

**h-adaptives Verfahren für ebene finite
Elemente basierend auf dem
Fehlerschätzer nach Zienkiewicz und
Zhu**

Michael Hammer

Institut für Festigkeitslehre

26. Februar 2007

Vorwort

Die vorliegende Arbeit soll einen Einstieg in die adaptive linear elastische Finite Elemente Methode darstellen. Sie ist im Rahmen meines Diplomstudiums an der Technischen Universität Graz am Institut für Festigkeitslehre entstanden und wurde als Diplomarbeit eingereicht. Ich möchte mich an dieser Stelle bei O.Univ.-Prof. Dipl.-Ing. Dr.techn. Christian Celigoj für seine Unterstützung sowohl in fachlich theoretischen als auch praktischen Belangen bedanken. Darüber hinaus gilt mein Dank auch den Assistenten und Arbeitskollegen Dipl. Ing. Stefan Sollerer, Dipl. Ing. Manfred Ulz die mir geduldig mit Rat zur Seite standen.

Ganz besonderer Dank sei aber meiner Familie - insbesondere meinen Eltern Karoline und Erich Hammer - ausgesprochen, die mich in meinem Bestreben zu studieren stets ermutigt und nicht zu letzt das Studium finanziert haben.

Michael Hammer, Graz am 26. Februar 2007

Kurzfassung

Für eine Finite-Elemente-Berechnung ist eine geeignete Diskretisierung (Netz) des zu untersuchenden Körpers von zentraler Bedeutung. Numerische Ergebnisse können u.a. durch Netzverfeinerung (h-Adaptivität) verbessert werden. In dieser Arbeit wird ein a-posteriori-Verfahren vorgestellt, das auf einem Fehlerschätzer nach Zienkiewicz und Zhu beruht. Durch rekursive Anwendung - Teilung von Dreieckselementen (1. und 2. Ordnung) - wird ein Netz mit homogener Fehlerverteilung und maximalem globalen Fehler erreicht.

Abstract

The appropriate body discretization (mesh) is of vital importance for finite element analysis. Amongst other procedures numerical results can be improved by mesh refinement (h-adaptivity). This paper introduces an a posteriori procedure which is based on an error estimator developed by Zienkiewicz and Zhu. With the help of recursive appliance - splitting of triangle elements (1. and 2. order) - we achieve a homogeneous error distribution and a maximum global error.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Aufgabenstellung	1
1.2	Das linearelastische Problem	2
1.3	Die Finite Elemente Lösung - Die Diskretisierung	3
1.3.1	Hinweise für die Anwendung adaptiver FE Methoden	4
2	Fehleranalyse	5
2.1	Fehlernorm und Fehlermaß	5
2.2	Fehlerschätzung	6
3	Netzverfeinerung	9
3.1	Verfeinerungsstrategie	9
3.2	Adaptiver Verfeinerungsalgorithmus	9
3.3	Netzverfeinerung auf Basis des Z(ienkiewicz)Z(hu) Fehlerschätzers	10
4	Anwendung auf das Dreieckselement	12
4.1	Formfunktionen	13
4.2	Verzerrungs - Verschiebungs - Beziehungen	14
4.3	Elastizitätsmatrix	16
4.3.1	Ebener Verzerrungszustand	16
4.3.2	Ebener Spannungszustand	17
4.4	Elementsteifigkeitsmatrix	17
4.5	Elementlastvektor	17
4.6	Berechnung der Knotenspannungen	20
4.6.1	Projektionskoeffizientenmatrix \mathbf{A}	20
4.6.2	Projektionslastvektor \mathbf{r}	23
4.7	Energienormen	23
4.7.1	Berechnung der Elementfehlnorm	23
4.7.2	Berechnung der Verzerrungsenergie	24
5	Literatur	25

A	Anhang - Beispiele	27
A.1	Dickwandiger Zylinder unter Innendruck	27
A.1.1	Ergebnisse	28
A.1.2	Eingabedaten	33
A.2	Kurzer Kragträger unter Streckenlast	36
A.2.1	Ergebnisse	36
A.2.2	Eingabedaten	40
A.3	Scheibe mit kreisförmigen Loch	42
A.3.1	Ergebnisse	43
A.3.2	Eingabedaten	49
B	Anhang - Software	55
B.1	SOOFEA	55
B.2	Formfunktionen	55
B.3	Numerische Integration	57
B.3.1	Implementierung mittels Funktoren	58
B.4	Lineare Algebra	60
B.5	Verfeinerungsalgorithmus für Dreiecksnetze	61
B.5.1	Adaptive Verfeinerung	61
B.5.2	Globale Verfeinerung	64
B.5.3	Generelle Hinweise	64

Abbildungsverzeichnis

2.1	Die Eigenschaften der FE Lösung anhand eines eindimensionalen Problems mit linearer Formfunktion	7
4.1	Das Dreieckselement	12
4.2	Die Koordinatensysteme am Dreieck	12
4.3	Äußere Lastaufbringung	18
4.4	Interpolation und Integration der Elementkraftdichte $\bar{\mathbf{t}}^e$	19
4.5	Knotengewichte bei Knotenquadratur am Dreieck	22
A.1	Dickwandiger Zylinder unter Innendruck	27
A.2	Modellproblem - Dickwandiger Zylinder unter Innendruck	28
A.3	Analytischer Spannungsverlauf	28
A.4	Verfeinerungsfaktor für Dreieckselement 1. Ordnung - $\epsilon_{max} = 7\%$	29
A.5	Spannungsdivergenz $\epsilon_{FE}^{analytisch}$ zwischen FE Lösung und analytischer Lösung je Verfeinerungsschritt - Dreieckselement 1. Ordnung	30
A.6	Verfeinerungsfaktor für Dreieckselement 2. Ordnung - $\epsilon_{max} = 0.2\%$	31
A.7	Spannungsdivergenz $\epsilon_{FE}^{analytisch}$ zwischen FE Lösung und analytischer Lösung je Verfeinerungsschritt - Dreieckselement 2. Ordnung	32
A.8	Kurzer Kragträger unter Streckenlast	37
A.9	Verfeinerungsfaktor für Dreieckselement 1. Ordnung - $\epsilon_{max} = 7\%$	38
A.10	Verfeinerungsfaktor für Dreieckselement 2. Ordnung - $\epsilon_{max} = 1\%$	39
A.11	Scheibe mit kreisförmigen Loch	42
A.12	Modellproblem - Scheibe mit kreisförmigen Loch	42
A.13	Analytischer Spannungsverlauf	44
A.14	Verfeinerungsfaktor für Dreieckselement 1. Ordnung - $\epsilon_{max} = 1\%$	45
A.15	Spannungsdivergenz $\epsilon_{FE}^{analytisch}$ zwischen FE Lösung und analytischer Lösung je Verfeinerungsschritt - Dreieckselement 1. Ordnung	46
A.16	Verfeinerungsfaktor für Dreieckselement 2. Ordnung - $\epsilon_{max} = 0.1\%$	47
A.17	Spannungsdivergenz $\epsilon_{FE}^{analytisch}$ zwischen FE Lösung und analytischer Lösung je Verfeinerungsschritt - Dreieckselement 2. Ordnung	48
B.1	Rote Teilung - R1	61
B.2	Grüne Teilung - R2	63
B.3	Schließen der Triangulation	63

B.4 Bildung neuer Knoten 64

Tabellenverzeichnis

2.1 Korrekturwerte für geschätzten relativen Fehler 8

1 Einleitung

1.1 Aufgabenstellung

Die Aufgabenstellung dieser Diplomarbeit dreht sich um die Aussage von Zienkiewicz und Zhu [16]:

The subject of error estimates for finite element solutions and a consequent adaptive analysis, in which the approximation is successively refined to reach pre-determined standards of accuracy, is central to the effective use of finite element codes for practical engineering analysis.

Eine praktische Anwendung solcher adaptiver Verfahren gliedert sich also in zwei Problemstellungen

Schätzung des Fehlers, der durch die Anwendung der Diskretisierung bei den Finiten Elementen gemacht wird.

Schrittweise Verfeinerung um die vorgegebene Genauigkeit der Lösung gewährleisten zu können.

Für den ersten Punkt sind schnelle und exakte Fehlerschätzer wünschenswert. Diese *a posteriori* Fehlerschätzer werden in zwei große Gruppen unterteilt, den *recovery based error estimator* und den *residual based error estimator* [15]. Für diese Diplomarbeit wurde der auch als ZZ-Fehlerschätzer (Zienkiewicz und Zhu) bekannte *recovery based error estimator* verwendet [16].

Die Schwierigkeit des zweiten Punktes besteht in der Entwicklung eines geeigneten Algorithmus zur lokalen Verfeinerung des Netzes. Hierfür muss eine geeignete Datenstruktur für das Netz gewählt werden, häufig kommen "hierarchische Netze" zur Anwendung. Des Weiteren muss bei der Verfeinerung Rücksicht auf die Randgeometrie des Gebietes genommen werden und die Gültigkeit des Netzes erhalten bleiben, d.h. es sollten keine hängenden Knoten entstehen. Es gibt zwei primäre Varianten zur Verfeinerung des Netzes:

h - Verfeinerung: Hier wird die Größe der Elemente durch eine geeignete Unterteilung der Mutterelemente in Tochterelemente reduziert

p - Verfeinerung: Hier wird die Ordnung der Polynome für die Ansatzfunktion eines Elementes erhöht.

Diese Arbeit beschränkt sich auf den Einsatz der h - Verfeinerung auf *lineare* und *bilineare* Dreieckselemente mit Hilfe des *recovery based* ZZ-Fehlerschätzers.

Im Folgenden möchte ich einen kurzen Überblick über das in meiner Arbeit angewandte Finite Elemente Verfahren für den Spezialfall des linearelastischen Problems geben. Ganz zuletzt hebe ich bereits hier besonders beachtenswerte Punkte hervor, welche später für eine Anwendung eines adaptiven Verfahrens unabdingbar sind.

1.2 Das linearelastische Problem

Ich möchte in dieser Arbeit auf die Lösung des linearelastischen Problems - wie in [16] beschrieben - eingehen. Dieses Problem ist durch folgende Differentialgleichung formuliert

$$\mathbf{L}u = \mathbf{q} \quad \text{in } \Omega \quad \mathbf{D}\mathbf{S}u = \mathbf{q} = \mathbf{0} \quad (1.1)$$

gültig im Gebiet Ω , mit der vorgeschriebenen Verschiebung

$$\mathbf{u} = \bar{\mathbf{u}} \quad \text{auf dem Rand } \Gamma_u \quad (1.2)$$

und der vorgeschriebenen Belastung

$$\mathbf{G}\mathbf{D}\mathbf{S}u = \bar{\mathbf{t}} \quad \text{auf dem Rand } \Gamma_t \quad (1.3)$$

wobei für den Rand verlangt wird

$$\Gamma = \Gamma_u \cup \Gamma_t \quad (1.4)$$

Hier beschreibt der Matrixoperator \mathbf{S} die Verzerrungen $\boldsymbol{\epsilon}$ als

$$\boldsymbol{\epsilon} = \mathbf{S}u \quad (1.5)$$

und die Elastizitätsmatrix \mathbf{D} definiert die Spannungen mit Hilfe des linearelastischen Materialgesetzes nach Hooke (Verzerrungs-Verschiebungs-Gesetz siehe auch [5], [12]) als

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\epsilon} \quad (1.6)$$

1.3 Die Finite Elemente Lösung - Die Diskretisierung

Wir setzen für die Finite Elemente Lösung

$$\mathbf{u} \approx \hat{\mathbf{u}} = \mathbf{N}\mathbf{a} \quad (1.7)$$

an, wobei \mathbf{a} die Knotenpunktverschiebungen eines Elementes darstellen. Durch das Galerkin Verfahren (Multiplikation der Differentialgleichung mit der Testfunktion und Integration über das Gebiet Ω) ergibt sich aus (1.1)

$$\mathbf{K}\mathbf{a} = \mathbf{f} = \mathbf{r} \quad (1.8)$$

mit

$$\begin{aligned} \mathbf{K} &= \int_{\Omega} (\mathbf{S}\mathbf{N})^T \mathbf{D} (\mathbf{S}\mathbf{N}) \, d\Omega \\ &= \int_{\Omega} \mathbf{B}^T \mathbf{D} \mathbf{B} \, d\Omega \\ \mathbf{f} &= \int_{\Omega} \mathbf{N}^T \mathbf{b} \, d\Omega + \int_{\Gamma} \mathbf{N}^T \bar{\mathbf{t}} \, d\Gamma \end{aligned}$$

\mathbf{b} sind die verteilten Volumskräfte und \mathbf{r} die konzentrierten Knotenkräfte und $\bar{\mathbf{t}}$ die verteilte Kraftdichte am Rand Γ des Gebietes. Bei der Berechnung von \mathbf{f} berücksichtigen wir weder Anfangsverzerrungen $\boldsymbol{\varepsilon}_0$ noch Anfangsspannungen $\boldsymbol{\sigma}_0$, weiters ist die Temperatur über dem gesamten Gebiet Ω konstant. Eine genauere Beschreibung des Aufbaus der hier angegebenen Matrizen und Vektoren in meiner Implementierung folgt in Kapitel 4.

Die Spannungen ergeben sich weiters nach Lösung des FE Gleichungssystems (1.8) unter Berücksichtigung von (1.6) aus

$$\hat{\boldsymbol{\sigma}} = (\mathbf{D}\mathbf{S}\mathbf{N})\mathbf{a} \quad (1.9)$$

Die Näherungslösungen $\hat{\mathbf{u}}, \hat{\boldsymbol{\sigma}}$ der schwachen Formulierung (1.8) unterscheiden sich von der exakten Lösung $\mathbf{u}, \boldsymbol{\sigma}$ der starken Formulierung (1.1). Diesen Unterschied bezeichnen wir als Fehler. Dies bedeutet für die Verschiebungen

$$\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}} \quad (1.10)$$

für Spannungen

$$\mathbf{e} = \boldsymbol{\sigma} \hat{\boldsymbol{\sigma}} \quad (1.11)$$

1.3.1 Hinweise für die Anwendung adaptiver FE Methoden

Randbedingungen Für die Lösung des Gleichungssystems (1.8) ist es erforderlich Randbedingungen einzubauen. Dies bedeutet im Prinzip Freiheitsgrade des Systems vorzugeben oder zu koppeln, sodass das Gleichungssystem regulär und somit lösbar wird.

Sei nun $\bar{\omega}_h^u$ die Indexmenge aller auf Γ_u liegenden Knoten, wobei Γ_u jene Teilmenge darstellt auf der Randbedingungen 1. Art (z.B. Verschiebungen) gegeben sind. Ist die Lösung für ein gegebenes Netz (eine gültige Diskretisierung $\bar{\Omega}_h$ des Gebietes Ω) gefragt, so kann man relativ einfach die entsprechenden Werte bzw. gegebenen Zusammenhänge der Freiheitsgrade der Knoten $\bar{\omega}_h^u$ in das GLS (1.8) einbauen. Hierfür gibt es verschiedene Möglichkeiten, auf die wir hier nicht näher eingehen werden, siehe [15] bzw. [3].

Wenn nun aber das Ursprungsnetz ${}^0\bar{\Omega}_h$ eine Verfeinerung erfährt (wie es bei adaptiven Verfahren der Fall ist), so wird für den allgemeinen Fall das neue Netz ${}^1\bar{\Omega}_h$ eine neue Indexmenge ${}^1\bar{\omega}_h^u$ von Knoten auf Γ_u aufweisen, welche nicht ident ist mit ${}^0\bar{\omega}_h^u$. Dies bedeutet mit anderen Worten, dass die Randbedingung eine Eigenschaft des Randes Γ_u ist und nicht eine Eigenschaft der Knoten. Es ist daher nicht möglich, die Randbedingungen als Eigenschaft der Knoten in das Ursprungsnetz zu übertragen, man muss bereits für den Inputdatensatz geometrische Randbereiche $\Gamma_{u,i}$ (im Fall von 2D Elementen sind dies einparametrische Kurvenstücke, Strecken, Kreise, usw.) definieren, an denen Randbedingungen gegeben sind.

Lasten Ähnliches wie für Randbedingungen gilt auch für Lasten. Da es physikalisch gesehen unmöglich ist, Einzelkräfte an Punkten mit unendlich kleiner Ausdehnung anzubringen, ist auch die "Kraft" oder besser Kraftdichte $\bar{\mathbf{t}}$ als Eigenschaft des Randbereiches Γ_t zu verstehen. Gilt es mehr als nur das Stammnetz ${}^0\bar{\Omega}_h$ zu rechnen auf dem die Indexmenge ${}^0\bar{\omega}_h^t$ durchaus im vorhinein gegeben sein mag, so muss die Information der äußeren Lasten im Inputdatensatz an die Randbereiche $\Gamma_{t,i}$ gekoppelt werden.

2 Fehleranalyse

2.1 Fehlernorm und Fehlermaß

Der punktweise Fehler ergibt sich wie in 1.3 beschrieben und in den Gleichungen (1.10) bzw. (1.11) gezeigt und ist im Allgemeinen schwierig zu bestimmen oder anzugeben. Eines der gebräuchlichsten Fehlermaße ist daher die "Energienorm", die geschrieben werden kann als

$$\|\mathbf{e}\| = \left(\int_{\Omega} \mathbf{e}^T \mathbf{L} \mathbf{e} \, d\Omega \right)^{\frac{1}{2}} \quad (2.1)$$

und welche sich im speziellen Fall des hier behandelten linearelastischen Problems berechnen lässt als

$$\begin{aligned} \|\mathbf{e}\| &= \left(\int_{\Omega} (\mathbf{S}\mathbf{e})^T \mathbf{D}(\mathbf{S}\mathbf{e}) \, d\Omega \right)^{\frac{1}{2}} \\ &= \left(\int_{\Omega} (\mathbf{e}_{\varepsilon})^T \mathbf{D}(\mathbf{e}_{\varepsilon}) \, d\Omega \right)^{\frac{1}{2}} \\ &= \left(\int_{\Omega} (\mathbf{e})^T \mathbf{D}^{-1}(\mathbf{e}) \, d\Omega \right)^{\frac{1}{2}} \end{aligned} \quad (2.2)$$

Die Definition dieser Fehlernorm wurde für das gesamte Gebiet vorgenommen, es ist aber zu bemerken, dass das Quadrat der Fehlernorm durch die Summierung der Elementbeiträge ermittelt werden kann

$$\|\mathbf{e}\|^2 = \sum_{e=1}^m \|\mathbf{e}^e\|^2 \quad (2.3)$$

wobei e das jeweilige Element identifiziert, und m die Anzahl der Elemente im Gebiet darstellt. Für ein optimales Netz sollte man versuchen die Elementfehlernorm über das Netz gleichmäßig zu verteilen.

Der absolute Wert der Fehlernorm hat wenig physikalische Bedeutung, was dazu führt, dass man einen relativen Fehler einführt

$$\frac{\|\mathbf{e}\|}{\|\mathbf{u}\|} \quad (2.4)$$

2.2 Fehlerschätzung

Die unter 2.1 angegebene Fehlernorm kann für die FE Lösung schon auf Grund der Definition nicht a priori angegeben werden. Dies liegt ganz einfach daran, dass die exakte Lösung (welche zur Berechnung des exakten Fehlers notwendig wäre, siehe (1.10) und (1.11)) nicht bekannt ist. Die Fehlernorm kann aber auch a posteriori nicht exakt berechnet sondern nur “geschätzt” werden, da im Allgemeinen lediglich die fehlerbehaftete FE Lösung vorliegt. Es ist nun die Aufgabe des Fehlerschätzers eine Näherung für die Fehlernorm anzugeben in dem eine “bessere” Lösung als die FE-Lösung ermittelt wird: *recovery based error estimator*.

Für die Testfunktionen (siehe (1.7)) wird lediglich C_0 Stetigkeit vorausgesetzt. Dies bedeutet einen nicht stetigen Verlauf der Spannungen $\hat{\boldsymbol{\sigma}}$ über das Lösungsgebiet Ω , da die Spannungen $\boldsymbol{\sigma}$ proportional den Verzerrungen $\boldsymbol{\varepsilon}$ sind (siehe (1.6)). Diese wiederum sind nach dem Verzerrungs-Verschiebungsgesetz (1.5) proportional der ersten Ableitung der Verschiebung \mathbf{u} . In Abbildung 2.1 werden die Eigenschaften der FE Lösung anhand eines eindimensionalen Problems mit linearer Formfunktion gezeigt.

Bei dem hier besprochenen Fehlerschätzer besteht die Aufgabe darin, einen stetigen Spannungsverlauf, dem höhere Exaktheit unterstellt wird als sie die FE Lösung aufweist, zu ermitteln. Im Wesentlichen geschieht dies durch Ermittlung von Knotenspannungen, die anschließend mit Hilfe der Formfunktionen interpoliert werden. Hier haben sich verschiedene Verfahren für den *recovery* Prozess zur Generierung der Knotenspannung aus den Elementspannungen, welche in den Gausspunkten gegeben sind, entwickelt. In dieser Arbeit erfolgt die Knotenmittelung der Spannungen durch ein Projektionsverfahren wie in (2.6) angegeben. Daneben gibt es aber auch noch andere Verfahren wie *Superconvergent patch recovery - SPR* oder *Recovery by equilibration of patches - REP* (siehe [15]). Wie bereits erwähnt, werden für den gewünschten C_0 stetigen Spannungsverlauf in Folge diese neu gewonnenen Knotenspannungen mit den selben Formfunktionen wie die Verschiebungen interpoliert (2.5):

$$\boldsymbol{\sigma}^* = \mathbf{N}\bar{\boldsymbol{\sigma}}^* \quad (2.5)$$

$$\int_{\Omega} \mathbf{N}^T \boldsymbol{\sigma}^* \hat{\boldsymbol{\sigma}} \, d\Omega = 0 \quad (2.6)$$

Durch Einsetzen der von (2.5) in (2.6) ergibt sich (2.7)

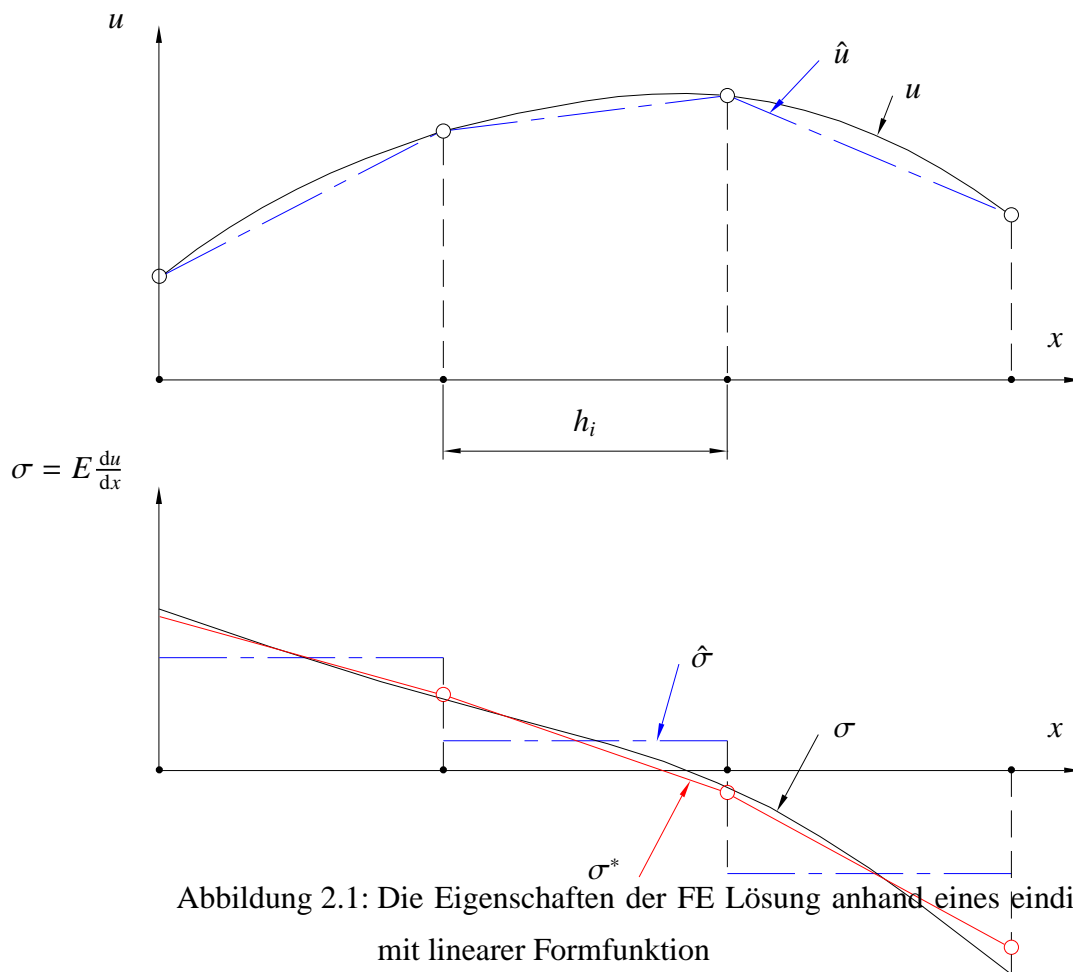


Abbildung 2.1: Die Eigenschaften der FE Lösung anhand eines eindimensionalen Problems mit linearer Formfunktion

$$\bar{\sigma}^* = \mathbf{A}^{-1} \int_{\Omega} \mathbf{N}^T \mathbf{D} \mathbf{S} \mathbf{N} \, d\Omega \mathbf{a} \quad (2.7)$$

wobei

$$\mathbf{A} = \int_{\Omega} \mathbf{N}^T \mathbf{N} \, d\Omega$$

Mit den in Folge gewonnen Knotenspannungen kann ein neuer, gegebenüber $\hat{\sigma}$ “verbesserte” Spannungsverlauf σ^* angegeben werden. Diese intuitive Aussage, dass σ^* eine “bessere” Näherung darstellt als die direkte FE Lösung $\hat{\sigma}$ wird dadurch untermauert, dass σ^* eine um 1 höhere Polynomordnung als $\hat{\sigma}$ hat. Diese neuen geglätteten nun C_0 stetigen Spannungen σ^* sind ebenfalls in Abbildung 2.1 ersichtlich. Dies führt automatisch zur Überlegung, den Fehler durch

$$\mathbf{e} = \hat{\boldsymbol{\sigma}} - \boldsymbol{\sigma}^* \quad (2.8)$$

abzuschätzen. Damit kann nun eine Näherung für die Fehlnorm, wie unter (2.2) angegeben, berechnet werden. Man beachte, dass in Folge alle Werte, die durch den *recovery* Prozess gewonnen werden, mit einem $\hat{}$ und alle daraus geschätzten Werte mit einer links hochgestellten $\hat{}$ gekennzeichnet sind.

$$\|\mathbf{e}\| = \|\hat{\mathbf{e}}\| = \left(\int_{\Omega} (\hat{\boldsymbol{\sigma}} - \boldsymbol{\sigma}^*)^T \mathbf{D} (\hat{\boldsymbol{\sigma}} - \boldsymbol{\sigma}^*) d\Omega \right)^{\frac{1}{2}} \quad (2.9)$$

Somit kann nun auch ein Effizienzindex angegeben werden, welcher ein Maß für die Güte des Fehlerschätzers darstellt.

$$\text{Effizienzindex} = \frac{\|\hat{\mathbf{e}}\|}{\|\mathbf{e}\|} \quad (2.10)$$

Mit Hilfe von (2.4) ist der relative Fehler gegeben durch

$$\frac{\|\hat{\mathbf{e}}\|}{\|\mathbf{u}\|} = \frac{\|\hat{\mathbf{e}}\|}{\|\hat{\mathbf{u}}\|} = \sqrt{\frac{\|\hat{\mathbf{e}}\|^2}{\|\hat{\mathbf{u}}\|^2 + \|\hat{\mathbf{e}}\|^2}} \quad (2.11)$$

Abschließend geben Zienkiewicz und Zhu in ihrer Arbeit [16] Korrekturfaktoren an, welche von Elementtyp abhängig sind, und die Fehlerschätzung dem exakten Fehler annähern, siehe Tabelle 2.1. D.h. der korrigierte Wert für $\|\hat{\mathbf{e}}\|$ ergibt sich zu

$$\|\hat{\mathbf{e}}\|_{\text{korrigiert}} = \|\hat{\mathbf{e}}\| \cdot \text{Korrekturfaktor} \quad (2.12)$$

<i>Elementtyp</i>	
Bilineares 4-Knoten Viereckselement	1.1
Biquadratisches 9-Knoten Viereckselement	1.6
Lineares 3-Knoten Dreieckselement	1.3
Quadratisches 6-Knoten Dreieckselement	1.4

Tabelle 2.1: Korrekturwerte für geschätzten relativen Fehler

3 Netzverfeinerung

3.1 Verfeinerungsstrategie

Die Verfeinerungsstrategie hängt davon ab welche Genauigkeit von der Lösung gefordert wird. Es ist üblich, einen maximalen relativen Fehler zu fordern. Das bedeutet, dass am Ende der Berechnung (d.h. bereits nach eventuell mehrmaliger Verfeinerung) der relative Fehler kleiner/gleich einem maximal zulässigen relativen Fehler $\bar{\epsilon}$ ist.

$$\frac{\|\mathbf{e}\|}{\|\mathbf{u}\|} \leq \bar{\epsilon} \quad (3.1)$$

Natürlich kann man dies praktisch nicht mit dem exakten relativen Fehler überprüfen, denn es steht einem ja lediglich die Fehlerschätzung $\hat{\mathbf{u}}$ zur Verfügung. Ich möchte die Verfeinerungsstrategie aber unabhängig von einem gewählten Fehlerschätzer formulieren und in Folge postulieren, den exakten Fehler, der durch die FE Approximation entstanden ist, zu kennen. Wenn wir nun voraussetzen, dass der Fehler gleichmäßig über das Netz verteilt ist, können wir die Bedingung (3.1) in ein Kriterium für jedes Element i (m ist die Anzahl der Elemente) überführen.

$$\|\mathbf{e}\|_i \leq \frac{\|\mathbf{u}\|}{m} = \sqrt{\frac{\|\hat{\mathbf{u}}\|^2 + \|\mathbf{e}\|^2}{m}} = \bar{\epsilon}_m \quad (3.2)$$

Da der Fehler $\|\mathbf{e}\|_i$ elementweise berechnet wird, kann sehr einfach das Kriterium formuliert werden, welches die Notwendigkeit einer Verfeinerung des Elementes definiert

$$r_i = \frac{\|\mathbf{e}\|_i}{\bar{\epsilon}_m} \quad (3.3)$$

$r_i \leq 1$ keine Verfeinerung notwendig, der Elementfehler ist kleiner/gleich dem erlaubten

$r_i > 1$ Verfeinerung notwendig, der Elementfehler ist größer dem erlaubten

3.2 Adaptiver Verfeinerungsalgorithmus

Für die Verfeinerung des Netzes können nun unterschiedliche Algorithmen herangezogen werden. Einerseits kann ein hierarchischer Algorithmus verwendet werden, der aufbauend auf das Ursprungsnetz nur einzelne Elemente unterteilt. Andererseits wäre es auch denkbar, das Netz mit Hilfe der vorhandenen Information über die Fehlerverteilung neu zu generieren.

Neugenerierung des Netzes Hier kann der Wert von ν_i Auskunft über den Grad der Unterteilung des Netzes an gewissen Stellen geben. Wenn die aktuelle Elementgröße mit h_i identifiziert werden kann und wir die Konvergenzrate als $O(h^p)$ voraussetzen dann kann man sagen, dass

$$h = \frac{h_i}{\nu_i^{1/p}} \quad (3.4)$$

gewählt werden kann für Elemente in denen $\nu_i > 1$ ist. p sei hier die Polynomordnung der Testfunktionen. Diese einfache Konvergenzregel gilt nicht im Bereich von Singularitäten. Trotzdem kann mit derartigen Algorithmen die gesuchte Genauigkeit bereits in einem bis sehr wenigen Schritten erreicht werden. In der Umsetzung bedeuten derartige Verfeinerungen aber ein Neuvernetzen der Geometrie, d.h. man braucht effiziente Netzgeneratoren, in denen Größeninformationen verarbeitet werden können.

Adaptive Verfeinerung von “markierten” Elementen auf Basis eines Startnetzes

Wir haben uns in dieser Arbeit auf diese Methode beschränkt. Hier besteht ebenfalls Optimierungspotential wenn *hierarchische Netze* und darauf aufbauend *hierarchische Lösungsverfahren* verwendet werden. D.h., es werden die Information über das Mutternetz gespeichert und nach dem Rechendurchlauf nicht einfach verworfen.

3.3 Netzverfeinerung auf Basis des Zienkiewicz)Z(hu)

Fehlerschätzers

Nun ist noch zu berücksichtigen, wie man die Verfeinerung bei geschätztem und nicht exaktem Fehlerwert vornimmt. Das bedeutet in diesem Fall, dass man nicht $\|\mathbf{e}\|$ sondern lediglich $\|\hat{\mathbf{u}}\|$ ermitteln kann. Für die Berechnung des maximal zulässigen Elementfehlers ist im wesentlichen $\|\mathbf{e}\|^2 + \|\hat{\mathbf{u}}\|^2$ notwendig, aber lediglich $\|\mathbf{e}\|^2 + \|\hat{\mathbf{u}}\|^2$ bekannt. Mit folgender Näherung kann nun aber auch $\|\mathbf{e}\|$ unter Berücksichtigung der Korrektur $\|\hat{\mathbf{u}}\|$ angeähert werden

$$= \sqrt{\frac{\|\mathbf{e}\|^2}{\|\hat{\mathbf{u}}\|^2 + \|\mathbf{e}\|^2}} \sqrt{\|\hat{\mathbf{u}}\|^2 + \|\mathbf{e}\|^2} \quad (3.5)$$

In Kombination mit (3.2) führt dies zu folgendem Ausdruck fuer den maximal zulässigen Elementfehler

$$\bar{\epsilon}_m = \sqrt{\frac{1}{m(1-\alpha^2)}} \quad (3.6)$$

Damit lässt sich nun für jedes Element abschätzen

$$\epsilon_i = \frac{\|\mathbf{e}\|_i}{\bar{\epsilon}_m} \quad (3.7)$$

4 Anwendung auf das Dreieckselement

Die zuvor beschriebenen Konzepte habe ich anhand eines linearen bzw. quadratischen Dreieckselementes für den ebenen Verzerrungs- bzw. Spannungszustand umgesetzt. Hier möchte ich kurz auf die Anwendung der FE Methode und des Fehlerschätzers in diesen Fällen eingehen.

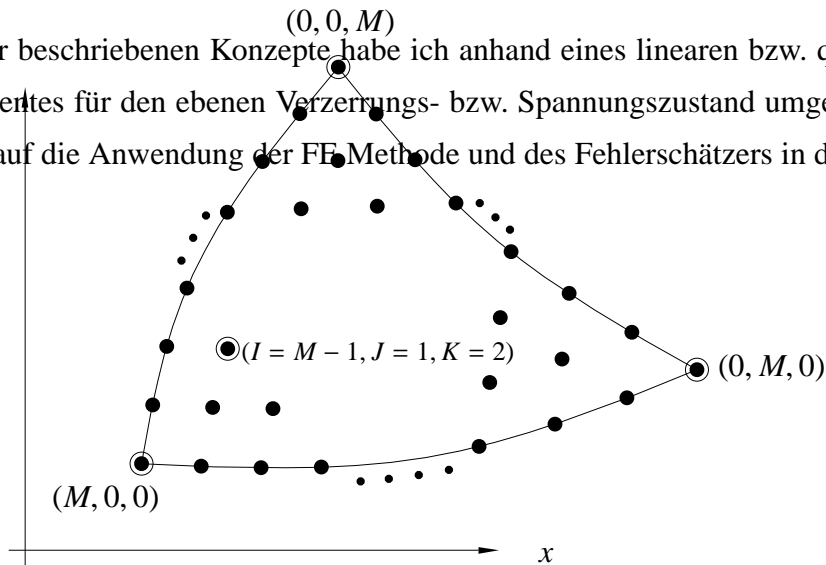


Abbildung 4.1: Das Dreieckselement

Zur Beschreibung des Dreieckselementes wurden 3 Koordinatensysteme verwendet, welche in Abbildung 4.2 zu sehen sind

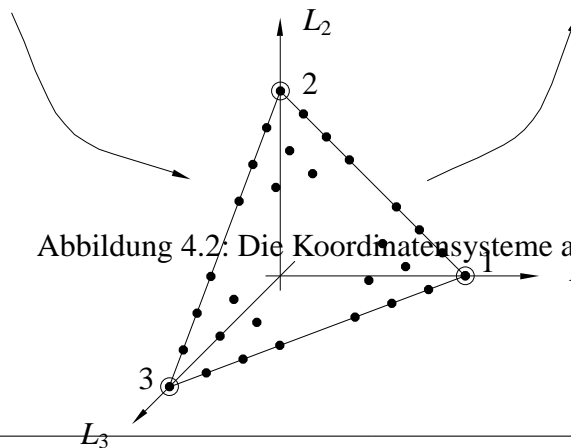
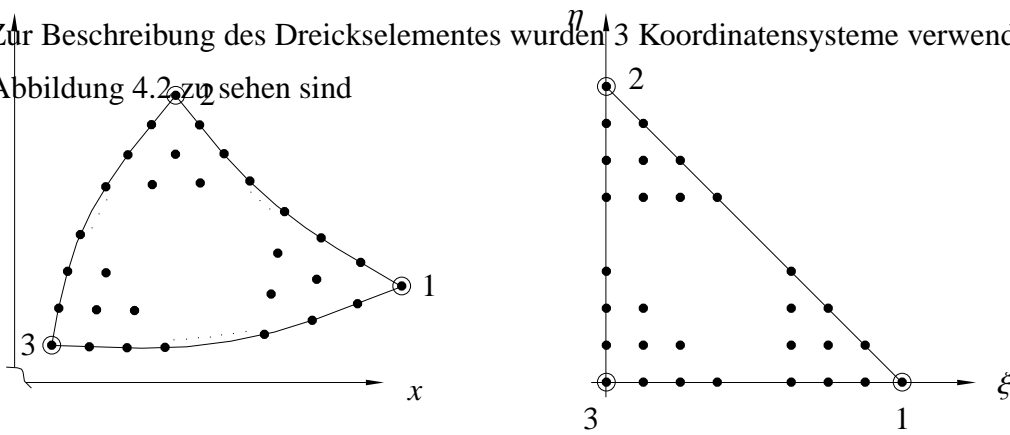


Abbildung 4.2: Die Koordinatensysteme am Dreieck

4.1 Formfunktionen

Die beschriebenen Formfunktionen sind in dieser Form für Dreieckselemente der Lagrange Klasse mit beliebiger Ordnung gültig, aus diesem Grund sei in Folge die Ordnung der Formfunktionspolynome gleich M . Dann kann für einen Knoten i mit den Indizes $i = (I, J, K)$ (zur Bezeichnungsweise siehe Abbildung 4.1) mit folgendem Ausdruck die Formfunktion $N_{(I,J,K)}$ angegeben werden (in Anlehnung an [15], p.181, equ.(8.35))

$$N_{(I,J,K)}(L_1, L_2, L_3) = l_I^I(L_1) l_J^J(L_2) l_K^K(L_3) \quad (4.1)$$

Dies gilt unter der Zwangsbedingung

$$I + J + K = M \quad \text{für} \quad I, J, K \in \{0, M\} \quad (4.2)$$

Hier stellt l_k^n das Lagrange Polynom der Ordnung n an der Stützstelle k darstellt.

$$l_k^n(\cdot) = \prod_{\substack{i=0 \\ i \neq k}}^n \frac{i - k}{i - k} \quad (4.3)$$

Des Weiteren sei angemerkt, dass die Anzahl der Knoten je Element mit $\frac{(M+1)(M+2)}{2}$ gegeben ist. Mit Hilfe dieser Formfunktionen (gegeben in natürlichen Koordinaten L_1, L_2, L_3) ist es möglich die Verschiebungen des Elementes mit Hilfe der Knotenpunktverschiebungen \mathbf{a}_i an einer beliebigen Stelle innerhalb des Elementes anzugeben

$$\hat{\mathbf{u}}^e(L_1, L_2, L_3) = \sum_{i=1} N_i(L_1, L_2, L_3) \mathbf{a}_i \quad (4.4)$$

mit $\mathbf{a}_i = [a_{x,i} \ a_{y,i}]^T$ und $\hat{\mathbf{u}} = [\hat{u} \ \hat{v}]^T$. Um diese Operation als Matrix-Vektor-Multiplikation durchführen zu können (wie in (1.7) gezeigt) ist es notwendig folgenden Vektor zu definieren

$$\mathbf{a} = [a_{x,1} \ a_{y,1} \ a_{x,2} \ a_{y,2} \ \dots \ a_{x,M} \ a_{y,M}]^T \quad (4.5)$$

und schließlich ergibt sich (1.7) zu

$$\hat{\mathbf{u}}^e(L_1, L_2, L_3) = \underbrace{\begin{bmatrix} N_1 & 0 & N_2 & 0 \\ 0 & N_1 & 0 & N_2 \end{bmatrix}}_{\mathbf{N}^e} \underbrace{\begin{bmatrix} \mathbf{0} \\ 0 \\ a_{x,1} \\ a_{y,1} \\ a_{x,2} \\ a_{y,2} \\ \vdots \\ a_{x,} \\ a_{y,} \end{bmatrix}}_{\mathbf{a}^e} \quad (4.6)$$

4.2 Verzerrungs - Verschiebungs - Beziehungen

Mit Hilfe der interpolierten Verschiebungen, die an jedem Punkt des Elementes definiert sind, können die Verzerrungen (wie bereits in (1.5) gezeigt) angegeben werden. Siehe hierfür [5] bzw. [12] für den Zusammenhang von Verzerrungen und Verschiebungen:

$$\boldsymbol{\varepsilon} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{Bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix}}_{\mathbf{S}} \underbrace{\begin{Bmatrix} u \\ v \end{Bmatrix}}_{\mathbf{u}} \quad (4.7)$$

Die FE Approximation besteht nun darin, \mathbf{u} durch $\hat{\mathbf{u}}$ anzunähern (siehe wiederum (1.7)) und somit ergibt sich

$$\boldsymbol{\varepsilon} \approx \hat{\boldsymbol{\varepsilon}} = \mathbf{S}\hat{\mathbf{u}} = \mathbf{S}\mathbf{N}\mathbf{a} = \mathbf{B}\mathbf{a} \quad (4.8)$$

mit einer Verzerrungs-Verschiebungs-Matrix \mathbf{B}^e für das Element

$$\mathbf{B}^e(L_1, L_2, L_3) = \begin{bmatrix} \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial x} & 0 & \frac{\partial N}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial y} & 0 & \frac{\partial N}{\partial y} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & \frac{\partial N}{\partial y} & \frac{\partial N}{\partial x} \end{bmatrix} \quad (4.9)$$

Es ist somit die Aufgabe gestellt, Ableitungen der Gestalt $\frac{\partial N_i}{\partial x}$ und $\frac{\partial N_i}{\partial y}$ zu bilden. Da N_i aber in der bis jetzt angegebenen Form eine Funktion der natürlichen Koordinaten darstellt ist es notwendig eine Verbindung zwischen den natürlichen und den kartesischen Koordinaten herzustellen. Dies Abbildung erfolgt über die Jacobi-Matrix. In dem Fall der natürlichen

Koordinaten sind (L_1, L_2, L_3) aber durch die Zwangsbedingung $L_1 + L_2 + L_3 = 1$ nicht voneinander unabhängig. Die Jacobi Matrix würde durch die Abbildung von 3 nicht unabhängige auf 2 unabhängige Koordinaten rechteckförmig. Dieses Problem kann man auf unterschiedliche Arten lösen, wir haben uns entschieden die natürlichen Koordinaten (L_1, L_2, L_3) durch die isoparametrischen Koordinaten (r, s) zu ersetzen mit folgender Bedingung

$$\begin{aligned} L_1 &= L \\ L_2 &= L \\ L_3 &= 1 - L \end{aligned} \quad (4.10)$$

Durch obige Bedingungen ist es einfach sowohl $N(L_1, L_2, L_3)$ also auch $N(r, s)$ auszuwerten. Nun hat sich das Problem der Bildung der Ableitungen $\frac{\partial N_i}{\partial x}$ und $\frac{\partial N_i}{\partial y}$ auf das Problem der Bildung der Ableitungen $\frac{\partial N_i}{\partial r}$ und $\frac{\partial N_i}{\partial s}$ verlagert

$$\begin{aligned} \frac{\partial N_i}{\partial x} &= \frac{\partial N_i}{\partial r} \frac{\partial r}{\partial x} + \frac{\partial N_i}{\partial s} \frac{\partial s}{\partial x} \\ \frac{\partial N_i}{\partial y} &= \frac{\partial N_i}{\partial r} \frac{\partial r}{\partial y} + \frac{\partial N_i}{\partial s} \frac{\partial s}{\partial y} \end{aligned} \quad (4.11)$$

Für die Determinante $\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial r}, \frac{\partial}{\partial s}$ ist wie oben bereits erwähnt die Jacobi-Matrix wie folgt definiert

$$\mathbf{J} = \frac{\partial \mathbf{x}}{\partial \mathbf{r}} = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial s} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial s} \end{bmatrix}$$

Für die Bestimmung der Ableitungen $\frac{\partial N_i}{\partial x}$ und $\frac{\partial N_i}{\partial y}$ sind aber (wie in (4.11) ersichtlich) gerade die inversen Einträge notwendig. D.h. es gilt die inverse Jacobi-Matrix zu berechnen

$$\mathbf{J}^{-1} = \frac{\partial \mathbf{r}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial r}{\partial x} & \frac{\partial r}{\partial y} \\ \frac{\partial s}{\partial x} & \frac{\partial s}{\partial y} \end{bmatrix}$$

Nun kann letztendlich die Verzerrungs-Verschiebungs-Matrix mit Hilfe der soeben berechneten Ausdrücke in isoparametrischen Koordinaten $\mathbf{B}(r, s)$ angegeben werden

$$\mathbf{B}^e(r, s) = \underbrace{\begin{bmatrix} \frac{\partial N_0}{\partial r} \frac{\partial r}{\partial x} + \frac{\partial N_0}{\partial s} \frac{\partial s}{\partial x} & 0 & | & \frac{\partial N_1}{\partial r} \frac{\partial r}{\partial x} + \frac{\partial N_1}{\partial s} \frac{\partial s}{\partial x} & 0 \\ 0 & \frac{\partial N_0}{\partial r} \frac{\partial r}{\partial y} + \frac{\partial N_0}{\partial s} \frac{\partial s}{\partial y} & | & 0 & \frac{\partial N_1}{\partial r} \frac{\partial r}{\partial y} + \frac{\partial N_1}{\partial s} \frac{\partial s}{\partial y} \\ \frac{\partial N_0}{\partial r} \frac{\partial r}{\partial y} + \frac{\partial N_0}{\partial s} \frac{\partial s}{\partial y} & \frac{\partial N_0}{\partial r} \frac{\partial r}{\partial x} + \frac{\partial N_0}{\partial s} \frac{\partial s}{\partial x} & | & \frac{\partial N_1}{\partial r} \frac{\partial r}{\partial y} + \frac{\partial N_1}{\partial s} \frac{\partial s}{\partial y} & \frac{\partial N_1}{\partial r} \frac{\partial r}{\partial x} + \frac{\partial N_1}{\partial s} \frac{\partial s}{\partial x} \end{bmatrix}}_{2 \text{ Einträge}} \quad (4.12)$$

und somit

$$\hat{\boldsymbol{\epsilon}}^e(r, s) = \mathbf{B}^e(r, s) \mathbf{a}^e \quad (4.13)$$

4.3 Elastizitätsmatrix

Wie in (1.9) angezeigt, lassen sich die Spannungen $\hat{\boldsymbol{\sigma}}$ (mit der Voraussetzung nicht vorhandener Anfangsverzerrungen) ermitteln durch

$$\hat{\boldsymbol{\sigma}}^e(r, s) = \mathbf{D} \hat{\boldsymbol{\epsilon}}^e(r, s) \quad (4.14)$$

Im Fall der ebenen Scheibenelemente unterscheidet man die beiden Fälle ‘‘Ebener Verzerrungszustand’’ und ‘‘Ebener Spannungszustand’’ (siehe [5] bzw. [12]).

4.3.1 Ebener Verzerrungszustand

Hier geht man davon aus, dass $\epsilon_{zz} = \epsilon_{zx} = \epsilon_{yz} = 0$ ist.

$$\mathbf{D} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \quad (4.15)$$

4.3.2 Ebener Spannungszustand

Hier geht man davon aus, dass $\sigma_{zz} = \sigma_{zx} = \sigma_{yz} = 0$ ist.

$$\mathbf{D} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & & \\ & 1 & \\ 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix} \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \quad (4.16)$$

4.4 Elementsteifigkeitsmatrix

Die Höhe der Scheibe sei von nun an als $h(x, y)$ bzw. durch Koordinatentransformation als $h(r, s)$ gegeben.

$$\mathbf{K}^e = \int_{\Omega^e} \mathbf{B}^{eT} \mathbf{D} \mathbf{B}^e h \, d(x, y) \quad (4.17)$$

Die Integration über das Gebiet Ω^e ist unangenehm, geeigneter wäre die Integration über das Referenzgebiet des isoparametrischen Referenzelementes in (r, s) (zu den Bezeichnungen siehe Abb. 4.2). Die Integralabbildung ist aber auf Grund der bereits bekannten Jacobi-Matrix \mathbf{J} einfach durchgeführt

$$\mathbf{K}^e = \int \mathbf{B}^{eT}(r, s) \mathbf{D} \mathbf{B}^e(r, s) h \det \mathbf{J} \frac{\partial x}{\partial r} \, d(r, s) \quad (4.18)$$

Diese Elementsteifigkeitsmatrizen sind in Folge durch geeignete Assemblierung zur globalen Steifigkeitsmatrix \mathbf{K} über das Gebiet Ω zusammensetzen.

4.5 Elementlastvektor

Warum in meinem Fall der adaptiven FE Analyse ein Elementlastvektoren aus Kraftdichten ermittelt werden sollte und nicht über vereinzeltete Knotenkräfte ist unter Kapitel 1.3.1 beschrieben. Da für diese Anwendung von mir keine verteilten Volumskräfte \mathbf{b} berücksichtigt werden, beschränkt sich der Elementlastvektor \mathbf{f}^e auf

$$\mathbf{f}^e = \int_{\Gamma_i^e} \mathbf{N}^{eT} \bar{\mathbf{t}}^e \, d \quad (4.19)$$

Hier stellt sich die Frage, wie man die Kraftdichteverteilung $\bar{\mathbf{t}}$ auf dem Rand Γ_i an das Programm übergibt. In meiner Umsetzung wurde dies erreicht, in dem die Kraftdichte auf einem Randabschnitt $\Gamma_{i,p}$ gegeben ist (z.B. Druck p auf dem Randabschnitt $\Gamma_{i,p}$ der stets in Richtung des Normalenvektors \mathbf{n} zeigt). Für jeden Knoten i welcher auf dem Rand Γ_i liegt wird nun ein Kraftdichtevektor $\bar{\mathbf{t}}_i^e$ übertragen. Diese Verteilung der Kraftdichte $\bar{\mathbf{t}}$ auf die Randknoten $\bar{\omega}_h^i$ (kennzeichne den Verfeinerungsschritt und somit das aktuelle Netz $\bar{\Omega}_h$) ist auch nach jeder Verfeinerung durchzuführen, da sich die Indexmenge der betroffenen Randknoten nach jeder Verfeinerung verändert.

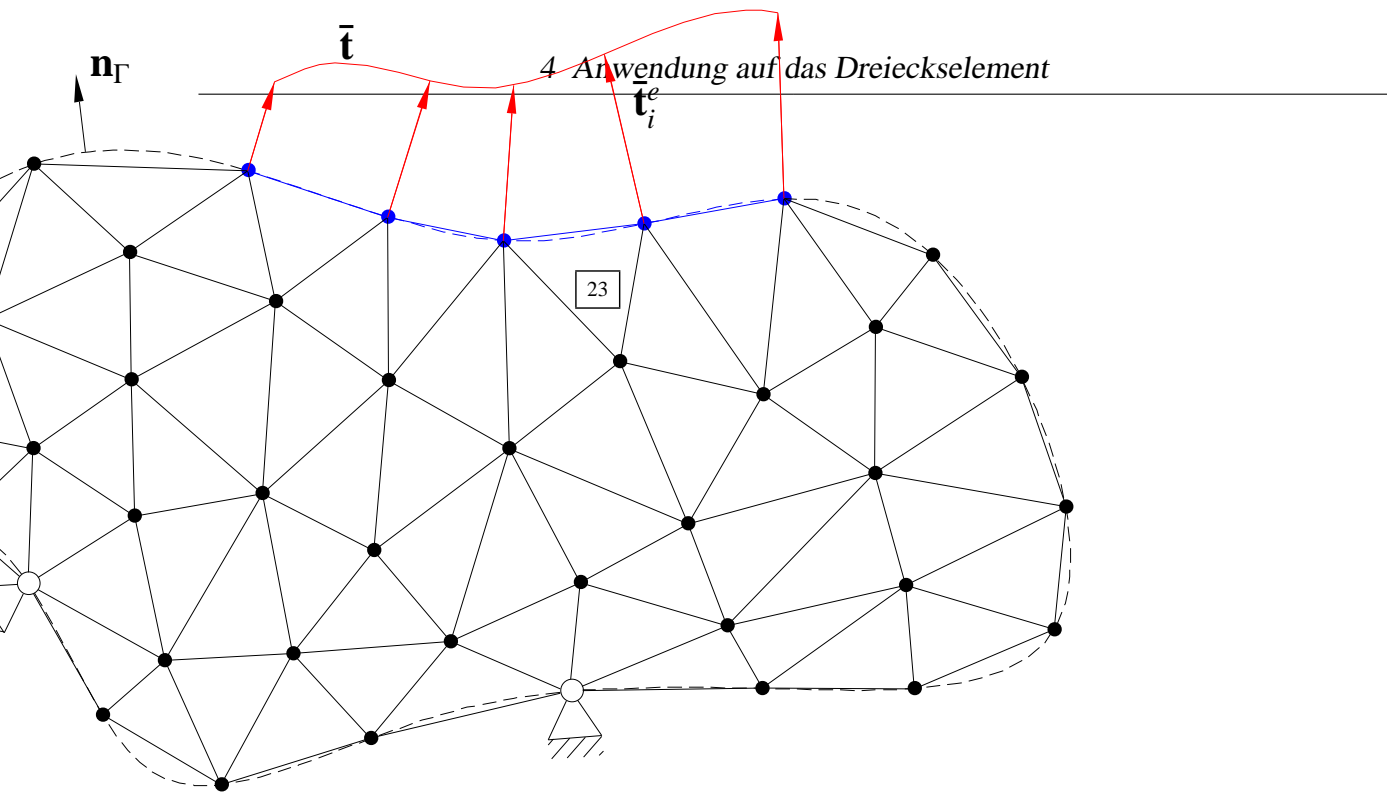


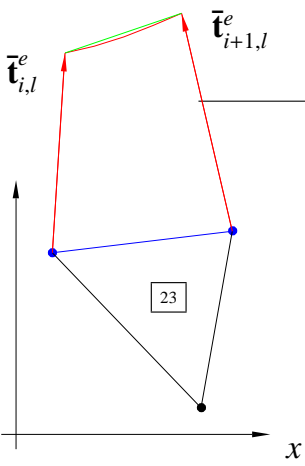
Abbildung 4.3: Äußere Lastaufbringung

Für die Ermittlung des Elementlastvektors \mathbf{f}^e ist es nun notwendig, das Integral (4.20) für jede Kante auf dem Rand Γ_t auszuwerten. Noch sind aber nur die Kraftdichtevektoren $\bar{\mathbf{t}}_i^e$ in den Knoten bekannt. D.h., es ist notwendig diese Knotenkraftdichten über die Kanten zu interpolieren. Dafür verwende ich die selben Formfunktionen wie für die Verschiebungen, lediglich die Interpolationsmatrix \mathbf{N} muss für die Kante neu erzeugt werden.

Wie für die Verschiebungen in (4.4) interpolieren wir die Knotenkraftdichtevektoren in folgender Form (wobei k sind die Menge aller Knoten auf einer Kante eines Elements) für je eine Kante l des Elements

$$\hat{\mathbf{t}}_l^e(\mathbf{x}) = \sum_{k=0}^M N_k(\mathbf{x}) \bar{\mathbf{t}}_k^e \quad (4.20)$$

wie bereits erwähnt bringen wir dies wieder in die Form einer Matrix-Vektor-Multiplikation.



4 Anwendung auf das Dreieckselement

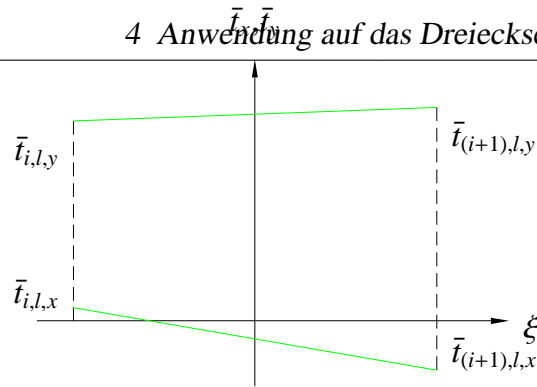


Abbildung 4.4: Interpolation und Integration der Elementkraftdichte $\bar{\mathbf{t}}^e$

$$\hat{\bar{\mathbf{t}}}_l^e(\xi) = \underbrace{\begin{bmatrix} N_0(\xi) & 0 & M(\xi) & 0 & & M(\xi) & N & 0 \\ 0 & N_0(\xi) & 0 & M(\xi) & & & & Q_M(\xi) \end{bmatrix}}_{\mathbf{N}^e \text{ mit Spaltenanzahl } 2M} \underbrace{\begin{bmatrix} \bar{t}_{0,x} \\ \bar{t}_{0,y} \\ \bar{t}_{1,x} \\ \bar{t}_{1,y} \\ \vdots \\ \bar{t}_{M,x} \\ \bar{t}_{M,y} \end{bmatrix}}_{\hat{\bar{\mathbf{t}}}_l^e} \quad (4.21)$$

Wie auch bei den Verschiebungen nähere ich $\bar{\mathbf{t}}_l^e$ mit $\hat{\bar{\mathbf{t}}}_l^e$ an und führe diese über die Kante interpoliert Kraftdichte in das Integral (4.20) ein

$$\hat{\mathbf{f}}_l^e = \int_{\xi} \mathbf{N}^{eT} \mathbf{N}^e \hat{\bar{\mathbf{t}}}_l^e d\xi \quad (4.22)$$

Die Auswertung dieses Integrals in kartesischen Koordinaten ist mühsam und kann einem verallgemeinerten numerischen Integrierer nicht übergeben werden. Daher erfolgt auch hier wieder die bereits in Abb 4.4 angedeutete Abbildung der einparametrischen Randkurve (Kante) im \mathbb{R}^2 auf das Referenzintervall $\xi \in [-1, +1]$ im $^1\mathbb{R}^E$ s gilt $d\mathbf{x} = h d\xi$, wobei $d\xi$ die differentielle

Länge der Kante darstellt (siehe [3])

$$dl = \det \mathbf{J} \, dr \quad \text{mit} \quad \det \mathbf{J} = \left[\left(\frac{\partial x}{\partial r} \right)^2 + \left(\frac{\partial y}{\partial r} \right)^2 \right]^{\frac{1}{2}} \quad (4.23)$$

daraus ergibt sich nun das auswertbare Integral

$$\hat{\mathbf{f}}_l^e = \int_0^1 \mathbf{N}^{eT}(\xi) \mathbf{N}(\xi) \hat{\mathbf{f}}_l^e(\xi) h \det \frac{\mathbf{J}}{\partial} d\xi \quad (4.24)$$

Diese Kantenlastvektoren $\hat{\mathbf{f}}_l^e$ müssen noch zu einem Elementlastvektor $\mathbf{f}^e = \hat{\mathbf{f}}^e$ assembliert werden. Dieser Vorgang muss für jede Last auf Γ_l durchgeführt werden. Es kann durchaus der Fall sein, dass mehrere unterschiedliche Lasten auf einem Element und sogar auf unterschiedliche Kanten wirken.

4.6 Berechnung der Knotenspannungen

Wie theoretisch bereits in Kapitel 2.2 angeführt ist es für die hier implementierte Fehlerschätzung notwendig, Knotenspannungen aus den in den Gausspunkten bekannten Spannungen (FE Lösung) zu ermitteln. Bei der beschriebenen Methode ist folgendes Gleichungssystem zu lösen

$$\int_{\Omega} \mathbf{N}^T \mathbf{N} \, d\Omega \, \bar{\boldsymbol{\sigma}}^* = \int_{\Omega} \mathbf{N}^T \mathbf{D} \mathbf{B} \, d\Omega \, \mathbf{a}$$

$$\mathbf{A} \, \bar{\boldsymbol{\sigma}}^* = \mathbf{r} \quad (4.25)$$

mit der Projektionskoeffizientenmatrix \mathbf{A} und dem Projektionslastvektor \mathbf{r} , auf deren Berechnung in Folge noch näher eingegangen wird.

4.6.1 Projektionskoeffizientenmatrix \mathbf{A}

Hierfür muss man noch einmal kurz auf die Gestalt der hier verwendeten Interpolationsmatrix $\mathbf{N} = \mathbf{N}$ eingehen, welche verwendet wird, um die Knotenspannungen zu interpolieren. In meinem Fall hat dies folgende Gestalt

$$\boldsymbol{\sigma}^{*e}(r, s) = \underbrace{\begin{bmatrix} N_1 & N_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{N}^e(r, s)} \underbrace{\begin{bmatrix} xx,1 \\ xx,2 \\ \vdots \\ xx,(n) \\ yy,1 \\ yy,2 \\ \vdots \\ yy,(n) \\ xy,1 \\ xy,2 \\ \vdots \\ xy,(n) \end{bmatrix}}_{\bar{\boldsymbol{\sigma}}^{*e}} \quad (4.26)$$

daraus lässt sich für ein Element die Matrix \mathbf{A}^e angeben

$$\mathbf{A}^e = \int_{\Omega^e} \mathbf{N}^e T \mathbf{N}^e h d(x, y) = \int \mathbf{N}^e T(r, s) \mathbf{N}^e(r, s) h \det \mathbf{J} \frac{\partial x}{\partial r} d(r, s) \quad (4.27)$$

Diese Matrix \mathbf{A}^e hat (hier beispielhaft für das lineare Dreieckselement) folgende Gestalt

$$\begin{vmatrix} & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & & & & & & & & 0 \\ 0 & 0 & 0 & & & & & & & & 0 \\ 0 & 0 & 0 & & & & & & & & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & & 0 \end{vmatrix}$$

Sowohl die besetzten Einzelblöcke als auch die gesamte Matrix sind vom Typ

$$\tilde{\mathbf{A}} = \int_{\Omega} \tilde{\mathbf{N}}^T \mathbf{c} \tilde{\mathbf{N}} d\Omega \quad (4.28)$$

wobei hier in unserem Fall $\mathbf{c} = \mathbf{I} h \det \mathbf{J} \frac{\partial x}{\partial r}$ ist. Derartige Matrizen sind z.B. auch als Mass-matrizen häufiger anzufinden. Das Integral kann mittels verschiedener Verfahren durch eine Diagonalisierung (auch *lumping* genannt) angenähert werden (siehe [15]).

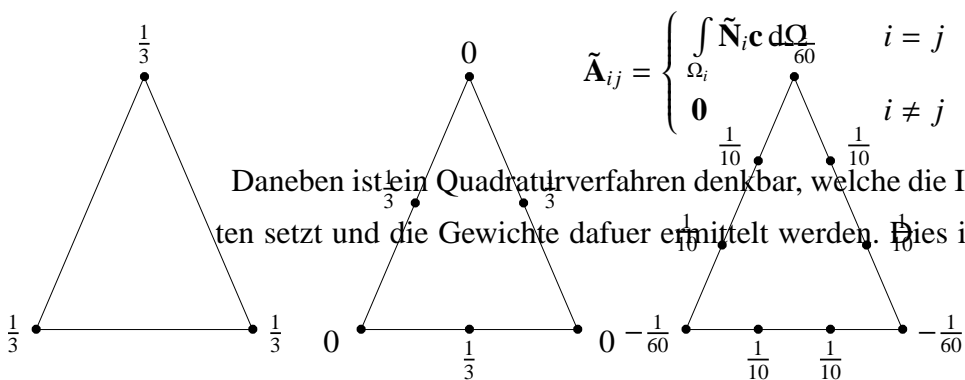
Die Formfunktionen $\tilde{\mathbf{N}}$ werden durch stückweise konstante Formfunktionen ersetzt. Diese gewählten Formfunktionen müssen lediglich folgenden Bedingungen genügen

$$\tilde{\mathbf{N}}_i = \tilde{N}_i \mathbf{I} \quad \text{mit} \quad \sum_i \tilde{N}_i = 1$$

damit wird

$$\tilde{\mathbf{A}}_{ij} = \begin{cases} \int_{\Omega_i} \mathbf{c} \, d\Omega & i = j \\ \mathbf{0} & i \neq j \end{cases} \quad (4.29)$$

Versteht man (4.29) als Bedingung zur Erhaltung der Menge \mathbf{c} über das Element (z.B. die Masse im Fall der Massematrix), so liegt die Überlegung nahe, dass die Verteilung von \mathbf{c} über das Element der Gestalt zu erfolgen hat, dass das Integral über das Element erhalten bleibt. Diese Überlegung führt zur "Reihensummenmethode", welche vereinfacht angegeben werden kann (wenn die Freiheitsgrade der Knoten alle die selbe physikalische Bedeutung haben) mit



Daneben ist ein Quadraturverfahren denkbar, welche die Integrationspunkte in die Knoten setzt und die Gewichte dafür ermittelt werden. Dies ist auch jenes Verfahren, das

Abbildung 4.5: Knotengewichte bei Knotenquadratur am Dreieck

auf Grund seiner einfachen Anwendbarkeit in unserem Fall für lineare Dreieckselemente zum Einsatz gekommen ist.

Alle diese Verfahren münden in eine Diagonalmatrix \mathbf{A}^e , die geeignet zu einer globalen diagonalisierten Projektionskoeffizientenmatrix \mathbf{A} assembliert werden muss, und mit deren Hilfe

sich die Auswertung von (4.26) sehr einfach gestaltet. Natürlich wäre auch eine gewöhnliche numerische Integration möglich. Diese bedeutet lediglich einen erhöhten Rechenaufwand, da die Elemente der Matrix \mathbf{A}^e den doppelten Polynomgrad der Formfunktionen aufweisen und daher entsprechend viele Integrationspunkte gewählt werden müssen.

4.6.2 Projektionslastvektor \mathbf{r}

Dieser lässt sich bereits mit der in 4.6.1 angegebenen Interpolationsmatrix und der gewohnten Abbildung auf das Referenzelement berechnen mit

$$\begin{aligned} \mathbf{r}^e &= \int_{\Omega^e} \mathbf{N}^{eT} \mathbf{D} \mathbf{B}^e h \, d(x, y) \mathbf{a}^e \\ &= \int \mathbf{N}^{eT}(r, s) \mathbf{D} \mathbf{B}^e(r, s) h \det \mathbf{J} \frac{\partial \mathbf{x}}{\partial \mathbf{r}} \, d(r, s) \mathbf{a}^e \end{aligned} \quad (4.30)$$

Dieser für jedes Element ermittelte Projektionslastvektor \mathbf{r}^e muss noch geeignet zum globalen Projektionslastvektor \mathbf{r} assembliert werden.

4.7 Energienormen

Für das vorgestellte Verfahren sind die beiden Energienormen $\|\mathbf{e}\|$ (bzw. deren Abschätzung ${}^0\|\mathbf{e}\|$) und $\|\hat{\mathbf{u}}\|$ von Interesse. Es wurde bereits in (2.3) gezeigt, dass die globale Norm aus der Quadratsumme der Einzelnormen berechnet werden kann.

4.7.1 Berechnung der Elementfehlnorm

Aus diesem Grund führe ich hier nur mehr die Berechnung des Normquadrates für ein Element an.

$$\|\mathbf{e}^e\|^2 = \int_{\Omega^e} \mathbf{e}^{eT} \mathbf{D} \mathbf{e}^e h \, d(x, y) \quad (4.31)$$

Die Auswertung dieses Integrals erfolgt mit Hilfe numerischer Integration. D.h. an den Integrationspunkten muss der Werte von \mathbf{e}^e berechnet werden. Das ist auch deswegen sinnvoll,

weil nur in den Integrationspunkten (Gaußpunkten) die Spannungen $\hat{\boldsymbol{\sigma}}^e(r, s)$ aus der FE Lösung bekannt sind.

$$\begin{aligned} \mathbf{e}^e &= \boldsymbol{\sigma}^e \hat{\boldsymbol{\sigma}}^e \\ \mathbf{e}^e \quad {}^0\mathbf{e}^e &= \boldsymbol{\sigma}^{*e} \hat{\boldsymbol{\sigma}}^e = \mathbf{N}^e(r, s) \bar{\boldsymbol{\sigma}}^{*e} \hat{\boldsymbol{\sigma}}^e(r, s) \end{aligned} \quad (4.32)$$

damit lässt sich nun mit Hilfe der “geschätzten” Knotenspannungen und der Abbildung auf das Referenzelement das Quadrat der Elementfehlnorm angeben zu

$${}^0\|\mathbf{e}^e\|^2 = \int {}^0\mathbf{e}^e(r, s)^T \mathbf{D}^{-1} {}^0\mathbf{e}^e(r, s) h \det \mathbf{J}_{\frac{\partial \mathbf{x}}{\partial \mathbf{r}}} d(r, s) \quad (4.33)$$

woraus sich die Abschätzung der globalen Fehlnorm (analog zu (2.3)) aus der Summe über die Elemente e ergibt

$${}^0\|\mathbf{e}\| = \left(\sum_{e=1}^m {}^0\|\mathbf{e}^e\|^2 \right)^{\frac{1}{2}} \quad (4.34)$$

4.7.2 Berechnung der Verzerrungsenergie

Hier kann neben der elementweisen Berechnung auch die globale Berechnung mit Hilfe der globalen Steifigkeitsmatrix erfolgen.

$$\begin{aligned} \|\hat{\mathbf{u}}\| &= \left(\int_{\Omega} \hat{\boldsymbol{\varepsilon}}^T \mathbf{D} \hat{\boldsymbol{\varepsilon}} d\Omega \right)^{\frac{1}{2}} \\ &= \mathbf{a}^T \mathbf{K} \mathbf{a}^{\frac{1}{2}} \end{aligned} \quad (4.35)$$

Elementweise ergibt sich ganz analog

$$\|\hat{\mathbf{u}}^e\| = \mathbf{a}^{eT} \mathbf{K}^e \mathbf{a}^e^{\frac{1}{2}} \quad (4.36)$$

und somit die globale Verzerrungsenergie

$$\|\hat{\mathbf{u}}\| = \left(\sum_{e=1}^m {}^0\|\hat{\mathbf{u}}^e\|^2 \right)^{\frac{1}{2}} \quad (4.37)$$

5 Literatur

- [1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK Users' Guide, 3rd ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.
- [2] G. Archer, G. Fenves, C. Thewalt, A new object-oriented finite element analysis program architecture, *Computers and Structures* 70 (1999) 63–75.
- [3] K.-J. Bathe, *Finite Element Procedures*, Prentice-Hall, 1996.
- [4] C. Celigoj, *Methode der Finiten Elemente*, Institut für Festigkeitslehre, Graz (1998).
- [5] C. Celigoj, *Festigkeitslehre*, Institut für Festigkeitslehre, Graz (2004).
- [6] G. Cowper, Gaussian Quadrature Formulas For Triangles, *Internat. J. Num. Meth. Eng.* 7 (1973) 405–408.
- [7] J. Dongarra, Basic Linear Algebra Subprograms Technical Forum Standard, *International Journal of High Performance Applications and Supercomputing* 16 (1) (2002) 1–111.
- [8] J. Dongarra, Basic Linear Algebra Subprograms Technical Forum Standard, *International Journal of High Performance Applications and Supercomputing* 16 (2) (2002) 115–199.
- [9] E. Gamma, R. Helm, R. Johnson, J. Vlissides, *Design Patterns - Elements of Reusable Object-Oriented Software*, Addison Wesley, 1995.
- [10] K. Girkmann, *Flächentragwerke*, Sprinter-Verlag, 1956.
- [11] H. Goering, H.-G. Roos, L. Tobiska, *Finite-Element-Methode*, Akademie Verlag, 1993.
- [12] H. Parkus, *Mechanik der festen Körper*, Springer, 1998.
- [13] B. Stroustrup, *The C++ Programming Language - Special Edition*, AT&T, 2006.
- [14] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, *Numerical Recipes in C++*, Cambridge University Press, 2005.

- [15] O. Zienkiewicz, R. Taylor, *The Finite Element Method - The Basis*, vol. 1, 5th ed., Butterworth-Heinemann, 2000.
- [16] O. Zienkiewicz, J. Zhu, A simple error estimator and adaptive procedure for practical engineering analysis, *Internat. J. Num. Meth. Eng.* 24 (2) (1987) 337–357.

A Anhang - Beispiele

In diesem Kapitel stellen wir einige repräsentative Beispiele als Benchmark für den Fehlerschätzer und das adaptive Verfahren vor.

A.1 Dickwandiger Zylinder unter Innendruck

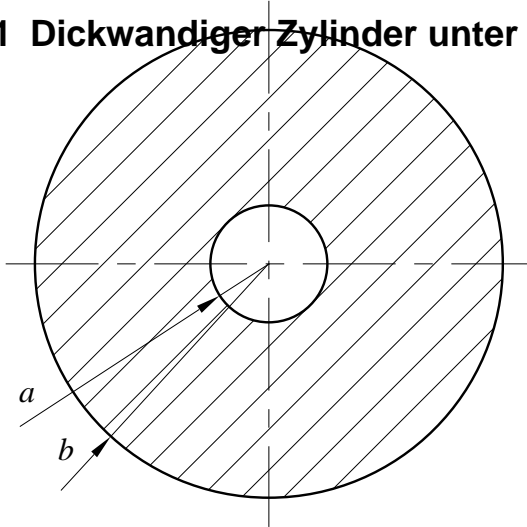


Abbildung A.1: Dickwandiger Zylinder unter Innendruck

Auf Grund der Rotationssymmetrie der Geometrie und der Belastung wird nur ein Teilstück berechnet. In diesem Fall haben wir uns (auf Grund der einfach zu realisierenden Randbedingungen) für das Viertelsystem entschieden welches in Abbildung A.2 dargestellt ist.

Bei der Vernetzung wurde spezielle Rücksicht auf die Symmetrie des Netzes bezüglich des ersten Medians genommen. Dies ergibt symmetrische Ergebnisse um diese Achse und kann sehr leicht zur Kontrolle verwendet werden.

Für dieses ebene Scheibenproblem existiert eine analytische Lösung (siehe [10]) des Spannungsproblems

$$r_r = \frac{p}{h} \frac{b^2}{b^2 - a^2} \left(1 - \frac{a^2}{r^2} \right) \quad (\text{A.1})$$

$$\varphi_\varphi = \frac{p}{h} \frac{b^2}{b^2 - a^2} \left(1 + \frac{a^2}{r^2} \right) \quad (\text{A.2})$$

Der exakte Spannungsverlauf ist in Abbildung A.3 dargestellt.

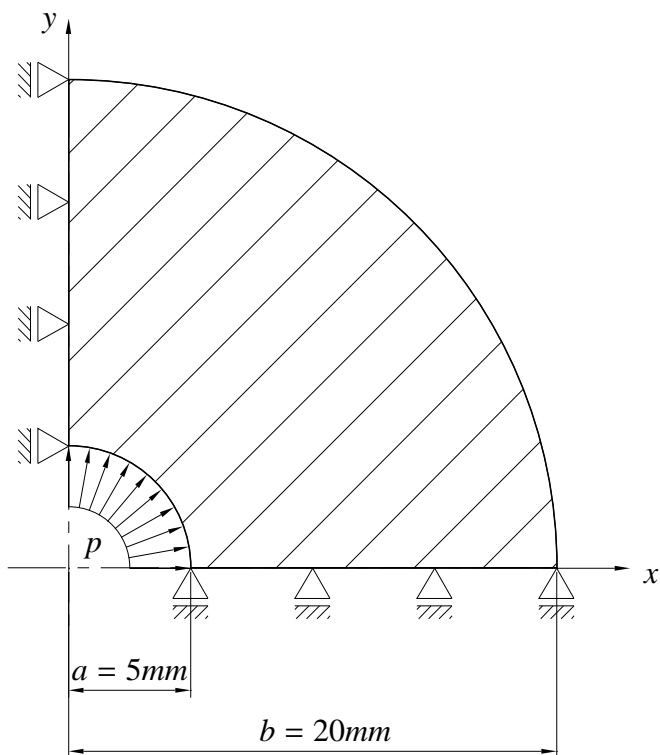


Abbildung A.2: Modellproblem - Dickwandiger Zylinder unter Innendruck

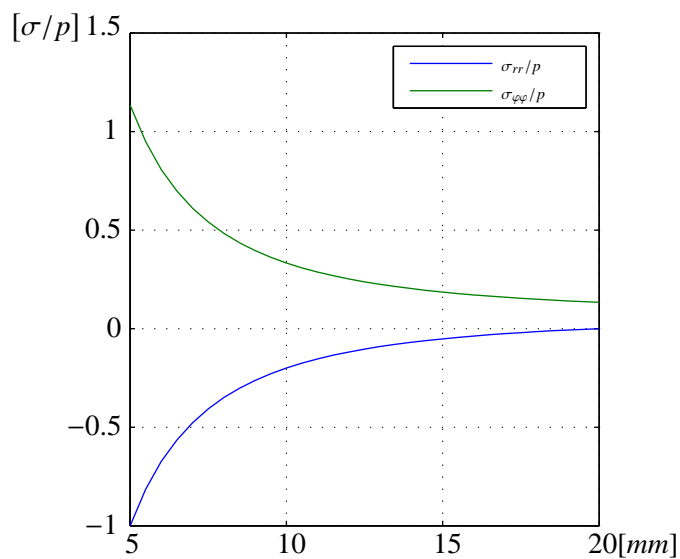


Abbildung A.3: Analytischer Spannungsverlauf

A.1.1 Ergebnisse

(a)	(b)
=	=
59.7605%	35.3308%
18	50
Frei-	Frei-
heits-	heits-
gra-	gra-
de	de
(c)	(d)
=	=
20.1154%	10.3689%
162	548
Frei-	Frei-
heits-	heits-
gra-	gra-
de	de
(e)	(f)
=	Netz
6.0104%	nach
719	dem
Frei-	letz-
heits-	ten
gra-	Ad-
de	ap-
	ti-
	ons-
	schrift

Abbildung A.4: Verfeinerungsfaktor für Dreieckselement 1. Ordnung - $\epsilon_{max} = 7\%$

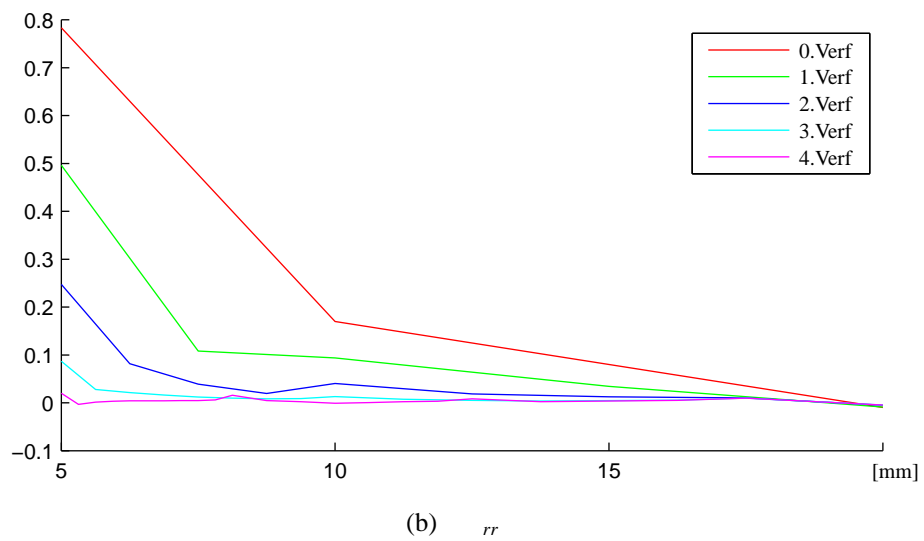
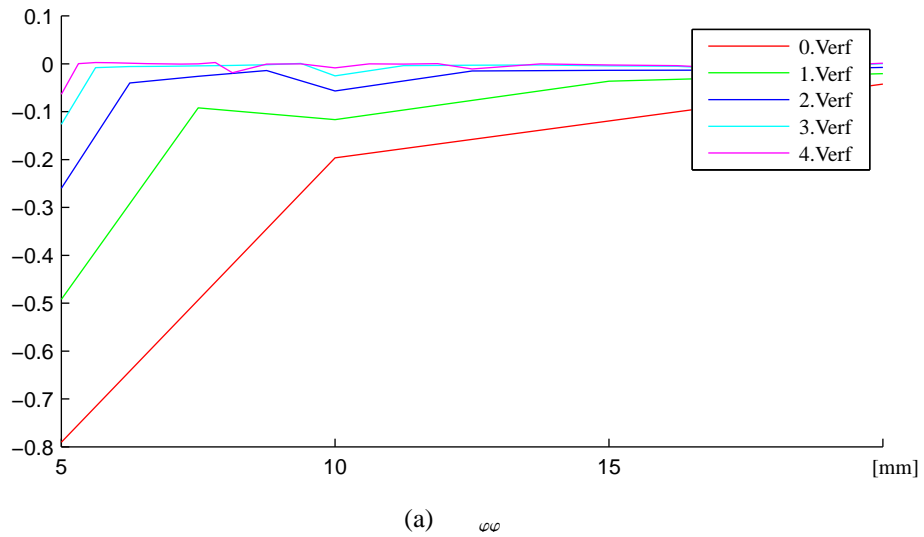
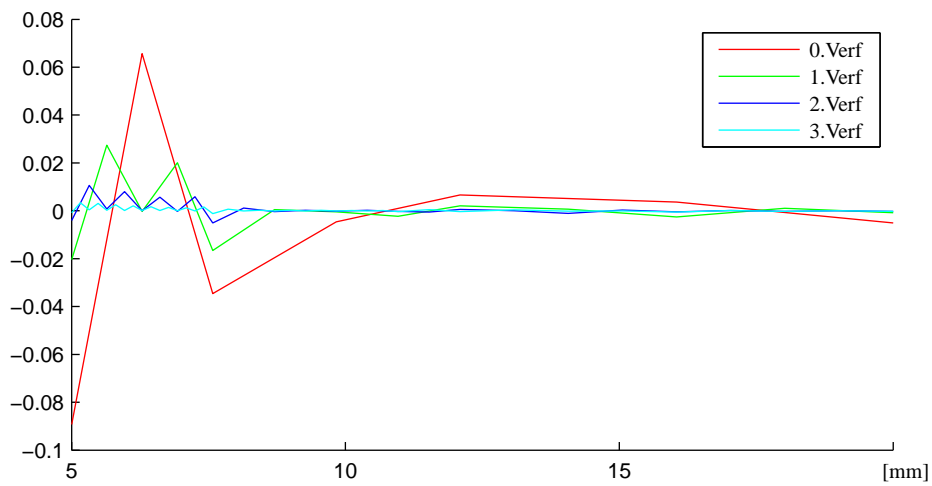


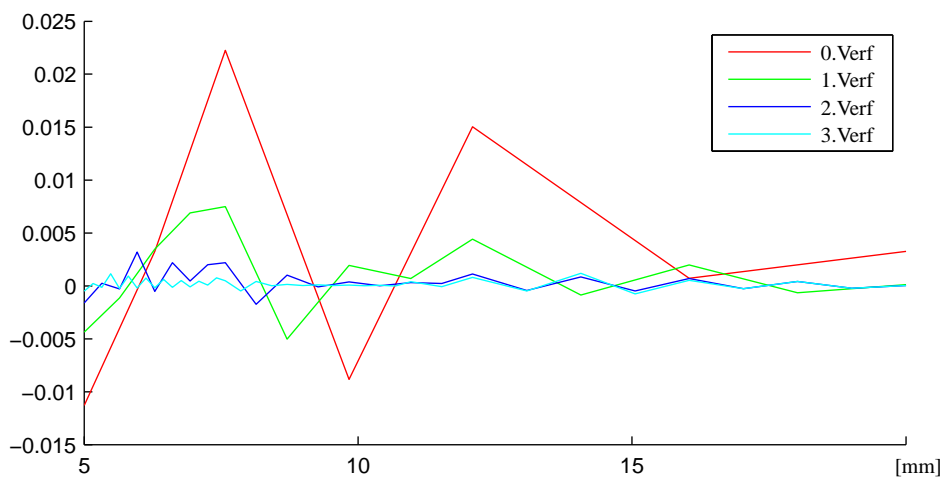
Abbildung A.5: Spannungsdiﬀerenz $\sigma_{FE} - \sigma_{analytisch}$ zwischen FE Lösung und analytischer Lösung je Verfeinerungsschritt - Dreieckselement 1. Ordnung

(a)		(b)
=		=
6.2739%		1.9469%
112		390
Frei-		Frei-
heits-		heits-
gra-		gra-
de		de
(c)		(d)
=		=
0.5353%		0.1544%
1450		4622
Frei-		Frei-
heits-		heits-
gra-		gra-
de		de
	(e)	
	Netz	
	nach	
	dem	
	letz-	
	ten	
	Ad-	
	ap-	
	ti-	
	ons-	
	schrift	

Abbildung A.6: Verfeinerungsfaktor für Dreieckselement 2. Ordnung - $\epsilon_{max} = 0.2\%$



(a) $\varphi\varphi$



(b) rr

Abbildung A.7: Spannungsdi fferenz $=_{FE} \text{ analytisch}$ zwischen FE L osung und analytischer L osung je Verfeinerungsschritt - Dreieckselement 2. Ordnung

A.1.2 Eingabedaten

Elemente 1. Ordnung thick_cylinder_1.xml

```

<?xml version="1.0"?>
<problem>
  <global id="thick_cylinder_1" dim="2" coord_sys_type="cartesian"/>
  <refine adaptivity="1" max_percentage_error="0.07" max_refinement_depth="0"/>
  <node>
    <n x="5.0000000000" y="0.0000000000">1</n>
    <n x="3.5355339059" y="3.5355339059">2</n>
    <n x="0.0000000000" y="5.0000000000">3</n>
    <n x="10.0000000000" y="0.0000000000">4</n>
    <n x="7.0710678119" y="7.0710678119">5</n>
    <n x="0.0000000000" y="10.0000000000">6</n>
    <n x="20.0000000000" y="0.0000000000">7</n>
    <n x="14.1421356237" y="14.1421356237">8</n>
    <n x="0.0000000000" y="20.0000000000">9</n>
  </node>
  <boundary>
    <b id="line" gnn="1_4_7" start="1" end="7">1</b>
    <b id="arc" gnn="7_8_9" start="7" end="9">2</b>
    <b id="line" gnn="9_6_3" start="9" end="3">3</b>
    <b id="arc" gnn="3_2_1" start="3" end="1">4</b>
  </boundary>
  <element>
    <e gnn="1_4_5" m="1" et="1">1</e>
    <e gnn="1_5_2" m="1" et="1">2</e>
    <e gnn="3_2_5" m="1" et="1">3</e>
    <e gnn="3_5_6" m="1" et="1">4</e>
    <e gnn="4_7_8" m="1" et="1">5</e>
    <e gnn="4_8_5" m="1" et="1">6</e>
    <e gnn="6_5_8" m="1" et="1">7</e>
    <e gnn="6_8_9" m="1" et="1">8</e>
  </element>
  <element_type>
    <et id="triangle" real_const_set="2" gauss_points="1">1</et>
  </element_type>
  <constraint>
    <c id="boundary" b="1" y="0.0">1</c>
    <c id="boundary" b="3" x="0.0">2</c>
  </constraint>
  <load>
    <l id="pressure" b="4" value="1.0">1</l>
  </load>
  <material>
    <m id="linear_elastic" real_const_set="1">1</m>
  </material>
  <real_const_set>
    <rsc data="210000.0:0.3">1</rsc>
  </real_const_set>

```

```

    <rcs data="1:0:1.0">2</rcs>
  </real_const_set>
</problem>

```

Elemente 2. Ordnung thick_cylinder_2.xml

```

<?xml version="1.0"?>
<problem>
  <global id="thick_cylinder_2" dim="2" coord_sys_type="cartesian"/>
  <refine adaptivity="1" max_percentage_error="0.002" max_refinement_depth="2"/>
  <node>
    <n x="0.0000000000e+00" y="2.0000000000e+01">1</n>
    <n x="0.0000000000e+00" y="5.0000000000e+00">2</n>
    <n x="0.0000000000e+00" y="1.60455117534e+01">3</n>
    <n x="0.0000000000e+00" y="1.20910235069e+01">4</n>
    <n x="0.0000000000e+00" y="9.83386343594e+00">5</n>
    <n x="0.0000000000e+00" y="7.57670336499e+00">6</n>
    <n x="0.0000000000e+00" y="6.28835168250e+00">7</n>
    <n x="5.0000000000e+00" y="0.0000000000e+00">8</n>
    <n x="1.29409522552e+00" y="4.82962913144e+00">9</n>
    <n x="2.50000000053e+00" y="4.33012701862e+00">10</n>
    <n x="3.53553390679e+00" y="3.53553390507e+00">11</n>
    <n x="4.33012701984e+00" y="2.49999999842e+00">12</n>
    <n x="4.82962913208e+00" y="1.29409522316e+00">13</n>
    <n x="2.00000000000e+01" y="0.00000000000e+00">14</n>
    <n x="6.28835168250e+00" y="0.00000000000e+00">15</n>
    <n x="7.57670336499e+00" y="0.00000000000e+00">16</n>
    <n x="9.83386343594e+00" y="0.00000000000e+00">17</n>
    <n x="1.20910235069e+01" y="0.00000000000e+00">18</n>
    <n x="1.60455117534e+01" y="0.00000000000e+00">19</n>
    <n x="1.96157056080e+01" y="3.90180644081e+00">20</n>
    <n x="1.84775906493e+01" y="7.65366864944e+00">21</n>
    <n x="1.66293922438e+01" y="1.11114046638e+01">22</n>
    <n x="1.41421356195e+01" y="1.41421356279e+01">23</n>
    <n x="1.11114046540e+01" y="1.66293922504e+01">24</n>
    <n x="7.65366863846e+00" y="1.84775906539e+01">25</n>
    <n x="3.90180642916e+00" y="1.96157056103e+01">26</n>
    <n x="2.20366172457e+00" y="1.61116903342e+01">27</n>
    <n x="1.61116903333e+01" y="2.20366172595e+00">28</n>
    <n x="4.52772994301e+00" y="6.63810576286e+00">29</n>
    <n x="6.63810576283e+00" y="4.52772994299e+00">30</n>
    <n x="3.48239383026e+00" y="9.50079825941e+00">31</n>
    <n x="2.52873210596e+00" y="5.55417143447e+00">32</n>
    <n x="1.27873210569e+00" y="7.17745960766e+00">33</n>
    <n x="1.27873210569e+00" y="5.88910792516e+00">34</n>
    <n x="9.50079825883e+00" y="3.48239383125e+00">35</n>
    <n x="7.17745960801e+00" y="1.27873210530e+00">36</n>
    <n x="5.55417143543e+00" y="2.52873210450e+00">37</n>
    <n x="5.88910792551e+00" y="1.27873210530e+00">38</n>
  </node>
</problem>

```

```

<n x="1.53504856580e+01" y="6.03049605067e+00">39</n>
<n x="6.03049604380e+00" y="1.53504856612e+01">40</n>
<n x="9.27472953434e+00" y="1.31827581482e+01">41</n>
<n x="1.03200656471e+01" y="1.03200656517e+01">42</n>
<n x="1.31827581431e+01" y="9.27472953992e+00">43</n>
<n x="5.45265956188e+00" y="9.36068817194e+00">44</n>
<n x="9.36068817063e+00" y="5.45265956365e+00">45</n>
<n x="9.90004201581e+00" y="2.20366172595e+00">46</n>
<n x="1.21572020868e+01" y="2.20366172595e+00">47</n>
<n x="2.20366172457e+00" y="1.21572020877e+01">48</n>
<n x="2.20366172457e+00" y="9.90004201674e+00">49</n>
<n x="4.49899783758e+00" y="5.41406134700e+00">50</n>
<n x="5.41406134723e+00" y="4.49899783690e+00">51</n>
<n x="6.49799567463e+00" y="6.49799567539e+00">52</n>
<n x="4.40732344914e+00" y="1.22233806685e+01">53</n>
<n x="1.22233806666e+01" y="4.40732345191e+00">54</n>
<n x="6.77821585103e+00" y="2.55746421059e+00">55</n>
<n x="2.55746421138e+00" y="6.77821585033e+00">56</n>
</node>
<boundary>
  <b id="line" gnn="1_3_4_5_6_7_2" start="1" end="2">1</b>
  <b id="arc" gnn="2_9_10_11_12_13_8" start="2" end="8">2</b>
  <b id="line" gnn="8_15_16_17_18_19_14" start="8" end="14">3</b>
  <b id="arc" gnn="14_20_21_22_23_24_25_26_1" start="14" end="1">4</b>
</boundary>
<element>
  <e gnn="53_1_4_27_3_48_" m="1" et="1">1</e>
  <e gnn="53_25_1_40_26_27_" m="1" et="1">2</e>
  <e gnn="54_14_21_28_20_39_" m="1" et="1">3</e>
  <e gnn="54_18_14_47_19_28_" m="1" et="1">4</e>
  <e gnn="52_56_10_29_32_50_" m="1" et="1">5</e>
  <e gnn="52_53_56_44_31_29_" m="1" et="1">6</e>
  <e gnn="55_52_12_30_51_37_" m="1" et="1">7</e>
  <e gnn="55_54_52_35_45_30_" m="1" et="1">8</e>
  <e gnn="6_56_53_33_31_49_" m="1" et="1">9</e>
  <e gnn="56_2_10_34_9_32_" m="1" et="1">10</e>
  <e gnn="6_2_56_7_34_33_" m="1" et="1">11</e>
  <e gnn="54_55_16_35_36_46_" m="1" et="1">12</e>
  <e gnn="55_8_16_38_15_36_" m="1" et="1">13</e>
  <e gnn="12_8_55_13_38_37_" m="1" et="1">14</e>
  <e gnn="21_23_54_22_43_39_" m="1" et="1">15</e>
  <e gnn="23_25_53_24_40_41_" m="1" et="1">16</e>
  <e gnn="53_52_23_44_42_41_" m="1" et="1">17</e>
  <e gnn="52_54_23_45_43_42_" m="1" et="1">18</e>
  <e gnn="16_18_54_17_47_46_" m="1" et="1">19</e>
  <e gnn="4_6_53_5_49_48_" m="1" et="1">20</e>
  <e gnn="10_12_52_11_51_50_" m="1" et="1">21</e>
</element>
<element_type>

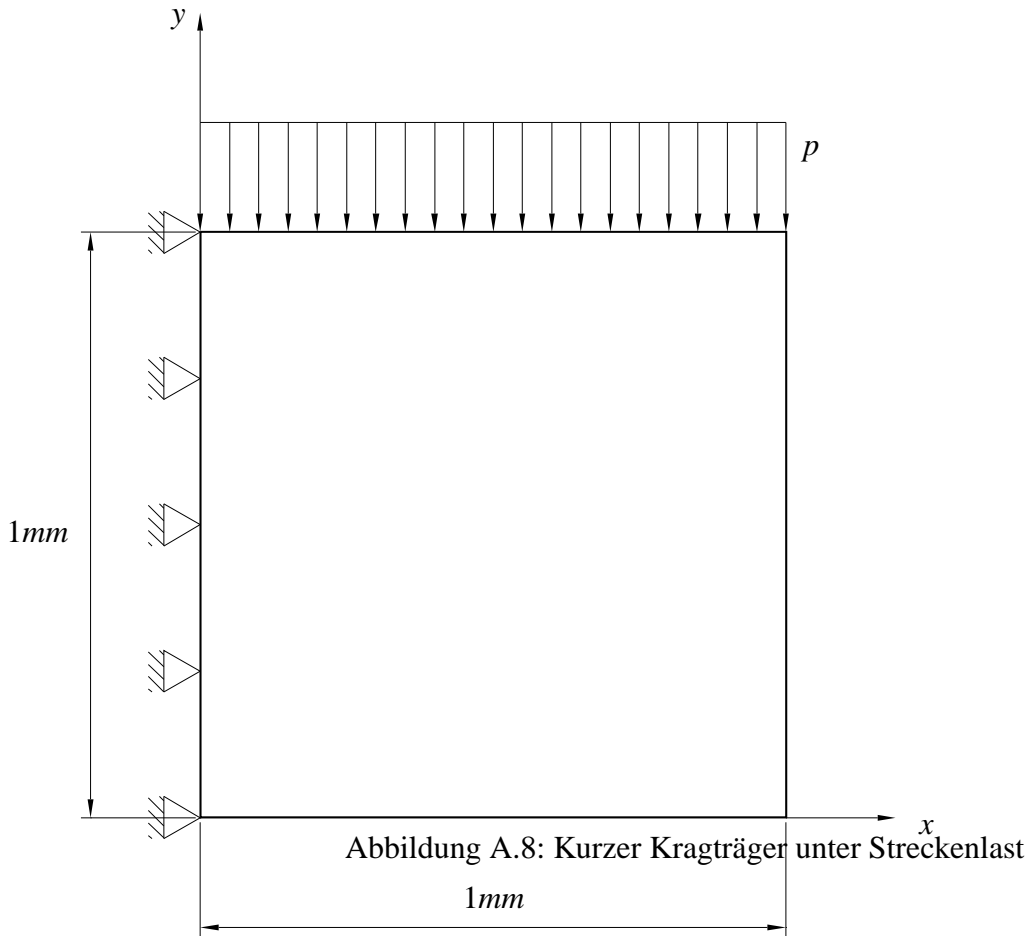
```

```
<et id="triangle" real_const_set="2" gauss_points="3">1</et>
</element_type>
<constraint>
  <c id="boundary" b="1" x="0.0">1</c>
  <c id="boundary" b="3" y="0.0">2</c>
</constraint>
<load>
  <l id="pressure" b="2" value="1.0">1</l>
</load>
<material>
  <m id="linear_elastic" real_const_set="1">1</m>
</material>
<real_const_set>
  <rcs data="210000.0:0.3">1</rcs>
  <rcs data="2:0:1.0">2</rcs>
</real_const_set>
</problem>
```

A.2 Kurzer Kragträger unter Streckenlast

Für diesen Fall kann keine Symmetrie genutzt werden. D.h. das in Abbildung A.8 dargestellt Problem ist zugleich die Basis des FE Modelles. Dieses Beispiel wurde trotz der fehlenden analytischen Lösung gewählt, weil es hier zu ausgeprägten Spannungsspitzen an den Ecken der Einspannung kommt. Dadurch ist die Wirkung der Adaptivität gut erkennbar. Dieses Beispiel wurde als ebener Verzerrungszustand gerechnet.

A.2.1 Ergebnisse



<p>(a)</p> <p>=</p> <p>45.7589%</p> <p>18</p> <p>Fhg.</p> <p>(0.Verf.)</p>	<p>(b)</p> <p>=</p> <p>36.6686%</p> <p>50</p> <p>Fhg.</p> <p>(1.Verf.)</p>
<p>(c)</p> <p>=</p> <p>25.0407%</p> <p>158</p> <p>Fhg.</p> <p>(2.Verf.)</p>	<p>(d)</p> <p>=</p> <p>15.8857%</p> <p>488</p> <p>Fhg.</p> <p>(3.Verf.)</p>
<p>(e)</p> <p>=</p> <p>5.73397%</p> <p>2536</p> <p>Fhg.</p> <p>(6.Verf.)</p>	<p>(f)</p> <p>Netz</p> <p>nach</p> <p>dem</p> <p>letz-</p> <p>ten</p> <p>Ad-</p> <p>ap-</p> <p>ti-</p> <p>ons-</p> <p>schrift</p>

Abbildung A.9: Verfeinerungsfaktor für Dreieckselement 1. Ordnung - $\epsilon_{max} = 7\%$

(a)	(b)
=	=
11.3574%	7.07149%
50	162
Fhg.	Fhg.
(1.Verf.)	(2.Verf.)
(c)	(d)
=	=
4.22327%	2.59091%
550	988
Fhg.	Fhg.
(3.Verf.)	(4.Verf.)
(e)	(f)
=	Netz
0.79056%	nach
1980	dem
Fhg.	letz-
(7.Verf.)	ten
	Ad-
	ap-
	ti-
	ons-
	schrift

Abbildung A.10: Verfeinerungsfaktor für Dreieckselement 2. Ordnung - $\epsilon_{max} = 1\%$

A.2.2 Eingabedaten

Elemente 1. Ordnung short_cantilever_1.xml

```

<?xml version="1.0"?>
<problem>
  <global id="short_cantilever_1" dim="2" coord_sys_type="cartesian"/>
  <refine adaptivity="1" max_percentage_error="0.07" max_refinement_depth="0"/>
  <node>
    <n x="0.0" y="0.0">1</n>
    <n x="0.5" y="0.0">2</n>
    <n x="0.0" y="0.5">3</n>
    <n x="0.5" y="0.5">4</n>
    <n x="1.0" y="0.0">5</n>
    <n x="1.0" y="0.5">6</n>
    <n x="0.0" y="1.0">7</n>
    <n x="0.5" y="1.0">8</n>
    <n x="1.0" y="1.0">9</n>
  </node>
  <boundary>
    <b id="line" gnn="1_2_5" start="1" end="5">1</b>
    <b id="line" gnn="5_6_9" start="5" end="9">2</b>
    <b id="line" gnn="9_8_7" start="9" end="7">3</b>
    <b id="line" gnn="7_3_1" start="7" end="1">4</b>
  </boundary>
  <element>
    <e gnn="1_4_3" m="1" et="1">1</e>
    <e gnn="1_2_4" m="1" et="1">2</e>
    <e gnn="2_6_4" m="1" et="1">3</e>
    <e gnn="2_5_6" m="1" et="1">4</e>
    <e gnn="3_8_7" m="1" et="1">5</e>
    <e gnn="3_4_8" m="1" et="1">6</e>
    <e gnn="4_9_8" m="1" et="1">7</e>
    <e gnn="4_6_9" m="1" et="1">8</e>
  </element>
  <element_type>
    <et id="triangle" real_const_set="2" gauss_points="1">1</et>
  </element_type>
  <constraint>
    <c id="boundary" b="4" x="0.0" y="0.0">1</c>
  </constraint>
  <load>
    <l id="pressure" b="3" value=" 1.0">1</l>
  </load>
  <material>
    <m id="linear_elastic" real_const_set="1">1</m>
  </material>
  <real_const_set>
    <rcs data="210000.0:0.3">1</rcs>
    <rcs data="1:1:1.0">2</rcs>
  </real_const_set>

```

```
</real_const_set>
</problem>
```

Elemente 2. Ordnung short_cantilever_2.xml

```
<?xml version="1.0"?>
<problem>
  <global id="short_cantilever_2" dim="2" coord_sys_type="cartesian"/>
  <refine adaptivity="1" max_percentage_error="0.01" max_refinement_depth="0"/>
  <node>
    <n x="0.00" y="0.00">1</n>
    <n x="0.50" y="0.00">2</n>
    <n x="1.00" y="0.00">3</n>
    <n x="0.00" y="0.50">4</n>
    <n x="0.50" y="0.50">5</n>
    <n x="1.00" y="0.50">6</n>
    <n x="0.00" y="1.00">7</n>
    <n x="0.50" y="1.00">8</n>
    <n x="1.00" y="1.00">9</n>
  </node>
  <boundary>
    <b id="line" gnn="1_2_3" start="1" end="3">1</b>
    <b id="line" gnn="3_6_9" start="3" end="9">2</b>
    <b id="line" gnn="9_8_7" start="9" end="7">3</b>
    <b id="line" gnn="7_4_1" start="7" end="1">4</b>
  </boundary>
  <element>
    <e gnn="1_3_9_2_6_5" m="1" et="1">1</e>
    <e gnn="1_9_7_5_8_4" m="1" et="1">2</e>
  </element>
  <element_type>
    <et id="triangle" real_const_set="2" gauss_points="3">1</et>
  </element_type>
  <constraint>
    <c id="boundary" b="4" x="0.0" y="0.0">1</c>
  </constraint>
  <load>
    <l id="pressure" b="3" value=" 1.0">1</l>
  </load>
  <material>
    <m id="linear_elastic" real_const_set="1">1</m>
  </material>
  <real_const_set>
    <rcs data="210000.0:0.3">1</rcs>
    <rcs data="2:1:1.0">2</rcs>
  </real_const_set>
</problem>
```

A.3 Scheibe mit kreisförmigen Loch

Die Aufgabenstellung (Abbildung A.11) weist eine doppelte Symmetrie auf. Diese wird zur Modellierung ausgenutzt. Das somit entstandene Modell ist in Abbildung A.12 ersichtlich.

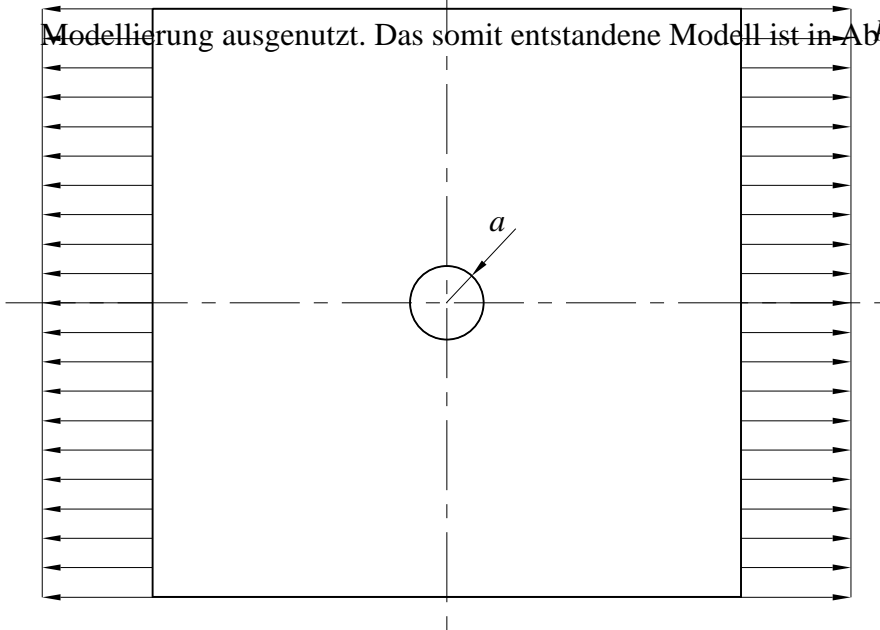
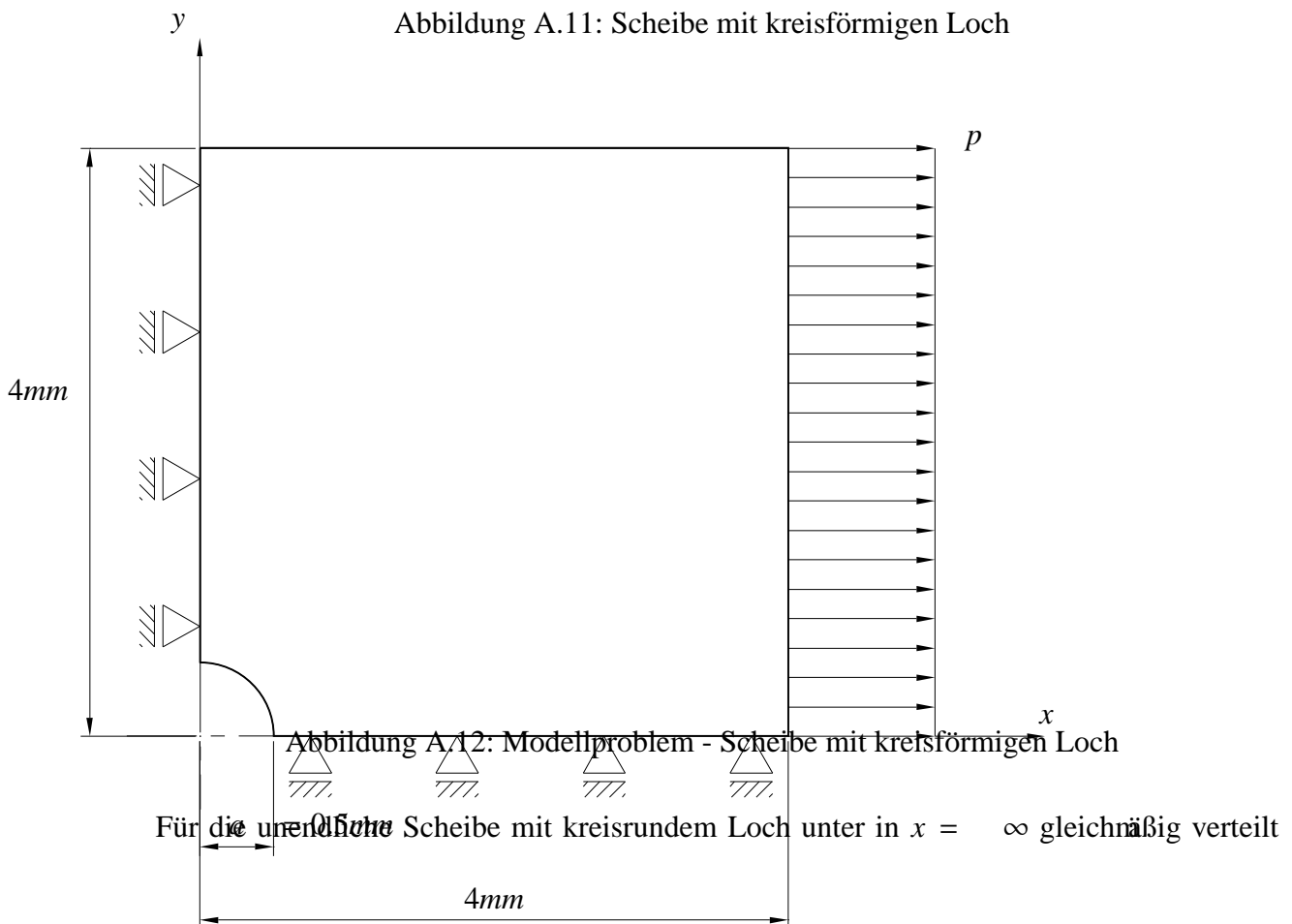


Abbildung A.11: Scheibe mit kreisförmigen Loch



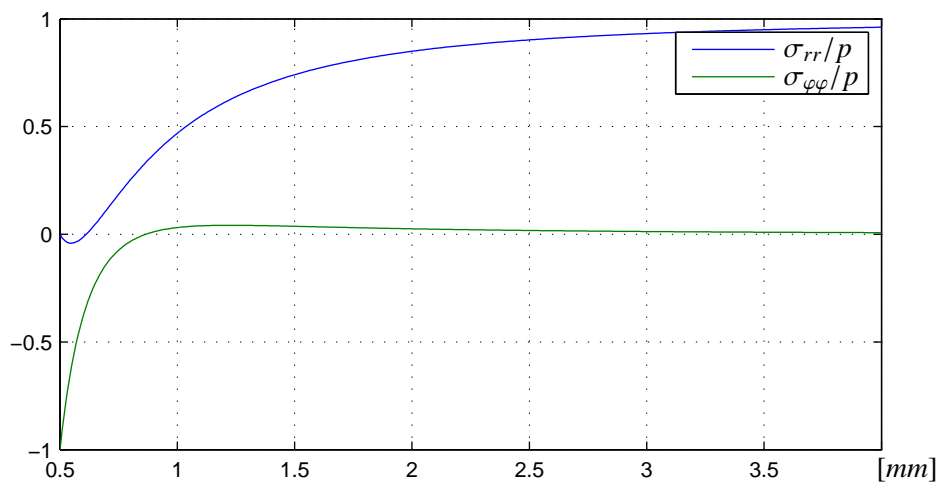
angreifenden Zugkräften existiert eine analytische Lösung. Es ist nicht möglich mit Hilfe von FE die unendliche Scheibe zu rechnen, daher bleibt eine Restabweichung von der analytischen Lösung auch für beliebig feines Netz bestehen. Diese Abweichung ist vor allem am Rand der Scheibe erkennbar.

$$r_r = \frac{p}{2h} \left[1 - \frac{a^2}{r^2} + 1 - \frac{4a^2}{r^2} + \frac{3a^4}{r^4} \right] \cos(2\varphi) \quad (\text{A.3})$$

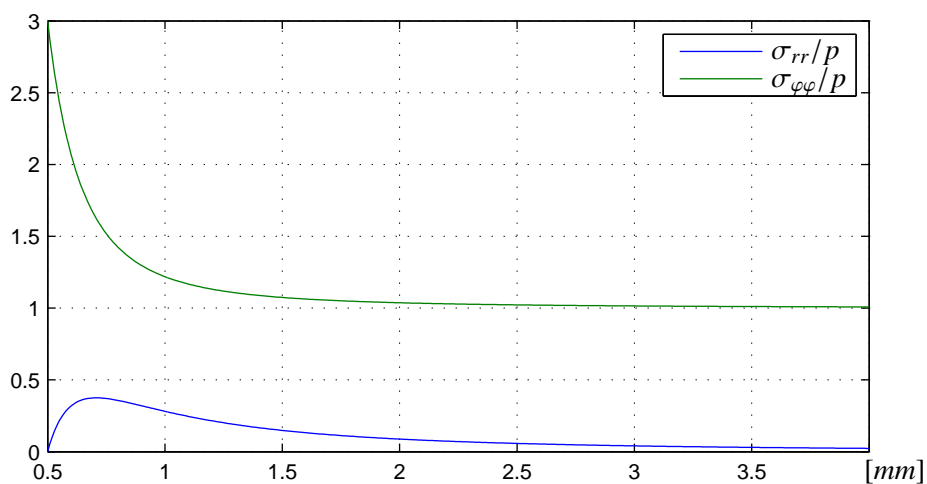
$$\varphi_{\varphi} = \frac{p}{2h} \left[1 + \frac{a^2}{r^2} - 1 + \frac{3a^4}{r^4} \right] \cos(2\varphi) \quad (\text{A.4})$$

Die Lösung des Spannungsproblems ist in folgender Abbildung A.13 angegeben.

A.3.1 Ergebnisse



(a) Spannungsverlauf entlang r auf $\varphi = 0$



(b) Spannungsverlauf entlang r auf $\varphi = \pi/2$

Abbildung A.13: Analytischer Spannungsverlauf

(a)	(b)
=	=
8.67829%	6.12626%
50	166
Fhg.	Fhg.
(0.Verf.)	(1.Verf.)
(c)	(d)
=	=
3.40442%	1.75641%
538	1526
Fhg.	Fhg.
(2.Verf.)	(3.Verf.)
(e)	(f)
=	Netz
0.97380%	nach
3874	dem
Fhg.	letz-
(4.Verf.)	ten
	Ad-
	ap-
	ti-
	ons-
	schrift

Abbildung A.14: Verfeinerungsfaktor für Dreieckselement 1. Ordnung - $\epsilon_{max} = 1\%$

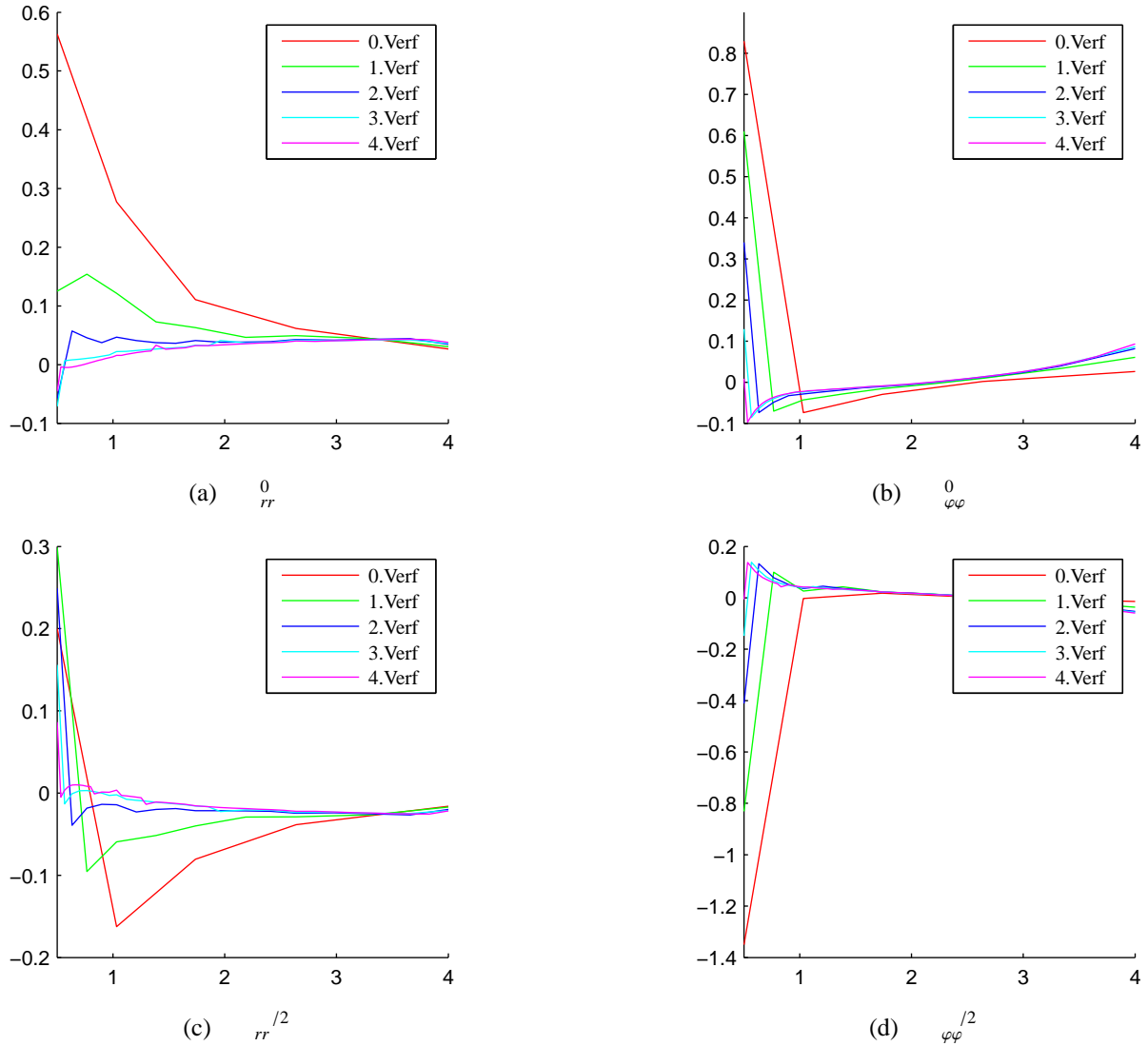


Abbildung A.15: Spannungsdi erenz $\sigma_{FE} - \sigma_{analytisch}$ zwischen FE L osung und analytischer L osung je Verfeinerungsschritt - Dreieckselement 1. Ordnung

(a)		(b)
=		=
2.39236%		0.78038%
166		602
Fhg.		Fhg.
(0.Verf.)		(1.Verf.)
(c)		(d)
=		=
0.23079%		0.08053%
2028		4080
Fhg.		Fhg.
(2.Verf.)		(3.Verf.)
	(e)	
	Netz	
	nach	
	dem	
	letz-	
	ten	
	Ad-	
	ap-	
	ti-	
	ons-	
	schrift	

Abbildung A.16: Verfeinerungsfaktor für Dreieckselement 2. Ordnung - $\epsilon_{max} = 0.1\%$

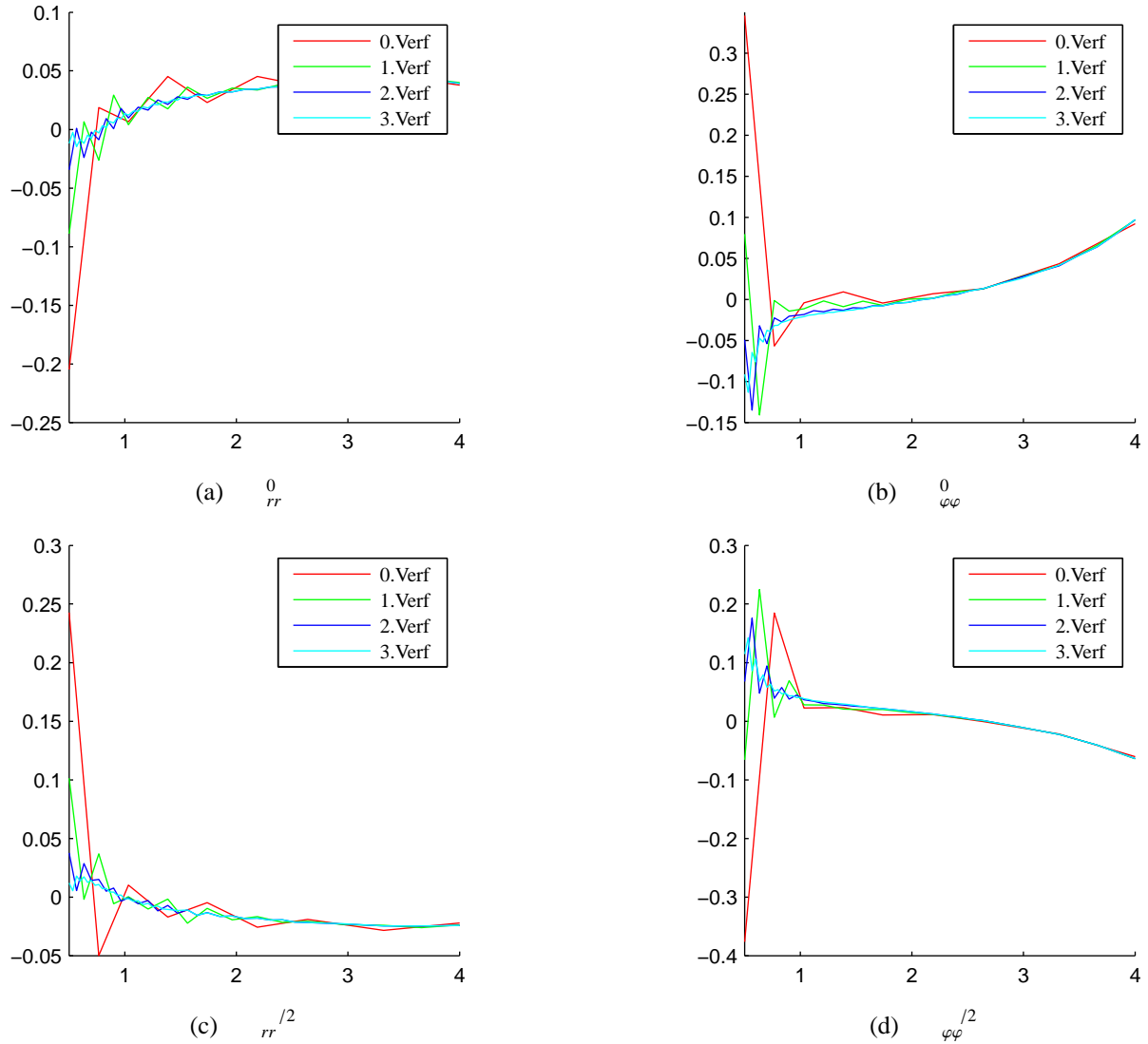


Abbildung A.17: Spannungsdi erenz $\sigma_{FE} - \sigma_{analytisch}$ zwischen FE L osung und analytischer L osung je Verfeinerungsschritt - Dreieckselement 2. Ordnung

A.3.2 Eingabedaten

Elemente 1. Ordnung kirsch_1.xml

```

<?xml version="1.0"?>
<problem>
  <global id="kirsch_1" dim="2" coord_sys_type="cartesian"/>
  <refine adaptivity="1" max_percentage_error="0.01" max_refinement_depth="1"/>
  <node>
    <n x="4.0000000000" y="4.0000000000">1</n>
    <n x="4.0000000000" y="1.8408330639">2</n>
    <n x="1.8408330639" y="4.0000000000">3</n>
    <n x="2.8573677233" y="1.9760108722">4</n>
    <n x="1.9815805950" y="2.8114287161">5</n>
    <n x="3.1437099654" y="0.9096389058">6</n>
    <n x="0.8939847277" y="3.1312300619">7</n>
    <n x="1.5659474617" y="1.5387717535">8</n>
    <n x="0.8544453223" y="2.1703063003">9</n>
    <n x="2.2225596106" y="0.7313505930">10</n>
    <n x="4.0000000000" y="0.0000000000">11</n>
    <n x="0.0000000000" y="4.0000000000">12</n>
    <n x="2.6386224933" y="0.0000000000">13</n>
    <n x="0.0000000000" y="2.6386224933">14</n>
    <n x="1.3258600548" y="0.6505566451">15</n>
    <n x="0.6851591493" y="1.1646346171">16</n>
    <n x="1.7371501597" y="0.0000000000">17</n>
    <n x="0.0000000000" y="1.7371501597">18</n>
    <n x="0.7125596103" y="0.4685829072">19</n>
    <n x="0.3502544290" y="0.7037110493">20</n>
    <n x="1.0317843375" y="0.0000000000">21</n>
    <n x="0.0000000000" y="1.0317843375">22</n>
    <n x="0.3535533845" y="0.3535533845">23</n>
    <n x="0.5000000000" y="0.0000000000">24</n>
    <n x="0.0000000000" y="0.5000000000">25</n>
  </node>
  <boundary>
    <b id="line" gnn="1_3_12" start="1" end="12">1</b>
    <b id="line" gnn="12_14_18_22_25" start="12" end="25">2</b>
    <b id="arc" gnn="25_23_24" start="25" end="24">3</b>
    <b id="line" gnn="24_21_17_13_11" start="24" end="11">4</b>
    <b id="line" gnn="11_2_1" start="11" end="1">5</b>
  </boundary>
  <element>
    <e gnn="25_23_20" m="1" et="1">1</e>
    <e gnn="23_24_19" m="1" et="1">2</e>
    <e gnn="20_23_19" m="1" et="1">3</e>
    <e gnn="25_20_22" m="1" et="1">4</e>
    <e gnn="22_20_16" m="1" et="1">5</e>
    <e gnn="19_24_21" m="1" et="1">6</e>
    <e gnn="19_21_15" m="1" et="1">7</e>
  </element>

```

```

<e gnn="22_16_18" m="1" et="1">8</e>
<e gnn="16_19_15" m="1" et="1">9</e>
<e gnn="15_21_17" m="1" et="1">10</e>
<e gnn="19_16_20" m="1" et="1">11</e>
<e gnn="15_17_10" m="1" et="1">12</e>
<e gnn="18_16_9" m="1" et="1">13</e>
<e gnn="9_16_8" m="1" et="1">14</e>
<e gnn="18_9_14" m="1" et="1">15</e>
<e gnn="14_9_7" m="1" et="1">16</e>
<e gnn="10_17_13" m="1" et="1">17</e>
<e gnn="10_13_6" m="1" et="1">18</e>
<e gnn="15_10_8" m="1" et="1">19</e>
<e gnn="8_10_4" m="1" et="1">20</e>
<e gnn="15_8_16" m="1" et="1">21</e>
<e gnn="9_8_5" m="1" et="1">22</e>
<e gnn="14_7_12" m="1" et="1">23</e>
<e gnn="12_7_3" m="1" et="1">24</e>
<e gnn="3_7_5" m="1" et="1">25</e>
<e gnn="5_7_9" m="1" et="1">26</e>
<e gnn="3_5_1" m="1" et="1">27</e>
<e gnn="10_6_4" m="1" et="1">28</e>
<e gnn="4_6_2" m="1" et="1">29</e>
<e gnn="2_6_11" m="1" et="1">30</e>
<e gnn="11_6_13" m="1" et="1">31</e>
<e gnn="4_2_1" m="1" et="1">32</e>
<e gnn="8_4_5" m="1" et="1">33</e>
<e gnn="5_4_1" m="1" et="1">34</e>
</element>
<element_type>
  <et id="triangle" real_const_set="2" gauss_points="1">1</et>
</element_type>
<constraint>
  <c id="boundary" b="2" x="0.0">1</c>
  <c id="boundary" b="4" y="0.0">2</c>
</constraint>
<load>
  <l id="pressure" b="5" value="1.0">1</l>
</load>
<material>
  <m id="linear_elastic" real_const_set="1">1</m>
</material>
<real_const_set>
  <rcs data="210000.0:0.3">1</rcs>
  <rcs data="1:0:1.0">2</rcs>
</real_const_set>
</problem>

```

Elemente 2. Ordnung kirsch_2.xml

```
<?xml version="1.0"?>
<problem>
  <global id="kirsch_2" dim="2" coord_sys_type="cartesian"/>
  <refine adaptivity="1" max_percentage_error="0.001" max_refinement_depth="0"/>
  <node>
    <n x="4.000000000" y="4.000000000">1</n>
    <n x="4.000000000" y="2.9204165320">2</n>
    <n x="2.9204165320" y="4.000000000">3</n>
    <n x="3.4286838616" y="2.9880054361">4</n>
    <n x="2.9907902975" y="3.4057143581">5</n>
    <n x="4.000000000" y="1.8408330639">6</n>
    <n x="1.8408330639" y="4.000000000">7</n>
    <n x="3.4286838616" y="1.9084219681">8</n>
    <n x="1.9112068295" y="3.4057143581">9</n>
    <n x="2.4194741591" y="2.3937197942">10</n>
    <n x="2.8573677233" y="1.9760108722">11</n>
    <n x="1.9815805950" y="2.8114287161">12</n>
    <n x="3.5718549827" y="1.3752359849">13</n>
    <n x="1.3674088958" y="3.5656150310">14</n>
    <n x="3.0005388443" y="1.4428248890">15</n>
    <n x="1.4377826614" y="2.9713293890">16</n>
    <n x="2.2116575925" y="1.7573913129">17</n>
    <n x="1.7737640283" y="2.1751002348">18</n>
    <n x="1.4180129587" y="2.4908675082">19</n>
    <n x="2.5399636669" y="1.3536807326">20</n>
    <n x="4.000000000" y="0.9204165320">21</n>
    <n x="0.9204165320" y="4.000000000">22</n>
    <n x="3.1437099654" y="0.9096389058">23</n>
    <n x="0.8939847277" y="3.1312300619">24</n>
    <n x="0.8742150250" y="2.6507681811">25</n>
    <n x="2.6831347880" y="0.8204947494">26</n>
    <n x="1.5659474617" y="1.5387717535">27</n>
    <n x="1.2101963920" y="1.8545390269">28</n>
    <n x="1.8942535361" y="1.1350611732">29</n>
    <n x="3.5718549827" y="0.4548194529">30</n>
    <n x="0.4469923639" y="3.5656150310">31</n>
    <n x="0.8544453223" y="2.1703063003">32</n>
    <n x="2.8911662294" y="0.4548194529">33</n>
    <n x="2.2225596106" y="0.7313505930">34</n>
    <n x="0.4469923639" y="2.8849262776">35</n>
    <n x="1.4459037583" y="1.0946641993">36</n>
    <n x="1.1255533055" y="1.3517031853">37</n>
    <n x="0.4272226612" y="2.4044643968">38</n>
    <n x="2.4305910520" y="0.3656752965">39</n>
    <n x="0.7698022358" y="1.6674704587">40</n>
    <n x="1.7742098327" y="0.6909536191">41</n>
    <n x="4.000000000" y="0.000000000">42</n>
    <n x="0.000000000" y="4.000000000">43</n>
    <n x="3.3193112467" y="0.000000000">44</n>
```

```

<n x="0.0000000000" y="3.3193112467">45</n>
<n x="0.4272226613" y="1.9537282300">46</n>
<n x="1.9798548852" y="0.3656752965">47</n>
<n x="2.6386224933" y="0.0000000000">48</n>
<n x="0.0000000000" y="2.6386224933">49</n>
<n x="1.3258600548" y="0.6505566451">50</n>
<n x="1.0055096021" y="0.9075956311">51</n>
<n x="0.6851591493" y="1.1646346171">52</n>
<n x="2.1878863265" y="0.0000000000">53</n>
<n x="0.0000000000" y="2.1878863265">54</n>
<n x="1.5315051073" y="0.3252783226">55</n>
<n x="0.3425795747" y="1.4508923884">56</n>
<n x="1.0192098326" y="0.5595697762">57</n>
<n x="0.6988593798" y="0.8166087621">58</n>
<n x="1.7371501597" y="0.0000000000">59</n>
<n x="0.0000000000" y="1.7371501597">60</n>
<n x="1.1788221962" y="0.3252783226">61</n>
<n x="0.5177067891" y="0.9341728332">62</n>
<n x="0.3425795748" y="1.0982094773">63</n>
<n x="1.3844672486" y="0.0000000000">64</n>
<n x="0.0000000000" y="1.3844672486">65</n>
<n x="0.7125596103" y="0.4685829072">66</n>
<n x="0.5314070196" y="0.5861469782">67</n>
<n x="0.8721719739" y="0.2342914536">68</n>
<n x="0.3502544290" y="0.7037110493">69</n>
<n x="0.1751272146" y="0.8677476934">70</n>
<n x="1.0317843375" y="0.0000000000">71</n>
<n x="0.0000000000" y="1.0317843375">72</n>
<n x="0.5330564974" y="0.4110681459">73</n>
<n x="0.3519039067" y="0.5286322169">74</n>
<n x="0.6062798052" y="0.2342914536">75</n>
<n x="0.1751272146" y="0.6018555246">76</n>
<n x="0.7658921688" y="0.0000000000">77</n>
<n x="0.0000000000" y="0.7658921688">78</n>
<n x="0.3535533845" y="0.3535533845">79</n>
<n x="0.1913417727" y="0.4619397819">80</n>
<n x="0.4619397819" y="0.1913417578">81</n>
<n x="0.5000000000" y="0.0000000000">82</n>
<n x="0.0000000000" y="0.5000000000">83</n>
</node>
<boundary>
  <b id="line" gnn="1_3_7_22_43" start="1" end="43">1</b>
  <b id="line" gnn="43_45_49_54_60_65_72_78_83" start="43" end="83">2</b>
  <b id="arc" gnn="83_80_79_81_82" start="83" end="82">3</b>
  <b id="line" gnn="82_77_71_64_59_53_48_44_42" start="82" end="42">4</b>
  <b id="line" gnn="42_21_6_2_1" start="42" end="1">5</b>
</boundary>
<element>
  <e gnn="83_79_69_80_74_76" m="1" et="1">1</e>

```

```

<e gnn="79_82_66_81_75_73" m="1" et="1">2</e>
<e gnn="69_79_66_74_73_67" m="1" et="1">3</e>
<e gnn="83_69_72_76_70_78" m="1" et="1">4</e>
<e gnn="72_69_52_70_62_63" m="1" et="1">5</e>
<e gnn="66_82_71_75_77_68" m="1" et="1">6</e>
<e gnn="66_71_50_68_61_57" m="1" et="1">7</e>
<e gnn="72_52_60_63_56_65" m="1" et="1">8</e>
<e gnn="52_66_50_58_57_51" m="1" et="1">9</e>
<e gnn="50_71_59_61_64_55" m="1" et="1">10</e>
<e gnn="66_52_69_58_62_67" m="1" et="1">11</e>
<e gnn="50_59_34_55_47_41" m="1" et="1">12</e>
<e gnn="60_52_32_56_40_46" m="1" et="1">13</e>
<e gnn="32_52_27_40_37_28" m="1" et="1">14</e>
<e gnn="60_32_49_46_38_54" m="1" et="1">15</e>
<e gnn="49_32_24_38_25_35" m="1" et="1">16</e>
<e gnn="34_59_48_47_53_39" m="1" et="1">17</e>
<e gnn="34_48_23_39_33_26" m="1" et="1">18</e>
<e gnn="50_34_27_41_29_36" m="1" et="1">19</e>
<e gnn="27_34_11_29_20_17" m="1" et="1">20</e>
<e gnn="50_27_52_36_37_51" m="1" et="1">21</e>
<e gnn="32_27_12_28_18_19" m="1" et="1">22</e>
<e gnn="49_24_43_35_31_45" m="1" et="1">23</e>
<e gnn="43_24_7_31_14_22" m="1" et="1">24</e>
<e gnn="7_24_12_14_16_9" m="1" et="1">25</e>
<e gnn="12_24_32_16_25_19" m="1" et="1">26</e>
<e gnn="7_12_1_9_5_3" m="1" et="1">27</e>
<e gnn="34_23_11_26_15_20" m="1" et="1">28</e>
<e gnn="11_23_6_15_13_8" m="1" et="1">29</e>
<e gnn="6_23_42_13_30_21" m="1" et="1">30</e>
<e gnn="42_23_48_30_33_44" m="1" et="1">31</e>
<e gnn="11_6_1_8_2_4" m="1" et="1">32</e>
<e gnn="27_11_12_17_10_18" m="1" et="1">33</e>
<e gnn="12_11_1_10_4_5" m="1" et="1">34</e>
</element>
<element_type>
  <et id="triangle" real_const_set="2" gauss_points="3">1</et>
</element_type>
<constraint>
  <c id="boundary" b="2" x="0.0">1</c>
  <c id="boundary" b="4" y="0.0">2</c>
</constraint>
<load>
  <l id="pressure" b="5" value="1.0">1</l>
</load>
<material>
  <m id="linear_elastic" real_const_set="1">1</m>
</material>
<real_const_set>
  <rcs data="210000.0:0.3">1</rcs>

```

```
<rcs data="2:0:1.0">2</rcs>  
</real_const_set>  
</problem>
```

B Anhang - Software

Hier sollen noch die Kernpunkte der Implementierung erläutert werden welche im Rahmen dieser Arbeit entstand.

B.1 SOOFEA

Als dahinteliegendes Framework für die Implementierung wurde das in einem früheren Projekt entstandene Finit Elemente Framework SOOFEA (Software For Object Oriented Finite Element Analysis) verwendet. Aufgabe von SOOFEA ist es eine Umgebung zur Verfügung zu stellen in der die Modelldaten und Analysedaten geeignet gespeichert werden können. Zu diesem Zweck wurden elementare Klassen der FE Analyse geschaffen (so wie Node, Element, Model usw.), Datenstrukturen zur Speicherung derer entwickelt und Entwurfsmuster eingearbeitet welche individuelle Elementfunktionen mit den Daten interagieren lassen. Damit ist es möglich sich primär auf die Algorithmen des einzelnen finiten Elements, bzw. des Materialgesetzes zu konzentrieren. Das Design basiert auf den in [2] vorgestellten Entwurf und stellt eine Weiterentwicklung im Bereich der Elementcodeankopplung, der Numerik und dem Input/Output-Handling dar. Ein wesentlicher Teil von SOOFEA ist nämlich die Interaktion mit dem Input- bzw. Outputdatensatz. Zu diesem Zweck wurden verschiedene Input- bzw. Outputfilter geschrieben, welche eine Kommunikation mit GiD als Pre- und Postprocessor erlauben oder Datenformate basierend auf XML ermöglichen.

B.2 Formfunktionen

In Kapitel 4 wurde bereits gezeigt, dass für die Implementierung des gewählten Dreieckselementes die Lagrange Polynome $l_i^m(\cdot)$ und deren Ableitung $\frac{\partial l_i^m(\cdot)}{\partial}$ benötigt werden. Das war die Motivation eine Klasse Lagrange zu entwickeln welche diese beiden Funktionalitäten als Methode bietet.

Listing 1: Klasse Lagrange

```
class Lagrange
{
public:
    Lagrange(unsigned max_order, double & constants);
    ~Lagrange();

    double operator()(unsigned order, unsigned index, double var);
};
```

```

    double derivative(unsigned order, unsigned index, double var);
protected:
    unsigned max_order_;
    double constants;
};

```

Die Operatormethode `double operator()(unsigned order, unsigned index, double var)` liefert dabei mit $m = \text{order}$, $i = \text{index}$ und $\text{var} = \text{var}$

$$l_i^m(\text{var}) = \prod_{\substack{j=0 \\ j \neq i}}^m \frac{j}{i-j} \quad (\text{B.1})$$

und die Methode `double derivative(unsigned order, unsigned index, double var)` liefert die Ableitung

$$\begin{aligned} \frac{\partial l_i^m(\text{var})}{\partial \text{var}} &= \prod_{\substack{j=0 \\ j \neq i}}^m \frac{1}{i-j} \frac{\partial}{\partial \text{var}} \left(\prod_{\substack{j=0 \\ j \neq i}}^m (\text{var} - j) \right) \\ &= \prod_{\substack{j=0 \\ j \neq i}}^m \frac{1}{i-j} \sum_{\substack{k=0 \\ k \neq j}}^m \left(\prod_{\substack{j=0, j \neq i \\ j \neq k}}^m (\text{var} - j) \right) \end{aligned} \quad (\text{B.2})$$

Zur Initialisierung der Objekte muss dem Konstruktor die Ordnung der Lagrange Polynome und ein Array mit den Werten der Stützstellen übergeben werden. Da für unser Dreieckselement die Ordnung und die Stützstellen der Lagrange Polynome a priori bekannt sind, haben wir als Schnittstelle zum Elementcode noch eine Interfaceklasse `TriangleShape` entwickelt, welche sofort die Werte für eine Formfunktion $N_{(I,J,K)}(r, s)$ (`double shape()`) und deren Ableitung nach r oder s (`double derivativeShape()`) liefert

Listing 2: Klasse `TriangleShape`

```

class TriangleShape : public Shape
{
public:
    TriangleShape(unsigned order);
    virtual ~TriangleShape();

    double shape(unsigned I, unsigned J,
                 double r, double s);

    double derivativeShape(unsigned derivative,
                           unsigned I, unsigned J,
                           double r, double s);
};

```

Mit Hilfe dieser Klassen lassen sich recht einfach die Einträge in den Interpolationsmatrizen \mathbf{N} , der Verzerrungs-Verschiebungs-Matrix \mathbf{B} und den Jacobi-Matrizen \mathbf{J} ermitteln.

B.3 Numerische Integration

Auf Grund der häufigen Anwendung numerischer Integration ist ein numerischer Integrator für C++ entstanden der in der Lage ist sowohl skalare Funktionen als auch Arrays aus Funktionen über ein entsprechendes Referenzgebiet zu integrieren. Zur Zeit sind Gewichte und Koordinaten der Integrationspunkte für die Gauß-Legendre Quadratur vorbereitet. Das gewählte System kann aber um beliebige numerische Integrationsverfahren erweitert werden, solange Sie folgender Definition genügen

$$I = \int F(\xi) d\xi = \sum_{i,j,k,\dots} w_{i,j,k,\dots} F(\xi_{i,j,k,\dots}) + R_n \quad (\text{B.3})$$

wobei Ω über das zu integrierende Referenzgebiet darstellt und ξ sei ein beliebiger Ortsvektor in \mathbb{R}^m .

Integration über einen Skalar Hier ist eine skalare Funktion $F(\xi)$ gegeben. Für den implementierten Spezialfall der Gauß-Legendre-Quadratur werden ein- bzw. mehrdimensionale Integrale durch den folgenden Zusammenhang berechnet. Dabei werden sowohl die Koordinaten des Ortsvektors der Integrationspunkte $\xi_{i,j,k,\dots} = [x_i, y_j, z_k, \dots]^T$ als auch die Gewichte w_i in einem Datensatz bereitgestellt.

$$I = \int F(\xi) d\xi = \sum_i w_i F(\xi_i) \quad (\text{B.4})$$

$$\sum_{i,j} w_{i,j} F(\xi_{i,j}) \quad (\text{B.5})$$

$$\sum_{i,j,k} w_{i,j,k} F(\xi_{i,j,k}) \quad (\text{B.6})$$

Integration über ein Array Hier ist eine Array beliebiger Dimension $\mathbf{F}(\xi)$ gegeben. Jedes Element des Arrays $\mathbf{F}_{(m,n,o,\dots)}$ wird analog zu dem skalaren Fall integriert.

$$I_{(m,n,o)} = \int \mathbf{F}_{(m,n,o)}(\xi) d\xi$$

$$\sum_i w_i \mathbf{F}_{(m,n,o)}(\xi_i) \quad (\text{B.7})$$

$$\sum_{i,j} w_{i,j} \mathbf{F}_{(m,n,o)}(\xi_{i,j}) \quad (\text{B.8})$$

$$\sum_{i,j,k} w_{i,j,k} \mathbf{F}_{(m,n,o)}(\xi_{i,j,k}) \quad (\text{B.9})$$

Sonderfall Dreiecksgebiet Die beiden Dimension sind nicht getrennt voneinander integrierbar. Daraus ergibt sich eine etwas veränderte Integrationsvorschrift in der die Gewichte nicht mehr nur für eine Dimensionsrichtung sondern in der Ebene gegeben sein müssen. (siehe [6])

$$I_D = \int_D \mathbf{F}(\xi) d\xi$$

$$\sum_{i,j} w_{i,j} \mathbf{F}(\xi_{i,j}) \quad (\text{B.10})$$

Eine ähnliche Integration würde sich für höherdimensionale Integrale über nicht parallele Gebiete (z.B. Tetraeder) ergeben.

B.3.1 Implementierung mittels Funktoren

Für die Implementierung des Integrators ist vor allem die einfache Übergabe des mathematischen Objektes $F(\xi)$ ein zentraler Punkt. In diesem Fall haben wir uns für die Kapselung eines Funktionspointers in einen Funktor entschieden. (siehe hierzu auch [13])

Listing 3: Ausschnitt aus der Signatur der Klasse FunctorScalar

```
template <class Item>
class FunctorScalar : public Functor
{
public:
    virtual ~FunctorScalar() {};

    FunctorScalar(Item const obj_ptr, double (Item:: fct_ptr 1D)(double r));
};
```

```

    FunctorScalar(Item const obj_ptr, double (Item:: fct_ptr_2D)(double r, double s));

    virtual double operator()(double r);
    virtual double operator()(double r, double s);
protected:
    Item obj_ptr;

    double (Item:: fct_ptr_1D)(double r);
    double (Item:: fct_ptr_2D)(double r, double s);
};

```

Dabei wird bei der Initialisierung im Konstruktor dem (Item::*fct_ptr_) die entsprechende Methode, welche den Funktionswert von $F(\xi)$ liefert, zugewiesen. Durch *function overloading* ist es auch möglich mit einem skalaren Funktor vom ein- bis zum mehrdimensionalen Fall alle Funktionen $F(\xi)$ zu übergeben. Diese Funktoren werden gemeinsam mit der Anzahl an Integrationspunkten in eine Dimensionsrichtung dem als Singleton (siehe [9]) implementierten Integrator übergeben.

Listing 4: Ausschnitt aus der Signatur der Klasse NumInt

```

class NumInt
{
public:
    static NumInt createNumInt(const std::string& num_int_file);
    static NumInt const getInstance();
    static void terminate();

    double intOneDimGaussLegendre(Functor& functor, unsigned points);
    double intTwoDimGaussLegendre(Functor& functor, unsigned points);
protected:
    static NumInt instance;
};

```

Ein beispielhafter Aufruf des Integrators für 3 Integrationspunkte könnte im eindimensionalen Fall für eine Funktion $F(x) = 2x^2 + 1$ also folgendermaßen aussehen

Listing 5: Beispielhafter Aufruf des Integrators für eine skalare Funktion

```

class A
{
public:
    double function1D(double r)
    {
        return( 2. * r + 1. );
    }
};

```

```
int main()
{
    NumInt::createNumInt("num_int.xml");

    FunctorScalar<A> functor1D = FunctorScalar<A>( new A(), &A::function1D );
    double I = NumInt::getInstance() > intOneDimGaussLegendre(functor1D,3);
}
```

Im Fall eines zu integrierenden Arrays ist dem Integrator neben der Anzahl der Integrationspunkte auch noch die Dimension des Arrays und die Anzahl der Einträge je Dimension zu übergeben. Dies würde für ein zweidimensionales Array mit der Größe 3x3 wie folgt aussehen

Listing 6: Beispielhafter Aufruf des Integrators für ein Array

```
class A
{
public:
    double array1D(double r, double s, unsigned long index);
};

int main()
{
    NumInt::createNumInt("num_int.xml");

    FunctorArray<A> functor1D_array = FunctorArray<A>( new A(), &A::array1D );
    unsigned long size[2] = {3,3};
    Array<double> array = NumInt::getInstance() >
        intOneDimArrayGaussLegendre(functor1D_array,3,dimension,size);
}
```

B.4 Lineare Algebra

Datenstrukturen Es erfolgte eine template orientierte Implementierung in Anlehnung an die Template Numerical Toolkit und die Vorgaben in [14]. Diese an Fortran orientierte Ordnung des Datenarrays in ein eindimensionales lineares Feld erlaubt eine direkte Kompatibilität mit vorhanden numerischen Bibliotheken für die Lösung der linearen Gleichungssysteme.

Lösung der linearen Gleichungssysteme Es wurden die Bibliotheken LAPACK (siehe [1]) bzw. BLAS (siehe [7] und [8]) für die Lösung der Gleichungssysteme bzw. Manipulation der Datenstrukturen verwendet.

B.5 Verfeinerungsalgorithmus für Dreiecksnetze

Die Implementierung hält sich hier primär an den Algorithmus beschrieben in [11] (abgeleitet aus dem Programm PLTMG). Im Wesentlichen geht es darum ein reguläres Dreiecksnetz gezielt zu verfeinern. Dazu werden wie in 3.3 beschrieben die Elemente mit zu großem Fehler markiert. Im Anschluss werden diese Elemente zerteilt und abschließend das Netz soweit bearbeitet, dass wieder eine gültige Triangulation des Gebietes entsteht.

B.5.1 Adaptive Verfeinerung

Um den Algorithmus besser beschreiben zu können möchte ich als erstes folgende Regeln aus [11] zitieren und zusammenfassen:

R1 Zerlegung eines Dreiecks durch Halbierung aller Seiten in 4 kongruente Dreiecke (*rote Teilung* - Abbildung B.1)

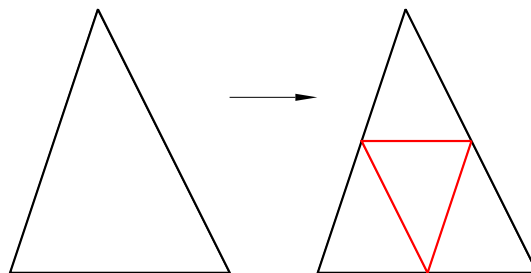


Abbildung B.1: Rote Teilung - **R1**

R2 Zerlegung eines Dreiecks in 2 Dreiecke durch Halbierung einer Seite (*grüne Teilung* - Abbildung B.2 - Strichlierte Linie als Teilung)

R3 Schließen der Triangulation - Abbildung B.3

- **R3a** auf Dreieck mit mindestens zwei unterteilten Kanten Anwendung von **R1**
- **R3b** auf Dreieck mit einer mindestens zweimal unterteilten Kante **R1**

Mit Hilfe der nun definierten Regeln zur Teilung von Einzeldreiecken ist es möglich einen Algorithmus zur regulären Verfeinerung von Dreiecksnetzen anzugeben (siehe Algorithm 1).

Algorithm 1 Adaptive Verfeinerung von Dreiecksnetzen

```
for Alle markierten Elemente in  ${}^l_h$  do  
    Regel R1 auf aktuelles Element anwenden  
end for  
while Regeln in Schleife anwendbar do  
    for Alle Elemente in  ${}^l_h$  do  
        if Aktuelles Element aus R2 entstanden then  
             $element =$  Mutterelement des aktuellen Elementes  
        else  
             $element =$  Aktuelles Element  
        end if  
        Regel R3a auf  $element$  anwenden  
        Regel R3b auf  $element$  anwenden  
    end for  
    for Alle Elemente in  ${}^l_h$  do  
        Regel R2 auf aktuelles Element anwenden  
    end for  
    for Alle Elemente in  ${}^{l+1}_h$  do  
        Regel R2 auf aktuelles Element anwenden  
    end for  
end while
```

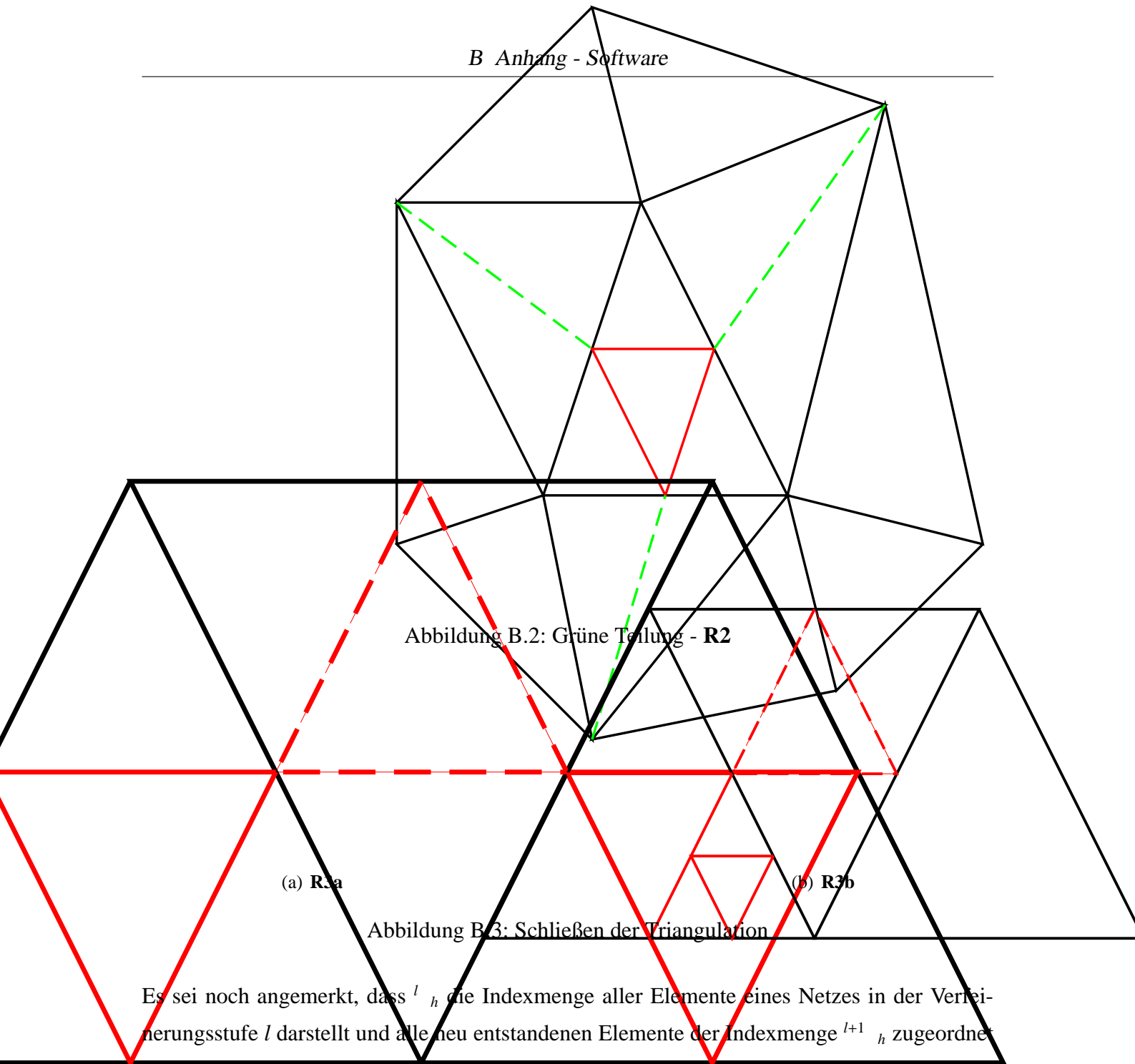


Abbildung B.2: Grüne Teilung - R2

(a) R3a

(b) R3b

Abbildung B.3: Schließen der Triangulation

Es sei noch angemerkt, dass I_h^l die Indexmenge aller Elemente eines Netzes in der Verfeinerungsstufe l darstellt und alle neu entstandenen Elemente der Indexmenge I_h^{l+1} zugeordnet werden müssen.

Zur effektiven Implementierung des Algorithmus ist ein strukturiertes hierarchisches Netz sinnvoll, d.h. 2D Elemente bestehen aus Kanten und diese wiederum aus Knoten. Über diese Objektstruktur wird noch die Information der Mutter-Tochterbeziehung von Elementen und Kanten gelegt

B.5.2 Globale Verfeinerung

Eine globale Verfeinerung kann sehr leicht erreicht werden. Dazu muss lediglich die Regel **R1** auf alle Dreiecke im Netz angewendet werden.

B.5.3 Generelle Hinweise

Während der Teilung von Dreiecken mit **R1** kommt es zur Bildung neuer Knoten. Falls diese neuen Knoten auf Kanten liegen welche Teilmengen des Randes darstellen so ist darauf zu achten, dass die neuen Knoten auch auf Ω zu liegen kommen, ganz besonders gilt dies für krummlinig berandete Gebiete Ω . Für alle restlichen neu entstehenden Knoten innerhalb des Gebietes Ω genügt es die Knoten auf den Streckenhalbierungspunkt der Verbindungsstrecke zwischen den benachbarten Knoten zu legen.

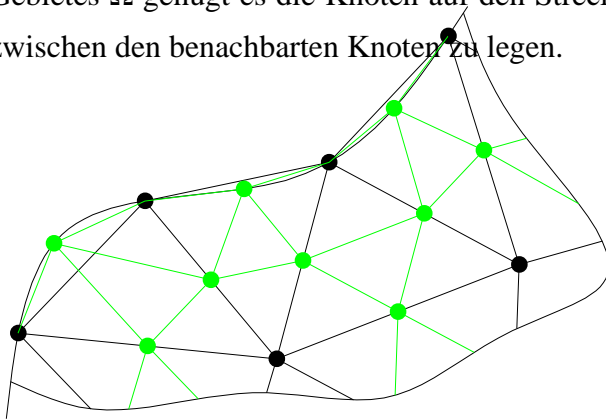


Abbildung B.4: Bildung neuer Knoten