Gemischte Least-Squares-FEM für Elastoplastizität

Von der
Fakultät für Mathematik und Physik
der

Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades eines

DOKTORS DER NATURWISSENSCHAFTEN
Dr. rer. nat.

genehmigte Dissertation
von

Dipl.-Math. Jörg Kubitz
geboren am 20.07.1978 in Hannover

2007

Referent: Prof. Dr. Gerhard Starke
Korreferent: Prof. Dr. Bernd Simeon
Tag der Promotion: 18.01.2007

Danksagung

Mein Dank gilt zunächst Herrn Prof. Dr. Gerhard Starke, der mich auf dieses Thema gebracht hat und meine Arbeit betreut hat.

Der Deutschen Forschungsgemeinschaft (DFG) gilt mein Dank dafür, dass sie diese Arbeit durch ein Stipendium finanziert hat.

Auch den Mitgliedern des Graduiertenkolleg 615 möchte ich an dieser Stelle für den interdisziplinären Austausch meinen Dank ausdrücken. Insbesondere wäre ohne die Vorarbeiten von Dr. Johannes Korsawe (der mich in die Implementierung der Least-Squares-FEM einwies) und Dr. André Geilenkothen (der mich in die Plastizität einführte) diese Arbeit nicht möglich gewesen. Außerdem half Dr. Frank S. Attia mir immer wieder beim Verständnis der Funktional-Analysis und bereicherte meinen Arbeitsalltag.

Meiner Familie danke ich für alle Unterstützung und insbesondere meinem Bruder Matthias Kubitz für die fleißige Unterstützung in der Rechtschreibung.

Und letzendlich gebührt für alle Ideen Dem Dank, der die Menschen mit Verstand gesegnet hat.

Jörg Kubitz

Zusammenfassung

In dieser Arbeit wird eine gemischte Least-Squares-Formulierung für elastoplastische Materialverformungen untersucht. Die Least-Squares-Formulierung

$$\min_{\sigma_h, u_h} \|\text{div } \sigma_h\|_{0,\Omega}^2 + \|\sigma_h - \mathcal{C}^{\text{ep}} \epsilon(u_h)\|_{0,\Omega}^2$$

zeichnet sich durch ihre einfache Struktur aus. Rechnerisch günstig ist, dass das Least-Squares-Funktional bereits ohne zusätzlichen Aufwand einen lokalen a-posteriori Fehlerschätzer liefert.

Bei der Implementierung mit der Methode der finiten Elemente (FEM) ist ein nichtglattes Optimierungsproblem zu lösen. Ein Nachteil der LS-Formulierung der sich spätestens durch die übliche Integration mit Quadraturformeln ergibt ist, dass die Optimierungsaufgabe $\min_x f(x) := \frac{1}{2} \|R(x)\|^2$ des nicht glatten LS-Funktionals manchmal nicht die Lösung des Gleichungssystems $\nabla f(x) = 0$ ist. Durch eine an das Problem angepasste Glättung wird dieser Nachteil umgangen.

Außerdem werden verbesserte Algorithmen zur adaptiven Gittergenerierung, zum Transfer der Geschichtsdaten zwischen zwei Triangulierungen und zur Berechnung der elasto-plastischen Systemmatrizen vorgeschlagen, die sich unabhängig von der Least-Formulierung einsetzen lassen.

Die Eigenschaften der Formulierung und neuen Algorithmen werden mit numerischen Beispielen für perfekte Prandtl-Reuss-Plastizität demonstriert. Die Beispiele wurden h-adaptiv mit quadratischen konformen Elementen für die Verschiebung und quadratischen Raviart-Thomas-Elementen für die Spannung berechnet. Alle Berechnungen wurden mit Autodifferentiation durchgeführt, die sich als hilfreich erwies.

Während sich ein Teil der Arbeit damit befasst, bisherige Forschungsergebnisse darzustellen und in neuem Kontext anzuwenden, sind nach dem Kenntnisstand des Autors folgende Ideen neu:

- die analytische Lösung der Konsistenzgleichung bei exponentieller Härtung mit Hilfe der Lambert-W-Funktion,
- dazu eine schnelle Approximation der Lambert-W-Funktion,
- die ans Problem angepasste ϵ-Glättung für Funktionen mit Knicken,
- die Einführung interpolatorischer Quadraturformeln mit Nullgewichten zum Transfer der Geschichtsdaten zwischen zwei Triangulierungen,
- ein adaptiver Verfeinerungsalgorithmus um optimale Triangulierungen mit wenigen mehrfach verfeinernden Schritten anzunähern,
- ein schneller (vektorisierbarer) Algorithmus zur konformen Entfeinerung von Triangulierungen, der ohne eine hierarchische Datenstruktur auskommt,
- die Ausnutzung additiv linear separabler Freiheitsgrade in nichtlinearen Lösern bei gemischten Formulierungen.

Stichworte: *Elastoplastizität, gemischte Finite-Elemente-Methode, a-posteriori Fehlerschätzer, Methode kleinster Quadrate, Lambert-W-Funktion, adaptive Verfeinerung*

ABSTRACT

This thesis considers a mixed least-squares formulation for elastoplastic material deformation. The implementation with the finite element method leads to a nonsmooth nonlinear optimisation problem. The nonsmoothnes sometimes impacts on the Gauss-Netwon method while in classic Ritz-Galerkin formulations the nonsmoothness does never impact on the convergence of the Newton method. This drawback is handled by a problem-adapted smoothing.

Furthermore improved algorithms for adaptiv mesh generation, for the transfer of the material history among two different meshes and for the assembling of the elastoplastic system matrices are proposed. These can be used independend of the least-squares formulation.

The properties of the formulation and the new algorithms are demonstrated with numeric examples for Prandtl-Reuss elastoplasticity. The examples are computed h-adaptive with quadratic conforming elements for the displacement and quadratic Raviart-Thomas elements for the stress. All computations are accomplished with automatic differentiation, which shows to be helpful.

The least-squares formulation

$$\min_{\sigma_h, u_h} \|\operatorname{div} \sigma_h\|_{0,\Omega}^2 + \|\sigma_h - \mathcal{C}^{\mathrm{ep}} \epsilon(u_h)\|_{0,\Omega}^2$$

features an easy structure. A computational benefit is that the local evaluation of the least-squares functional serves as a local a-posteriori error estimator. A drawback that at the latest arises from using the usual numerical integration is that the optimisation problem $\min_x f(x) := \frac{1}{2} \|R(x)\|^2$ of the nonsmooth least-squares functional sometimes is not the solution of the equality $\nabla f(x) = 0$. But it reveals that this does only have a small impact on the computation of finite element approximations.

While some parts of this thesis deals only with usual methods that are applied in a new context, the following ideas are new to the best of the author's knowledge:

- the analytic solution of the local equality for exponentional hardening with the Lambert W-function,
- a fast approximation of the Lambert-W function,
- the problem-adapted ϵ-smoothing for functions with kinks,
- the introduction of interpolatoric cubature formulas with zero weights for the transfer of the material history between two meshes,
- an adaptive refinement method for the approximation of optimal triangulations with a few multiple refinement steps,
- a fast algorithm for conforming derefinement without any hierarchic data structures,
- the utilisation of additive linear seperable degrees of freedom in nonlinear solvers for mixed formulations.

Keywords: *Elastoplasticity, mixed finite element method, a-posteriori error estimator, least-squares method, Lambert W-function, adaptive refinement*

Inhaltsverzeichnis

Einleitung

Was Plastizität bedeutet. Anschaulich betrachtet handelt diese Arbeit von der Berechnung der Verformungen von Gegenständen. Wirken physikalische Kräfte auf einen Körper, verändert dieser seine Form nach Gesetzmäßigkeiten, die es in einem mathematischen Modell zu beschreiben und dann zu berechnen gilt. Kehrt der Körper nach dem Wegfall der äußeren Kräfte in seine Ursprungsform zurück, so spricht man von *elastischer* Deformation. Eine *plastische* Veränderung des Materials liegt dann vor, wenn der Körper seine Verformung hingegen behält. In der Praxis hat man es mit einer Mischung aus diesen beiden Effekten zu tun, ein gebogener Gegenstand schnappt ein bisschen zurück, aber nicht genau in seine Ursprungslage. Dieses Materialverhalten nennt man *elasto-plastisch*.

Anwendungsmöglichkeiten. Die hier dargestellten Methoden eignen sich dafür, numerische Approximationen zu berechnen, wie Gegenstände unter Belastung ihre Form verändern. Als bekanntestes Beispiel dafür sind Computer-simulierte Crash-Tests der Automobilindustrie zu nennen. Auch Architekten müssen ihre Bauten rechnerisch auf Stabilität prüfen lassen, bevor diese gebaut werden. Neben der einfachen Deformationsvorhersage und Bauteilüberprüfung lassen sich dadurch sogar Bauteile in ihrer Form und Materialeigenschaft optimieren.

Für den Anwender solcher Berechnungen ist es wichtig, dass die Ergebnisse nicht nur zuverlässig und genau sind sondern auch schnell mit kostengünstiger Computerhardware berechnet werden können. Die Kosten der Rechenzeit ergeben sich im Wesentlichen aus Rechenleistung (Geschwindigkeit) und Speicherplatz des Rechners. Daher gilt es die angewendeten Materialmodelle, Konstruktionsmodelle und Rechenalgorithmen so zu verbessern, dass die Computer für eine bestimmte Aufgabe unter Verwendung von wenig Speicher möglichst schnell zu einem hinreichend genauen Ergebnis kommen.

Da zur Zeit viele langsame Prozessoren, die zusammen ebenso schnell rechnen können wie ein großer Prozessor, billiger in Herstellung und Unterhalt sind, sollten die verwendeten Algorithmen möglichst gut parallelisierbar (d.h. auf mehrere Rechner aufteilbar) seien. Manche Aufgaben sind sogar so komplex, dass sie sich mit heutigen Rechnern überhaupt nicht lösen lassen, wenn das Problem nicht auf mehrere Computer verteilt werden kann.

In dieser Arbeit vernachlässigte Materialeigenschaften. In der Natur kosten alle Vorgänge Zeit und nach dem Anlegen äußerer Kräfte treten Verformungen wegen der Massenträgheit nur langsam ein. Am Beispiel einer zähen Flüssigkeit

(Honig) sieht man auch noch einen weiteren Effekt: Durch innere Reibung kommt es zu einem viskosen Fließen. Es gibt sogar Materialien, die zunächst ihre plastische Verformung beibehalten, aber nach Erhitzen und Abkühlen in ihre ursprüngliche Form zurückkehren - sogenannte Formgedächtnislegierungen. Auch bei thermischen Veränderungen verformen sich viele Körper (dehnen sich unter Hitze- oder Kälteeinwirkungen). All diese Materialeigenschaften werden in dieser Arbeit nicht näher behandelt, lassen sich jedoch in ähnlicher Weise berechnen.

Die Geschichte der Least-Squares-Formulierung. Zum Lösen von Randwertproblemen wurde in der Mechanik lange Zeit erfolgreich die Ritz-Galerkin-Formulierung zusammen mit der Finiten-Element-Methode (FEM) verwendet. Seit 1965 wurde versucht solche Variations-Formulierungen auch auf andere Anwendungen zu übertragen. Dabei zeigten sich z.B. bei Strömungs- und Transportproblemen numerische Schwierigkeiten. Diese konnten 1974 mit Least-Squares-Formulierungen umgangen werden.

Die Methode kleinster Quadrate (*Least-Squares*) wurde 1805 erstmals von Legendre publiziert, jedoch bereits vorher von Gauss zur statistischen Analyse eingesetzt ([**32**, S. 209ff]). Dieser verwendete dabei Gewichtungen proportional zur Genauigkeit einzelner Messwerte. In der Least-Squares-Finite-Element-Methode (LSFEM) werden die Gewichte hingegen am besten proportional zu Quadraturgewichten gewählt.

In den letzten Jahren erschienen auch vermehrt Veröffentlichungen zur Anwendung der LSFEM in der Mechanik bei linearer Elastizität. Literaturübersichten über die LSFEM findet sich in [**8**] und [**41**]. Eine Übersicht über verschiedene Least-Squares-Formulierungen ist außerdem in [**7**] zu finden.

Überblick über diese Arbeit. Wir geben zunächst in Kapitel 1 die verwendeten Materialmodelle an, fassen das zu lösende Differential-Algebraische-Gleichungssystem zusammen und beschreiben dann kurz die verwendete Zeitdiskretisierung. Es bleibt dann in jedem Zeitschritt eine partielle Differentialgleichung zu lösen. Dafür wird eine Least-Squares-Formulierung in Kapitel 2 aufgestellt. Wir zeigen dann, dass das Least-Squares-Funktional für bestimmte Materialmodelle ein effizienter Fehlerschätzer ist. Anschließend wenden wir eine Diskretisierung mit finiten Elementen auf die Least-Squares-Formulierung an, um h-adaptive Beispiele in Kapitel 3 zu berechnen.

Um die Berechnungen durchzuführen verwenden wir zunächst das Gauss-Newton-Verfahren und zeigen in Kapitel 4 an Beispielen, dass dies nicht immer zur Minimalstelle konvergieren muss, aber dass bereits nach zwei Iterationen mit einem Multilevelalgorithmus eine ausreichend gute Näherung vorliegt. Bei der Ritz-Galerkin Formulierungen konvergiert hingegen das (nichtglatte) Newton-Verfahren lokal immer ([**6**]). Dass Gauss-Newton-Verfahren benötigt die partielle Ableitung des Residuums. Es wird gezeigt, dass sich Autodifferentiation dazu eignet, diese zu berechnen.

An den Beispielen, die in der Programmiersprache "Matlab" implementiert wurden, wird in Kapitel 5 deutlich, dass der Zeitaufwand sich im Wesentlichen auf zwei Teilprobleme verteilt:

(1) Das Lösen der linearen Gleichungssysteme
(2) Das Aufstellen der Systemmatrix.

Dazu werden Verbesserungsvorschläge erarbeitet. Die Anzahl der Quadraturpunkte in der Least-Squares-FEM (LSFEM) kann wesentlich reduziert werden, wodurch der Gesamtaufwand halbiert wird. Außerdem kann bei exponentieller Verfestigung das nichtlineare lokale Gleichungssystem analytisch mit Hilfe der Lambert-W-Funktion gelöst werden. Wir zeigen dann, wie die Lambert-W-Funktion schneller als bisher berechnet werden kann.

Weiter fällt auf, dass bei gemischten Formulierungen die Spannungen nur linear in das nichtlineare Residuum einfließen. Wir untersuchen daher, wie dies algorithmisch genutzt werden kann.

Dann wird in Kapitel 6 beschrieben, wie ein Transfer der Geschichtsdaten zwischen zwei Zeitschritten (von einer Gebietszerlegung auf ein andere) durchgeführt werden kann. Anders als in bisherigen Arbeiten wird der Transfer der Geschichtsdaten erreicht, indem ein eindeutiges Interpolationspolynom durch die Werte an den Quadraturpunkte gelegt wird. Dazu müssen, je nach Quadraturformeln, zusätzliche (nullgewichtete) Quadraturpunkte eingeführt werden. Zusammen mit zueinander konformen Triangulierungen kann der entstehende Transferfehler dabei wesentlich kleiner sein als bei bisherigen Methoden.

Anschließend wird in Kapitel 7 gezeigt, wie zueinander konforme Triangulierungen durch Ver- und Entfeinerung konstruiert werden können. Durch die Wahl einer speziellen Entfeinerung wir dabei erreicht, dass konforme Mehrgitterverfahren möglich sind.

Zum Schluss wird gezeigt, wie die Triangulierungen effizienter als bisher adaptiviert werden können. Dies wird mit wenigen parallelisierbaren mehrfach verfeinernden Schritten erreicht.

KAPITEL 1

Die grundlegenden Materialgleichungen

Die in dieser Arbeit behandelten Materialeigenschaften lassen sich durch Spannungen σ und Verzerrungen ϵ im Material beschreiben. Diese beide Größen sind Matrizen (auch Tensoren 2. Stufe genannt):

$$\sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{pmatrix}, \epsilon = \begin{pmatrix} \epsilon_{11} & \epsilon_{12} & \epsilon_{13} \\ \epsilon_{21} & \epsilon_{22} & \epsilon_{23} \\ \epsilon_{31} & \epsilon_{32} & \epsilon_{33} \end{pmatrix}.$$

Die Verzerrung im Material resultieren aus Verschiebungen u der Materialteilchen, die sich sich als Vektor (also Tensor 1. Stufe) schreiben lassen:

$$u = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}.$$

All diese Größen sind Funktionen des Ortes x und der Zeit t: $u = u(\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3, t)$, $\epsilon = \epsilon(\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3, t)$, $\sigma = \sigma(\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3, t)$. Aus Gründen der Übersichtlichkeit wird auf diese Funktionsparameter verzichtet und alle folgenden Formeln sind punktweise zu verstehen. Bei kleinen Deformationen sind die Verzerrungen eines Materials gerade die symmetrischen Ableitungen der Verschiebung:

$$\epsilon = \frac{1}{2}(\nabla u + (\nabla u)^T).$$

1.1. Der Elastische Materialtensor und die verwendete Notation

Linear elastisches Material wird beim hookeschen Gesetz durch einen linearen Zusammenhang von Spannung und elastischer Verzerrung ϵ^e beschrieben:

$$\sigma = \mathcal{C}\epsilon^e. \tag{1}$$

Für den Materialtensor $\mathcal{C}$ gelten folgende Eigenschaften (siehe [**57**, S. 74]): Wenn das Material sich in jedem Ort gleich verhält, also $\mathcal{C}$ nicht von x abhängt, nennt man das Material homogen. Eine Unabhängigkeit von der Ausrichtung bzw. Drehung des Materials bezeichnet man als isotrop. Ist schließlich $\mathcal{C}$ eine Konstante (d.h. hängt nicht von $\epsilon(u)$ ab) und es ist $\sigma = \mathcal{C} \cdot \epsilon(u)$, so spricht man von linearer Elastizität. Der homogene, isotrope, lineare Materialtensor hängt dann nur von zwei so genannten Lamé-Parametern λ und μ ab:

$$\mathcal{C}\epsilon(u) \quad = \quad \lambda I \operatorname{tr} \epsilon(u) + 2\mu\epsilon(u).$$

Zur Implementierung eines isotropen Materialgesetzes ist es praktisch σ und ϵ als Vektoren und $\mathcal{C}$ als Matrix aufzufassen. (Matlab unterstützt zwar höher dimensionale Felder, aber keine Multiplikationen für Tensoren). Eingebürgert hat sich die Schreibweise $\sigma = (\ \sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{13}, \sigma_{23}\ )^T$, die Gebrauch von der Tatsache macht, dass der Spannungstensor σ^* symmetrisch ist. In der hier benutzen gemischten Formulierung ist σ eine primäre Variable die nicht zwangsläufig symmetrisch ist. Daher wird in dieser Arbeit eine Notation der Form

$$\sigma = (\ \sigma_{11}, \sigma_{12}, \sigma_{13}, \sigma_{21}, \sigma_{22}, \sigma_{23}, \sigma_{31}, \sigma_{32}, \sigma_{33}\ )^T,$$
$$\epsilon = (\ \epsilon_{11}, \epsilon_{12}, \epsilon_{13}, \epsilon_{21}, \epsilon_{22}, \epsilon_{23}, \epsilon_{31}, \epsilon_{32}, \epsilon_{33}\ )^T$$

verwendet. Dadurch ergibt sich für $\mathcal{C}$ die Gestalt einer Matrix:

$$\mathcal{C} = \begin{pmatrix} 2\mu+\lambda & 0 & 0 & 0 & \lambda & 0 & 0 & 0 & \lambda \\ 0 & 2\mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2\mu & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2\mu & 0 & 0 & 0 & 0 & 0 \\ \lambda & 0 & 0 & 0 & 2\mu+\lambda & 0 & 0 & 0 & \lambda \\ 0 & 0 & 0 & 0 & 0 & 2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\mu & 0 \\ \lambda & 0 & 0 & 0 & \lambda & 0 & 0 & 0 & 2\mu+\lambda \end{pmatrix}$$

Außerdem ist im folgenden bei Normen von Tensoren die Frobeniusnorm

$$\|\sigma\| := \sqrt{\sum_{i,j} \sigma_{ij}^2}$$

gemeint.

Im folgenden wird ein homogenes, isotropes Material angenommen, das sich linear verhält, solange keine plastische Verformung auftritt.

1.2. Materialmodelle für Elasto-Plastizität

Während man bei elastischen Materialmodellen davon ausgeht, dass die Spannungen beliebig große Werte annehmen können, geht man bei plastischem Materialverhalten davon aus, dass der Bereich der elastischen Verzerrungen beschränkt und somit auch der Bereich der Spannungen beschränkt ist. Ist das Maß der elastischen Verzerrung größer als ein materialabhängiger Wert kommt es zum plastischen Fließen des Materials. In der Vergangenheit wurden viele Materialmodelle aufgestellt, die sich durch das Maß und das daraus resultierende Fließverhalten unterscheiden. Den Rand der Menge aller zu einem Zeitpunkt lokal zulässigen Spannungen nennt man Fließfläche.

Als Maß kommen nur Eigenschaften der Spannung in Frage, die invariant gegenüber Koordinatentransformationen, also unabhängig vom Betrachtungswinkel sind. Dies gilt für die Eigenwerte λ_i des Spannungstensors. Die Eigenwerte von σ

werden in diesem Anwendungsfall Hauptspannungen genannt (siehe z.B. [**27**, S. 42]).

Als erste skalare Invariante der Verzerrung bezeichnet man die Spur $\operatorname{tr}\epsilon = \sum_{i=1}^{3}\epsilon_{ii} = \sum_{i=1}^{3}\lambda_i$. Sie ist ein Maß für die Volumenveränderung. Die erste skalare Invariante der Spannung nennt man $I_1 := \operatorname{tr}\sigma$.

Die Volumenveränderung hat bei Metallgitterstrukturen physikalisch keinen Einfluss auf die sonstigen Verzerrungen und umgekehrt. Man unterteilt kleine Verzerrungen daher additiv in den volumetrischen Anteil $\operatorname{vol}\epsilon = \frac{1}{3}\operatorname{tr}\epsilon$ und den deviatorischen Anteil $\operatorname{dev}\epsilon = \epsilon - \operatorname{vol}\epsilon$. Dabei gilt $\operatorname{dev}\epsilon : \operatorname{vol}\epsilon = 0$, wobei der Doppelpunkt das Skalarprodukt von zwei Tensoren bezeichnet:

$$\epsilon : \sigma := \sum_i \sum_j \epsilon_{ij}\sigma_{ij}.$$

Bei großen Verzerrungen braucht man hingegen eine multiplikative Unterteilung [**57**, S. 90].

Die zweite Invariante lautet $I_2 = \lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_3\lambda_1 = \frac{1}{2}\|\sigma\|^2$ (ziehe z.B. [**64**, S. 85]). Man bezeichnet mit J_2 die zweite Invariante der deviatorischen Spannung. Das von von Mises aufgestellte Kriterium lautet $\sqrt{J_2} = \|\operatorname{dev}\sigma\| \leq R := \sqrt{\frac{2}{3}}\sigma_Y$, wobei R den Radius der Fließfläche angibt. Man spricht dort daher auch von J_2-Plastizität.

Die dritte Invariante $I_3 = \lambda_1\lambda_2\lambda_3 = \det(\sigma)$ findet seltener Anwendung in der Plastizitätstheorie (aber z.B. in [**43**]). Auch andere Verknüpfungen der Eigenwerte wurden sehr früh verwendet: Trescas Kriterium war beispielsweise $\max_{i\neq j}|\lambda_i - \lambda_j|$. Sein Nachteil ist, dass es nicht überall differenzierbar ist.

Die Verzerrung ϵ kann man klassischer Weise additiv aufteilen in eine elastische Verzerrung ϵ^e und eine plastische Verzerrung ϵ^p.

$$\epsilon = \epsilon^p + \epsilon^e. \tag{2}$$

Da die Spannungen im Material nach (1) auf rein elastische Verzerrungen zurückgeführt werden, folgt für den plastischen Anteil an der Verzerrung aus (2) :

$$\epsilon^p = \epsilon - \mathcal{C}^{-1}\sigma. \tag{3}$$

Man spricht außerdem von isotroper Verfestigung des Materials, wenn sich im Laufe der Materialevolution die Fließfläche ausdehnt. Umgekehrt gibt es auch Entfestigung, bei der die Fließfläche zunehmend kleiner wird. Dies ist ein lokaler Effekt, der mit Hilfe von einer inneren Materialzustandsvariablen α formuliert wird. Eine weitere beobachtete Materialzustandsänderung ist eine kinematische Verschiebung der Fließfläche um den Vektor β. Man fasst alle inneren Materialzustände zu einem Vektor $q = (\alpha,\ \beta)^t$ zusammen.

Allgemeiner als der lineare Zusammenhang des hookeschen Gesetzes (1) lautet der konstitutive Zusammenhang von Spannung σ und Verschiebung u:

$$\sigma \quad = \quad \sigma(\epsilon(u), q) \text{ (konstitutive Beziehung).} \tag{4}$$

Dies wird in der Literatur manchmal auch als $\sigma = \mathcal{C}^{\text{ep}}\epsilon(u)$ geschrieben. Ein allgemeine Beschreibung der zulässigen Spannungen lautet:

$$\psi(\sigma, q) \leq 0 \text{ (Entladung).} \tag{5}$$

Reine J_2-Plastizität kann man dann schreiben als:

$$\psi = \|\text{dev } (\sigma - \beta)\| - \sqrt{\tfrac{2}{3}} K(\alpha) \tag{6}$$

Von exponentieller Verfestigung spricht man, wenn

$$K(\alpha) = K_0 + H\alpha + (K_{inf} - K_0)(1 - e^{-\omega\alpha}).$$

Verschwindet der letzte Summand, so spricht man von linearer Verfestigung.

Bei $\psi(\sigma, q) = 0$ setzt plastisches Fließen ein. Man bezeichnet die Flußweite - auch Konsistenzparameter genannt - mit γ. Dieser muss folgende Bedingungen erfüllen:

$$\gamma \geq 0 \text{ (Ladebedingung),} \tag{7}$$

$$\gamma \cdot \psi(\sigma, q) = 0 \text{ (Komplementarität),} \tag{8}$$

$$\gamma \cdot \dot{\psi}(\sigma, q) = 0 \text{ (Konsistenzbedingung).} \tag{9}$$

Die Richtung $r(\sigma, q)$ in die das Material fließt, wenn plastisches Fließen einsetzt, wird durch

$$\dot{\epsilon}^p = \gamma \cdot r(\sigma, q) \text{ (Flussregel)} \tag{10}$$

gegeben. Von assoziierter Flussrichtung spricht man, wenn die Flussrichtung ein Vielfaches von $\frac{\partial \psi}{\partial \sigma}$ ist. Auch die inneren Materialzustände erfahren beim plastischen Fließen eine Veränderung:

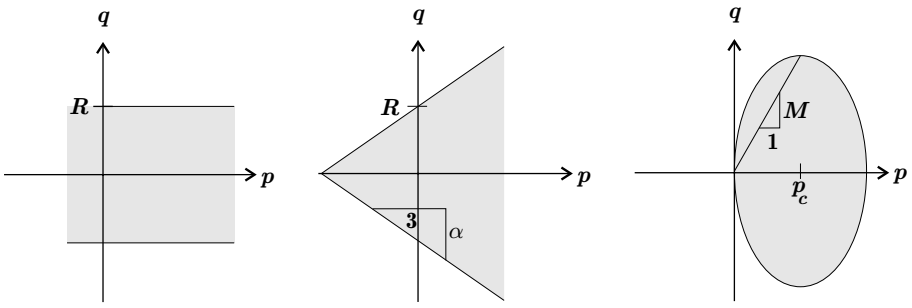
$$\dot{q} = \gamma \cdot h(\sigma, q) \text{ (Verfestigungsgesetz),} \tag{11}$$

In dieser Arbeit wird von assoziiertem Fließen für die inneren Materialzustände ausgegangen:

$$\dot{q} = \gamma \cdot \frac{\partial \psi}{\partial q} \text{ (assoziierte innere Materialevolution).} \tag{12}$$

Die meisten numerischen Beispiele in dieser Arbeit wurden mit perfekter Prandtl-Reuss-Plastizität berechnet. Dies bezeichnet ein linear isotropes elastisches Materialverhalten im Innern der von Mises Fließfläche und gar keine Verfestigung ($H = 0$). Außerdem wird dabei die assoziierte (Levy-Saint Venant) Fließrichtung verwendet ([**57**, S. 89]).

In Böden kommt es dazu, dass einzelne Sandkörner in Lücken fallen, wenn diese entstehen und die Reibung klein genug ist. Der volumetrische Anteil des Verzerrungstensors ist ein Indiz dafür und wird deswegen in Modellen für Böden benutzt: Das Drucker-Prager-Modell für Böden hat einen weiteren Materialparameter: den



(a) von Mises (b) Drucker-Prager (c) Cam-Clay

ABBILDUNG 1.1. Zulässige Spannungsbereiche verschiedener Plastizitäts-Modelle

Reibungskoeffizient $\mu_2 < \sqrt{2}$. Mit diesem lautet das Drucker-Prager-Modell

$$\psi \quad = \alpha I_1 + \sqrt{J_2} - R = \quad \sqrt{\frac{3}{2}}\mu_2 \,\mathrm{vol}\ \sigma + \|\mathrm{dev}\ \sigma\| - R \leq 0$$

Das modifizierte Cam-Clay-Modell ([**10**],[**26**]) für Lehmböden lautet mit abgekürzter Schreibweise für deviatorische Spannung $q = \sqrt{\frac{3}{2}}\,\|\mathrm{dev}\ \sigma\|$ und volumetrischem Druck $p = -\frac{1}{3}tr(\sigma)$ (in der Bodenmechanik wird manchmal mit entgegengesetztem Vorzeichen gearbeitet):

$$\psi = \frac{q^2}{M^2} + p(p - p_c).$$

Dabei gibt p_c den Vorverfestigungsdruck und M die Steigung der kritischen Geraden an (siehe Abbildung 1.1). Verfestigung wird in [**10**] durch eine Evolution von p_c implementiert. Eine Besonderheit beim Cam-Clay-Modell ist, dass auch die Elastizitätsparameter nicht konstant gegeben sind, sondern auch eine Evolution erfahren. Außerdem wird dort ein poröses Medium mit einer variablen Sättigung modelliert.

Im Prinzip können Materialmodelle mit beliebig vielen Materialparametern (und Zustandsvariablen) konstruiert werden. Diese aus Versuchen zu gewinnen, bereitet aber großen Aufwand. Daher beschränkt man sich meist auf wenige Materialparameter, deren physikalisch-mechanische Bedeutung zudem verstanden wird und die mit einfachen Versuchen (möglichst ohne inverse Parameteridentifikation) gefunden werden können.

Bei granularen Medien sind die Fließregeln überdies nicht assoziativ. Nichtassoziative Fließregeln führen indes wiederum klassischer Weise auf nicht symmetrische Materialtensoren. Ein neueres Modell [**22**] legt allerdings dar, dass auch

nicht-assoziative Fließregeln immer zu einem symmetrischen Materialtensor führen. Dazu wird eine Unterscheidung zwischen irreversiblem und plastischem Fließen getroffen. Da der nicht symmetrische Anteil allein auf den irreversiblen Fluss zurückzuführen sei, erhält man einen symmetrischen Materialtensor.

1.3. Die Impulsbilanz

Im Inneren eines Materials herrschen unter anderem folgende Kräfte:

(1) Kräfte durch innere Spannungen div σ,
(2) Impulskräfte durch Trägheit $\rho \cdot \dot{v}$ (mit Dichte ρ und Geschwindigkeit $v = \dot{u}$) und
(3) weitere Volumenkräfte f wie z.B. Gravitation, Elektrostatik oder Magnetismus.

Die Impulserhaltung liefert folgende Gleichung (siehe z.B.[**62**, S. 34]):

$$\text{div } \sigma = f + \rho \cdot \ddot{u} \text{ (lokale Impulsbilanz).} \tag{13}$$

Die Divergenz div für eine Matrix ist dabei zeilenweise definiert durch $\text{div } u := \nabla \cdot u := \sum_{i=1}^{3} \frac{\partial u}{\partial x_i}$, also

$$\text{div } (\sigma) := \begin{pmatrix} \text{div } (\sigma_{11} & \sigma_{12} & \sigma_{13}) \\ \text{div } (\sigma_{21} & \sigma_{22} & \sigma_{23}) \\ \text{div } (\sigma_{31} & \sigma_{32} & \sigma_{33}) \end{pmatrix} := \begin{pmatrix} \frac{\partial \sigma_{11}}{\partial x} + \frac{\partial \sigma_{12}}{\partial y} + \frac{\partial \sigma_{13}}{\partial z} \\ \frac{\partial \sigma_{21}}{\partial x} + \frac{\partial \sigma_{22}}{\partial y} + \frac{\partial \sigma_{23}}{\partial z} \\ \frac{\partial \sigma_{31}}{\partial x} + \frac{\partial \sigma_{32}}{\partial y} + \frac{\partial \sigma_{33}}{\partial z} \end{pmatrix}.$$

Im stationären Fall (eingeschwungener Zustand) ist das System in Ruhe und die Impulskräfte verschwinden. Auch bei sehr langsamen Bewegungen kann man die Impulsbilanz vernachlässigen, was im folgenden getan wird, da der Impuls proportional zur Geschwindigkeit und somit verschwindend gering ist. Vernachlässigt man weiterhin alle Volumenkräfte, erhält man aus (13) folgende Kraftgleichung:

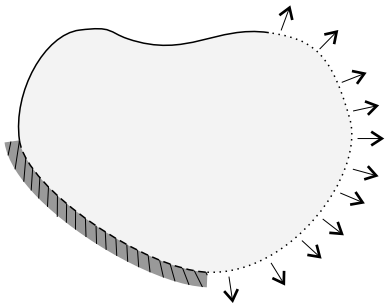
$$\text{div } (\sigma) = 0. \tag{14}$$

1.4. Definition des plastischen Anfangswertproblems

Das elasto-plastische Anfangswertproblem modelliert einen Festkörper, dessen Zustand man zu einem Zeitpunkt $t = 0$ kennt. Gesucht ist nach der Verschiebung des Festkörpers im Laufe der Zeit, in Abhängigkeit von Randbedingungen, die auch von der Zeit t abhängen können.

Sei $\Omega \subset \mathbb{R}^3$ ein zusammenhängendes Gebiet mit fast überall Lipschitz-stetigem Rand $\Gamma = \partial\Omega$. Dann seien $\Gamma_D \subset \Gamma$ und $\Gamma_N \subset \Gamma$ nicht überlappende Teilmengen vom Rand, d.h. $\Gamma_D \cap \Gamma_N = 0$, $\Gamma_D \cup \Gamma_N = \Gamma$. Dazu sei die Rand-Normalen n gegeben.

Es sei die Verschiebung eine stetige Abbildung (kein Materialbruch) $u : \Omega \times [0, t_{\max}] \to \mathbb{R}^3$, die Spannung $\sigma : \Omega \times [0, t_{\max}] \to \mathbb{R}^{3\times 3}$, die internen Zustandsvariablen $q : \Omega \times [0, t_{\max}] \to \mathbb{R}^m$ und die Fließgrenze gegeben durch: $\psi : \mathbb{R}^{3\times 3} \times \mathbb{R}^m \to \mathbb{R}$, sowie zwei weitere Funktionen $r : \mathbb{R}^{3\times 3} \times \mathbb{R}^m \to \mathbb{R}$, und $h : \mathbb{R}^{3\times 3} \times \mathbb{R}^m \to \mathbb{R}^m$ gegeben, welche die Richtung und Art der Materialverfestigung beschreiben. Das elastische Materialverhalten sei eine lineare Abbildung $\mathcal{C} : \mathbb{R}^{3\times 3} \to \mathbb{R}^{3\times 3}$.



Dirichletrand (gestrichelt) und Neumannrand (gepunktet)

ABBILDUNG 1.2. Gebietsränder

Die klassische raten-unabhängige Plastizität (siehe [**57**, S. 83]) lässt sich dann zusammenfassen durch folgende Gleichungen (14), (1), (10), (11), (5), (7)-(9):

$$
\begin{aligned}
\operatorname{div}\ \sigma &= 0 \text{ in } \Omega \text{ (Impulsbilanz)},\\
\sigma &= \mathcal{C}(\epsilon(u) - \epsilon^p) \text{ (hookesches Gesetz)},\\
\dot{\epsilon}^p &= \gamma \cdot r(\sigma, q) \text{ (Material-Evolution: Flussregel)},\\
\dot{q} &= \gamma \cdot h(\sigma, q) \text{ (innere Material-Evolution: Verfestigungsgesetz)},
\end{aligned}
$$

$$
\begin{aligned}
\gamma &\geq 0 \text{ (Lade-Bedingung)},\\
\psi(\sigma, q) &\leq 0 \text{ (Entladung)},\\
\gamma \cdot \psi(\sigma, q) &= 0 \text{ (Komplementarität)},\\
\gamma \cdot \dot{\psi}(\sigma, q) &= 0 \text{ (Konsitenz-Bedingung)},
\end{aligned}
\tag{15}
$$

$$
\begin{aligned}
u &= u_\Gamma \text{ auf } \Gamma_D \text{ (Dirichlet-Randbedingung)},\\
\sigma \cdot n &= \sigma_\Gamma \text{ auf } \Gamma_N \text{ (Neumann-Randbedingung)},
\end{aligned}
$$

$$
\begin{aligned}
u_{|t=0} &= u_0 \text{ (Ausgangslage)},\\
q_{|t=0} &= q_0 \text{ (innerer Startzustand)},\\
\epsilon^p_{|t=0} &= \epsilon^p_0 \text{ (initiale plastische Verzerrung)}.
\end{aligned}
$$

Vorausgesetzt wird dabei, dass u_0, q_0 und ϵ^p_0 zulässige Ausgangswerte sind, also das Differentialgleichungssystem zum Zeitpunkt $t = 0$ erfüllen.

1.5. Zeitdiskretisierung durch Returnmapping

Um das Differential-Algebraische-Gleichungssystem (DAE) (15) zu lösen ist eine Zeitdiskretisierung nötig.

Eine DAE ist im Allgemeinen gegeben durch eine Funktion F für die gelten soll:

$$0 = F(t, y, \dot{y}) \, \forall t \in [t_0, T]. \tag{16}$$

Gesucht ist dabei $y(t)$ und die Anfangswerte von y sind zum Zeitpunkt t_0 vorgegeben.

Ein s-stufiges Runge-Kutta-Verfahren für gewöhnliche Differentialgleichungen (ODEs) ist gegeben durch eine Koeffizientenmatrix A mit Zeilen A_i, Quadraturpunkte c_j und Quadraturgewichte b_j.

Formal kann man ein Runge-Kutta-Verfahren für ODEs auf DAEs übertragen, indem man bei einer Zeitschrittlänge τ approximiert:

$$y \approx \hat{y}_{n-1} + \tau A_i \hat{f} \text{ für } i = 1, .., s,$$

$$\dot{y} \approx \hat{f}.$$

Man erhält damit Runge-Kutta-Verfahren für DAEs:

$$\begin{aligned} F(t_{n-1} + \tau c_i, \hat{y}_{n-1} + \tau A_i \hat{f}, \hat{f}_i) &= 0 \text{ für } i = 1, .., s, \\ \hat{y}_n &= \hat{y}_{n-1} + \tau b^T \hat{f}_i \end{aligned} \tag{17}$$

Da DAEs aber im Allgemeinen keine ODEs sind ist i.A. nicht zu erwarten, dass man bei der formalen Übertragung von Runge-Kutta-Verfahren für ODEs auf DAEs die gleiche Konvergenzordnung wie bei ODEs erzielen kann. Tatsächlich muß A weitere Bedingungen erfüllen, deren Nichterfüllung zu einer Reduktion der Konvergenzordnung führen kann ([**44**]). Die Reduktion der Konvergenzordnung hängt wesentlich vom ganzzahligen Index i einer DAE ab, für den mehrere unterschiedliche Definitionen existieren z.B. dass die DAE nach i-facher Differentiation eine ODE ergibt. Eine ODE ist also immer eine DAE vom Index 0. Viskoeleastisches Fließen führt auf eine DAE vom Index 1 und plastisches Fließen zum Index 2. Mechanische Systeme welche den Impuls nicht vernachlässigen haben in ihrer Grundform sogar den Index 3.

Die Ordnungsreduktion für Index-1 DAEs ist bei den bekannten Runge-Kutta Methoden sehr moderat bis gar nicht vorhanden. Aber bereits beim Index-2 reduziert sich die Konvergenz für die meisten Koeffizientensätze auf eine Konvergenzordnung $\hat{p} \leq \frac{p}{2}$.

Die bei der Verwendung von Alexanders DIRK-Verfahren (nach [**2**]) bemerkte Reduktion der Konvergenzordnung ist daher nicht, wie in [**61**, S. 470] angemerkt auf eine mangelnde Regularität der Lösung zurückzuführen, sondern ein Resultat der formalen Übertragung. Die numerischen Experimente in [**14**] zeigen, dass eine Konvergenzordung von $\hat{p} = 2$ mit dem Radau-IIA-Verfahren möglich ist und erst dann die mangelnde Regularität Wirkung zeigt. Büttner und Simeon schlugen darin eine genauere Bestimmung des Übergangs vom elastischen zum plastischen

Materialverhalten vor, die auch einen Hauptteil von der Doktorarbeit [**1**] ausmachte. In [**15**] finden sich schließlich Konvergenz und Stabilitätsanalysen für den Fall assoziativer Fließregel.

Die Entwicklung numerischer Methoden für DAEs ist immer noch ein aktiver Forschungsbereich, und die Reduktion der Konvergenzordnung scheint vermeidbar zu sein durch die Abkehr von der direkten formalen Übertragung (17) (siehe z.B. [**36**]).

Wir verwenden das implizite Euler-Verfahren ($A = 1$, $b = 1$, $c = 1$). Dieses leidet nicht unter einer Ordnungsreduktion ([**34**]). Es liefert Inkremente σ^{inc}, u^{inc}, q^{inc} um aus den Geschichtsvariablen des vorhergehenden Zeitschrittes σ^{old}, u^{old}, q^{old} aktuellen Werte im Zeitschritt t zu berechnen:

$$\begin{aligned} \sigma_t &= \sigma^{\text{old}} + \sigma^{\text{inc}} \\ u_t &= u^{\text{old}} + u^{\text{inc}} \\ q_t &= q^{\text{old}} + q^{\text{inc}} \end{aligned} \tag{18}$$

Im Fall der perfekten Plastizität führt dies auf das sogenannte Returnmapping-Schema:

Die Komplementaritätsbedingungen (5), (7)-(9) werden dabei in jedem Zeitschritt gelöst, indem die Versuchsspannungen

$$\sigma^{trial} := \mathcal{C}(\epsilon(u_t) - \epsilon_t^p) = \sigma^{\text{old}} + \mathcal{C}\epsilon(u_t)$$

in einen gültigen Spannungszustand überführen werden:

$$\sigma_t \quad = \quad \begin{cases} \sigma^{trial} & \text{wenn } \psi(\sigma^{trial}, q_t) \leq 0 \\ \sigma^{trial} - \mathcal{C}\gamma^* r(\sigma_t, q_t) & \text{wenn } \psi(\sigma^{trial}, q_t) \geq 0 \end{cases} \tag{19}$$

Dazu muss die lokale Gleichung

$$\psi(\sigma_t, q_t) = 0 \text{ wenn } \psi(\sigma_t^{trial}, q_t) \geq 0 \tag{20}$$

unter der Nebenbedingung

$$(\frac{\partial \psi}{\partial \sigma}(\sigma_t, q_t))^T \cdot \frac{\partial \psi}{\partial \sigma}(\sigma_t, q_t) \quad > \quad 0. \tag{21}$$

gelöst werden. Bedingung (20) stellt sicher, dass die Spannung auf die Fließfläche projiziert wird, während (21) nötig ist um die Nullstelle von ψ auszuzeichnen, die näher an der elastischen Versuchsspannung liegt. Kurioserweise wird letztere Bedingung in keiner bisherigen Publikation explizit erwähnt, obwohl das Gleichungssystem (20) allein zwei Lösungen hat.

Wenn $r(\sigma_t, q_t) = r(\sigma_t^{trial}, q_t) =: \eta$ ist, vereinfacht sich (20) zu der eindimensionalen Gleichung:

$$\psi(\lambda^*) := \psi(\sigma_t^{trial} - \mathcal{C}\gamma^*\eta, q_t) = 0$$

Im Fall der perfekten Prandtl-Reuss-Plastizität erhält man dann die klassische Returnmapping-Projektion

$$\sigma_t = P_R \sigma^{trial} = \begin{cases} \text{vol}\,(\sigma^{trial}) + R \frac{\text{dev}\ \sigma^{trial}}{\|\text{dev}\ \sigma^{trial}\|} & \|\text{dev}\ \sigma^{trial}\| \geq R \\ \sigma^{trial} & \|\text{dev}\ \sigma^{trial}\| \leq R \end{cases}, \tag{22}$$

Allgemeiner erhält man aus (19) eine Funktion $\sigma(\epsilon(u^{\text{inc}}), \sigma^{\text{old}}, q^{\text{old}})$. Da in jedem Zeitschritt die Geschichtsvariablen konstant sind, lassen wir diese der Übersichtlichkeit halber weg und erhalten mit (4), (18) und (15) für jeden Zeitschritt das Randwertproblem (das ein System partieller Differentialgleichungen ist):

$$\begin{aligned} \text{div}\ \sigma_t &= 0 \text{ in } \Omega \text{ (Impulsbilanz)}, \\ \sigma_t &= \sigma(\epsilon(u^{\text{inc}})) \text{ (konstitutive Gleichung)}, \\ u_t &= u_\Gamma \text{ auf } \Gamma_D \text{ (Dirichlet-Randbedingung)}, \\ \sigma_t \cdot n &= \sigma_\Gamma \text{ auf } \Gamma_N \text{ (Neumann-Randbedingung)}. \end{aligned} \tag{23}$$

KAPITEL 2

Die gemischte Least-Squares-Formulierung

Ist ein Gleichungssystem

$$\begin{aligned} r_1(x) &= 0 \\ r_2(x) &= 0 \\ &\vdots \\ r_n(x) &= 0 \end{aligned}$$

kurz

$$R(x) = 0$$

gegeben, besteht die Methode kleinster Quadrate (Least-Squares) darin, das Funktional

$$F(x) := \sum_{i=1}^{n} r_i(x)^2 = \|R\|^2$$

zu minimieren:

$$\min_{x \in X} F(x)$$

Ist der Ansatzraum X hinreichend groß gewählt, so dass das Gleichungssystem eine Lösung $x^* \in X$ hat, dann ist $F(x^*) = 0$. Außerdem kann man dann die einzelnen Residuen r_i unterschiedlich gewichten

$$F_D(x) := \sum_{i=1}^{n} w_i r_i(x)^2 = \|DR\|^2, D = \begin{pmatrix} \sqrt{w_1} & 0 & \cdots & 0 \\ 0 & \sqrt{w_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{w_n} \end{pmatrix}$$

ohne dass sich die Lösung x^* ändert. Ist der Ansatzraum für x zu klein, erhält man ein (möglichst kleines) Residuum $R(x^*) \neq 0$, das von den positiven Gewichten w_i abhängt.

Wir wollen diese Methode verwenden um (23) mit einem Finite-Element-Ansatz $u_h \in V_h$ (h bezeichnet den größten Durchmesser aller Elemente in der verwendeten Triangulierung $\mathcal{T}$) zu lösen. Dabei kann die Erfüllung der Dirichlet Randbedingung fest durch den Ansatzraum V_h vorgegeben werden. Außerdem lassen sich die Gleichungen zusammenfassen zur Differentialgleichung 2. Ordnung unter einer Neumann-Randbedingung:

$$\begin{aligned} \text{div } \sigma(\epsilon(u_h)) &= 0 \text{ in } \Omega \\ \sigma_t \cdot n &= \sigma_\Gamma \text{ auf } \Gamma_N \end{aligned} \tag{24}$$

Die reine Least-Squares-Formulierung lautet dann unter Verwendung der Sobolev-Norm und einem Randintegral

$$\min_{u_h \in V_h} \|\text{div } \sigma(\epsilon(u_h))\|^2_{0,\Omega} + \|\sigma(\epsilon(u_h)) \cdot n - \sigma_\Gamma\|^2_{\Gamma_N} .$$

Diese Formulierung hat jedoch den Nachteil, dass es schwierig ist einen Finite-Element-Ansatzraum V_h anzugeben, so dass $\sigma(\epsilon(u_h)) \in H(\text{div }, \Omega)$ ist. Stattdessen wählen wir einen gemischten Ansatz: Betrachtet man σ als eigenständige Variable ist (23) nur eine Differentialgleichung 1. Ordnung und man kann die Neumann Randbedingung in den Ansatzraum W_h für σ_h einbauen. Man erhält dann die gemischte Least-Squares-Formulierung

$$\min_{u_h, \sigma_h \in V_h \times W_h} \|\text{div } \sigma_h\|^2_{0,\Omega} + \|\sigma(\epsilon(u_h)) - \sigma_h\|^2_{0,\Omega} .$$

Alternativ kann man dies auch schreiben als

$$\min_{u, \sigma \in V \times W} \|R(\sigma_h, u_h)\|^2_{0,\Omega}$$

mit

$$R(\sigma_h, u_h) = \begin{pmatrix} \sigma(\epsilon(u_h)) - \sigma_h \\ \text{div } \sigma_h \end{pmatrix}.$$

Die Gewichtung der konstitutiven Beziehung mit $\mathcal{C}^{-1/2}$ führt zu dem Least-Squares-Funktional

$$F(\sigma_h, u_h) = \|\text{div } \sigma_h\|^2_{0,\Omega} + \left\|\mathcal{C}^{-1/2}(\sigma_h - \sigma(\epsilon(u_h)))\right\|^2_{0,\Omega} . \tag{25}$$

2.1. Das Least-Squares-Funktional als Fehlerschätzer:

Im folgenden zeigen wir, dass das Least-Squares-Funktional (25) ein effizienter Fehlerschätzer ist. Dabei müssen wir uns auf Materialmodelle mit Verfestigung beschränken. Für Materialmodelle ohne Verfestigung kann ohnehin nicht gezeigt werden, dass (23) eine Lösung hat. In [**47**] findet man einen Beweis dafür, dass es aber immer eine eindeutige Lösung eines mit linearer Verfestigung regularisierten Problems gibt und dass diese mit Verfestigungsparameter $H \to 0$ gegen eine Lösung eines Variationsproblems konvergiert, welches eine Lösung des ursprünglichen Problems ist, falls diese existiert. Außerdem wird dort gezeigt, dass wenn man die Gitterweite h in gewisse Relation zum Regularisierungsparameter des Problems setzt, die analytische Finite-Element-Lösung gegen eine eindeutige Lösung konvergiert.

Wir betrachten assoziative J_2-Plastizität mit Verfestigung nach (5), (6), (10) und (12).

Dabei gelte

$$\begin{aligned} K(\alpha) &\geq K_0 > 0, \\ K'(\alpha) &\geq K_1 > 0, \end{aligned} \tag{26}$$

was für lineare und exponentielle Verfestigung zutrifft, nicht jedoch für perfekte Plastizität. Wir definieren folgende Unterräume von Sobolev-Räumen $H(\Omega)$:

$$H_{\Gamma_N}(\text{div }, \Omega) := \{s \in H(\Omega)^3 \ : \ \text{div } \ s \in H(\Omega), \ v \cdot n = 0 \text{ auf } \Gamma_N\},$$

$$\overset{1}{H}_{\Gamma_D}(\Omega) := \{v \in H^1(\Omega) \ : \ v = 0 \text{ auf } \Gamma_D\}.$$

Gesucht ist die Lösung von (23)

$$\begin{aligned} \text{div } (\sigma + \sigma^{\text{old}}) &= 0 \\ \sigma - \sigma(\epsilon(u)) &= 0. \end{aligned} \tag{27}$$

Um die folgenden Gleichungen abzukürzen wird im folgenden die Norm

$$\|\sigma\|_c := \left\| C^{-1/2}\sigma \right\|_{0,\Omega}$$

verwendet sowie die Differenzen $\tau := \sigma_h - \sigma$, $\tilde{\tau} := \sigma(\epsilon(u_h)) - \sigma(\epsilon(u))$, $v := u_h - u$ abgekürzt.

Um zu zeigen, dass das nichtineare Least-Squares-Funktional ein effizienter Fehlerschätzer ist braucht man für den Beweis nach Starke ([**58**]) zunächst, dass die plastische Verzerrung klein gegenüber der elastischen Verzerrung im Sinne von folgendem Lemma ist:

LEMMA 2.1.1. *Unter den Voraussetzungen (26) existiert eine Konstante $C_R \in [0,1)$, so dass für alle $u, u_h \in H^1_{\Gamma_D}(\Omega)^3$*

$$\|\mathcal{C}\epsilon(v) - \tilde{\tau}\|_c \leq C_R \|\mathcal{C}\epsilon(v)\|_c . \tag{28}$$

BEWEIS. Dafür reicht es zu zeigen, dass die plastische Verzerrung punktweise klein gegenüber der elastischen Verzerrung ist (mit der Schreibweise $\sigma_h^{trial} := \sigma^{\text{old}} + \mathcal{C}\epsilon(u_h)$ und entsprechendem γ_h):

$$\left\| \gamma \frac{\text{dev } \sigma^{trial}}{\|\text{dev } \sigma^{trial}\|} - \gamma_h \frac{\text{dev } \sigma_h^{trial}}{\left\|\text{dev } \sigma_h^{trial}\right\|} \right\| \leq C_R \left\|\text{dev } \sigma^{trial} - \text{dev } \sigma_h^{trial}\right\|$$

Dies gilt (Details siehe [**58**]) unter den Voraussetzungen (26) mit

$$C_R = \max\left(\left(1 + \frac{K_1}{3\mu}\right)^{-1}, \left(1 + \sqrt{\frac{2}{3}}\frac{K_0}{2\mu}(\sup_{\text{x}} \|\text{dev } \epsilon(u(\text{x}))\|)^{-1}\right)^{-1} \right) < 1.$$

□

THEOREM 2.1.2. *Sei zu fest vorgegebenem σ^{old}, α^{old} die Lösung von (27) gegeben durch $\sigma \in H_{\Gamma_N}(\text{div }, \Omega)^3$, $u \in H^1_{\Gamma_D}(\Omega)^3$ dann wird das Fehlermaß*

$$\|e\|_*^2 := \|\text{div } (\sigma - \sigma_h)\|_{0,\Omega}^2 + \left\|\mathcal{C}^{-1/2}(\sigma - \sigma_h)\right\|_{0,\Omega}^2 + \left\|\mathcal{C}^{1/2}\epsilon(u - u_h)\right\|_{0,\Omega}^2 \tag{29}$$

unter den Voraussetzungen (26) durch das Least-Squares-Funktional (25)

$$\eta := \sqrt{F}$$

eingeschachtelt :

$$C_{eff}\eta \leq \|e\|_* \leq C_{rel}\eta,$$

wobei C_{rel} und C_{eff} zwei positive Konstanten sind, die nicht vom Lamé-Parameter λ abhängen.

BEWEIS. Zu zeigen ist also die Zuverlässigkeit

$$\|e\|_* \leq C_{\mathrm{rel}}\eta \tag{30}$$

und die Effizienz

$$C_{\mathrm{eff}}\eta \leq \|e\|_* \tag{31}$$

des Fehlerschätzers.

Wir zeigen daher unabhängig voneinander erst die Effizienz und anschliessend - mit Hilfe der kornschen Ungleichung - die Zuverlässigkeit. Für beide Beweisteile fügen wir die exakten Lösung u, σ von (27) in das Least-Squares-Funktional (25) ein und erhalten:

$$\begin{aligned} F(\sigma_h, u_h) &= \|\mathrm{div}\ (\sigma_h - \sigma)\|^2_{0,\Omega} + \left\|\mathcal{C}^{-1/2}\left(\sigma_h - \sigma(\epsilon(u_h)) - (\sigma - \sigma(\epsilon(u)))\right)\right\|^2_{0,\Omega} \\ &= \|\mathrm{div}\ \tau\|^2_{0,\Omega} + \|\tau - \tilde{\tau}\|^2_c . \end{aligned} \tag{32}$$

Beweisteil 1: Die Effizienz

Um die Effizienz des Fehlerschätzers zu zeigen ist die hintere Norm in (32) nach oben abzuschätzen. Dazu wird die Dreiecksungleichung und (28) angewandt:

$$\begin{aligned} \|\tau - \tilde{\tau}\|_c &= \|\tau - \mathcal{C}\epsilon(v) + \mathcal{C}\epsilon(v) - \tilde{\tau}\|_c \\ &\leq \|\tau - \mathcal{C}\epsilon(v)\|_c + \|\mathcal{C}\epsilon(v) - \tilde{\tau}\|_c \\ &\leq \|\tau - \mathcal{C}\epsilon(v)\|_c + C_R \|\mathcal{C}\epsilon(v)\|_c \\ &\leq \|\tau\|_c + \|C\epsilon(v)\|_c + C_R \|\mathcal{C}\epsilon(v)\|_c \\ &= \|\tau\|_c + (1 + C_R) \|\mathcal{C}\epsilon(v)\|_c \end{aligned}$$

Die Ungleichung $(A + B)^2 \leq 2(A^2 + B^2)$ liefert dann

$$\|\tau - \tilde{\tau}\|^2_c \leq 2 \|\tau\|_c + 2(1 + C_R)^2 \|\mathcal{C}\epsilon(v)\|^2_c ,$$

so dass mit (32) folgt:

$$\begin{aligned} F(\sigma_h, u_h) &\leq \|\mathrm{div}\ \tau\|^2_{0,\Omega} + 2 \|\tau\|^2_c + 2(1 + C_R)^2 \|\mathcal{C}v\|^2_c \\ &\leq 2(1 + C_R)^2 \|e\|^2_* . \end{aligned}$$

Die Effizienz (31) des Fehlerschätzer ist also gegeben mit

$$C_{\mathrm{eff}} = \sqrt{\frac{1}{2}}(1 + C_R)^{-1} < 1.$$

Beweisteil 2: Die Zuverlässigkeit

Um die Zuverlässigkeit (30) zu zeigen wird zunächst die Zerlegung einer symmetrischen Matrix $\hat{\tau} := \tau - \tilde{\tau}$ in ihren symmetrischen und asymmetrischen Teil eingeführt:

$$\hat{\tau} = \mathrm{sy}\ \hat{\tau} + \mathrm{as}\ \hat{\tau} \text{ mit } \mathrm{sy}\ \hat{\tau} = \frac{1}{2}(\hat{\tau} + \hat{\tau}^T),\ \mathrm{as}\ \hat{\tau} = \mathrm{sy}\ \hat{\tau} = \frac{1}{2}(\hat{\tau} - \hat{\tau}^T).$$

Asymmetrischer und symmetrischer Teil sind zueinander orthogonal

$$(\text{sy } \hat{\tau}, \text{as } \hat{\tau}) = \frac{1}{4}((\hat{\tau}, \hat{\tau}) - (\hat{\tau}^T, \hat{\tau}^T) + (\hat{\tau}^T, \hat{\tau}) - (\hat{\tau}, \hat{\tau}^T)) = 0,$$

so dass folgende Ungleichung gilt:

$$\|\hat{\tau}\|^2 = \|\text{sy } \hat{\tau}\|^2 + \|\text{as } \hat{\tau}\|^2 \geq \|\text{as } \hat{\tau}\|^2 . \tag{33}$$

Da die Spannungen rein symmetrisch sind und der Materialtensor $\mathcal{C}$ die Symmetrie erhält ist

$$\text{as } (\mathcal{C}^{-1/2}(\sigma(\epsilon(u)))) = 0.$$

Normiert man außerdem die Materialparameter auf $\mu = 1$ erhält man aus (33) und as $\tilde{\tau} = 0$

$$\begin{aligned} 2\|\tau - \tilde{\tau}\|_c^2 &\geq 2\left\|\text{as } (\mathcal{C}^{-1/2}(\tau))\right\|_{0,\Omega}^2 \\ &= \|\text{as } \tau\|_{0,\Omega}^2 . \end{aligned} \tag{34}$$

Aus (32) und (34) folgt weiter

$$\begin{aligned} 3F(\sigma_h, u_h) &= 3\|\text{div } \tau\|_{0,\Omega}^2 + 3\|\tau - \tilde{\tau}\|_c^2 \\ &\geq 1\|\text{div } \tau\|_{0,\Omega}^2 + 1\|\tau - \tilde{\tau}\|_c^2 + \|\text{as } \tau\|_{0,\Omega}^2 . \end{aligned} \tag{35}$$

Einfügen von $\mathcal{C}^{1/2}\epsilon(u_h - u)$ in den mittleren Term liefert die vom frei wählbaren $\rho \in (0,1)$ abhängige Ungleichung

$$\begin{aligned} \|\tau - \tilde{\tau}\|_c^2 &= \|\tau - \mathcal{C}\epsilon(v) + \mathcal{C}\epsilon(v) - \tilde{\tau}\|_c^2 \\ &\geq (1-\rho)\|\tau - \mathcal{C}\epsilon(v)\|_c^2 - (\frac{1}{\rho} - 1)\|\mathcal{C}\epsilon(v) - \tilde{\tau}\|_c^2 \\ &\geq (1-\rho)\|\tau - \mathcal{C}\epsilon(v)\|_c^2 - (\frac{1}{\rho} - 1)\mathcal{C}_R^2\|\mathcal{C}\epsilon(v)\|_c^2 . \end{aligned} \tag{36}$$

Die letzte Ungleichung folgte dabei aus (28). Der Rest des Beweises ist dadurch unabhängig von irgendwelchen Plastizitätseffekten (und kann analog zum Beweis für den elastischen Fall in **[16]** fortgesezt werden) und man hat, mit (36) in (35) eingesetzt, nur noch

(37)

$$\begin{aligned} G(\tau, v) &:= \frac{\|\text{div } \tau\|_{0,\Omega}^2 + \|\text{as } (\tau)\|_{0,\Omega}^2}{1-\rho} + \|\tau - \mathcal{C}\epsilon(v)\|_c^2 - \frac{C_R^2}{\rho}\|\mathcal{C}\epsilon(v)\|_c^2 \\ &= \frac{\|\text{div } \tau\|_{0,\Omega}^2 + \|\text{as } (\tau)\|_{0,\Omega}^2}{1-\rho} + \|\tau\|_c^2 - 2\langle\tau, \epsilon(v)\rangle_{0,\Omega} + (1 - \frac{C_R^2}{\rho})\|\mathcal{C}\epsilon(v)\|_c^2 \end{aligned}$$

nach unten abzuschätzen. Die Zerlegung von τ in seinen symmetrischen und asymmetrischen Teil liefert nach anschließender partieller Integration folgende Ungleichung für das auftretende Skalarprodukt:

$$\begin{aligned}
\langle \tau, \epsilon(v) \rangle_{0,\Omega} &= \langle \text{sy } \tau, \epsilon(v) \rangle_{0,\Omega} + \langle \text{as } \tau, \epsilon(v) \rangle_{0,\Omega} \\
&= \langle \text{sy } \tau, \epsilon(v) \rangle_{0,\Omega} \\
&= \langle \text{sy } \tau, \nabla v \rangle_{0,\Omega} \\
&= \langle \tau, \nabla v \rangle_{0,\Omega} - \langle \text{as } \tau, \nabla v \rangle_{0,\Omega} \\
&= - \langle \text{div } \tau, v \rangle_{0,\Omega} - \langle \text{as } \tau, \nabla v \rangle_{0,\Omega} \\
&\leq \frac{1}{2\delta}(- \langle \text{div } \tau, v \rangle_{0,\Omega} - \langle \text{as } \tau, \nabla v \rangle_{0,\Omega}) + \frac{\delta}{2}(\|v\|^2_{0,\Omega} + \|\nabla v\|^2_{0,\Omega}) \qquad (38)
\end{aligned}$$

Dabei ist $\delta \in (0,1)$ frei wählbar. Die kornsche Ungleichung (siehe [**11**, Abschnitt VI.3])

$$\|v\|^2_{0,\Omega} + \|\nabla v\|^2_{0,\Omega} \leq C_K \|\mathcal{C}\epsilon(v)\|^2_c$$

liefert schließlich mit (38) in (37) eingesetzt

$$\begin{aligned}
G(\tau, v) &\geq (\frac{1}{1-\rho} - \frac{1}{\delta})(\|\text{div } \tau\|^2_{0,\Omega} + \|\text{as } (\tau)\|^2_{0,\Omega}) + (1 - \frac{C_R^2}{\rho} - C_K\delta) \|\mathcal{C}\epsilon(v)\|_c + \|\tau\|^2_c \\
&\geq (\frac{1}{1-\rho} - \frac{1}{\delta}) \|\text{div } \tau\|^2_{0,\Omega} + (1 - \frac{C_R^2}{\rho} - C_K\delta) \|\mathcal{C}\epsilon(v)\|_c + \|\tau\|^2_c
\end{aligned}$$

Nun kann man $\rho \in (C_R^2, 1)$ so wählen, dass

$$\frac{C_K(1-\rho)}{1 - \frac{C_R^2}{\rho}} < 1$$

gilt (da die linke Seite mit $\rho \to 1$ verschwindet) und folglich $\delta \in (0,1)$ mit

$$\delta = \left(\frac{1-\rho}{C_K} \left(1 - \frac{C_R^2}{\rho} \right) \right)^{1/2}$$

wählen, wodurch man die Zuverlässigkeit (30) des Least-Squares-Funktionals erhält. □

Durch die lokale Auswertung des Least-Squares-Funktionals erhält man außerdem einen lokalen isotropen Fehlerschätzer für jedes finite Element $T \in \mathcal{T}$:

$$\eta_T = \|R(u_h, \sigma_h)\|_{0,T}$$

Wir vergleichen seine Form kurz mit anderen üblichen Fehlerschätzern: Ein Residuen-basierter Fehlerschätzer in der FEM ist beispielsweise ([**59**, S. 25])

$$\eta_T = \left(h_T^2 \|R_T(u_T)\|^2_T + \sum_{E \subset T} \frac{1}{2} h_E \|R_E(u_T)\|^2_E \right)^{1/2}$$

mit

$$\begin{aligned} R_T(u_\mathcal{T}) &= \text{div } \sigma(u_\mathcal{T}) + f \text{ und} \\ R_E(u_\mathcal{T}) &= \begin{cases} [n\sigma(u_\mathcal{T})] & \text{für alle inneren Kanten} \\ n\sigma(u_\mathcal{T}) - \sigma_\Gamma & \text{für alle Neumann-Randkanten} \\ 0 & \text{für alle Dirichlet-Randkanten} \end{cases}, \end{aligned}$$

wobei $[\sigma]$ den Sprung zwischen zwei Elementen bezeichnet. Zum Teil wird als Fehlerschätzer auch nur der Sprung an den Kanten verwendet z.B. in [**19**] bei gemischter FEM.

Einen ganz anderen Ansatz gehen gradientenbasierte Fehlerschätzer die aus $\sigma(u_h)$ durch eine Patchrecovery-Nachbearbeitung eine bessere Lösung σ_h konstruieren und die Differenz $\|\sigma_h - \sigma(u_h)\|$ als Fehlerschätzer verwenden (z.B. [**56**]).

Im Vergleich zu den residualen Fehlerschätzern entfällt beim Least-Squares-Fehlerschätzer also der Sprungterm und dafür kommt ein Konsistenzterm $\|\sigma_h - \sigma(u_h)\|_T$ hinzu, der auch bei gradientenbasierten Fehlerschätzern verwendet wird.

2.2. Der verwendete Finite-Element-Ansatz und seine Approximationsgüte

Ein finites Element ist ein Element (hier Dreieck) T aus einer Zerlegung (hier Triangulierung) $\mathcal{T}$ zusammen mit einem endlich dimensionalen (finiten) Raum $V(T)$, der durch Basisfunktionen $\phi_i : T \to V(T) \in \mathbb{R}^n$ aufgespannt wird.

$$v(x) = \sum_i^n \alpha_i \phi_i(x),\ \alpha \in \mathbb{R}^n$$

Die Skalare α_i werden als Freiheitsgrade bezeichnet.

Lagrange finite Elemente haben Basisfunktionen, die an Punkten p_i den Wert eins haben und auf allen anderen Lagrange-Punkten verschwinden

$$\phi_i(p_j) = \delta_{i,j}.$$

Ihre Freiheitsgrade kann man anschaulich durch Punkte in einer grafischen Darstellung des Elementes kennzeichnen (Abbildung 2.1 auf der nächsten Seite).

Zu gegebenen Elementen gehört ein Finite-Element-Raum V_h, der Funktionen auf ganz $\mathcal{T}_h$ definiert.

Es bezeichne $P_k(T)$ den Raum der Polynome vom Grad $\leq k$ auf T in $\dim(T)$ Veränderlichen

$$P_k(T) = \{ \sum_{\beta_1+..+\beta_n \leq k} \alpha_{\beta_1,..,\beta_n} \prod_{i=1}^{\dim T} x_i^{\beta_i} \text{ mit } \alpha_{\beta_1,..,\beta_n} \in \mathbb{R} \}$$

und $(P_k(T))^n$ bezeichnet entsprechende vektorwertige Funktionen mit n unabhängigen polynomialen Komponenten.

Am weitesten verbreitet sind polynomiale finite Elemente $V_h(T) \subset (P_k(T))^n$. Finite Elemente heißen k-vollständig bezüglich des Polynomgrads k, wenn $(P_k(T))^n \subset V_h(T)$.

Mann erhält den Finite-Element-Raum der konformen Elemente durch

$$P_k = \{v \in C_0 : \ v_{|K} \in P_k(T)\}.$$

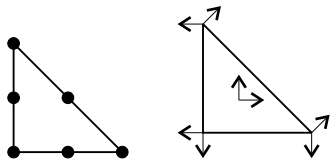
Für den Fluss $\sigma_{1,i} \in H_{\Gamma_N}(\text{div }, \Omega)$ wurden in [**46**] folgende finite Elemente eingeführt: Ist T ein n-Simplex-Element, dann lassen sich Raviart-Thomas-Finite-Element-Räume definieren als [**12**, S.116]:

$$RT_k(T) = (P_k(T))^n + P_k(T) \cdot x, \ x \in T$$

Wohlgemerkt sind die RT_k in jeder Vektorkomponente unvollständige Polynome vom Grad $k+1$. Es ist z.B. für $k = 1$ und $n = 2$ (Dreieckselemente)

$$\begin{aligned} RT_1(T) &= \begin{pmatrix} \alpha_1 + \alpha_2 x_1 + \alpha_3 x_2 \\ \alpha_4 + \alpha_5 x_1 + \alpha_6 x_2 \end{pmatrix} + (\alpha_7 x_1 + \alpha_8 x_2) \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= \begin{pmatrix} \alpha_1 + & \alpha_2 x_1 + & \alpha_3 x_2 + & \alpha_7 x_1^2 + & \alpha_8 x_1 x_2 + & 0 \cdot x_2^2 \\ \alpha_4 + & \alpha_5 x_1 + & \alpha_6 x_2 + & 0 \cdot x_1^2 + & \alpha_7 x_1 x_2 + & \alpha_8 x_2^2 \end{pmatrix} \forall \alpha \in \mathbb{R}^8. \end{aligned}$$

Die Basisfunktionen werden dabei über Freiheitsgrade definiert, die den Fluss $\sigma(p_i) \cdot n_i$ in bestimmten Punkten p_i und bestimmte Richtungen n_i messen. Für die p_i auf dem Rand von T werden die Randnormalen verwendet um $H(\text{div })$-Konformität über Ω zu erreichen (siehe Abbildung 2.1).



(a) P_2 (b) RT_1

ABBILDUNG 2.1. Die verwendeten Elemente für u_h und σ_h

Für finite Elemente mit vollständigen Polynomen vom Grad k über einer Triangulierung $\mathcal{T}_h$, gilt nach [**11**, S. 73] folgende Approximationsabschätzung für $0 \leq s < k \geq 1$:

$$\inf_{u_h} \|u - u_h\|_s = O(h^{k+1-s})|u|_{k+1} \ \forall u \in H^{k+1} \tag{39}$$

Um den Ansatzraum nicht unnötig groß zu wählen ist es wünschenswert, wenn die Lösung u^* einer Finite-Element-Formulierung ebenfalls diese Konvergenzordnung erreicht:

$$\|u - u_h^*\|_s = O(h^{k+1-s})|u|_{k+1}$$

Insbesondere ist in der FEM für quadratische Elemente ($k = 2$) also eine Methode mit Konvergenzordnung drei für die Verschiebungen

$$\inf_{u_h} \|u - u_h\|_0 = O(h^3)|u|_3$$

und Konvergenzordnung zwei für die Spannungen erwünscht:

$$\inf_{\sigma_h} \|\sigma - \sigma_h\|_0 = O(h^2)|\sigma|_2$$

Um eine Konvergenzordnung $k + 1$ für den Least-Squares-Fehlerschätzer η zu erreichen ist es nach (31), (29) notwendig, dass $\epsilon(u_h)$, σ_h und div σ_h mit Konvergenzordnung $k + 1$ approximiert werden können. In dieser Hinsicht sind in der gemischten LS-Formulierung die Raviart-Thomas-Elemente besonders geeignet zur Diskretisierung von σ: denn offensichtlich sind $RT_k(T)$ Elemente k-vollständig und ihre Divergenz div $(RT_k(T)) = P_k(T)$ ebenso. Also haben sie nach (39) die gleiche Approximationsgüte

$$\inf_{\sigma_h \in RT_k(T)} \|\text{div } \sigma - \text{div } \sigma_h\|_0 = O(h^{k+1})|\text{ div } \sigma|_{k+1}$$

für σ wie auch für div σ. Um auch $\epsilon(u)$ mit gleicher Approximationsgüte annähern zu können brauchen wir außerdem für die Verschiebung u Elemente mit k-vollständigem Gradienten. Dazu bieten sich die konformen Elemente vom Grad $p = k + 1$ an.

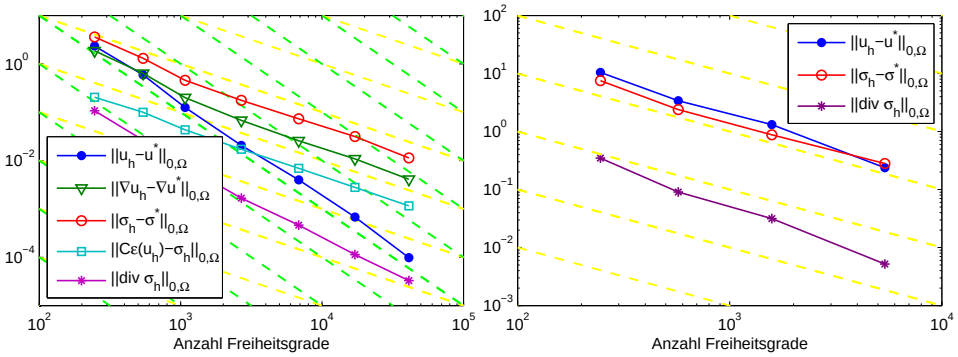
Wir wählen quadratische (RT_1) Raviart-Thomas-Elemente für die Spannung und quadratische konforme Elemente für die Verschiebung. Unsere Least-Squares-Formulierung mit quadratischen Elementen schafft damit (genügend Regularität der Lösung vorausgesetzt) eine Konvergenzordnung 2, durch die gemischt Formulierung auch für Spannung und Impuls:

$$\inf_{u_h, \sigma_h} \sqrt{F(u_h, \sigma_h)} = O(h^2)|\sigma|_3. \tag{40}$$

Wenn N die Anzahl der Freiheisgrade bezeichnet ist h bei einer gleichmäßgen Verfeinerung proportional zu N^{-d}. Man kann daher auch schreiben

$$\inf_{u_h, \sigma_h} \sqrt{F(u_h, \sigma_h)} = O(N^{-\frac{p}{d}})$$

Abbildung 2.2 auf der nächsten Seite (a) zeigt darüber hinaus, dass im elastischen Fall für die Verschiebungen numerisch sogar eine höhere Konvergenzordnung erzielt wird. In Abbildung 7.8 auf Seite 89 ist dargestellt, dass die tatsächlich erreichte Konvergenzordnung durch mangelnde Regularität geringer ist. Gleichmäßige Verfeinerung bei dem Beispielproblem aus dem nächsten Kapitel führt zu einer Konvergenzordnung von ungefähr 1,8 im elastischen Fall und zu 1,4 bei plastischer Verformung. Daher verwendet man adaptive Verfeinerungen. Abbildung 3.6 auf Seite 43 zeigt, dass bei adaptiver Verfeinerung die Reduktion $\eta^2 = O(N^{-2})$ im elastischen Fall erreicht wird, während mit zunehmender Plastizität die Konvergenzordnung geringer wird.



(a) elastischer Fall (b) plastischer Fall

ABBILDUNG 2.2. Fehlerreduktion (durch adaptive Verfeinerung) verschiedener Terme. Die gelben gestrichelten Linien zeigen die Steigung der Konvergenzordnung $\|u - u_h\|_{1,\Omega} = O(N^{-1})$ an. Die grünen gestrichelten Linien zeigen die Steigung der Konvergenzordnung $\|u - u_h\|_{0,\Omega} = O(N^{-2})$ im elastischen Fall an

Im Vergleich zur FEM wird bei gemischter LSFEM die Impulsbilanz mit höherer Genauigkeit erfüllt, dazu werden jedoch zusätzliche Freiheitsgrade für σ_h benötigt. Es hängt also vom Anwendungsfall ab, welche Methode vorzuziehen ist. Ist die rechte Seite der Impulsbilanz a-priori bekannt - z.B. bei div $\sigma = 0$, ist keine Approximation des Impulses nötig und die gemischte Least-Squares-FEM bringt keinen asymptotischen Approximationsvorteil.

2.3. Ebene Materialmodellierung

Hat man es in der Realität immer mit Gegenständen mit Abmessungen in drei Dimensionen zu tun, kann es dennoch ausreichen mit einem zweidimensionalen Modell zu arbeiten. Wird die Materialdicke einer ebenen Scheibe nämlich verschwindend klein im Vergleich zu den Ausdehnungen der Scheibe, so verschwindet auch der Einschnüreffekt des Materials und Spannungen treten dann hauptsächlich in Richtungen der Materialebene auf. Man spricht dann vom ebenen Spannungszustand in dem Berechnungen auf eine zweidimensionale Ebene reduziert werden können und a-priori Spannungen in z-Richtung ausgeschlossen sind:

$$\sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} & 0 \\ \sigma_{21} & \sigma_{22} & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Hat man es schließlich mit einer unendlich dünnen Scheibe zu tun, bleiben die Verschiebungen in dieser Ebene. In der Literatur wird dies ebener Verzerrungszustand genannt und es gilt:

$$u = \begin{pmatrix} u_1 \\ u_2 \\ 0 \end{pmatrix}, \epsilon = \begin{pmatrix} \epsilon_{11} & \epsilon_{12} & 0 \\ \epsilon_{21} & \epsilon_{22} & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Ebener Spannungs- und Verzerrungszustand schließen sich im Allgemeinen jedoch aus, da der effektive Materialtensor eine ebene Verzerrung nicht zwangsläufig in eine ebene Spannung überführen muss.

Das hookesche Gesetz für zweidimensionale isotrope Materialien kann rein zweidimensional formuliert werden:

$$\begin{aligned} \epsilon_{11} &= \frac{1}{E_2}(\sigma_{11} - \nu\sigma_{22}) \\ \epsilon_{22} &= \frac{1}{E_2}(\sigma_{22} - \nu\sigma_{11}) \\ \epsilon_{12} &= \frac{1}{2G_2}\sigma_{12} \end{aligned}$$

Wobei sich die 2-D Konstanten (mit Index 2) aber je nach zweidimensionalem Modell von 3-D Materialkonstanten (hier mit Index 3 geschrieben) unterscheiden (siehe z.B. [**38**, S. 54]):

Plane strain	Plane stress
$E_2 = \frac{E_3}{1-\nu_2^2}$	$E_2 = E_3$
$\nu_2 = \frac{\nu_3}{1-\nu_2}$	$\nu_2 = \nu_3$
$G_2 = \mu_2 = G_3$	$G_2 = \mu_2 = G_3$

Diese Arbeit beschränkt sich in der Implementierung auf den ebenen Verzerrungszustand. Obwohl Formulierungen mit ebenem Spannungs- oder Verzerrungszustand eigentlich nur auf rein zweidimensionale Probleme abzielen kommt es in klassischen Formulierungen durch volumetrische plastische Effekte dazu, dass σ_{33} nicht verschwindet. In der Arbeit [**17**] wurde eine reine zweidimensionale Formulierung für von Mises-Plastizität entwickelt, die sogar recht einfach zu implementieren ist. Allerdings ist für allgemeinere Materialmodelle der Aufwand ungleich höher. Wir benutzen daher einen anderen Ansatz:

2.4. Ein neuer gemischter zweidimensionaler Ansatz

Um Freiheitsgrade einzusparen ist es wünschenswert so wenig Freiheitsgrade wie möglich als primäre Variablen zu verwenden. Andererseits zeigt [**17**], dass eine rein zweidimensionale Formulierung sehr umständlich sein kann. Die wesentliche Idee ist es daher lokal dreidimensional zu rechnen, jedoch nur rein zweidimensional zu approximieren. Setzt man für die Verschiebung u nur zweidimensionale primäre Variablen

$$u = \begin{pmatrix} u_1 \\ u_2 \\ 0 \end{pmatrix}$$

an können trotzdem Spannungen σ_{33} auftreten. Man kann also in einer gemischten FEM den Spannungstensor mit folgenden Freiheitsgraden ansetzen:

$$\sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} & 0 \\ \sigma_{21} & \sigma_{22} & 0 \\ 0 & 0 & \sigma_{33} \end{pmatrix}$$

Außer der konstitutiven Beziehung taucht in den Gleichgewichtsbedingungen nur die Divergenz von σ auf. Es ist

$$\operatorname{div}(\sigma) = \begin{pmatrix} \operatorname{div}(\sigma_{11} & \sigma_{12} & 0) \\ \operatorname{div}(\sigma_{21} & \sigma_{22} & 0) \\ \operatorname{div}(0 & 0 & \sigma_{33}) \end{pmatrix} = \begin{pmatrix} \frac{\partial \sigma_{11}}{\partial x} + \frac{\partial \sigma_{12}}{\partial y} \\ \frac{\partial \sigma_{21}}{\partial x} + \frac{\partial \sigma_{22}}{\partial y} \\ 0 \end{pmatrix}.$$

Es taucht also keine Raumableitung von σ_{33} auf. Damit ist die einzige Bedingung an σ_{33} die konstitutiven Beziehung (4). Es ist also im Gleichgewicht

$$\sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} & 0 \\ \sigma_{21} & \sigma_{22} & 0 \\ 0 & 0 & (\sigma(\epsilon(u)))_{33} \end{pmatrix}.$$

Somit ist es nicht nötig Freiheitsgrade für σ_{33} zu investieren, sondern diese Komponente kann direkt aus dem Verschiebungsgradienten berechnet werden.

KAPITEL 3

Numerische Ergebnisse

In der Literatur finden sich viele verschiedene Verfahren um plastisches Verhalten zu simulieren. Auch innerhalb der FEM gibt es viele Unterschiede, die durch die Wahl der Gebietszerlegung, der verwendeten finiten Elemente, der benutzten Quadraturformeln, dem Material-Modell usw. entstehen.

Um verschiedene Verfahren zur Lösung eines Problems miteinander vergleichen zu können ist es notwendig den tatsächlichen Fehler jedes einzelnen Verfahrens numerisch zu messen, wenn die analytischen Fehlerabschätzungen unscharf sind. In [**45**, S. 385] wird deshalb vorgeschlagen ein Modell vorzugeben und die berechneten Ergebnisse miteinander anhand von einzelnen Funktionswerten zu vergleichen. Der dortige Benchmarktest 1c wird mit der hier vorgestellten Methode verglichen. Es ist ein Anfangswertproblem mit Prandtl-Reuss-Plastizität über einer $100\text{mm} \times 100\text{mm}$ großen Lochplatte mit einem zentralen Loch von 10mm Radius (Abbildung 3.1(a)). Angenommen wird dabei ein ebener Verzerrungszustand. Als Materialparameter werden ein Elastizitätsmodul $E = 206900,00\ \text{kN/mm}^2$, Querkontraktion $\nu = 0,29$ und die Grenzspannung $\sigma_0 = 450,00\ \text{N/mm}^2$ vorgegeben. Auf die obere und untere Plattenseite wirkt eine Rand-Normalen-Kraft von $100\ \text{N/mm}^2$ mal einem zeitabhängigem Lastfaktor. Der Lastfaktor durchläuft einen Zyklus $0 \ldots 4,5 \ldots -4,5 \ldots 0$. Die Geschwindigkeit dabei ist irrelevant, da es sich um ein ratenunabhängiges Modell handelt.

Das Modell setzt die innere Kraft $f = 0$ und die Ausgangsparameter $u_0 = 0$, $q_0 = 0$ und $\epsilon_0^p = 0$ voraus.

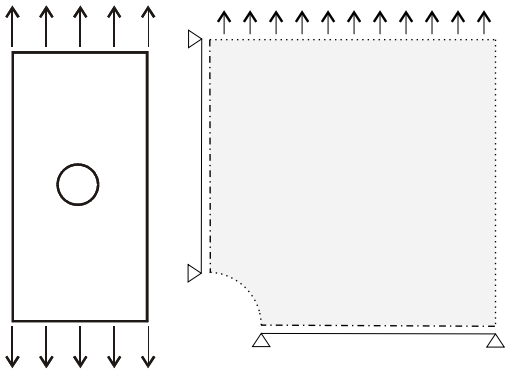
Die Materialparameter lassen sich umrechnen zu den Lamé-Konstanten [**11**, S. 251]

$$\begin{aligned} \mu &= \frac{E}{2(1+\nu)} \approx 80194\ \text{kN/mm}^2 \\ \lambda &= \frac{2\mu\nu}{1-2\nu} \approx 110744\ \text{kN/mm}^2 \end{aligned}$$

Der Dirichlet-Rand ist leer, die Lösung also nicht eindeutig. Aus Symmetriegründen reicht es aber nur ein Scheibenviertel (Abbildung 3.1(b)) zu betrachten. Werden dort an den Symmetrieachsen die Dirichlet-Bedingungen auf den Randnormalen gefordert, erhält das Problem eine eindeutige Lösung.

Die Ergebnisse unserer Berechnungen sind in den Abbildungen 3.2 auf Seite 39 und 3.3 auf Seite 40 dargestellt.

Verglichen werden die Werte an bestimmten Punkten im Material:

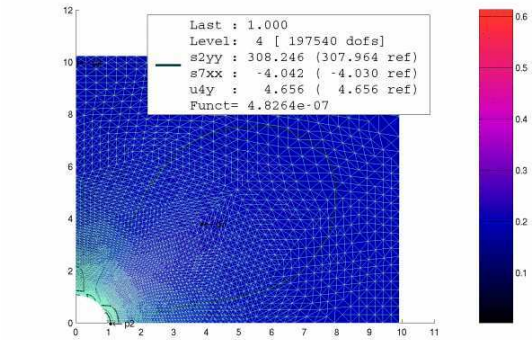


(a) Lochplatte (b) Lochscheibenviertel

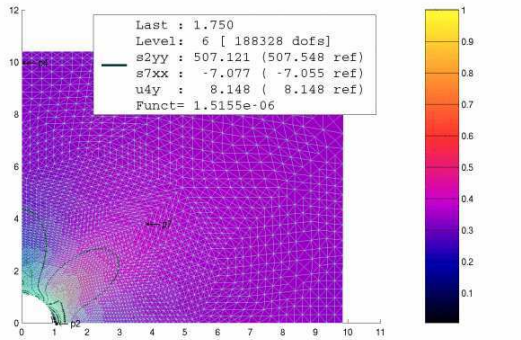
ABBILDUNG 3.1. Das Benchmarkproblem

- Punkt 2 liegt direkt am Loch, dort treten die größten Spannungen auf.
- Punkt 4 liegt in der Mitte der Kante, an der Kraft aufgetragen wird.
- Punkt 5 liegt an der Ecke gegenüber dem Loch
- Punkt 7 liegt auf der Diagonalen zwischen Loch und der gegenüberliegenden Ecke, auf einem Drittel der Strecke.

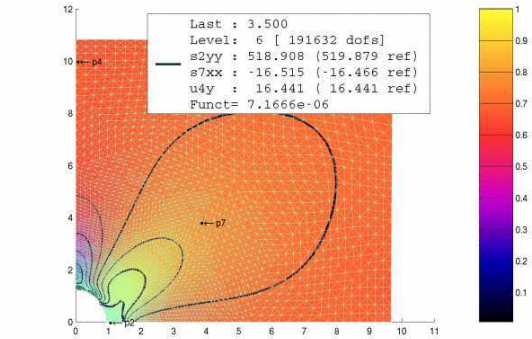
Die Referenzwerte in [**45**] wurden mit ca. 200 000 Freiheitsgraden von Wieners und Wittnum berechnet. In unseren Tests verwendeten wir ebensoviele Freiheitsgrade. Unsere Methode liefert für das Benchmarkproblem die Werte in Tabelle 1 auf Seite 41. Ein grafischer Vergleich findet sich in den Abbildungen 3.4 auf Seite 42 und 3.5 auf Seite 43. Die Verschiebungswerte am unkritischen Punkt 5 stimmen weitestgehend überein, während sich am Punkt 2 - dort wo adaptiv am meisten verfeinert wird - sich mit der Zeit stärkere Abweichungen für die Verschiebungen akkumulieren. Derartige Abweichungen sind auch bei anderen Arbeitsgruppen aufgetreten ([**45**, S. 395, Abbildung 11.14 (a)]), allerdings in entgegengesetzter Richtung. Da es keine analytische Referenzlösung gibt ist zunächst unklar, welche Werte besser sind. Die erfolgreiche Reduktion des Fehlerschätzers (Abbildung 3.6 auf Seite 43) zeigt auf jeden Fall, dass unsere Ergebnisse für jeden einzelnen Zeitschritt nahe an der exakten Lösung sind.



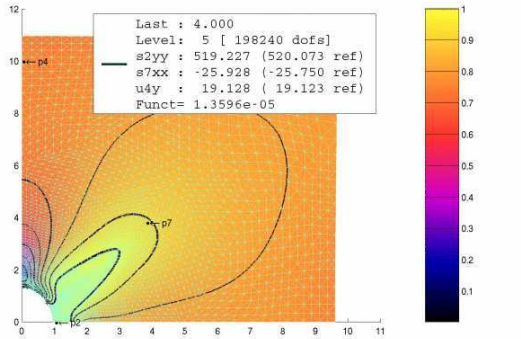
(a) rein elastischer Zustand



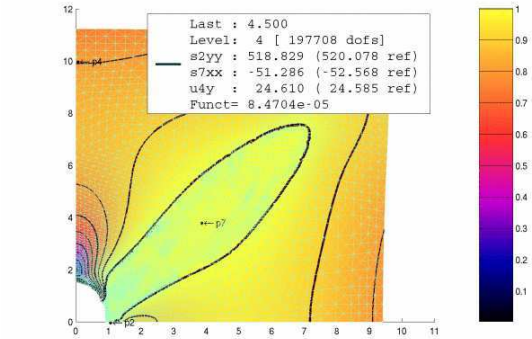
(b) Einsetzen der plastischen Verformung am Punkt 2



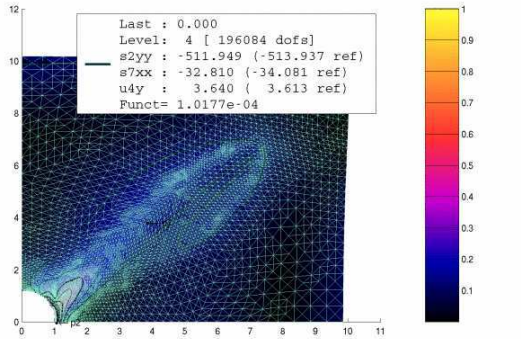
(c) kleine plastische Zone um den Punkt 2



(d) immer schnelleres Anwachsen der plastischen Zone

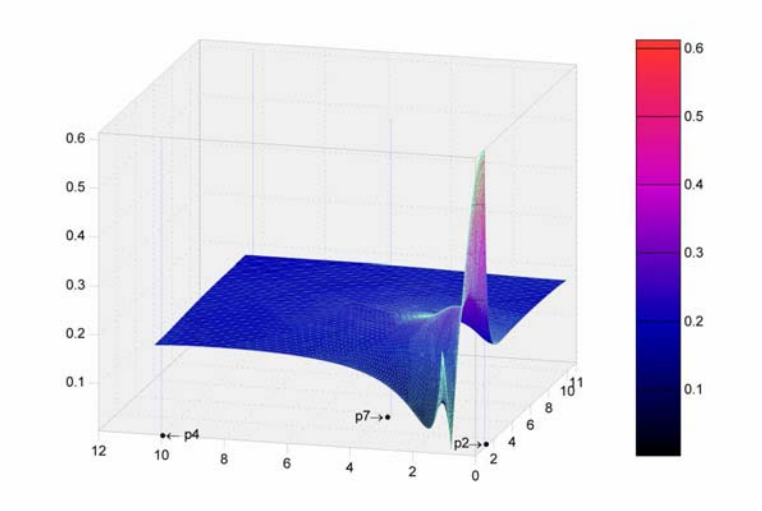


(e) groß ausgedehnte plassiche Zone: Sie umfasst jetzt auch Punkt 7

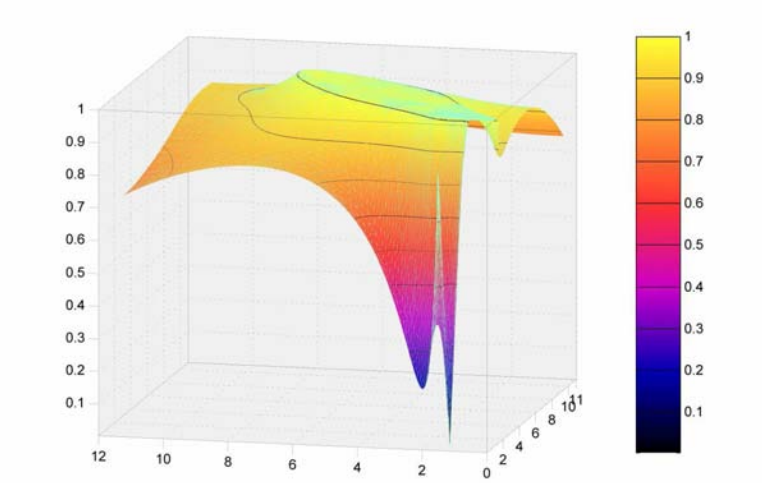


(f) Nach Wegfall der äußeren Last: Durch die plastischen Verformungen bleiben Restspannungen im Material. Die adaptive Verfeinerung lässt das plastisch verformte Gebiet weiter erahnen

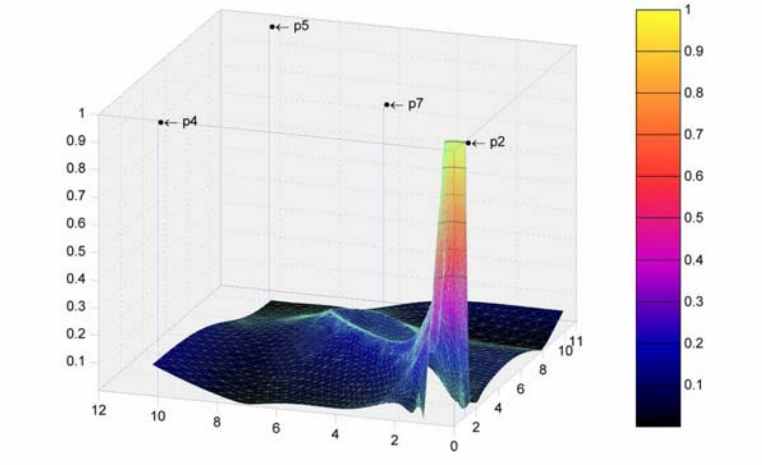
ABBILDUNG 3.2. Norm der deviatorischen Spannung in Abhängigkeit vom Ort zu sechs unterschiedlichen Zeitpunkten. Die Skala ist so normiert, dass beim Wert 1 das plastische Fließen stattfindet. Die Verformungen des Materials sind um den Faktor 500 überzeichnet dargestellt. Die benutzte Triangulierung ist in hellblau eingezeichnet.



(a) im elastischen Fall



(b) im plastischen Fall: deutlich zu erkennen ist die geringe Regularität der Lösung durch den Knick am Rand der plastischen Zone



(c) nach Wegfall der äußern Last

ABBILDUNG 3.3. 3D-Plots der deviatorischen Spannungsnorm

TABELLE 1. Approximierte Funktionswerte an einzelnen Punkten bei ca. 200 000 Freiheitsgraden im Vergleich zu den Referenzwerten (in den Klammern)

Lastfaktor	$\sigma_{yy}(p2)$	$\sigma_{xx}(p7)$	$u_y(p4)$	$u_x(p2)$	$u_x(p5)$
1.000	308.246(307.964)	-4.042(-4.030)	4.656(4.656)	0.473(0.473)	1.706(1.706)
1.750	507.121(507.548)	-7.077(-7.055)	8.148(8.148)	0.828(0.828)	2.985(2.985)
2.000	511.424(511.882)	-8.130(-8.105)	9.314(9.314)	0.938(0.938)	3.410(3.410)
2.250	514.407(515.076)	-9.262(-9.231)	10.485(10.485)	1.028(1.028)	3.834(3.834)
2.500	516.666(517.169)	-10.483(-10.447)	11.661(11.661)	1.093(1.093)	4.254(4.254)
2.750	517.585(518.443)	-11.807(-11.766)	12.844(12.844)	1.125(1.125)	4.672(4.672)
3.000	518.263(519.184)	-13.239(-13.199)	14.034(14.034)	1.120(1.120)	5.087(5.087)
3.250	518.652(519.627)	-14.797(-14.755)	15.232(15.320)	1.075(1.076)	5.497(5.497)
3.500	518.908(519.879)	-16.515(-16.466)	16.441(16.441)	0.982(0.985)	5.903(5.903)
4.000	519.227(520.073)	-25.928(-25.750)	19.128(19.123)	0.968(0.999)	6.600(6.602)
4.125	519.395(520.078)	-33.120(-32.869)	19.954(19.948)	1.030(1.069)	6.711(6.713)
4.250	519.527(520.080)	-51.938(-51.517)	20.952(20.944)	1.146(1.198)	6.752(6.756)
4.375	519.228(520.080)	-54.087(-54.243)	22.324(22.310)	1.358(1.441)	6.645(6.650)
4.500	518.829(520.078)	-51.286(-52.568)	24.610(24.585)	1.736(1.907)	6.170(6.180)
4.250	442.910(443.087)	-50.303(-51.560)	23.448(23.421)	1.619(1.789)	5.743(5.754)
4.000	365.691(366.096)	-49.289(-50.553)	22.284(22.257)	1.501(1.670)	5.316(5.327)
3.750	288.368(289.104)	-48.286(-49.545)	21.120(21.093)	1.382(1.552)	4.890(4.901)
3.500	211.722(212.112)	-47.277(-48.538)	19.956(19.929)	1.264(1.434)	4.463(4.474)
3.250	134.997(135.121)	-46.275(-47.530)	18.792(18.765)	1.146(1.316)	4.037(4.048)
3.000	57.949(58.129)	-45.272(-46.523)	17.628(17.601)	1.027(1.197)	3.610(3.622)
2.750	-19.105(-18.863)	-44.278(-45.515)	16.464(16.437)	0.909(1.079)	3.184(3.195)
2.500	-96.804(-95.855)	-43.260(-44.508)	15.300(15.273)	0.791(0.961)	2.757(2.769)
2.250	-173.647(-172.842)	-42.260(-43.500)	14.136(14.109)	0.673(0.843)	2.331(2.342)
2.000	-250.549(-249.829)	-41.236(-42.493)	12.972(12.945)	0.554(0.724)	1.905(1.916)
1.750	-327.445(-326.825)	-40.225(-41.485)	11.808(11.781)	0.436(0.606)	1.478(1.489)
1.500	-405.141(-403.816)	-39.224(-40.478)	10.644(10.617)	0.318(0.488)	1.052(1.063)
1.250	-469.853(-466.134)	-38.218(-39.470)	9.480(9.453)	0.199(0.369)	0.625(0.637)
1.000	-479.824(-480.904)	-37.187(-38.452)	8.316(8.289)	0.082(0.252)	0.199(0.210)
0.750	-492.148(-493.388)	-36.144(-37.410)	7.150(7.123)	-0.032(0.138)	-0.226(-0.215)
0.500	-501.864(-502.753)	-35.071(-36.338)	5.982(5.955)	-0.140(0.030)	-0.651(-0.640)
0.250	-507.600(-509.503)	-33.961(-35.229)	4.812(4.785)	-0.239(-0.070)	-1.075(-1.064)
0.000	-511.949(-513.937)	-32.810(-34.081)	3.640(3.613)	-0.329(-0.161)	-1.498(-1.486)
-0.250	-514.696(-516.773)	-31.600(-32.889)	2.465(2.438)	-0.407(-0.240)	-1.919(-1.908)
-1.000	-517.794(-519.531)	-27.718(-29.012)	-1.079(-1.105)	-0.543(-0.387)	-3.174(-3.163)
-1.250	-518.913(-519.786)	-26.318(-27.611)	-2.267(-2.293)	-0.561(-0.403)	-3.590(-3.579)
-1.500	-519.077(-519.925)	-24.862(-26.153)	-3.458(-3.485)	-0.561(-0.401)	-4.004(-3.993)
-2.000	-518.660(-520.039)	-21.767(-23.047)	-5.854(-5.881)	-0.494(-0.339)	-4.825(-4.814)
-3.000	-518.896(-520.078)	-12.086(-13.407)	-10.824(-10.850)	-0.310(-0.211)	-6.389(-6.379)
-4.000	-518.558(-520.081)	52.445(49.090)	-17.292(-17.267)	-0.761(-0.870)	-7.336(-7.346)
-4.500	-518.695(-520.078)	50.869(54.813)	-24.629(-24.444)	-1.904(-2.381)	-6.162(-6.238)
-4.000	-365.096(-366.095)	48.901(52.798)	-22.301(-22.116)	-1.668(-2.145)	-5.309(-5.385)
-3.000	-57.408(-58.131)	44.837(48.768)	-17.645(-17.460)	-1.195(-1.672)	-3.603(-3.679)
-2.000	250.442(249.834)	40.822(44.738)	-12.989(-12.804)	-0.722(-1.198)	-1.898(-1.973)
-1.000	480.572(480.904)	36.714(40.698)	-8.333(-8.148)	-0.250(-0.726)	-0.192(-0.268)
0.000	508.893(513.935)	32.326(36.327)	-3.657(-3.472)	0.156(-0.314)	1.504(1.429)

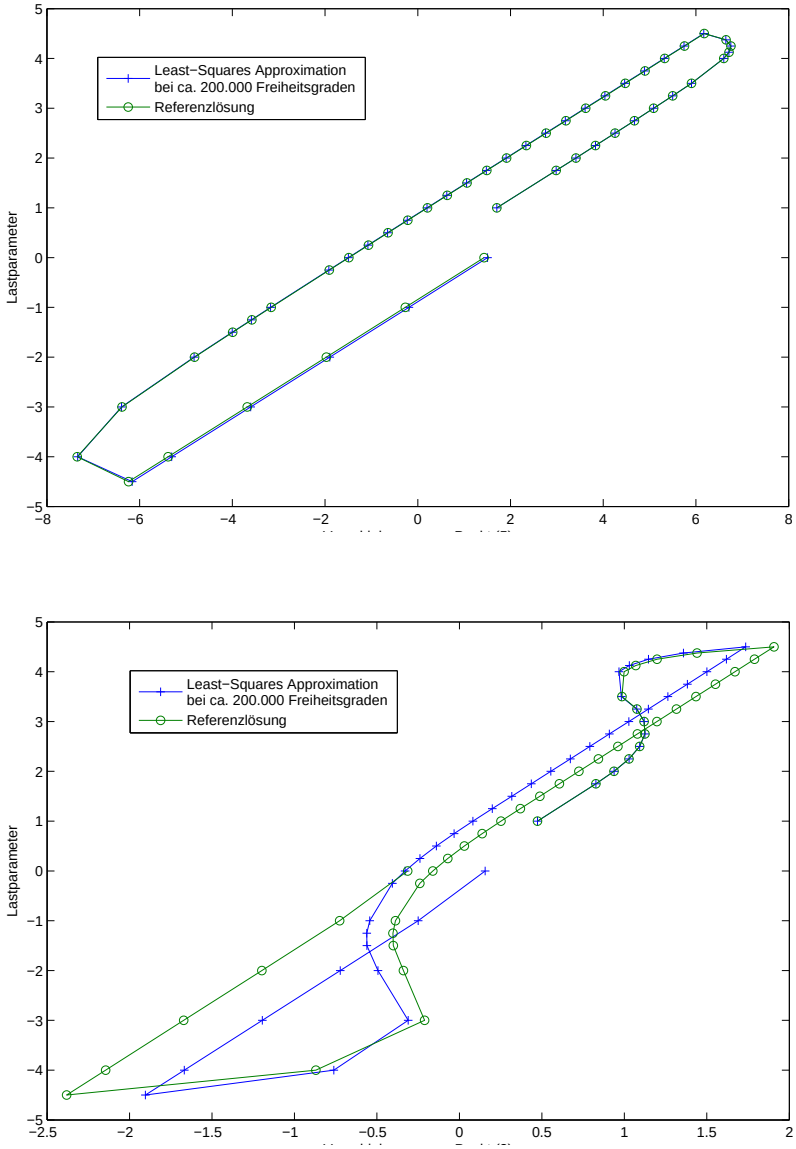


ABBILDUNG 3.4. Approximierte Verschiebungswerte an einzelnen Punkten im Vergleich zur Referenzlösung.

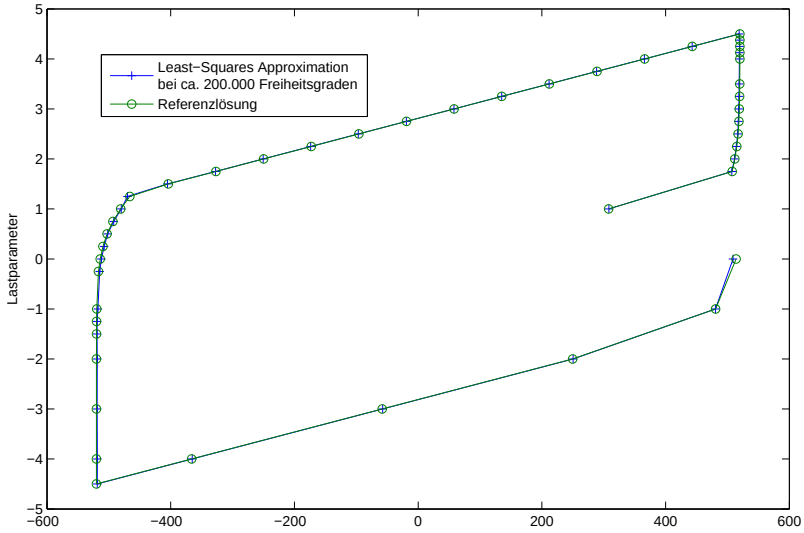


ABBILDUNG 3.5. Approximierte Spannungen an einem einzelnen Punkt im Vergleich zur Referenzlösung. Die Spannungswerte am kritischen Punkt 2 stimmen weitestgehend mit der Referenzlösung überein.

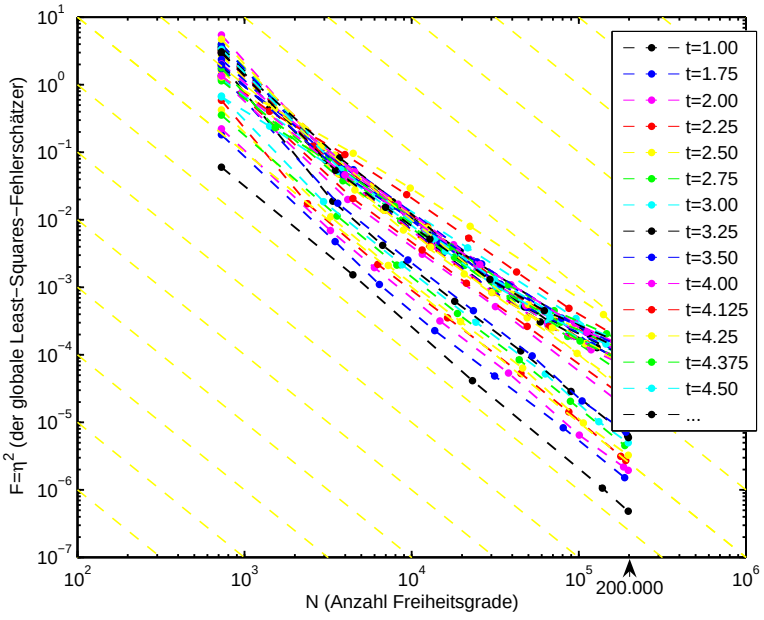


ABBILDUNG 3.6. Reduktion des Fehlers zur verschiedenen Zeitschritten bei zunehmender Anzahl von Freiheitsgraden (durch adaptive Verfeinerung). Die gelben gestrichelten Linien zeigen die Steigung der maximal erwartbaren Konvergenzordnung $\eta^2 = O(N^{-2})$ an.

KAPITEL 4

Die Minimierung des Funktionals

Um die Optimierungsaufgabe

$$\min_{x\in\mathbb{R}^n} \|R(x)\|^2$$

mit den Freiheitsgraden $x = (u_h, \sigma_h)$ zu lösen verwenden wir die Gauss-Newton-Iteration. Diese basiert darauf, in jedem Iterationsschritt i das Residuum R mit Hilfe der Jacobi-Matrix J_i zu linearisieren und dann ein lineares Ausgleichsproblem zu lösen:

$$\min_{p\in\mathbb{R}^n} \|R(x_i) + J_i p\|^2 .$$

Solange $J^T J$ nicht singulär ist, kann man aus der notwendigen Bedingung erster Ordnung

$$\begin{aligned} 0 &= \nabla \left\| \tilde{R}(p) \right\|^2 . \\ &= \nabla \|R(x_i) + J_i p\|^2 \\ &= \nabla (R(x_i) + J_i p)^T (R(x_i) + J_i p) \\ &= \nabla (R(x_i)^T R(x_i) + (J_i p)^T J_i p + 2 J_i^T R(x_i) p) \\ &= 2 J_i^T J_i p + 2 J_i^T R(x_i) \end{aligned}$$

auch die Normalengleichung

$$p = -(J_i^T J_i)^{-1} J_i^T R(x_i)$$

schlussfolgern. So erhält man die Gauss-Newton-Iteration:

$$x_{i+1} = x_i - (J(x)_i^T J(x_i))^{-1} J(x_i)^T R(x_i) \text{ (Gauss-Newton-Iteration).} \tag{41}$$

Allgemeiner lässt sie sich mit Hilfe der verallgemeinerten Moore-Penrose-Inversen schreiben als:

$$x_{i+1} = x_i - J_i^+ R(x_i). \tag{42}$$

In den Abbildungen 4.1 auf der nächsten Seite und 4.3 auf Seite 47 ist das bei uns beobachtete Konvergenzverhalten dokumentiert: Man erhält, wenn das Funktional im Minimum glatt ist, lineare Konvergenz mit einer Konvergenzgeschwindigkeit unabhängig von der Anzahl der Unbekannten. Dabei fällt auf, dass bereits nach wenigen Schritten der Wert des Least-Squares-Funktionals kaum noch sinkt.

Um bei feinen Triangulationen Rechenaufwand zu sparen verwenden wir einen einfachen Multilevelalgorithmus: Angefangen mit einer groben Triangulierung wird

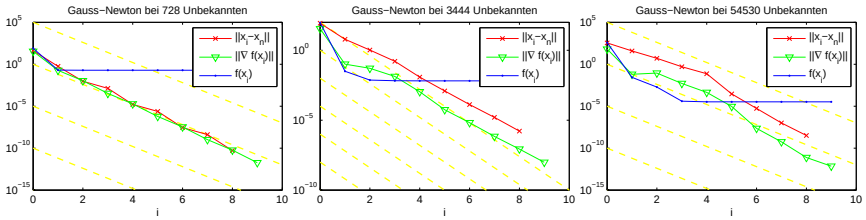


ABBILDUNG 4.1. Konvergenz des Gauss-Newton-Verfahrens wenn das Funktional im Minimum glatt ist. Zum Vergleich sind in allen Abbildungen gelb gestrichelten Linien mit gleicher Steigung eingetragen.

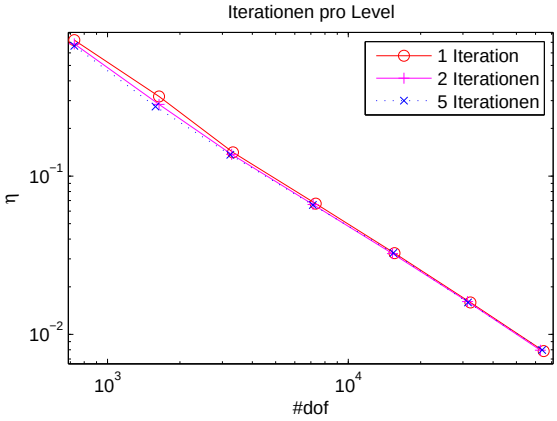


ABBILDUNG 4.2. Verwendung eines Multilevelalgorithmus bei perfekter Plastizität.

eine Approximation mit konstanter Anzahl von Gauss-Newton-Schritten berechnet. Danach wird die Triangulierung adaptiv verfeinert (Anzahl der Freiheitsgrade in etwa verdoppelt) und die Werte des gröberen Gitters auf das feinere prolongiert und dort das Gleichungssystem wieder mit derselben Anzahl an Gauss-Newton-Schritten iteriert usw.

Das beeindruckende Ergebnis dieses einfachen /-Zyklus ist in Abbildung 4.2 dokumentiert: Dargestellt ist das Verhalten des Fehlers bei konstanter Anzahl von Gauss-Newton-Iterationen pro Triangulation. Man erkennt, dass bereits nach zwei Iterationen pro Level sich der Approximationsfehler nicht mehr wesentlich verbessern lässt.

4.1. Knicke im Least-Squares-Funktional

Während in den meisten Beispielrechnungen ein lineares Konvergenzverhalten wie in Abbildung 4.1 beobachtet wurden, zeigt sich manchmal (genau dann, wenn in mindestens einem Quadraturpunkt die Lösung direkt auf dem Übergang von elastischem zu plastischem Materialverhalten liegt) auch wie in Abbildung 4.3 auf der nächsten Seite links zu sehen keine Konvergenz des Gauss-Newton-Verfahrens. Wir untersuchen dieses Verhalten näher:

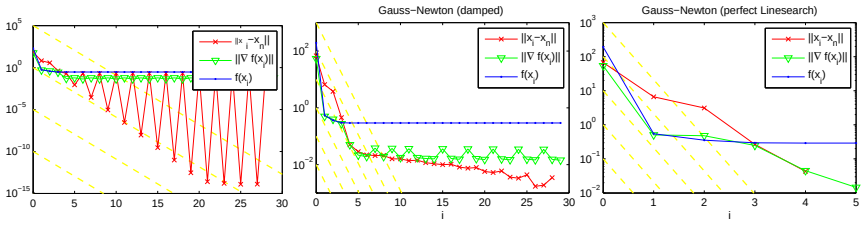


ABBILDUNG 4.3. Konvergenz von (gedämpften) Gauss-Newton-Verfahren wenn das Funktional einen Knick im Minimum hat

Benutzt man eine Dämpfung der allgemeinen Iteration $x_{i+1} = x_i + p_i$ mit einem skalaren Dämpfungsparameter α_i

$$x_{i+1} = x_i + \alpha_i p_i$$

der mit perfekter Liniensuche

$$\min_{\alpha_i \in \mathbb{R}} f(x_i + \alpha_i p_i) \text{ (perfekte Liniensuche)}$$

gefunden wird, kann man in unseren Beispielen ein konvergentes Verhalten beobachten (Abbildung 4.3 rechts). Es kann jedoch nicht gezeigt werden, dass die Iterierten gegen die gesuchte Minimalstelle konvergieren, da $\|\nabla F\|$ nach unten beschränkt bleibt. Dadurch zeigt sich, dass die Iterierten gegen einen Knick im Funktional konvergieren.

Woher kommen die Knicke? Wir zeigen im folgenden kurz an einem Beispiel, dass die Differenzierbarkeit des LS-Funktionals von den Ansatzfunktionen abhängt: Dazu betrachten wir ein sehr vereinfachtes Funktional

$$F_{\text{Bsp}}(u) := \int_{\Omega_{\text{Bsp}}} \|R_{\text{Bsp}}(u,\ \mathsf{x})\|^2 \, d\mathsf{x}, \text{ mit } R_{\text{Bsp}} := \max(0, \gamma_{\text{Bsp}}) + 1$$

mit $\gamma_{\text{Bsp}}(\epsilon(u(\alpha))) := \epsilon(u(\alpha)) - 1$, dem finiten Ansatz $u(\alpha) := \alpha \cdot \phi_{\text{Bsp}}(x)$ und der Basisfunktion $\phi_{\text{Bsp}}(x) := \sin(x)$ für alle $\mathsf{x} \in \Omega_{\text{Bsp}} := [-\frac{\pi}{2}, +\frac{\pi}{2}]$. Dann ist

$$F_{\text{Bsp}}(\alpha) = \begin{cases} 1 & \text{für } \alpha \leq 1 \\ 1 + 2(\alpha I(b(\alpha)) - I(0)) - b(\alpha) & \text{für } \alpha \geq 1 \end{cases}$$

mit den Hilfs-Integrationsgrenzen

$$b(\alpha) = \arccos \frac{1}{\alpha}$$

und der Stammfunktion von $\cos(x)^2$

$$I(\mathsf{x}) = \frac{1}{2}\cos(\mathsf{x})\sin(\mathsf{x}) + \frac{1}{2}\mathsf{x}.$$

Differenzierung ergibt dann

$$F'_{\text{Bsp}}(\alpha) = \begin{cases} 0 & \text{für } \alpha \leq 1 \\ \frac{1}{\alpha^2(\alpha^2-1)^{\frac{1}{2}}}(\arccos(\frac{1}{\alpha})\alpha^2(\alpha^2-1)^{\frac{1}{2}} + 1 + \alpha^2 - 2\alpha) & \text{für } \alpha \geq 1 \end{cases}$$

Nun folgt mit $\lim_{\alpha\to 1}\arccos(\frac{1}{\alpha})=0$, dass $\lim_{x\to 1}F'_{\mathrm{Bsp}}(\alpha)=0$. Daher ist die Ableitung stetig - das Funktional hat keinen Knick. Man könnte also auf die Idee kommen statt polynomialen finiten Elementen trigonometrische Ansatzfunktionen zu verwenden. Wendet man aber dann numerische Quadratur

$$\tilde{F}_{Bsp}(u)=\sum_i w_i\,\|R_{Bsp}(u,\,\mathsf{x}_i)\|^2\approx\int_{\Omega_{\mathrm{Bsp}}}\|R_{\mathrm{Bsp}}(u,\,\mathsf{x})\|^2\,d\mathsf{x}$$

an, wird die nicht nichtglatte max-Funktion mit nicht verschwindenden Gewicht w_i gewichtet und man erhält eine nichtglatte Funktion $\tilde{F}_{Bsp}$. Spätestens durch numerische Quadratur erhält also das LS-Funktional einen Knick, selbst wenn man die Ansatzfunktionen so geschickt wählen würde, dass bei exakter Integration kein Knick im Funktional existiert.

Da wir für eine effiziente Implementierung um numerische Quadratur nicht herumkommen interessieren wir uns nun dafür, wann Knicke in einem konvexen Funktional die lokale Konvergenz von (Gauss-)Newton-Verfahren beeinflusst.

Dazu betrachten wir die allgemeine Optimierungsaufgabe

$$\min_{x\in\mathbb{R}^n} f(x),$$

wobei $f(x)$ nicht überall stetig differenzierbar seien soll, aber immerhin fast überall. Wir nehmen aber an, dass f streng konvex ist, so dass eine eindeutige Lösung x^* existiert. Wir zeigen nun durch ein Beispiel, dass wenn das Minimum an einem nichtdifferenzierbaren Punkt von f liegt, im Allgemeinen keine lokale Konvergenz zur Minimalstelle durch (Gauss-)Newton-Verfahren erreicht wird, selbst wenn gedämpft wird. Es sei:

$$f(x_1,x_2):=\begin{cases}f_1(x_1,x_2) & \text{wenn } x_1>0,\ x_2\geq 0\\ f_2(x_1,x_2) & \text{wenn } x_1\leq 0,\ x_2\geq 0\\ f_3(x_1,x_2) & \text{wenn } x_1\leq 0,\ x_2<0\\ f_4(x_1,x_2) & \text{wenn } x_1>0,\ x_2<0\end{cases},$$

$$f_1(x):=\frac{1}{2}(x-y)^TJ(x-y)+1,$$

$$\begin{aligned}f_2(x_1,x_2) &:= f_1(-x_1,x_2)\ \text{(Achsenspiegelung)},\\ f_3(x_1,x_2) &:= f_1(-x_1,-x_2)\ \text{(Punktspiegelung)},\\ f_4(x_1,x_2) &:= f_1(x_1,-x_2)\ \text{(Achsenspiegelung)},\end{aligned}$$

$$J=\begin{pmatrix}5 & 3\\ 3 & 5\end{pmatrix}\ \text{und}\ y=\begin{pmatrix}0\\ -1\end{pmatrix}.$$

Die gesuchte Lösung liegt bei $x^*=0$. Es ist $f=\max(f_1,f_2,f_3,f_4)$. Daraus folgt sofort, dass f stetig ist. Ausserdem ist f streng konvex (was man auch schön an den konvexen Höhenlinen in Abbildung 4.4 sieht) und jedes f_i hat darüber hinaus

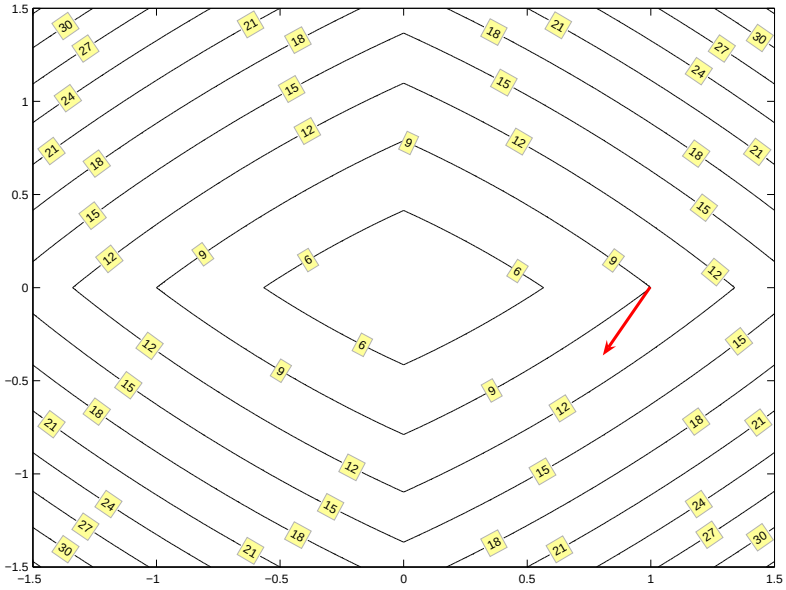


ABBILDUNG 4.4. Beispiel dafür, dass Konvexität nicht für die Konvergenz glatter Optimierungsverfahren ausreicht.

dieselbe positive definite Jacobi-Matrix. Also auf den ersten Blick eine Funktion mit tollen Eigenschaften - bis auf die Knicke bei $x_1 = 0$ und $x_2 = 0$. Aber mit einem Startvektor $x_0 = (1,\ 0)^T$ passiert folgendes: Es ist $f(x_0) = f_1(x_0)$ nach Definition von f. Der negative Gradient von f_1 ist dort aber keine Abstiegsrichtung von f: Setzt man die Richtung

$$\begin{aligned} p := -\nabla f_1(x_0) &= -\nabla(\frac{1}{2}(x_0 - y)^T J(x_0 - y)) \\ &= -\nabla(\frac{1}{2}x_0^T J x_0 - x_0^T J y) = -Jx_0 + Jy = -J(x_0 - y) \\ &= -(3,\ 5)^T \end{aligned}$$

mit Dämpfungsparameter α in f ein ergibt sich:

$$\begin{aligned} f(x_0 + \alpha p) &= f_4(x_0 + \alpha p)\ \forall \alpha > 0 \\ &= \frac{1}{2}(x - y)^T J(x - y) + 1 \\ &= 9 + 16\alpha + 40\alpha^2 \\ \Rightarrow \nabla f_4(x_0 + \alpha p) &= 80\alpha + 16 > 0\ \forall \alpha > 0. \end{aligned}$$

Weiter gilt dasselbe für jeden Startvektor $(\lambda, 0)^T$, also insbesondere mit $\lambda \to 0$ für einen Punkt beliebig nahe an der Lösung. Also konvergieren bei nichtglatten Optimierungsproblemen selbst gedämpfte Gradientenverfahren nicht unbedingt gegen die Minimalstelle - nicht einmal lokal. Ebenso zeigt die Newton-Richtung immer auf ein Minimum der f_i, also selbst an der Lösung x^* von der Lösung weg - hin zum Punkt y oder seiner Spiegelung. Dasselbe gilt für alle Verfahren höherer Ordnung, da alle höheren Ableitungen wegen der konstanten Jacobi-Matrix verschwinden.

Um das Gauss-Newton-Verfahren verlässlich zur Konvergenz zu bringen, konstruieren wir im folgenden ein ε-regularisierters (geglättetes) Funktional F_ε. Regularisierung für Elastoplastizität wird gefordert z.B. in [**35**] für die numerische Konvergenz gemischter (dualer) Methoden, in [**47**] bei perfekter Plastizität für die Existenz einer Lösung in Ω_h und losgelöst vom Anwendungsfall in [**20**] für globale Konvergenz des Newton-Verfahrens (während [**6**] nur lokale Konvergenz zeigt). Außerdem wird Regularisierung in [**53**, 3.4.2] gefordert um die Existenz und Eindeutigkeit einer Lösung des Anfangswertproblems zu beweisen. Mit $\varepsilon \to 0$ soll dabei jeweils $\|F_\epsilon - F\|$ verschwinden.

Doch wie groß ist ein ε zu wählen, bzw. wie klein muss es werden? Han & Reddy ([**35**]) verwenden einen a-posteriori Fehlerschätzer um $\|u_{h,\varepsilon} - u_h\|$ abzuschätzen. Repin ([**47**]) stellt fest, dass es bei Regularisierung durch Verfestigung ein ε geben muss, für dass $\|u_{h,\varepsilon} - u\|$ minimal werden muss und dass dies im Allgemeinen für $\varepsilon \neq 0$ der Fall sein wird. Er schlägt vor man solle das ε aus numerischen Experimenten gewinnen.

4.2. Eine neue - dem Problem angepasste Regularisierung

Wir verwenden eine Regularisierung des Least-Squares-Funktionals und zeigen wie klein ε_h (abhängig von h) gewählt werden muss, so dass der Approximationsfehler hinreichend klein ist. Dabei wird in der Analyse eine uniforme (keine adaptive) Verfeinerung vorausgesetzt.

Wir wiederholen das Optimierungsproblem für perfekte Prandtl-Reuss-Plastizität (22 auf Seite 24):

$$\min_{u,\sigma} \int_\Omega \|\text{div } \sigma\|^2 + \|P_R \mathcal{C}\epsilon(u) - \sigma\|^2$$

Durch numerische Integration wird dies zu

$$\min_{u,\sigma} \sum_i w_i (\|\text{div } \sigma(\mathsf{x}_i)\|^2 + \|P_R \mathcal{C}\epsilon(u(\mathsf{x}_i)) - \sigma(\mathsf{x}_i)\|^2)$$

Das Optimierungsproblem kann man auch schreiben als:

$$\min F,\ F = \sum_i^n w_i \|R_i\|^2 = \|DR\|^2$$

mit einer Diagonalmatrix D mit den Wurzeln der Quadratur-Gewichte und

$$R_i = \begin{bmatrix} \mathcal{C}\epsilon(u) - 2\mu n\gamma \\ \text{div } \sigma \end{bmatrix},$$

Der einzige nichtlineare (nicht differenzierbare) Term des Residuums ist hierbei

$$P_R \mathcal{C}\epsilon(u) := \mathcal{C}\epsilon(u) - 2\mu n\gamma$$

Bei perfekter Prandtl-Reuss-Plastizität ist die Fließrichtung, gegeben durch $n = \frac{\text{dev } C\epsilon(u)}{\|\text{dev } C\epsilon(u)\|}$, eine nichtlineare aber glatte Funktion mit $\|n\| = 1$. Der nicht differenzierbare Teil steckt allein im Konsistenzparameter

$$\gamma = \max(0, \gamma^*).$$

Wir verwenden stattdessen einen geglätteten Konsistenzparameter

$$\tilde{\gamma} = \text{softmax}(0, \gamma^*)$$

mit

$$\text{softmax}(0,x) = \begin{cases} 0 & \text{für } x < \varepsilon \\ x & \text{für } x > \varepsilon \\ \frac{1}{4\epsilon}(x+\epsilon)^2 & \text{sonst} \end{cases}$$

Für die softmax Funktion gilt:

$$\|\max - \text{softmax}\| \leq \frac{1}{2}\varepsilon,$$

$$\left\| \frac{\partial \,\text{softmax}(0,x)}{\partial^2 x} \right\| \leq \frac{1}{2}\varepsilon^{-1}$$

Man erhält somit ein verändertes Residuum $\tilde{R}$ mit

$$\left\| \tilde{R}_i - R_i \right\| = \left\| \begin{bmatrix} 2\mu n\tilde{\gamma} - 2\mu n\gamma \\ 0 \end{bmatrix} \right\| \leq 2\mu\frac{1}{2}\varepsilon = O(\varepsilon)$$

Für eine reguläre Familie von uniformen Verfeinerungen $\mathcal{T}_h$ ist die Anzahl der Quadraturpunkte (für eine d-dimensionale Formulierung) $n = O(\frac{1}{h^d})$. Die dazugehörenden Gewichte sind $w_i = O(h^d)$. Die Glättung verändert allerdings nur Werte am Rand der plastischen Zone (die nur $d-1$ dimensional ist), so dass effektiv nur für $\tilde{n} = O(\frac{1}{h^{d-1}})$ Summanden eine Abweichung entsteht. Dadurch erhält man

$$\begin{aligned} \left\| F - \hat{F} \right\| &\leq \sum_i^n w_i \left\| (R_i - \tilde{R}_i) \right\|^2 = O(\frac{1}{h^{d-1}}) \cdot O(h^d) \cdot \max_i \left\| (R_i - \tilde{R}_i) \right\|^2 \\ &\leq O(h^1) O(\varepsilon)^2 \\ &= O(\varepsilon^2 h) \end{aligned}$$

Wir erinnern uns, dass bei hinreichender Regularität der Lösung (40) gilt:

$$\inf_{u_h, \sigma_h} F(u_h, \sigma_h) = O(h^{2(k+1)})$$

Um den durch die Glättung verursachten Fehler in der gleichen Größenordnung zu halten ist also

$$\varepsilon^2 h = O(h^{2(k+1)})$$

zu gewährleisten. Daher setzen wir für die verwendeten quadratische Elemente ($k = 1$)

$$\varepsilon := O(h^{k+\frac{1}{2}}) = O(h^{\frac{3}{2}})$$

und erhalten dann einen Approximationsfehler in vernachlässigbarer Größenordnung.

KAPITEL 5

Verminderung des Zeitaufwandes

Um das Verfahren weiter optimieren zu können ist es hilfreich zu überlegen und zu messen, welches die zeit- und speicheraufwendigsten Teile sind. Wir vermuten zunächst, dass das Lösen der Normalengleichung der zeitaufwendigste Teil ist und suchen daher einen effizienten Löser. Tabelle 1 zeigt Testergebnisse in Matlab Version 7.1.0.124 (R14) Service Pack 3 auf einem aktuellen Centrino-Notebook. Angegeben ist der Zeitaufwand zum Lösen der auftretenden positiv definiten Normalengleichungen für drei verschiedene Systemgrößen. Gegenübergestellt sind der eingebaute Matlab-Löser, das bereits integrierte "UMFPACK" das durch einen kleinen Trick benutzt werden kann, "CHOLMOD" und 10 Schritte mit dem CG-Verfahren. Es zeigt sich, dass der direkte Löser CHOLMOD in unserem Anwendungsfall allen anderen Lösern gegenüber überlegen ist.

In den Abbildungen 5.2 und 5.3 sind Messergebnisse in Abhängigkeit von der Systemgröße dargestellt. Man erkennt, dass der Aufwand für die meisten Teilaufgaben sich wie erwartet in etwa proportional zu der Anzahl der Freiheitsgrade verhält. Kleine Abweichungen von der Linearität ergeben sich u.a. dadurch, dass das Aufstellen von Matrizen in Matlab mittels `sparse()` eine Sortierung der Indizes mit $O(n\log(n))$ Zeitaufwand beinhaltet. Die Faktorisierung zeigt ein überlineares Verhalten sowohl im Speicher- als auch im Rechenaufwand. Der theoretische Speicheraufwand beträgt bei geschickter Permutation $O(n\log n)$, während der Zeitaufwand mit $O(n^{3/2})$ beschränkt ist. Damit ist das Lösen asymptotisch am aufwendigsten.

In Abbildung 5.1 ist jedoch gut zu erkennen, dass selbst für sehr große Gleichungssysteme der Lösungsaufwand nicht am aufwendigsten ist. Das direkte Lösen ist sogar weniger aufwendig als die Matrizenmultiplikation

$$A = J^T J \text{ mit } J \in \mathbb{R}^{m\times n}.$$

Dies erklärt sich dadurch, dass bei Least-Squares-Formulierungen normalerweise m ein Vielfaches von n (siehe Abbildung 5.4(b)) ist.

n=54866	n=109662	n=222166	Befehlszeile:
4,3sek	11.9sek	32,2sek	`x=A\b;`
2,8sek	6,7sek	15,1sek	`i=max(find(A(1,:)));A(1,i)=A(1,i)*(1+2*eps);x=A\b;`
1,6sek	3,5sek	8,4sek	`x=cholmod(A,b); %cholmod muss installiert werden`
n=54866	n=109662	n=222166	Befehlszeile:
2,4sek	4,3sek	8,8sek	`x=pcg(A,x,0,10);`

TABELLE 1. Performancevergleich verschiedener Löser

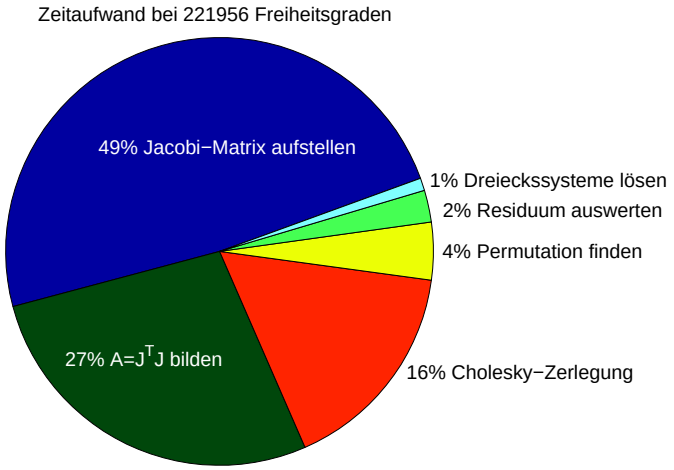


ABBILDUNG 5.1. Prozentualer Vergleich des Zeitaufwandes einzelner Programmteile.

Am aufwendigsten ist es die Matrix selbst aufzustellen. Der Aufwand zum Aufstellen der Jacobi-Matrix ist proportional zu m und es ist m proportional zur Anzahl der Quadraturpunkte. Daher ist es wünschenswert die Anzahl der Quadraturpunkte gering zu halten.

5.1. Unterintegration

Weniger Quadraturpunkte bedeuten kleinere Jacobi-Matrizen (siehe Abbildung 5.4 links und Mitte) ohne dass sich die Größe der Systemmatrix A (rechts) ändert. Wir versuchen daher die Anzahl der Quadraturpunkte möglichst gering zu halten:

In der Galerkin-FEM gilt die Faustregel, dass man die Integrale des Residuums

$$\int_\Omega \epsilon(u_h) : \mathcal{C}\epsilon(u_h)$$

exakt berechnen muss ([**21**, S. 201]) um mit konformen Elementen die Konvergenzordnung erzielen zu können, die sich bestmöglich erreichen lässt. Also wählt man als Quadraturgrad d den Polynomgrad von $\epsilon(u_h) : \mathcal{C}\epsilon(u_h)$

$$d = 2(p-1),$$

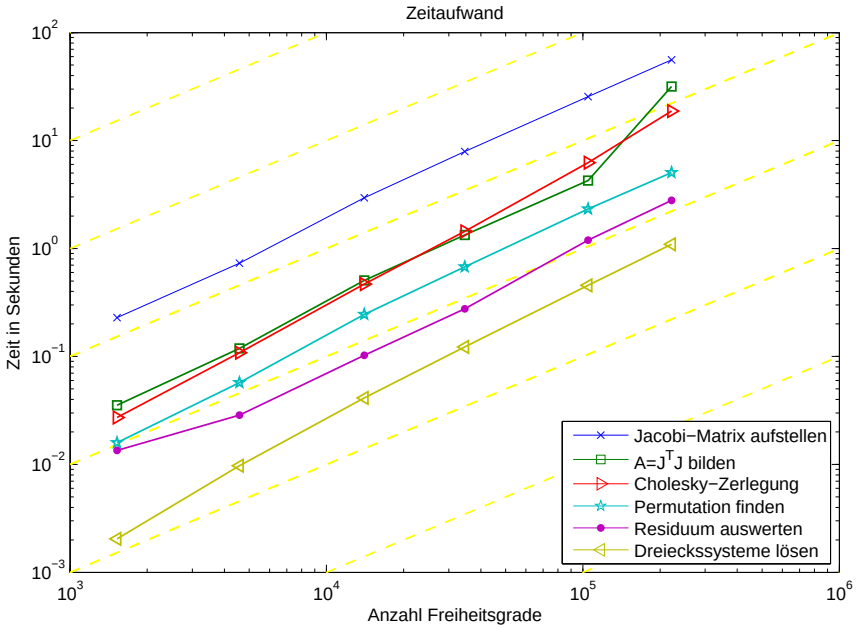


ABBILDUNG 5.2. Zeitaufwand der wichtigsten Programmteile in Abhängigkeit von der Anzahl der Freiheitsgrade. Die gelben gestrichelten Linien demonstrieren lineares Verhalten in der doppelt logarithmischen Darstellung.

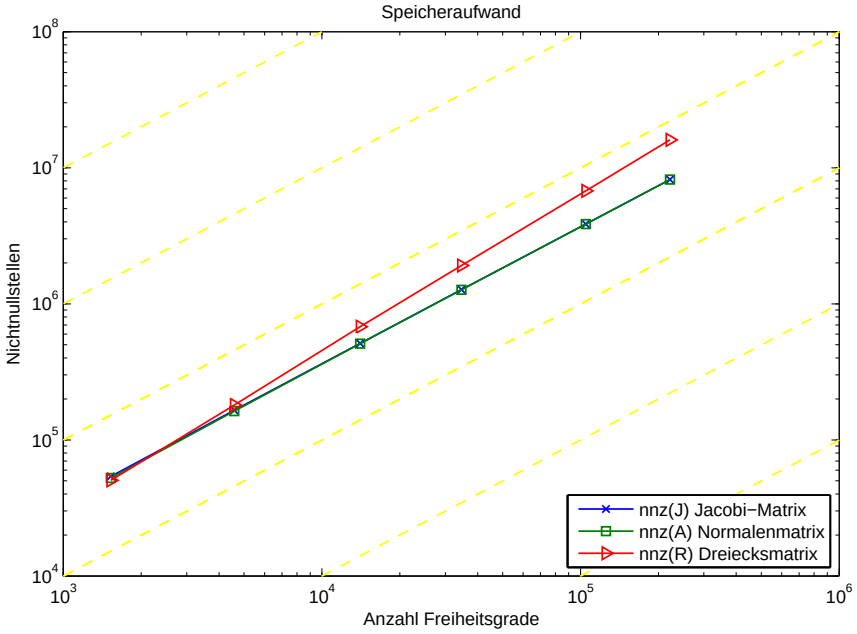


ABBILDUNG 5.3. Speicheraufwand aufgestellter Matrizen in Abhängigkeit von der Anzahl der Freiheitsgrade. Die gelben gestrichelten Linien demonstrieren lineares Verhalten.

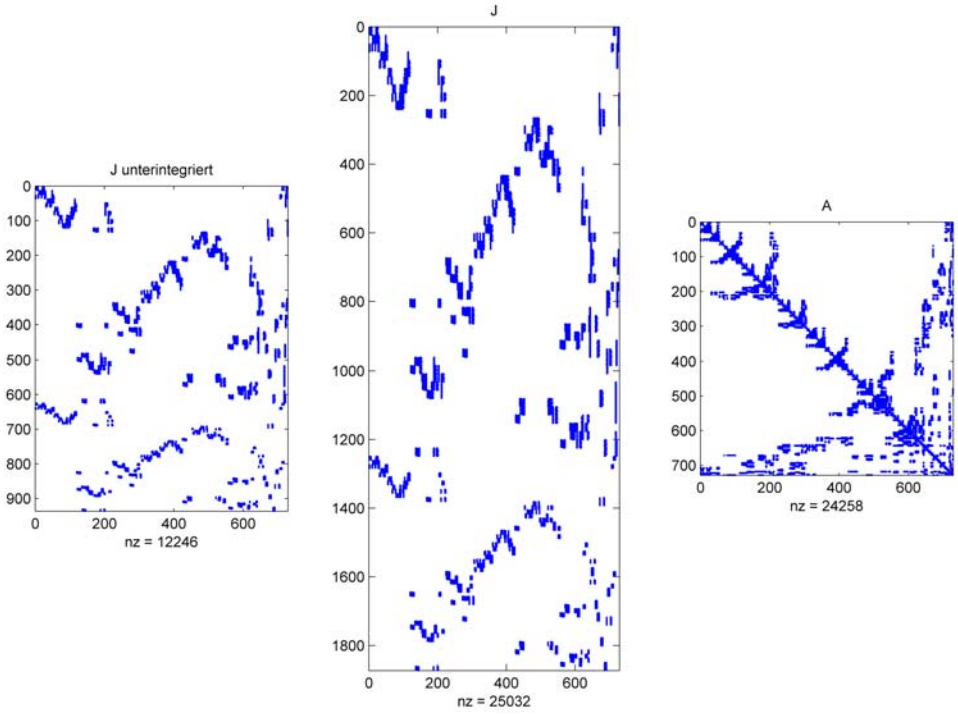


ABBILDUNG 5.4. Besetztheitsstruktur auftauchender Matrizen (in minimal degree Ordnung)

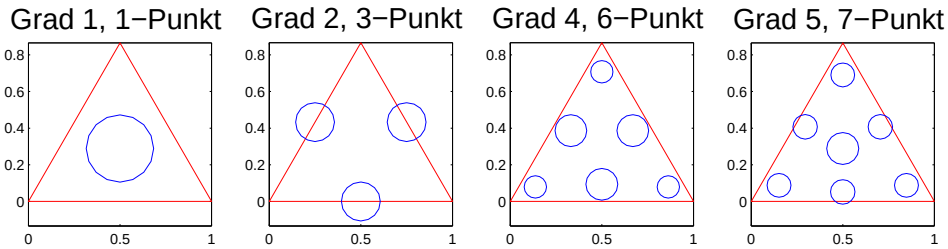


ABBILDUNG 5.5. Ausgewählte Quadraturformeln. Die Kreismittelpunkte stellen die Quadraturpunkte dar, und die Kreisflächen verdeutlichen die unterschiedlichen Gewichte.

welcher für die exakte Integration (bis auf Rundungsfehler) der Variationsformulierung im linear elastischen Fall ($\mathcal{C}$ konstant) ausreicht, wenn $u_{h|T\in\mathcal{T}} \in \mathbb{P}_p^3$. Ein kleineres d führt zur Unterintegration, die von Ingenieuren teilweise zur Stabilisierung ihrer Methoden eingesetzt wird.

Bei der gemischten LSFEM hingegen hat ein Teil der Integrale die Form

$$\int_\Omega \|\sigma_h - \sigma(u_h)\|^2 .$$

Wegen des Quadrates der Norm ist hier zur exakten Integration ein Exaktheitsgrad

$$d = \deg \|\sigma_h\|^2 = 2p$$

für $\sigma_{h|T\in\mathcal{T}} \in \mathbb{P}_p^{3\times 3}$. notwendig.

Es fällt auf, dass in der gemischten LSFEM nicht für alle Residuen derselbe Exaktheitsgrad benötigt wird um im linearen Fall eine exakte Integration zu bekommen: Hat σ_h wie u_h den Polynomgrad p so ist div σ vom Grad $p-1$. Folglich ist $\deg \|\text{div } \sigma\|^2 = 2(p-1)$. Wir wählen als Beispiel $p = 2$ und erhalten für den konstitutiven Term Polynome vom Grad 4, während der Divergenz-Term nur vom Grad 2 ist. Bei einem Dreieck reichen drei Quadraturpunkte auf den Seitenmitten zur exakten Integration für Polynome vom Grad 2, während für den Quadraturgrad 4 doppelt so viele Quadraturpunkte benötigt werden (siehe Abbildung 5.5 auf der vorherigen Seite).

Anders als bei Variationsformulierung hat bei der Methode der kleinsten Quadrate die Anzahl und Art der Quadraturpunkte noch eine weitere Bedeutung: Jede lokale Gleichung des Randwertproblems liefert an jedem Quadraturpunkt ein Residuum. Deshalb ist bei der LSFEM d mindestens so groß zu wählen, dass nicht weniger Gleichungen m als Unbekannte n auftreten [**40**, S. 75f]. Für die LSFEM wurde bereits beobachtet, dass Unterintegration zu brauchbaren Ergebnissen führen kann [**40**, S. 110-117]. Jiang meint sogar, exakte Integration bei der LSFEM würde zu schlechteren Ergebnissen führen [**40**, S. 144].

Ein numerischer Test (Abbildung 5.6 auf der nächsten Seite) zeigt, das in der LSFEM ein Quadraturgrad von

$$d = 2p - 2$$

auch für den konstitutiven Term ausreicht. Dazu wurde die Minimalstelle $\tilde{x}$ in einem unterintegriertem Funktional $\tilde{f}$ gesucht und zur Kontrolle mit dem exakt integrierten Fehlerschätzer f der Fehler abgeschätzt. Die Abbildung zeigt, dass $f(\tilde{x}) > f(x)$ ist, die Differenz aber - auch asymptotisch - unerheblich ist. Dadurch, dass durch diese hinreichende Unterintegration die Jacobi-Matrix nur halb so viele Zeilen hat, halbiert sich der Aufwand zum Aufstellen der Jacobi-Matrix. In Abbildung 5.6 ist im rechten Teilbild zu erkennen, dass sich diese Unterintegration auch im Gesamtzeitaufwand in folgendem Sinne auszahlt: Den exakt integrierten Fehlerschätzer durch Raumverfeinerung unter eine bestimmte Schwelle zu drücken gelingt mit Unterintegration wesentlich schneller als mit der genaueren Optimierung des exakt integrierten Funktionals.

5.2. Die lokale Berechnung des Konsistenzparameters

Anders als bei elastischer Verformung werden bei plastischer Verformung zur Aufstellung der Systemmatrizen nichtlineare Funktionen benötigt, deren Berechnung wesentlich teurer ist. Während man bei J_2-Plastizität nicht umhinkommt, die Wurzel aus $\left\|\text{dev } \sigma^{trial}\right\|^2$ zu berechnen, können wir hingegen die Anzahl der Auswertung von Exponentialfunktionen bei exponentieller Verfestigung wesentlich reduzieren:

Bei exponentieller Verfestigung ist ein lokales Gleichungssystem zu lösen. Bisher wurde dafür zumeist ein Newton-Verfahren verwendet. Erfahrungsgemäß braucht

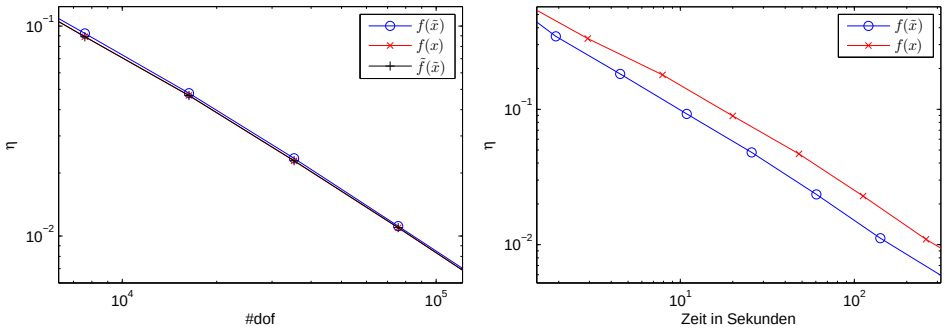


ABBILDUNG 5.6. Ergebnisse der Unterintegration. Dabei ist $\tilde{x}$ Minimalstelle von $\tilde{f}$ (mit Quadraturgrad 2 approximiert) und x Minimalstelle von f (exakt mit Quadraturgrad 4 integriert).

man etwa sechs Iterationen um die Lösung anzunähern. In jedem Iterationsschritt ist dabei mindestens eine Exponentialfunktion auszuwerten.

In diesem Abschnitt wird gezeigt, wie sich mit Hilfe einer anderen transzendenten Funktion eine analytische Lösung für das lokale Gleichungssystem finden lässt wenn exponentielle Verfestigung und assoziierte Flußrichtung vorliegt. Im nächsten Abschnitt wird dann gezeigt, wie sich diese transzendente Funktion mit Auswertung von nur einer Logarithmusfunktion hinreichend annähern lässt.

Dazu fassen wir zunächst noch einmal die zugrunde liegenden lokalen Gleichungen (nach [**57**, S.91]) zusammen: Exponentielle Verfestigung (43) wird ausgedrückt durch eine Funktion

$$\psi(\sigma,\gamma) \quad = \quad \|\text{dev}\ (\sigma)\| - \sqrt{\tfrac{2}{3}} K\,(\gamma) \tag{43}$$

mit

$$K(\alpha) = K_0 + H\alpha + (K_{inf} - K_0)(1 - \exp -\omega\alpha)$$
$$\sigma = \sigma_{trial} - \mathcal{C} n \gamma$$
$$n = \frac{\text{dev}\ (\sigma)}{\|\text{dev}\ (\sigma)\|} = \frac{\text{dev}\ (\sigma_{trial})}{\|\text{dev}\ (\sigma_{trial})\|} \text{ (assoziierter Flußrichtung).}$$

Dazu gehört die Evolution des internen Verfestigungsparameters α über den Konsistenzparameter λ

$$\dot{\alpha} = \lambda\sqrt{\tfrac{2}{3}}\ .$$

Wird das Zeitinkrement von λ abgekürzt durch $\gamma := \Delta\lambda$ ergibt sich für einen Zeitschritt:

$$\alpha_{new} = \alpha_{old} + \gamma\sqrt{\tfrac{2}{3}}$$

gesucht ist die Lösung γ^* des Gleichungssystem

$$\psi(\sigma^{trial}, \gamma) = 0.$$

Die Versuchsspannung σ^{trial} ist dabei fest (von dem globalen Löser) vorgegeben und so handelt es sich um ein eindimensionales Optimierungsproblem

$$\psi(\gamma^*) = 0 \tag{44}$$

für das meist ein (lokales) Newton-Verfahren verwendet wird. Eine direkte Lösung ergibt sich wie folgt:

Die Verfestigung lässt sich in Abhängigkeit von γ ausdrücken:

$$\begin{aligned} K(\alpha(\gamma)) &= K_0 + H\left(\alpha_{old} + \gamma\sqrt{\tfrac{2}{3}}\right) + (K_{\inf} - K_0)\left(1 - \exp -\omega\left(\alpha_{old} + \gamma\sqrt{\tfrac{2}{3}}\right)\right) \\ &= [K_{\inf} + H\alpha_{old}] + [\sqrt{\tfrac{2}{3}}H]\cdot\gamma - \sqrt{\tfrac{3}{2}}a\cdot\exp -h\cdot\gamma \end{aligned}$$

mit den Konstanten $a := \sqrt{\frac{2}{3}}(K_{\inf} - K_0)\exp -\omega\alpha_{old}$ und $h := \sqrt{\frac{2}{3}}\omega$. Also ist mit abgekürzter Schreibweise für die Flussrichtung $d = \text{dev}\ (\sigma_{trial})$:

$$\begin{aligned} \psi(\gamma) + \sqrt{\tfrac{2}{3}}K(\alpha(\gamma)) &= \left\| d - \frac{d}{\|d\|}2\mu\gamma \right\| = \left\| \left(1 - \frac{2\mu\gamma}{\|d\|}\right) d \right\| \\ &= |1 - \frac{2\mu\gamma}{\|d\|}| \cdot \|d\| = |\|d\| - 2\mu\gamma|. \end{aligned}$$

An dieser Stelle folgern wir aus (21) $2\mu\gamma \leq \|d\|$ und erhalten:

$$\begin{aligned} \psi(\gamma) &= \|d\| - 2\mu\gamma - \sqrt{\tfrac{2}{3}}K(\gamma) \\ &= [\|d\| - \sqrt{\tfrac{2}{3}}K_{\inf} - \sqrt{\tfrac{2}{3}}H\alpha_{old}] - [2\mu + \tfrac{2}{3}H]\cdot\gamma + a\cdot\exp -h\cdot\gamma \\ &= c - b\cdot\gamma + a\cdot\exp -h\cdot\gamma \end{aligned}$$

Mit den Konstanten $b := 2\mu + \frac{2}{3}H$ und $c := \|d\| - \sqrt{\frac{2}{3}}K_{\inf} - \sqrt{\frac{2}{3}}H\alpha_{old}$. Damit hat die Gleichung (44) also folgende Gestalt :

$$\begin{aligned} c - b\cdot\gamma + a\cdot\exp(-h\cdot\gamma) &= 0 \\ \Rightarrow a\cdot\exp(-h\cdot\gamma) &= b\gamma - c \\ \Rightarrow a &= (b\gamma - c)\cdot\exp(h\cdot\gamma) \\ \Rightarrow a\frac{h}{b} &= \left(h\cdot\gamma - c\frac{h}{b}\right)\cdot\exp(h\cdot\gamma) \\ \Rightarrow a\frac{h}{b}\exp(-c\frac{h}{b}) &= \left(h\cdot\gamma - c\frac{h}{b}\right)\cdot\exp(h\cdot\gamma)\exp(-c\frac{h}{b}) \\ &= \left(h\cdot\gamma - c\frac{h}{b}\right)\cdot\exp(h\cdot\gamma - c\frac{h}{b}) \end{aligned}$$

Wir substituieren nun die Konstante $x := a\frac{h}{b}\exp -c\frac{h}{b}$ und eine Variable $w := h\cdot\gamma - c\frac{h}{b}$. und erhalten damit die transzendente Gleichung $x = w\cdot\exp w$ mit der für $x > 0$ eindeutigen Lösung $w = W(x)$ (Abbildung 5.7). Setzt man alle substituierten

Terme wieder ein, erhält man die gesuchte Lösung

$$\begin{aligned}\gamma &= \frac{w}{h} + \frac{c}{b} \\ &= \frac{1}{h} W\left(a\frac{h}{b}\exp\left(-c\frac{h}{b}\right)\right) + \frac{c}{b} \\ &= \frac{1}{\sqrt{\frac{2}{3}}\omega} W\left(\left(\left(K_{\text{inf}} - K_0\right)\exp\left(-\omega\alpha_{old}\right)\right)\frac{\frac{2}{3}\omega}{2\mu + \frac{2}{3}H} e^{\left(\|d\| - \sqrt{\frac{2}{3}}K_{\text{inf}} - \sqrt{\frac{2}{3}}H\alpha_{old}\right)\frac{-\sqrt{\frac{2}{3}}\omega}{w\mu + \frac{2}{3}H}}\right) + \\ &\quad + \frac{\|d\| - \sqrt{\frac{2}{3}}K_{\text{inf}} - \sqrt{\frac{2}{3}}H\alpha_{old}}{2\mu + \frac{2}{3}H} \\ &= \sqrt{\frac{3}{2}}\frac{1}{\omega} W\left(\left(K_{\text{inf}} - K_0\right)\frac{\omega}{3\mu + H}\exp\left(\left(\sqrt{\frac{3}{2}}\|d\| - K_{\text{inf}} + 3\mu\alpha_{old}\right)\frac{-\omega}{3\mu + H}\right)\right) + \\ &\quad + \sqrt{\frac{3}{2}}\left(\frac{\sqrt{\frac{3}{2}}\|d\| - K_{\text{inf}} - H\alpha_{old}}{3\mu + H}\right).\end{aligned}$$

Für $\left(K_{\text{inf}} - K_0\right)\frac{\omega}{3\mu+H} > 0$ liefert die Definition von $W(x)$ eine eindeutige Lösung, also insbesondere, wenn gleichzeitig $K_{\text{inf}} > K_0$, $\omega > 0$ und $H > -3\mu$. Dies ist bei für Metall üblichen Parametersätzen der Fall. Zu fest gegebenen Materialkonstanten H, K_0, K_{inf} lässt sich dann auch weiter umformen:

(45)

$$\begin{aligned}\gamma\left(\alpha_{old}, d\right) = \sqrt{\frac{3}{2}}\frac{1}{\omega} W\left(\exp\left(\left(\sqrt{\frac{3}{2}}\|d\| - K_{\text{inf}} + 3\mu\alpha_{old} + r\right)\frac{-\omega}{3\mu + H} + r\right)\right) + \\ + \sqrt{\frac{3}{2}}\left(\frac{\sqrt{\frac{3}{2}}\|d\| - K_{\text{inf}} - H\alpha_{old}}{3\mu + H}\right)\end{aligned}$$

mit der Konstanten $r = \log\left(\left(K_{\text{inf}} - K_0\right)\frac{\omega}{3\mu+H}\right)$. Benötigt wird also eine Möglichkeit $W(\exp y)$ zu berechnen.

5.3. Die Berechnung der Lambert-W-Funktion

Die Gleichung

$$x = We^W$$

definiert die Umkehrfunktion $W(x)$ die heutzutage Lambert-W-Funktion genannt wird. Sie ist eindeutig definiert für $x \geq 0$. Für $-e^{-1} < x < 0$ existieren zwei reelle Lösungen der Gleichung, $W(x)$ ist als die größere der beiden Lösungen definiert. In unserem Anwendungsfall (45) brauchen wir die numerische Lösungen für $W\left(\exp y\right)$. Das ist sehr praktisch, da $W\left(x\right) \approx \log x = y$ ein gute Näherung ist, deren Berechnung weder die Logarithmusfunktion noch die Exponentialfunktion braucht. In dem Artikel [**25**] wird die Ableitung für Lamberts W-Funktion hergeleitet:

$$W(x) = \begin{cases} \frac{W(x)}{x(1+W(x))} & \text{für } x \neq 0 \\ 1 & \text{für } x = 0 \end{cases}$$

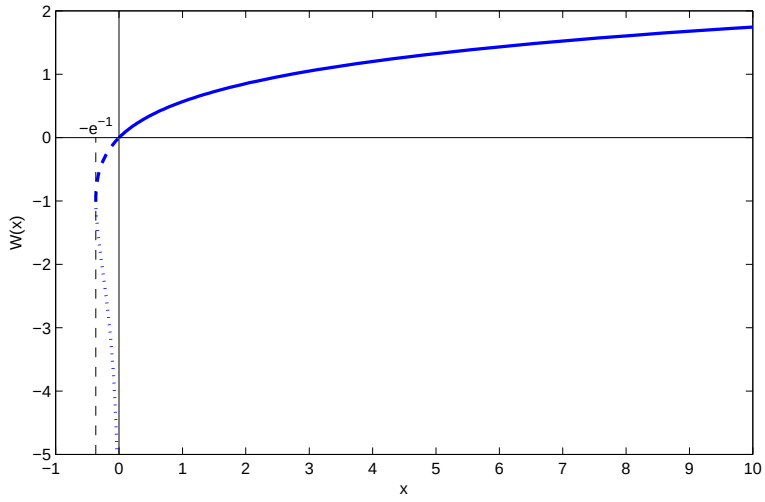


ABBILDUNG 5.7. Lamberts W-Funktion. In dieser Anwendung interessiert nur der durchgezogene Funktionsverlauf für $x > 0$.

Daraus folgt sofort die noch einfachere Ableitung für $W(\exp y)$:

$$\frac{\partial}{\partial y} W(\exp y) = \frac{W(\exp y)}{(1 + W(\exp y))}$$

In [**24**] wird Halleys Methode (Konvergenz dritter Ordnung) auf $w \exp w = x$ angewandt und man erhält so eine Iterationsvorschrift, die sich für Langzahlarithmetik eignet:

$$E := \exp w_j$$

$$w_{j+1} := w_j - \frac{w_j E - x}{E\,(w_j + 1) - \frac{(w_j+2)(w_j E - x)}{2w_j + 2}} \quad \text{(Iteration 3. Ordnung)}$$

Für Fließkomma-Arithmetik lassen sich jedoch bessere Verfahren angeben: [**28**] geben Iterationen mit Konvergenz dritter und vierter Ordnung an. Ausgehend von der logarithmierten Gleichung $w + \log(w) = \log(x)$ erhalten sie durch die Iterationsverfahren 4. und 3. Ordnung:

$$z_j := \log(x) - \log(w_j) - w_j = y - \log(w_j) - w_j$$

$$w_{j+1} := w_j \left(1 + \frac{z_j\left((1 + w_j)\,(1 + w_j + 2/3 z_j) - 1/2 z_j\right)}{(1 + w_j)\left((1 + w_j)\,(1 + w_j + 2/3 z_j) - z_j\right)}\right) \quad \text{(Iteration 4. Ordnung)}$$

$$\text{bzw: } w_{j+1} := w_j \left(1 + \frac{z_j\,(1 + w_j + 2/3 z_j)}{(1 + w_j)\,(1 + w_j + 2/3 z_j) - 1/2 z_j}\right) \quad \text{(Iteration 3. Ordnung)}$$

Als Startnäherung wird dabei verwendet:

$$w_0 = \begin{cases} y - \frac{24y^2 + 48y - 72}{7y^2 + 58y + 127} & \text{für } \exp y := x > x_s, \\ \frac{6x + 8x^2}{6 + 14x + 5x^2} & \text{für } x \le x_s := 0.7385, \end{cases}$$

$$\text{bzw: } w_0 = \begin{cases} y, & \text{für } \exp y := x > x_s \\ \frac{6x + 8x^2}{6 + 14x + 5x^2} & \text{für } x \le x_s := 0.646 \end{cases}$$

In [**4**] und darauf folgenden Veröffentlichungen wurden weiter Approximationen (insbesondere für $x < 0$) angegeben, die aber für $x > 0$ wegen extensiven Gebrauchs von transzendenten Funktionen wesentlich langsamer sind.

5.4. Eine neue Methode zur Approximation der Lambert-W-Funktion

Bei näherer Betrachtung entpuppen sich die in dem Artikel [**28**] beschrieben Iterationen als Padé-Approximationen von $W(w_j \exp w_j + z_j)$ um $z_j = 0$, so dass sie heutzutage leicht mit Programmen zum symbolischen Rechnen nachvollzogen werden können, wenngleich diese derzeit die Ausdrücke noch nicht optimal vereinfachen. Mit dem Computeralgebra-System *Maple* lässt sich beispielsweise das Iterationsverfahren 4ter Ordnung in wenigen Zeilen reproduzieren:

```
with(numapprox);
pw := pade(LambertW(wj*exp(wj+zj)),zj=0,[2,1]):
w  := simplify(pw,symbolic);
```

Die erste Zeile gibt an, dass Maple die Routine `pade` einbindet, die in dem Paket `numapprox` enthalten ist. Die zweite Zeile erstellt die 2/1 Padé-Approximation und die dritte Zeile vereinfacht den gewonnen Ausdruck. Der Parameter `symbolic` ist dabei unerlässlich, um Vereinfachungen zu treffen, die mit unserer Annahme $x > 0$ übereinstimmen. Mit dieser Methode lassen sich dann auch leicht Verfahren höherer Ordnung konstruieren in dem einfach höhere Padé-Approximationen verwendet werden. Ebenso lässt sich obige Startnäherungen für $x > x_s$ als 2/2 Padé-Approximationen von $W(\exp y) - y$ um $y = 1$ nachvollziehen zu der noch $y = log(x)$ addiert werden muss. Dies ist nach zahlentheoretischen Überlegungen übrigens die einzige Stelle, um der eine Taylorentwicklung rationale Koeffizienten haben kann. Und rationale Koeffizienten sind wünschenswert um sich nicht um Rundungsfehler der Koeffizienten kümmern zu müssen. Folgender Maple Einzeiler leistet das Gewünschte:

```
w0 := subs(y=log(x),simplify(pade(LambertW(exp(y))-y,y=1,[2,2]))+y);
```

Diese Sichtweise legt nahe, dass man stattdessen gleich eine 3/2 Padé-Approximationen von $W(\exp y))$ um $y = 1$ benutzt:

```
w32 := simplify(subs(y=log(x),pade(LambertW(exp(y)),y=1,[3,2])));
```

Diese braucht allerdings eine Multiplikation mehr (wegen dem Faktor von y^3) und außerdem ist der relative Fehler von `w32` für $x \to \infty$ zwar endlich aber nicht verschwindend, während für `w0` $\lim_{x\to\infty} \frac{w_0(x)-W(x)}{W(x)} = 0$ gilt. Andererseits zeigt eine Taylor-Reihenentwicklung des Fehlers e_j für den Iterationsschritt $w_{j+1} = w_j(1+e_j)$ um $\frac{z_j}{1+w_j} = 0$

```
e := subs(zj=zw*(1+wj),LambertW(wj*exp(wj+zj))/wj-1):
e := subs(zw=zj/(1+wj),simplify(series(e,zw=0,1),symbolic));
```

$$e_{j+1} = 0 + O(\frac{z_j}{1+w_j}),$$

dass mit $x \to \infty \Rightarrow w_0 \to \infty \Rightarrow e_1 \to 0$. Das heißt also, dass für große x bereits nach einem Iterationsschritt der relative Fehler verschwindet, auch wenn der relative Fehler der Startnäherung nicht verschwindet. Insgesamt zeigt sich, dass die Startnäherung `w32` der Startnäherung w_0 vorzuziehen ist, da sie in Verbindung mit einer Nachiteration eine bessere Lösung liefert, aber nur wenig teurer ist.

Nach ein paar geschickten Substitutionen erhalten wir die folgenden Formeln zur Approximation von $W(x)$:

$$
\begin{aligned}
y &= \log x \\
w_0 &= \frac{49381993 + 48445604y + 21269310y^2 + y4974836y^3 + 543937y^4}{8(10883927 + 3732519y + 891981y^2 + 68533y^3)} \\
w &= w_0 + 1 \\
z &= \frac{y - \log w_0 - w_0}{w^2} \\
V &= 540 + (-720 + 120w)w \\
W &= \frac{w_0(1+(V+(-810+(1800+(-1140+144w)w)w+(135+(-540+(720+(-352+36w)w)w)w)z)z}{V+(-1080+(2160+(-1200+144w)w)w+(405+(-1080+(1020+(-384+36w)w)w)w)z)z)zw}
\end{aligned}
$$

Die Frage, die sich nun stellt ist, wie viel Iterationsschritte die Verfahren brauchen um doppelte Genauigkeit nach ANSI-IEEE 754 Standard zu erreichen und ob Iterationsverfahren höherer Ordnung, die zwangsläufig weniger Schritte benötigen, auch wirklich schneller sind, oder ob der Rechenaufwand dafür zu groß ist. Die Entwickler bei Intel empfehlen jedenfalls, man solle keine noch so langen Polynome scheuen.

Um die benötige Iterationszahl messen zu können, ist erst einmal festzustellen, welche Rechengenauigkeit überhaupt erreicht werden kann. Wir gehen davon aus, dass die Berechnungen ebenfalls mit Fließkommazahlen doppelter Genauigkeit durchgeführt werden. Fest steht auf jeden Fall von Vornherein, dass dies bei der Berechnung der transzendenten Funktionen zum Rundungsdilemma führt [**31**]: Man kann kein korrekt gerundetes Ergebnis berechnen. Zweitens gilt dies bereits für die Implementationen der transzendenten Funktionen `exp` und `log`, für die der IEEE Standard deshalb auch keine korrekte Rundung vorschreibt. Das führt dazu, dass sie auf unterschiedlicher Hardware zu verschiedenen Ergebnissen kommen können. In [**9**] wird ein Verfahren vorgestellt, was die korrekte Rundung für Polynome mit festen Koeffizienten überprüft. Die Bedingungen dafür erfüllen die hier vorgestellten Polynome jedoch nicht. Als Genauigkeit soll daher genügen, dass das Ergebnis nur wenige *Ulps* ("Units in the last Place") von der Lösung liegt, dass also zwischen dem exakten und dem berechneten Wert nicht viele andere Gleitkommazahlen liegen.

Angewendet auf die Exponentialgleichung konvergiert ein Newton-Verfahren bei konstant vorgegebener Startnäherung gar nicht mit fester Iterationszahl. Für $x \to \infty$ wird der relative Fehler nach endlich vielen Iterationen beliebig schlecht. Das Newton-Verfahren angewandt auf die logarithmierte Gleichung konvergiert nach 5

Algorithm 1 Die Matlab Implementation zum Berechnen von $W(x)$ für $x > 0,2$

```
y=log(x);
w0=(49381993+(48445604+(21269310+(4974836+543937.*y).*y).*y).*y)...
    ./(8.*(10883927+(3732519+(891981+68533.*y).*y).*y));
w=w0+1;
z=(y-log(w0)-w0)./(w.^2);
w= w0.*(1+(540+(-720+120.*w).*w+ ( -810+(1800+(-1140+144.*w).*w).*w+...
             (135+(-540 +(720 +(-352+36.*w).*w).*w).*w).*z).*z)...
   ./(540+(-720+120.*w).*w+ (-1080+(2160+(-1200+144.*w).*w).*w+...
             (405+(-1080+(1020+(-384+36.*w).*w).*w).*w).*z).*z.*w);
```

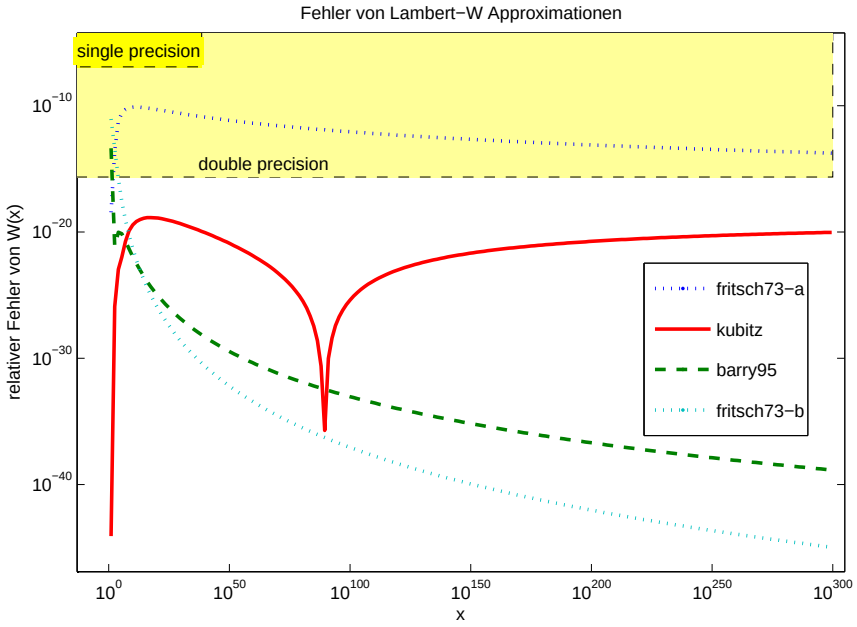


Abbildung 5.8. Verschiedene Approximationen für $W(x)$ für $x \to \infty$ im Vergleich. Die hier neu vorgestellte Approximation ist für einen relativen Fehler von weniger als 10^{-16} optimiert.

Iterationen. Für die oben genannten Verfahren 3. und 4. Ordnung reichen 2 Iterationsschritte, wegen der guten Startnäherungen. Das hier konstruierte Verfahren 5. Ordnung konvergiert schon nach nur einem Iterationsschritt auf die gewünschte Genauigkeit, wenn als Startnäherung eine 4/3-Padé-Approximation verwendet wird (siehe Abbildung 5.8). Tests auf unterschiedlichen Plattformen haben ergeben, dass dies durchweg schneller ist als die anderen Verfahren: Auf aktuellen CPUs verhält sich die Matlab Implementation (Algorithmus 1) der hier vorgeschlagenen Approximation fast doppelt so schnell wie die bisher best bekannte (Tabelle 2).

5.5. Eine neue Autodifferentiationklasse für Matlab

Um die Jacobi-Matrix J zu bestimmen ist die (dünnbesetzte) partielle Ableitung des Residuums R zu bestimmen. Diese kann man analytisch ausrechnen und implementieren. Wie die korrekten "konsistenten Tangenten" für verschiedene

	Intel Centrino bei 2 GHz	AMD Opteron bei 2,6 GHz
Fritsch73	0,62 Sekunden	0,51 Sekunden
Kubitz05	0,44 Sekunden	0,29 Sekunden

TABELLE 2. Perfomancevergleich zum Auswerten von 10 Millionen $W(\log x)$ Funktionen

Materialgesetze berechnet werden können, füllt viele Seiten in Ingenieurshandbüchern, die Implementierung kostet Zeit und ist sehr fehlerträchtig. Anstatt hier die Jacobi-Matrix analytisch herzuleiten wurde daher in dieser Arbeit automatische Differentiation verwendet. Diese erlaubt es auch neue Materialgesetze in wenigen Minuten zu implementieren: Dafür ist nur der konstitutive Zusammenhang neu zu implementieren, ohne sich um dessen Tangente kümmern zu müssen. Automatische Differentiation kann außerdem verwendet werden um die lokale Konsistenzgleichung zu lösen.

Automatische Differentiation erlaubt die Berechnung von Ableitungen beliebiger computerimplementierter Funktionen. Die Berechnung der Ableitung in eine Richtung braucht nur ein kleines Vielfaches (maximal das Vierfache an Operationen) der Berechnung der Funktion selber [**33**, S.10]. Unterschieden wird zwischen Vorwärts- und Rückwärtsdifferentiation. Rückwärtsdifferentiation ist effizienter, wenn die Anzahl der Eingabevariablen, nach denen abgleitet werden soll, viel größer ist als die Zahl der Ausgabevariablen. In dieser Arbeit wird die Differentiation für die (Gauss-)Newton Verfahren verwendet, wo (durch Unterintegration) die Systemmatrizen in etwa gleich viele Spalten wie Zeilen haben. Daher wird die einfacher zu implementierende Vorwärtsdifferentiation benutzt.

Die einzig frei verfügbare Autodifferentiationsklasse für Matlab war bisher Intlab aus [**52**]. Leider war diese gänzlich ungeeignet für die in dieser Arbeit verwendeten Gleichungssysteme: Erstens waren manche Ableitungen mit Schleifen implementiert, was in Matlab sehr langsam ist. Zweitens wurden globale Variablen verwendet, so dass es unmöglich war Gleichungssysteme zu schachteln. Drittens wurden manche Matlab interne Fehler im Umgang mit dünnbesetzten Matrizen nicht umgangen. Ausserdem fehlten ein paar benötigte Operatorüberladungen.

Daher wurde eine komplett neue Toolbox mit einer Klasse `autodiff` innerhalb dieser Dissertation entworfen. Damit kann von einer (fast) beliebigen in Matlab implementierten Funktion $R : \mathbb{R}^n \to \mathbb{R}^m$ die Jacobi-Matrix $J(x)$ durch

$$\texttt{J} = \texttt{jacobian}(\texttt{R}(\texttt{autodiff}(\texttt{x}))$$

berechnet werden. Für nicht differenzierbare Funktionen wird ein Element aus der verallgemeinerten Ableitung zurückgeliefert. Auch die (für Dämpfung benötigte) Richtungsableitung $J(x)[p]$ von R an der Stelle x in Richtung p ist leicht berechenbar mit

$$\texttt{j} = \texttt{jacobian}(\texttt{R}(\texttt{autodiff}(\texttt{x} + \texttt{p} * \texttt{autodiff}(0))))$$

Die Implementierung ist auf dünnbesetzte Ableitungen spezialisiert. Die Ableitungen wurden vollständig vektorisiert (ohne Schleifen) implementiert, so dass die Ableitung wesentlich schneller als in Intlab berechnet wurden. Ein paar Implementierungen wurden in Intlabs Version 5.2 übernommen.

Die Performance-Messergebnisse in Abbildung 5.1 auf Seite 54 zeigen, dass der Mehraufwand durch die Autodifferentiation gering sein muss, da das Berechnen der Jacobi-Matrix gerade mal doppelt so lange dauert wie die Matrix-Multiplikation $A = J^T J$. Das Berechnen der Jacobi-Matrix dauert sogar nur das 20-fache der Berechnung des Residuums, obwohl durchschnittlich 37 Nicht-Nulleinträge pro Freiheitsgrad in A zu finden sind.

5.6. Optimierung additiv linear seperabler Funktionale

In der gemischten LSFEM für Elastoplastizität fließt die Spannung nur als linear gewichteter Summand in das Residuum ein. Nur die Verschiebungen gehen nichtlinear in das Funktional ein. In diesem Abschnitt wird gezeigt, wie man dies ausnutzen kann indem bei direkten Lösern ein Teil der Faktorisierung unabhängig von dem nichtlinearen Problem durchgeführt werden kann.

Wir wiederholen zunächst zur Veranschaulichung das Least-Squares-Residuum

$$R(\sigma_h, u_h) = \begin{pmatrix} \sigma(\epsilon(u_h)) - \sigma_h \\ \text{div } \sigma_h \end{pmatrix}.$$

Der einzige nichtlineare Term darin ist $\sigma(\epsilon(u_h))$, der unabhängig von σ_h ist.

In [**51**],[**5**, S. 351] werden Gauss-Newton-Algorithmen für allgemein seperable Funktionale

$$R(y, z) = G(z)y - h(z)$$

behandelt. Dabei lässt sich substituieren

$$y(z) = G(z)^+ h(z),$$

so dass lediglich ein nichtlineares Optimierungsproblem in z zu lösen ist. Ein, in der Literatur noch nicht behandelter, Sonderfall tritt auf wenn, wie bei gemischten Plastizitätsformulierungen, G konstant ist. Es ist dann:

$$R(y, z) = Gy - h(z)$$

Wir folgern, dass man y aus $h(z)$ durch eine lineare Abbildung berechnen kann:

$$y^*(z^*) = G^+ h(z^*) \tag{46}$$

Die führt zu einem Optimierungsproblem allein in z:

$$\min_z \frac{1}{2} \|Gy(z) - h(z)\|_2^2 \tag{47}$$

Nur die Freiheitsgrade z für die Verschiebung u sind also für das nichtlineare Problem wichtig. Dadurch lassen sich fast die Hälfte an Freiheitsgraden eliminieren und

die Gauss-Newton-Iteration (42) vereinfacht sich so zu:

$$z_{k+1} = z_k - J_k^+ R_k$$

mit

$$\begin{aligned} J_k &= (G, h'(z_k)), \\ y_{k+1} &= G^+ h(z_k) = (G^T G)^{-1} G^T h(z_k). \end{aligned}$$

Da das Iterationsverfahren ohne eine Startnäherung für y auskommt, muss nach dem Lösen des nichtlinearen Gleichungssystems nachträglich y^* aus (46) berechnet werden. Im Anwendungsfall der gemischten Plastizität ($\sigma = y$) ist dies dem Patchrecovery-Postprocessing aus [**56**] ähnlich, wo zunächst mit der Galerkin-FEM u_h berechnet wird und anschliessend σ_h durch eine Least-Squares-Näherung gewonnen und verbessert wird. Der wesentliche Unterschied ist, dass dort die Superkonvergenz von u_h an Quadraturpunkten als Annahme vorausgesetzt wurde. Die Superkonvergenz hängt aber von dem verwendeten Gitter ab (siehe [**63**]), bei der gemischten (LS-)FEM ist aber keine Superkonvergenz nötig.

Es fällt auf, dass eine Faktorisierung von $A := G^T G$ (z.B. die Cholesky-Zerlegung $A = R^T R$) für alle Iterationsschritte wieder verwendet werden kann und man erhält:

$$\begin{aligned} y^*(z^*) &= (G^T G)^{-1} G^T h(z) \\ &= A^{-1} G^T h(z) \\ &= R \backslash (R^T \backslash (G^T h(z))) \end{aligned}$$

Einsetzen in (47) liefert dann ein Minimierungproblem allein in z. Dabei ist allerdings zu beachten, dass bei Substitution von ganz σ die Jacobi-Matrix der Gauss-Newton-Iteration leider voll besetzt wird, auch wenn eine Finite-Element-Diskretisierung benutzt wird. Da dies inakzeptabel ist sollte man nicht ganz σ substituieren: Substituiert man nur die Freiheitsgrade von σ die im Inneren eines Elementes liegen ist die Jacobi-Matrix weiterhin dünn besetzt. Man erhält so quasi eine statische Kondensation der inneren Freiheitsgrade der Spannung. Numerische Beispiele brachten bei quadratischen Raviart-Thomas-Elementen jedoch keinen nennenswerten Geschwindigkeitsgewinn, was wohl daran liegt, dass nur relativ wenige Freiheitsgrade (zwei von 14 pro Element) substituiert werden können.

KAPITEL 6

Transfer der Geschichtsdaten

Im Folgenden wird gezeigt, wie herkömmlicherweise der Transfer der Gechichtsdaten vollzogen wurde. An einem numerischen Beispiel wird dann gezeigt, dass dabei entweder ein verhältnismäßig großer Approximations-Fehler entsteht oder der numerische Aufwand groß ist. Danach wird gezeigt wie sich die bisherige Methode der Polynominterpolation auf beliebige Quadraturformeln verallgemeinern lässt und dass diese - zusammen mit geeigneten Gebietszerlegungen nicht nur schnell ist, sondern auch einen fehlerlosen Transfer erlaubt.

Die Verwendung einer Runge-Kutta-Zeitdiskretisierung und anschließende Raumdiskretisierung mit finiten Elementen führt dazu, dass die Raumdiskretisierungen in jedem Zeitschritt unterschiedlich sein können (siehe Beispiel in Abbildung 6.1). Es sind dann Summen (z.B. $v^{\text{old}} + v_h$) von Funktionen zu berechnen, die aus unterschiedlichen (Finite-Element-) Teilräumen von V (z.B. $v^{\text{old}} \in V_{\mathcal{T}^{\text{old}}} \subset V$, $v \in V_{\mathcal{T}} \subset V$) stammen. Die Berechnungen in jedem Zeitschritt müßte daher in einem erweitertem Raum $V_{\mathcal{T}^{\text{old}}} + V_{\mathcal{T}}$ (der bei konformen Elementen durch eine Triangulierung $\mathcal{T} \cup \mathcal{T}^{\text{old}}$ induziert wird) erfolgen, oder man verwendet eine Approximation $\tilde{v}^{\text{old}} \in V_{\mathcal{T}}$ an v^{old} um in der aktuellen Triangulierung $\mathcal{T}$ rechnen zu können. Der Transferfehler $\left\|\tilde{v}^{\text{old}} - v^{\text{old}}\right\|_{0,\Omega}$ führt dann zu einem Approximationsfehler von $\tilde{v}_h$ im aktuellen Zeitschritt. Tritt in jedem Zeitschritt der Länge τ ein Approximationsfehler $\|\tilde{v_h} - v_h\|_{0,\Omega} < C_{\text{app}}$ auf, erhält man beispielsweise für das Euler-Schema am Ende der Zeitreihe eine pessimistische Fehlerschranke von $O(C_{\text{app}}\tau^{-1})$ (Die Fehlerschranken für Rundungsfehler sind allerdings sehr pessimistisch, der stochastische Erwartungswert des Fehlers liegt nach [**13**, S. 75] nur bei $C\tau^{-1/2}$). Da v_h ohnehin einen Approximationsfehler in der Größenordnung $O(h^{k+1})$ beinhaltet, kann man sich mit einem Transfer-Fehler in gleicher Größenordnung begnügen. Wir benötigen also einen Transfer-Operator

$$T_V : V_{\mathcal{T}^{\text{old}}} \to V_{\mathcal{T}}$$

um eine Finite-Element-Funktion von einer Triangulation auf eine andere zu übertragen.

Außerdem wird ein weiterer Transfer-Operator benötigt um allein aus Werten an Quadraturpunkten eine Finite-Element-Funktion zu formen: In den meisten Plastizitätsformulierungen wird der Konsistenzparameter γ nicht als Finite-Element-Funktion angesetzt, da keine Ortsableitungen von ihm benötigt werden.

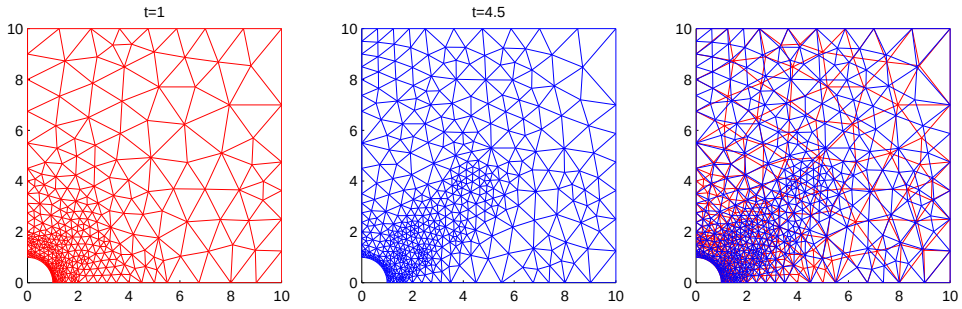


ABBILDUNG 6.1. Zwei gute Triangulierungen für einzelne Zeitschritte und das daraus resultierende Dilemma, dass diese nicht zusammen passen!

Stattdessen wird der Konsistenzparameter nur lokal an jedem Quadraturpunkt berechnet. Da die Materialgeschichte d.h. die internen Materialparameter q von γ abhängen, sind diese dann auch zunächst nur an den Quadraturpunkten bekannt. Solange für jeden Zeitschritt dieselbe Triangulierung verwendet wird, ist dies auch nicht weiter schlimm: Die Quadraturpunkte ändern sich nicht und so kann lokal mit den Werten an den Quadraturpunkten die Linienmethode angewandt werden. Außerdem ist man in der Regel nur an den Spannungen und Verschiebungen des Materials interessiert, der sonstige Materialzustand selbst ist meist uninteressant. Daher kommt man bei konstanter Triangulierung nicht in die Verlegenheit, eine Funktion für q angeben zu müssen. Anders bei adaptiven Verfeinerungen - wie sie in dieser Arbeit benutzt werden. Hier müssen die Werte von q an Quadraturpunkten x_i aus dem alten Gitter $\mathcal{T}^{\text{old}}$ auf Quadraturpunkten $\tilde{\mathsf{x}}_i$ des neuen Gitters $\mathcal{T}$ übertragen werden. Dafür ist ein geeigneter Transfer-Operator

$$T_Q : (q_{\mathcal{T}^{\text{old}}}(\mathsf{x}))_{x \in Q(\mathcal{T}^{\text{old}})} \to (\tilde{q}_{\mathcal{T}}(\tilde{x}))_{x \in Q(\mathcal{T})}$$

nötig. Dabei bezeichne $Q(\mathcal{T})$ die Menge der verwendeten Quadraturpunkte auf der Triangulierung $\mathcal{T}$.

6.1. Herkömmliche Transfer-Operatoren

Verschiedene Transfer-Operatoren wurden z.B. in [**42**] vorgestellt und numerische Vergleiche verschiedener Methoden finden sich in [**29**]. Die benutzten Transfer-Operatoren unterscheiden sich algorithmisch (also vom Zeitaufwand) im Wesentlichen bisher dadurch wie lokal (oder global) sie arbeiten. Hier eine kurze Zusammenfassung bisheriger Methoden:

(1) Eine Möglichkeit für T_Q ist eine Interpolation der Werte an den Quadraturpunkten auf Werte an den neuen Quadraturpunkten. Dabei stellt sich die Frage nach guten Interpolationsfunktionen.
   (a) Konstante Interpolationsfunktionen führen dazu dass ein Quadraturpunkt die Werte des nächstgelegenen Quadraturpunkts des alten Gitter erhält.

(b) Alternativ kann man die Werte mittels natürlicher Koordinaten linear gewichtet aus den umliegenden Quadraturpunkten gewinnen.

(2) Einen anderen Weg gehen Patch-Recovery (z.B. [**65**]) Methoden: Hier wird die Annahme, dass die Lösung stetig ist dahingehend benutzt, dass global auf dem gesamten neuen Gitter (oder auf einem lokalen Patch von mehreren zusammenhängenden Elementen) eine stetige Finite-Element-Bestapproximation der Werte auf dem alten Gitter gesucht wird. Dazu wird global (oder semi-lokal) ein lineares Least-Squares-Problem

$$\min_{v_{\mathcal{T}} \in V_{\mathcal{T}}} \sum_{x \in Q(\mathcal{T})} \|v_{\mathcal{T}^{\text{old}}}(x) - v_{\mathcal{T}}(x)\|_2^2 \tag{48}$$

gelöst, was einen globalen Transfer-Operator $G^{\mathcal{T}} : V_{\mathcal{T}^{\text{old}}} \to V_{\mathcal{T}}$ definiert. Um aus diesem Transferoperator einen Transferoperator T_Q zu machen ist vorweg aus den Werten an den Quadraturpunkten eine Funktion $v_{\mathcal{T}^{\text{old}}}$ zu konstruieren und abschließend wird die Finite-Element-Funktion an den neuen Quadraturpunkten (durch Interpolation) ausgewertet.

(3) Ein anderer Ansatz zerlegt T_Q sogar in vier Teilschritte:

(a) Die Werte der Quadraturpunkte werden zunächst auf die Knotenpunkte der alten Triangulierung übertragen (siehe Abschnitt 6.2 auf der nächsten Seite). Dadurch werden an den Knotenpunkte mehrere Werte pro Punkt erzeugt (für jedes anliegendes Element einer).

(b) Die Werte an jedem Punkt auf den Elementkanten werten gemittelt, wodurch eine eindeutige lineare konforme Finite-Element-Funktion $\tilde{q}$ entsteht. Dieser Schritt wird Glättung $S : V_{\mathcal{T}^{\text{old}}} \to V_{\mathcal{T}^{\text{old}}}$ genannt.

(c) Diese stetige Finite-Element-Funktion wird an den Knotenpunkten der neuen Triangulierung (durch Polynominterpolation $Ip^{\mathcal{T}} : V_{\mathcal{T}^{\text{old}}} \to V_{\mathcal{T}}$) ausgewertet.

(d) Diese Finite-Element-Funktion wird an den neuen Quadraturpunkten (durch Interpolation) ausgewertet.

Eine Vergleich der Methoden ergibt, dass Methode 1a) eine sehr lokale (und somit schnelle) und leicht implementierbare Methode ist, die allerdings großen Approximationsfehlern unterliegen muss, da die Interpolationsfunktion nicht stetig ist. Methode 1b) liefert eine glatte Interpolation allerdings ist die Berechnung natürlicher Koordinaten verhältnismäßig aufwendig, da sie gebrochen rationale Funktionen sind. Noch aufwendiger ist Methode 2 wo ein globales Gleichungssystem gelöst werden muss. Methode 3 hingegen arbeitet elementweise (also schnell). Eine schöne Eigenschaft der reinen Interpolation der Methode 1 geht allerdings durch die Glättung in Teilschritt b) verloren: Bereits der Transfer zwischen zwei identischen Triangulierungen (bzw. auch dort wo sich lokal ein Netz nicht verändert hat) ist nicht mehr die Identität! Auch wenn man also alle Statusvariablen an den Quadraturpunken auf sich selbst transferiert, ist bereits das physikalische Gleichgewicht gestört.

Auch beim Patch-Recovery zwischen zwei unterschiedlichen Triangulierungen ist das Gleichgewicht nach dem Transfer in den neuen Quadraturpunkten im Allgemeinen nicht gegeben. Als Abhilfe gegen gestörte Gleichgewichte lösen Ingenieure auf dem neuen Gitter zunächst erneut das alte nichtlineare Gleichgewichtsproblem um das Gleichgewicht wiederherzustellen. Dies ist so aufwendig wie das Lösen auf dem alten Gitter (bis auf das bereits einen guter Startwert vorliegt), der gesamte Rechenaufwand verdoppelt sich dadurch fast.

Die Approximation einer H^r-Funktion bringt nach (39) einen maximalen Fehler in der Größenordnung

$$\left\|\tilde{v}^{\text{old}} - v^{\text{old}}\right\|_{0,\Omega} = O(h^r)|v^{\text{old}}|_r.$$

Wir zeigen an einem numerischen Beispiel, dass der Approximationsfehler in diesem Anwendungsfall asymptotisch geringer aber signifikant ist: Wir approximieren dazu eine Finite-Element-Funktion mit einer Finiten-Element-Funktion auf einer anderen Zerlegung. Als Beispiel wählen wir zwei Triangulierungen mit in etwa gleicher allgemeiner Approximationsgüte h (Abbildung 6.2). Auf einer der beiden Triangulierungen approximieren wir die Funktion

$$u(\mathsf{x}, y) := \frac{1}{100}(\frac{1}{3}\mathsf{x}^3 + y^2\mathsf{x})$$

mit einer Finiten-Element-Funktion u_h aus quadratischen standard-konformen Elementen. Dann bilden wir $v^{\text{old}} := \frac{\partial}{\partial x}u$ was eine nicht stetige Finite-Element-Approximation von

$$v(\mathsf{x}, y) := \frac{1}{100}(\mathsf{x}^2 + y^2)$$

liefert. In Abbildung 6.3 sind die Approximationsfehler dargestellt: Der Approximationsfehler $\left\|v^{\text{old}} - v\right\|_{0,\Omega}$ kann sowohl durch Glättung S als auch durch das Lösen eines globalen Ausgleichsproblems G vermindert werden. Beim Übertrag auf die andere Triangulierung durch Polynominterpolation Ip^{new} kommt es danach aber zu einem Approximationsfehler, der einer Approximation mit halber Gitterweite entspricht - unabhängig davon, ob vorab geglättet wurde oder nicht.

Während das Lösen eines globalen Ausgleichsproblems also einen qualitativ wesentlich besseren Variablentransfer ermöglicht als die Polynominterpolation ist die Polynominterpolation wesentlich schneller.

Daher wird im Folgenden versucht die Transfergüte der Polynominterpolation durch geeignete Raum-Zerlegungen zu verbessern und für allgemeine Quadraturformeln zu ermöglichen.

6.2. Eine neue Idee: Interpolatorische Nullgewichts-Quadraturformeln

Um einen besseren Transferoperator zu erhalten betrachten wir zunächst wie - und wann - die Extrapolation in Methode 3 Schritt a) bisher funktioniert und verallgemeinern dann das Verfahren durch Hinzuname weiterer nullgewichteter Quadraturpunkte auf beliebige Quadraturformeln:

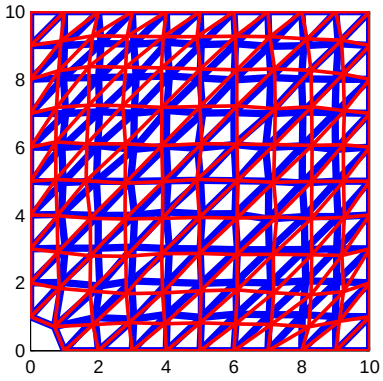


ABBILDUNG 6.2. Zwei Test-Triangulierungen zwischen denen ein exemplarischer Variablentransfer vermessen wird

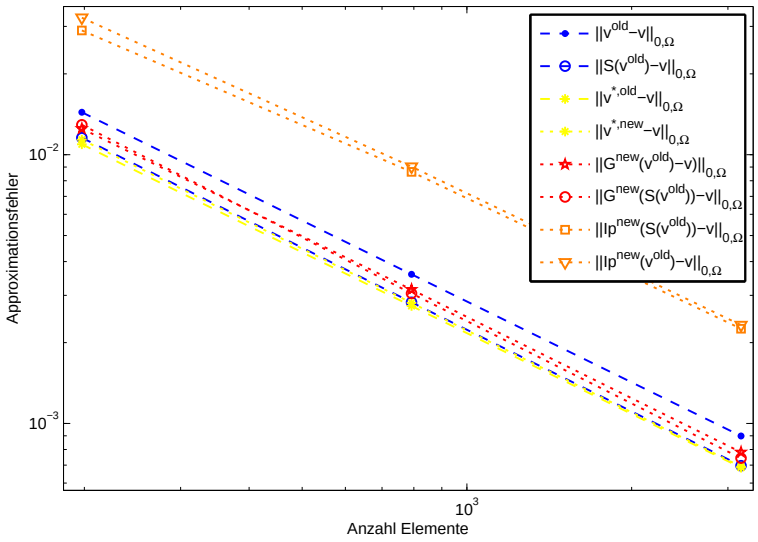


ABBILDUNG 6.3. Approximationsfehler von verschiedenen Transferoperatoren und ihren Zwischenschritten. Zum Vergleich sind in gelb die minimal möglichen Approximationsfehler auf den beiden Triangulierungen eingetragen.

Bei üblichen 1-Punkt-Quadraturformeln (die bei linearen Elementen für u_h benutzt werden) werden konstante Funktionen im Element angenommen und daher die Werte an den Knotenpunkten gleich den Werten an dem Quadraturpunkt gesetzt. Bei 4-Punkt- bzw. 3-Punkt-Quadraturformeln für (bi-)quadratische Polynome über Vierecken bzw. Dreiecken ist die Extrapolation durch eine (bi-)lineare Abbildung möglich: dazu wird eine (bi-)lineare Funktion durch die alten Quadraturpunkte eines Elementes gelegt.

Zu beliebigen vorgegebenen Quadraturformeln vom Grad d_Q wird man im Allgemeinen jedoch kein eindeutiges Interpolationspolynom vom Grad d_I finden. Selbst bei optimalen Quadraturformeln (mit minimaler Anzahl Quadraturpunkte zu fest vorgegebenem Grad d_Q) ist die Anzahl der Quadraturpunkte ungleich der Anzahl

der Freiheitsgrade eines Polynoms vom Grad d_I. Als Beispiel sei der 7-Punkt-Stern vom Grad 5 über dem Dreieck (Abbildung 5.5 auf Seite 56 rechts) genannt: Polynome in zwei Veränderlichen vom Grad 3 haben 6 und Polynome vom Grad 4 haben 10 Freiheitsgrade, aber kein vollständiges Polynom hat genau 7 Freiheitsgrade. Man könnte auf die Idee kommen solche Quadraturformeln zu meiden, aber - anders als bei Gauss-Quadraturformeln im Zweidimensionalen - ist für mehrdimensionale Quadratur keine Familie von Gauss-Quadraturformeln bekannt.

Trotzdem wollen wir Teilschritt a) für beliebige Quadraturformeln verallgemeinern: Wir fügen zusätzliche Quadraturpunkte mit Nullgewichten ein, mit denen die Interpolationsaufgabe eindeutig wird. Im Folgenden seien dabei Dreieckselemente angenommen.

Interpolationsformeln vom Grad k sind eine Menge von Punkten, so dass man genau ein Polynom vom Grad k durch skalare Werte an diesen Punkten legen kann. Beispiele dafür sind die ersten drei Quadraturformeln in Abbildung 5.5 auf Seite 56. Die Lagrange-Punkte vollständiger finiter Elemente (Abbildung 2.1 links) sind ebenfalls Interpolationsformeln. Jede Interpolationsformel induziert automatisch eine Quadraturformel vom Grad $d \geq k$. Ihre Gewichte erhält man durch das Lösen eines linearen Gleichungssystems. Allgemein heißt eine Quadraturformel interpolatorisch, wenn ihre Gewichte eindeutig durch die Quadraturpunkte bestimmt sind. Die ersten drei Beispiele in Abbildung 5.5 zeigen, dass der Quadraturgrad doppelt so groß sein kann wie der Interpolationsgrad k. Eine Interpolationsformel im Zweidimensionalen hat genau $N = \frac{1}{2}(k+2)(k+1)$ Punkte.

Ein weiteres Beispiel ist damit die 15-Punkt-Quadraturformel vom Grad $d = 8$ (Sie ist allerdings leider nur rotationssymmetrisch und hat Punkte außerhalb des Dreiecks). Sie ist interpolatorisch und ihre Punkte bilden eine Interpolationsformel vom Grad $k = 4$. Auch hier ist der Quadraturgrad doppelt so groß wie der Interpolationsgrad. Das legt die Vermutung nah, dass es eine Familie von N_k^d-Quadraturformeln mit $d = 2k$ gibt. Es wurden auch schon mehrere 10_3^6-Quadraturformel gefunden aber bisher keine 21_5^{10}-, 28_6^{12}-, oder 36_7^{14}- ... Quadraturformeln. Die Rekorde für Quadraturformeln vom Grad 10, 12 und 14 liegen bisher bei dem Minimum von 22, 33 und 42 Punkten ([**23**]).

Braucht man eine Interpolationsformel die gleichzeitig eine gute Quadraturformel ist weiß man also im Allgemeinen noch nicht, welche Quadraturformel dafür am besten ist. Eine gute Quadraturformel in unserem Sinne hat möglichst wenige Quadraturpunkte mit ausschließlich positiven Gewichten. Um eine stetige Interpolation über einer Triangulierung zu haben (beziehungsweise eine Glättung lokal durchführen zu können) wünscht man sich außerdem Interpolationsformeln mit $k+1$ Punkten auf jeder Kante.

Die Idee ist nun eine gute Quadraturformel zu einer guten Interpolationsformel zu erweitern. Zusätzliche Interpolationspunkte können mit Null gewichtet zu der Quadraturformel hinzugenommen worden, so dass man interpolatorische Nullgewichts-Quadraturformeln erhält.

Angenommen man benutzt eine beliebige Quadraturformel vom Grad d und braucht dazu ein Interpolationsformel vom Grad k wie kann man dann die Punkte der Quadraturformel um eine Menge von Punkten erweitern um die Interpolationsaufgabe eindeutig zu machen? Dies geht bei interpolatorischen Quadraturformeln immer, wenn $k \geq d$ ist und bei manchen Quadraturformeln (Beispiele s.o) auch mit $k \geq \frac{1}{2}d$:

Sind die M Quadraturpunkte gegeben durch

$$\begin{pmatrix} \mathsf{x}_i \\ y_i \end{pmatrix}_{i=1..M}$$

dann nehme man einfach eine beliebige Interpolationsformel vom Grad k mit $N = \frac{1}{2}(k+2)(k+1)$ Punkten

$$\begin{pmatrix} \mathsf{x}_i \\ y_i \end{pmatrix}_{i=M+1,M+2,..,M+(N+1)}$$

dazu und stelle die Matrix A mit den Monom-Zeilen

$$a_i = \left[\mathsf{x}_i^l y_i^m\right]_{0 \leq l+m \leq k}$$

auf. Die ersten N linear unabhängigen Zeilen von A mit den Indizes I liefern dann eine Interpolationsformel $(\mathsf{x}, y)_{i \in I}$, denn diese Zeilen können die Werte einer Funktion v an den dazugehörigen Punkten auf die Koeffizienten α_{lm} der Monome abbilden:

$$[\alpha_{lm}]_{0 \leq l+m \leq k} = A^{-1} \left[v \begin{pmatrix} \mathsf{x}_i \\ y_i \end{pmatrix} \right]_{i \in I}$$

Um eine möglichst glatte Interpolation über einer Triangulierung zu erzielen kann man (für $k > 0$) eine Interpolationsformel mit jeweils $k+1$ Punkten auf jeder Kante nehmen und diese Punkte mit den niedrigsten Indizes innerhalb der Interpolationsformel versehen bevor man obiges Verfahren verwendet. Dadurch erhält man eine interpolatorische Nullgewichts-Quadraturformel mit maximal vielen Punkten auf den Kanten - allerdings nicht zwangsläufig mit $k+1$ Punkten auf jeder Kante.

Die $M_k^d B_K^D O$ -Beispielformeln in Abbildung 6.4 auf der nächsten Seite zeigen Quadraturformeln vom Grad d mit M Quadraturpunkten die zu Interpolationsformeln vom Grad $k \leq d$ erweitert wurden. Sie haben B Punkte auf dem Rand und dort Quadraturgrad D und Interpolationsgrad K. Um $D > K$ zu erreichen wurden auf dem Rand Lobatto-Quadraturpunkte als Interpolationspunkte ergänzt. Manche Formeln haben O Punkte außerhalb des Dreiecks die wegen ihrer winzigen Gewichte mit einem Pluszeichen markiert wurden um sie zu visualisieren. Die minimale Quadraturformel für $d = 3$ hat einen negativ gewichteten Quadraturpunkt. Es zeigt sich, dass auch für $k = d$ manchmal nicht $D = d$ (und nicht einmal $K = d$) erreicht werden kann. Um eine stetige Polynominterpolation vom Quadraturgrad über einem Gitter erreichen zu können kann also nicht jede beliebige Quadraturformel benutzt werden.

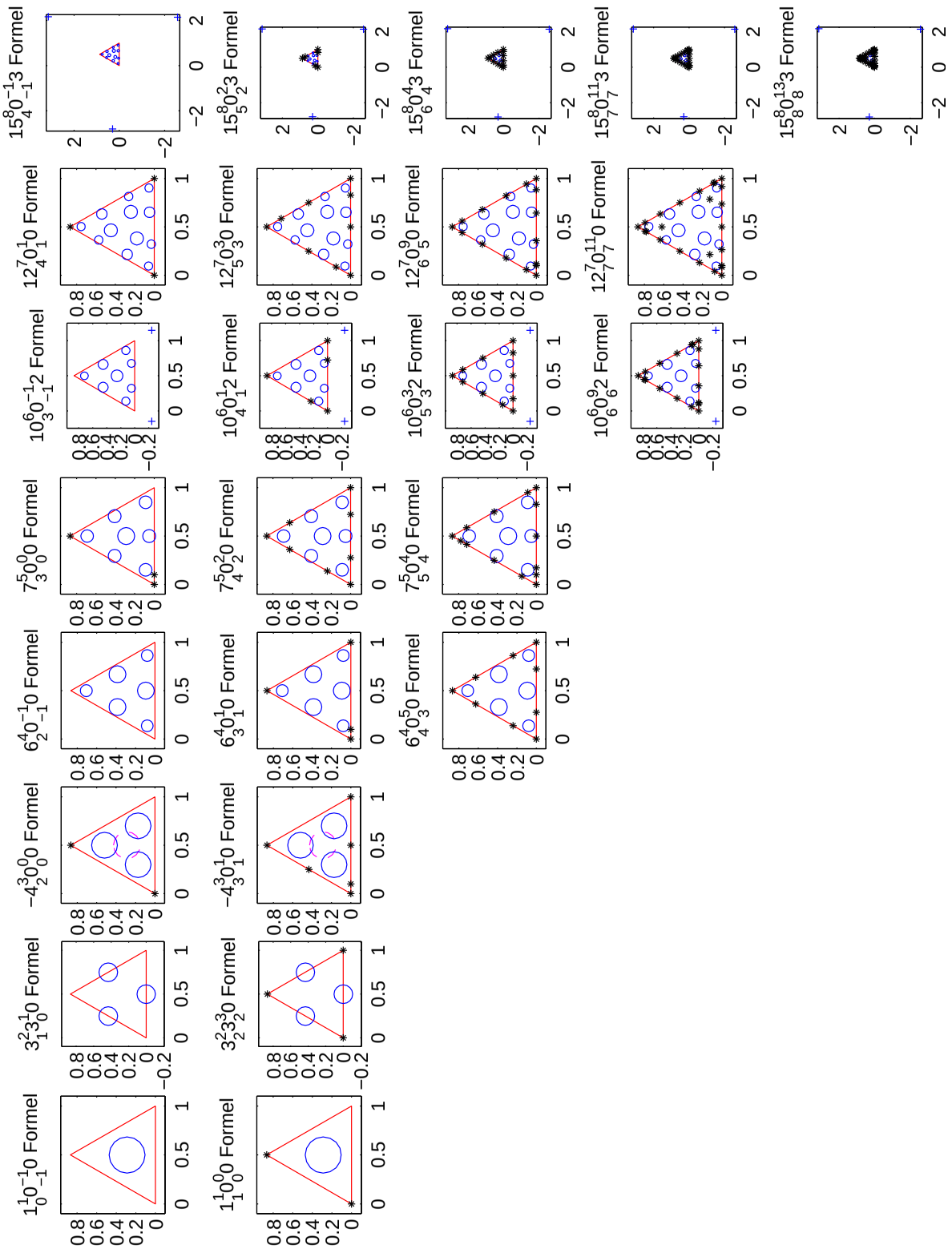


ABBILDUNG 6.4. Beispiele interpolatorischer Nullgewichts-Quadraturformeln. Schwarze Sternchen markieren die ergänzten Interpolationspunkte.

6.3. Ein neues Verfahren zum Transfer der Geschichtsdaten

Im vorherigen Abschnitt wurde gezeigt, dass durch Hinzunahme von nullgewichteten Quadraturpunkten aus jeder Quadraturformel eine interpolatorische (Nullgewichts-) Quadraturformeln erzeugt werden, so dass die Interpolationsaufgabe eindeutig lösbar wird. Die Frage liegt auf der Hand wie sich zusätzliche (nullgewichtete) Quadraturpunkte auf die Rechenzeit auswirken und andererseits wieso nullgewichtete Quadraturpunkte überhaupt etwas verändern können:

Zunächst sind nullgewichtete Quadraturpunkte für das Lösen des nichtlinearen Gleichungssystems unnötig: In der Galerkin-FEM werden die Werte an diesen Punkten durch die Nullgewichtung ignoriert und in der LSFEM werden die korrespondierenden Zeilen im Residuum ebenfalls mit Null gewichtet - also ignoriert. Da man beim Lösen des globalen Gleichungssystems also auf die nullgewichtete Quadraturpunkte verzichten kann erhöhen sie den Rechenaufwand nicht. Wählt man allerdings statt optimalen Quadraturformeln absichtlich Quadraturformeln, die bereits Interpolationsformeln sind (z.B. die nodalen Basispunkte als Quadraturpunkte) rechnet man mit bis zu dreimal so vielen nicht-nullgewichteten Quadraturpunkten. Dasselbe gilt, wenn man bewusst gute, aber (in gewissem Maße) auch gut zu Interpolationsformeln vervollständigende Quadraturformeln wählt: Man muss mit mehr Quadraturpunkten rechnen. Mehr Quadraturpunkte bedeuten einen Mehraufwand beim Aufstellen der Systemmatrizen. Das Lösen der linearen Probleme hingegen ist in der Galerkin-FEM davon nicht betroffen, da die Größe der Matrizen allein durch die Anzahl der Testfunktionen vorgeben wird und auch in der LSFEM wird das lineare Gleichungssystem (bei Verwendung der Normalen-Gleichung) allein durch die Anzahl der Unbekannten dimensioniert. Dauert das Lösen des globalen Gleichungssystems viel länger als das Aufstellen der Systemmatrizen ist der Mehraufwand durch nicht-optimale Quadraturformeln eher gering.

Sind die Unbekannten des globalen nichtlinearen Gleichungssystems erst einmal berechnet, lassen sich nachträglich die lokalen Werte an den Quadraturpunkten bestimmen. Der Aufwand dafür ist sehr gering im Vergleich zum Aufstellen der Systemmatrizen, da nun keine partiellen Ableitungen mehr berechnet werden müssen. Der Mehraufwand für zusätzliche nullgewichtete Quadraturformeln ist dementsprechend vernachlässigbar. Die Werte an den nullgewichteten Quadraturpunkten lassen sich nutzen um die Extrapolationsaufgabe in dem dritten Transfer-Operator zu lösen. Dabei spielen Quadraturgewichte keine Rolle mehr. Auch bei der Least-Squares-Approximation des Patch-Recovery (48) gibt es keine Unterscheidung zwischen normalen und nullgewichteten Quadraturpunkten.

In den numerischen Rechnungen dieser Arbeit wurde eine Modifikation des dritten Transfer-Operators benutzt. Während der dritte Transfer-Operator eine lokale Glättung vorschreibt, wurden die Werte an den Quadraturpunkten in der Triangulation des nächsten Zeitschrittes noch einfacher bestimmt: Es werden elementweise Interpolationspolynome durch die Werte an den (teilweise nullgewichteten) Quadraturpunkten der Triangulation des alten Zeitschrittes gelegt und an den neuen

Quadraturpunkten ausgewertet. Der Approximationsfehler bei dem Transfer zwischen zwei inkompatiblen Triangulierungen, der durch den Wegfall der Glättung entsteht, ist dabei unerheblich (siehe Abbildung 6.3 auf Seite 73).

Verwendet man nun zusätzlich kompatible Triangulierungen (Abschnitt 7), dann ist die Lösung der Interpolationsaufgabe bereits eine Lösung von

$$\min_{v_{\mathcal{T}} \in V_{\mathcal{T}}} \|v_{\mathcal{T}} - v_{\mathcal{T}^{\text{old}}}\|_{0,\Omega}$$

wenn $\mathcal{T}$ feiner als $\mathcal{T}^{\text{old}}$ ist, da dann wegen $V_{\mathcal{T}^{\text{old}}} \subset V_{\mathcal{T}}$ mit der Identität die Bestapproximation erreicht wird. Dies wäre nicht der Fall, wenn man den üblichen Glättungsschritt beibehalten würde. Ist hingegen $\mathcal{T}^{\text{old}}$ feiner als $\mathcal{T}$ (hat also eine Vergröberung stattgefunden) liefert

$$\min_{v \in V_{\mathcal{T}}} \|v_{\mathcal{T}} - v_{\mathcal{T}^{\text{old}}}\|_{0,\Omega}^2 = \min_{v_{\mathcal{T}} \in V_{\mathcal{T}}} \sum_{(w,\mathsf{x}) \in QP(\mathcal{T} \cup \mathcal{T}^{\text{old}})} w \, \|v_{\mathcal{T}}(\mathsf{x}) - v_{\mathcal{T}^{\text{old}}}(\mathsf{x})\|^2 \tag{49}$$

ein globales lineares Ausgleichsproblem (Dabei bezeichnet QP die Menge der Paare von Quadraturgewichten mit Quadraturpunkten). Dieses kann jedoch für jedes Element in $\mathcal{T}$ separat (lokal) gelöst werden, wenn man den Finite-Element-Raum $V_{\mathcal{T}} = \{v : \, v|_T \in \mathrm{P_k} \; \forall T \in \mathcal{T}\}$ ohne Übergangsbedingungen zwischen den Elementen wählt. Andererseits könnte es passieren, dass im nächsten Schritt dort wieder verfeinert wird wo im letzten Schritt vergröbert wurde. Dann hätte man dort durch das Lösen von (49) lokal Information verloren. Stattdessen ist es möglich auf das explizite Lösen von (49) zu verzichten und stattdessen mit einem $v_{\mathcal{T}^+} \in V_{\mathcal{T} \cup \mathcal{T}^{\text{old}}}$ zu arbeiten, die Statusvariablen also über einer Quadraturtriangulierung $\mathcal{T}^+ := \mathcal{T} \cup \mathcal{T}^{\text{old}}$ zu definieren und diese von Zeitschritt zu Zeitschritt zu verfeinern. Man erhält dann auch im Fall, dass $\mathcal{T}^{\text{old}}$ feiner als $\mathcal{T}$ ist, keinen Approximationsfehler.

Quadraturtriangulierungen können auch zwischen inkompatiblen Triangulierungen mit Hilfe einer Delaunay-Zerlegung berechnet werden, aber dies ist nicht nur aufwendig (da alle Schnittpunkte der Kanten gefunden werden müssten) sondern ist im Allgemeinen eine Zerlegung, die viel mehr Elemente hat als das neue oder alte Gitter alleine. Berechnet man das Integral in (49) nur durch Quadratur auf dem neuen Gitter (statt auf $\mathcal{T}^+$) erhält man einen zusätzlichen Quadraturfehler (siehe [**30**, 5.4]).

Zum exakten Lösen des globalen Gleichungssystems ist also über der Quadraturtriangulierung zu integrieren (die Ansatzfunktionen für u und σ werden jedoch weiter durch $\mathcal{T}$ induziert). Dies erfordert zusätzliche Quadraturpunkte, die einen Mehraufwand zum Aufstellen der Jacobi-Matrix kosten. Im Plastizitäts-Benchmark hat sich gezeigt, dass die so erzeugte Quadraturtriangulierung am Ende der Belastung jedoch gerade mal etwa doppelt so viele Elemente aufweist wie die Triangulierung der Unbekannten (Abbildung 6.5), da sich die Lokalität der Lösung nicht stark ändert. Bei anderen Anwendungen z.B. Wellendynamik, wo sich die Lokalität (durch fortschreitende Wellenfronten) mit der Zeit stark ändert, ist dies allerdings nicht

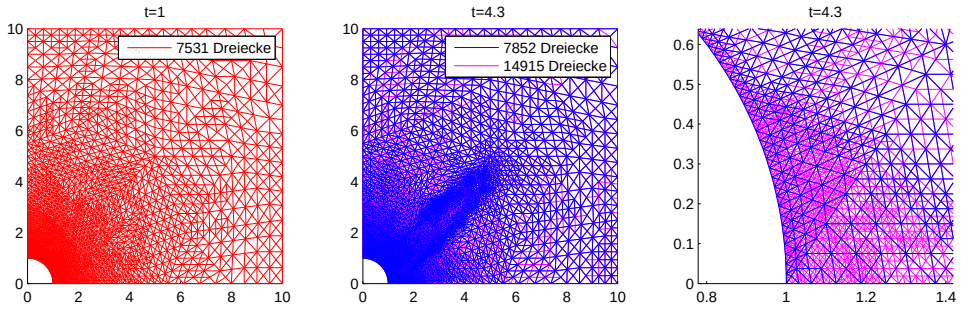


ABBILDUNG 6.5. Eine Quadraturtriangulierung (lila) bei kompatiblen Triangulierungen. In rot eine FEM-Triangulierung im ersten Zeitschritt, in blau ein spätere Triangulierung. In der Vergrößerung ist die Entfeinerung zugunsten der Verfeinerung des plastischen Randes zu erkennen.

zu empfehlen. Als Kompromiss kann man die Quadraturtriangulierung vergröbern, wenn die Anzahl der Quadratur-Elemente eine bestimmte Grenze überschreitet.

Es wurde also gezeigt, dass - mit Einführung einer speziellen Quadraturtriangulierung, und unter Annahme dass $\mathcal{T}$ lokal eine Verfeinerung von $\mathcal{T}^{\text{old}}$ (oder andersrum) ist - eine elementweise Interpolation einen verschwindenden Approximationsfehler

$$\left\| v_{\mathcal{T}^+} - v_{\mathcal{T}^{\text{old}}} \right\|_{0,\Omega} = 0$$

liefert. Bleibt noch zu zeigen, wie solche Triangulationen konstruiert werden können, was Thema des folgenden Kapitels ist.

KAPITEL 7

Adaptivität: Problemangepasste Triangulierungen

In diesem Kapitel wird gezeigt wie ausgehend von einer groben Triangulation $\mathcal{T}_H$, Triangulationen schnell adaptiv konstruiert werden können, so dass in einer Zeitreihe mit Zeitschritten $i \in I$ alle auftretenden Triangulationen $\mathcal{T}_i$ zwei Bedingungen erfüllen:

(1) Alle Triangulationen $\mathcal{T}_i$ sind Verfeinerungen der groben Triangulation $\mathcal{T}_H$:

$$\mathcal{T}_H \subset \mathcal{T}_i$$

(2) Für alle $i, j \in I$ ist entweder T_i lokal eine Verfeinerung von $\mathcal{T}_j$ oder andersrum.

Der Grund für Bedingung 2 ist der im vorherigen Kapitel vorgestellte Transferoperator. Bedingung 1 induziert zusammen mit konformen Elementen geschachtelte Finite-Element-Räume, die zwei Vorteile aufweisen: Zum einen hat man in der h-adaptiven FEM bereits eine Lösung u_H auf dem gröberen Gitter berechnet und kann diese dann durch eine interpolierende Prolongation $u_h^p = I_H^h(u_H)$ als Startwert für die iterative Berechnung des nichtlinearen Lösers verwenden (eine lineare Abbildung) und zum anderen kann man die transponierte davon als Restriktion $I_h^H(u_h) = (I_H^h)^T u_h$ in einem Mehrgitterverfahren verwenden. Bei nicht geschachtelten Räumen sind zwar auch Prolongationen und Restriktionen definierbar, aber nicht so gute.

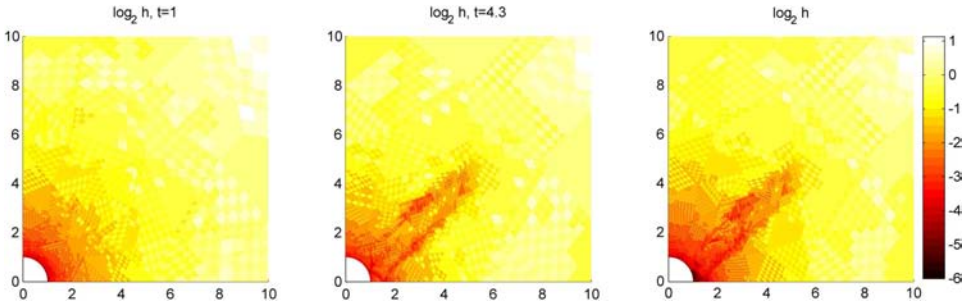


ABBILDUNG 7.1. Die lokale Gitterweite h zu unterschiedlichen Zeitpunkten und rechts von der dabei entstandenen Quadraturtriangulierung.

Bisher findet sich in der Literatur kein Hinweis darauf, dass jemand den ernsthaften Versuch unternommen hat die Raumzerlegungen so zu wählen. (Lediglich wurde probiert das aktuelle Gitter als Verfeinerung des alten Gitters zu wählen

und mit diesem verfeinerten Gitter alle Berechnungen vom ersten Zeitschritt an zu wiederholen - siehe z.B [**62**, S. 284]).

7.1. isotrope Verfeinerung

Die adaptive h-FEM verlangt nach lokalen Verfeinerungen, also studieren wir kurz welche Methoden es dafür gibt und welche in unserer Anwendung geeignet sind:

Eine feinere Zerlegung kann erzeugt werden, indem durch einen Fehlerschätzer eine Dichtefunktion $\bar{h}(\times) : \Omega \to \mathbb{R}$ aufgestellt wird, welche den gewünschten maximalen Elementdurchmesser beschreibt. Dann kann man zufällig eine entsprechend dichte Punktmenge mit Punkten aus Ω wählen und diese nach Delaunay triangulieren. In einem zweiten Schritt kann dann die lokale Qualität des Gitters verbessert werden, indem z.B. der größte Inenwinkel ([**3**]) aller Dreiecke durch eine Verschiebung der Punkte minimiert wird. Andere lokale Qualitätsmaße wurden vorgeschlagen und in [**55**] miteinander verglichen. Die daraus entstehenden Zerlegungen haben sehr gute allgemeine Approximationsgüten, aber einen entscheidenden Nachteil: im Allgemeinen sind die dadurch erzeugten Finite-Element-Räume nicht ineinander geschachtelt.

$$U_{\mathcal{T}_H} \not\subseteq U_{\mathcal{T}_h}$$

Konforme Verfeinerungen können nur konstruiert werden, indem neue Kanten innerhalb eines Dreieckes eingefügt werden, oder auch Kanten zerteilt werden. Nur Kanten innerhalb eines Dreieckes einzufügen ohne die Seiten zu unterteilen verbessert jedoch den Durchmesser nicht wesentlich, dieser konvergiert dann höchstens gegen eine Seitenbreite. Daher ist es sinnvoll die Kanten zu zerteilen, also neue Punkte auf den Kantenmittelpunkten einzufügen. Bleibt nur die Wahl, welche Seiten alles verfeinert werden sollen und wie die neuen Punkte miteinander verbunden werden. Dazu gibt es bisher vier unterschiedliche Algorithmen:

(1) Verfeinerung der längsten Seite (*longest side*) (LS) nach [**48**]
(2) Je nachdem, welcher Punkt des Dreiecks als letztes zur Triangulierung dazu kam, wird die gegenüberliegende Seite zerteilt (*newest vertext bisection*) (NVB) nach [**39**].
(3) Einführen von neuen Knotenpunkten auf allen drei Kantenmittelpunkten. Der Knoten auf der längsten Seite wird mit der gegenüberliegenden Ecke und den anderen beiden neuen Knoten über neue Kanten verbunden. (*four-triangles longest-side*) (4TLS) nach [**50**] (in der Literatur auch manchmal *4T-LE* genannt).
(4) Einführen von neuen Knotenpunkten auf allen drei Kantenmittelpunkten, die nur untereinander durch neue Kanten verbunden werden (reguläre Verfeinerung). Die Innenwinkel ändern sich dabei nicht.

Es ist bekannt, dass nach einer endlichen Anzahl von Verfeinerungen die LS zur NVB übergeht. Danach sind zwei aufeinanderfolgene Verfeinerungen aller Dreiecke auch gleichzeitig eine einzelner Verfeinerung der 4TLS-Verfeinerung.

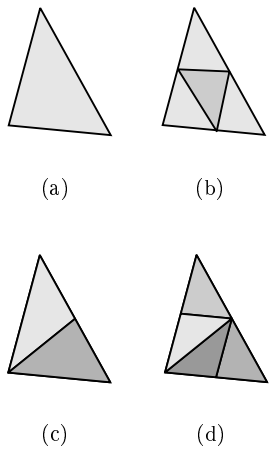


ABBILDUNG 7.2. Dreiecksverfeinerungen von Ausgangsdreick (a), reguläre Verfeinerung (b), LS- (c) und 4TLS-Verfeinerung (d).

Optimal bezüglich der Approximationsgüte h wäre eine Delaunay-Triangulierung der drei Seitenmittelpunkte und der drei Ecken eines einzelnen Dreiecks. Allerdings ist das nicht aller Weisheit Ende: Es ist nämlich auch bekannt, dass lineare Elemente auf Triangulationen mit Kanten in nur drei statt vier Richtungen bessere Approximationseigenschaften haben. Ausschließlich reguläre Verfeinerungen können diese Eigenschaft erhalten. Bisherige Fehlermaße berücksichtigen dies jedoch nicht.

Reguläre Unterteilungen eines Dreiecks - in drei gleiche Dreiecke - zeichnen sich außerdem dadurch aus, dass alle entstehenden Dreiecke die gleiche Form wie das grobe Dreieck haben. Das kann man algorithmisch nutzen um die Aufstellung der Systemmatrizen zu beschleunigen, genauso wie bei strukturierten Gittern. Auch für die anderen Verfeinerungsstrategien kann gezeigt werden, dass es nur beschränkt viel unterschiedliche Dreiecksformen geben kann unabhängig davon wie oft verfeinert wird ([**49**]). Um dies auszunutzen müsste aber getestet werden, welche - der endlich vielen Formen - das jeweilige Dreieck hat.

Unterteilung in nur zwei Dreiecke haben den Vorteil, dass sie mehr lokale Adaptivität bei der letzten Verfeinerung bieten, ansonsten ist im Zweidimensionalen generell die reguläre Verfeinerung am besten, da das Grobgitter hinreichend gut gewählt werden kann. Im Dreidimensionalen gibt es jedoch kein äquivalent zur regulären Verfeinerung, während sich die anderen Verfeinerungen auch auf höherer Dimensionen übertragen lassen.

Bis hierher wurde nur die Unterteilung eines einzelnen Dreiecks behandelt. Um eine zulässige feine Triangulation zu erhalten ist es noch notwendig hängende Knoten zu beheben (oder die Berechnungen für Triangulationen mit hängenden Knoten zu implementieren).

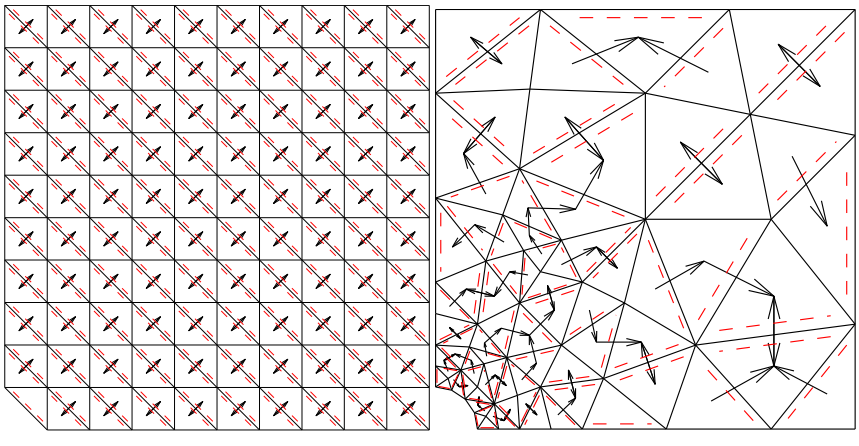
Hängende Knoten entstehen, wenn eine innere Kante von nur einem anliegenden Dreieck unterteilt wird. Die Freiheitsgrade an dieser Kante können dann durch die

Funktionswerte des benachbarten nicht-verfeinerten Elements substituiert werden. Es gibt bisher keine ausreichenden Konvergenzabschätzungen für diese Verfeinerungsstrategien. Empirisch haben sich jedoch (vor allem bei Vierecks-Elementen, wo dies die einzige mögliche nicht entartende lokale Verfeinerungsstrategie ist) hängende Knoten als ausreichend erwiesen, wenn höchstens ein hängender Knoten pro Element vorkommt. Dies kann erzwungen werden, indem sukzessiv auch benachbarte Elemente verfeinert werden, wenn diese Bedingung verletzt wird.

Eine zulässige Triangulierung (ohne hängende Knoten) wird erreicht, wenn alle Dreiecke mit hängenden Knoten trianguliert werden. Dabei werden allerdings mitunter in diesen Dreiecken nur nicht-längste Seiten zerteilt. Daher LS-verfeinert man diese Dreiecke normalerweise und dieser Prozess wird solange fortgesetzt bis es keine hängenden Knoten mehr gibt. Die Verfeinerung eines einzelnen Elementes kann daher die Verfeinerung vieler anderer Elemente nach sich ziehen. Welche Dreiecke davon alles betroffen sind beschreibt nach [**49**] der Ausbreitungspfad (siehe Abbildung 7.3)

$$P(T) = \{\bar{T} \in \mathcal{T},\ \bar{T} \text{ muss mitverfeinert werden, bei Verfeinerung von } T\}.$$

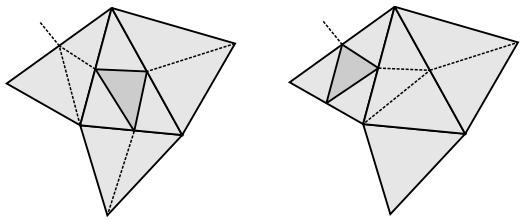
In [**60**] wurde gezeigt, das bei gleichmäßiger 4TLS-Verfeinerung $|P|$ asymptotisch klein wird, was aber bei adaptiver Verfeinerung nicht seien muss (und auch nicht gilt siehe Abbildung 7.10). Diese Art der Verfeinerung ist in der Matlab PDE-Toolbox implementiert. Die dabei entstehenden Gitter haben eine sehr unregelmäßige Struktur, haben aber nach unten beschränkte Innenwinkel, sind konform im Sinne von $\mathcal{T}_h \subseteq \mathcal{T}_H$ und man erhält so eine Serie von hierarchischen zulässigen Triangulierungen.



(a) (b)

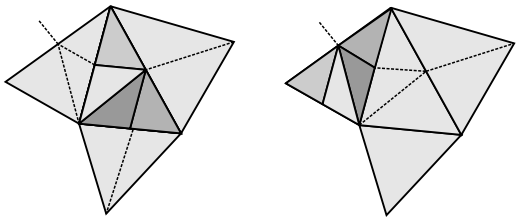
ABBILDUNG 7.3. Der Ausbreitungsgspfad in Triangulierungen. Die gestrichelten roten Linien markieren die längsten Seiten der Dreiecke.

Bisher wurde in der Literatur diese Art der Verfeinerung insbesondere als Nachteil angesehen, weil die Gitterqualität durch die eingeschobenen LS-Verfeinerungen



(a) (b)

ABBILDUNG 7.4. reguläre Gitterverfeinerungen



(a) (b)

ABBILDUNG 7.5. 4TLS Gitterverfeinerungen

schlechter ist, als wenn nur regulär verfeinert worden wäre. Bei der regulären Verfeinerung hängen die erzeugten Gitter jedoch davon ab wo im vorherigen Schritt bereits verfeinert wurde wenn mehrfach aufeinanderfolgend lokal verfeinert wird (siehe Abbildung 7.4) - anders bei der LS-Verfeinerungen (Abbildung 7.5). Als Abhilfe wurde die Rot-Grün Verfeinerung vorgeschlagen. Bei der Rot-Grün Verfeinerung werden zunächst alle ausgewählten Dreiecke unterteilt. Die dabei entstehenden Kanten nennt man rote Kanten. Danach werden Dreiecke mit hängenden Knoten unterteilt, die hierbei entstehenden unerwünschten (grünen) Kanten werden vorgemerkt. Soll das Gitter erneut verfeinert werden, werden zunächst die grünen Kanten entfernt. Diese Verfeinerung liefert also im Allgemeinen keine hierarchischen Gitter, aber ermöglicht bessere (im Sinne der Approximation) Triangulierungen.

7.2. Ein neuer Algorithmus zur Wahl der Verfeinerungsparameter

Im Folgenden wird davon ausgegangen, dass man sich für eine sukzessive LS-Verfeinerung entschieden hat und die Starttriangulation so gut gewählt wurde, dass diese gleichzeitig eine NVB ist. Die Reihenfolge der Unterteilungen ist dann egal.

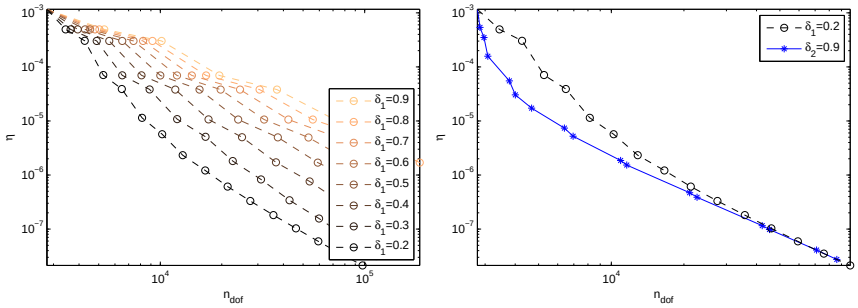
Hat man sich entschieden wie man verfeinern möchte, bleibt noch die Wahl welche und wie viele Dreiecke verfeinert werden sollen - ausgehend von lokalen Fehlerschätzer η_T für alle $T \in \mathcal{T}_H$. Dazu wurde bisher in der Literatur folgendes Verfahren verwendet:

(50) lösen → Fehler schätzen → markieren → verfeinern

Zum Verfeinern markiert werden dabei alle $T \in \mathcal{T}_H$ mit $\eta_T > \eta_0$ mit einer Schranke η_0, die bisher auf zwei unterschiedliche Weisen berechnet wurden: Entweder wurde η_0 so gewählt, dass mindestens $\delta_1|\mathcal{T}|$ Dreiecke verfeinert werden oder

$$\eta_0 = (1 - \delta_2) \max_{T \in \mathcal{T}_H} \eta_T$$

gewählt ([**18**]). Für $\delta = 1$ werden also stets alle und für $\delta = 0$ kein Element verfeinert. Der Verfeinerungsparameter δ wurde dabei bisher handgewählt. Beidemal verfeinert ein kleines δ (siehe Abbildung 7.6) die Zerlegung nur sehr lokal (was dem Wunsch nach Adaptivität entspricht). Aber dadurch, dass nur wenig neue Elemente hinzukommen, muss man den ganzen Prozess inklusive Lösen des Gleichungssystems sehr oft wiederholen bis man eine feine Triangulierung hat.



(a) Erreichte Fehlerreduktion je nach Verfeinerungsparameter.

(b) Die Verfahren verhalten sich asymptotisch gleich.

ABBILDUNG 7.6. Herkömmliche Verfeinerungsstrategien.

Indem wir uns nun Gedanken darüber machen, was die Verfeinerung alles leisten sollte, werden wir diese einfache Vorgehensweise verwerfen und andere Parameter einführen.

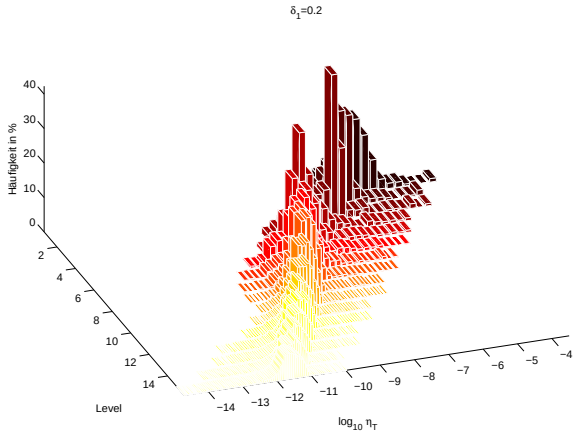
Allgemein sollte ein Verfeinerungsalgorithmus möglichst viele Verfeinerungen vornehmen (also auch eventuelle Mehrfachteilungen eines Dreiecks) ohne, dass man dies später bereut. Bereuen könnte man eine lokale Verfeinerung zum einen, wenn der Gesamtfehler dadurch nicht wesentlich reduziert wurde und zum anderen, wenn die Anzahl der Elemente in T_h so groß geworden ist, dass man u_h (aus Speicher- oder Zeitmangel) nicht mehr berechnen kann. Letzteres wollen wir dadurch umgehen, dass die Anzahl der Elemente begrenzt seien soll durch eine Konstante $n_T^{\max}$:

$$|\mathcal{T}_h| \leq n_T^{\max}. \tag{51}$$

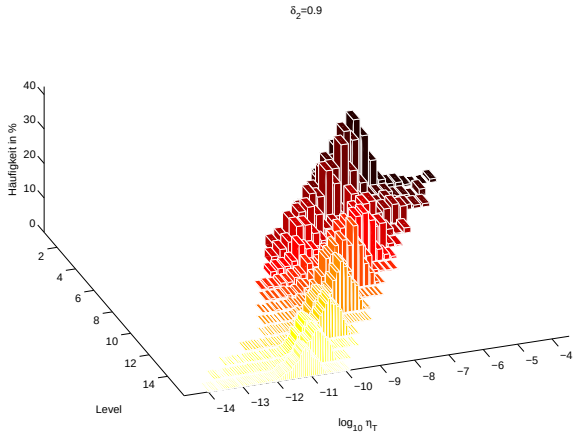
Andererseits wünschen wir uns mindestens $\delta|\mathcal{T}_H|$ neue Elemente

$$|\mathcal{T}_h| \geq (1 + \delta)|\mathcal{T}_H| \tag{52}$$

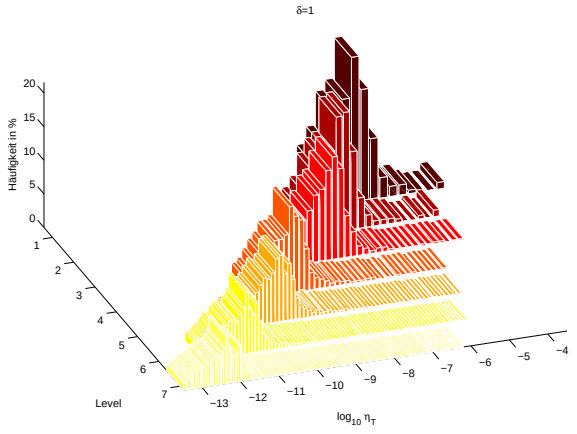
damit das feinste Gitter hinreichend schnell gefunden wird. Offensichtlich können sich Bedingung (51) und (52) widersprechen, dann ist (51) wichtiger. Bedingung (52) kann man für $\delta > 1$ nur erreichen, wenn ein Element mehr als zweigeteilt wird.



(a)



(b)



(c)

ABBILDUNG 7.7. Histogrammentwicklung der lokalen Fehler bei adaptiver Verfeinerungen (a) und (b) im Vergleich zur gleichmäßigen Verfeinerung (c)

Wir betrachten nun, wie der Fehler sich nach einer Teilung eines Elementes lokal verhält um diese Abschätzungen für eine mehrfache Teilung verwenden zu können. Für einen asymptotisch exakten Fehlerschätzer gilt nach (30), (31), (39):

$$\eta = \sqrt{\sum_{T\in\mathcal{T}} \eta_T^2} = O(h^k)$$

Wir können annehmen, dass sich asymptotisch auch der lokale Fehler so verhält (durch Mangel an Regularität kann die Konvergenzrate k allerdings geringer sein - siehe Abbildung 7.8):

$$\eta_T = O(h^k).$$

Dadurch können wir a-priori abschätzen, dass eine Unterteilung eines Elementes $T_H = \bigcup T_h^i$ in $n = 2$ Elemente T_h^1, T_h^2 das lokale Fehlerquadrat nicht nur auf die Teilelemente aufteilt

$$\eta_{T_H}^2 \geq \sum_{T_h^i \subset T_H} \eta_{T_h^i}^2$$

sondern auch noch dem entsprechend der verbesserten Approximationsgüte verringert:

$$\eta_{T_h^i}^2 \approx \left(\frac{h_{T_h^i}}{h_{T_H}}\right)^{2k} \frac{1}{n} \eta_{T_H}^2$$

so dass man durch eine Konstante $c_{n,k} > 1$, die von k und der Art der Aufteilung abhängt, abschätzen kann

$$\eta_{T_h^i}^2 \approx c_{n,k} \eta_{T_H}^2. \tag{53}$$

Eine Vierteilung eines Dreiecks ($n = 4$) liefert im Idealfall eine Halbierung von h

$$\left(\frac{h_{T_h^i}}{h_{T_H}}\right) = \frac{1}{2},\ i = 1, 2, 3, 4$$

und man erhält so

$$c_{4,k} = \frac{1}{4} \cdot 2^{-2k} = 2^{-2(k+1)}$$

Für eine doppelte Zweiteilung eines Elementes sollte die gleiche Abschätzung gelten, so dass man die Abschätzung

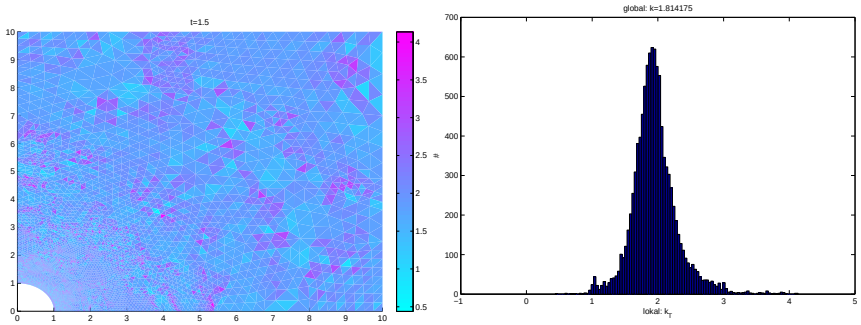
$$c_{2,k} = \sqrt{c_{4,k}} = \sqrt{2^{-2(k+1)}} = 2^{-k-1}$$

erhält. Allgemein ergibt sich im d-dimensionalen bei einer gleichmäßigen Teilung in n Elemente

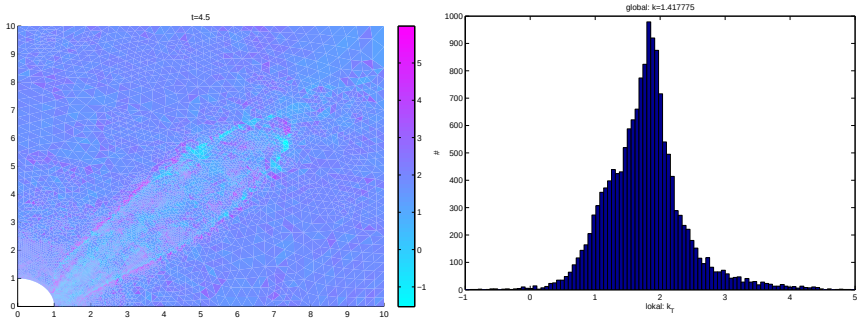
$$c_{n,k} = \frac{1}{n} \sqrt[d]{\frac{1}{n}}^{2k} = n^{-\frac{2}{d}k-1}.$$

Und insbesondere folgt bei quadratischen Dreieckeselementen in der LSFEM nach Zweiteilung bzw Vierteilung also:

$$c_{2,2} = \frac{1}{8},\ \text{bzw. } c_{4,2} = 2^{-4} = \frac{1}{64}$$



(a) elastischer Fall (b) Histogramm



(c) plastischer Fall (d) Histogramm

ABBILDUNG 7.8. Die gemessene lokale Konvergenzrate des Fehlerschätzers im elastischen (a,b) und plastischen Fall (b,c). Bei perfekter Plastizität ist wegen mangelnder Regularität eine langsamere Konvergenz am Rand der plastischen Zone zu beobachten. Diese vermindert auch die globale Konvergenz.

Diese Konstanten kann man algorithmisch ausnutzen: Bei einer effizienten Zerlegung wären alle η_T ungefähr gleichgroß. Würde man jeweils nur ein Dreieck mit größtem Fehler zweiteilen ergäbe sich nach endlich vielen Schritten ein Spektrum der Fehlerquadrate von

$$c_{2,k} \max_{T\in\mathcal{T}} \eta_T^2 \leq \eta_T^2 \leq \max_{T\in\mathcal{T}} \eta_T^2 \; \forall T \in \mathcal{T}$$

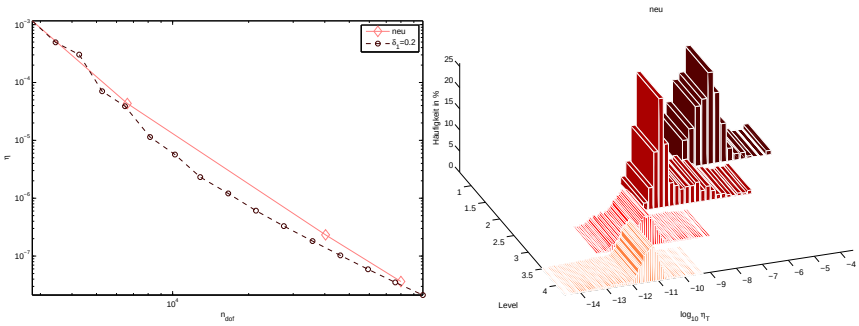
Eine Vierteilung von Dreiecken würde zu einem entsprechend größeren Spektrum führen

$$c_{4,k} \max_{T\in\mathcal{T}} \eta_T^2 \leq \eta_T^2 \leq \max_{T\in\mathcal{T}} \eta_T^2 \; \forall T \in \mathcal{T}$$

Die bereits vorgeschlagene Strategie die Dreiecke abhängig vom lokalen Fehler zweizuteilen oder vierzuteilen ist daher sinnvoll, aber unnötig, wenn man wie folgt vorgeht: Markiert wird immer nur ein einziges Element. Durch (53) kann dann der neue Fehler in der Verfeinerung abgeschätzt werden. Nach (50) können wir dann ohne erneutes Lösen direkt fortfahren mit wiederholten Schritten von

(54) Fehlerschätzung aktualisieren → markieren → verfeinern

Abbrechen kann man, sobald (52) erfüllt ist und man muss abbrechen sobald (51) beim nächsten Verfeinern verletzt werden würde. Es ist möglich, dass wenn δ zu groß gesetzt wird die Aktualisierung der Fehlerschätzung zu grob schätzt, weil sich der Fehler anfangs anders verhält als asymptotisch für $h \to 0$. Sollte dies eintreten, kann man nach dem nächsten Lösen und Fehlerschätzen versehentlich zu stark verfeinerte Regionen wieder entfeinern. In den numerischen Beispielen hat sich herausgestellt, dass dieser Fall für $\delta \leq 3$ fast nie auftritt, so dass man getrost auf Entfeinerung verzichten kann. Bereits für $\delta = 1$ wird zwischen jedem Lösen die Elementzahl mindestens verdoppelt und damit ist der Gesamtaufwand für das Lösen auf allen Gittern gerade mal das Doppelte von dem Lösen auf dem feinsten Gitter.



(a) (b)

ABBILDUNG 7.9. Der neue Algorithmus erstellt mit wenigen Schritten sehr lokal verfeinerte Triangulierungen mit vielen Freiheitsgraden

7.3. Eine neue, verbesserte Markierungsstrategie

In Anlehnung an die klassischen Markierungsstrategien kann man das Element mit größtem Fehler zum Verfeinern markieren. Dies ist jedoch keineswegs optimal: Um die entstehende Zerlegung zulässig zu machen, müssen nach jeder Unterteilung noch Dreiecke entsprechend des Ausbreitungpfades geteilt (im schlimmsten Fall viergeteilt) werden um hängende Knoten zu vermeiden. Dadurch könnte sich das Spektrum bis zu

$$(c_{4,k})^{n_U} c_{2,k} \max_{T \in \mathcal{T}} \eta_T^2 \leq \eta_T^2 \leq \max_{T \in \mathcal{T}} \eta_T^2 \; \forall T \in \mathcal{T}$$

vergrößern, wobei der Exponent n_U die maximale Anzahl der unerwünschten Unterteilungen eines einzelnen Dreiecks angibt. Meistens ist zwar $n_U = 1$, da sich die Fehler glatt im Gebiet verhalten. Aber trotzdem ergibt sich eine signifikante Verschlechterung der Schranken.

Um eine effiziente Triangulierung zu erhalten ist es notwendig diesen Effekt mit in die Markierungsstrategie einzubeziehen. Dazu ist es notwendig im voraus den Ausbreitungspfad aufzustellen und ihn in die Effizienzabschätzung mit einzubeziehen.

Bisher wurden pauschal ein Element mit größtem Fehler verfeinert, unter der Annahme, dass dort durch Einführung neuer Freiheitsgrade der Gesamtfehler am meisten reduziert werden kann. Korrekt ergibt sich die Effizienz e_T der Verfeinerung eines einzelnen Elementes aber durch das Verhältnis von Verbesserung zu Aufwand und die effizientesten Dreiecke sind als erstes zu markieren.

Die Verbesserung ist in diesem Fall die (erwartete) Fehlerreduktion aller durch den Ausbreitungspfad betroffenen Dreiecke. Und der induzierte Aufwand ist in etwa proportional zur Anzahl aller neu eingeführten Freiheitsgrade (im gesamten Ausbreitungspfad). Insgesamt erhält man also

$$e_T = \frac{\eta_{T_H}^2 - \sum_{T_h \in P(T_H)} \eta_{T_h}^2}{n_{dof,h} - n_{dof,H}}. \tag{55}$$

Abbildung 7.11 auf Seite 93 zeigt, dass sich damit bessere adaptive Verfeinerungen erzeugen lassen, da der Ausbreitungspfad (siehe Abbildung 7.10 auf der nächsten Seite) nicht verschwindet.

7.4. Ein neuer Beitrag zur parallelisierbaren Verfeinerung

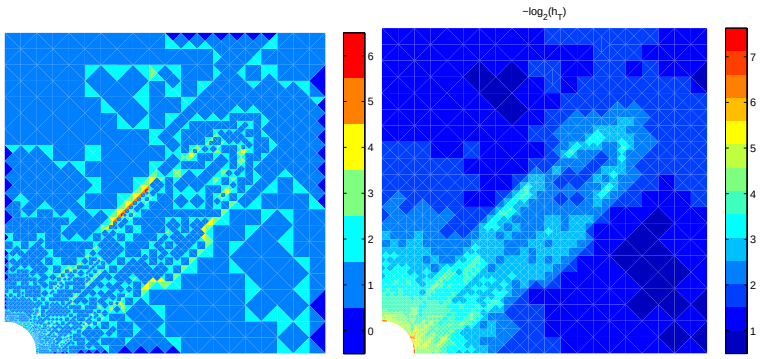
Ein parallelisierbarer Algorithmus zur Verfeinerung findet sich bereits in [**37**]. Der dortige Algorithmus hat jedoch keinen Parameter für die maximale Anzahl der Elemente die entstehen dürfen sondern verfeinert einfach alle markierten Elemente. Also ist es dort notwendig nicht zu viele Elemente zu markieren. Doch wie viele man maximal markieren darf ist schwierig abzuschätzen.

Die Überlegungen aus den beiden vorherigen Abschnitten legten nahe, dass man immer nur ein Element markiert und zwar das Effizienteste. Wendet man nun aber sequenziell einen parallelisierten Algorithmus mit immer nur einem markierten Element an hat man doch wieder nur einen sequenziellen Algorithmus. Wir müssen uns also einerseits überlegen wie viel Elemente man gleichzeitig markieren kann ohne die maximale Elementzahl zu überschreiten und andererseits ohne ineffizient zu markieren. Ineffizient markieren würde bedeuten ein wenig effizientes Element zu markieren ohne vorher effizientere Elemente zu markiert. Verfeinert man das effizienteste Element und zwangsläufig mit ihm alle Elemente im Ausbreitungspfad, dann sinkt nach (53) das Fehlerquadrat in diesen Elementen um einen Faktor. Nimmt man weiter an, dass sich der Nenner in (55) konstant verhält, sinkt dann auch die Effizienz dieser Elemente ebenso um einen konstanten Faktor:

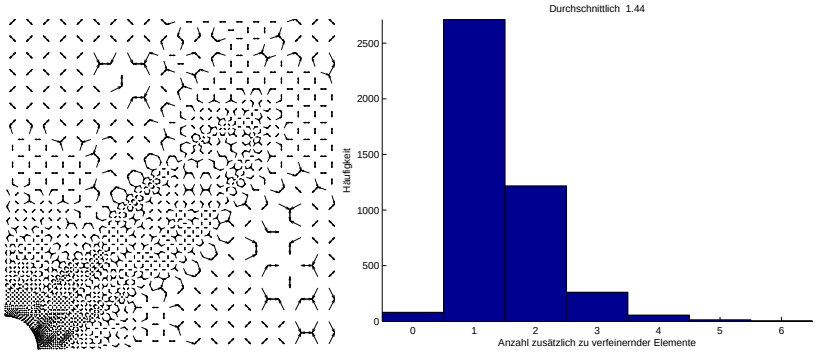
$$e_{T_h^i} \approx c e_{T_H}.$$

Hat ein grobes Element eine größere Effizienz als dieser Schätzwert kann man es auf jeden Fall verfeinern bevor das effizienteste Element noch mal verfeinert wird. Man kann also alle Elemente mit

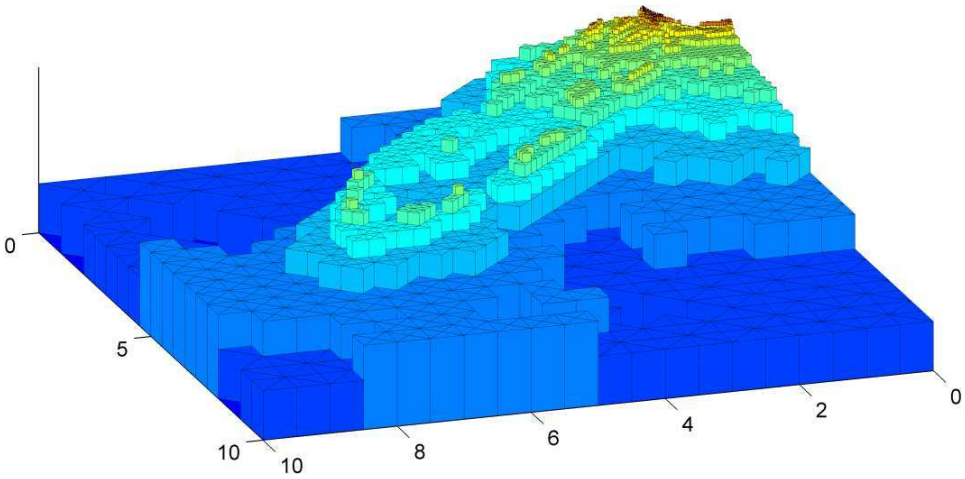
$$e_{T_H} > c \max_{T \in \mathcal{T}_H} e_{T_H}$$



(a) (b)

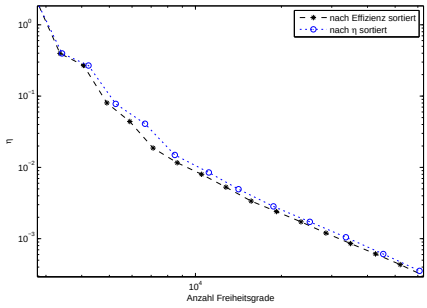


(c) (d)



(e) 3D-Plot der lokalen Elmentdurchmesser. Dabei ist gut zu erkennen, dass immer mehrere Elemente gleichzeitig verfeinert werden mussten.

ABBILDUNG 7.10. Die Anzahl zusätzlich zu verfeinernder Elemente (a) und (d) bei feinen adaptiven Triangulierungen (b) und (e) entsprechend des Ausbreitungspfades (c).



(a)

ABBILDUNG 7.11. Numerischer Vergleich der herkömmlichen Markierungsstrategie mit der neuen Effizienstrategie.

gleichzeitig markieren (und parallel verfeinern) ohne ineffizient zu werden. Sollte bei der Verfeinerung $|\mathcal{T}_h| > n_T^{\max}$ werden, darf man diese Verfeinerung nicht übernehmen. Stattdessen kann man dann interpolierende Suche (siehe z.B. [**54**, S. 240]) verwenden um das größte $\tilde{c} \in (c, 1]$ zu finden mit $|\mathcal{T}_h| \leq n_T^{\max}$, wobei man testweise die Elemente mit

$$e_{T_H} > \tilde{c} \max_{T \in \mathcal{T}_H} e_{T_H}$$

markiert. Dafür braucht man gerade mal $O(\log(\log(n_T^{\max}))$ Versuche.

7.5. Konforme Entfeinerung

Wie kann man also Zerlegungen erzeugen, so dass lokal entweder $\mathcal{T}_i$ eine Verfeinerung von $\mathcal{T}_{i+1}$ oder andersherum ist? Wenn man $\mathcal{T}_{i+1}$ als adaptive Verfeinerung von $\mathcal{T}_i$ erzeugt liegt diese Eigenschaft auf der Hand. Leider ist dies nicht praktikabel, da die Verfeinerung zusätzliche Freiheitsgrade einführt, so dass sie Bedingung (51) verletzen würde, wenn $\mathcal{T}_i$ bereits maximal viele Elemente besitzt. Um lokal zu verfeinern ist es also notwendig an anderer Stelle Freiheitsgrade einzusparen. Dazu ist ein Entfeinern notwendig.

Ein beliebiges Entfernen von Punkten aus einer Triangulierung und anschließende Neuvernetzung der Punkte führt im Allgemeinen nicht auf geschachtelte Triangulierungen. In [**48**] wurde ein Entfeinerungsalgorithmus vorgeschlagen der sicherstellt, dass eine Schachtelung $\mathcal{T}_i \subset \mathcal{T}_{i+1}$ erhalten bleibt. Dazu wird davon ausgegangen, dass man weiß wie $\mathcal{T}_i$ aus dem gröbsten Gitter $\mathcal{T}_H$ entstanden ist. Genauer gesagt braucht der Algorithmus alle Zwischen-Verfeinerungen $\{\mathcal{T}_j\}$ von $\mathcal{T}_H$ nach $\mathcal{T}_i$ um dann in $\{\mathcal{T}_j\}$ eine Triangulierung T_k zu finden in der das zu entfeinernde Dreieck gerade erst erzeugt worden ist. Die vorhergehende Triangulierung T_{k-1} enthält dann die gewünschte Entfeinerung. Dabei fällt auf, dass ein Entfeinern eines einzelnen Dreiecks in der Regel das Entfeinern vieler anderer Dreiecke erzwingt. Dies ist beim Markieren der zu entfeinernden Dreiecke zu beachten. Ob es sich lohnt ein Dreieck (und mit ihm viele unschuldige Dreiecke) zu entfeinern kann man dann analog zur Effizienz bei der Markierung zur Verfeinerung entscheiden.

Der Algorithmus aus [**48**] hat zwei Nachteile: Erstens muss man entweder alle Zwischenschritte $\{\mathcal{T}_j\}$ speichern oder ein hierarchische (Baum-)Struktur der Verfeinerungen anlegen. Leider unterstützt Matlab aber von Haus aus keine Baum-Datenstrukturen. Ein nachträgliches Implementieren von Bäumen würde zu sehr langsamen Programmen führen, da die Speicherverwaltung von kleinen Datenstrukturen nicht gerade zu den Stärken von Matlab gehört. Außerdem muss zweitens für jedes zu verfeinernde Dreieck separat in der Baumstruktur gesucht werden. Eine parallelisierbare (vektorisierbare) Implementierung davon liegt nicht auf der Hand.

In dieser Arbeit wurde daher ein anderer Algorithmus entwickelt: Unter einer sehr einfachen Bedingung an die Ausgangs-Triangulation

$$\max_{E\in E(K_i)} |E| < 2 \cdot \min_{E\in E(K_i)} |E| \; \forall K_i \in \mathcal{T}$$

ist die NVB immer eine LS-Verfeinerung, wenn die Reihenfolge der Punkte/Kanten entsprechend eindeutig gewählt sind: Die Kanten der Dreiecke werden in dieser Arbeit eindeutig gegen den Uhrzeigersinn orientiert durchnummeriert und die längste Seite im Dreieck (gibt es mehrere davon wird die Kante mit kleinster Anfangskoordinate) zuerst gespeichert. Eine NVB/LS-Verfeinerung ergibt sich dann wenn bei der Zerteilung immer zuerst die erste Kante eines Dreiecks geteilt wird.

Die NVB hat den Vorteil, dass die entstehenden Triangulierungen unabhängig von der Reihenfolge der lokalen Verfeinerungen sind. Um eine konform entfeinerte Triangulierung $\mathcal{T}^-$ aus einer Triangulation $\mathcal{T}$ zu erzeugen reicht es dann, ausgehend vom gröbsten Gitter $\mathcal{T}_H$ dieses sukzessiv dort zu $\mathcal{T}_{j+1}$ zu verfeinern, wo Punkte in $\mathcal{T}$ Kantenhalbierende von $\mathcal{T}_j$ sind und dabei kein Dreieck erzeugt würde, welches entfeinert werden soll. Die Entfeinerung kann somit auf Verfeinerungen zurückgeführt werden. Da die Verfeinerung vektorisiert implementiert ist ergibt sich automatisch ein vektorisierter (parallelisierbarer) Algorithmus für die Entfeinerung. Als Datenstruktur braucht man außerdem nur das gröbste Gitter und das letzte feine Gitter zu speichern um daraus das neue Gitter zu konstruieren, die Zwischenschritte werden nicht benötigt und auch keine hierarchische Datenstruktur.

Anstatt aus dem Gitter $\mathcal{T}_i$ ein Gitter $\mathcal{T}_{i+1}$ durch sowohl Verfeinerung als auch Entfeinerung zu konstruieren bietet es sich dann auch an im Zeitschritt $i+1$ ein Gitter allein durch Verfeinerung des gröbsten Gitters zu konstruieren. Dank NVB ist die dabei entstehende Triangulierung automatisch kompatibel zu $\mathcal{T}_i$ ohne dass $\mathcal{T}_i$ benutzt wird.

Literaturverzeichnis

[1] Andrew John Abbo. *Finte Element Algorithms for Elastoplasticity and Consolidation*. PhD thesis, University of Newcastle, 1997. 3rd Edition 2005.

[2] Roger Alexander. Diagonally implicit Runge-Kutta methods for stiff o.d.e.'s. *SIAM J. Numer. Anal.*, 14(6):1006–1021, 1977.

[3] I. Babuska and A. K. Aizi. On the angle condition in the finite element method. *SIAM J. Numer. Anal.*, 13:214–226, 1976.

[4] D. A. Barry, S. J. Barry, and P. J. Culligan-Hensley. Algorithm 743: Wapr: A fortran routine for calculating real values of the w-function. *ACM Transactions on Mathematical Software.*, 21(2):172–181, 1995.

[5] Ake Björck. *Numerical Methods for least squares Problems*. SIAM, 1996.

[6] Radim Blaheta. Convergence of Newton-type methods in incremental return mapping analysis of elasto-plastic problems. *Comput. Methods Appl. Engrg*, 147:167–185, 1997.

[7] Pavel Bochev and Max Gunzburger. Least-squares finite element methods. In *Proceedings of the International Congress of Mathematicians*, Madrid, 2006. Invited Lectures.

[8] Pavel B. Bochev and Max D. Gunzburger. Finite element methods of least-squares type. *SIAM Rev.*, 40:789–837, 1998.

[9] Sylvie Boldo and Marc Daumas. A simple test qualifying the accuracy of Horner's rule for polynomials. Technical report, Laboratoire de l' Informatique du Parallèlisme, 2003.

[10] Ronaldo I. Borja and Seung R. Lee. Cam-Clay plasticity, part 1: Implicit integration of elastoplastic constitutive relations. *Computer Methods in Applied Mechanics and Engineering*, 78:49–72, 1990.

[11] Dietrich Braess. *Finite Elements: Theory, fast solvers, and applications in solid mechanics*. Cambridge University Press, Cambridge, 2001. 2. Auflage.

[12] Franco Brezzi and Michel Fortin. *Mixed and Hybrid Finite Element Methods*. Springer Series in Computational Mathematics. Springer, New York, 1991.

[13] J. C. Butcher. *Ordinary Differential Equations*. John Wiley & Sons, 2003.

[14] Jörg Büttner and Bernd Simeon. Index 2 daes in plasticity theory. In *Proc. Appl. Math. Mech.*, volume 1, 2002.

[15] Jörg Büttner and Bernd Simeon. Time integration of the dual problem of elastoplasticity by Runge-Kutta methods. *SIAM J. Numer. Anal.*, 41(4):1564–1584, 2003.

[16] Zhiqiang Cai, Johannes Korsawe, and Gerhard Starke. An adaptive least squares mixed finite element method for the stress-displacement formulation of linear elasticity. *Numerical Methods for Partial Differential Equations*, 21:132–148, 2005.

[17] Antonio Capsoni and Leone Corradi. A plane strain formulation of the elastic-plastic constutive law for hardening von Mises materials. *Int. J. Solids Structures*, 32(23):3515–3531, 1995.

[18] Carsten Carstensen. State of the art on the error reduction in AFEM. *PAMM*, 5:827–828, 2005.

[19] Carsten Carstensen and R. H. W. Hoppe. Error reduction and convergence for an adaptive mixed finite element method. *Mathematics of computation*, 75:1033–1042, 2006.

[20] X. Chen, L. Qi, and D. Sun. Global and superlinear convergence of the smoothing Newton method and its application to general box constrained variational inequalities. *Mathematics of Computation*, 67(222):519–540, 1998.

[21] Philippe G. Ciarlet. *The Finite Element Method for Elliptic Problems*. North-Holland Publishing Company, Amsterdam, 1980.

[22] I. F. Collins. Associated an non-associated aspects of the constitutive laws for coupled elastic/plastic materials. *The Innternational Journal of Geomechanics*, 2(2):259–267, 2002.

[23] Ronald Cools. An encyclopaedia of cubature formulas. *Journal of Complexity*, 19(3):445–453, 2003.

[24] R. M. Corless, D. E. G. Hare G. H. Gonnet, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. 1996.

[25] Robert M. Corless, Gaston H. Gonnet, D. E. G. Hare, and David J. Jeffrey. Lambert's W function in Maple. 1993.

[26] R. O. Davis and A. P. Selvadurai. *Plasticity and Geomechanics*. Cambridge University Press, 2002.

[27] Hans Echenbauer and Walter Schnell. *Elastizitätstheorie*. BI-Wissenschaftsverlag, Mannheim, 1993. 3., vollständig überarbeite und erweiterte Auflage.

[28] F. N. Fritsch, R. E. Shafer, and W. P. Crowley. Solution of the transcendental equation $w \exp w = x$. *Communications of the ACM*, 16(2):123–124, 1973.

[29] Michael Gee. Remeshing für finite elemente berechnungen mit großen deformationen. Master's thesis, Institut für Baustatik der Universität Stuttgart, 1999.

[30] P. L. George, H. Borouchaki, P. J. Frey, P. Laug, and E. Saltel. Mesh generation and mesh adaptivity. In Erwin Stein, R. Borst, and T.J.R. Hughes, editors, *Encyclopedia of Computational Mechanics*, volume 1, pages 141–155. Wiley, 2004.

[31] David Goldberg. What every computer scientist should know about floatin-point arithmetic. *ACM Computing Surveys*, 23(1), 1991.

[32] Herman H. Goldstine. *A History of Numerical Analysis from the 16th through the 19th Century*. Studies in the History of Mathematics and Physical Science 2. Springer Verlag, New York, 1977.

[33] Andreas Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Frontiers in applied mathematics. SIAM, 2000.

[34] Ernst Hairer, Christian Lubich, and Michel Roche. *the Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods*, volume 1409 of *Lectur Notes in Mathematics*. Springer-Verlag, 1989.

[35] Weimin Han and B. Daya Reddy. On the finite element method for mixed variational inequalities arising in elastoplasticity. *SIAM Journal on Numerical Analysis*, 32(6):1778–1807, 1995.

[36] Immaculada Higueras and Berta Garcia-Celayeta. Runge-kutta methods for daes. a new approach. *Journal of Computational and Applied Mathematics*, 111:49–61, 1999.

[37] Mark T. Jones and Paul E. Plassmann. Parallel algorithms for adaptive mesh refinement. *SIAM J. Sci. Comput.*, 18(3):686–708, 1997.

[38] Mark Kachanov, Boris Shafiro, and Igor Tsukrov. *Handbook of Elasticity Solutions*. Kluwer Academic Publishers, 2003.

[39] W.F. Mitchell. Adaptive refinement for arbitrary finite-element spaces with hierarchical basis. *J. Comput. Appl. Math.*, 36:65–78, 1991.

[40] Bo nan Jiang. *The Least-Squares Finite Element Method*. Springer, Heidelberg, 1998.

[41] Hermann G. Matthies Oliver Kayer-Herold. Least-squares FEM literature review. Technical report, TU Braunschweig, August 2005.

[42] D. Peric, Ch. Hochard, M.Dutko, and D.R.J. Owen. Transfer operators for evolving meshes in small strain elasto-plasticity. *Comput. Method Appl. Mech. Engrg.*, 137:331–344, 1996.

[43] Dunja Peric and Chune Huang. Analytical solutions for the theree-invariant Cam clay model: Drained and undrained loading. In *16th ASCE Engineering Mechanics Conference*, Seattle, 2003. University of Washington.

[44] L. R. Petzold. Order results for implicit Runge-Kutta methods applied to differential / algebraic systems. *SIAM J. Numer. Anal.*, 23:837–852, 1986.

[45] E. Ramm, E. Rank, R. Rannacher, K. Schweizerhof, E. Stein, W. Wendladn, G. Wittum, P. Wriggers, and W. Wunderlich. *Error-controlled Adaptive Finite Elements in Solid Mechanics*. Wiley, 2002.

[46] P. A. Raviart and J. M. Thomas. A mixed finite element method for 2-nd order elliptic problems. *Mathemathical Aspects of the Finite Element Method. Lecture Notes in Math.*, 606:292–315, 1977.

[47] Sergey I. Repin. Errors of finite element method for perfect elasto-plastic problems. *Mathematical Models and Methods in Applied Sciences*, 6(5):587–604, 1996.

[48] Maria-Cecilia Rivara. Selective refinement/derefinement algorithms for sequences of nested triangulations. *Int. J. Num. Meth. Engineering*, 28:2889–2906, 1989.

[49] Maria-Cecilia Rivara. New mathematical tools and techniques for the refinement and/or improvement of unstructured triangulations. 1996.

[50] Maria Cecilia Rivara and Gabriel Iribarren. The 4-triangles longest-side partition of triangles and linear refinement algorithms. *Mathematics of Computation*, 65(216):1485–1502, October 1996.

[51] Axel Ruhe and Per Ake Wedin. Algorithms for separable nonlinear least squares problems. *SIAM Review*, 22(3):318–337, 1980.

[52] Siegfried M. Rump. Fast and parallel interval arithmetic. *BIT*, 39:534–554, 1999.

[53] Monika Schulz. *A-Posteriori-Fehlerschätzer für mittels Finiter Elemente modellierte elasto-plastische Verformungsvorgänge*. PhD thesis, Universität Stuttgart, 1997.

[54] Robert Sedgewick. *Algorithmen*. Addison-Wesley, 1991.

[55] Jonathan Richard Shewchuk. What is a good linear element? - interpolation, conditioning, and quality measures. In *Eleventh International Meshing Roundtable*, pages 115–126, Ithaca, New York, 2002. Sandia National Laboratories.

[56] R. C. C. Silva, A. F. Loula, and J.N.C. Guerreiro. Local gradient and stress recovery for trangular elements. *Computers and Structures*, 82:2083–2092, 2004.

[57] J.C. Simo and T.J.R. Hughes. *Computational Inelasticity*. Springer, 2000. Corrected second printing.

[58] Gerhard Starke. An adaptive least-squares mixed finite element method for elasto-plasticity. to appear.

[59] Holger Steeb. *Fehlerschätzer für FE-Berechnungen bei entfestigenden Materialien*. PhD thesis, Universität Stuttgart, 2002.

[60] J. P. Suarez, A. Plaza, and G. F. Carey. Propagation path properties in iterative longest-edge refinement, 2003.

[61] Christian Wieners. Multigrid methods for Prandtl-Reuss plasticity. *Numerical Linear Algebra with Applications*, 6:457–478, 1999.

[62] P. Wriggers. *Nichtlineare Finite-Element-Methoden*. Springer, Heidelberg, 2001.

[63] Zhimin Zhang and Ahmed Naga. A new finite element gradient recovery method: Superconvergence property. *SIAM Journal on Scientific Computing*, 26(4):1192–1213, 2005.

[64] O. C. Zienkiewicz and R. L. Taylor. *Finite Element Method for Solid and Structural Mechanics*. Elsevier, Oxford, 2005. 6. Auflage.

[65] O. C. Zienkiewicz and J. Z. Zhu. The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique. *International Journal for Numerical Methods in Engineering*, 33(7):1331–1364, 1992.

Lebenslauf

Jörg Michael Kubitz

20.07.1978	Geboren in Hannover
1984-1988	Grundschule Krähenwinkel / Langenhagen, Klassensprecher
1988-1990	Orientierungstufe Brink / Langenhagen, 1 Jahr Klassensprecher
1990-1992	Gymnasium Langenhagen, 1 Jahr im Schülerrat, Teilnahme an Computer SG und Schach AG
1992-1997	Gymnasium Lüchow, Abitur, Teilnahmen an Mathematik AG, Elektronik AG, Schach AG, Informatik AG, 3 Jahre Teilnahmen am Wettbewerb Jugend forscht, diverse Preise und Sonderpreise
4.-15.10.93	Praktikum bei SKF Lüchow, EDV-Abteilung
1996-1998	Selbständige EDV-Auftragsarbeiten für das Ingenieurbüro Waßmuth, Bösel / Lüchow
1997-1998	Wehrdienst im Nachschubbataillon Burg / Magdeburg. Berufsfortbildung im Rhetorik- und Schweisser-Kurs
1998-2003	Studium der Mathematik mit Studienrichtung Informatik, Universität Hannover, Diplom-Abschluss. Nebenbei ehrenamtliche Arbeit in der christlichen Drogenarbeit Neues Land Hannover, Jugendgruppenleitung in der Evangelisch-Freikirchlichen Gemeinde Bachstraße
2002-2003	Studentische Hilfskraft am Institut für Angewandte Mathematik
ab 1.6.2003	Drittmittelstelle an der Leibniz Universität Hannover
seit 1.1.2004	Doktorand mit Stipendium der DFG im Graduiertenkolleg 615