),gauss_edge...
89                          (2,:)).*ansatz_val_edge(counter,:));
90                  f1(2*k) = f1(2*k)+J_edge*sum(wi_edge.*...
91                      surf_load_y(gauss_edge(1,:),gauss_edge...
92                          (2,:)).*ansatz_val_edge(counter,:));
93                  counter = counter-1;
94              end
95          end
96      case 2
97          counter = 1;
98          for k = 1:3;
99              if ismember(triangle(k,i),common_boundary)
100                  f1(2*k-1) = f1(2*k-1)+J_edge*sum(wi_edge.*...
101                      surf_load_x(gauss_edge(1,:),gauss_edge...
102                          (2,:)).*ansatz_val_edge(counter,:));
103                  f1(2*k) = f1(2*k)+J_edge*sum(wi_edge.*...
104                      surf_load_y(gauss_edge(1,:),gauss_edge...
105                          (2,:)).*ansatz_val_edge(counter,:));
106                  counter = counter+1;
107              end
108          end
109      end
110  end
111
112  % assembling into the global stiffness matrix A and the load
113  % vector f:
114  for k = 1:3
115      for l = 1:3
116          A(2*tri(k)-1:2*tri(k),2*tri(l)-1:2*tri(l)) =
117              A(2*tri(k)-1:2*tri(k),2*tri(l)-1:2*tri(l)) +
118              S(2*k-1:2*k,2*l-1:2*l);
119      end
120
121      f(2*tri(k)-1:2*tri(k)) = f(2*tri(k)-1:2*tri(k))+f1(2*k-1:2*k);
122  end
123
124 end
125
126 end

```

Code D.22: Assemblierung der Steifigkeitsmatrix und des Lastvektors für das Kontaktproblem

D. Quellcode

```

1 function rho_p =
2     eval_rho_p(nodes, triangles, edges, midpoints, midtri, u_S,
3     eps_V, vol_load_x, vol_load_y, surf_load_x, surf_load_y, neumann_ind,
4     lambda, mu)
5 %EVAL_RHO_P evaluates the local contribution rho_P of rho_S, given
6 % the nodes, triangles, midpoints of the edges (of the triangles)
7 % and die midpoints-triangle-ordering (midtri). Also it is given
8 % the solution auf die local defect problem eps_V and the function
9 % fun, which is part of the integral in rho_S. u_S are, as always,
10 % the Galerkin solution.
11
12 % Initializing :
13 np = size(nodes,2);
14 rho_p = zeros(np,1);
15
16 % Hat- and Bubble- functions :
17 phi_P = @(xi,eta) [1-xi-eta; xi; eta];
18 phi_E = @(xi,eta) [4*xi.*(1-xi-eta); 4*xi.*eta; 4*eta.*(1-xi-eta)];
19 hat_edge = @(xi) [1/2.*(1-xi); 1/2.*(1+xi)];
20
21 % Evaluation of the weights and values of the function for the
22 % surface integral:
23 [wi,~,phi_E_values] = quad_tri([0,1,0;0,0,1],phi_E,7);
24 [~,~,phi_P_values] = quad_tri([0,1,0;0,0,1],phi_P,7);
25
26 for k = 1:np
27     % Ordering triangle ↔ reference function and support of phi_P:
28     [phi_p_local,w_p] = find(triangles(1:3,:)==k);
29     % indices of the edges in the considered triangle:
30     E_index = midtri(:,w_p);
31
32     % Evaluation of the surface integral of rho_p over w_p:
33     for j = 1:length(w_p)
34         % points of the considered triangle with Jacobi determinant:
35         mypoi = nodes(:,triangles(1:3,w_p(j)));
36         x = mypoi(1,:);
37         y = mypoi(2,:);
38         J = (x(2)-x(1))*(y(3)-y(1))-(x(3)-x(1))*(y(2)-y(1));
39         % Gauss-points and local contributions of eps_V:
40         eps_V_local_x = eps_V(2*E_index(:,j)-1);
41         eps_V_local_y = eps_V(2*E_index(:,j));
42         [~,gauss,~] = quad_tri(mypoi,@(x,y) 0,7);
43         % the functionvalues of the considered local hatfunction
44         phi_P =
45         phi_P_values = phi_P_values(phi_p_local(j),:);
46         % evaluation of the first integral of rho_p:
47         rho_p(k) = rho_p(k) + J *
48             (sum(wi.*vol_load_x(gauss(1,:),gauss(2,:)).*...
49                 (eps_V_local_x'*phi_E_values).*(phi_P_values)) +
50                 sum(wi.*vol_load_y(gauss(1,:),gauss(2,:)).*...
51                     (eps_V_local_y'*phi_E_values).*(phi_P_values)));
52     end
53
54     % Evaluation of the boundary integral over gamma_sigma:
55     if ismember(nodes(k),neumann.ind)
56         help_ind1 = ismember(w_p,neumann.ind);
57         help_ind2 = sum(help_ind1);
58         for j = 1:length(w_p)
59             if help_ind2(j) == 2
60                 local_point_index = find(help_ind1);
61             end
62         end
63     end
64 end

```

```

49     global_point_index =
50         triangles(local_point_index,w_p(j));
51     my_poi = nodes(:,global_point_index);
52     [wi_edge, gauss_edge, J_edge, ansatz_val_edge] =
53         quad_edge(my_poi, hat_edge, 3);
54     edge_ind = find(prod(ismember(edges(1:2,:), ...
55         global_point_index)));
56
57     if edges(1,edge_ind) == k
58         rho_p(k) =
59             rho_p(k)+J_edge*(sum(wi_edge.*surf_load_x
60             (gauss_edge(1,:),gauss_edge(2,:)).*...
61             ansatz_val_edge(1,:)) +
62             sum(wi_edge.*surf_load_y
63             (gauss_edge(1,:),gauss_edge(2,:)).*...
64             ansatz_val_edge(1,:)));
65     else
66         rho_p(k) =
67             rho_p(k)+J_edge*(sum(wi_edge.*surf_load_x
68             (gauss_edge(1,:),gauss_edge(2,:)).*...
69             ansatz_val_edge(2,:)) +
70             sum(wi_edge.*surf_load_y
71             (gauss_edge(1,:),gauss_edge(2,:)).*...
72             ansatz_val_edge(2,:)));
73     end
74   end
75 end
76
77 % Evaluation of the line integral over the edges E\in E_p:
78 % Evaluaton of the set E_p of the edges , which contain the
79 % point P:
80 [~,E_p] = find(midpoints(3:4,:)==k);
81 flag = midpoints(2,:);
82 % Evaluation of the normal-fluxes j_E for all edges in E_p:
83 j_E =
84   normal_flux(E_p,flag,nodes,triangles,midpoints,midtri,u_S,
85   lambda,mu);
86
87 for i = 1:length(E_p)
88   % evaluating the points of the edges:
89   edge_poi_ind = midpoints(3:4,E_p(i));
90   edge_poi = nodes(:,edge_poi_ind);
91
92   % affin transformation of the local integral on the
93   % global edge by multiplication of the jacobian:
94   laenge = norm(edge_poi(:,1)-edge_poi(:,2));
95   global_phiPE_int = 1/3*laenge;
96
97   % determination of the functionvalues of eps_V(x_E):
98   epsV_loc = eps_V([2*E_p(i)-1,2*E_p(i)]);
99
100   % adding the line integral to the local contribution
101   rho_p:
102   rho_p(k) = rho_p(k)+(j_E(:,i)'*epsV_loc)*global_phiPE_int;
103 end
104 end
105 end

```

Code D.23: Berechnung der lokalen Anteile von ρ_S für das Kontaktproblem

D. Quellcode

```

1  function [S, f_local, J] =
2    local_mat(points, lambda, mu, uS_local_x, uS_local_y, option)
%LOCALMAT computes the Jacobian J, a local stiffness matrix S and if
%option='bubble' a local vector f_local, which will be needed for
%a(u_S,*) in rho_S. LOCALMAT expects the nodes (points) of the
%local triangle, the z-values uS_local of this nodes and an
%option, which can be 'linear' or 'bubble.
3
4 % initialization of f_local:
5 f_local = zeros(6,1);
6
7 % x- and y-values of the nodes:
8 x = points(1,:);
9 y = points(2,:);
10
11 % the Jacobian J and other factors for the affine transformation from
% the reference element onto a arbitrary triangle:
12 J = (x(2)-x(1))*(y(3)-y(1)) - (x(3)-x(1))*(y(2)-y(1));
13 a = 1/J * (mu*(x(3)-x(1))^2 + (2*mu+lambda)*(y(3)-y(1))^2);
14 b = -1/J * (mu*(x(3)-x(1))*(x(2)-x(1)) +
% (2*mu+lambda)*(y(3)-y(1))*(y(2)-y(1)));
15 c = -1/J * (mu*(y(3)-y(1))*(y(2)-y(1)) +
% (2*mu+lambda)*(x(3)-x(1))*(x(2)-x(1)));
16 d = 1/J * (mu*(x(3)-x(1))*(y(2)-y(1)) +
% lambda*(y(3)-y(1))*(x(2)-x(1)));
17 e = -1/J * (mu+lambda)*(x(3)-x(1))*(y(3)-y(1));
18 f = 1/J * (mu*(x(2)-x(1))^2 + (2*mu+lambda)*(y(2)-y(1))^2);
19 g = 1/J * (mu*(y(3)-y(1))^2 + (2*mu+lambda)*(x(3)-x(1))^2);
20 h = 1/J * (mu*(x(2)-x(1))*(y(3)-y(1)) +
% lambda*(x(3)-x(1))*(y(2)-y(1)));
21 i = -1/J * (mu+lambda)*(x(2)-x(1))*(y(2)-y(1));
22 j = 1/J * (mu*(y(2)-y(1))^2 + (2*mu+lambda)*(x(2)-x(1))^2);
23
24 switch lower(option)
25   case {'linear'}
26     % local stiffness matrix for linear shape functions:
27     S = 1/2 * [ a+2*b+f, d+e+h+i, -a-b, -e-h, -b-f, -d-i;
28                 d+e+h+i, 2*c+g+j, -d-e, -c-g, -h-i, -c-j;
29                 -a-b, -d-e, a, e, b, d;
30                 -e-h, -c-g, e, g, h, c;
31                 -b-f, -h-i, b, h, f, i;
32                 -d-i, -c-j, d, c, i, j ];
33
34   case {'bubble'}
35     % local stiffness matrix for quadratic shape functions:
36     S = 4/3* diag([a+b+f, c+g+j, a+b+f, c+g+j, a+b+f, c+g+j]);
37
38     % gradient of u_S:
39     gradu_S_x = gradu(points, uS_local_x);
40     gradu_S_y = gradu(points, uS_local_y);
41     % transformation onto the reference element:
42     gradu_S_x = [x(2)-x(1), y(2)-y(1); x(3)-x(1),
43                  y(3)-y(1)]*gradu_S_x';
44     gradu_S_y = [x(2)-x(1), y(2)-y(1); x(3)-x(1),
45                  y(3)-y(1)]*gradu_S_y';
46     gradu_S = [gradu_S_x; gradu_S_y];
47     % f_local:
48     f_local = -(([a, b, e, d]*gradu_S).*[0; 0; 2/3; 0; -2/3; 0] +
% ([b, f, h, i]*gradu_S).*[-2/3; 0; 2/3; 0; 0; 0]...
% +([e, h, g, c]*gradu_S).*[0; 0; 0; 2/3; 0; -2/3] + ([d, i, c,
% j]*gradu_S).*[0; -2/3; 0; 2/3; 0; 0]);

```

```

48     otherwise
49         error('Unknown option');
50     end
51 end
52 end

```

Code D.24: Berechnung der lokalen Steifigkeitsmatrix für das Kontaktproblem

```

1 function j_E = normal_flux(E_p,given_flag, nodes, triangles, edges,
2   edge_triangles, uS_values, lambda, mu)
%NORMALFLUX computes the normal flux for all given edges E \in E_p.
% The other given objects are: The nodes of the triangulation, the
% indices of the triangles-nodes ordering, the
% midpoints/edges-matrix edges, the edges-triangle-ordering
% edge_triangle and the functionvalues of the Galerkin solution
% uS_values.
3
4 % Initializing:
5 j_E = zeros(2, size(E_p));
6
7 % Evaluate the neighbours of all edges in the edges-set E_p:
8 [neighbours, flag] = neighbourhood(E_p, edge_triangles, 'edges');
9
10 % Computation of the single normal fluxes:
11 for k = 1:length(E_p)
12     % Evaluation of the edge-points:
13     edge_poi_ind = edges(3:4,E_p(k));
14     edge_poi = nodes(:,edge_poi_ind);
15
16     % Calculation of the gradient of u_S on T_1 and T_2:
17     neigh_tri_ind = neighbours(:,k);
18     neigh_tri = triangles(1:3,neigh_tri_ind);
19     p_T = nodes(:,neigh_tri);
20     p_T1 = p_T(:,1:3);
21     p_T2 = p_T(:,4:6);
22
23     % The calculated u_S values onto T_1 and T_2, where the first
24     % three rows are x-coordinates and the last three the
25     % y-coordinates:
26     uS_T1 = uS_values([2*neigh_tri(:,1)-1,2*neigh_tri(:,1)]);
27
28     if flag(k) == 0
29         uS_T2 = zeros(6,1);
30     else
31         uS_T2 = uS_values([2*neigh_tri(:,2)-1,2*neigh_tri(:,2)]);
32     end
33
34     graduS_T_x = [gradu(p_T1,uS_T1(1:3));gradu(p_T2,uS_T2(1:3))];
35     graduS_T_y = [gradu(p_T1,uS_T1(4:6));gradu(p_T2,uS_T2(4:6))];
36
37     % the normalvector from T_1 to T_2:
38     connect = edge_poi(:,2)-edge_poi(:,1);
39     orth_connect = [-connect(2);connect(1)];
40     n = 1/norm(orth_connect)*orth_connect;
41
42     % Verification, if n shows from T_1 to T_2:
43     p3_T1 = setdiff(p_T1',edge_poi','rows');
44     edge_test = (p3_T1-edge_poi(:,1))';

```

D. Quellcode

```

44 if edge-test*n > 0
45     n = -n;
46 end
47
48 strain1 = [ graduS_T_x(1,1), 1/2*(graduS_T_x(1,2)+graduS_T_y(1,1));
49             1/2*(graduS_T_x(1,2)+graduS_T_y(1,1)), graduS_T_y(1,2)];
50 strain2 = [ graduS_T_x(2,1), 1/2*(graduS_T_x(2,2)+graduS_T_y(2,1));
51             1/2*(graduS_T_x(2,2)+graduS_T_y(2,1)), graduS_T_y(2,2)];
52
53 if (given_flag(k) > 0)
54     stress1 = 2*mu(1)*strain1 + lambda(1)*trace(strain1)*eye(2);
55     stress2 = 2*mu(1)*strain2 + lambda(1)*trace(strain2)*eye(2);
56 else
57     stress1 = 2*mu(2)*strain1 + lambda(2)*trace(strain1)*eye(2);
58     stress2 = 2*mu(2)*strain2 + lambda(2)*trace(strain2)*eye(2);
59 end
60 j_E(:,k) = stress2*n - stress1*n;
61 end
62
63 end

```

Code D.25: Bestimmung des Normalenspannungsflusses über eine Kante E

```

1 function [wi,gauss,J,ansatz_values] =
2 quad_edge(points,ansatz_fun,num_of_nodes)
%QUAD_EDGE computes the weights , Gauss-points and functionvalues of
the shape functions , which are needed for the quadrature over an
edge of a triangle. The matrix points stores the x-,y-values of
the nodes of the edge , ansatz_fun a vector of shape functions and
num_of_nodes the number of nodes for the quadrature formula.
3
% x- and y-values of the points:
4 x = points(1,:);
5 y = points(2,:);
6
7 switch num_of_nodes
% determination of the weights and the local points xi,eta:
8 case 1
9     wi = 2;
10    local_poi = 0;
11
12 case 2
13     wi = [1,1];
14     local_poi = [-sqrt(1/3),sqrt(1/3)];
15
16 case 3
17     wi = [5/9,8/9,5/9];
18     local_poi = [-sqrt(3/5),0,sqrt(3/5)];
19
20 case 4
21     wi = [18-sqrt(30),18+sqrt(30),18+sqrt(30),18-sqrt(30)]/36;
22     local_poi =
23         [-sqrt(3/7+2/7*sqrt(6/5)), -sqrt(3/7-2/7*sqrt(6/5)),
24          sqrt(3/7-2/7*sqrt(6/5)),sqrt(3/7+2/7*sqrt(6/5))];
25
26 case 5
27     wi = [(322-13*sqrt(70))/900,(322+13*sqrt(70))/900,128/225,
28             (322+13*sqrt(70))/900,(322-13*sqrt(70))/900];
29     local_poi =
30         [-1/3*sqrt(5+2*sqrt(10/7)), -1/3*sqrt(5-2*sqrt(10/7)),
31          0,1/3*sqrt(5-2*sqrt(10/7)),1/3*sqrt(5+2*sqrt(10/7))];
32
33 end

```

```

29     otherwise
30         error('keine Quadraturformel bekannt');
31     end
32
33 % evaluation of the Gauss-points by using xi and eta with the points
34 % x and y:
35 l = sqrt((x(2)-x(1))^2+(y(2)-y(1))^2);
36 parameter = 1/2.*local_poi+l/2;
37 gauss(1,:) = x(1) + parameter.* (x(2)-x(1));
38 gauss(2,:) = y(1) + parameter.* (y(2)-y(1));
39
40 % evaluating the jacobian:
41 J = 1/2;
42
43 % computing the functionvalues of the shape functions in the local
44 % coordinates:
45 ansatz_values = ansatz_fun(local_poi);
46
47 end

```

Code D.26: Gewichte und Stützstellen für die Quadratur über eine Kante

```

1 function
2     [u_S ,points ,edges ,triangles ,midtri ,midpoints ,rhoS_plot ,IQ_plot ,
3      J_error ,osc_term ,osc1_term ,osc2_term ,recursion_depth ,
4      degree_of_freedom ,time_vec] = adaptive_refinement_solution
5     (points ,edges ,triangles ,lambda ,mu ,solution ,vol_load_fun_x ,
6      vol_load_fun_y ,surf_load_fun_x ,surf_load_fun_y ,obstacle ,geo_data ,
7      J_exact ,max_error ,para_rho ,para_osc ,max_points ,max_recursion)
8 %ADAPTIVE_REFINEMENT SOLUTION uses the adaptive refinement strategy
9 % shown
10 %in chapter 4 and evaluates the solution on a adaptive refined mesh.
11
12 % initializing all local values:
13 u_S = solution;
14 J_u = J_exact;
15 rhoS_plot = zeros(max_recursion,1);
16 IQ_plot = zeros(max_recursion,1);
17 J_error = zeros(max_recursion,1);
18 osc_term = zeros(max_recursion,1);
19 osc1_term = zeros(max_recursion,1);
20 osc2_term = zeros(max_recursion,1);
21 recursion_depth = 1;
22 time_vec = zeros(max_recursion,1);
23 degree_of_freedom = zeros(max_recursion,1);
24
25 while 1
26     tic
27     % further initializations:
28     np = size(points ,2);
29     degree_of_freedom(recursion_depth) = np;
30
31     % evaluate the boundary indices and conditions:
32     [~,~,H,~]=assemb(geo_data.mygeomb,points ,edges);
33     [~,boundary_ind] = find(H(:,1:np));
34     [gamma_D,gamma_N,gamma_C] =
35         boundary_disjunction(boundary_ind ,points );
36
37     num_of_dirichlet_points = length(gamma_D);
38     adjacency_D = sparse(2*num_of_dirichlet_points ,2*np);

```

D. Quellcode

```

31   for i = 1:num_of_dirichlet_points
32     adjacency_D(2*i-1:2*i,2*gamma_D(i)-1:2*gamma_D(i)) = eye(2);
33   end
34   dirichlet_bound = sparse(2*(num_of_dirichlet_points),1);
35
36   num_of_contact_points = length(gamma_C);
37   B = sparse(num_of_contact_points,2*np);
38   % for simplicity just n = (0,1):
39   for j = 1:num_of_contact_points
40     if points(2,gamma_C(j))>0
41       B(j,2*gamma_C(j)-1:2*gamma_C(j)) = [0,-1];
42     else
43       B(j,2*gamma_C(j)-1:2*gamma_C(j)) = [0,1];
44     end
45   end
46
47   % assembling of the matrix/vector data and the evaluation of the
48   % Dirichlet boundary data:
49   [A,f] = assemble(points,triangles,vol_load_fun_x,vol_load_fun_y,
50                     surf_load_fun_x,surf_load_fun_y,gamma_N,lambda,mu,7,'linear');
51
52   % evaluation of the midpoints:
53   [midpoints,midtri] = midpoints_of_triangle(triangles,points);
54   nmp = size(midpoints,2);
55
56   % computation of the functionvalues of the obstacle onto the mesh
57   % nodes and midpoints:
58   z_obs_prob = obstacle(points(1,gamma_C),points(2,gamma_C));
59
60   contact_midpoints = zeros(5,num_of_contact_points-1);
61   contact_count = 1;
62   for k = 1:nmp
63     if (ismember(midpoints(3,k),gamma_C) &&
64         ismember(midpoints(4,k),gamma_C))
65       contact_midpoints(1:4,contact_count) = midpoints(:,k);
66       contact_midpoints(5,contact_count) = k;
67       contact_count = contact_count+1;
68     end
69   end
70   z_obs_midpoints =
71     obstacle(contact_midpoints(1,:),contact_midpoints(2,:)); % hier fehlt noch neue Zuordnung!!!
72
73   if size(z_obs_prob,1) == 1
74     z_obs_prob = z_obs_prob';
75     z_obs_midpoints = z_obs_midpoints';
76   end
77
78   % solution of the variational inequality with
79   % active-set/inner-points-method:
80   u_S = sparse(u_S);
81   z_obs_prob = sparse(z_obs_prob);
82   opts = optimset('Algorithm','interior-point-convex','LargeScale',
83                  'on','Display','off');
84   [u_S,J_uS] =
85     quadprog(A,-f,B,z_obs_prob,adjacency_D,dirichlet_bound,
86              [],[],u_S,opts);
87   point_shift = zeros(2,np);
88   point_shift(1:2*np) = u_S;
89
90   new_points = points+point_shift;
91   figure(2)
92

```

```

83 pdeplot(new-points ,edges ,triangles)
84
85 % computing the functionvalues of u_S onto the midpoints:
86 u_S_mid = zeros(num_of_contact_points-1,1);
87
88 for j = 1 : num_of_contact_points-1
89     index = contact_midpoints([3,4],j);
90
91     if contact_midpoints(2,j)>0
92         u_S_mid(j) = [(u_S(2*index(1)-1)+u_S(2*index(2)-1))/2,
93                     (u_S(2*index(1))+u_S(2*index(2)))/2]*[0; -1/2];
94     else
95         u_S_mid(j) = [(u_S(2*index(1)-1)+u_S(2*index(2)-1))/2,
96                     (u_S(2*index(1))+u_S(2*index(2)))/2]*[0; 1/2];
97     end
98 end
99
100 % the solution of the local defect problem by assembling the
101 % matrix with the bubble functions and using the equations in
102 % (4.10) and (4.11):
103 [A_Q,rhoS_phiE] =
104 assemble(points,triangles,vol_load_fun_x,vol_load_fun_y,
105 surf_load_fun_x,surf_load_fun_y,gamma_N,lambda,mu,7,'bubble',
106 u_S);
107 %[eps_V,rho_E,d_E,~] = defect_problem_solution(points,triangles,
108 %midtri,rhoS_phiE,u_S_mid,z_obs_midpoints);
109 B_Q = sparse(num_of_contact_points-1,2*nmp);
110 % for simplicity just n = (0,1):
111 for j = 1:num_of_contact_points-1
112     if contact_midpoints(2,j)>0
113         B_Q(j,2*contact_midpoints(5,j)-1:2*contact_midpoints(5,j))
114             = [0,-1/2];
115     else
116         B_Q(j,2*contact_midpoints(5,j)-1:2*contact_midpoints(5,j))
117             = [0,1/2];
118     end
119 end
120 length(z_obs_midpoints)
121 length(u_S_mid)
122 [eps_V,~] = quadprog(A_Q,-rhoS_phiE,B_Q,z_obs_midpoints+u_S_mid,
123 [],[],[],[],[],opts);
124
125 % the hierarchical error estimator: evaluation of rho_S(eps_V)
126 % with the equation beyond (4.68):
127 rhoS_glob = eps_V'*rhoS_phiE;
128 rhoS_plot(recursion_depth) = rhoS_glob;
129
130 % computation of -I_Q(eps_V) with (4.6):
131 IQ_plot(recursion_depth) = -1/2*(eps_V.^2)*diag(A_Q) + rhoS_glob;
132
133 % calculation of the error of -I(e):
134 J_error(recursion_depth) = J_uS-J_u;
135
136 % evaluating the local contributions of rho_S with Lemma 4.14:
137 rho_p = eval_rho_p(points,triangles,edges,midpoints,midtri,u_S,
138 eps_V,vol_load_fun_x,
139 vol_load_fun_y,surf_load_fun_x,surf_load_fun_y,
140 gamma_N,lambda,mu);
141
142 % rho_p
143 % find(rho_p>1)
144 % points(:,find(rho_p>1))

```

D. Quellcode

```

130    sum(rho_p)
131    rhoS_glob
132    % determination of the sets N0, N0+, N+, N++, N0- for the
133    % oscillationterms:
134    % inner_points_omega = inner_points(H);
135    % N0_set = N0(u_S,inner_points_omega,z_obs_prob);
136    % Nplus_set = Nplus(N0_set,inner_points_omega,points);
137    % [N0plus_set,N0minus_set] = N0plusminus(N0_set,points,
138    % triangles,midpoints,midtri,load_fun,obstacle,u_S);
139    % Nplusplus_set = Nplusplus(Nplus_set,midpoints,rho_E,d_E);
140    %
141    % evaluation of the oscillationterms:
142    % [osc1_term(recursion_depth),osc1_local] =
143    % osc1(N0plus_set,z_obs_prob,points,triangles,u_S);
144    % [osc2_term(recursion_depth),osc2_local] =
145    % osc2(Nplusplus_set,N0minus_set,points,
146    % triangles,midpoints,load_fun);
147    % osc_local = osc1_local + osc2_local;
148    % osc_term(recursion_depth) = sqrt(osc1_term(recursion_depth)^2 +
149    % osc2_term(recursion_depth)^2);
150
151    % calculating the indices of the triangles, which have to be
152    % refined:
153    refine_triangle = find_triangle_refinement(rho_p,rhoS_glob,
154        [],[],triangles,para_rho,0);
155
156    % refinement of the mesh:
157    [p_h,e_h,t_h,uS_h] = refinemesh(geo_data.mygeomg,points,edges,
158        triangles,u_S,refine_triangle);
159
160    % termination criterion: if the number of the nodes is too large,
161    % no triangle will be refined or the hierarchical error
162    % estimator is small enough:
163    if isempty(refine_triangle) || rhoS_glob < max_error ||
164        recursion_depth == max_recursion || length(p_h) > max_points
165        length(p_h)
166        fprintf('%s %f.\n', 'Die Rekursionstiefe ist ',
167            recursion_depth);
168        break;
169    else
170        points = p_h;
171        edges = e_h;
172        triangles = t_h;
173        u_S = uS_h;
174        recursion_depth = recursion_depth + 1;
175    end
176    time_vec(recursion_depth) = toc;
177 end
178
179 % elimination of the zeros in the vectors of the error/-estimator:
180 rhoS_plot = rhoS_plot(1:recursion_depth);
181 IQ_plot = IQ_plot(1:recursion_depth);
182 J_error = J_uS;%J_error(1:recursion_depth);
183 osc_term = osc_term(1:recursion_depth);
184 osc1_term = osc1_term(1:recursion_depth);
185 osc2_term = osc2_term(1:recursion_depth);
186 degree_of_freedom = degree_of_freedom(1:recursion_depth);
187 end

```

Code D.27: Adaptive Verfeinerung des Gitters und Lösung des Kontaktproblems

```

1 function start_example4
2
3 clear
4 clear all
5 clc
6
7 % loading of the geometry data (non-constant boundary):
8 data = load('mycylinder.mat');
9 % loadfunction f (here contant):
10 function z = load1(x,y)
11     z = zeros(size(x));
12 end
13 vol_load_x = @(x,y) load1(x,y);
14
15 function z = load2(x,y)
16     z = zeros(size(x));
17     index = find(y >= 0);
18
19     z(index) = -10*ones(size(index));
20 end
21 vol_load_y = @(x,y) load2(x,y);
22
23 function z = load3(x,y)
24     z = zeros(size(y));
25
26     index = find(x>- 0.1 & x<0.1);
27     z(index) = -1000*ones(size(index));
28 end
29
30 surf_load_x = @(x,y) load1(x,y);
31 surf_load_y = @(x,y) load3(x,y);
32 % loading the exakt data for the given problem:
33 % data_exact = load('fval_log_u.mat');
34 % J_u = data_exact.fval;
35 J_u = 0;
36 % obstacle function:
37 function z = my_obs(x,y)
38     z = 1-sqrt(1.001-x.^2);
39 %     z = zeros(size(y));
40 %
41 %     index1 = find(y>=1-1/sqrt(2) & y<=1 & x<0);
42 %     z(index1) = sqrt((x(index1)+1).^2+y(index1).^2);
43 %
44 %     index2 = find(y>=1-1/sqrt(2) & y<=1 & x>=0);
45 %     z(index2) = sqrt((x(index2)-1).^2+y(index2).^2);
46 %
47 %     index3 = find(y<1-1/sqrt(2) & y>0);
48 %     z(index3) = y(index3)./(1-y(index3));
49 %
50 %     index4 = find(y==0);
51 %     z(index4) = sqrt(x(index4).^2+1)- 1;
52 end
53 my_obstacle = @(x,y) my_obs(x,y);
54
55 % mechanical values:
56 E = [7000,10^6];
57 nu = [0.3,0.45];
58 lambda = nu.*E./((1+nu).*(1-2*nu));
59 mu = E./(2.*(1+nu));
60
61 % initializing the mesh:

```

D. Quellcode

```

62 h = 0.05;
63 [p,e,t] = initmesh(data.mygeomg, 'Hmax',h, 'MesherVersion', 'R2013a');
64
65 % plot of the mesh with the nodes and midpoints/edges:
66 figure(1)
67 %subplot(2,1,1);
68 pdemesh(p,e,t);
69 title('numbering of the nodes and triangles', 'FontSize',12);
70
71 ntri = size(t,2);
72 np = size(p,2);
73
74 for j = 1 : ntri
75     tri = t(1:3,j);
76     poi = p(:,tri);
77     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;
78     text(adv(1),adv(2), num2str(j), 'FontSize',9);
79 end
80
81 for i = 1 : np
82     text(p(1,i),p(2,i), num2str(i), 'FontSize',9);
83 end
84
85
86 % initialization of the global values:
87 u_S = [];
88 itermax = 1;           % maximum iteration depth
89 nmax = 1000000;        % maximum number of nodes
90 eps = 0.001;           % upper bound for the hierarchical error
91 estimate
92 theta_rho = 0.7;       % contraction parameter for local contributions
93          % of the error estimate
93 theta_osc = 0.3;       % contraction parameter for local contributions
93          % of the oscillations
94
95 tic
96 % adaptive algorithm:
97 [u_S,p,e,t,midtri,midpoints,rhoS_plot,IQ_plot,J_error,osc_term,
97  osc1_term,osc2_term,recursion_depth, degree_of_freedom] =
97  adaptive_refinement_solution(p,e,t,lambda,mu,u_S,vol_load_x,
97  vol_load_y,surf_load_x,surf_load_y,my_obstacle,
97  data,J_u,eps,theta_rho,theta_osc,nmax,itermax);
98 toc
99
100 subplot(2,1,2);pdemesh(p,e,t);
101 title('numbering of the edges/midpoints', 'FontSize',12);
102
103 for j = 1 : ntri
104     tri = midtri(1:3,j);
105     poi = midpoints(:,tri);
106     adv = (poi(:,1)+poi(:,2)+poi(:,3))/3;
107     text(adv(1),adv(2), num2str(j), 'FontSize',9);
108 end
109
110 for i = 1 : length(midpoints)
111     text(midpoints(1,i),midpoints(2,i), num2str(i), 'FontSize',9);
112 end
113
114 % plot of the error and the error estimator:
115 figure(2);
116 subplot(2,1,1);

```

```

117 plot(1:recursion_depth,osc1_term,:o',1:recursion_depth,osc2_term,
      '-.*',1:recursion_depth,osc_term,:x');
118 ymin = min([min(osc_term),min(osc1_term),min(osc2_term)])-5;
119 ymax = max([max(osc_term),max(osc1_term),max(osc2_term)])+5;
120 axis([0.5,recursion_depth+0.5,ymin,ymax]);
121 legend('osc1','osc2','oscillation','location','best');
122
123 subplot(2,1,2);
124 plot(1:recursion_depth,J_error,'-o',...
125      1:recursion_depth,IQ_plot,'-.*',1:recursion_depth,rhoS_plot,'-x');
126 ymin = min([min(J_error),min(IQ_plot),min(rhoS_plot)])-0.5;
127 ymax = max([max(J_error),max(IQ_plot),max(rhoS_plot)])+0.5;
128 axis([0.5,recursion_depth+0.5,ymin,ymax]);
129 legend('functional error','estimated error','error
      indicator','location',...
      'best');
130
131 % plot of the solution:
132 u_S = full(u_S);
133
134 figure(3);
135 pdeplot(p,e,t,'xydata',u_S,'zdata',u_S,'mesh','on','colormap','jet',...
      'colorbar','off');
136 %title('solution of the obstacle problem','FontSize',15)
137
138 figure(4);
139 pdeplot(p,e,t,'zdata',u_S);
140 %title('solution of the obstacle problem','FontSize',15)
141
142 end

```

Code D.28: Startdatei des Beispiels ?? für das Kontaktproblem

Selbstständigkeitserklärung

Hiermit erkläre ich, Cornelius Rüther, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Stellen der Arbeit, die wörtlich oder sinngemäß aus Veröffentlichungen oder aus anderweitigen fremden Äußerungen entnommen wurden, sind als solche kenntlich gemacht. Ferner erkläre ich, dass die Arbeit noch nicht in einem anderen Studiengang als Prüfungsleistung verwendet wurde.

Hannover, den 4. Dezember 2014

Unterschrift