Numerical Methods
for Variational Phase-Field Fracture Problems

Katrin Mang and Thomas Wick

AG Wissenschaftliches Rechnen (GWR)

Institut für Angewandte Mathematik (IfAM)

Gottfried Wilhelm Leibniz Universität Hannover (LUH)

Welfengarten 1, 30167 Hannover, Germany

`https://www.ifam.uni-hannover.de/`

katrin.mang@ifam.uni-hannover.de

thomas.wick@ifam.uni-hannover.de

Last update:
Friday, July 19, 2019

1 Foreword

1.1 Classes and courses

The first version of these notes are a result of the spring block lecture, course no.: 327.018,

Numerical methods for variational phase-field fracture problems
https://www.dk-compmath.jku.at/Courses/2018s/phase-field-fracture-problems

taught by the second author in March 2018 at the Johannes Kepler University (JKU) Linz in Austria. This block lecture comprised seven lectures à 90 minutes.

The notes were further extended for the class

Numerical methods for contact problems: application to variational phase-field fracture propagation
http://www.thomaswick.org/links/ankuendigung_vpff_Wick_Mang_Noii.pdf

taught at the Leibniz University Hannover in the winter semester 2018/2019 in a $2+1+1$ class (lecture, theoretical and practical exercises). The programming part is based on

DOpElib [53, 72] www.dopelib.net `Examples/PDE/InstatPDE/Example8`.

In the year 2019 (June 26), these notes served as basis for a summer school lecture at Hasselt University (Belgium):

Phase field methods for fractured media
https://www.uhasselt.be/summer-school-phase-field-modeling-2019

These notes also contain materials from another summer school on Computational Mechanics of Materials and Structures (COMMAS) given at the University of Stuttgart (Germany) in Fall 2016 entitled with

Numerical methods and adaptivity for multiphysics phase-field fracture
http://www.thomaswick.org/commas_stuttgart_fall_2016.html

1.2 Short historical background and brief information

The **variational approach to fracture** was introduced in the year 1998 by G. A. Francfort and J.-J. Marigo. Shortly later, in the year 2000, a suggestion for the numerical treatment was proposed by B. Bourdin, G. A. Francfort and J.-J. Marigo.

In 2009 and 2010 important extensions with regard to the physics have been added by H. Amor, J.-J. Marigo and C. Maurini and C. Miehe, F. Welschinger and M. Hofacker, respectively. In the last mentioned paper, the variational approach was also called **phase-field approach**.

In these lecture notes, we simply call the approach as **variational phase-field fracture (VPFF)**.

These notes summarize the **early developments** and then focus on **current numerical analysis, numerical techniques, and software implementations** in which both authors have been involved since some time; starting back in the year 2013 during the Postdoc time of the second author at ICES (now Oden Institute) in Austin/Texas with M.F. Wheeler and A. Mikelić. Current developments and extensions are summarized at the end of these lecture notes in Section 18.

These notes also contain **results** (a simplified numerical analysis for relationships in model and discretization parameters, iteration on an extrapolation in some terms, comparison of different techniques imposing the crack irreversibility constraint, parallel scalability tests) that **have not yet been published elsewhere**. Moreover, several issues are explained with the help of 'simplified' problems, most often the obstacle problem. Here, **excursuses** shall help in understanding and some of them are delivered with **open-source online code**. Discussions on typical issues of multiphysics problems such as moving interfaces and interface conditions are provided in detail as well.

We hope that students, PhD students, postdoctoral researchers as well as peers find these lecture notes useful.

In the case of comments or mistakes, please let us know via email. The addresses are given on the title page.

Enjoy reading!

Katrin Mang & Thomas Wick

(Linz, March 2018 and Hannover, July 2019)

2 Acknowledgments

Research related to this work is being supported by the doctoral program Computational Mathematics of the Johannes Kepler University Linz, Austria; the German Priority Program 1748 (DFG SPP 1748) **Reliable Simulation Techniques in Solid Mechanics. Development of Non-standard Discretization Methods, Mechanical and Mathematical Analysis**

https://www.uni-due.de/spp1748/

with the project **Structure Preserving Adaptive Enriched Galerkin Methods for Pressure-Driven 3D Fracture Phase-Field Models**

https://www.uni-due.de/spp1748/adaptive_enriched_galerkin_methods

and the FWF project P-29181 **Goal-oriented error control for phase-field fracture coupled to multiphysics problems**

http://www.numa.uni-linz.ac.at/Research/Projects/P29181/index.shtml

Further thanks go to the German Priority Program 1962 (DFG SPP 1962)

https://spp1962.wias-berlin.de/

to project No. 17 **Optimizing Fracture Propagation Using a Phase-Field Approach**

https://spp1962.wias-berlin.de/project.php?projectID=17

We wish to thank Mary F. Wheeler (ICES, UT Austin, US), Andro Mikelić (Univ. Lyon, France), Sanghyun Lee (Florida State University, US), Timo Heister (University of Utah, US) for endless discussions on phase-field fracture from the year 2013 on. Furthermore, both authors want to thank Winnifried Wollner (UT Darmstadt, Germany), Mirjam Walloth (TU Darmstadt, Germany), Ira Neitzel (Univ. Bonn, Germany), Nima Noii (LUH, Germany), Meng Fan (one year-research stay at LUH, Germany), Bernhard Endtmayer (RICAM Linz, Austria), Daniel Jodlbauer (RICAM Linz, Austria), and Ulrich Langer (JKU Linz, Austria) for recent exchange of ideas.

Finally, we deeply thank our families for their understanding and support! Here, the second author wants to specifically thank Marie-Cécile : she is a hero taking the major care on our children - thank you so much!

Contents

1 Foreword **2**
1.1 Classes and courses 2
1.2 Short historical background and brief information 3

2 Acknowledgments **5**

3 Motivation **13**
3.1 The WhatHowWhy: What is phase-field? How is phase-field realized? Why phase-field? 13
3.2 Background information on approximating interfaces and discontinuities 16
3.3 An example of a practical problem: fracture and damage in screws 18

4 Notation and fundamental techniques **19**
4.1 Spatial dimension 19
4.2 Independent variables 19
4.3 Function, vector and tensor notation 19
4.4 Partial derivatives 19
4.5 Gradient, divergence, trace, Laplace, rotation 20
4.6 Vector spaces 21
4.7 Normed spaces 21
4.8 Chain rule 22
4.9 Transformation of integrals: substitution rule / change of variables 22
4.10 Gauss-Green theorem / divergence theorem 23
4.11 Integration by parts and Green's formula 23
4.12 Fundamental lemma of calculus of variations 24
4.13 Domains 25
4.14 Solution variables in a phase-field fracture setting 26
4.15 Summary of variables, parameters, and constitutive quantities 27

5 Modeling of variational phase-field fractures **29**
5.1 A first simplified phase-field model 29
5.1.1 Comments on the simplified phase-field problem Formulation 5.1 30
5.2 Pros and Cons of variational phase-field fracture modeling 32
5.3 Excursus I: the obstacle problem 32
5.3.1 Variational formulation 33
5.3.2 Strong formulation 34
5.3.3 Lagrange multiplier formulations 34
5.3.4 Brief introduction to one possibility to treat the obstacle constraint 35
5.4 Preliminaries to modeling quasi-static brittle fracture 35
5.5 Griffith's model from 1920 39
5.6 Francfort and Marigo's model from 1998 40
5.6.1 Specific forms of $E_T(C,u)$ 41
5.6.2 Properties 41
5.6.3 Weak formulation of quasi-static brittle phase-field fracture 43
5.7 Thermodynamic extensions and interpretations 44
5.7.1 Energy splitting for fracture under tension and compression 44
5.7.2 A weak phase-field fracture formulation using stress splitting 45
5.7.3 Derivation of a strain history field/replacing the irreversibility constraint 46
5.7.4 A weak phase-field fracture formulation using strain history field and stress splitting . . 48
5.8 Comments on energy formulations versus Euler-Lagrange PDEs 48

6 Classifications of PDEs **51**
6.1 Differential equations 51

6.2 Single equations versus PDE systems 52
6.3 Linear versus nonlinear 53
6.3.1 Definition of linearity 53
6.3.2 Classes of nonlinearity 53
6.3.3 Examples 54
6.4 Decoupled versus coupled 55
6.5 Variational equations versus variational inequalities 57

7 Numerical modeling part I: Regularization and discretization 61
7.1 Outline of the three numerical modeling chapters 61
7.1.1 Outline of Chapter 7 61
7.1.2 Outline of Chapter 8 61
7.1.3 Outline of Chapter 10 61
7.1.4 Demonstration of numerical modeling in terms of excursus and simulations 61
7.2 The crack irreversibility constraint 61
7.2.1 Strain history function 61
7.2.2 Penalization/augmented Lagrangian 62
7.2.3 Primal-dual active set method 63
7.3 Spatial discretization 64
7.3.1 A penalized, spatially discretized version 64
7.3.2 Numerical analysis of a simplified, prototype problem 65
7.4 A phase-field model for dynamic fracture (extensive exercise); towards space-time 71
7.4.1 The model (partially incomplete - to be augmented in the exercise) 71
7.4.2 Tasks 71
7.4.3 Bochner spaces - space-time functions 72
7.4.4 Space-time formulation of the elastic wave equation 73
7.5 Further models of time-dependent phase-field fracture 74
7.5.1 Quasi-static displacements, time-dependent phase-field system 75
7.5.2 Wave equation for displacements, quasi-static phase-field system 75
7.5.3 Wave equation for displacements, time-dependent phase-field system 75

8 Numerical modeling part II: Linearizations, nonlinear and linear solvers 77
8.1 Linearization techniques 77
8.2 Fixed-point iteration - general procedure 77
8.3 Partitioned (staggered) schemes for coupled problems 78
8.4 Staggered fixed-point iteration applied to phase-field fracture 78
8.5 Linearization of φ via extrapolation in time 78
8.5.1 Prototype examples 79
8.5.2 A linear-in-time extrapolation 79
8.6 An iteration on the extrapolation 80
8.7 Differentiation in Banach spaces 80
8.8 Newton's method in $\mathbb{R}$ - the Newton-Raphson method 82
8.8.1 Newton's method: overview. Going from $\mathbb{R}$ to Banach spaces 84
8.8.2 A basic algorithm for a residual-based Newton method 84
8.9 Inexact Newton 85
8.10 Details on the discretization and nonlinear/linear systems 85
8.10.1 Stationary linearized elasticity 85
8.10.2 Nonlinear elasticity 86
8.11 Abstract schemes for monolithic formulations and their numerical solution 87
8.11.1 Variational equations 87
8.11.2 Variational inequalities: penalization 88
8.12 Loops 89
8.13 Excursus II: Numerical modeling and implementation of the obstacle problem 89
8.13.1 Problem statement (recalling) 89
8.13.2 Variational forms 89

8.13.3 Regularization (penalization) of the inequality constraint . . . 90
8.13.4 Newton's method . . . 90
8.13.5 Discretization . . . 90
8.13.6 Details and discussions . . . 91
8.13.7 Implementation in DOpElib . . . 92
8.13.8 Mesh refinement studies with $\gamma = \frac{\bar{\gamma}}{h^2}$ . . . 93
8.13.9 Mesh refinement studies with $\gamma = \frac{\bar{\gamma}}{h^2_{\text{coarse}}}$ . . . 93
8.13.10 Further implementation exercises . . . 93
8.14 Newton-type methods applied to variational phase-field fracture . . . 94
8.14.1 Formulations . . . 94
8.14.2 A monotonicity-based Newton algorithm . . . 94
8.14.3 A successful basic Newton implementation with line search in C++ . . . 95
8.14.4 The Jacobian matrix . . . 97
8.14.5 Block structure of the Jacobian, solution vector, and right hand side . . . 98
8.15 A modified Newton method with Jacobian modification . . . 99
8.15.1 Computing the control parameter ω . . . 99
8.15.2 The modified Newton algorithm . . . 100
8.15.3 On the choice of a and b . . . 101
8.16 Excursus: Modified Newton for functional minimization in $\mathbb{R}^2$ . . . 102
8.16.1 Problem statement . . . 102
8.16.2 Derivatives . . . 103
8.16.3 Newton's method . . . 104
8.16.4 Numerical results and discussion . . . 104
8.16.5 Final comments . . . 105
8.17 A combined Newton algorithm and using a primal-dual active set strategy . . . 106
8.17.1 Problem statement . . . 106
8.17.2 A primal-dual active set strategy . . . 107
8.17.3 A combined Newton algorithm . . . 108
8.17.4 Solving the PDE in the Nonactive Set $\mathcal{N}$; including extrapolation of φ in the solid bulk term . . . 109
8.18 Linear solution inside Newton's method of monolithic phase-field fracture systems . . . 110
8.18.1 Fully monolithic approximation . . . 110
8.18.2 A block-diagonal preconditioner for the extrapolated scheme . . . 111
8.18.3 A very short hint to parallel computing . . . 111
8.19 Excursus III: linear and nonlinear solver performances for the obstacle problem . . . 112
8.19.1 Setup and goals . . . 112
8.19.2 Results . . . 112
8.19.3 Discussion . . . 114
8.20 The augmented Lagrangian loop and an inexact version . . . 114
8.20.1 Augmented Lagrangian penalization . . . 114
8.20.2 An adaptive Newton stopping criterion for an inexact augmented Lagrangian method . 114
8.21 Updating the strain history field when working with the variational equality system . . . 115

9 Simulations I: Single edge notched shear test 117
9.1 Relation to reality . . . 117
9.2 Goals of our computations . . . 118
9.3 Configuration . . . 119
9.4 Boundary conditions . . . 120
9.5 Initial conditions . . . 120
9.6 Parameters . . . 120
9.7 Quantities of interest . . . 120
9.8 Discussion of findings . . . 120

10 Numerical modeling part III: Adaptivity and goal functionals 123
10.1 Motivation of mesh adaptivity 123
10.2 Principles of error estimation 123
10.3 Efficiency, reliability, basic adaptive algorithm 124
10.4 Goal-oriented error estimation using duality arguments: dual-weighted residuals (DWR) 125
10.4.1 Problem statement 125
10.4.2 Lagrangian, primal, adjoint 125
10.4.3 Excursus: Variational principles and Lagrange multipliers in mechanics 126
10.4.4 First-order optimality system 129
10.4.5 Linear problems and linear goal functionals (Poisson) 130
10.4.6 Nonlinear problems and nonlinear goal functionals 132
10.5 Approximation of the adjoint solution for the primal estimator $\rho(u_h)(\cdot)$ 134
10.6 Approximation of the primal solution for the adjoint estimator $\rho^*(z_h)(\cdot)$ 135
10.7 Measuring the quality of the error estimator η 135
10.8 Localization techniques 135
10.8.1 The classical way of error localization of the primal estimator for linear problems 136
10.8.2 The classical way for the combined estimator 137
10.8.3 A variational primal-based error estimator with PU localization 138
10.8.4 PU localization for the combined estimator 138
10.9 Comments to adjoint-based error estimation 139
10.10 Mesh refinement strategies 139
10.10.1 How to refine marked cells 140
10.10.2 Convergence of adaptive algorithms 140
10.11 Goal-oriented a posteriori (primal) error estimator for phase-field fracture 141
10.12 Mesh refinement strategies for quasi-stationary (incremental) problems 142
10.13 Excursus IV: DWR for the obstacle problem using simple penalization 143
10.13.1 Problem statement 143
10.13.2 A posteriori PU-DWR error estimator (primal versions) 143
10.13.3 Numerical results and discussion 144
10.13.4 Literature on a posteriori error estimation for the obstacle problem 148
10.14 Predictor-corrector mesh adaptivity 148
10.14.1 The main algorithm 148
10.14.2 Goals and illustrations 149
10.14.3 Performance 151
10.15 Adaptive time step control 151
10.16 Final comments to error estimation in numerical simulations 152

11 Simulations II: a PU-DWR method for a slit modeled by phase-field 155
11.1 Configuration and manufactured solution 155
11.2 Boundary conditions 155
11.3 Initial condition 155
11.4 Parameters 156
11.5 Goal functional and reference value 156
11.6 PU-DWR error estimator in primal form 156
11.7 Code snippet of the PU-DWR loop implemented in deal.II 157
11.8 Numerical results 158

12 Simulations III: screw test: time adaptivity and iteration on extrapolation 159
12.1 Configuration 159
12.2 Boundary conditions 159
12.3 Initial condition 159
12.4 Parameters 160
12.5 Quantity of interests 160
12.6 Numerical results and discussion 160

13 Crack width, volume, interfaces - pressurized cracks 165
13.1 Computing the crack width . . . 165
13.2 Computing the total crack volume . . . 166
13.3 Pressurized fractures . . . 166
13.3.1 Fracture and solid domains . . . 166
13.3.2 An interface law between the fluid and solid sub-domains . . . 167
13.3.3 Modification of the energy functional . . . 167
13.3.4 A variational formulation of pressurized phase-field fractures . . . 168
13.3.5 The strong form . . . 169

14 Simulations IV: Parallel computing of a pressurized fracture 171
14.1 Model problem . . . 171
14.2 Numerical solution algorithm . . . 172
14.3 Problem setup - Sneddon/Lowengrub . . . 172
14.3.1 Configuration . . . 172
14.3.2 Boundary conditions . . . 173
14.3.3 Initial conditions . . . 173
14.3.4 Parameters . . . 173
14.3.5 Discretization and DoFs . . . 173
14.3.6 Quantities of interest . . . 174
14.3.7 Programming code . . . 174
14.3.8 Computing machine . . . 174
14.4 Numerical results . . . 175
14.4.1 Solution plot and locally refined meshes . . . 175
14.4.2 Evaluation of the crack opening displacement . . . 175
14.4.3 GMRES iterations per nonlinear (combined Newton) step . . . 176
14.4.4 Average number of combined Newton iterations per loading step . . . 176
14.4.5 Parallel computing . . . 176

15 Practicing in DOpElib `Examples/PDE/InstatPDE/Example8` 179
15.1 Installation of deal.II . . . 179
15.2 Installation of DOpElib . . . 179
15.3 Building and running the example . . . 179
15.4 Files . . . 180
15.5 Important files for working with the PDE and changing the program . . . 180
15.6 First changes to acquaint with the example . . . 180
15.7 Examples of possible tasks in the course . . . 181
15.7.1 Meeting 1 . . . 181
15.7.2 Meeting 2 . . . 181
15.7.3 Meeting 3 . . . 181
15.7.4 Meeting 4 - 7 (Choice 1) . . . 181
15.7.5 Meeting 4 - 7 (Alternative 1, advanced) . . . 181
15.7.6 Meeting 4 - 7 (Alternative 2, advanced) . . . 181
15.7.7 Meeting 4 - 7 (Alternative 3) . . . 182
15.7.8 Meeting 4 - 7 (Alternative 4, advanced) . . . 182
15.7.9 Meeting 4 - 7 (Alternative 5, advanced) . . . 182

16 Computational convergence analysis 183
16.1 Discretization error . . . 183
16.1.1 Relationship between h and N (DoFs) . . . 183
16.1.2 Discretization error . . . 184
16.1.3 Computationally-obtained convergence order: example for time step sizes k . . . 185
16.1.4 Computationally-obtained convergence order: example for the spatial mesh parameter h 186
16.2 Iteration error . . . 187

17 Wrap-up - Quiz 189

18 Outlook 191

18.1 Extensions, applications and further literature 191

18.2 Some open questions 192

19 The end 193

Bibliography 195

Index 205

3 Motivation

These lecture notes are an attempt to introduce variational phase-field fracture (VPFF) on a basic level. The prerequisites are numerical methods for partial differential equations, and possibly continuum mechanics, nonlinear techniques, and variational inequalities. To date, no introductory lecture notes exist and we put some efforts to introduce the early developments as well as recent numerical techniques. At the end, we also provide a list of further extensions and applications. The focus is more on the mathematical and numerical side rather than mechanical developments. This reflects the fact that both authors are applied mathematicians.

3.1 The WhatHowWhy: What is phase-field? How is phase-field realized? Why phase-field?

Key idea As shown in Figure 1, the most characteristic feature is that a smoothed indicator variable varying from 0 (fracture/damage) to 1 (intact material) is used to approximate (lower-dimensional) discontinuities in a displacement field (highlighted in Figure 2). The approximation width is represented through a regularization parameter $\varepsilon > 0$.

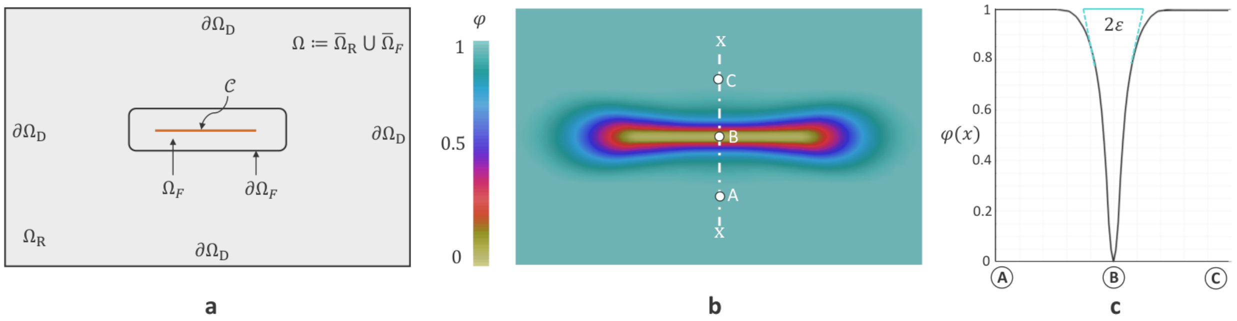

Figure 1: Prototype setup: the unbroken domain is denoted by Ω_R and $\mathcal{C}$ is the fracture. The latter is approximated by the domain Ω_F. The half thickness of Ω_F is ε. The fracture boundary is $\partial\Omega_F$ and the outer boundary is $\partial\Omega_D$. The corresponding realization using phase-field is shown in the right sub-figure. Here, the lower-dimensional fracture ($\varphi = 0$) is approximated with the phase-field variable. The transition zone with $0 < \varphi < 1$ has the thickness of ε on each side of the fracture. Consequently, Ω_F can be represented with the help of φ. Figure taken from [135].

Typical goals and questions The most immediate goals of the above idea is to address some of the following questions:

1. Can we compute an (a priori unknown) fracture path?

2. Can we work multiple fractures and fracture networks?

3. Does this fracture path depend on the numerical discretization method, mesh, etc.?

4. Do boundary conditions influence the fracture path?

5. Under which conditions does φ 'converge' to the lower-dimensional surface? (Hint: Γ-convergence; not covered in this lecture!)

6. What are possible constitutive relations for the governing equations?

7. How do we design 'good' numerical schemes in terms of feasibility, efficiency, and robustness?

8. What is the relationship between regularization (i.e., ε and later an elasticity regularization κ), discretization, and material parameters?

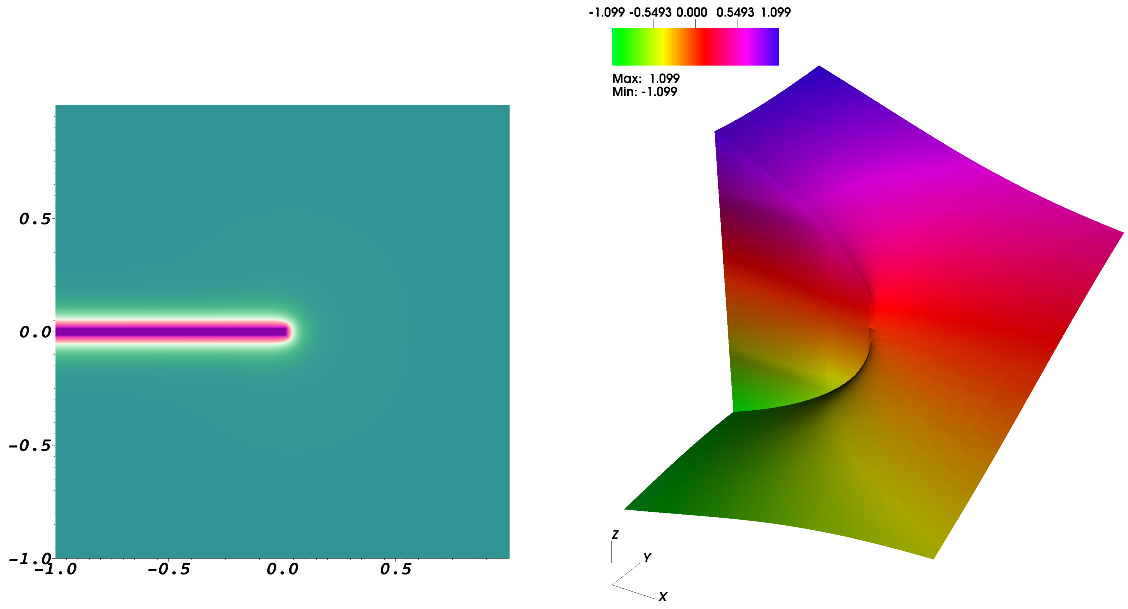

Figure 2: The crack is denoted in red color in the left snapshot of the phase-field. On the right side, a 3D plot of the displacement field shows the discontinuity along the line $(x, 0)$ for $-1 \le x \le 0$.

9. Is it possible to perform a rigorous numerical analysis of the proposed algorithms?

10. For which settings can we perform a mathematical analysis (well-posedness, a priori error estimations, and so forth)?

11. What are advanced possibilities to further enhance the accuracy while keeping the computational cost at a reasonable level?

12. What are typical functionals of interest, i.e., in what physical quantities (entire solution? Parts of a solution? Stress values? Local deflections and deformations?) is an engineer or practitioner interested in?

13. Is it possible to apply the method from ‘simple’ academic test cases to practical field problems?

14. Can we determine with sufficient accuracy crack tip mechanics and the fracture speed?

15. Can we compute the fracture width, the fracture volume, and fracture length with sufficient accuracy?

Advantages VPFF is a regularized approach that has (as many numerical methods) advantages and shortcomings. A list from the author's experience is as follows. The first advantage is a continuum description based on first physical principles to determine the unknown crack path [64] and the computation of curvi-linear and complex crack patterns. The model allows for nucleation, branching and merging and post-processing of certain quantities such as stress intensity factors become redundant. This in turn allows an easy handling of fracture networks (see Figure 3) in possibly highly heterogeneous media.

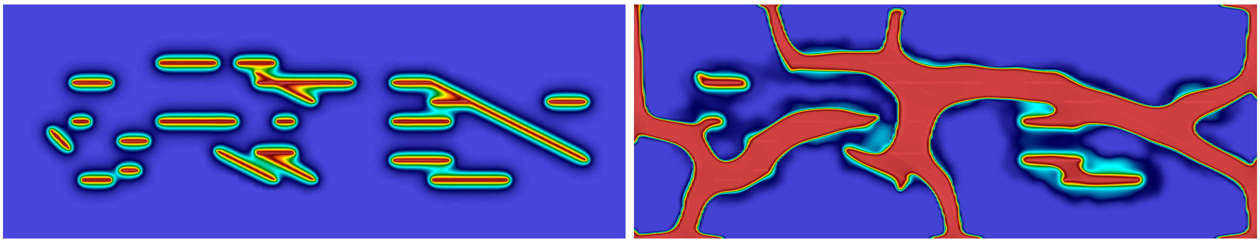

Figure 3: Fracture network (initial configuration at left) with growing fractures (at right) using a phase-field model. Red colors indicate the fracture ($\varphi = 0$) and blue colors the unbroken zone ($\varphi = 1$). The transition zone is indicated in yellow/green. This computation has been done with the model presented in [125].

The formulation described in a variational framework allows finite element discretizations and corresponding analyses. Phase-field is a fixed-mesh approach in which no re-meshing or update of basis functions to resolve the crack path is needed. The mathematical model permits any dimension $d = 2, 3$, and in case the software allows as well, phase-field fracture applies conveniently to three-dimensional simulations (see Figure 4).

Shortcomings

- The mesh (e.g., using finite elements) needs to resolve the interface to a certain accuracy, i.e., the relationship between spatial discretization parameter and phase-field regularization parameter.
- On the energy level, the formulation is non-convex which makes it challenging for both theory and design of numerical algorithms.
- A second challenge is the computational cost since either additional iterations in an alternating approach [25] are required or a fully nonlinear problem has to be solved. However, our quasi-monolithic approach [82, 106] has low Newton iteration numbers and therein the major cost goes into solving the vector-valued elasticity problem that also applies using most other crack models (such as extended/generalized finite elements for instance).
- Two bigger downsides are the accurate crack width computations, despite that some recent ideas have been proposed [107, 132].
- The smeared fracture transition zone when additional physics (in terms of Dirichlet or Neumann interface conditions) shall be described in, or around the fracture or on the fracture boundary. Here, local mesh adaptivity helps to a great extent in order to achieve sufficient accuracy but still will not track the crack boundary in an exact fashion.

As demonstrated in many studies (see Section 18), ideas have been proposed how to cope with these challenges. In view of the increasing popularity, it, however, seems that the advantages outweigh the shortcomings for today's applications.

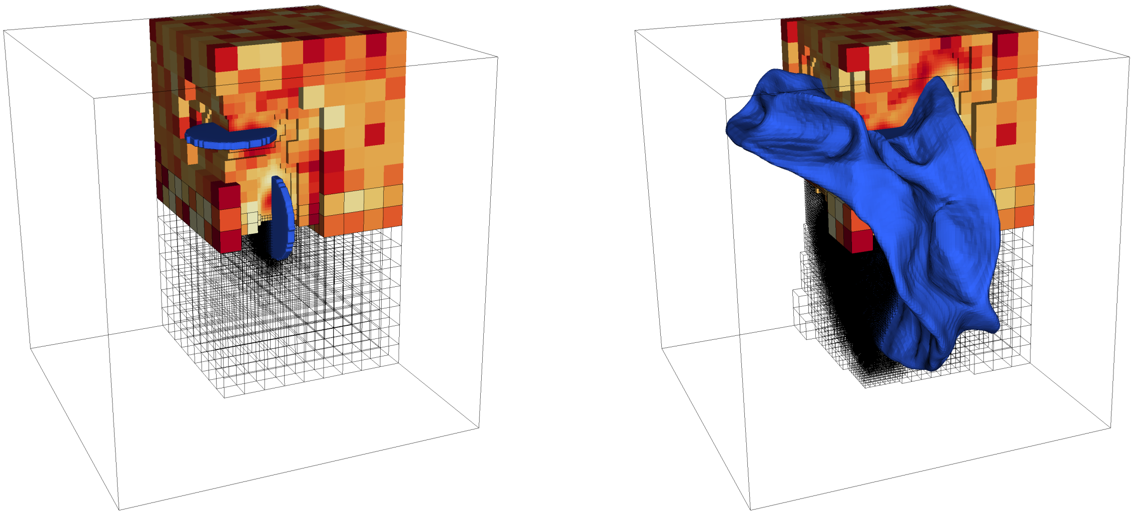

Figure 4: A 3D setting: two penny-shaped fractures grow, then join and later branch due to the heterogeneity of the solid. This simulation shows that phase-field can cope with complex 3D situations (of course the 'picture norm' is neither a rigorous proof nor a final evidence for correct numerics!). The computation was performed using high performance parallel computing and predictor-corrector local mesh adaptivity. The figure is an edited version taken from [179].

3.2 Background information on approximating interfaces and discontinuities

To approximate interfaces such as the solid discontinuity in fracture mechanics, there exist two basic methods:

- Interface-Tracking
- Interface-Capturing

In methods, where the domain is decomposed into elements or cells (finite volumes, finite elements or isogeometric analysis), using interface-tracking aligns mesh edges with the interface. These are so-called **fitted methods**. For moving interfaces, the mesh elements need to be moved as well; see Figure 5. However, mesh elements may be deformed too much such that the approach fails if not taken care of (expensive) re-meshing in a proper way.

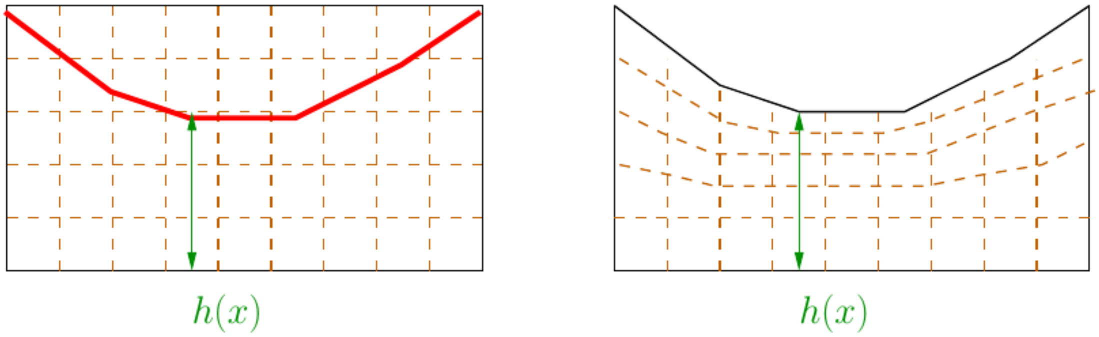

Figure 5: Left: the mesh is fixed and the interface must be captured. Right: interface-tracking in which the interface is located on mesh edges.

In interface-capturing methods (unfitted methods), the domain and consequently the single elements stay fixed; see Figure 5 (left). Here, the interface can move freely through the domain. Mesh degeneration is not a problem, but capturing the interface is difficult. In this approach a further classification can be made:

- Lower-dimensional approaches
- Diffusive techniques.

The first method comprises extended/generalized finite elements, cut-cell methods, finite cell methods, and locally modified finite elements. Diffusive methods are the famous level-set method or phase-field methods - the topic of this lecture notes.

Short summary Finally, using interface-tracking or interface-capturing approaches are a compromise between computational and implementation efforts and the accuracy of the desired interface approximation. While in general interface-capturing are easier to implement and can deal in an easier way with moving and evolving interfaces, the accuracy for the same number of degrees of freedom is lower than comparable interface-tracking approaches. The latter are, however, more challenging when interfaces are moving, propagating - in particular in 3D. To this end, as said just before: it is a compromise as many things in life.

3.3 An example of a practical problem: fracture and damage in screws

We briefly present in Figure 6 a situation in which phase-field fracture modeling was used to compare qualitatively with some experiments.

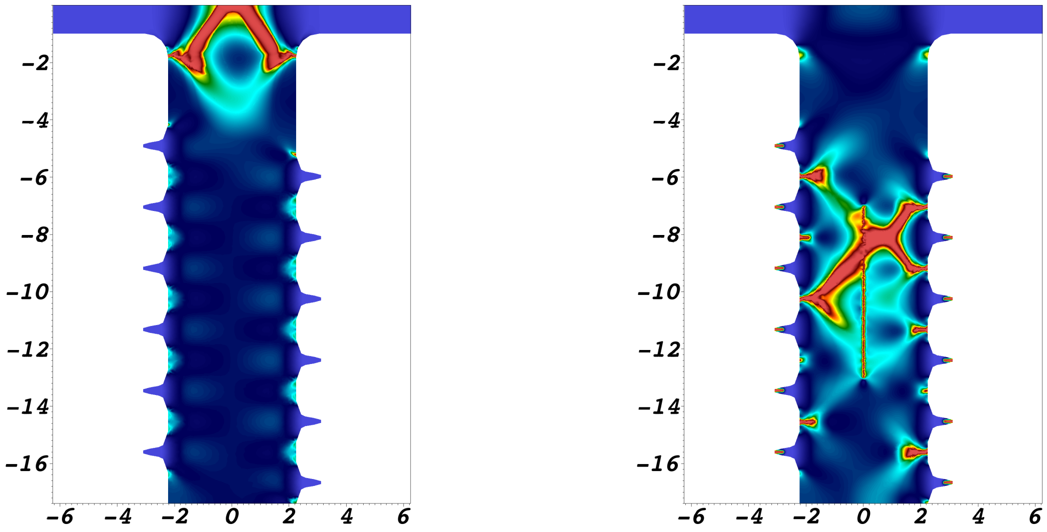

Figure 6: Left: Tension test of a screw. At the beginning (not shown here), the screw is completely intact and not damaged. Applying tensile forces on the head, while fixing at the bottom, high stresses develop in the region where the head is attached. Finally, the screw breaks (red fracture). Right: material inconsistencies inside the screw yield total damage in the body of the screw than at the head. Qualitative similar observations have been made in experiments. Details can be found in [170].

Exercise 1. *Collect own examples from real-life and practical applications in which fracture and damage problems arise.*

4 Notation and fundamental techniques

In this chapter, we provide the basic notation and some fundamental techniques. At the end a summary in form of a table is provided containing all important variables, parameters, quantities, and notation to look them up for later chapters.

4.1 Spatial dimension

Let $d \in \mathbb{N}$ be the spatial dimension. We use $\mathbb{R} = 1D$, $\mathbb{R}^2 = 2D$, and $\mathbb{R}^3 = 3D$.

4.2 Independent variables

A point in $\mathbb{R}^d$ is denoted by

$$x = (x_1, \dots, x_d).$$

The variable for 'time' is denoted by t. The euclidian scalar product is denoted by $(x, y) = x \cdot y = \sum_{i=1}^d x_i y_i$.

4.3 Function, vector and tensor notation

Functions are denoted by

$$u := u(x)$$

if they only depend on the spatial variable $x = (x_1, \dots, x_d)$. If they depend on time and space, they are denoted by

$$u := u(t, x).$$

In these notes, we consider phase-field fractures as quasi-static problems. For this reason, t is considered to be a **loading increment parameter** rather than a 'true' time.

Usually, in physics or engineering, vector-valued and tensor-valued quantities are denoted in bold font size or with the help of arrows. Unfortunately, in mathematics, this notation is only sometimes adopted. We continue this crime and do not distinguish scalar, vector, and tensor-valued functions. Thus for points in $\mathbb{R}^3$ we write:

$$x := (x, y, z) = \mathbf{x} = \vec{x}.$$

Similar for functions from a space $u : \mathbb{R}^3 \supseteq U \to \mathbb{R}^3$:

$$u := (u_x, u_y, u_z) = \mathbf{u} = \vec{u}.$$

And also similar for tensor-valued functions (which often have a bar or two bars under the tensor quantity), as for example the Cauchy stress tensor $\sigma_s \in \mathbb{R}^{3\times 3}$ of a solid, we write:

$$\sigma_s := \underline{\sigma}_s = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{pmatrix}.$$

4.4 Partial derivatives

We use:

$$\frac{\partial u}{\partial x} = \partial_x u,$$

and

$$\frac{\partial u}{\partial t} = \partial_t u,$$

and

$$\frac{\partial^2 u}{\partial t \partial t} = \partial_t^2 u,$$

and

$$\frac{\partial^2 u}{\partial x \partial y} = \partial_{xy} u.$$

4.5 Gradient, divergence, trace, Laplace, rotation

It is convenient to work with the **Nabla-operator** to define derivative expressions, as well-known in physics. The gradient of a single-valued function $v : \mathbb{R}^n \to \mathbb{R}$ reads:

$$\nabla v = \begin{pmatrix} \partial_1 v \\ \vdots \\ \partial_n v \end{pmatrix}.$$

The gradient of a vector-valued function $v : \mathbb{R}^n \to \mathbb{R}^m$ is called **Jacobian matrix** and reads:

$$\nabla v = \begin{pmatrix} \partial_1 v_1 & \dots & \partial_n v_1 \\ \vdots & & \vdots \\ \partial_1 v_m & \dots & \partial_n v_m \end{pmatrix}.$$

The divergence for vector-valued functions $v : \mathbb{R}^n \to \mathbb{R}^n$ is defined as:

$$\text{div } v := \nabla \cdot v := \nabla \cdot \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} = \sum_{k=1}^n \partial_k v_k.$$

The divergence for a tensor $\sigma \in \mathbb{R}^{n\times n}$ is defined as:

$$\nabla \cdot \sigma = \big(\sum_{j=1}^n \frac{\partial \sigma_{ij}}{\partial x_j}\big)_{1\le i\le n}.$$

The trace of a matrix $A \in \mathbb{R}^{n\times n}$ is defined as

$$tr(A) = \sum_{i=1}^n a_{ii}.$$

Definition 4.1 (Laplace operator)**.** *The Laplace operator of a two-times continuously differentiable scalar-valued function $u : \mathbb{R}^n \to \mathbb{R}$ is defined as*

$$\Delta u = \sum_{k=1}^n \partial_{kk} u.$$

Definition 4.2. *For a vector-valued function $u : \mathbb{R}^n \to \mathbb{R}^m$, we define the Laplace operator component-wise as*

$$\Delta u = \Delta \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^n \partial_{kk} u_1 \\ \vdots \\ \sum_{k=1}^n \partial_{kk} u_m \end{pmatrix}.$$

Let us also introduce the **cross product** of two vectors $u, v \in \mathbb{R}^3$:

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \times \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} u_2 v_3 - u_3 v_2 \\ u_3 v_1 - u_1 v_3 \\ u_1 v_2 - u_2 v_1 \end{pmatrix}.$$

With the help of the cross product, we can define the **rotation**:

$$\text{rot} v = \nabla \times v = \begin{pmatrix} \partial_x \\ \partial_y \\ \partial_z \end{pmatrix} \times \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} \partial_y v_3 - \partial_z v_2 \\ \partial_z v_1 - \partial_x v_3 \\ \partial_x v_2 - \partial_y v_1 \end{pmatrix}.$$

4.6 Vector spaces

Let $\mathbb{K} = \mathbb{R}$. In fact, $\mathbb{K} = \mathbb{C}$ would work as well as any general field. But we restrict our attention in the entire lecture notes to real numbers $\mathbb{R}$.

Definition 4.3 (Vector space)**.** *A **vector space** or **linear space** over a field $\mathbb{K}$ is a nonempty set X (later often denoted by V, U or also W). The space X contains elements $x_1, x_2, \ldots$, which are the so-called **vectors**. We define two algebraic operations:*

- *Vector addition: $x + y$ for $x, y \in X$.*
- *Multiplication of vectors with scalars: αx for $x \in X$ and $\alpha \in \mathbb{K}$.*

These operations satisfy the usual laws that they are commutative, associative, and satisfy the distributive laws.

4.7 Normed spaces

Let X be a linear space. The mapping $|| \cdot || : X \to \mathbb{R}$ is a **norm** if

i) $||x|| \geq 0 \quad \forall x \in X$ (Positivity)

ii) $||x|| = 0 \Leftrightarrow x = 0$ (Definiteness)

iii) $||\alpha x|| = |\alpha| \, ||x||, \quad \alpha \in \mathbb{K}$ (Homogeneity)

iv) $||x + y|| \leq ||x|| + ||y||$ (Triangle inequality)

A space X is a normed space if the norm properties are satisfied. If condition ii) is not satisfied, the mapping is called a **semi-norm** and denoted by $|x|_X$ for $x \in X$.

Definition 4.4. *Let $\| \cdot \|$ be a norm on X. Then $\{X, \| \cdot \|\}$ is called a (real) normed space.*

Example 4.5. *We provide some examples:*

1. *$\mathbb{R}^n$ with the Euclidian norm $\|x\| = (\sum_{i=1}^n x_i^2)^{1/2}$ is a normed space.*
2. *Let $\Omega := [a, b]$. The space of continuous functions $C(\Omega)$ endowed with the **maximum norm***
$$\|u\|_{C(\Omega)} = \max_{x \in \Omega} \|u(x)\|$$
is a normed space.
3. *The space $\{C(\Omega), \| \cdot \|_{L^2}\}$ with the norm*
$$\|u\|_{L^2} = \Big(\int_\Omega u(x)^2 \, dx \Big)^{1/2},$$
is a normed space.

Definition 4.6. *Two norms are equivalent if converging sequences have the same limits.*

Proposition 4.7. *Two norms $|| \cdot ||_A, || \cdot ||_B$ on X are equivalent if and only if there exist two constants $C_1, C_2 > 0$ such that*
$$C_1 ||x||_A \leq ||x||_B \leq C_2 ||x||_A \quad \forall x \in X.$$
The limits are the same.

Proof. See e.g., [168]. □

Remark 4.8. *This statement has indeed some immediate consequences. For instance, often convergence of an iterative scheme is proven with the help of the Banach fixed point scheme in which the contraction constant q must be smaller than 1. It is important that not all norms may satisfy $q < 1$, but when different norms are equivalent and we pick one that satisfies $q < 1$, we can proof convergence.*

4.8 Chain rule

Let the functions $g : (a,b) \to \mathbb{R}^{m+1}$ and $f : \mathbb{R}^{m+1} \to \mathbb{R}$ and their composition $h = f(g) \in \mathbb{R}$ be given and specifically $g := (t,x) := (t, x_1, x_2, x_3, \ldots, x_m)$:

$$\begin{aligned}
D_t h(x) &= D_t f(g(x)) = D_t f(t,x) = D_t f(t, x_1, x_2, \ldots, x_m) \\
&= \sum_{k=0}^{m} \partial_k f(g(x)) \cdot \partial_t g_k \\
&= \sum_{k=0}^{m} \partial_k f(t, x_2, \ldots, x_m) \cdot \partial_t x_k, \quad \text{where } x_0 := t \\
&= \partial_t f \cdot \partial_t t + \sum_{k=1}^{m} \partial_k f(t, x_2, \ldots, x_k) \cdot \partial_t x_k \\
&= \partial_t f + \nabla f \cdot (\partial_t x_1, \cdots, \partial_t x_m)^T.
\end{aligned}$$

For instance, $m = 3$ means that we deal with a four-dimensional continuum (t,x,y,z).

Remark 4.9. *See also [98][p. 54 and 93] for definitions of the chain rule.*

4.9 Transformation of integrals: substitution rule / change of variables

One of the most important formulas in continuum mechanics and variational formulations is the substitution rule that allows to transform integrals from one domain to another.

In 1D it holds:

Proposition 4.10 (Substitution rule in 1D)**.** *Let $I = [a,b]$ be given. To transform this interval to a new interval, we use a mapping $T(I) = [\alpha, \beta]$ with $T(a) = \alpha$ and $T(b) = \beta$. If $T \in C^1$ (a continuously differentiable mapping) and monotonically increasing (i.e., $T' > 0$), we have the transformation rule:*

$$\int_\alpha^\beta f(y)\,dy = \int_{T(a)}^{T(b)} f(y)\,dy = \int_a^b f(T(x))\,T'(x)\,dx.$$

Proof. Any real analysis (calculus) book. Here Analysis 2, Rolf Rannacher, Heidelberg University [141]. □

Remark 4.11. *In case that $T' < 0$ the previous Proposition still holds true, but with a negative sign:*

$$\int_\alpha^\beta f(y)\,dy = \int_{T(b)}^{T(a)} f(y)\,dy = -\int_{T(a)}^{T(b)} f(y)\,dy = \int_a^b f(T(x))\,(-T'(x))\,dx.$$

For both cases with $T' \neq 0$ the formula works and finally yields the following Theorem:

Theorem 4.12. *Let $I = [a,b]$ be given. To transform this interval to a new interval $[\alpha, \beta]$, we employ a mapping T. If $T \in C^1$ (a continuously differentiable mapping) and $T' \neq 0$, it holds:*

$$\int_{T(I)} f(y)\,dy := \int_\alpha^\beta f(y)\,dy = \int_a^b f(T(x))\,|T'(x)|\,dx =: \int_I f(T(x))\,|T'(x)|\,dx.$$

Proof. Any real analysis (calculus) book. Here Analysis 2, Rolf Rannacher, Heidelberg University [141]. □

Remark 4.13. *We observe the relation between the integration increments:*

$$dy = |T'(x)|\,dx.$$

Example 4.14. *Let T be an affin-linear transformation defined as*

$$T(x) = ax + b.$$

Then it holds,

$$dy = |a| dx.$$

In higher dimensions, we have the following result of the substitution rule (also known as **change of variables** under the integral):

Theorem 4.15. *Let $\Omega \subset \mathbb{R}^n$ be an open, measurable, domain. Let the function $T : \Omega \to \mathbb{R}$ be of class C^1, one-to-one (injective) and Lipschitz-continuous. Then:*

- *The domain $\widehat{\Omega} := T(\Omega)$ is measurable.*
- *The function $f(T(\cdot))|detT'(\cdot)| : \Omega \to \mathbb{R}$ is (Riemann)-integrable.*
- *For all measurable sub-domains $M \subset \Omega$ it holds the substitution rule:*

$$\int_{T(M)} f(y)\,dy = \int_M f(T(x))|detT'(x)|\,dx,$$

and in particular, as well for $M = \Omega$.

Proof. Any real analysis (calculus) book. See e.g., [141] or [98][Chapter 9]. □

Remark 4.16. *In continuum mechanics, T' is the so-called **deformation gradient** and $J := det(T')$ is called the **volume ratio**.*

4.10 Gauss-Green theorem / divergence theorem

The Gauss-Green theorem or often known as **divergence theorem**, is one of the most useful formulas in continuum mechanics and numerical analysis.

Let $\Omega \subset \mathbb{R}^n$ an bounded, open domain and $\partial\Omega$ of class C^1.

Theorem 4.17 (Gauss-Green theorem / divergence theorem)**.** *Suppose that $u := u(x) \in C^1(\bar{\Omega})$ with $x = (x_1, \ldots, x_n)$. Then:*

$$\int_\Omega u_{x_i}\,dx = \int_{\partial\Omega} un_i\,ds, \quad \text{for } i = 1, \ldots, n.$$

In compact notation, we have

$$\int_\Omega div\, u\,dx = \int_{\partial\Omega} u \cdot n\,ds$$

for each vector field $u \in C^1(\bar{\Omega}; \mathbb{R}^n)$.

Proof. The proof is nontrivial. See for example [98]. □

4.11 Integration by parts and Green's formula

An important concept is **integration by parts**.

From the divergence Theorem 4.17, we obtain immediately:

Proposition 4.18 (Integration by parts)**.** *Let $u, v \in C^1(\bar{\Omega})$. Then:*

$$\int_\Omega u_{x_i} v\,dx = -\int_\Omega uv_{x_i}\,dx + \int_{\partial\Omega} uvn_i\,ds, \quad \text{for } i = 1, \ldots, n.$$

In compact notation:

$$\int_\Omega \nabla uv\,dx = -\int_\Omega u\nabla v\,dx + \int_{\partial\Omega} uvn\,ds.$$

Proof. Use this proof as an exercise. Apply the divergence theorem to uv. □

We obtain some further results, which are very useful, but all are based directly on the integration by parts. For this reason, it is even more important to know the divergence theorem and integration by parts formula.

Proposition 4.19 (Green's formula)**.** *Let $u, v \in C^2(\bar{\Omega})$. Then it holds:*

$$
\begin{aligned}
\int_\Omega \Delta u \, dx &= \int_{\partial\Omega} \partial_n u \, ds, \\
\int_\Omega \nabla u \cdot \nabla v \, dx &= -\int_\Omega \Delta u \, v \, dx + \int_{\partial\Omega} v \, \partial_n u \, ds.
\end{aligned}
$$

Proof. Apply integration by parts. □

Proposition 4.20 (Green's formula in 1D)**.** *Let $\Omega = (a, b)$. Let $u, v \in C^2(\bar{\Omega})$. Then:*

$$
\int_\Omega u'(x) \cdot v'(x) \, dx = -\int_\Omega u''(x) \, v(x) \, dx + [u'(x)v(x)]_{x=a}^{x=b}
$$

Proof. Apply integration by parts. □

4.12 Fundamental lemma of calculus of variations

This lemma provides tools to transfer a variational (weak) formulation to the corresponding strong form. For detailed descriptions including proofs, we refer to [46]. To this end, we introduce

Definition 4.21 (Continuous functions with compact support)**.** *Let $\Omega \subset \mathbb{R}^n$ be an open domain.*

- *The set of continuous functions from Ω to $\mathbb{R}$ with **compact support** is denoted by $C_c(\Omega)$. Such functions vanish on the boundary.*
- *The set of **smooth functions** (infinitely continuously differentiable) with compact support is denoted by $C_c^\infty(\Omega)$.*

Proposition 4.22 (Fundamental lemma of calculus of variations in 1D)**.** *Let $\Omega = [a, b]$ be a compact interval and let $w \in C(\Omega)$. Let $\phi \in C^\infty$ with $\phi(a) = \phi(b) = 0$, i.e., $\phi \in C_c^\infty(\Omega)$. If it holds*

$$
\int_\Omega w(x)\phi(x) \, dx = 0, \quad \forall \phi \in C^\infty,
$$

then, $w \equiv 0$ in Ω.

Proof. We perform an indirect proof. We suppose that there exists a point $x_0 \in \Omega$ with $w(x_0) \neq 0$. Without loss of generality, we can assume $w(x_0) > 0$. Since w is continuous, there exists a small (open) neighborhood $\omega \subset \Omega$ with $w(x) > 0$ for all $x \in \omega$; otherwise $w \equiv 0$ in $\Omega \setminus \omega$. Let ϕ now be a positive test function (recall that ϕ can be arbitrary, specifically positive if we wish) in Ω and thus also in ω. Then:

$$
\int_\Omega w(x)\phi(x) \, dx = \int_\omega w(x)\phi(x) \, dx.
$$

But this is a contradiction to the hypothesis on w. Thus $w(x) = 0$ for all in $x \in \omega$. Extending this result to all open neighborhoods in Ω we arrive at the final result. □

Proposition 4.23 (Fundamental lemma of calculus of variations in $\mathbb{R}^n$)**.** *Let $\Omega \subset \mathbb{R}^n$ be an open domain and let w be a continuous function. Let $\phi \in C^\infty(\Omega)$ have a compact support in Ω. If*

$$
\int_\Omega w(x)\phi(x) \, dx = 0 \quad \forall \phi \in C^\infty(\Omega),
$$

then, $w \equiv 0$ in Ω.

Proof. Similar to the 1D version. □

4.13 Domains

- Let $B = \bar{\Omega} \cup \bar{C} \subset \mathbb{R}^d, d = 1, 2, 3$ be an open, bounded and connected domain.
- Let $\partial\Omega$ be a smooth boundary of B or a polyhedral domain.
- The boundary may be split into four parts: $\partial\Omega = \partial\Omega_0 \cup \partial\Omega_1 \cup \partial\Omega_2 \cup \partial\Omega_D$.
- The domain B consists of an unbroken part and a broken part. The latter contains the crack or fracture C, a lower dimensional object with $C \subset \mathbb{R}^{d-1}$.

The idea of a variational regularized phase-field consists of approximating $C \in \mathbb{R}^{d-1}$ by a *thick* surface of the width ϵ in all directions in $\mathbb{R}^d$. An illustration is provided in Figure 7. When ϵ tends to zero, the hope is that the original fracture C is recovered. The mathematical technology is the so-called Γ-convergence, e.g., [33].

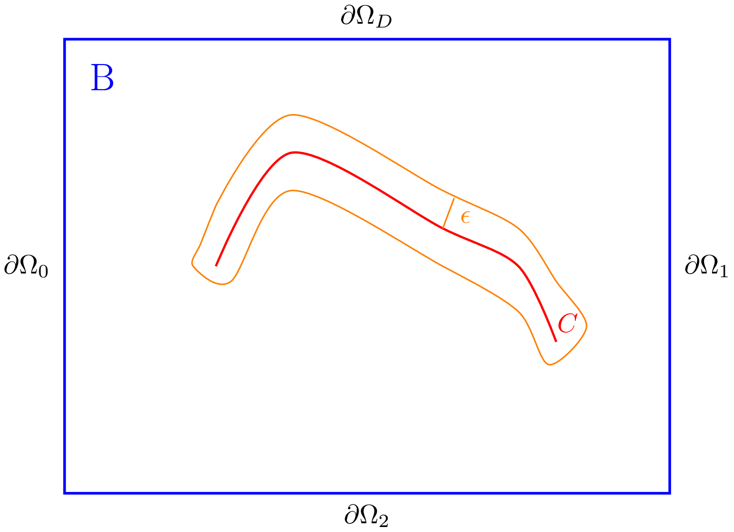

Figure 7: Simple setup of a fracture $C \in \mathbb{R}^{d-1}$ with a mushy zone of width 2ϵ and the boundaries $\partial\Omega_D$, $\partial\Omega_0$, $\partial\Omega_1$ and $\partial\Omega_2$.

A typical setup for the displacement boundary conditions could be:

- $u = 0$ on $\partial\Omega_2$ (homogeneous Dirichlet; domain fixed);
- $\partial_n u = 0$ on $\partial\Omega_0$ and $\partial\Omega_1$ (homogeneous Neumann; traction free);
- $u = u_D$ on $\partial\Omega_D$ (non-homogeneous Dirichlet; pulling apart the domain; tension).

4.14 Solution variables in a phase-field fracture setting

The unknown solution variables are:

- vector-valued displacements $u : B \to \mathbb{R}^d$
- a smoothed scalar-valued indicator phase-field function $\varphi : B \to [0, 1]$.

The latter describes the crack path in a smeared fashion. Here,

- $\varphi = 0$ denotes the crack region
- $\varphi = 1$ characterizes the unbroken material
- $0 < \varphi < 1$ are intermediate values constituting a smooth transition zone dependent on the regularization parameter ε. In engineering or physics, ε is often a so-called length-scale parameter. This may be justified since this zone weakens the material and is a physical transition zone from the unbroken material to a fully damaged state.

4.15 Summary of variables, parameters, and constitutive quantities

SYMBOL	*DESCRIPTION*	*UNIT*
d	Dimension	
x	Position	m
t	Time; loading step in quasi-static (incremental) problems	s
h	Spatial discretization parameter	m
B	Total domain: $B = \overline{\Omega} \cup \overline{C}$	m^3
C	Fracture (lower-dimensional)	m^2
I	Time/loading interval	s
ε	Phase-field regularization parameter	m
κ	Phase-field regularization parameter in the bulk	dimensionless
u	Displacements	m
φ	Phase-field variable	dimensionless
∇u	Gradient of displacements	dimensionless
Δu	$\Delta u = \nabla \cdot \nabla u$: Laplacian	1/m
$e(u)$	$e = e(u) = \frac{1}{2}(\nabla u + \nabla u^T)$, linearized strain tensor	dimensionless
$\partial_t u$	$\partial_t u = v$: velocity (time-derivative displacements)	m/s
$\partial_t \varphi$	Time-derivative phase-field	1/s
$E(\cdot)$	Energy functional	J
U	Total solution vector, often in these notes: $U = (u, \varphi)$	
$A(\cdot)(\cdot)$	Semi-linear form to express weak/variational forms	
(x, y)	Scalar product: $\int_B x \cdot y \, dB$ with $x, y \in \mathbb{R}^d$	
(A, C)	Scalar product: $\int_B A : C \, dB$ with $A, C \in \mathbb{R}^{d \times d}$	
p	In general: pressure	$\mathrm{Pa} = \mathrm{N/m^2} = \mathrm{kg/ms^2}$
p	Obstacle problem: Lagrange multiplier	problem-dependent
∇p	Gradient of pressure	$\mathrm{Pa/m} = \mathrm{N/m^3} = \mathrm{kg/m^2s^2}$
n	Normal vector	m
I	Identity matrix/tensor	dimensionless
$\sigma \cdot n$	Traction force	$\mathrm{Pa\,m} = \mathrm{N/m}$
ρ	Density	kg/m^3
f	Force (such as gravity)	$\mathrm{N/kg} = \mathrm{m/s^2}$
σ_s	Cauchy solid stress tensor	$\mathrm{Pa/m^2}$
G	Energy release rate	$\mathrm{J/m^2} = \mathrm{N/m}$
μ	Lamé parameter / shear modulus	$\mathrm{Pa} = \mathrm{N/m^2} = \mathrm{kg/ms^2}$
λ	Lamé parameter	$\mathrm{Pa} = \mathrm{N/m^2} = \mathrm{kg/ms^2}$
ν_s	Poisson's ratio	dimensionless
E_Y	Young's modulus	$\mathrm{Pa} = \mathrm{N/m^2} = \mathrm{kg/ms^2}$
W	Strain energy density functional per unit volume	$\mathrm{J/m^3} = \mathrm{N/m^2} = \mathrm{Pa}$

5 Modeling of variational phase-field fractures

In this chapter, we concentrate on the modeling of variational phase-field fractures. The original work, forming the basis, can be found in [27, 28, 64] and [117, 121].

The goal will be the modeling of phase-field fractures in a variational form. As already mentioned in Chapter 3, there is a wide field of interest and applications of fracture simulation that motivates the numerical formulation of phase-field fractures. First, the problem is formulated in strong form and the historical milestones are listed and discussed.

5.1 A first simplified phase-field model

To determine both solution variables u and φ we need two equations. Displacements can be computed with the help of linearized elasticity, e.g., [44]. For the moment, however, we work with the Laplacian to make it as simple as possible: Find $u : \Omega \to \mathbb{R}$ such that

$$-\nabla \cdot (a\nabla u) = f \quad \text{in } \Omega, \quad \text{plus bc on } \partial\Omega, \tag{1}$$

where 'bc' = 'boundary conditions' and with a material coefficient $a > 0$. Soon, we will see that in phase-field fracture, the parameter a depends on $x \in \Omega$ as well as on a second solution variable.

The phase-field function φ describes the crack path ($\varphi = 1$ for unbroken solid and $\varphi = 0$ for a fully damaged material) and is in the limit a lower-dimensional manifold. Here, [27] proposed to employ an Ambrosio-Tortorelli elliptic functional (1990/1992) [9, 10]. Further details will follow later. For now, we are satisfied to know that we work with an elliptic functional, whose corresponding PDE is very familiar to us (again a Laplacian with a reaction term and a parameter $\epsilon > 0$): Find $\varphi : \Omega \to \mathbb{R}$ such that

$$-\epsilon\Delta\varphi - \frac{1}{\epsilon}(1-\varphi) = g \quad \text{in } \Omega, \quad \text{plus bc on } \partial\Omega, \tag{2}$$

where $g : \Omega \to \mathbb{R}$ is a right hand side force.

However, an additional constraint is the crack irreversibility (the crack cannot heal in time), which is imposed as:

$$\partial_t\varphi \leq 0. \tag{3}$$

Consequently, the equation determining φ becomes an inequality and is linked to the constraint (3) via a compatibility condition. Therefore, the system is very close to the well-known and well-studied **obstacle problem**, see e.g., the famous books [95, 96]. However, the equations themselves are stationary, but the inequality constraint is a time-dependent condition, which differs from the standard obstacle problem.

By means of these definitions, a **simplified** phase-field fracture problem can be formulated as follows:

Formulation 5.1 (A simplified strong problem formulation)**.** *Find a displacement function $u : B \times I \to \mathbb{R}^d$ and a phase-field indicator function $\varphi : B \times I \to [0,1]$, where $I := (0,T]$ is the 'time'/loading interval, such that*

$$-\nabla \cdot (\varphi^2\nabla u) = f \quad \textit{in } B \times I, \quad (u\textit{-equation}) \tag{4}$$

$$\varphi|\nabla u|^2 - \epsilon\Delta\varphi - \frac{1}{\epsilon}(1-\varphi) \leq 0 \quad \textit{in } B \times I, \quad (\varphi\textit{-equation}) \tag{5}$$

$$\partial_t\varphi \leq 0 \quad \textit{in } B \times I, \quad (\textit{crack irreversibility}) \tag{6}$$

$$\Big[\varphi|\nabla u|^2 - \epsilon\Delta\varphi - \frac{1}{\epsilon}(1-\varphi)\Big] \cdot \partial_t\varphi = 0 \quad \textit{in } B \times I. \quad (\textit{compatibility condition}) \tag{7}$$

To formulate a well-posed problem, boundary and initial conditions are needed:

$$\begin{aligned}
u(x,t) &= u_D(x,t) && \textit{on } \partial\Omega_D \times I,\\
u(x,t) &= 0 && \textit{on } \partial\Omega_2 \times I,\\
\varphi^2\nabla u \cdot n &= 0 && \textit{on } (\partial\Omega_1\partial\Omega_0) \times I,\\
\partial_n\varphi &= 0 && \textit{on } \partial B \times I,\\
\varphi(x,0) &= \varphi_0 && \textit{on } B \times \{0\},
\end{aligned}$$

with an initial fracture φ_0 and with $\epsilon > 0$ as the so-called phase-field regularization parameter. The boundaries are set as in Figure (8).

Remark 5.2 (Coupling terms)**.** *Observing* (1) *and* (2)*, we obtain Formulation 5.1 by setting $a = \varphi^2$ and $g = -\varphi|\nabla u|^2$.*

Remark 5.3 (Boundary conditions)**.** *The first two boundary conditions on u are non-homogeneous and homogeneous Dirichlet conditions, respectively. The third condition is a homogeneous Neumann condition for the displacements. The fourth condition is a homogeneous Neumann condition for phase-field and the last condition is the initial condition for phase-field.*

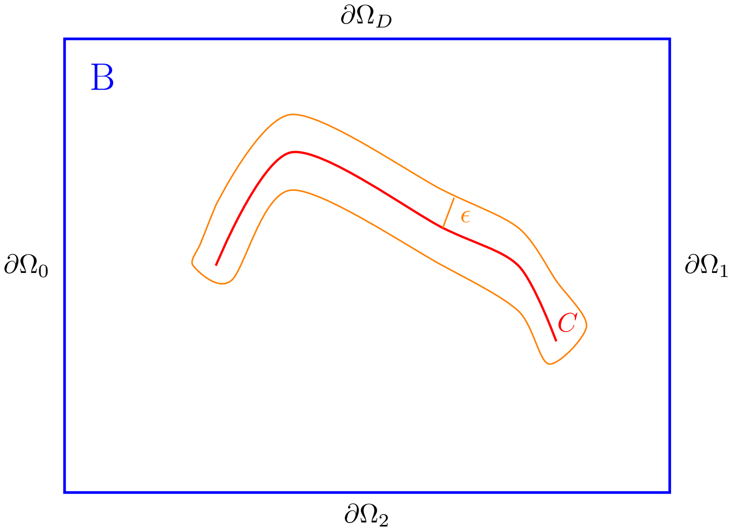

Figure 8: Simple sketch of an initial crack C in a domain B.

Remark 5.4. *Because the main terms* (4) *and* (5) *are not time-dependent, but the constraint* (6) *depends on time, the irreversibility constraint is discretized in time to get a quasi-static formulation via:*

$$\frac{\varphi^{n+1} - \varphi^n}{k},$$

where $\varphi^{n+1} := \varphi(t_{n+1})$ and the time step size $k = t_{n+1} - t_n$. This implies that for $\varphi^{n+1} \leq \varphi^n$, and it must hold for physical modeling assumptions that $\varphi^{n+1} < \varphi^n$, so the fracture(s) increase(s) or stay(s) in its length, but cannot heal.

5.1.1 Comments on the simplified phase-field problem Formulation 5.1

The strong problem formulation gives the basic system of all ongoing considerations and should be understood in detail. This is why we give some useful comments in the following:

1) System (4) to (7) shares many similarities with the obstacle problem. The Laplace problem with initial values can be formulated in $1D$ as

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega = (0,1), \\ u(0) &= u(1) = 0. \end{aligned}$$

The obstacle problem is defined as

$$\begin{aligned} -\Delta u &\geq f \quad \text{in } \Omega, \\ u &\geq g \quad \text{in } \Omega, \\ (f + \Delta u)(u - g) &= 0 \quad \text{in } \Omega. \end{aligned}$$

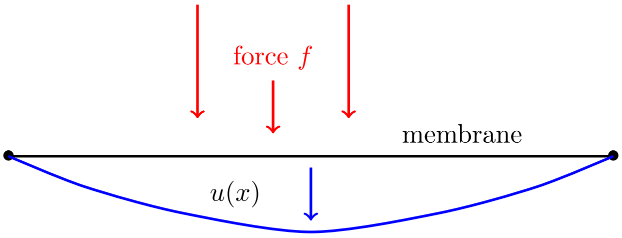

Figure 9: Elastic membrane. Deformation $u(x)$ induced by a force f.

The obstacle problem is a free boundary problem that splits the domain Ω into two parts $\mathcal{N}$ and $\mathcal{A}$. If $f + \Delta u = 0$, then we solve the PDE and it holds $u > g$ in the so-called non-active set $\mathcal{N}$, where the obstacle condition is not active. On the other hand, if $f + \Delta u < 0$, we are 'sitting' on the obstacle, i.e., $u = g$, and we are in the active set $\mathcal{A}$.

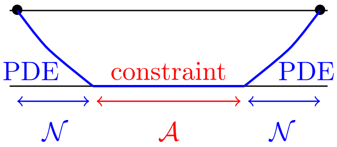

Figure 10: Obstacle problem: an elastic membrane touches the table in the active set $\mathcal{A}$ in which the obstacle condition is active. In the inactive set $\mathcal{N}$, the PDE has to be solved.

2) For a correct spatial discretization, ϵ must be bigger than h, i.e., $h = o(\epsilon)$; see Section 7.3.2. For computational investigations we refer to [83] and [173]. In physics or engineering, ϵ is often called the length-scale parameter.

3) System (4) to (7) is nonlinear (i.e. quasi-linear), in a monolithic fashion, such that $\{u, \varphi\}$ are solved simultaneously. Keep in mind, that if we decouple the system, the single expressions (4) and (5) become linear. However, (5) - (7) forms an inequality and therefore, we nevertheless deal with a nonlinear problem since the underlying space is only a set and not anymore a linear space.

4) Formulation 5.1 is a time-dependent, quasi-linear and a variational inequality system.

5) The term $-\nabla \cdot (\varphi^2 \nabla u)$ reads in a variational formulation $(\varphi^2 \nabla u, \nabla w)$ for all admissible test functions w. We introduce the notation

$$E'_R(u, \varphi)(w) := (\varphi^2 \nabla u, \nabla w) = \int_B \varphi^2 \nabla u \nabla w \, dx.$$

Integrating yields the energy of the system:

$$E_R(u, \varphi) = \frac{1}{2} \int_B \varphi^2 |\nabla u|^2 \, dx. \tag{8}$$

Considering φ and u simultaneously as solution variables (a so-called monolithic formulation), then the integral is of 4th order. The corresponding Hessian E'' is indefinite. This makes it difficult for the numerical nonlinear solution. Of course, as we easily observe, fixing one variable, yields a standard 2nd order problem, where we know how to solve it. The main question in such an approach is whether this iterating procedure converges and if yes, what can be said about the order of convergence.

Exercise 2. *Similar to* (8)*, we can define the following functional in* $\mathbb{R}^2$*: Take* $F : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *with*

$$F(x, y) = x^2 y^2$$

1. Compute ∇F *and* $H_F(x, y)$ *and compute the stationary points.*

2. *Design a numerical scheme to solve the minimization problem:*

$$\min F(x, y)$$

Hint: For Newton methods solving similar problems, see [92] or [177].

5.2 Pros and Cons of variational phase-field fracture modeling

The proposed variational phase-field fracture approach is not the only way to formulate a fracture problem (other methods are listed in the introduction of [180]), but there are good reasons to use it as we described in Chapter 5.

The following list shows some pros and cons of the VPFF approach.

PROS

- the continuum description is based on first physical principles (also holds true for other fracture approaches though)
- VPFF enables the computation of unknown crack paths
- VPFF allows curvilinear, complex crack patterns, nucleation, branching and merging, which are the most striking features (see Chapter 3)
- the variational formulation is given on a fixed mesh, so there is no need to update basis functions in a special fashion or to align the mesh
- VPFF is also *relatively* simple to realize in three dimensions (see Figure 4 and hints for the computational cost in [83])
- the Galerkin FEM can be used, where error estimations and mesh adaptivity methods etc. do already exist

CONS

- the energy functional is non-convex in u and φ
- the fully-coupled system is quasi-linear
- Small h is required because $h \ll \varepsilon$
- For accurate investigations of tip phenomena (e.g., branching) phase-field might be too inaccurate because stress approximations and stress intensity factors may require numerical methods with high-accuracy
- smeared crack interfaces are challenging if interface laws from physics shall be described on the crack boundary
- computation of the crack width defined as $w := [u \cdot n]$ is challenging

5.3 Excursus I: the obstacle problem

To deepen the understanding of modeling a variational phase-field fracture approach, let us go back one step and observe an elastic membrane touching a table which is illustrated in Figure 10. For the mathematical settings we refer again to [95, 96].

Let $\Omega \subset \mathbb{R}^n$. The contact boundary $\Gamma := \partial\mathcal{N} \cap \partial\mathcal{A}$ between the inactive set $\mathcal{N}$ and the active set $\mathcal{A}$ is a priori known (see again Figure 10). We seek displacements $u : \Omega \to \mathbb{R}$. The potential energy $E(u)$ can be defined as

$$E(u) = \int_\Omega [\mu(\sqrt{1 + |\nabla u|^2} - 1) - fu] \, dx \overset{Taylor}{\approx} \int_\Omega [\frac{\mu}{2}|\nabla u|^2 - fu] \, dx,$$

with a material parameter $\mu > 0$.

Formulation 5.5 (Physical state)**.** *Physically, the state without any constraint can be formulated as*

$$\min_{u \in V} E(u) \text{ with } V := \{v \in H^1(\Omega) | v = u_0 \text{ on } \partial\Omega\}.$$

Formulation 5.6. *The minimization problem with a constraint can be written as*

$$\min_{u \in V, u \geq g} E(u), \quad \text{with } g \in L^2.$$

The admissible function space can be defined as follows:

$$K := \{v \in H^1(\Omega) | v = u_0 \text{ on } \partial\Omega, v \geq g \text{ in } \Omega\},$$

such that we can write:

$$\min_{u \in K} E(u).$$

Definition 5.7 (Convex set)**.** *We say K is a convex set if*

$$u, v \in K : \quad \theta u + (1-\theta)v \in K.$$

for $0 \leq \theta \leq 1$.

Definition 5.8 (Convex functional)**.** *A functional $E : K \to \mathbb{R}$ is convex if and only if*

$$E(\theta u + (1-\theta)v) \leq \theta E(u) + (1-\theta)E(v)$$

for all $u, v \in K$ and $0 \leq \theta \leq 1$.

Using these definitions, we obtain:

Formulation 5.9.

$$E(u) = \min_{v \in K} E(v),$$

where $u \in K$ is the minimum of the energy functional.

5.3.1 Variational formulation

We derive a variational formulation in this section. Let $\theta v + (1-\theta)u = u + \theta(v-u) \in K$ with $\theta \in [0,1]$. Then, we can follow by using the previously defined convex set properties:

$$\begin{aligned}
E(u + \theta(v-u)) &\geq E(u) \\
\Rightarrow \frac{d}{d\theta} E(u + \theta(v-u))|_{\theta=0} &\geq \frac{d}{d\theta} E(u), \quad \text{(notice that } E(u) \text{ has no } \theta \text{ dependence)} \\
\Leftrightarrow \frac{d}{d\theta} E(u + \theta(v-u))|_{\theta=0} &\geq 0 \\
\Rightarrow \frac{d}{d\theta} \int_\Omega (\mu \frac{1}{2} |\nabla(u + \theta(v-u))|^2 - f(v-u))\, dx|_{\theta=0} &\geq 0 \\
\Rightarrow \int_\Omega (\mu \nabla(u + \theta(v-u)) \cdot \nabla(v-u) - f(v-u))\, dx|_{\theta=0} &\geq 0 \\
\Rightarrow \int_\Omega (\mu \nabla u \cdot \nabla(v-u) - f(v-u))\, dx &\geq 0 \\
\Leftrightarrow \int_\Omega \mu \nabla u \cdot \nabla(v-u)\; dx - \int_\Omega f(v-u)\; dx &\geq 0 \\
\Leftrightarrow \int_\Omega \mu \nabla u \cdot \nabla(v-u)\; dx - \int_\Omega f(v-u)\; dx &\geq 0 \quad \forall v \in K.
\end{aligned}$$

In short hand notation (see Section 4.15), the last line is usually written as:

$$(\mu \nabla u, \nabla(v-u)) \geq (f, v-u).$$

With these calculations, we obtain the following variational form:

Formulation 5.10 (Variational formulation of the obstacle problem)**.** *The obstacle problem in a variational PDE formulation is given by: Find $u \in K$ such that*

$$(\mu \nabla u, \nabla(v-u)) \geq (f, v-u),$$

for $K = \{v \in H^1 | v = u_0$ on $\partial\Omega,\ v \geq g$ in $\Omega\}$.

In the contact problem of a membrane, we have an active set $\mathcal{A} = \Omega \backslash \mathcal{N}$, where the membrane touches the table with $\mathcal{A} = \{x \in \Omega : \; u = g\}$.The active set is also called contact zone . The inactive set is called $\mathcal{N}$ and the free boundary is defined as $\Gamma := \partial\mathcal{A} \cap \partial\mathcal{N}$. We define the sub-regions as

- $\mathcal{A} = \Omega \setminus \mathcal{N}$: active set or contact zone (we sit on the obstacle), defined by
$$\mathcal{A} = \{x \in \Omega : \; u = g\}$$
- $\mathcal{N}$ is the inactive set (we solve the PDE)
- $\Gamma = \partial\mathcal{A} \cap \partial\mathcal{N}$ is the free boundary

5.3.2 Strong formulation

Using the fundamental lemma of calculus of variations (e.g., [46]), we derive the strong form:

Formulation 5.11 (Strong form)**.** *Find $u : \Omega \to \mathbb{R}$, such that*

$$\begin{aligned} -\nabla \cdot (\mu \nabla u) &\geq f && \textit{in } \Omega, \\ u &\geq g && \textit{in } \Omega, \\ (\nabla \cdot (\mu \nabla u) + f)(u - g) &= 0 && \textit{in } \Omega, \\ u &= g && \textit{on } \partial\Omega, \\ u &= g && \textit{on } \Gamma, \\ \mu \nabla u \cdot n &= \mu \nabla g \cdot n && \textit{on } \Gamma. \end{aligned}$$

Remark 5.12 (Complementary condition)**.** *The complementary condition*

$$(\nabla \cdot (\mu \nabla u) + f)(u - g) = 0,$$

is a necessary condition since otherwise the first two inequalities have no direct relationship. In the variational form this is ensured by the function space K.

5.3.3 Lagrange multiplier formulations

Often, it is of interest to work directly with the linear function space V rather than with the closed convex set K. The prize to pay is a second solution variable (which is more expensive in the solution approach though). To this end, a Lagrange multiplier $p \in L^2$ as an additional solution variable is introduced:

$$p = \begin{cases} 0 & \text{if } x \in \mathcal{N} \\ -\nabla \cdot (\mu \nabla u) - f & \text{if } x \in \mathcal{A} \end{cases}$$

With this definition it also becomes clear that the physical meaning of the Lagrange multiplier is a force in this problem. This force acts against the PDE in order to fulfill the constraint $u \geq g$.

Using the Lagrange multiplier, the strong form reads:

Formulation 5.13 (Lagrange multiplier strong form)**.** *Find $u : \Omega \to \mathbb{R}$ and $p : \Omega \to \mathbb{R}$, such that*

$$\begin{aligned} -\nabla \cdot (\mu \nabla u) - p &= f && \textit{in } \Omega, \\ u &\geq g && \textit{in } \Omega, \\ p &\geq 0 && \textit{in } \Omega, \\ p(u - g) &= 0 && \textit{in } \Omega, \\ u &= u_0 && \textit{on } \partial\Omega, \\ u &= g && \textit{on } \Gamma, \\ \mu \nabla u \cdot n &= \mu \nabla g \cdot n && \textit{on } \Gamma. \end{aligned}$$

Remark 5.14. *This formulation is clearly more expensive since the Lagrange multiplier is considered to be a solution variable. Nonetheless, a recent implementation for phase-field fracture was done in [112].*

We now turn to the weak setting:

Formulation 5.15 (Lagrange multiplier weak form)**.** *The weak form reads formally: Find* $\{u, p\} \in V \times N$*:*

$$(\mu \nabla u, \nabla \phi) - (p, \phi) = (f, \phi) \quad \forall \phi \in V$$
$$(u - g, q - p) \geq 0 \quad \forall q \in N$$

where $N := \{q \in Q^* | q \geq 0\}$ *with* Q^* *being the dual space of a Hilbert space* Q*. For details see [95][p. 38ff].*

Exercise 3. *Formulate a Lagrange multiplier formulation for the simplified phase-field Formulation 5.1.*

5.3.4 Brief introduction to one possibility to treat the obstacle constraint

In practice, we face the question, how to realize the obstacle constraint. One possibility (in its basic form not the best!) is penalization, which is a well-known technique in nonlinear programming, e.g., [158, 159]. The idea is to asymptotically fulfill the constraint by including an additional term that acts against the optimization goal if the constraints are violated.

We introduce the penalty parameter $\rho > 0$ (and have $\rho \to \infty$ in mind for later). As before: Find $u \in K$:

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

A penalized version reads: Find $u_\rho \in H_0^1$:

$$(\mu \nabla u_\rho, \nabla v) - \rho \int_\Omega [g - u]_+ v \, dx = (f, v) \quad \forall v \in H_0^1.$$

Here, $[x]_+ = \max(x, 0)$.

Indeed we have

- $u_\rho \geq g$ yields $0 \geq g - u$. Thus $[g - u]_+ = 0$
- $u_\rho < g$ yields $0 < g - u$. Thus $[g - u]_+ > 0$

Remark 5.16 (Ill-conditioning)**.** *For large* ρ *(which are necessary to enforce the constraint), the system matrix becomes ill-conditioned, i.e., the condition number is large since some entries are zero and others have their regular values. Therefore, the stability of the discrete system is heavily affected resulting in a significant error propagation (e.g., round-off errors due to machine precision) in the linear and nonlinear solution. Consequently, one has to find a trade-off between sufficiently small and large* ρ*. An extension is an augmented Lagrangian method or active set method. A detailed discussion can be found for instance in [133].*

5.4 Preliminaries to modeling quasi-static brittle fracture

To follow the development of phase-field fracture models, we start with the model of Griffith (1920) [73], who was one of the first who developed a model to compute **brittle fractures**. His idea is depicted in this section. Years later, the great achievement of Francfort and Marigo [64] was to relate a variational model (formulated in terms of the energy) to Griffith's model. Our plan is to proceed as follows: we first provide some preliminary explanation in this section and explain Griffith's historical model in Section 5.5. Then, we recapitulate the variational model proposed by Francfort and Marigo in Section 5.6.

Definition 5.17 (Critical energy release rate)**.** *The toughness (in this context better:* ***critical energy release rate****)* G_C *is a fundamental quantity that describes the resistance of a material to break. Or in other words,* G_C *is the energy required to create an infinitesimal crack at a point* $x \in \bar{B}$*, where* B *is the domain as defined before.*

Definition 5.18 (Current energy release rate)**.** *Beside the critical energy release rate it is of interest to know the* ***current energy release rate*** *G. Formally the energy rate is defined as*

$$G := -\frac{\partial(U-V)}{\partial A}, \tag{9}$$

where U is the potential energy available in the specimen $B \subset \mathbb{R}^d$ to create new fractures, V is the work associated to external forces and A is the crack length per area. The current energy release rate has the unit $[G] = \frac{J}{m^2}$.

Proposition 5.19. *Using G_C and G, we can formulate three possible situations:*

- *the critical energy release rate is equal to the current energy release rate: $G = G_c$ implies crack propagation*
- *the critical energy release rate is smaller than the current energy release rate: $G < G_c$ implies no crack propagation*
- *the critical energy release rate is larger than the current energy release rate: $G > G_c$ implies an unstable crack growth (no unique opinion in mechanics, what unstable means in this context)*

In the following, we introduce the (crack) **surface energy** E_S, which connects G_C with the energy of the current fracture.

Definition 5.20 (Surface energy)**.** *The surface energy of the crack $C \subset \bar{B}$ is defined as:*

$$E_S(C) = \int_C G_C(s)\ ds = G_C \mathcal{H}^{d-1}(C),$$

where $\mathcal{H}$ is the Hausdorff measure [79]. For smooth surfaces the Hausdorff measure corresponds to the length of a crack C (1D) in a 2D setting and the crack area (2D) in a 3D setting.

Remark 5.21 (Toughness)**.** *If $G_C(x) > 0$, then an increasing $E_S(C)$ means that more fractures arise and the fracture lengths increase. If $G_c(x) \equiv \infty$, we have maximal toughness and the material cannot break.*

Before we define the bulk energy, we introduce the function space for the displacement variable as

$$\mathbb{C}(C, u) := \{u \in H^1(B\backslash C) \mid u = u_D \text{ on } \partial\Omega_D\}.$$

This definition of the deformation of the solid allows to formulate material laws and constitutive relations in the following. The used space $H^1(B\backslash C)$ space can be vector-valued.

Definition 5.22 (Linearized strain tensor)**.** *The linearized strain tensor is defined as*

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

Definition 5.23. *We denote by $W(e(u))$ the elastic (strain) energy density.*

Example 5.24. *We give three examples of energy density functions.*

- *In the case of the Laplacian or Poisson equation, the energy density function is*

  $$W(e(u)) = \frac{\mu}{2}|\nabla u|^2,$$

  where $\mu > 0$ is a material parameter. The energy is defined as

  $$E_B(u) = \int_B [\frac{1}{2}\mu|\nabla u|^2 - fu]dx. \tag{10}$$

  The PDEs which we have to solve, are the Euler-Lagrangian equations:

  $$E_B'(u)(\psi) = \int_B [\frac{1}{2}\mu\nabla u \cdot \nabla\psi - f\psi]dx. \tag{11}$$

- *Another example is the linearized Saint-Venaint Kirchhoff model with the stress tensor*

  $$\sigma(u) = 2\mu e(u) + \lambda\, tr(e(u))\mathcal{I},$$

  where $\lambda, \mu > 0$ are the Lamé parameters.

- *For a general elastic model, we have*

  $$W(e(u)) = Ae(u) : e(u)$$

  is the energy density with the Frobenius scalar product, with $A \in \mathbb{R}^{d\times d\times d\times d}$ and $Ae(u) =: \sigma(u)$ is the stress tensor.

Exercise 4. *Derive from the general expression $Ae(u)$, the stress tensor $\sigma(u) = 2\mu e(u) + \lambda\, tr(e(u))\mathcal{I}$.*

Exercise 5. *Write down $|e(u)|^2$ component-wise.*

Definition 5.25. *In engineering, material properties are often defined in terms of Young's modulus E_Y and the Poisson ratio ν_s. The relationship to the Lamé parameters is as follows:*

$$\begin{aligned} \nu_s &= \frac{\lambda}{2(\lambda+\mu)}, \quad & E_Y &= \frac{\mu(\lambda+2\mu)}{\lambda+\mu}, \\ \mu &= \frac{E_Y}{2(1+\nu_s)}, \quad & \lambda &= \frac{\nu_s E_Y}{(1+\nu_s)(1-2\nu_s)} \end{aligned}$$

In the following, we define a second energy, the **bulk energy**. The interplay of surface and bulk energy will be the key description to formulate fracture growth.

Definition 5.26 (Bulk energy)**.** *The bulk energy is defined as*

$$E_R(C,u) = \int_{B\setminus C=\Omega} W(e(u))\; dx - \int_{B\setminus C=\Omega} f\cdot u\; dx.$$

In fracture settings, often the driving forces are boundary conditions and therefore $f \equiv 0$. For this reason, we simply often work with:

$$E_R(C,u) = \int_{B\setminus C=\Omega} W(e(u))\; dx.$$

Further details on the bulk energy can be found in [89].

In the following, elementary properties of the energy integrals are listed.

Proposition 5.27. *The bulk and surface energies shall satisfy the following elementary properties:*

a) E_S is strictly monotonically increasing in C

b) E_R is monotonically decreasing in C for any fixed $u \in \mathbb{C}(C,u)$

Formulation 5.28 (Total energy in B)**.** *Let C be a given crack and u_D a given loading on $\partial\Omega_D$. Then the* ***total energy*** *$E_T(C,u)$ is defined as*

$$E_T(C,u) = E_R(C,u) + E_S(C).$$

Formally we determine C and u by solving

$$\min_{u\in\mathbb{C}} E_T(C,u). \tag{12}$$

This would mean finding the global minimum, which is too costly and also physically not always achievable. For this reason, local minima are sought. In the following, we think about the relation to crack irreversibility and further the relation of E_T to an existing fracture model (Griffith's model 1920). The fractures, we are interested in, are called brittle (versus ductile), quasi static (versus dynamic or fully-time-dependent) fractures.

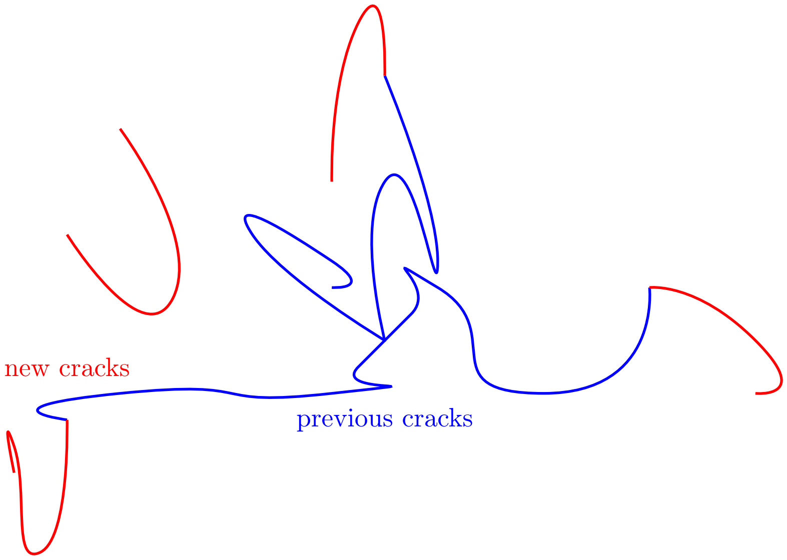

Figure 11: Crack evolution: old (in blue) and new cracks (in red).

To determine the evolution of the crack(s)

$$u(x,t) = u_D(x,t) \text{ on } \partial\Omega_D \times I.$$

The idea is to minimize $E_T(C,u)$ at time t among all cracks C, which contain all previous cracks $C(s), s < t$. A sketch is provided in Figure 11. A global energy minimization is a convenient mathematical postulate, but not always justified by related physics. In practice (like in non-convex optimization) we rather want to get **local** minima. Future geometries and crack locations are limited by previous crack patterns because of the crack irreversibility constraint. Irreversiblity means that the crack cannot heal or existing fractures will not disappear. According to [64], we assume a monotonically increasing loading (MIL) with

$$u = u_D = \begin{cases} t\ u_0(x), & t \geq 0 \\ 0, & t < 0. \end{cases}$$

5.5 Griffith's model from 1920

Along with Griffith, a fracture propagates if the rate of elastic energy (G) decrease per unit surface area of the incremental step is equal to the quasi-static critical energy release rate (G_C). The surface energy is a macroscopic description of the lattice debonding observed on the microscopic level.

In the frame of Griffith, the fracture process is quasi-static. Griffith's law holds true for:

- brittle-fractures, e.g. glass, concrete (in contrast to ductile fracture, e.g. wood, steel),
- but a preexisting crack is assumed,
- and a well-defined crack path is assumed.

To understand Griffith's laws, further definitions are necessary.

Definition 5.29 (Crack)**.** *A crack or fracture is defined as follows:*

$$C(l) = C_0 \cup \{x(s); 0 \leq s \leq l\}.$$

An illustration is provided in Figure 12. As a reminder, the length of a curve can be computed via a line integral $L_0 = \int_0^{x_l} ds$.

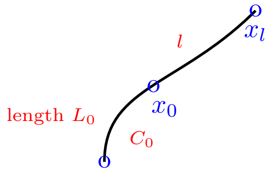

Figure 12: Old C_0 and new part $\{x(s)\}$ of a crack resulting in $C(l)$. The new crack contains the old crack.

Definition 5.30 (Bulk and surface energies)**.** *We define the potential and surface energies, respectively:*

$$\begin{aligned} \textit{Potential energy: } P(t,l) &:= E_R(C(l), u_0) && \rightarrow \textit{monotonically decreasing} \\ \textit{Surface energy: } Q(t,l) &:= E_S(C(l)) - E_S(C_0) && \rightarrow \textit{monotonically increasing} \end{aligned}$$

Here, u_0 is a given initial displacement field. We shall investigate the trajectory of a crack along its path:

$$t \mapsto l(t),$$

where t is the time or loading step. Finally, we define the (current) energy release rate:

$$G := -\frac{\partial P}{\partial l}.$$

The unit is

$$[G] = \frac{J}{m^2}.$$

Proposition 5.31 (Griffith's law)**.** *Let $l(t)$ be absolutely continuous in t, then Griffith's law of crack evolution is satisfied:*

a) $l(0) = 0$ *(initial crack)*

b) $\partial_t l(t) \geq 0$ *(crack can only grow; irreversibility)*

c) $G \leq G_c$ *(energy release rate is bounded by the critical energy release rate)*

d) $(-G + G_c)\partial_t l = 0$ *(compatibility condition; crack can only grow when energy release rate is critical)*

The proof can be found in Proposition 4.8 in [64].

5.6 Francfort and Marigo's model from 1998

In 1998, Francfort and Marigo [64] generalized Griffith's idea to allow predicting crack growth, crack nucleation and crack initiation.

Proposition 5.32 (Laws by Francfort and Marigo [64])**.** *Let $u(x,t)$ on $\partial\Omega_D$ and MIL. Then $C(t)$ should satisfy:*

a) *Irreversibility: $C(t)$ is an increasing function in t*
$C(t) = C_0$ is the initial fracture for $t \leq 0$

b) *Energy minimization:*

$$E_T(C(t), u(t)) \leq E_T(C, u(t)) \; \forall \cup_{s<t} \mathbb{C}(s) \subset C$$

c) *Constraint on the set of the solution:*

$$E_T(C(t), u(t)) \leq E_T(C(s), u(s)) \quad \forall s < t$$

This last law avoids that too many fracture solutions are admissible. More details are given by Francfort and Marigo [64].

Remark 5.33. *In contrast to Griffith's approach, the variational model of Francfort and Marigo does allow crack growth with an unknown crack path and crack nucleation.*
Furthermore, if $t \to \infty$ with an MIL, a mechanical failure (thus total cracking) of the specimen arises. Thermodynamic explanation of the previous laws and the feature of failure for $(t \to \infty)$ are given by Miehe, Welschinger and Hofacker [117].

Let us consider the following question: What has Griffith's model in common with the variational approach of Francfort and Marigo?

Assumption 5.34. *In [64], the following assumptions were made:*

- *2D situation; elasticity*
- *no contact of the lips of the crack as depicted in Figure 13*

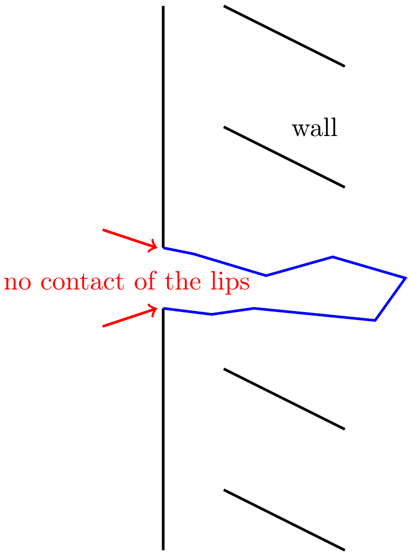

Figure 13: Model assumption of Francfort and Marigo: no contact of the crack lips.

- *Griffith needs a given crack path; crack path sufficiently smooth (rectifiable curve C)*
- *MIL*
- *the crack is parametrized by arc-length $x(s)$ with $x(0) = x_0$*

5.6.1 Specific forms of $E_T(C,u)$

We provide some examples of possible energy functionals:

a) Given the deformation $u : \Omega \to \mathbb{R}$ as a scalar-valued function, we deal with

$$E(C,u) = \frac{1}{2}\int_\Omega |\nabla u|^2 \; dx + \int_C G_C(x) \; ds - \int_\Omega f \cdot u \; dx.$$

b) Given $u : \Omega \to \mathbb{R}^d$ as the vector-valued elasticity, the energy functional is given as

$$E(C,u) = \frac{1}{2}\int_\Omega Ae(u) : e(u) \; dx + \int_C G_C(x) \; ds,$$

where the stress $Ae(u)$ with the elasticity tensor A of fourth order is defined as

$$Ae(u) = \sigma(u) = 2\mu e(u) + \lambda \mathrm{tr}(e(u))\mathcal{I}.$$

In case of the Saint-Venaint-Kirchhoff model $e(u) := \frac{1}{2}(\nabla u + \nabla u^T)$, $\mathcal{I}$ is the identity matrix and $\mu, \lambda > 0$ are the Lamé parameters. This linear stress-strain relationship follows Hook's law.

In principle, we could now compute any fracture problem. However E_R is defined on $\Omega \subset \mathbb{R}^d$ and E_S is defined in $C \subset \mathbb{R}^{d-1}$. This causes numerical problems because it is difficult to develop schemes that can compute quantities/equations simultaneously in $\mathbb{R}^d$ **and** $\mathbb{R}^{d-1}$. For this reason, E_S will be approximated via terms (integrals) on the entire domain Ω.

5.6.2 Properties

We discuss some important questions and aspects in the following:

Q1) How can we mathematically and numerically treat the computation of $\int_C G_C(x) \; ds$?

↪ An option is to set up an extension to $\int_B$ along to Ambrosio-Tortorelli elliptic functionals (1990/1992) [10], which can be motivated by image processing. In contrast, Francfort and Marigo considered W^1 spaces and further extensions.

↪ leads to Modica and Mortola (1977) [128]. They approximated the perimeter functional by elliptic functionals.

A1): Free discontinuity problems ([10], [9])
Unknown pair: (u,Γ): Γ is varying in a class of closed subsets of a fixed open set $\Omega \subset \mathbb{R}^d$
$u : \Omega\backslash\Gamma \to \mathbb{R}^d$ e. g. $u \in C^1(\Omega\backslash\Gamma)$ or $u \in W^{1/p}(\Omega\backslash\Gamma)$
$\min E(u,\Gamma)$, which is the same as in a fracture setting
Apply Ambrosio-Tortorelli to $E_T(u,C)$:

Proposition 5.35 (Regularized phase-field for Laplacian)**.** *The non-regularized energy functional reads:*

$$E_T(u,C) = \frac{1}{2}\int_\Omega |\nabla u|^2 \; dx + G_c H^{d-1}(C).$$

Using Ambrosio-Tortorelli, we replace the sharp lower-dimensional crack C by a smoothed indicator function φ and we obtain the following regularized version:

$$E_{T,\epsilon}(u,\varphi) = \frac{1}{2}\int_B [(1-\kappa)\varphi^2 + \kappa]|\nabla u|^2 \; dx + \frac{1}{2}G_C \int_B (\frac{1}{\epsilon}(1-\varphi)^2 + \epsilon|\nabla\varphi|^2) \; dx$$

with $u \in H^1(B)$, $0 \le \varphi \le 1$, $\varphi \in L^\infty$.

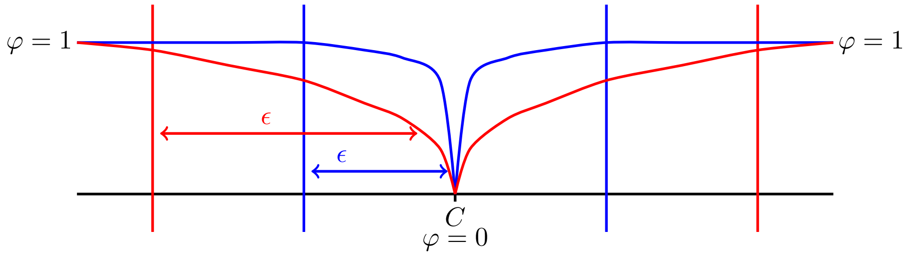

Figure 14: The phase-field variable φ is a smoothed indicator function.

The regularization parameter κ is numerically useful when $\varphi \to 0$ such that the discrete system matrix remains regular:

$$[(1-\kappa)\varphi^2 + \kappa]|\nabla u|^2 \quad \Rightarrow \quad \kappa|\nabla u|^2.$$

Mathematically, $\varphi^2|\nabla u|^2$ works as well (see Braides 1998 [33]). Using Γ-convergence (e.g., again [33]) for $\epsilon \to 0$:

$$E_{T,\epsilon}(u,\varphi) = \begin{cases} E(u,C) & \text{if } \varphi = 1 \text{ a.e. in } B \\ +\infty, & \text{otherwise} \end{cases}$$

Let us finally give some qualitative explanations of the choice of

$$\int_B (\frac{1}{\epsilon}(1-\varphi)^2 + \epsilon|\nabla\varphi|^2)\, dx.$$

The first term is actually the term we are interested in, namely determining values that are 0 (crack) or 1 (unbroken solid). In fact for $\varepsilon \to 0$ we enforce the behavior of sharp changes between 0 and 1 since the first term dominates. For regularity purposes, we add the gradient penalty term that smooths the transition from 0 to 1. The larger ε, the smoother is the transition zone.

Q2) How to solve $\min E_T(C,u)$?

↪ first-order optimality condition: Euler-Lagrange system

$$\min E_T(C,u) \Rightarrow E'(C,u) = 0$$

↪ Only stationary points (also saddle-points)

A2): Given $E_{T,\varepsilon}$, we obtain the Euler-Lagrange system by taking the directional derivatives:

Proposition 5.36. *The Euler-Lagrange equations in variational formulation are given by:*

$$\begin{aligned} E'_{T,\epsilon,u}(u,\varphi)(w) &= \int_B [(1-\kappa)\varphi^2 + \kappa]\nabla u \cdot \nabla w\, dx, \\ E'_{T,\epsilon,\varphi}(u,\varphi)(\psi) &= \int_B (1-\kappa)\varphi|\nabla u|^2\psi\, dx + G_C \int_B (\frac{1}{\epsilon}(1-\varphi)(-\psi) + \epsilon\nabla\varphi\cdot\nabla\psi)\, dx. \end{aligned}$$

In numerical analysis, often a slightly different notation is adopted that we introduce for the convenience of the reader:

Proposition 5.37. *We define the combined solution vector and test functions, respectively:*

$$\begin{aligned} U &= (u,\varphi), \quad \textit{(trial function)} \\ \Psi &= (w,\psi), \quad \textit{(test function)} \end{aligned}$$

Introducing a semi-linear form $A(\cdot)(\cdot)$ in which the first argument may be nonlinear and the second argument is linear, the previous statements are rewritten as follows:

$$\begin{aligned} A_1(U)(\Psi) &= E'_{T,\epsilon,u}(u,\varphi)(w) = \Big([(1-\kappa)\varphi^2+\kappa]\nabla u, \nabla w\Big) \\ A_2(U)(\Psi) &= E'_{T,\epsilon,\varphi}(u,\varphi)(\psi) = \Big((1-\kappa)\varphi|\nabla u|^2, \psi\Big) + (-\frac{G_C}{\epsilon}(1-\varphi),\psi) + (G_C\epsilon\nabla\varphi, \nabla\psi). \end{aligned}$$

These formulations are similar to the u- and φ-equation discussed earlier in (4) and (5).

Q3) How to prescribe/include and analyze the **crack irreversibility** constraint?

A3) Crack irreversibility:

$$\partial_t l(t) \geq 0 \ \Leftrightarrow \partial_t\varphi \leq 0.$$

This condition can be interpreted as an entropy condition in the thermodynamic sense.

In general, the energy form as well as the variational form include constraints in function spaces, where the strong formulation contains the compatibility formulation which is similar to the obstacle problem. See [74] or [95, 96] for further details.

5.6.3 Weak formulation of quasi-static brittle phase-field fracture

Let us introduce the loading interval

$$0 = t_0 < t_1 < t_2 < \ldots < t_N = T,$$

with the end loading value T. The loading step size is given as $k_{n+1} = t_{n+1} - t_n$. If the loading step are uniformly distributed we simply use $k := k_{n+1}$.

To define the function spaces for the deformation u and the phase-field function φ, for a quasi-static formulation first the time derivative term $\partial_t\varphi$ is discretized via

$$\partial_t\varphi \approx \frac{\varphi - \varphi^{n-1}}{k} \leq 0,$$

with k is the time step size or incremental step. Therefore it holds $\varphi \leq \varphi^{n-1}$.

Definition 5.38 (Function spaces)**.**

$$\begin{aligned} V &= \{u \in H^1(B) | u = u_D \ on\ \partial\Omega_D, u = 0 \ on\ \partial\Omega_2\}, \\ W &= \{\varphi \in H^1(B) |\ \varphi \leq \varphi^{n-1} \leq 1 \ in\ B\}. \end{aligned}$$

The weak formulation in the following can be seen as the starting point for all ongoing considerations.

Formulation 5.39 (Weak formulation)**.** *For the loading steps $n = 0, 1, .., N$ find $U = (u, \varphi) \in V \times W$ with $\varphi(0) = \varphi_0$ such that*

$$A(U)(\Psi - (0,\psi)) \leq 0 \quad \forall \Psi = (w,\psi) \in V \times (W \cap L^\infty),$$

where

$$A(U)(\Psi - (0,\psi)) = A_1(U)(\Psi) + A_2(U)(\Psi - (0,\psi)),$$

with

$$A_1(U)(\Psi) = 0 \quad and\ A_2(U)(\Psi - (0,\psi)) \leq 0,$$

and

$$\begin{aligned} A_1(U)(\Psi) \ &= \Big([(1-\kappa)\varphi^2+\kappa]\sigma, \nabla w\Big), \\ A_2(U)(\Psi - (0,\psi)) \ &= \big((1-\kappa)\varphi\sigma : e(u), \psi - \varphi\big) + \left(-\frac{G_C}{\epsilon}(1-\varphi), \psi - \varphi\right) + (G_C\epsilon\nabla\varphi, \nabla(\psi - \varphi)), \end{aligned}$$

where $\sigma = 2\mu e(u) + \lambda tr(e(u))I$.

Formulation 5.40. *The corresponding energy formulation to the previous weak problem reads:*

$$E_{T,\epsilon}(u,\varphi) = \frac{1}{2}\int_B [(1-\kappa)\varphi^2 + \kappa]\sigma : e(u)\ dx + \frac{G_C}{2}\int_B \Big(\frac{1}{\epsilon}(1-\varphi)^2 + \epsilon|\nabla\varphi|^2\Big)\ dx. \qquad (13)$$

Exercise 6. *Differentiate (formally)* (13) *in order to obtain the weak Formulation 5.39. Hint: For differentiation in Banach spaces, please see Section 8.7.*

Exercise 7. *Derive the strong formulation of Formulation 5.39. Hint: Please see the simplified Formulation 5.1 to get some ideas how a strong form may look like.*

Exercise 8. *Derive the weak formulation for non-homogeneous traction forces on* $\partial\Omega_0$ *and* $\partial\Omega_1$*.*

5.7 Thermodynamic extensions and interpretations

In this section, we briefly discuss two extensions based on thermodynamic arguments:

- Distinguishing fracture development due to tension and compression [11] and [121]
- Replacing the inequality constraint by observing the maximal strain energy [117]

5.7.1 Energy splitting for fracture under tension and compression

Along with [11] and [117], there are important extensions of the brittle-fracture model in the weak Formulation 5.39. For detailed comparisons of both models we refer to [7] and other comments can be found in [24][p. 79]. We decompose the stress tensor (or directly the energy) additively into

- its tensile part σ^+ and
- its compressive part σ^-.

From the physical/engineering point of view, the energy degradation yielding new fractures or fracture propagation is only caused by tensile stresses. This motivates the following modification of the solid stress tensor:

$$\sigma := \big((1-k)\varphi^2 + k\big)\sigma^+ + \sigma^-,$$

in which the phase-field variable only acts on σ^+, but not on σ^-.

An illuminating numerical test is the single edge notched shear test, see Chapter 9 and also Figure 15, in which the fracture only has a path as in the experiment when splitting is used, but not when the standard non-split stress tensor is employed.

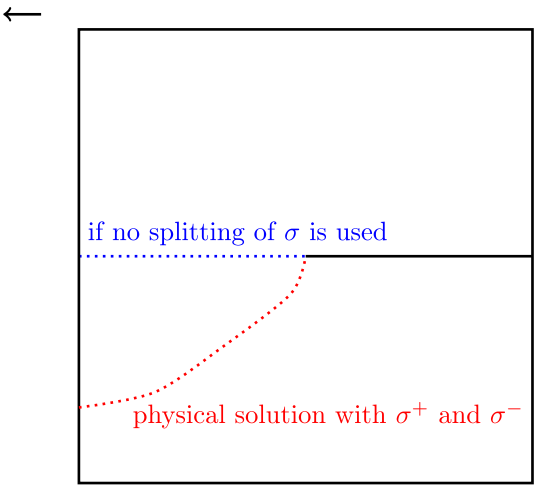
⟵
if no splitting of σ is used
physical solution with σ^+ and σ^-

Figure 15: Single edge notched shear test: sketch what happens when stress is split or not.

5.7.1.1 Strain energy splitting à la Miehe et al. First, the energy can be split into tensile σ^+ and compressive parts σ^-, where damage just happens due to tensile stresses. The term

$$\frac{1}{2}\int_B [(1-\kappa)\varphi^2 + \kappa]Ae(u) : e(u),$$

where $\sigma(u) := Ae(u)$ is the stress tensor again. Stress splitting yields:

$$\frac{1}{2}\int_B [(1-\kappa)\varphi^2 + \kappa]\sigma^+ : e(u) + \frac{1}{2}\int_B \sigma^- : e(u).$$

The tensile and compressive stresses itself are defined as

$$\begin{aligned}\sigma^+ &:= 2\mu e^+ + \lambda < \mathrm{tr}(e) > \mathcal{I},\\ \sigma^- &:= 2\mu(e - e^+) + \lambda(\mathrm{tr}(e) - < \mathrm{tr}(e) >)\mathcal{I},\end{aligned}$$

with

$$\begin{aligned} e &:= P\Lambda P^T,\\ e^+ &:= P\Lambda^+ P^T,\\ e^- &:= P\Lambda^- P^T,\\ \Lambda &= \begin{pmatrix}\lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3\end{pmatrix},\\ \Lambda^+ &= \begin{pmatrix}<\lambda_1> & & \\ & <\lambda_2> & \\ & & <\lambda_3>\end{pmatrix},\\ \Lambda^- &= \Lambda - \Lambda^+,\\ < x > &:= \begin{cases} x, & \text{if } x > 0,\\ 0, & \text{if } x \le 0.\end{cases}\end{aligned}$$

Therein, P is the matrix that contains in its columns the eigenvectors corresponding to the eigenvalues $\lambda_i, i = 1, 2, 3$.

5.7.1.2 Strain energy splitting à la Amor et al. The Amor et al. model is somewhat simpler. Here, the two stress contributions are given by:

$$\begin{aligned}\sigma^+ &:= \kappa tr^+(e)I + 2\mu e_D,\\ \sigma^- &:= \kappa tr^-(e)I,\end{aligned}$$

with $\kappa = \frac{2}{d}\mu + \lambda$ and the deviatoric part of the strain tensor $e = \frac{1}{2}(\nabla u + \nabla u^T)$ is defined as

$$e_D := e - \frac{1}{d}tr(e)I, \quad d = 2, 3.$$

Moreover,

$$tr^+(e) = \max(tr(e), 0), \quad tr^-(e) = tr(e) - tr^+(e).$$

5.7.2 A weak phase-field fracture formulation using stress splitting

We summarize the previous developments into our second main formulation describing phase-field fracture propagation in brittle materials:

Formulation 5.41 (Weak formulation using stress splitting). *For the loading steps $n = 0, 1, .., N$ find $U = (u, \varphi) \in V \times W$ with $\varphi(0) = \varphi_0$ such that*

$$A(U)(\Psi - (0,\psi)) \le 0 \quad \forall \Psi = (w,\psi) \in V \times (W \cap L^\infty),$$

where

$$A(U)(\Psi - (0,\psi)) = A_1(U)(\Psi) + A_2(U)(\Psi - (0,\psi))$$

with

$$A_1(U)(\Psi) = 0 \quad \textit{and } A_2(U)(\Psi - (0,\psi)) \le 0,$$

and

$$\begin{aligned} A_1(U)(\Psi) = &= \Big([(1-\kappa)\varphi^2 + \kappa]\sigma^+, \nabla w\Big) + \big(\sigma^-, \nabla w\big) \\ A_2(U)(\Psi - (0,\psi)) = &= \big((1-\kappa)\varphi\sigma^+ : e(u), \psi - \varphi\big) + \Big(-\frac{G_C}{\epsilon}(1-\varphi), \psi - \varphi\Big) + (G_C\epsilon\nabla\varphi, \nabla(\psi - \varphi)). \end{aligned}$$

5.7.3 Derivation of a strain history field/replacing the irreversibility constraint

Thermodynamic interpretations discussed by Miehe et al. 2010 [117] are concerned with the fracture energy resulting in a derivation of a strain history field. The surface integral $E_S(\varphi)$ can be interpreted from a thermodynamic viewpoint.

Proposition 5.42 (Thermodynamical properties of $E_S(\varphi)$). *Let us define as before*

$$E_S(\varphi) = \frac{1}{2}\int_B G_C(\frac{1}{\epsilon}(1-\varphi)^2 + \epsilon|\nabla\varphi|^2)\, dx.$$

We postulate:

↪ *Cracking is a dissipative procedure.*

↪ *The system looses the ability to perform mechanical work (e.g. like friction).*

↪ *Dissipation is expressed through crack irreversibility, which is the work to create a diffusive crack topology.*

Remark 5.43. *The crack irreversibility constraint can be eliminated, which yields an alternative formulation resulting in a variational equation directly.*

Proposition 5.44 (Crack dissipation). *We have*

$$\begin{aligned} \frac{d}{dt}E_S(\varphi) &= \int_B G_C(-\frac{1}{\epsilon}(1-\varphi)\partial_t\varphi + \epsilon\nabla\varphi\cdot\nabla\partial_t\varphi)\, dx \\ &= \int_B G_C(-\frac{1}{\epsilon}(1-\varphi) - \epsilon\Delta\varphi)\partial_t\varphi\, dx. \end{aligned}$$

Proof. The proposition can be proven via differentiation (chain rule) plus partial differential integration and the use of a homogeneous Neumann condition for φ. □

Thermodynamics assume a positive dissipation ⇒ $\frac{d}{dt}E_S(\varphi) \ge 0$ (which is an entropy condition). But this means:

$$\int_B G_C\underbrace{(-\frac{1}{\epsilon}(1-\varphi) - \epsilon\Delta\varphi)}_{\le 0}\underbrace{\partial_t\varphi}_{\le 0}\, dx \ge 0.$$

The first term, the negative variational derivative is smaller or equal to zero and can be rewritten as $\frac{1}{\epsilon}(1-\varphi) + \epsilon\Delta\varphi \ge 0$. The time derivative of φ is smaller or equal to zero because of the natural assumption of irreversible cracking.

Let us introduce a so-called *driving force field* β according to Miehe et al.(2010) [117]. So we can formulate the inequality constraint of the regularized crack energy as:

$$\frac{1}{\epsilon}(1-\varphi)+\epsilon\Delta\varphi \geq \beta \geq 0.$$

Remark 5.45. *Compare this to the very beginning in which we already motivated a forcing function f in the phase-field equation* (2)*.*

We will see that the variable β corresponds to the φ-derivative in the u-equation:

$$\begin{array}{rll} \text{Laplacian:} & \beta := (1-\kappa)\varphi\ |\nabla u|^2 & \\ \text{Elasticity:} & \beta := (1-\kappa)\varphi\ \sigma : e(u) & \text{(without stress splitting)} \\ \text{Elasticity:} & \beta := (1-\kappa)\varphi\ \sigma^+ : e(u) & \text{(with stress splitting).} \end{array}$$

Thus

$$G_C(\frac{1}{\epsilon}(1-\varphi)+\epsilon\Delta\varphi) \geq (1-\kappa)\varphi\sigma^+ : e(u). \tag{14}$$

On the other hand, we recall the compatibility condition:

$$\Big[(1-\kappa)\varphi\sigma^+ : e(u) + G_C(-\frac{1}{\epsilon}(1-\varphi)-\epsilon\Delta\varphi)\Big]\cdot\partial_t\varphi = 0. \tag{15}$$

For a growing crack, we combine (14) and (15). In (15), we can have a variation in the crack length when the first expression is zero:

$$(1-\kappa)\varphi\sigma^+ : e(u) + G_C(-\frac{1}{\epsilon}(1-\varphi)-\epsilon\Delta\varphi) = 0.$$

In this case $\partial_t\varphi < 0$ (strictly). But this also means that in (14), we have

$$G_C(\frac{1}{\epsilon}(1-\varphi)+\epsilon\Delta\varphi) = \beta.$$

Moreover, we obtain the following result:

Proposition 5.46. *Let $\psi_e^+ := \sigma^+ : e(u)$, which is the energy storage function. If $\psi_e^+ \to \infty$, then $\varphi \to 0$.*

Proof. Let $\varphi > 0$:

$$\begin{aligned} & \frac{G_C(\frac{1}{\epsilon}(1-\varphi)+\epsilon\Delta\varphi)}{(1-\kappa)\varphi} \geq \psi_e^+ \\ \overset{\max}{\Longrightarrow}\quad & \frac{G_C(\frac{1}{\epsilon}(1-\varphi)+\epsilon\Delta\varphi)}{(1-\kappa)\varphi} = \max_u \psi_e^+ =: \mathcal{H} \end{aligned} \tag{16}$$

This means,

$$\psi_e^+ \to \infty \Rightarrow \text{ LHS } \to 0, \quad \text{thus, } \varphi \to 0.$$

□

Corollary 5.47. *The previous proposition shows that we can enforce crack growth when the maximal strain energy ψ_e^+ is taken because in this case in equation (14) the equality holds true, due to the compatibility condition (15). The equation (16) then yields $\partial_t\varphi > 0$ strictly for all time t. This allows to eliminate the crack irreversibility condition on the continuous level. The function $H := \max_{s\in[0,t]}\psi_e^+(u(s))$ (recall $u = u(s) = u(s,x)$, where s is the temporal variable and x the spatial variable) is a so-called strain history function [117].*

Proof. It holds

$$G_C(\frac{1}{\epsilon}(1-\varphi) + \epsilon\Delta\varphi) \geq (1-\kappa)\varphi\sigma^+ : e(u)$$

Taking the maximum over space and time yields:

$$G_C(\frac{1}{\epsilon}(1-\varphi) + \epsilon\Delta\varphi) = (1-\kappa)\varphi \max_{s\in[0,t]} \psi_e^+(s). \tag{17}$$

Since Equation (17) is an equality at all times $s \in [0,t]$, the compatibility condition yields $\partial_t\varphi < 0$. □

5.7.4 A weak phase-field fracture formulation using strain history field and stress splitting

Our third main formulation for describing phase-field fracture is finally:

Formulation 5.48 (Alternative Formulation inspired by Miehe et al. 2010 [117])**.** *For the loading steps $n = 0, 1, .., N$, find $U = (u,\varphi) \in V \times H^1$ with $\varphi(0) = \varphi_0$, such that*

$$A(U)(\Psi) = 0 \quad \forall \Psi = (w,\psi) \in V \times (H^1 \cap L^\infty),$$

where

$$A(U)(\Psi) = A_1(U)(\Psi) + A_2(U)(\Psi),$$

with

$$A_1(U)(\Psi) = 0 \quad \textit{and } A_2(U)(\Psi) = 0,$$

and

$$\begin{aligned} A_1(U)(\Psi) = & \Big([(1-\kappa)\varphi^2 + \kappa]\sigma^+, \nabla w\Big) + (\sigma^-, \nabla w), \\ A_2(U)(\Psi) = & \big((1-\kappa)\varphi H, \psi\big) + \left(-\frac{G_C}{\epsilon}(1-\varphi), \xi\right) + (G_C\epsilon\nabla\varphi, \nabla\psi) \quad \forall \Psi \in V \times H^1(B) \cap L^\infty. \end{aligned}$$

Remark 5.49. *The alternative formulation offers a system of equations, which simplifies the system significantly. The strain-history function $\mathcal{H}$ can be tricky to implement. This is why the strain-history need to be stored in each FEM quadrature point. The regularity of $\mathcal{H}$ in space and time is an open question.*

5.8 Comments on energy formulations versus Euler-Lagrange PDEs

Modeling phase-field fracture in elasticity as done in these lecture notes allows us to start from an energy formulation and seeking global (or better: local) minima, e.g., [28]. By differentiating of the energy functional, we obtain a PDE in variational form, the so-called Euler-Lagrange equations. Solving this PDE gives us not only minima, but all stationary points. Specifically, it can happen that our solution is a saddle-point. Therefore, energy formulations yield a smaller, but stronger, class of solutions and are advantageous from the physical point of view.

On the other hand, PDE formulations allow incorporating a much broader class of models. For instance, the Stokes equations (being symmetric) can be formulated with the help of an energy functional. But the extension to the incompressible Navier-Stokes equations does not have a corresponding energy form. Also, simple equations of parabolic type (such as the heat equation) do not allow for an energy concept. The same holds for the Biot equations describing flow and deformation of porous media. For further details on formulating phase-field fracture models within such frameworks, we refer the reader to [124][Sect. 2.4] or [172] or [163][Sect. 3] or [126][Sect. 2.3].

For some models, one can still get an energy formulation. For instance, in quasi-static fluid-filled fracture in porous media, some authors interpret the given system as an incremental problem and analyze their formulations on a given time step. On the time-discretized level, we can (possibly!) again formulate elliptic, symmetric PDEs that can be derived from a corresponding energy formulation, e.g., [118, 119]. Of course, this is not possible for all extensions, for instance, phase-field fracture propagation coupled with fluid-structure interaction (using the Navier-Stokes equations).

6 Classifications of PDEs

In this chapter, we briefly recapitulate some classification of PDEs that are relevant for variational phase-field fracture problems. The following characteristic features can be distinguished:

- Single equations and PDE systems
- Nonlinear problems:
  - Nonlinearity in the PDE
  - The function set is not a vector space yielding a variational inequality
- Coupled PDE systems

Since variational inequalities themselves are a very rich research field, we devote extra attention.
To this end, we work with the following four sections:

- PDE systems: Section 6.2
- Nonlinear problems: Section 6.3
- Coupled problems: Section 6.4
- Variational inequalities: Section 6.5

Of course, all three types can be mixed together resulting in a nonlinear, coupled, variational inequality system. And this is exactly the situation that we face in variational phase-field fracture problems.

6.1 Differential equations

Let us first clarify the definition of a differential equation.

Definition 6.1 (Differential equation). *A differential equation is a mathematical equation that relates the sought (unknown) function with its derivatives.*

Differential equations can be split into two classes:

Definition 6.2 (Ordinary differential equation (ODE)). *An ordinary differential equation (ODE) is an equation (or equation system) involving an unknown function of one independent variable and certain of its derivatives.*

Definition 6.3 (Partial differential equation (PDE)). *A partial differential equation (PDE) is an equation (or equation system) involving an unknown function of two or more variables and certain of its partial derivatives.*

For a general description of ODEs and PDEs the multi-index notation is commonly used.

- A multi-index is a vector $\alpha = (\alpha_1, \dots, \alpha_n)$, where each component $\alpha_i \in \mathbb{N}_0$. The order is

$$|\alpha| = \alpha_1 + \ldots + \alpha_n.$$

- For a given multi-index we define the partial derivative:

$$D^\alpha u(x) := \partial_{x_1}^{\alpha_1} \cdots \partial_{x_n}^{\alpha_n} u$$

- If $k \in \mathbb{N}_0$, we define the set of all partial derivatives of order k:

$$D^k u(x) := \{D^\alpha u(x) : |\alpha| = k\}.$$

Example 6.4. *Assume the problem dimension $n = 3$. Then, $\alpha = (\alpha_1, \alpha_2, \alpha_3)$. For instance, let $\alpha = (2, 0, 1)$. Then $|\alpha| = 3$ and $D^\alpha u(x) = \partial_x^2 \partial_z^1 u(x)$.*

Definition 6.5 (Evans [59]). *Let $\Omega \subset \mathbb{R}^n$ be open. Here, n denotes the total dimension including time. Furthermore, let $k \geq 1$ be an integer that denotes the order of the differential equation. Then, a differential equation can be expressed as: Find $u : \Omega \to \mathbb{R}$, such that*

$$F(D^k u, D^{k-1}u, \dots, D^2u, Du, u, x) = 0 \quad x \in \Omega,$$

where

$$F : \mathbb{R}^{n^k} \times \mathbb{R}^{n^{k-1}} \times \mathbb{R}^{n^2} \times \mathbb{R}^n \times \mathbb{R} \times \Omega \to \mathbb{R}.$$

Example 6.6. *We provide some examples. Let us assume the spatial dimension to be 2 and the temporal dimension is 1. That is for a time-dependent ODE (ordinary differential equation) $n = 1$ and for time-dependent PDE cases, $n = 2 + 1 = 3$, and for stationary PDE examples $n = 2$.*

1. *ODE model problem: $F(Du, u) := u' - au = 0$ where $F : \mathbb{R}^1 \times \mathbb{R} \to \mathbb{R}$. Here, $k = 1$.*
2. *Laplace operator: $F(D^2u) := -\Delta u = 0$ where $F : \mathbb{R}^4 \to \mathbb{R}$. That is $k = 2$ and lower derivatives of order 1 and 0 do not exist. We notice that in the general form it holds*

$$D^2u = \begin{pmatrix} \partial_{xx}u & \partial_{yx}u \\ \partial_{xy}u & \partial_{yy}u \end{pmatrix},$$

and for the Laplacian:

$$D^2u = \begin{pmatrix} -\partial_{xx}u & 0 \\ 0 & -\partial_{yy}u \end{pmatrix}.$$

3. *Heat equation: $F(D^2u, Du) = \partial_t u - \Delta u = 0$ where $F : \mathbb{R}^9 \times \mathbb{R}^3 \to \mathbb{R}$. Here, the order is $k = 2$ is the same as before, but a lower-order derivative of order 1 in form of the time derivative does exist. We provide again details on D^2u:*

$$D^2u = \begin{pmatrix} \partial_{tt}u & \partial_{xt}u & \partial_{yt}u \\ \partial_{tx}u & \partial_{xx}u & \partial_{yx}u \\ \partial_{ty}u & \partial_{xy}u & \partial_{yy}u \end{pmatrix},$$

and for the heat equation:

$$D^2u = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\partial_{xx}u & 0 \\ 0 & 0 & -\partial_{yy}u \end{pmatrix},$$

4. *Wave equation: $F(D^2u) = \partial_t^2 u - \Delta u = 0$ where $F : \mathbb{R}^9 \to \mathbb{R}$. Here it holds specifically*

$$D^2u = \begin{pmatrix} \partial_{tt}u & 0 & 0 \\ 0 & -\partial_{xx}u & 0 \\ 0 & 0 & -\partial_{yy}u \end{pmatrix},$$

6.2 Single equations versus PDE systems

Definition 6.7 (Single equation). *A single PDE consists of determining one solution variable, e.g.,*

$$u : \Omega \subset \mathbb{R}^d \to \mathbb{R}.$$

Typical examples are Poisson's problem, the heat equation, wave equation, Monge-Ampère equation, Hamiliton-Jacobi equation, p-Laplacian.

Definition 6.8 (PDE system). *A PDE system determines a solution vector*

$$u = (u_1, \dots, u_d) : \Omega^d \to \mathbb{R}^d.$$

For each $u_i, i = 1, \dots, d$, a PDE must be solved. Inside these PDEs, the solution variables may depend on each other or not. Typical examples of PDE systems are linearized elasticity, nonlinear elasto-dynamics, Maxwell's equations.

6.3 Linear versus nonlinear

In this section, classifications of **linear** and **nonlinear** differential equations are provided. Simply speaking: each differential equation, which is not linear, is called nonlinear. However, in the nonlinear case, a further refined classification can be undertaken. The concept of nonlinearity applies to both single equations and PDE systems.

6.3.1 Definition of linearity

Let $\{U, \|\cdot\|_U\}$ and $\{V, \|\cdot\|_V\}$ be normed spaces over $\mathbb{R}$.

Definition 6.9 (Linear mappings)**.** *A mapping $T : U \to V$ is called **linear** or **linear operator** when*

$$T(u) = T(au_1 + bu_2) = aT(u_1) + bT(u_2),$$

for $u = au_1 + bu_2$ and for $a, b \in \mathbb{R}$ and $u_1, u_2 \in U$.

Example 6.10. *We discuss two examples:*

1. *Let $T(u) = \Delta u$. Then:*

$$T(au_1 + bu_2) = \Delta(au_1 + bu_2) = a\Delta u_1 + b\Delta u_2 = aT(u_1) + bT(u_2).$$

Thus T is linear.

2. *Let $T(u) = (u \cdot \nabla)u$. Then:*

$$T(au_1 + bu_2) = ((au_1 + bu_2) \cdot \nabla)(au_1 + bu_2) \neq a(u_1 \cdot \nabla)u_1 + b(u_2 \cdot \nabla)u_2 = aT(u_1) + bT(u_2).$$

Here, T is nonlinear.

3. *Let $T(u) = |u|$. Then:*

$$T(au_1 + bu_2) = |au_1 + bu_2| \leq |au_1| + |bu_2| \leq |a||u_1| + |b||u_2|.$$

This is obviously not the same as

$$aT(u_1) + bT(u_2) = a|u_1| + b|u_2|.$$

6.3.2 Classes of nonlinearity

We cite from [59]:

Definition 6.11 (Evans[59])**.** *Differential equations are divided into linear and nonlinear classes as follows:*

1. *A differential equation is called **linear** if it is of the form:*

$$\sum_{|\alpha| \leq k} a_\alpha(x) D^\alpha u - f(x) = 0.$$

2. *A differential equation is called **semi-linear** if it is of the form:*

$$\sum_{|\alpha| = k} a_\alpha(x) D^\alpha u + a_0(D^{k-1}u, \ldots, D^2u, Du, u, x) = 0.$$

Here, nonlinearities may appear in all terms of order $|\alpha| < k$, but the highest order $|\alpha| = k$ is fully linear.

3. *A differential equation is called **quasi-linear** if it is of the form:*

$$\sum_{|\alpha| = k} a_\alpha(D^{k-1}u, \ldots, D^2u, Du, u, x) D^\alpha u + a_0(D^{k-1}u, \ldots, D^2u, Du, u, x) = 0.$$

Here, full nonlinearities may appear in all terms of order $|\alpha| < k$, in the highest order $|\alpha| = k$, nonlinear terms appear up to order $|\alpha| < k$.

4. *If none of the previous cases applies, a differential equation is called (fully)* ***nonlinear****. Moreover, if the underlying set is not a vector space (but for instance, a closed, convex subset of a vector space), we deal with a variational inequality that is clearly also* ***not linear****, thus* ***nonlinear****.*

Remark 6.12. *We notice that nonlinear PDEs are in general non-smooth and less regular than linear PDEs. A graphical illustration is given in terms of the obstacle problem that has kinks in the solution on the inner free boundary; see Figure 10. This holds even true when all problem data are very smooth. This has the consequence that for the numerical (spatial) treatment, low-order discretizations (e.g., low-order FEM) are in many cases sufficient.*

6.3.3 Examples

How can we now proof in practice, whether a PDE is linear or nonlinear? The simplest way is to go term by term and check the linearity condition from Definition 6.9.

Example 6.13. *We provide again some examples:*

1. *All differential equations from Example 6.6 are* ***linear****. Check term by term Definition 6.9!*

2. *Euler equations (fluid dynamics, special case of Navier-Stokes with zero viscosity). Here, $n = 2+1+1 = 4$ (in two spatial dimensions) in which we have 2 velocity components, 1 pressure variable, and 1 temporal variable. Let us consider the momentum part of the Euler equations:*

$$\partial_t v_f + v_f \cdot \nabla v_f + \nabla p_f = f.$$

Here the highest order is $k = 1$ (in the temporal variable as well as the spatial variable). But in front of the spatial derivative, we multiply with the zero-order term v_f. Consequently, the Euler equations are ***quasi-linear*** *because a lower-order term of the solution variable is multiplied with the highest derivative.*

3. *Navier-Stokes momentum equation:*

$$\partial_t v_f - \rho_f \nu_f \Delta v_f + v_f \cdot \nabla v_f + \nabla p_f = f.$$

Here $k = 2$. But the coefficients in front of the highest order term, the Laplacian, do not depend on v_f. Consequently, the Navier-Stokes equations are neither fully nonlinear nor quasi-linear. The first order convection term $v_f \cdot \nabla v_f$ is nonlinear. Thus the Navier-Stokes equations are ***semi-linear****.*

4. *A fully* ***nonlinear*** *situation would be:*

$$\partial_t v_f - \rho_f \nu_f (\Delta v_f)^2 + v_f \cdot \nabla v_f + \nabla p_f = f.$$

Example 6.14 (Development of numerical methods for nonlinear equations)**.** *In case you are given a nonlinear IBVP (initial-boundary value problem) and want to start developing numerical methods for this specific PDE, it is often much easier to start with appropriate simplifications in order to build and analyze step-by-step your final method. Let us assume that we want to solve the following nonlinear time-dependent PDE*

$$\nabla u \partial_t^2 u + u \cdot \nabla u - (\Delta u)^2 = f.$$

Then, you could tackle the problem as follows:

1. *Consider the linear equation:*

$$\partial_t^2 u - \Delta u = f,$$

which is nothing else than the wave equation.

2. *Add a slight nonlinearity to make the problem semi-linear:*

$$\partial_t^2 u + u \cdot \nabla u - \Delta u = f$$

3. *Add ∇u such that the problem becomes quasi-linear:*

$$\nabla u \partial_t^2 u + u \cdot \nabla u - \Delta u = f.$$

4. *Make the problem fully nonlinear by considering $(\Delta u)^2$:*

$$\nabla u \partial_t^2 u + u \cdot \nabla u - (\Delta u)^2 = f.$$

In each step, make sure that the corresponding numerical solution makes sense and that your developments so far are correct. Then proceed to the next step.

Remark 6.15. *The contrary to the previous example works as well and arises very often in practice. And should be kept in mind! Often you are given a very complicated PDE (or PDE system). If you have undertaken the implementation and you recognize a difficulty, you can (should!) reduce the PDE term by term and make it step by step simpler up to a linear version. The same procedure holds true for mathematical analysis. If a proof does not work, we try to neglect terms or to linearize them.*

Exercise 9. *Classify the following PDEs into linear and nonlinear PDEs (including a short justification):*

$$\begin{aligned} -\nabla \cdot (|\nabla u|^{p-2} \nabla u) &= f, \\ det(\nabla^2 u) &= f, \\ \partial_t^2 u + 2a\partial_t u - \partial_{xx} u &= 0, \quad a > 0, \\ \partial_t u + u \partial_x u &= 0, \\ \partial_{tt} - \Delta u + m^2 u &= 0, \quad m > 0. \end{aligned}$$

How can we linearize nonlinear PDEs? Which terms need to be simplified such that we obtain a linear PDE?

Exercise 10. *Characterize and classify with the previous techniques the strong formulation of a phase-field fracture problem derived in Exercise 7.*

6.4 Decoupled versus coupled

We shall specify how coupled PDE systems can be characterized.

Definition 6.16 (Coupled PDEs)**.** *A coupled PDE (system) consists of at least two PDEs (or PDE systems) that interact with each other. Namely, the solution of one PDE will influence the solution of the other PDE. In such coupled problems, the underlying PDEs are based on different physical laws and conservation properties determining solution variables with different physical meanings. Furthermore:*

1. *A coupled PDE system can be linear when the solution variables enter into the PDEs in a linear way (e.g., the Biot system for describing flow in porous media);*

2. *In most cases, however, a coupled PDE system is nonlinear (e.g., phase-field fracture, fluid-structure interaction).*

Remark 6.17 (Incompressible Navier-Stokes)**.** *The Navier-Stokes equations also constitute a* ***nonlinear*** *PDE system, but* ***not a coupled*** *PDE system. The nonlinearity is not caused by the coupling of velocities and pressure, but arises from the nonlinear coupling in the velocities only. The Navier-Stokes equations are not a coupled PDE system because velocities and pressure describe common physics, namely flow.*

Exercise 11. *Classify the following:*
Find $p : \Omega \to \mathbb{R}$ and $u : \Omega \to \mathbb{R}^d$ such that for $t \geq 0$:

$$\begin{aligned} \partial_t p + \alpha \nabla \cdot \partial_t u - \nabla \cdot \frac{K}{\nu}(\nabla p - g) &= f \quad \text{in } \Omega \times I, \\ -\nabla \cdot (\sigma - \alpha p) &= 0 \quad \text{in } \Omega \times I, \end{aligned} \tag{18}$$

with $\alpha \in [0,1], K > 0, \nu > 0, g$ is a given function not depending on p, and σ is defined as before.

Answers: linear, single PDE for p, PDE system for u, so in total it is a PDE system; equality, coupling depends on α! If $\alpha = 0$, no coupling, parabolic system. If $\alpha > 0$, we have a coupled system because u influences the system of p and viceversa. The PDE is coupled via volume. The system is non-stationary (time-dependent) with a time-dependent, parabolic PDE for p and an stationary, elliptic PDE for u.

Remark 6.18. *The equations in the Exercise above is the so-called Biot system for modeling flow through porous media. Further, it is one of the rare examples in which coupling does not introduce nonlinearities.*

Exercise 12. *Find $u : \Omega \to \mathbb{R}^d$, $\varphi : \Omega \to \mathbb{R}$ for $t \geq 0$ such that:*

$$\begin{aligned} -\nabla \cdot (\varphi^2 \sigma) &= 0f \quad \text{in } \Omega \times I, \\ \varphi |\nabla u|^2 - \epsilon \Delta \varphi - \frac{1}{\epsilon}(1-\varphi) &\leq 0, \\ \partial_t &\leq 0, \end{aligned}$$

where $I = [0,T]$. For both examples, simplify the systems, are they instationary or stationary? Nonlinear? Which order?

Answers: If we solve for u and φ simultaneously (monolithic approach), then the system is nonlinear! $\varphi^2\sigma(u) \sim \varphi^2 \nabla u \;\Rightarrow$ nonlinear;
if we decouple, then the u-equation is linear. Decouple means, we split the equations and solve them separately (partitioned/staggered approach). However the φ-equations are still nonlinear because of the inequality constraint. The system is always coupled, independent of solving it monolithically or in a partitioned procedure. It is a PDE system (system for u, single PDE for p); the system is quasi-stationary (time-dependence only in the inequality constraint); Variational inequality for u, variational inequality for p, so in total it is a variational inequality PDE system.
Volume coupling of the entire domain, which means that in contrast to interface coupling where the PDEs are coupled on a boundary inside the domain (e.g. FSI); Remark on the nonlinearity: in the monolithic case, we have a quasi-linear system.

Remark 6.19. *As we see later, coupled problems may also decouple by changing certain coefficients (e.g., Biot equations in porous media) or designing solution algorithms based on partitioned procedures. Due to decoupling, the single semi-linear forms may even become linear (e.g., Biot equations), but not always (e.g., fluid-structure interaction in which the sub-problems NSE and elasticity are still nonlinear themselves).*

In a formal way, we write for two variational forms:

$$\begin{aligned} &\text{Find } u_1 \in V_1: \quad A_1(\{u_1,u_2\})(\varphi_1) = F_1(\varphi_1) \quad \forall \varphi_1 \in V_1, \\ &\text{Find } u_2 \in V_2: \quad A_2(\{u_1,u_2\})(\varphi_1) = F_2(\varphi_2) \quad \forall \varphi_2 \in V_2, \end{aligned}$$

where A_1 and A_2 are semi-linear forms representing the partial differential operators. The coupled problem is the sum of both operators $A = A_1 + A_2$ and yields:

$$\text{Find } U \in V: \quad A(U)(\Psi) = F(\Psi) \quad \forall \Psi \in V,$$

where $U = (u_1, u_2)$ and $\Psi = (\varphi_1, \varphi_2)$.

Remark 6.20. *This concept can be easily extended to n PDEs rather than only* 2.

Remark 6.21 (Semi-linear)**.** *The wording semi-linear should not confuse. A semi-linear form is a weak form of a nonlinear PDE in which the test function is still linear. Nonlinear equations can be distinguished into semi-linear, quasi-linear and fully nonlinear. Therefore, the notation of a semi-linear form does not indicate with which level of nonlinearity we are dealing with.*

As seen previously in the examples, the coupling can be of various types:

- **Volume coupling** (the PDEs live in the same domain) and exchange information via volume terms, right hand sides, coefficients.
  Example: Biot equations for modeling porous media flow, phase-field fracture, Cahn-Hilliard phase-field models for incompressible flows.

- **Interface coupling** (the PDEs live in different domains) and the information of the PDEs is only exchanged on the interface.
  Example: fluid-structure interaction.

The coupling is first of all based on physical principles. Based on this information, appropriate numerical schemes can be derived. Specifically, interface-based coupling **is not simple!** Imagine a regular finite element mesh and assume that the interface cuts through the mesh elements. For a more detailed description of interface treatments we refer the reader back to Section 3.2.

Exercise 13. *Describe and justify whether phase-field fracture is an interface or volume-coupled technique?*

Exercise 14. *Decouple Formulation 5.39 and state whether the resulting scheme is still nonlinear or changes its behavior.*

6.5 Variational equations versus variational inequalities

Variational inequalities generalize weak forms such that constraints on the solution variables can be incorporated.

Let $(V, \|\cdot\|)$ be a real Hilbert space and K a nonempty, closed, and convex subset of V. We define the mapping $A : K \to V^*$, where V^* is the dual space of V. Then, we define the abstract problem:

Formulation 6.22 (Abstract variational inequality)**.** *Find $u \in K$ such that*

$$A(u)(\varphi - u) \geq 0 \quad \forall \varphi \in K.$$

If K is a linear subspace (and not only a set!) of V, then we have the linearity properties, see Section 6.3.1, which allows us to define for $u \in K$:

$$\varphi := u \pm w \in K,$$

for each $w \in K$. Since the semi-linear form $A(\cdot)(\cdot)$ is linear in the second argument, we obtain:

$$A(u)(w) \geq 0 \quad \text{and} \quad A(u)(-w) \geq 0,$$

from which follows:

$$A(u)(w) = 0 \quad \forall w \in K.$$

We see that weak equations are contained in the class of variational inequalities. Hence, the latter one describes a more general setting.

6.5.0.1 On the notation

In the literature, the notation of the duality pairing is often found for describing weak forms and variational inequalities. This notation highlights in which spaces we are working. The previous statements read:

Formulation 6.23 (Abstract variational inequality – duality pairing notation)**.** *Find $u \in K$ such that*

$$\langle A(u), \varphi - u \rangle \geq 0 \quad \forall \varphi \in K.$$

Here, we see more easily that $A(u) \in V^*$ and $\varphi - u \in V$.

If K is a linear subspace (and not only a set!) of V, then

$$\varphi := u \pm w \in K,$$

for each $w \in K$. Since the semi-linear form $A(\cdot)(\cdot)$ is linear in the second argument, we obtain:

$$\langle A(u), w \rangle \geq 0 \quad \text{and} \quad \langle A(u), -w \rangle \geq 0,$$

from which it follows:

$$\langle A(u), w \rangle = 0 \quad \forall w \in K.$$

6.5.0.2 Minimization of functionals Let $\{V, \|\cdot\|\}$ be a reflexive, real Hilbert space, K be a nonempty, closed and convex subset of V, and $F : K \to \mathbb{R}$ be a real functional defined on K.

Formulation 6.24 (Minimization problem)**.** *Find $u \in K$ such that*

$$F(u) = \inf_{v \in K} F(v),$$

or equivalently

$$F(u) \leq F(v) \quad \forall v \in K.$$

Example 6.25. *An example for F is the elastic potential energy. See, for instance, the equivalence relations in [178].*

Definition 6.26 (Convexity)**.** *A functional $F : K \to \mathbb{R}$ is convex if and only if*

$$F(\theta u + (1-\theta)v) \leq \theta F(u) + (1-\theta)F(v) \quad u, v \in K, \quad 0 \leq \theta \leq 1.$$

The functional is strictly convex if $<$ holds true.

Definition 6.27. *The Gâteaux-derivative of F is denoted as*

$$\lim_{\varepsilon \to 0} \frac{d}{d\varepsilon} F(u + \varepsilon v) = F'(u)(v) = \langle F'(u), v \rangle.$$

Theorem 6.28. *Let K be a subset of a normed linear space V and let $F : K \to \mathbb{R}$ be Gâteaux differentiable. If u is a minimizer of F in K, then u can be characterized as follows:*

1. *If K is a nonempty, closed, and convex subset of V, then*
$$F'(u)(v-u) \geq 0 \quad \forall v \in K.$$

2. *If K is a nonempty, closed, and convex subset of V and $u \in int(K)$, then*
$$F'(u)(v) \geq 0 \quad \forall v \in K.$$

3. *If K is a linear variety with respect to w (i.e., K is a linear subspace translated by a factor w with respect to the origin), then*
$$F'(u)(v) = 0 \quad \forall v \in V,$$
and $v - w \in K$.

4. *If K is a linear subspace of V, then*
$$F'(u)(v) = 0 \quad v \in K.$$

Proof. The proof is sketched in the following three bullet points:

1. The set K is convex. Thus $w := u + \theta(v-u) \in K$ for $\theta \in [0,1]$ and for every $u, v \in K$. If u is a minimizer of K, then we have
$$F(u) \leq F(w) = F(u + \theta(v-u)) \quad \forall v \in K.$$
Differentiating yields:
$$\frac{d}{d\theta}[F(u + \theta(v-u)) - F(v)]|_{\theta=0} = F'(u)(v-u) \geq 0.$$

2. If $u \in int(K)$ and any $v \in K$. Then, there exists $\varepsilon \geq 0$ such that $u + \varepsilon v \in K$. Therefore, we insert the defined element into the previous characterization and obtain:
$$F'(u)(u + \varepsilon v - u) = F'(u)(\varepsilon v) \geq 0.$$
Since ε is arbitrary, it implies the assertion.

3. Let $K = \{w\} + M$, where M is a linear subspace of V. Then, $w - v \in K$ implies $v \in M$. The same holds for $-v \in M$. Then,

$$F'(u)(v) = 0 \quad v \in V \qquad \text{such that } v - w \in K.$$

4. Take $w = 0$. Then,

$$F'(u)(v) = 0 \quad v \in K.$$

□

7 Numerical modeling part I: Regularization and discretization

7.1 Outline of the three numerical modeling chapters

In Chapter 5, we derived variational phase-field fracture models on the continuous level. Especially, we derived three different formulations:

- Formulation 5.39 using an inequality constraint;
- Formulation 5.41 using an inequality constraint and stress splitting into tensile and compressive parts;
- Formulation 5.48 using a strain history function and stress splitting.

7.1.1 Outline of Chapter 7

To carry out computer simulations we need to **regularize** and **discretize** these models. There, we mostly focus on Formulation 5.39 and provide sometimes hints to the other two formulations. In fact, Formulation 5.41 does not differ from the first one from an algorithmic point of view.

7.1.2 Outline of Chapter 8

In Chapter 8 we address the nonlinear and linear numerical solutions of the discretized problems.

7.1.3 Outline of Chapter 10

In Chapter 10, we address some special topics such as local mesh adaptivity, design of algorithms for imposing the irreversibiliy constraint, the crack width computation, and pressurized fractures.

7.1.4 Demonstration of numerical modeling in terms of excursus and simulations

The three numerical chapters are complemented with **excursuses on the obstacle problem** (Section 5.3, Section 8.13, Section 8.19, Section 10.13) and simplified settings (Section 10.4.3, Section 8.16) as well as detailed **numerical simulations** (Chapter 9, Chapter 11, Chapter 12, Chapter 14) using the full phase-field fracture formulation.

7.2 The crack irreversibility constraint

Basically, four procedures exist to treat the crack irreversibility:

1. Fixing crack nodes by Dirichlet values [27, 101];
2. Using a strain history function [117];
3. Using penalization [122, 169];
4. Using a primal-dual active set method [82];
5. Working with a Lagrange multiplier formulation [112].

To date, only partial comparisons have been performed (recently in [67]) and moreover, the basic methods have their advantages and shortcomings. In Section 9, we perform some further numerical comparisons.

7.2.1 Strain history function

Imposing the irreversibility constraint via a strain history field was discussed in Section 5.7.3.

7.2.2 Penalization/augmented Lagrangian

Regularizing the inequality constraint requires to discretize $\partial_t \varphi$ via

$$\begin{aligned} \partial_t \varphi &\approx \frac{\varphi - \varphi^{n-1}}{k} \leq 0, \\ \Rightarrow \quad \varphi &\leq \varphi^{n-1} \\ \Rightarrow \varphi - \varphi^{n-1} &\leq 0 \\ \Rightarrow \gamma[\varphi^{n+1} - \varphi^n]_+, \end{aligned}$$

with $\gamma > 0$, $[x]_+ := \max(0, x)$ and the time step size $k := t_{n+1} - t_n$. The idea is to introduce the penalization parameter $\gamma > 0$, which penalizes the PDE when the constraint is violated. Keep in mind, that mathematically one should show for $\gamma \to \infty$, that the *correct* PDE is obtained. Numerically $\gamma \gg 0$ will lead to ill-conditioned Jacobians (bad for linear solvers) and higher nonlinearities from which the nonlinear solver will suffer (see [133, p. 505]). We also refer to computations and available code for the obstacle problem in Section 8.13.

The second equation of the problem formulation looks like

$$A_2(U)(\Psi) + (\gamma[\varphi^{n+1} - \varphi^n]_+, \psi) = 0. \tag{19}$$

In comparison to Formulation 5.39, the variational Equation (19) is not an inequality any more:

Formulation 7.1 (Formulation 5.39 as a penalized version)**.** *For the loading steps $n = 0, 1, .., N$ find $U = (u, \varphi) \in V \times W$ with $\varphi(0) = \varphi_0$, such that*

$$A(U)(\Psi) = 0 \quad \forall \Psi = (w, \psi) \in V \times (H^1 \cap L^\infty),$$

where

$$A(U)(\Psi) = A_1(U)(\Psi) + A_2(U)(\Psi),$$

with

$$A_1(U)(\Psi) = 0 \quad \textit{and } A_2(U)(\Psi) = 0,$$

and

$$\begin{aligned} A_1(U)(\Psi) &= \Big([(1-\kappa)\varphi^2 + \kappa]\sigma, \nabla w\Big), \\ A_2(U)(\Psi) &= \big((1-\kappa)\varphi\sigma : e(u), \psi\big) + \left(-\frac{G_C}{\epsilon}(1-\varphi), \psi\right) + (G_C \epsilon \nabla\varphi, \nabla\psi) \\ &\quad + (\gamma[\varphi^{n+1} - \varphi^n]_+, \psi). \end{aligned}$$

Remark 7.2. *On the energy level $(\gamma[\varphi - \varphi^n]_+, \psi)$ reads [169] :*

$$\frac{\gamma}{2} ||[\varphi - \varphi^n]_+||^2_{L^2}$$

From a mathematical point of view, a 4th-order regularization would simplify the analysis and Newton's method since the problem has sufficient regularity to differentiate up to second-order derivatives.

$$\begin{aligned} &\Rightarrow \frac{\gamma}{4} ||[\varphi - \varphi^n]_+||^4_{L^2} \\ &\Rightarrow \textit{PDE-level } J' = \gamma([\varphi - \varphi^n]^3_+, \xi)_{L^2} \\ &\Rightarrow \textit{Jacobian in Newton's method } J'' = 3\gamma([\varphi - \varphi^n]^2_+ \delta e, \psi)_{L^2} \end{aligned}$$

See [130] for further details.

Remark 7.3. *Beside a simple penalization, in recent publications, there are proposed other, better, techniques how to integrate the inequality constraint into the main problem formulation. An augmented Lagrangian formulation for VPFF is described in [169]; original references on the augmented Lagrangian method are, for instance, [169] and [133, Sect. 17].*

7.2.3 Primal-dual active set method

Another approach is a so-called **primal-dual active set** method (see e.g., [91]) or **semi-smooth Newton** method [86, 87]. In [86], the equivalence of primal-dual active set methods and semi-smooth Newton methods is shown. In active set methods, the domain is split into an active and a nonactive set. In the former, the PDE is not solved because the constraint holds. This method is applied to VPFF in Heister et al. [82] and details are given as well in Section 8.17.

In the current section, we briefly recall the basic idea, following closely [86, Sect. 2]. We consider the following complementarity problem (similar to our Section 5.3.3):

Formulation 7.4. *Find a solution u and a Lagrange multiplier p such that*

$$
\begin{aligned}
Au + p &= f,\\
u &\leq g,\\
p &\geq 0,\\
(p, u-g) &= 0.
\end{aligned}
$$

Here, $(\cdot,\cdot)$ denotes the scalar product in $\mathbb{R}^n$, A is a well-posed matrix (for details see [86]), and $f, g \in R^n$.

If A is symmetric positive definite, the above system is the optimality system to

$$
\begin{aligned}
&\min E(u) = \frac{1}{2}(u, Au) - (f, u),\\
&\text{subject to } u \leq g.
\end{aligned}
$$

The trick is to reformulate the complementarity conditions above in an equivalent fashion:

$$
\begin{aligned}
&u \leq g, \quad \geq 0, \quad (p, u-g) = 0,\\
\Leftrightarrow\quad &C(u,p) = 0, \quad \text{where } C(u,p) = p - \max\{0, p + c(u-g)\}.
\end{aligned}
$$

The parameter c is assumed to be positive and the max-operation has to be understood component-wise. With this, we obtain the equivalent formulation:

Formulation 7.5. *Find a solution u and a Lagrange multiplier p such that*

$$
\begin{aligned}
Au + p &= f,\\
C(u,p) &= 0.
\end{aligned}
$$

Now, we are already in the position to formulate a primal-dual active set strategy. For a given pair (u, p), the constraint formulation $C(u,p) = 0$ is used as a predictor step to determine the next active and inactive sets. These two sets are given by:

$$
\begin{aligned}
N &= \{i : p_i + c(u-g)_i \leq 0\},\\
A &= \{i : p_i + c(u-g)_i > 0\}.
\end{aligned}
$$

Here, i denote the index of the solutions (for instance located at the DoFs when using Q_1 finite elements).

The algorithm reads:

Algorithm 7.6 (Primal-dual active set algorithm)**.** *The basic primal-dual active set algorithm has the following steps:*

1. *Initialize u^0 and p^0. Set $l = 0$.*
2. *Set N^k and A^k.*
3. *Solve*

$$
\begin{aligned}
Au^{k+1} + p^{k+1} &= f,\\
u^{k+1} &= g \text{ on } A^k, \quad \text{and} \quad p^{k+1} = 0 \text{ on } N^k.
\end{aligned}
$$

4. Stop, if a given stopping criterion is fulfilled. If not, set $k \to k+1$ and go to Step 2.

Here, $u^{k+1} = g$ has to be understood component-wise for each index i: $u_i^{k+1} = g_i$ for $i \in A^k$.

Remark 7.7. *In [86], this algorithm is interpreted as a semi-smooth Newton method with corresponding convergence results.*

7.3 Spatial discretization

We assume that the reader has taken classes on numerical methods for partial differential equations in which spatial and temporal discretization have been discussed. To recapitulate the most important features we refer to our own notes [178] or the famous references cited therein. Spatial discretization of VPFF does not require aligned meshes or enriched functions.

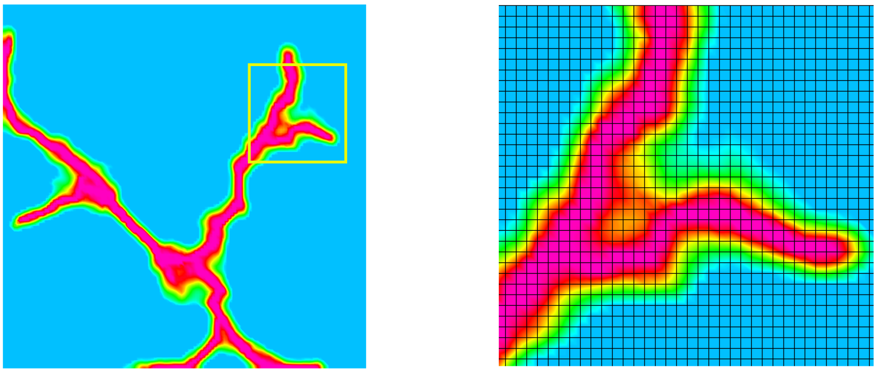

Figure 16: A small fracture network and zoom-in at the right. The mesh is regular and the fracture can vary freely. The main requirement is that $\varepsilon > h$, which is illustrated with the help of the green transition zone that goes over several mesh cells.

One can use the FEM (finite element method) or IGA (isogeometric analysis), depending on the problem setup and the implementation environment.

From Figure 16, we would heuristically think that the following should hold true:

Proposition 7.8. *It must hold that*

$$\varepsilon > h,$$

where ε is the phase-field regularization parameter and h the spatial discretization parameter.

Corollary 7.9. *When $\epsilon \to 0$ is of interest, then h must be very small requiring small meshes in the crack region. This is in particular, challenging in three dimensions. Local mesh adaptivity is indispensable.*

Remark 7.10. *We proof these relationships for a simplified situation in Section 7.3.2.*

7.3.1 A penalized, spatially discretized version

After spatial discretization, we arrive formally at the following result:

Formulation 7.11 (Formulation 7.1, spatially discretized with FEM)**.** *Let $V_h \subset V$ and $W_h \subset W$ be conforming finite element spaces. For the loading steps $n = 0, 1, .., N$ find $U_h = (u_h, \varphi_h) \in V_h \times W_h$ with $\varphi_h(0) = \varphi_{0,h}$, such that*

$$A(U_h)(\Psi_h) = 0 \quad \forall \Psi_h = (w_h, \psi_h) \in V_h \times W_h,$$

where

$$A(U_h)(\Psi) = A_1(U_h)(\Psi_h) + A_2(U_h)(\Psi_h),$$

with

$$A_1(U_h)(\Psi_h) = 0 \quad \textit{and } A_2(U_h)(\Psi_h) = 0,$$

and

$$\begin{aligned} A_1(U_h)(\Psi_h) &= \Big([(1-\kappa)\varphi_h^2 + \kappa]\sigma(u_h), \nabla w_h\Big), \\ A_2(U_h)(\Psi_h) &= \big((1-\kappa)\varphi_h\sigma(u_h) : e(u_h), \psi_h\big) + \Big(-\frac{G_C}{\epsilon}(1-\varphi_h), \psi_h\Big) + (G_C\epsilon\nabla\varphi_h, \nabla\psi_h) \\ &\quad + (\gamma[\varphi_h^{n+1} - \varphi_h^n]_+, \psi_h). \end{aligned}$$

7.3.2 Numerical analysis of a simplified, prototype problem

In this section, we perform a rigorous numerical analysis for the dependencies of the phase-field model parameters κ, ε and the spatial discretization parameter h.

7.3.2.1 Assumptions We make the following assumptions:

1. Laplace terms; no elasticity
2. Equations are decoupled
3. We assume as much regularity as needed; in particular, for the term $\varphi|\nabla u|^2$, please see below for details
4. No inequality constraint; thus ‘only’ linear equations
5. No proof that the iteration schemes that couples both equations will converge
6. Assuming $\kappa = o(\varepsilon)$ holds true [9, 33]
7. Even so that we will write in the following $\kappa \to 0$ and $\varepsilon \to 0$, the limits φ and u are not in H^1 and therefore the usual interpolation estimates cannot hold true. However, in order to get a ‘feeling’ what might happen and how the relationships could be, we assume:

$$\begin{aligned} \kappa &\to \kappa_{end} > 0 \quad \text{with } \kappa \gg \kappa_{end} \\ \varepsilon &\to \varepsilon_{end} > 0 \quad \text{with } \varepsilon \gg \varepsilon_{end}. \end{aligned}$$

This means that we always stay away from $\varepsilon = 0$ and $\kappa = 0$, but we allow for variations towards zero.

7.3.2.2 Equations Solid:

$$\begin{aligned} -\nabla \cdot ((\varphi^2 + \kappa)\nabla u) &= f \quad \text{in } \Omega \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned}$$

with $\varphi^2 \in L^\infty$ and $\kappa > 0$.

Phase-field:

$$\begin{aligned} -\varepsilon\Delta\varphi - \frac{1}{\varepsilon}(1-\varphi) &= g \quad \text{in } \Omega, \\ \varepsilon\partial_n\varphi &= 0 \quad \text{on } \partial\Omega, \end{aligned}$$

with $\varepsilon > 0$. The right hand side function would be actually

$$g = -\varphi|\nabla u|^2$$

(see our models before). If we assume $|\nabla u|^2 \in L^\infty$ (which is not clear at all if this is true), then we could easily do the numerical analysis for

$$-\varepsilon\Delta\varphi - \frac{1}{\varepsilon}(1-\varphi) + \varphi|\nabla u|^2 = 0.$$

Since the second and the third term are of the same order in φ we do not expect any difficulties for this system. For simplicity, we restrict our attention to

$$-\varepsilon\Delta\varphi - \frac{1}{\varepsilon}(1-\varphi) = g,$$

for some g not depending on φ.

7.3.2.3 Goals

1. Relation κ versus h;
2. Relation ε versus h.

7.3.2.4 Procedure Since both equations are linear we use the standard assumptions known for the Lax-Milgram lemma. The Lax-Milgram lemma gives the well-posedness of a partial differential equation. We can apply it, if we have a problem of the following form:
Find $u \in V : a(u)(w) = l(w)\ \forall w \in V$ with

i) $|l(u)| \leq C\|u\|_V$,

ii) $|a(u)(w) \leq \gamma\|u\|_V\|w\|_V$ (continuity of the bilinear form),

iii) $a(u)(u) \geq \alpha\|u\|_V^2$ (coercivity).

Then, the lemma of Lax-Milgram holds true.

Further, we use the Céa lemma and the Galerkin-orthogonality. If we have the best approximation property, we use interpolation results in order to obtain powers in h. By comparison, we will then see which choices for ε and κ are feasible and how these choices influence the convergence order measured in h. We will need the following norms: $\|u\|_V := \|u\|_{H^1}$ and $\|u\|_{H^1} = \int u^2 + (\nabla u)^2\, dx$. So let us start with the proof of the Céa lemma for the solid equation:

Solid Equation: Céa Lemma

Proof. The weak form of the solid equation reads as follows:

$$((\varphi^2 + \kappa)\nabla u, \nabla w) = (f, w) \qquad w \in V,$$

where $(\varphi^2 + \kappa)$ is $\alpha(\varphi, \kappa)$. In the crack it holds $\varphi^2 = 0$, which implies

$$(\kappa\nabla u, \nabla w) = (f, w).$$

We define $a(u, w) := (\alpha(\kappa)\nabla u, \nabla w)$ with $\alpha(\kappa) = \kappa$. For this Poisson problem we want to show

$$a(u, u) \geq \alpha(\kappa)||u||_V^2.$$

The difficulty is that $0 \leq \alpha_0 \leq \alpha(\kappa) \leq \alpha_\infty$. in particular, it is difficult to find α_0 if $\alpha(\kappa)$ tends to zero.
Let us start with a measurement of the discretization error in the norm V:

$$\begin{aligned}\alpha(\kappa)||u-u_h||_V^2 \leq a(u-u_h, u-u_h) = a(u-u_h, u-w_h)\ \forall\phi_h \in V_h \\ \leq \gamma||u-u_h||_V||u-\phi_h|| \\ \Rightarrow\ ||u-u_h||_V \leq \frac{\gamma}{\alpha(\kappa)}||u-w_h||_V.\end{aligned}$$

The first step is based on the Galerkin-orthogonality. $V_h \subset V$ is the conforming finite element space.

Remark 7.12 (Galerkin-Orthogonality). *We remember the Galerkin-orthogonality in detail. Assume to have a problem of the form:*
Find $u \in V$ such that

$$a(u)(w) = l(w) \; \forall w \in V.$$

On the discrete level, the problem reads as:
Find $u_h \in V_h$ such that

$$a(u_h)(w_h) = l(w_h) \; \forall w_h \in V_h.$$

In V_h, it follows:

$$\begin{aligned} a(u)(w_h) - a(u_h)(w_h) &= 0, \\ a(u-u_h)(w_h) &= 0. \end{aligned}$$

Further, it follows

$$\begin{aligned} a(u-u_h, u-w_h) &\leq \gamma \|u-u_h\| \|u-w_h\|, \\ \|u-u_h\|_V &\leq \frac{\gamma}{\alpha(\kappa)} \|u-w_h\|. \end{aligned}$$

So we proofed the Céa Lemma for the solid equation for $\varphi = 0$. For $\varphi \neq 0$, for example $\alpha(\varphi, \kappa) = (\varphi^2 + \kappa)$ and $\kappa \approx 0$ we just get an useless error estimate if $\kappa \to 0$. If for instance, $\varphi = 0$ and $\alpha(\varphi, \kappa) \approx \kappa$, it holds

$$\|u-u_h\| \leq \gamma \|u-w_h\|.$$

□

In a next step we propose the interpolation estimates with the goal of quantitative estimates in terms of the mesh size h.

Lemma 7.13. *(Interpolation estimates)*

$$\begin{aligned} ||u-u_h||_{L^2} &\leq ch^2 ||u||_{H^2} = \mathcal{O}(h^2), \\ ||u-i_h u||_{H^1} &\leq ch ||u||_{H^2} = \mathcal{O}(h). \end{aligned} \tag{20}$$

Starting from Equation (20), we choose $\phi_h := i_h u \in V_h$:

$$\begin{aligned} ||u-u_h||_V &\leq \frac{\gamma}{\alpha(\kappa)} ||u - i_h u||_V \\ &\leq \frac{c\gamma}{\alpha\kappa} h ||u||_{H^2} \\ &= c_2 \frac{h}{\alpha(\kappa)} ||u||_{H^2} = \mathcal{O}(h), \end{aligned}$$

if κ is fixed. Now compare h and $\alpha(\kappa)$. For simplicity we define $\kappa := \alpha(\kappa)$:

$$||u-u_h||_V \leq c_3 \frac{h}{\kappa} ||u||_{H^2}.$$

Here, we see the direct relation between h and κ. To get a useful estimate, we need $c_4 \frac{h}{\kappa} \to 0 \Leftrightarrow h = \sigma(\kappa)$. We have to treat two cases:

i) If $\kappa = ch$:

$$\Rightarrow ||u-u_h||_V \leq c_4 \frac{h}{h} ||u||_{H^2} = c_4 ||u||_{H^2}$$

⇒ no convergence in h in the V-norm (here H^1)

ii) $\kappa = ch^{\beta}, 0 < \beta < 1$

$$\begin{aligned} &\Leftrightarrow h = \sigma(\kappa) \\ &\Rightarrow ||u - u_h||_V \leq c_4 \frac{h}{h^{\beta}} ||u||_{H^2} = c_4 h^{1-\beta} ||u||_{H^2} \end{aligned}$$

$\Rightarrow$ convergence for $0 < \beta < 1$
$\Rightarrow$ the smaller β, the better the convergence rate

Proposition 7.14 (H^1-norm).

$$||u - u_h||_{H^1} = \mathcal{O}(h^{1-\beta}), \quad 0 < \beta < 1.$$

Proposition 7.15 (L^2-norm). *We assume that the Aubin-Nitsche lemma holds true. Then,*

$$||u - u_h||_{L^2} \leq c \frac{h^2}{h^{\beta}} ||u||_{H^2} = ch^{2-\beta} ||u||_{H^2}.$$

Remark 7.16. *For $\beta = 1$, we still have*

$$\|u - u_h\|_{L^2} = \mathcal{O}(h),$$

for which it holds

$$\|u - u_h\|_{H^1} = \mathcal{U}(1),$$

which implies no convergence.

Phase-field Equation: Céa Lemma Consider the phase-field equation. Find $\varphi : \Omega \to [0, 1]$ such that

$$\begin{aligned} -\epsilon \Delta \varphi - \frac{1}{\epsilon}(1 - \varphi) &= g, \quad \text{in } \Omega, \\ \epsilon \partial_n \varphi &= 0 \quad \text{on } \partial\Omega, \end{aligned}$$

with $g := -\tilde{\varphi} |\nabla u|^2$. This is assumed to be given since we work in a partitioned approach. Further, we assume $\tilde{\varphi}$ to be given with $\tilde{\varphi} \approx \varphi$ such that we are allowed to use it as a right hand side. Now, we apply the same ideas as before for the solid equation. The equation, we have to treat now, is of elliptic type as before.
The weak form of the phase-field problem looks like

$$(\epsilon \nabla \varphi, \nabla \psi) - \frac{1}{\epsilon}(1 - \varphi, \psi) = (g, \psi).$$

Let us first show the Céa Lemma for the phase-field equation:

$$\begin{aligned} \alpha ||\varphi - \varphi_h||_V^2 &\leq a(\varphi - \varphi_h, \varphi - \varphi_h) \\ &= a(\varphi - \varphi_h, \varphi - \psi_h) \\ &\leq \gamma ||\varphi - \varphi_h||_V ||\varphi - \psi_h||_V \\ \Rightarrow ||\varphi - \varphi_h||_V &\leq \frac{\gamma}{\alpha} ||\varphi - \psi_h||. \end{aligned}$$

We have to pay attention with the dependencies of γ and α on ϵ. The parameters γ and α depend on ϵ.

Continuity It holds

$$|a(\varphi,\psi)| = |\epsilon(\nabla\varphi,\nabla\psi) - \frac{1}{\epsilon}(1-\varphi,\psi)|.$$

$$\begin{aligned}
\text{Here: } |\epsilon(\varphi,\psi)| &\leq \epsilon c_1 ||\varphi||_{H^1}||\psi||_{H^1},\\
|-\frac{1}{\epsilon}(1-\varphi,\psi)| &\leq \frac{1}{\epsilon}c_2||1-\varphi||_{L^2}||\psi||_{L^2},\\
|a(\varphi,\psi)| &\leq c_1||\varphi||_{H^1}||\psi||_{H^1} + \frac{1}{\epsilon}c_2||1-\varphi||_{L^2}||\psi||_{L^2}\\
&\leq \epsilon c_1||\varphi||_{H^1}||\psi||_{H^1} + \frac{c_2}{\epsilon}||\varphi||_{H^1}||\psi||_{H^1}\\
&\leq \max\left(c_1\epsilon, \frac{c_2}{\epsilon}\right)||\varphi||_{H^1}||\psi||_{H^1},
\end{aligned}$$

where we define the maximum function as X.

Coercitivity It holds

$$\begin{aligned}
a(\varphi,\varphi) &= \epsilon(\nabla\varphi,\nabla\varphi) - \frac{1}{\epsilon}(1-\varphi,\varphi)\\
&= \epsilon(\nabla\varphi,\nabla\varphi) + \frac{1}{\epsilon}(\varphi,\varphi) - \frac{1}{\epsilon}(1,\varphi), \quad 0\leq\varphi\leq 1\\
&\geq \epsilon(\nabla\varphi,\nabla\varphi) + \frac{1}{\epsilon}(\varphi,\varphi) - \frac{1}{2\epsilon}\|1\|^2 - \frac{1}{2\epsilon}\|\varphi\|^2\\
&= \epsilon(\nabla\varphi,\nabla\varphi) + \frac{1}{\epsilon}(\varphi,\varphi) - c \geq \ \min\left(\epsilon,\frac{1}{2\epsilon}\right)[(\nabla\varphi,\nabla\varphi)+(\varphi,\varphi)] - c\\
&= \min\left(\epsilon,\frac{1}{2\epsilon}\right)||\varphi||^2_{H^1} - c.
\end{aligned}$$

with $c := \frac{1}{2\epsilon}\|1\|^2$.

With $\alpha := \alpha(\epsilon)$, we can simplify the notation and we get

$$a(\varphi,\varphi) \geq \alpha||\varphi||^2_V, \quad \alpha := \alpha(\epsilon) = \tilde{\alpha}\epsilon.$$

The definition of $\tilde{\alpha}$ is determined by $\epsilon(\nabla\varphi,\nabla\psi)$ and not $\frac{1}{\epsilon}(1-\varphi,\psi)$ for $\epsilon\to 0$.
Let us recall the following problem, for instance along with Hanke-Bourgeois [77][Sect. 92, p. 690]:

$$\begin{aligned}
-\nabla\cdot(\sigma\nabla u) + cu &= f,\\
u &= 0,
\end{aligned}$$

with $0 < \sigma_0 \leq \sigma \leq \sigma_\infty$ and $0 \leq c \leq c_\infty$.
For this problem, it holds the Céa lemma as follows:

$$||u-u_h||_V \leq c\frac{\max(\sigma_\infty,c_\infty)}{c_\Omega\sigma_0}||u-\phi_h||_V.$$

Now the Céa Lemma for the PFF is stated as:

$$\begin{aligned}
||\varphi-\varphi_h||_V &\leq \frac{\max\left(c_1\epsilon,\frac{c_2}{\epsilon}\right)}{c\alpha(\epsilon)}||\varphi-\psi_h||_V\\
&= \tilde{c}\frac{\max\left(\epsilon,\frac{1}{\epsilon}\right)}{\alpha}||\varphi-\psi_h||_V.
\end{aligned}$$

For $\epsilon\to 0$:

$$\begin{aligned}
||\varphi-\varphi_h||_V &\leq \tilde{c}\frac{\frac{1}{\epsilon}}{\epsilon}||\varphi-\psi_h||_V\\
&= \tilde{c}\frac{1}{\epsilon^2}||\varphi-\psi_h||_V.
\end{aligned}$$

For the error estimates use the interpolation estimates 7.13 and the last equation above:

$$\begin{aligned}||\varphi - \varphi_h||_V &\leq \tilde{c}\frac{1}{\epsilon^2}||\varphi - \psi_h||_V \\ &= \tilde{c}\frac{1}{\epsilon^2}h||\varphi||_{H^2}.\end{aligned}$$

Ideally $c\frac{h}{\epsilon^2} \to 0$, which implies that $h \ll \epsilon$. Let us discuss some practical choices of ϵ:

i) $\epsilon^2 = h \Rightarrow \epsilon = \sqrt{h}$:
$\Rightarrow ||\varphi - \varphi_h||_V \leq c||\varphi||_{H^2} = \mathcal{O}(1)$ (no convergence)

ii) $\epsilon = h^{\frac{\beta}{2}}$: $\Rightarrow ||\varphi - \varphi_h||_V \leq ch^{1-\beta}||\varphi||_{H^2}$ with $0 < \beta < 1$

Proposition 7.17 (H^1-norm - 1st version)**.** *Choose* $\epsilon = h^{\frac{\beta}{2}},\ 0 < \beta < 1$. *Then*

$$||\varphi - \varphi_h||_V = \mathcal{O}(h^{1-\beta}).$$

So we get with the Aubin-Nitsche trick for the L^2-norm:

$$\begin{aligned}&||\varphi - \varphi_h||_{L^2} \leq c\frac{h^2}{\epsilon^2}||\varphi||_{H^2} \\ \Rightarrow\ &\epsilon = h^{\beta},\ 0 < \beta < 1 \Rightarrow ||\varphi - \varphi_h||_{L^2} = \mathcal{O}(h^{1-\beta}).\end{aligned}$$

So for the H^1- norm for the PFF it holds:

$$\begin{aligned}||\varphi - \varphi_h||_V &\leq c\frac{h}{\epsilon^2}||\varphi||_{H^2} \\ \Rightarrow\ &\epsilon = h^{\beta},\ 0 < \beta < 1 \\ \Rightarrow ||\varphi - \varphi_h||_{L^2} &\leq c\frac{h}{h^{2\beta}}||\varphi||_{H^2} = ch^{1-2\beta}||\varphi||_{H^2}\end{aligned}$$

To achieve convergence we have to choose:

$$\frac{1}{2} > \beta.$$

Proposition 7.18 (H^1-norm - 2nd version)**.** *For* $\epsilon = h^{\beta}$ *with* $0 < \beta < \frac{1}{2}$, *we have:*

$$||\varphi - \varphi_h||_V = \mathcal{O}(h^{1-2\beta}).$$

Again let us consider the L^2-norm, but an improved result:

Proposition 7.19. *Choosing* $\epsilon = h^{\frac{\beta}{2}}$. *Then*

$$\begin{aligned}||\varphi - \varphi_h||_{L^2} &\leq \frac{h^2}{\epsilon^2}||\varphi||_{H^2} = \frac{h^2}{(h^{\frac{\beta}{2}})^2}||\varphi||_{H^2} \\ &= h^{2-\beta}||\varphi||_{H^2} \\ &= \mathcal{O}(h^{2-\beta}), \quad 0 < \beta < 1.\end{aligned}$$

7.3.2.5 Summary

If ϵ is a constant, we get the best convergence in terms of the discretization parameter h.

MATHEMATICALLY	NUMERICALLY	PRACTICE
$h \ll \kappa \ll \epsilon$ $h = o(\epsilon),\ h = o(\kappa),\ \kappa = o(\epsilon)$	$h \ll \epsilon$, $h \ll \kappa$	$\kappa \sim 10^{-10}$ and $\epsilon = 2h$ or $\epsilon = 4h$

7.4 A phase-field model for dynamic fracture (extensive exercise); towards space-time

In this section, we concentrate on an extensive exercise that focuses on dynamic fracture.

Remark 7.20. *For a better adaptation to our lecture, the model is modified. Note, that the version presented below is commonly not used in the literature in this specific form.*

7.4.1 The model (partially incomplete - to be augmented in the exercise)

Let $\Omega \subset \mathbb{R}^n$, $n = 2$ be open and let $I := [0, T]$ with $T > 0$ being the end time value. Find $u : \Omega \times I \to \mathbb{R}^n$ and $\varphi : \Omega \times I \to \mathbb{R}$ such that

$$\begin{aligned}
\rho_s \partial_t^2 u - \nabla \cdot (\varphi^2 \nabla u) &= f \quad \text{in } \Omega \times I,\\
\partial_t^2 \varphi + \varphi |\nabla u|^2 - \varepsilon \Delta \varphi + \frac{1}{\varepsilon}(1 - \varphi) &\leq 0 \quad \text{in } \Omega \times I,\\
\partial_t \varphi &\leq 0 \quad \text{in } \Omega \times I,\\
u &= 0 \quad \text{on } \partial\Omega \times I,\\
\varepsilon \partial_n \varphi &= 0 \quad \text{on } \partial\Omega \times I,\\
u &= u_0 \quad \text{in } \Omega \times \{0\},\\
\partial_t u &= v_0 \quad \text{in } \Omega \times \{0\},\\
\varphi &= \varphi_0 \quad \text{in } \Omega \times \{0\},\\
\partial_t \varphi &= \chi_0 \quad \text{in } \Omega \times \{0\}.
\end{aligned}$$

7.4.2 Tasks

Exercise 15. *Using the above model, we consider the following tasks:*

1. *Write $\varphi^2 \nabla u$ component-wise.*
2. *Write $\varphi |\nabla u|^2$ component-wise.*
3. *The above model is incomplete because the compatibility condition is missing. Please write down this condition.*
4. *Which coupling: volume or surface (interface)?*
5. *Is the model linear or nonlinear?*
6. *Is it a variational equation or variational inequality?*
7. *Is the model stationary or time-dependent?*
8. *Derive the weak formulation.*
9. *Formulate a compact semi-linear form to describe the weak form.*
10. *Using the compact semi-linear form, write down the compact abstract problem as a root-finding problem.*
11. *Derive the energy formulation corresponding to the weak form?*
12. *Double-check that the energy derivative of the energy formulation yields the PDE in a weak form (actually this is also known as Euler-Lagrange equations).*
13. *What does it mean that the energy formulation exists?*
14. *Manipulate the PDE such that no energy formulation exists.*
15. *Derive a quasi-static version by manipulating the above model.*

16. *How can we linearize the problem?*

17. *Use penalization, see Section 5.3.4 and rewrite the variational inequality as variational equation.*

18. *In order to derive a Galerkin formulation for the discretization in space and time, we need to get rid of the second-order derivatives in time. Write down a model with first-order derivatives in time. How many solution variables do we have now?*
 Hint: For space-time formulations of second-order hyperbolic PDEs, we also refer to Section 7.4.4.

7.4.3 Bochner spaces - space-time functions

For the correct function spaces for formulating time-dependent variational forms, we define the Bochner integral. Let $I := [0,T]$ with $0 < T < \infty$ be a bounded time interval with end time value T. For any Banach space X and $1 \leq p \leq \infty$, the space

$$L^p(I,X)$$

denotes the space of L^p integrable functions f from the time interval I into X. This is a Banach space, the so-called Bochner space, with the norm, see [181],

$$\|v\|_{L^p(I,X)} := \Big(\int_I \|v(t)\|_X^p \, dt\Big)^{1/p}$$
$$\|v\|_{L^\infty(I,X)} := \operatorname*{ess\,sup}_{t\in I} \|v(t)\|_X.$$

Example 7.21. *For instance, we can define a H^1-space in time:*

$$H^1(I,X) = \Big\{ v \in L^2(I,X) |\, \partial_t v \in L^2(I,X) \Big\}.$$

Functions that are even continuous in time, i.e., $u : I \to X$, are contained in spaces like $C(I;X)$ with

$$\|u\|_{C(I;X)} := \max_{0\leq t\leq T} \|u(t)\| < \infty.$$

Definition 7.22 (Weak derivative of space-time functions)**.** *Let $u \in L^1(I;X)$. A function $v \in L^1(I;X)$ is the weak derivative of v, denoted as*

$$\partial_t u = v,$$

if

$$\int_0^T \partial_t \varphi(t) u(t)\, dt = -\int_0^T \varphi(t) v(t)\, dt,$$

for all test functions $\varphi \in C_c^\infty(I)$.

In particular, the following result holds:

Theorem 7.23 ([59])**.** *Assume $v \in L^2(I,H_0^1)$ and $\partial_t v \in L^2(I,H^{-1})$. Then, v is continuous in time, i.e.,*

$$v \in C(I,L^2),$$

(possibly redefined on a set of measure zero). Furthermore, the mapping

$$t \mapsto \|v(t)\|^2_{L^2(X)}$$

is absolutely continuous with

$$\frac{d}{dt}\|v(t)\|^2_{L^2(X)} = 2\langle \frac{d}{dt} v(t), v(t)\rangle,$$

for a.e. $0 \leq t \leq T$.

Proof. See Evans [59], Theorem 3 in Section 5.9.2. □

Remark 7.24. *The importance of this theorem lies in the fact that now the point-wise prescription of initial conditions makes sense in weak formulations.*

Remark 7.25. *More details of these spaces by means of the Bochner integral can be found in [49, 181] and also [59].*

7.4.4 Space-time formulation of the elastic wave equation

The above model consists of two wave equations, which are both space- and time-dependent. One of the most elegant ways for a variational formulation (neglecting the inequality constraints for a moment) is a space-time formulation. We briefly introduce the concept in this section.

Let us consider a second-order hyperbolic PDE as follows:

Formulation 7.26. *Let $\Omega \subset \mathbb{R}^n$ be open and let $I := [0, T]$ with $T > 0$. Find $u : \Omega \times I \to \mathbb{R}$ and $\partial_t u : \Omega \times I \to \mathbb{R}$ such that*

$$\begin{aligned} \rho\partial_t^2 u - \nabla \cdot (a\nabla u) &= f \quad \text{in } \Omega \times I, \\ u &= 0 \quad \text{on } \partial\Omega_D \times I, \\ a\partial_n u &= 0 \quad \text{on } \partial\Omega_N \times I, \\ u &= u_0 \quad \text{in } \Omega \times \{0\}, \\ v &= v_0 \quad \text{in } \Omega \times \{0\}. \end{aligned}$$

We need proper functions spaces required for the weak formulation. Let us denote L^2 and H^1 as the usual Hilbert spaces and H^{-1} as the dual space to H^1. For the initial functions u_0 and v_0 we assume:

- $u_0 \in H_0^1(\Omega)^n$;
- $v_0 \in L^2(\Omega)^n$.

For the right hand side (source term) we assume

- $f \in L^2(I, H^{-1}(\Omega))$, where $L^2(\cdot,\cdot)$ is a Bochner space; see Section 7.4.3.

We introduce the following short-hand notation:

- $H := L^2(\Omega)^n$;
- $V := H_0^1(\Omega)^n$;
- V^* is the dual space to V;
- $\bar{H} := L^2(I, H)$;
- $\bar{V} := \{v \in L^2(I, V) |\, \partial_t v \in \bar{H}\}$.

Remark 7.27. *We note that the initial values are well-defined due to Theorem 7.23.*

Theorem 7.28. *If the operator $A := -\nabla \cdot (a\nabla u)$ satisfies the coercivity estimate:*

$$(Au, u) \geq \beta \|u\|_1^2, \quad u \in V, \quad \beta > 0,$$

then there exists a unique weak solution with the following properties:

- $u \in \bar{V} \cap C(\bar{I}, V)$;
- $\partial_t u \in \bar{H} \cap C(\bar{I}, H)$;
- $\partial_t^2 u \in L^2(I, V^*)$.

Proof. See Lions and Magenes, Lions or Wloka. □

Definition 7.29. *The previous derivations allow us to define a compact space-time function space for the wave equation:*

$$\bar{X} := \{v \in \bar{V} |\, v \in C(\bar{I}, V), \partial_t v \in C(\bar{I}, H), \partial_t^2 v \in L^2(I, V^*)\}.$$

To design a Galerkin time discretization, we need to get rid of the second-order time derivatives and therefore usually the wave equation is re-written in terms of a mixed first-order system:

Formulation 7.30. *Let $\Omega \subset \mathbb{R}^n$ be open and let $I := [0,T]$ with $T > 0$. Find $u : \Omega \times I \to \mathbb{R}$ and $\partial_t u = v : \Omega \times I \to \mathbb{R}$ such that*

$$\begin{aligned} \rho\partial_t v - \nabla\cdot(a\nabla u) &= f \quad \text{in } \Omega\times I,\\ \rho\partial_t u - \rho v &= 0 \quad \text{in } \Omega\times I,\\ u &= 0 \quad \text{on } \partial\Omega_D\times I,\\ a\partial_n u &= 0 \quad \text{on } \partial\Omega_N\times I,\\ u &= u_0 \quad \text{in } \Omega\times\{0\},\\ v &= v_0 \quad \text{in } \Omega\times\{0\}. \end{aligned}$$

To derive a space-time formulation, we first integrate formally in space. Below we explain why we choose the notation for the test function in this way:

$$\begin{aligned} A_v(U)(\psi^u) &= (\rho\partial_t v, \psi^u) + (a\nabla u, \nabla\psi^u) - (f, \psi^u),\\ A_u(U)(\psi^v) &= (\rho\partial_t u, \psi^v) - (\rho v, \psi^v). \end{aligned}$$

And then in time:

$$\begin{aligned} \bar{A}_v(U)(\psi^u) &= \int_I \big((\rho\partial_t v, \psi^u) + (a\nabla u, \nabla\psi^u) - (f, \psi^u)\big)\, dt + (v(0) - v_0, \psi^u(0)),\\ \bar{A}_u(U)(\psi^v) &= \int_I \big((\rho\partial_t u, \psi^v) - (\rho v, \psi^v)\big)\, dt + (u(0) - u_0, \psi^v(0)). \end{aligned}$$

The total problem reads:

Formulation 7.31. *Find $U = (u,v) \in X_u \times X_v$ with $X_u = X$ and $X_v := \{w \in \bar{H} | w \in C(\bar{I}, H), \partial_t w \in L^2(I, V^*)\}$ such that*

$$\bar{A}(U)(\Psi) = 0 \quad \forall \Psi = (\psi^u, \psi^v) \in X_u \times X_v,$$

where

$$\bar{A}(U)(\Psi) := \bar{A}_v(U)(\psi^u) + \bar{A}_u(U)(\psi^v).$$

Such a formulation is the starting point for the discretization: either in space-time or using a sequential time-stepping scheme and for instance FEM in space.

Remark 7.32. *Be careful that the trial and test functions are switched in the wave equation. What does this mean? Formally, we compute in the first equation v using as test function ψ^u which belongs to the trial space of u. Vice versa for the second equation. That this choice can be justified can be seen in the following: the variable u needs higher-order regularity in space, therefore we work with the space X_u. Once differentiating in time, yields less regularity for v. In order to apply the correct boundary conditions and partial integration in the first equation $\bar{A}_v$ (to determine v), we need an object from the stronger space, namely X_u as test function.*

Remark 7.33. *These ideas can be used to formulate a space-time formulation of the dynamic phase-field fracture problem presented in Section 7.4.*

7.5 Further models of time-dependent phase-field fracture

In this section, we present further models of time-dependent fracture in which time derivatives do not only appear in the inequality constraint, but also in the main equations. For specific details on dynamic fracture models we refer to [24, 29, 88, 104, 105, 152] and extensions to phase-field coupled to fluid-structure interaction and fluid-filled fractures are discussed in [172] and [183], respectively.

We state three models:

1. only time-derivative in phase-field equation
2. wave equation in elasticity
3. both: wave equation and time-derivative

7.5.1 Quasi-static displacements, time-dependent phase-field system

Let $\Omega \subset \mathbb{R}^n$, $n = 2$ be open and let $I := [0, T]$ with $T > 0$ being the end time value. Define $g(\varphi) := (1-\kappa)\varphi^2+\kappa$. Find $u : \Omega \times I \to \mathbb{R}^n$ and $\varphi : \Omega \times I \to \mathbb{R}$ such that

$$\begin{aligned}
-\nabla \cdot (g(\varphi)\nabla u) &= f \quad \text{in } \Omega \times I,\\
\partial_t\varphi + \varphi|\nabla u|^2 - \varepsilon\Delta\varphi + \frac{1}{\varepsilon}(1-\varphi) &\leq 0 \quad \text{in } \Omega \times I,\\
\partial_t\varphi &\leq 0 \quad \text{in } \Omega \times I,\\
\big(\partial_t\varphi + \varphi|\nabla u|^2 - \varepsilon\Delta\varphi + \frac{1}{\varepsilon}(1-\varphi)\big)(\partial_t\varphi) &= 0 \quad \text{in } \Omega \times I,\\
u &= 0 \quad \text{on } \partial\Omega \times I,\\
\varepsilon\partial_n\varphi &= 0 \quad \text{on } \partial\Omega \times I,\\
\varphi &= \varphi_0 \quad \text{in } \Omega \times \{0\}.
\end{aligned}$$

7.5.2 Wave equation for displacements, quasi-static phase-field system

Let $\Omega \subset \mathbb{R}^n$, $n = 2$ be open and let $I := [0, T]$ with $T > 0$ being the end time value. Define $g(\varphi) := (1-\kappa)\varphi^2+\kappa$. Find $u : \Omega \times I \to \mathbb{R}^n$ and $\varphi : \Omega \times I \to \mathbb{R}$ such that

$$\begin{aligned}
\rho_s\partial_t^2 u - \nabla \cdot (g(\varphi)\nabla u) &= f \quad \text{in } \Omega \times I,\\
\varphi|\nabla u|^2 - \varepsilon\Delta\varphi + \frac{1}{\varepsilon}(1-\varphi) &\leq 0 \quad \text{in } \Omega \times I,\\
\partial_t\varphi &\leq 0 \quad \text{in } \Omega \times I,\\
\big(\varphi|\nabla u|^2 - \varepsilon\Delta\varphi + \frac{1}{\varepsilon}(1-\varphi)\big)(\partial_t\varphi) &= 0 \quad \text{in } \Omega \times I,\\
u &= 0 \quad \text{on } \partial\Omega \times I,\\
\varepsilon\partial_n\varphi &= 0 \quad \text{on } \partial\Omega \times I,\\
u &= u_0 \quad \text{in } \Omega \times \{0\},\\
\partial_t u &= v_0 \quad \text{in } \Omega \times \{0\}.
\end{aligned}$$

7.5.3 Wave equation for displacements, time-dependent phase-field system

Let $\Omega \subset \mathbb{R}^n$, $n = 2$ be open and let $I := [0, T]$ with $T > 0$ being the end time value. Find $u : \Omega \times I \to \mathbb{R}^n$ and $\varphi : \Omega \times I \to \mathbb{R}$ such that

$$\begin{aligned}
\rho_s\partial_t^2 u - \nabla \cdot (\varphi^2\nabla u) &= f \quad \text{in } \Omega \times I,\\
\partial_t\varphi + \varphi|\nabla u|^2 - \varepsilon\Delta\varphi + \frac{1}{\varepsilon}(1-\varphi) &\leq 0 \quad \text{in } \Omega \times I,\\
\partial_t\varphi &\leq 0 \quad \text{in } \Omega \times I,\\
\big(\partial_t\varphi + \varphi|\nabla u|^2 - \varepsilon\Delta\varphi + \frac{1}{\varepsilon}(1-\varphi)\big)(\partial_t\varphi) &= 0 \quad \text{in } \Omega \times I,\\
u &= 0 \quad \text{on } \partial\Omega \times I,\\
\varepsilon\partial_n\varphi &= 0 \quad \text{on } \partial\Omega \times I,\\
u &= u_0 \quad \text{in } \Omega \times \{0\},\\
\partial_t u &= v_0 \quad \text{in } \Omega \times \{0\},\\
\varphi &= \varphi_0 \quad \text{in } \Omega \times \{0\},\\
\partial_t\varphi &= \chi_0 \quad \text{in } \Omega \times \{0\}.
\end{aligned}$$

8 Numerical modeling part II: Linearizations, nonlinear and linear solvers

In this chapter, we face the numerical solution of the discretized problem. The nonlinearities require nonlinear solvers. In a next step, the inner linear subproblems must be solved.

8.1 Linearization techniques

We discuss linearization techniques in the following sections. The idea is to provide algorithmic frameworks that serve for the implementation. Concerning Newton's method for general problems there is not that much theory; see e.g., [51]. In general, one can say that in many problems the theoretical assumptions are not met, but nevertheless Newton's method works well in practice.

Our list of linearization methods is as follows:

- Fixed-point iteration;
- Linearization via time-lagging;
- Extrapolation in time;
- Newton's method.

8.2 Fixed-point iteration - general procedure

In ODE computations, applying a fixed-point theorem, namely the Banach fixed point theorem, is called a **Picard iteration**. The basic idea is to introduce an iteration using an index k and to linearize the nonlinear terms by taking these terms from the previous iteration $k-1$.

This is best illustrated in terms of an example. We assume to observe the PDE

$$-\Delta u + u^2 = f.$$

The variational formulation reads:

$$(\nabla u, \nabla \phi) + (u^2, \phi) = (f, \phi) \quad \forall \phi \in V.$$

An iterative scheme is constructed as follows:

Algorithm 8.1 (Fixed-point iterative scheme)**.** *For $k = 1, 2, 3, \ldots$ we seek $u^{k+1} \in V$ such that*

$$(\nabla u^k, \nabla \phi) + (u^k u^{k-1}, \phi) = (f, \phi) \quad \forall \phi \in V,$$

until a stopping criterion is fulfilled (choice one out of four):

- *Error criterion:*

$$\|u^k - u^{k-1}\| < TOL \quad \textit{(absolute)}$$
$$\|u^k - u^{k-1}\| < TOL\|u^k\| \quad \textit{(relative)}$$

- *Residual criterion:*

$$\|(\nabla u^k, \nabla \phi_i) + ([u^2]^k, \phi_i) - (f, \phi_i)\| < TOL, \quad \textit{(absolute)}$$
$$\|(\nabla u^k, \nabla \phi_i) + ([u^2]^k, \phi_i) - (f, \phi_i)\| < TOL\|(f, \phi_i)\|, \quad \textit{(relative)}$$

for all $i = 1, \ldots, dim(V_h)$.

Remark 8.2. *For time-dependent PDEs, a common linearization can be*

$$(u^2, \phi) \to (u\ u^{n-1}, \phi)$$

where $u^{n-1} := u(t_{n-1})$ is the solution of the previous time step. In this case, no additional fixed-point iteration needs to be constructed.

8.3 Partitioned (staggered) schemes for coupled problems

For coupled problems, very often, partitioned (or also so-called staggered) schemes are employed. The idea is to iterate between the different equations per time step.

Find $U = (u, v) \in V \times V$ such that:

$$\begin{aligned} A_u(U)(\psi^u) &= F(\psi^u), \\ A_v(U)(\psi^v) &= F(\psi^v). \end{aligned}$$

The idea is to iterate between both PDEs:

Algorithm 8.3. *Given an initial iteration value v^0. For $k = 1, 2, 3, \ldots$ iterate:*

$$\begin{aligned} A_u(u^k, v^{k-1})(\psi^u) &= F(\psi^u), \\ A_v(u^k, v^k)(\psi^v) &= F(\psi^v). \end{aligned}$$

Check the stopping criterion:

$$\max(\|u^k - u^{k-1}\|, \|v^k - v^{k-1}\|) < TOL.$$

If the stopping criterion is fulfilled, stop. If not, increment $k \to k + 1$.

8.4 Staggered fixed-point iteration applied to phase-field fracture

We consider Formulation 5.39 and solve the equations

$$\begin{aligned} A_1((u,\varphi))(w) &= \Big([(1-\kappa)\varphi^2 + \kappa]\sigma, \nabla w\Big), \\ A_2((u,\varphi))(\psi-\varphi) &= \big((1-\kappa)\varphi\sigma : e(u), \psi - \varphi\big) + \left(-\frac{G_C}{\epsilon}(1-\varphi), \psi - \varphi\right) + (G_C\epsilon\nabla\varphi, \nabla(\psi-\varphi)), \end{aligned}$$

with a fixed-point scheme.

Algorithm 8.4. *At a given loading (time) step n at t_n, let the initial iteration values be given as:*

$$u^0 := u^{n,0} := u^{n-1}, \qquad \varphi^0 := \varphi^{n,0} := \varphi^{n-1}.$$

We iterate for $k = 1, 2, \ldots$:

$$\begin{aligned} \textit{Given } \varphi^{k-1} \in W, \quad \textit{find } u^k \in V: \quad & A_1((u^k, \varphi^{k-1}))(w) = 0 \quad \forall w \in V, \\ \textit{Given } u^k \in V, \quad \textit{find } \varphi^k \in W: \quad & A_2((u^k, \varphi^k))(\psi - \varphi^k) \le 0 \quad \forall \psi \in W, \end{aligned}$$

until (for example, an absolute stopping criterion is fulfilled):

$$\max\{\|u^k - u^{k-1}\|, \|\varphi^k - \varphi^{k-1}\|\} < TOL.$$

8.5 Linearization of φ via extrapolation in time

A challenge in phase-field-based fracture formulations is related to the term

$$\big((1-\kappa)\varphi^2 + \kappa\big)\, \sigma^+(u);$$

see the Formulations 5.39 (here simply σ rather than σ^+), 5.41, or 5.48.

The related energy term is not convex simultaneously in both solution variables u and φ, and requires sophisticated solution algorithms.

8.5.1 Prototype examples

Two propositions on convexification are discussed in the following with the help of a simple example:

Proposition 8.5 (Convexity).

$$\begin{aligned} & E(x,y) = (\kappa + x^2)y^2 && \rightarrow \textit{ simplified problem} \\ x=0: \quad & E(0,y) = \kappa y^2 && \Rightarrow \textit{ strictly convex in } y \\ y=0: \quad & E(x,0) = 0 && \Rightarrow \textit{ only convex in } x, \end{aligned}$$

but if we choose a different energy functional we get a strictly convex function in both arguments:

$$\begin{aligned} & E(x,y) = (\kappa + x^2)y^2 + x^2 && \rightarrow \textit{ simplified problem} \\ x=0: \quad & E(0,y) = \kappa y^2 && \Rightarrow \textit{ strictly convex} \\ y=0: \quad & E(x,0) = x^2 && \Rightarrow \textit{ strictly convex.} \end{aligned}$$

With the knowledge of Proposition 8.5 we can transfer the results on the energy functional for phase-field fracture:

Proposition 8.6 (Convexification for phase-field fracture).

$$E(u,\varphi) = \frac{1}{2}\int_B (\kappa + \varphi^2)|\nabla u|^2 + \frac{1}{2}\int \frac{1}{\epsilon}(1-\varphi)^2 + |\nabla \varphi|^2.$$

Via Proposition 8.5 it holds that the term

$$\frac{1}{2}\int (\kappa + \tilde{\varphi}^2)|\nabla u|^2,$$

is strictly convex in u.

8.5.2 A linear-in-time extrapolation

One possible approach is based on a linear-in-time extrapolation of φ in Equation (8) in order to replace the non-convex $4th$-order term by a given coefficient in front of the elasticity.

Definition 8.7 (A linear extrapolation). *The extrapolated φ at time t_n is denoted by $\tilde{\varphi} := \tilde{\varphi}^n$ and defined by:*

$$\tilde{\varphi}^n = \varphi^{n-2}\frac{t_n - t_{n-1}}{t_{n-2} - t_{n-1}} + \varphi^{n-1}\frac{t_n - t_{n-2}}{t_{n-1} - t_{n-2}}.$$

This approximation is inserted into $\left((1-\kappa)\tilde{\varphi}^2 + \kappa\right)\sigma^+(u)$.

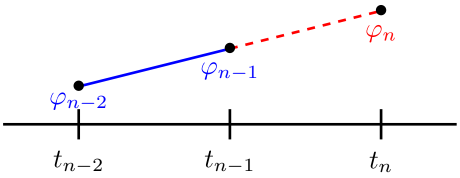

Figure 17: Linear extrapolation in time.

On one hand, this procedure is heuristic since for quasi-static fracture propagation, we cannot proof sufficient regularity in time; namely, the phase-field solution φ can have jumps in time. On the other hand, in [82], it has been numerically demonstrated that this procedure is robust.

8.6 An iteration on the extrapolation

We present a simple, but effective method to enhance the quality of the previous extrapolation. As shown in [175], for fast crack growth, the extrapolation introduces an approximation error.

To this end, we improve the extrapolation by an additional iteration. The method reads:

Algorithm 8.8 (Iterating on the extrapolation)**.** *We assume to be in time step t_n.*

1. *Let φ^{n-2} and φ^{n-1} be the given two previous time step solutions;*
2. *Set $\varphi^{n,-2} := \varphi^{n-2}$ and $\varphi^{n,-1} := \varphi^{n-1}$*
3. *Construct the linear extrapolation:*
$$\tilde{\varphi}^{n,0} = \varphi^{n,-2}\frac{t_n - t_{n-1}}{t_{n-2} - t_{n-1}} + \varphi^{n,-1}\frac{t_n - t_{n-2}}{t_{n-1} - t_{n-2}}$$
4. *Set $u^{n,0} := u^{n-1}$ and $\varphi^{n,0} := \varphi^{n-1}$;*
5. *For $i = 1, \dots, N$:*
   a) *Find $(u^{n,i}, \varphi^{n,i})$ by solving the displacement phase-field system with $u^{n,i-1}, \varphi^{n,i-1}, \tilde{\varphi}^{n,i-1}$.*
   b) *Construct a new extrapolation:*
$$\tilde{\varphi}^{n,i} = \varphi^{n,i-2}\frac{t_n - t_{n-1}}{t_{n-2} - t_{n-1}} + \varphi^{n,i-1}\frac{t_n - t_{n-2}}{t_{n-1} - t_{n-2}}.$$
   c) *Increment $i \to i + 1$.*
6. *Set $u^n := u^{n,N}$ and $\varphi^n := \varphi^{n,N}$.*

8.7 Differentiation in Banach spaces

For Newton's method, we need to differentiate the PDE in order to construct the Jacobian matrix. Despite that we assumed discretized problems, we introduce differentiation on the continuous level, thus differentiation in Banach spaces.

Definition 8.9 (Directional derivative in a Banach space)**.** *Let V and W be normed vector spaces and let $U \subset V$ be non-empty. Let $f : U \to W$ be a given mapping. If the limit*
$$f'(v)(h) := \lim_{\varepsilon \to 0} \frac{f(v + \varepsilon h) - f(v)}{\varepsilon}, \quad v \in U, h \in V,$$
exists, then $f'(v)(h)$ is called the directional derivative of the mapping f at v in the direction h. If the directional derivative exists for all $h \in V$, then f is called directionally differentiable at v.

Remark 8.10 (Notation)**.** *Often, the direction h is denoted by δv in order to highlight that the direction is related with the variable v. This notation is useful, when several solution variables exist and several directional derivatives need to be computed.*

Remark 8.11. *The definition of the directional derivative in Banach spaces is in perfect agreement with the definition of derivatives in $\mathbb{R}$ at $x \in \mathbb{R}$ (see [97]):*
$$f'(x) := \lim_{\varepsilon \to 0} \frac{f(x + \varepsilon) - f(x)}{\varepsilon},$$
and in $\mathbb{R}^n$ we have (see [98]):
$$f'(x)(h) := \lim_{\varepsilon \to 0} \frac{f(x + \varepsilon h) - f(x)}{\varepsilon}.$$
A function is called differentiable when all directional derivatives exist in the point $x \in \mathbb{R}^n$ (similar to the Gâteaux derivative). The derivatives in the directions $e_i, i = 1, \dots, n$ of the standard basis are the well-known ***partial derivatives****.*

Definition 8.12 (Gâteaux derivative)**.** *Let the assumptions hold as in Definition 8.9. A directional-differentiable mapping as defined in Definition 8.9, is called Gâteaux-differentiable at $v \in U$, if there exists a linear continuous mapping $A : U \to W$ such that*

$$f'(v)(h) = A(h),$$

holds true for all $h \in U$. Then, A is the Gâteaux derivative of f at v and we write $A = f'(v)$.

Remark 8.13. *The Gâteaux derivative is computed with the help of directional deriatives and it holds $f'(v) \in L(U, W)$. If $W = \mathbb{R}$, then $f'(v) \in U^*$.*

Definition 8.14 (Fréchet derivative)**.** *A mapping $f : U \to W$ is Fréchet-differentiable at $u \in U$ if there exists an operator $A \in L(U, W)$ and a mapping $r(u, \cdot) : U \to W$ such that it holds for each $h \in U$ with $u + h \in U$:*

$$f(u + h) = f(u) + A(h) + r(u, h),$$

with

$$\frac{\|r(u, h)\|_W}{\|h\|_U} \to 0 \quad \text{for } \|h\|_U \to 0.$$

The operator $A(\cdot)$ is the Fréchet derivative of f at u and we write $A = f'(u)$.

Definition 8.15 (Equivalent formulation denoting derivatives)**.** *In the literature and above, we have (at least) three common notations and ways to compute directional derivatives:*

$$\begin{aligned} f'(u)(h) &= \lim_{\varepsilon \to 0} \frac{f(u + \varepsilon h) - f(u)}{\varepsilon} \\ &= \frac{d}{d\varepsilon} f(u + \varepsilon h)|_{\varepsilon = 0} \\ &= f(u + h) - f(u) - r(u, h). \end{aligned}$$

Example 8.16. *The above bilinear form $a(u, \phi) = (\nabla u, \nabla \phi)$ is Fréchet-differentiable in the first argument u (of course also in the second argument, but u is the variable we are interested in):*

$$a(u + h, \phi) = (\nabla(u + h), \nabla \phi) = \underbrace{(\nabla u, \nabla \phi)}_{=a(u,\phi)} + \underbrace{(\nabla h, \nabla \phi)}_{=a'_u(u,\phi)(h)}.$$

Here the remainder term is zero, i.e., $r(u, h) = 0$, because the bilinear form is linear in u. Thus the Fréchet derivative of $a(u, \phi) = (\nabla u, \nabla \phi)$ is $a'_u(u, \phi)(h) = (\nabla h, \nabla \phi)$.

Second example is $J(u) = \int u^2 \, dx$. Here:

$$J(u + h) = \int (u + h)^2 \, dx = \underbrace{\int u^2 \, dx}_{J(u)} + \underbrace{\int 2uh \, dx}_{=A(h)} + \underbrace{\int h^2 \, dx}_{=r(u,h)}.$$

That $J(u)$ is really Fréchet-differentiable we need to check whether

$$\frac{\|r(u, h)\|_W}{\|h\|_U} \to 0 \quad \text{for } \|h\|_U \to 0.$$

Here we have

$$\int h^2 \, dx = \|h\|_V^2,$$

and therefore:

$$\frac{\|h\|_V^2}{\|h\|_V} = \|h\|_V.$$

For $h \to 0$ we clearly have $\|h\|_V \to 0$. Consequently, the directional derivative of $J(u) = \int u^2 \, dx$ is

$$J'(u)(h) = A(h) = \int_\Omega 2uh \, dx.$$

Example 8.17. *Another example:*

$$T(u) = u^2,$$

then

$$T'_u(u)(h) = 2u \cdot h.$$

Or for semi-linear forms:

$$a(u)(\phi) = (u^2, \phi).$$

Differentiation in the first argument yields:

$$a'_u(u)(h, \phi) = (2u \cdot h, \phi).$$

Example 8.18. *A further example:*

$$T(u) = u^3$$

then

$$T(u + h) = (u + h)^3 = u^3 + 3u^2h + 3uh^2 + h^3.$$

Here, we have now four terms (previously we always had the exact number of terms to describe $T(u), A(h)$ and $r(u, h)$.

- *The identification of $T(u) = u^3$ is obvious.*
- *According to the theory $A(h)$ is a linear operator in h. Therefore: $A(h) = 3u^2h$.*
- *The rest goes into $r(u, h) = 3uh^2 + h^3$.*

To check that we deal with a Fréchet derivative, we need to verify

$$\frac{\|r(u,h)\|_W}{\|h\|_U} \to 0 \quad \textit{for } \|h\|_U \to 0.$$

Here it is important to remark that u is fixed and we only vary in h. Since we have h^2 and h^3 in the nominator, but only h in the denominator, one can indeed check that $T(u) = u^3$ is Fréchet-differentiable.

8.8 Newton's method in $\mathbb{R}$ - the Newton-Raphson method

Let $f \in C^1[a, b]$ with at least one point $f(x) = 0$, and $x_0 \in [a, b]$ be a so-called initial guess. The task is to find $x \in \mathbb{R}$ such that

$$f(x) = 0.$$

In most cases it is impossible to calculate x explicitly. Rather we construct a sequence of iterates $(x_k)_{k\in\mathbb{R}}$ and hopefully reach at some point

$$|f(x_k)| < TOL, \quad \text{where } TOL \text{ is small, e.g., } TOL = 10^{-10}.$$

For all Newton derivations one has to start with a Taylor expansion. In our lecture we do this as follows. Let us assume that we are at x_k and can evaluate $f(x_k)$. Now we want to compute this next iterate x_{k+1} with the unknown value $f(x_{k+1})$. The Taylor expansion gives us:

$$f(x_{k+1}) = f(x_k) + f'(x_k)(x_{k+1} - x_k) + o(x_{k+1} - x_k)^2$$

We assume that $f(x_{k+1}) = 0$ (or very close to zero $f(x_{k+1}) \approx 0$). Then, x_{k+1} is the sought root and via neglecting the higher-order terms we obtain:

$$0 = f(x_k) + f'(x_k)(x_{k+1} - x_k).$$

Thus:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, 2, \ldots. \tag{21}$$

This iteration is allowed as long as $f'(x_k) \neq 0$.

Remark 8.19. *We see that Newton's method can be written as*

$$x_{k+1} = x_k + d_k, \quad k = 0, 1, 2, \ldots,$$

where the search direction is

$$d_k = -\frac{f(x_k)}{f'(x_k)}.$$

The iteration (21) terminates if a stopping criterion

$$\frac{|x_{k+1} - x_k|}{|x_k|} < TOL, \quad \text{or} \quad |x_{k+1} - x_k| < TOL, \quad \text{or } |f(x_{k+1})| < TOL \tag{22}$$

is fulfilled. All these tolerances (TOL) do not need to be the same, but sufficiently small and larger than machine precision.

Remark 8.20. *Newton's method belongs to fix-point iteration schemes with the iteration function:*

$$F(x) := x - \frac{f(x)}{f'(x)}. \tag{23}$$

For a fixed point $\hat{x} = F(\hat{x})$ it holds: $f(\hat{x}) = 0$.

The main result is given by:

Theorem 8.21 (Newton's method)**.** *The function $f \in C^2[a, b]$ has a root $\hat{x}$ in the interval $[a, b]$ and*

$$m := \min_{a \le x \le b} |f'(x)| > 0, \quad M := \max_{a \le x \le b} |f''(x)|.$$

Let $\rho > 0$ such that

$$q := \frac{M}{2m}\rho < 1, \quad K_\rho(\hat{x}) := \{x \in \mathbb{R} : |x - \hat{x}| \le \rho\} \subset [a, b].$$

Then, for any starting point $x_0 \in K_\rho(\hat{x})$, the sequence of iterations $x_k \in K_\rho(\hat{x})$ converges to the root $\hat{x}$. Furthermore, we have the a priori estimation

$$|x_k - \hat{x}| \le \frac{2m}{M} q^{2^k}, \quad k \in \mathbb{N},$$

and the a posteriori estimation

$$|x_k - \hat{x}| \le \frac{1}{m}|f(x_k)| \le \frac{M}{2m}|x_k - x_{k+1}|^2, \quad k \in \mathbb{N}.$$

Often, Newton's method is formulated in terms of a defect-correction scheme.

Definition 8.22 (Defect)**.** *Let $\tilde{x} \in \mathbb{R}$ be an approximation of the solution $f(x) = y$. The defect (or similarly the residual) is defined as*

$$d(\tilde{x}) = y - f(\tilde{x}).$$

Definition 8.23 (Newton's method as defect-correction scheme)**.**

$$\begin{aligned} f'(x_k)\delta x &= d_k, \quad d_k := y - f(x_k), \\ x_{k+1} &= x_k + \delta x, \quad k = 0, 1, 2, \ldots. \end{aligned}$$

The iteration is finished with the same stopping criterion as for the classical scheme. To compute the update δx we need to invert $f'(x_k)$:

$$\delta x = (f'(x_k))^{-1} d_k.$$

This step seems trivial but is the most critical one if we deal with problems in $\mathbb{R}^n$ with $n > 1$ or in function spaces. Because here, the derivative becomes a matrix. Therefore, the problem results in solving a linear equation system of the type $A\delta x = b$. Computing the inverse matrix A^{-1} is an expensive operation.

Remark 8.24. *This previous forms of Newton's method are already very close to the schemes that are used in research. We have to extend the scheme from $\mathbb{R}^1$ to higher dimensional cases such as nonlinear PDEs or optimization. Two important aspects in current research are the choice of*

- *good initial Newton guesses;*
- *globalization techniques.*

Two very good books on these topics, are [51, 133].

8.8.1 Newton's method: overview. Going from $\mathbb{R}$ to Banach spaces

Overview:

- Newton-Raphson (1D), find $x \in \mathbb{R}$ via iterating $k = 0, 1, 2, \ldots$ such that $x_k \approx x$ via:

$$\begin{aligned} \text{Find } \delta x \in \mathbb{R}: \quad & f'(x_k)\delta x = -f(x_k), \\ \text{Update:} \quad & x_{k+1} = x_k + \delta x. \end{aligned}$$

- Newton in $\mathbb{R}^n$, find $x \in \mathbb{R}^n$ via iterating $k = 0, 1, 2, \ldots$ such that $x_k \approx x$ via:

$$\begin{aligned} \text{Find } \delta x \in \mathbb{R}^n: \quad & F'(x_k)\delta x = -F(x_k), \\ \text{Update:} \quad & x_{k+1} = x_k + \delta x. \end{aligned}$$

  Here we need to solve a linear equation system to compute the update $\delta x \in \mathbb{R}^n$.

- Banach spaces, find $u \in V$, with $dim(V) = \infty$, via iterating $k = 0, 1, 2, \ldots$ such that $u_k \approx u$ via:

$$\begin{aligned} \text{find } \delta u \in V: \quad & F'(u_k)\delta u = -F(u_k), \\ \text{update:} \quad & u_{k+1} = u_k + \delta u. \end{aligned}$$

  Such a problem needs to be discretized and results again in solving a linear equation system in the defect step.

- Banach spaces, applied to variational formulations, find $u \in V$, with $dim(V) = \infty$, via iterating $k = 0, 1, 2, \ldots$ such that $u_k \approx u$ via:

$$\begin{aligned} \text{Find } \delta u \in V: \quad & a'(u_k)(\delta u, \phi) = -a(u_k)(\phi), \\ \text{Update:} \quad & u_{k+1} = u_k + \delta u. \end{aligned}$$

  As before, the infinite-dimensional problem is discretized resulting in solving a linear equation system in the defect step.

8.8.2 A basic algorithm for a residual-based Newton method

In this type of methods, the main criterion is a decrease of the residual in each step:

Algorithm 8.25 (Residual-based Newton's method)**.** *Given an initial guess x_0. Iterate for $k = 0, 1, 2, \ldots$ such that*

$$\begin{aligned} \textit{Find } \delta x \in \mathbb{R}^n: \quad & F'(x_k)\delta x_k = -F(x_k), \\ \textit{Update:} \quad & x_{k+1} = x_k + \lambda_k \delta x_k, \end{aligned}$$

with $\lambda_k \in (0, 1]$ (see the next sections how λ_k can be determined). A full Newton step corresponds to $\lambda_k = 1$. The criterion for convergence is the contraction of the residuals measured in terms of a discrete vector norm:

$$\|F(x_{k+1})\| < \|F(x_k)\|.$$

In order to save computational cost, close to the solution x^*, intermediate simplified Newton steps can be used. In the case of $\lambda_k = 1$ we observe

$$\theta_k = \frac{\|F(x_{k+1})\|}{\|F(x_k)\|} < 1.$$

If $\theta_k < \theta_{max}$, e.g., $\theta_{max} = 0.1$, then the old Jacobian $F'(x_k)$ is saved and used again in the next step $(k+1)$. Otherwise, if $\theta_k > \theta_{max}$, the Jacobian will be assembled. Finally, the stopping criterion is one of the following:

$$\begin{aligned} \|F(x_{k+1})\| &\leq TOL_N \quad \textit{(absolute)} \\ \|F(x_{k+1})\| &\leq TOL_N \|F(x_0)\| \quad \textit{(relative)} \end{aligned}$$

If the chosen stopping criterion is fulfilled, set $x^* := x_{k+1}$ and the (approximate) root x^* of the problem $F(x) = 0$ is found.

Remark 8.26. *A comprehensive overview and investigations of Newton's method and several variants can be found in [51].*

8.9 Inexact Newton

For large scale nonlinear PDEs, the inner linear systems are usually not solved with a direct method (e.g., LU), but with iterative methods (CG, GMRES, multigrid). Using such iterative methods yields an **inexact Newton scheme**. Here, we deal with two iterations:

- The outer nonlinear Newton iteration.
- The inner linear iteration.

Algorithm 8.27 (Inexact Newton method)**.** *Given an initial guess x_0. Iterate for $k = 0, 1, 2, \ldots$ such that*

$$\begin{aligned} \textit{Formulate } \delta x \in \mathbb{R}^n: \quad F'(x_k)\delta x_k &= -F(x_k), \\ \textit{Solve with an iterative linear method } F'(x_k)\delta x_k^i &= -F(x_k), \\ \textit{Check linear stopping criteria}\|\delta x_k^i - \delta x_k^{i-1}\| &< TOL_{Lin}, \\ \textit{Update: } x_{k+1} &= x_k + \lambda_k \delta x_k^i, \\ \textit{Check nonlinear stopping criteria, e.g., } \|x_{k+1} - x_k\| &< TOL_{New} \end{aligned}$$

with $\lambda_k \in (0, 1]$.

8.10 Details on the discretization and nonlinear/linear systems

8.10.1 Stationary linearized elasticity

As previously discussed, the spatial discretization is based upon a space V_h with a basis $\{\varphi_1, \ldots, \varphi_N\}$ where N is the dimension of this space. Here we solve the problem: Find $u_h \in V_h$ such that

$$A(u_h, \varphi_h) = F(\varphi_h) \quad \forall \varphi_h \in V_h.$$

This relation does in particular, hold for each test function $\varphi_i, i = 1, \ldots, N$:

$$A(u_h, \varphi_h^i) = F(\varphi_h^i) \quad \forall \varphi_h^i, i = 1, \ldots, N$$

The solution u_h we are seeking for, is a linear combination of all test functions, i.e., $u_h = \sum_{j=1}^N u_j \varphi_h^j$. Inserting this relation into the bilinear form $A(\cdot, \cdot)$, yields

$$\sum_{j=1}^N a(\varphi_h^j, \varphi_h^i) u_j = f(\varphi_h^i), \quad i = 1, \ldots, N.$$

It follows for the ingredients of the linear equation system:

$$u = (u_j)_{j=1}^N \in \mathbb{R}^N, \quad b = (f_i)_{i=1}^N \in \mathbb{R}^N, \quad A = a_{ij} = a(\varphi_h^J, \varphi_h^i).$$

The resulting linear equation systems reads:

$$Au = b.$$

Remark 8.28. *In the matrix A the rule is always as follows: the test function φ_h^i determines the row and the trial function φ_h^j the column. This does not play a role for symmetric problems (e.g., Poisson's problem) but becomes important for non-symmetric problems such as for example the Navier-Stokes problem (because of the convection term).*

Remark 8.29. *In the matrix, the degrees of freedom that belong to Dirichlet conditions (here only displacements since we assume Neumann conditions for the phase-field) are strongly enforced by replacing the corresponding rows and columns as usual in a finite element code.*

Example 8.30 (3D linearized elasticity)**.** *In this vector-valued problem, we have $3N$ test functions since the solution vector is an element of $\mathbb{R}^3$: $u_h = (u_h^{(1)}, u_h^{(2)}, u_h^{(3)})$. The boundary value problem looks as follows:*

$$\textit{Find } u_h \in V_h : \quad (\nabla u_h, \varphi_h) = (f, \varphi_h) \quad \forall \varphi_h \in V_h,$$

the bilinear form is tensor-valued:

$$a(\varphi_h^j, \varphi_h^i) = \int_\Omega \nabla\varphi_h^j : \nabla\varphi_h^i \, dx = \int_\Omega \begin{pmatrix} \partial_1\varphi_h^{1,j} & \partial_2\varphi_h^{1,j} & \partial_3\varphi_h^{1,j} \\ \partial_1\varphi_h^{2,j} & \partial_2\varphi_h^{2,j} & \partial_3\varphi_h^{2,j} \\ \partial_1\varphi_h^{3,j} & \partial_2\varphi_h^{3,j} & \partial_3\varphi_h^{3,j} \end{pmatrix} : \begin{pmatrix} \partial_1\varphi_h^{1,i} & \partial_2\varphi_h^{1,i} & \partial_3\varphi_h^{1,i} \\ \partial_1\varphi_h^{2,i} & \partial_2\varphi_h^{2,i} & \partial_3\varphi_h^{2,i} \\ \partial_1\varphi_h^{3,i} & \partial_2\varphi_h^{3,i} & \partial_3\varphi_h^{3,i} \end{pmatrix} dx.$$

Example 8.31 (Example in 1D)**.** *Let us illustrate and specify how the entries of the system matrix can be computed in 1D.*

8.10.2 Nonlinear elasticity

In this part, we go back to stationary problems but make them nonlinear. Rather than solving directly for v_h and p_h, we solve now for the (Newton) updates δv_h and δu_h.

The problem reads:

$$\text{Find } U_h \in X_h \text{ such that: } A(U_h)(\Psi_h) = F(\Psi_h) \quad \forall \Phi \in X_h.$$

To solve this problem, we need to iterate and we solve now for the updates and their representation with the help of the following shape functions:

$$\delta v_h = \sum_{j=1}^{N_V} \delta v_j \psi_h^{v,j}, \quad \delta u_h = \sum_{j=1}^{N_u} \delta u_j \psi_h^{u,j}$$

Given an initial guess $U_h^0 := \{v_h^0, u_h^0\}$, we must solve at each time t_n the problem:

Find $\delta U_h^n := \{\delta v_h^n, \delta u_h^n\} \in X_h$ such that: $A'(U_h^{n,l})(\delta U_h, \Psi_h) = -A(U_h^{n,l})(\Psi_h) + F(\Psi_h), \quad U_h^{n,l+1} = U_h^{n,l} + \delta U_h.$

Here,

$$\begin{aligned} A'(U_h^{n,l})(\delta U_h, \Psi_h) &= (\frac{1}{\delta t}\delta v_h, \psi_h^v) + \theta(F'(\delta u_h)\Sigma(u_h^n) + F(u_h^n)\Sigma'(\delta u_h), \nabla\psi_h^v) + (\frac{1}{\delta t}\delta u_h, \psi_h^v) - \theta(\delta v_h, \psi_h^u) \\ F(\Psi_h) &= (f_s, \psi_h^v) + (\frac{1}{\delta t}v_h^{n-1}, \psi_h^v) + (\frac{1}{\delta t}u_h^{n-1}, \psi_h^u) - (1-\theta)(F(u_h^{n-1})\Sigma(u_h^{n-1}), \nabla\psi_h^v) \\ &\quad + (1-\theta)(v_h^{n-1}, \psi_h^u), \\ A(U_h^{n,l})(\Psi_h) &= (\frac{1}{\delta t}v_h^n, \psi_h^v) + \theta(F(u_h^n)\Sigma(u_h^n), \nabla\psi_h^v) + (\frac{1}{\delta t}u_h^n, \psi_h^v) - \theta(v_h^n, \psi_h^u). \end{aligned}$$

The block structure reads:

$$\begin{pmatrix} \frac{1}{\delta t}M_{vv} & A_{vu} \\ M_{uv} & \frac{1}{\delta t}M_{uu} \end{pmatrix} \begin{pmatrix} \delta v \\ \delta u \end{pmatrix} = \begin{pmatrix} f_s - [\text{residual}] \\ 0 - [\text{residual}] \end{pmatrix}.$$

8.11 Abstract schemes for monolithic formulations and their numerical solution

8.11.1 Variational equations

In this section, we formulate an abstract scheme for variational formulations, which can be used to discretize several, nonlinear, coupled equations. Let us assume we have N equations given and possible inequalities are already regularized by penalization.

1. Formulate the given equations in variational forms:
$$\begin{aligned} A_1(U)(\Psi) &= F_1(\Psi)\\ A_2(U)(\Psi) &= F_2(\Psi)\\ &\vdots\\ A_N(U)(\Psi) &= F_N(\Psi)\end{aligned}$$

2. Formulate a compact short semi-linear form:
$$A(U)(\Psi) = \sum_{k=1}^{N} A_k(U)(\Psi).$$
This system is time-continuous, space-continuous and nonlinear.

3. Time discretization

   a) Classify A_k into time-dependent and stationary forms:
$$\begin{aligned} &\text{Stationary terms:} && A_k \quad \Rightarrow A_S\\ &\text{Time-dependent terms:} && A_k \quad \Rightarrow A_T, A_E, A_I\end{aligned}$$
   where A_S are stationary terms, A_T terms with time derivatives, A_E terms that are treated explicitly in a time-stepping scheme, A_I terms that are treated implicitly in a time-stepping scheme (e.g., pressure in Navier-Stokes).

   b) Approximate the continuous time derivative in A_T with a backward difference quotient yielding $A_T(U^{n,k}) \approx A_T(U^n)$.

   c) One-Step-θ: For $n = 1, \ldots, N_T$:
$$\begin{aligned} &\underbrace{A_T(U^{n,k})(\Psi) + \theta A_E(U^n)(\Psi) + A_I(U^n)(\Psi) + A_S(U^n)(\Psi)}_{=:A(U^n)(\Psi)}\\ &= -\underbrace{(1-\theta)A_E(U^{n-1})(\Psi)}_{=:A(U^{n-1})(\Psi)} + \underbrace{\theta F^n(\Psi) + (1-\theta)F^{n-1}(\Psi)}_{=:F^{n,n-1}(\Psi)}\end{aligned}$$

   Remark: Alternatively, the Fractional-Step-θ scheme can employed.

4. Spatial discretization:
$$A(U_h^n)(\Psi_h) = -A(U_h^{n-1})(\Psi_h) + F_h^{n,n-1}(\Psi_h)$$

5. Nonlinear solution (Newton):
$$\begin{aligned} &i) \quad \underbrace{A'(U_h^{n,j})(\delta U_h, \Psi_h)}_{=:AU} = \underbrace{-A(U_h^{n,j})(\Psi_h) - A(U_h^{n-1,j})(\Psi_h) + F_h^{n,n-1}(\Psi_h)}_{=:B}\\ &ii) \quad U_h^{n,j+1} = U_h^{n,j} + \omega \delta U_h\end{aligned}$$

6. Solve the linear system:
$$AU = B$$
If we use an iterative method, an appropriate preconditioner P^{-1} needs to be constructed such that
$$P^{-1}AU = P^{-1}B$$
where $P^{-1}A$ has a moderate condition number.

Exercise 16. *Take some time-dependent equation, for instance, nonstationary phase-field fracture (see Section 7.5) without inequality constraint and apply the above abstract steps.*

8.11.2 Variational inequalities: penalization

When terms are subject to inequality constraints, we present a modification of the previous algorithm using penalization.

1. Formulate the given equations in variational forms:
$$\begin{aligned}
&A_1(U)(\Psi) = F_1(\Psi)\\
&A_2(U)(\Psi) = F_2(\Psi)\\
&\vdots\\
&A_n(U)(\Psi) = F_N(\Psi)\\
&A_{n+1}(U)(\Psi - U) \geq F_N(\Psi - U)\\
&\vdots\\
&A_N(U)(\Psi - U) \geq F_N(\Psi - U)
\end{aligned}$$

2. Regularize the inequality constraints using penalty parameters $\gamma_{n+1}, \ldots, \gamma_N$ which yields:
$$\begin{aligned}
&A_{n+1}(U)(\Psi) - A_{\gamma_{n+1}}(U)(\Psi) = F_N(\Psi),\\
&\vdots\\
&A_N(U)(\Psi) - A_{\gamma_N}(U)(\Psi) = F_N(\Psi).
\end{aligned}$$

3. Formulate a compact short semi-linear form:
$$A(U)(\Psi) = \sum_{k=1}^{N} A_k(U)(\Psi) - \sum_{k=n+1}^{N} A_{\gamma_k}(U)(\Psi)$$
This system is regularized, time-continuous, space-continuous and nonlinear.

4. Apply the steps 3. - 6. from the previous section.

Exercise 17. *Take now a model from Section 7.5), with inequality constraint, and apply the above abstract steps.*

8.12 Loops

We briefly summarize in this section, how many loops are present in order to assemble the Jacobian matrix of a nonstationary nonlinear system of an initial/boundary value problem.

```
for n = 1,2,3, ... , N_T // timestepping
 for l = 1,2,3, ... , until ||res|| < TOL // Newton steps
  for K = 1,2,3, ... , N_K // all cell of the spatial mesh
   for i = 1,2,3, ... , N_V // Local degrees of freedom on a cell
    for j = 1,2,3, ... , N_V // Local degrees of freedom on a cell
     for q = 1,2,3, ... , N_Q // Quadrature points (in most cases Gaussian quadrature)
      A(U_h^{n,l})(\psi_h)
        -> A(\psi_h^j, \psi_h^i)_K
        -> \int_K A(\psi_h^j,\psi_h^i)
        -> \sum_{q=1}^{N_q} \omega_q A(\psi_h^j(\xi_q),\psi_h^i(\xi_q)).
```

Here, N_T, N_K, N_V and N_Q are a priori determined and denote the total number of time steps N_T, the total number of mesh cells N_K, the local total number of degrees of freedom on a cell N_V (e.g., Q_1^c has $N_V = 4$ for a scalar-valued problem in 2D and for example for Navier-Stokes: $3 * dim(Q_1^c) + dim P_1^{dc} = 4 + 4 + 4 + 3 = 15$ (local) degrees of freedom on a cell.) Finally, N_Q denotes the number of quadrature points to be used for the integration. Here, ω_q denote the quadrature weights and ξ_q the quadrature points.

8.13 Excursus II: Numerical modeling and implementation of the obstacle problem

In this section, we continue from the developments in Section 5.3 our developments in order to demonstrate regularization and Newton's method for a 'simple' model problem; namely the **obstacle problem**. The obstacle setting shares important similarities with phase-field fracture as we discussed previously: indeed the crack irreversibility constraint can be seen as an 'obstacle' condition in time. In the following, several details of the discretization, implementation, and simulation results are provided.

8.13.1 Problem statement (recalling)

We consider $\Omega = (0,1)^2$ and the following setting:

$$\begin{aligned} -\Delta u &\geq f \quad \text{in } \Omega, \\ u &\geq g \quad \text{in } \Omega, \\ (f + \Delta u)(g - u) &= 0 \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned}$$

with a right hand side $f = -1$. For comparison, we also recall the classical Poisson problem:

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned}$$

with $f = -1$. Plots of the numerical solutions are provided in Figure 18.

8.13.2 Variational forms

We set

$$\begin{aligned} a(u, \varphi) &= (\nabla u, \nabla \varphi), \\ l(\varphi) &= (f, \varphi). \end{aligned}$$

Define a convex set:

$$K := \{u \in H_0^1(\Omega) | \, u \geq g \text{ a.e. in } \Omega\}$$

and $V := H_0^1(\Omega)$. Then:

$$u \in K : \quad a(u, \varphi) \geq l(\varphi) \quad \forall \varphi \in K.$$

8.13.3 Regularization (penalization) of the inequality constraint

We regularize $u \geq g$ as follows:

$$\gamma([g-u]^+, \varphi),$$

with the (simple) penalization parameter $\gamma > 0$.

Remark 8.32. *Obviously, this term is non-smooth and renders the overall nonlinear problem for which we need a nonlinear solver (for instance Newton).*

Then:

$$u \in V: \quad a(u,\varphi) - \gamma([g-u]^+, \varphi) = l(\varphi) \quad \forall \varphi \in V.$$

8.13.4 Newton's method

We now formulate a root-finding problem. Redefine from before:

$$a_N(u)(\varphi) := a(u,\varphi) - \gamma([g-u]^+, \varphi) - l(\varphi).$$

Then:

$$u \in V: \quad a_N(u)(\varphi) = 0 \quad \forall \varphi \in V.$$

Newton: Initial guess $u_0 \in V$. Then iteration: for $k = 0, 1, 2, \ldots$:

$$\begin{aligned} u_k \in V: \quad a_N'(u_k)(\delta u, \varphi) &= -a_N(u_k)(\varphi) \quad \forall \varphi \in V \\ u_{k+1} &= u_k + \omega \delta u \end{aligned}$$

with a line search parameter $\omega \in (0,1]$. And with

$$a_N'(u_k)(\delta u, \varphi) := (\nabla \delta u, \nabla \varphi) + \gamma(\delta u, \varphi)_{B(u_k)}$$

with the set

$$B(u_k) := \{x \in \Omega | u_k(x) < g(x)\}.$$

8.13.5 Discretization

Use FEM. Conforming method: $V_h \subset V$ with $V_h := \{\varphi_1, \ldots, \varphi_M\}$ and $\dim(V_h) = M$. Take as FEM, for instance, linear (bilinear) elements, i.e., hat functions in 1D.

For a usual linear problem, we would try to solve for $u_{h,k} \in V_h$. Since the problem is nonlinear, we cannot determine $u_{h,k}$ directly, but use the nonlinear Newton iteration. Here, we determine $\delta u_h \in V_h$, which means:

$$\delta u_h = \sum_{j=1}^M u_j \varphi_j.$$

Then:

$$\begin{aligned} u_{h,k} \in V_h: \quad a_N'(u_{h,k})(\delta u_h, \varphi_h) &= -a_N(u_{h,k})(\varphi_h) \quad \forall \varphi_h \in V_h \\ u_{h,k+1} &= u_{h,k} + \omega \delta u_h \end{aligned}$$

The defect solution (first line of Newton's method) can be re-written as:

$$\begin{aligned} & a_N'(u_{h,k})(\sum_{j=1}^M u_j \varphi_j, \varphi_h) = -a_N(u_{h,k})(\varphi_h) \quad \forall \varphi_h \in V_h \\ \Leftrightarrow \quad & \sum_{j=1}^M u_j \, a_N'(u_{h,k})(\varphi_j, \varphi_h) = -a_N(u_{h,k})(\varphi_h) \quad \forall \varphi_h \in V_h \\ \Leftrightarrow \quad & \sum_{j=1}^M u_j \, a_N'(u_{h,k})(\varphi_j, \varphi_i) = -a_N(u_{h,k})(\varphi_i) \quad i = 1, \ldots, M. \quad \Leftrightarrow \quad A\delta U = B. \end{aligned}$$

Specifically,
$$A := \underbrace{(a'_N(u_{h,k})(\varphi_j,\varphi_i))_{i,j=1}^M}_{\in\mathbb{R}^{M\times M}}, \quad \delta U = \underbrace{(u_j)_{j=1}^M}_{\in\mathbb{R}^M}, \quad B := \underbrace{a_N(u_{h,k})(\varphi_i)_{i=1}^M}_{\in R^M}.$$

In explicit matrix notation:

$$A = \begin{pmatrix} (\nabla\varphi_1,\nabla\varphi_1) + \gamma(\varphi_1,\varphi_1)_B & \dots & (\nabla\varphi_M,\nabla\varphi_1) + \gamma(\varphi_M,\varphi_1)_B \\ \vdots & \ddots & \vdots \\ (\nabla\varphi_1,\nabla\varphi_M) + \gamma(\varphi_1,\varphi_M)_B & \dots & (\nabla\varphi_M,\nabla\varphi_M) + \gamma(\varphi_M,\varphi_M)_B \end{pmatrix}$$

Recall: test functions determine the row.

8.13.6 Details and discussions

We provide a bit more details in terms of heuristic discussions in this section. It is trivial to see that:

$$(\nabla\varphi_1,\nabla\varphi_1) = \int_\Omega \nabla\varphi_1\cdot\nabla\varphi_1\,dx \quad \to \quad \text{Laplacian},$$

and

$$\gamma(\varphi_1,\varphi_1) = \int_\Omega \varphi_1\cdot\varphi_1\,dx \quad \to \quad \text{(weighted) mass term.}$$

For $\gamma \gg 0$ (heavy enforcement of the constraint) and $u \ll g$ (large violations of the constraint), the linear system in the defect of Newton's method becomes ill-conditioned and Newton's method itself more nonlinear. Indeed

$$\gamma(\varphi_j,\varphi_i)_B$$

only acts in rows and columns in which the constraint is violated. We therefore have a large condition number

$$cond_2(A) \gg 1.$$

Furthermore, the condition numbers of the Laplacian and the mass term are:

$$(\nabla\varphi_j,\nabla\varphi_i) \sim \frac{1}{h}, \quad (\varphi_j,\varphi_i) \sim h$$

and for the right hand (f,φ_i), it holds:

$$(f,\varphi_i) \sim h.$$

For an asymptotic equilibrium we thus need:

$$\gamma \sim \frac{1}{h^2}.$$

In addition if we have material parameters (for instance Young's modulus or the critical energy release rate G_C if we have the phase-field system in mind), which enter into the Laplacian

$$(\alpha(x)\nabla\varphi_j,\nabla\varphi_i).$$

From this we can follow that

$$\gamma \sim \frac{\alpha(x)}{h^2}.$$

8.13.7 Implementation in DOpElib

For details on DOpElib within these lecture notes, we refer to Chapter 15. The programming code used for this example can be found on

http://www.thomaswick.org/links/Example_Obstacle_Simple_Penalization.zip

and is based on the official DOpElib example `dopelib/Examples/PDE/StatPDE/Example4/` (classical Poisson problem on the unit square).

- Some parameters are provided in `dope.prm`;
- The grid and important setups are in `main.cc`;
- The equations, g and γ are implemented in `local_pde.h`.

The number of DoFs is 4 225, the obstacle function is $g = -0.01$ and the penalization is chosen as $\gamma = \frac{\bar{\gamma}}{h^2}$ with $\bar{\gamma} = 0.1$.

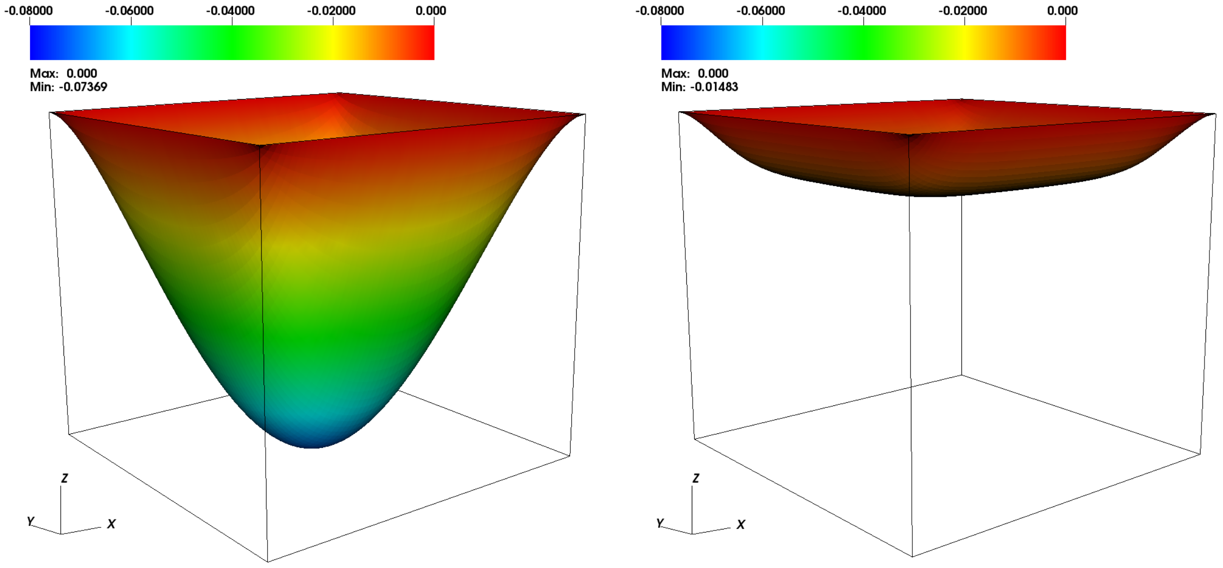

Figure 18: Left: 3d surface plot solution of the classical Poisson problem in $(0,1)^2$ and right hand side $f = -1$, as in Section 8.13.1. Right: obstacle problem with $g = -0.01$ and simple penalization.

The nonlinear solver behaves as follows:

```
Newton step: 0  Residual (abs.):   2.4414e-04
Newton step: 0  Residual (rel.):   1.0000e+00
Newton step: 1  Residual (rel.):   9.7262e-01   LineSearch {2}
Newton step: 2  Residual (rel.):   4.8631e-01   LineSearch {1}
Newton step: 3  Residual (rel.):   2.6331e-01   LineSearch {0}
Newton step: 4  Residual (rel.):   5.1674e-03   LineSearch {0}
Newton step: 5  Residual (rel.): < 1.0000e-11   LineSearch {0}
```

Evaluating the solution in the middle point yields

$$u(0.5, 0.5) = -0.0148305.$$

This shows that the constraint works, but it slightly relaxed the solution. This is a well-known shortcoming of simple penalization.

For a fair comparison, we briefly state the results for the classical Poisson problem:

```
Newton step: 0  Residual (abs.):   2.4414e-04
Newton step: 0  Residual (rel.):   1.0000e+00
Newton step: 1  Residual (rel.): < 1.0000e-11   LineSearch {0}
```

The solution value at the middle point is

$$u(0.5, 0.5) = -0.0736855.$$

8.13.8 Mesh refinement studies with $\gamma = \frac{\bar{\gamma}}{h^2}$

We now briefly investigate the behavior of the solution with respect to (uniform) mesh refinement. The numbers of DoFs are

```
4225 (above), 33282, 132098
```

We run the same code as before and obtain for the middle point and number of Newton iterations:

```
DoFs       u(0.5,0.5)          Newton iter.
-------------------------------------------
4225       -0.0148305          5
33282      -0.0112206          10
132098     -0.0103052          19
```

Thus we observe a dependence on h of both the solution and the solver performance.

8.13.9 Mesh refinement studies with $\gamma = \frac{\bar{\gamma}}{h^2_{\text{coarse}}}$

Here, we remove the dependency of h in the penalization constraint. We take the coarsest h_{coarse} and evaluate

$$\gamma = \frac{\bar{\gamma}}{h^2_{\text{coarse}}} = 204.8$$

On the finer levels, we take the same γ. Then we obtain:

```
DoFs       u(0.5,0.5)          Newton iter.
-------------------------------------------
4225       -0.0148305          5
33282      -0.01483            6
132098     -0.0148299          6
```

Here, the nonlinear solver shows constant iteration numbers. Moreover, the middle point $u(0.5, 0.5)$ converges qualitatively. However, of course, the constraint itself $g = -0.01$ is not that well approximated.

8.13.10 Further implementation exercises

In order to study the previous problem in more detail, here are some tasks:

1. Refine the mesh and observe the Newton iteration numbers and the displacement value $u(0.5, 0.5)$.
2. Change γ. What do you observe?
3. Change g. For instance, $g = -0.1$ and $g = -0.005$. What do you observe?
4. Introduce a material coefficient in the Laplacian. What do you observe for the numerical solution, the number of Newton iterations and $u(0.5, 0.5)$?
5. Implement an iteration scheme that allows to increase γ successively. Why is this useful? What do you observe for the Newton iteration numbers and $u(0.5, 0.5)$?
6. Finally, we refer to Section 10.13 in which the current excursus is extended to goal-oriented adaptive mesh refinement.

8.14 Newton-type methods applied to variational phase-field fracture

We present three Newton-type methods in the following three sections:

- In this Section 8.14, a classical monotonicity-based method is given;
- Section 8.15 presents a modified Newton method in which the Jacobian is altered;
- In Section 8.17 a Newton method is presented that combines treating the nonlinearity and the crack irreversibility constraint via a primal-dual active set strategy.

8.14.1 Formulations

Our formulation of interest reads as:

Formulation 8.33. *Given an initial phase-field $\varphi := \varphi^0$ and given either (time-dependent/time-like-dependent) non-homogeneous boundary data u_D or a pressure $p(t) \neq 0$. Compute for $n = 1, 2, 3, \ldots, N$ the incremental solution $U^n := U = \{u, \varphi\} \in \{u_D + V\} \times W$ such that*

$$A(U)(\Psi) := \overline{A}(U)(\Psi) + ([\Xi + \gamma(\varphi - \varphi^{n-1})]^+, \psi) = 0 \quad \forall \Psi \in V \times W,$$

where

$$\begin{aligned} \overline{A}(U)(\Psi) = &\Big(\big((1-\kappa)\varphi^2 + \kappa\big)\, \sigma^+(u), e(w)\Big) + (\sigma^-(u), e(w)) \\ &+ (1-\kappa)(\varphi\, \sigma^+(u) : e(u), \psi) \\ &+ G_c\Big(-\frac{1}{\varepsilon}(1-\varphi, \psi) + \varepsilon(\nabla\varphi, \nabla\psi)\Big). \end{aligned} \tag{24}$$

Remark 8.34 (Imposing the inequality constraint)**.** *In order to determine Ξ, we design an adaptive augmented Lagrangian formulation (the outer loop) in which we iterate according to the algorithm presented in [175] (based on [169]). Therein, we have also shown that the adaptive choice of the inner tolerance (namely for Newton's method) can significantly reduce further computational costs.*

Let us discuss spatial discretization, which is based on a Galerkin finite element scheme, introducing H^1 conforming discrete spaces $V_h \subset V$ and $W_h \subset W$ consisting of bilinear functions Q_1^c on quadrilaterals or trilinear functions on hexahedra (see e.g., [45]). The discretization parameter is denoted by h. The discretized version of Formulation 8.33 reads:

Formulation 8.35. *Given an initial phase-field $\varphi_h := \varphi_h^0$ and given either (time-dependent/time-like-dependent) non-homogeneous boundary data u_D^h or a pressure $p(t) \neq 0$. Compute for $n = 1, 2, 3, \ldots, N$ the incremental solution $U^n := U_h = \{u_h, \varphi_h\} \in \{u_D^h + V_h\} \times W_h$ such that*

$$A(U_h)(\Psi_h) := \overline{A}(U_h)(\Psi_h) + ([\Xi_h + \gamma(\varphi_h - \varphi_h^{n-1})]^+, \psi_h) = 0 \quad \forall \Psi_h \in V_h \times W_h.$$

8.14.2 A monotonicity-based Newton algorithm

First, we recapitulate a monotonicity-based Newton algorithm. Globalization[1] may be achieved with a damping strategy based on a backtracking line search algorithm. After having presented the algorithm, we explain the steps to change to a modified Newton scheme with Jacobian modification.

To measure the residuals and monitoring functions, we use the discrete norm $\|\cdot\| := \|\cdot\|_{l^2}$. At a given time instance t_n, we shall find the time step solution U^n using:

[1] The terminology 'globalization' is adopted from numerical optimization (e.g., [133]) or, in general, Newton methods (e.g., [51]) and means that the convergence radius of Newton's method is extended by, for example, line search or trust region methods.

Algorithm 8.36 (Residual-based Newton's method)**.** *In this type of methods, the main criterion is a decrease of the residual in each step. Choose an initial Newton guess U^0. For the iteration steps $k = 0, 1, 2, 3, \ldots$:*

1. *Find $\delta U^k := \{\delta u, \delta\varphi\} \in V \times W$ such that*

$$A'(U^k)(\delta U^k, \Psi) = -A(U^k)(\Psi) \quad \forall \Psi \in V \times W, \tag{25}$$
$$U^{k+1} = U^k + \lambda_k \delta U^k, \tag{26}$$

   for $\lambda_k = 1$.

2. *The criterion for convergence is the contraction of the residuals:*

$$\|A(U^{k+1})(\Psi)\| < \|A(U^k)(\Psi)\|. \tag{27}$$

3. *If (27) is violated, re-compute in (26) U^{k+1} by choosing $\lambda_k^l = 0.5$, and compute for $l = 1, ..., l_M$ (e.g. $l_M = 5$) a new solution*

$$U^{k+1} = U^k + \lambda_k^l \delta U^k$$

   until (27) is fulfilled for a $l^* < l_M$ or l_M is reached. In the latter case, no convergence is obtained and the program aborts.

4. *In case of $l^* < l_M$ we check next the stopping criterion:*

$$\|A(U^{k+1})(\Psi)\| \leq TOL_N.$$

   If this is criterion is fulfilled, set $U^n := U^{k+1}$. Else, we increment $k \to k+1$ and go to Step 1.

Remark 8.37 (Changes in the modified Newton scheme)**.** *In the modified Newton scheme that we present below in more detail, we shall work with a modified Jacobian $A'_\omega(U^k)(\delta U^k, \Psi)$ and no line search, i.e., $\lambda_k = 1$ for all k. Furthermore, Step 2 (inequality 27) is omitted. In place of Step 3, we compute heuristically a control parameter ω, which is derived in Section 8.15.*

Remark 8.38 (On using quasi-Newton steps)**.** *Usually, when the Newton reduction rate*

$$\theta_k = \frac{\|A(U^{k+1})(\Psi)\|}{\|A(U^k)(\Psi)\|},$$

was sufficiently good, e.g., $\theta_k \leq \theta_{max} < 1$ (where e.g. $\theta_{max} \approx 0.1$), a common strategy is to work with the 'old' Jacobian matrix, but with a new right hand side. This procedure is well established in the literature (see e.g., [51]) and works usually. In phase-field fracture, we found the contrary that the matrix $A'(U^k)(\delta U^k, \Psi)$ should be assembled at each Newton iteration step k such that it fits as well as possible to the corresponding right hand side $A(U^k)(\Psi)$. This is reasonable from a theoretical point of view since the matrix is indefinite and the problem non-convex. Thus smallest perturbations between matrix and right hand side, may lead to large mismatches, which result in a blow-up of the residual and therefore divergence of Newton's method causing the iteration to stop. In Section 9, we illustrate our experiences with the help of one example.

8.14.3 A successful basic Newton implementation with line search in C++

A successful example of a basic implementation of Newton's method is found in [171].

```
newton_iteration ()
{
  const double lower_bound_newton_residual = 1.0e-10;
  const unsigned int max_no_newton_steps  = 20;

  // Decision whether the system matrix should be build at each Newton step
  const double nonlinear_theta = 0.1;

  // Line search parameters
  unsigned int line_search_step;
  const unsigned int  max_no_line_search_steps = 10;
  const double line_search_damping = 0.6;
  double new_newton_residual;

  // Application of nonhomogeneous Dirichlet boundary conditions to the variational equations:
  set_nonhomo_Dirichlet_bc ();

  // Evaluate the right hand side residual
  assemble_system_rhs();

  double newton_residual = system_rhs.linfty_norm();
  double old_newton_residual = newton_residual;
  unsigned int newton_step = 1;

  if (newton_residual < lower_bound_newton_residual)
      std::cout << '\t' << std::scientific << newton_residuum << std::endl;

  while (newton_residual > lower_bound_newton_residual &&
         newton_step < max_no_newton_steps)
    {
      old_newton_residual = newton_residual;

      assemble_system_rhs();
      newton_residual = system_rhs.linfty_norm();

      if (newton_residuum < lower_bound_newton_residuum)
        {
          std::cout << '\t'
                    << std::scientific
                    << newton_residual << std::endl;
          break;
        }

      // Simplified Newton steps
      if (newton_residual/old_newton_residual > nonlinear_theta)
        assemble_system_matrix ();

      // Solve linear equation system Ax = b
      solve ();

      line_search_step = 0;
      for ( ; line_search_step < max_no_line_search_steps; ++line_search_step)
         {
          solution += newton_update;

          assemble_system_rhs ();
          new_newton_residual = system_rhs.linfty_norm();

          if (new_newton_residual < newton_residual)
            break;
          else
            solution -= newton_update;

          newton_update *= line_search_damping;
         }

      // Output to the terminal for the user
      std::cout << std::setprecision(5) <<newton_step << '\t' << std::scientific << newton_residual << '\t'
                << std::scientific << newton_residual/old_newton_residual  <<'\t' ;
      if (newton_residual/old_newton_residual > nonlinear_theta)
        std::cout << "r" << '\t' ;
      else
        std::cout << " " << '\t' ;
      std::cout << line_search_step  << '\t' << std::endl;

      // Goto next newton iteration, increment j->j+1
      newton_step++;
    }
}
```

8.14.4 The Jacobian matrix

To apply Newton's method for solving $A(U_h)(\Psi_h) = 0$, we need to compute the derivative of $A(U_h)(\Psi_h)$. We construct the Jacobian[2] by evaluating the directional derivative

$$A'(U)(\delta U, \Psi) := \lim_{s \to 0} \frac{A(U + s\delta U)(\Psi) - A(U)(\Psi)}{s}$$

with $\delta U := \{\delta u, \delta\varphi\} \in V \times W$, which represents later the Newton update. In detail, the Jacobian is given by:

$$\begin{aligned} A'(U)(\delta U, \Psi) = {} & \Big(2\delta\varphi(1-\kappa)\varphi\sigma^+(u) + \big((1-\kappa)\varphi^2 + \kappa\big)\ \sigma^+(\delta u), e(w)\Big) + (\sigma^-(\delta u), e(w)) + 2\,(\delta\varphi\,\varphi p, \operatorname{div}\, w) \\ & + (1-\kappa)\big(\delta\varphi\sigma^+(u) : e(u) + 2\varphi\ \sigma^+(\delta u) : e(u), \psi\big) + 2p(\delta\varphi\nabla\cdot u + \varphi\ \nabla\cdot\delta u, \psi) \\ & + G_c\Big(\frac{1}{\varepsilon}(\delta\varphi, \psi) + \varepsilon(\nabla\delta\varphi, \nabla\psi)\Big) \\ & + \gamma(\delta\varphi, \psi)_{A(\varphi)} \quad \forall \Psi := \{w, \psi\} \in V \times W, \end{aligned} \tag{28}$$

where

$$\mathcal{A}(\varphi) = \{x = (x_1, x_2, x_3) \in B \mid \Xi + \gamma\big(\varphi(x) - \varphi(x)^{n-1}\big) > 0\}.$$

In $\sigma^+(\delta u)$ and $\sigma^-(\delta u)$ we employ the derivative of e^+, which is given by

$$e^+(\delta u) = P(\delta u)\Lambda^+ P^T + P\Lambda^+(\delta u)P^T + P\Lambda^+ P^T(\delta u).$$

Exercise 18. *Understand and derive again by yourself the directional derivative* (28).

Remark 8.39 (on the critical term)**.** *We observe that the critical term in the matrix is contained in*

$$\Big(2\delta\varphi(1-\kappa)\varphi\sigma^+(u) + \big((1-\kappa)\varphi^2 + \kappa\big)\ \sigma^+(\delta u), e(w)\Big). \tag{29}$$

Consulting our computational experiences from [82, 175] where we designed a very efficient and robust method by neglecting the cross-term block, we conjecture that the most critical term is the off-diagonal contribution

$$2\delta\varphi(1-\kappa)\varphi\sigma^+(u).$$

Similar observations have been made in related studies on yield stress fluids (see [111]), where usually the derivative of the nonlinear factor causes most difficulties in the solution process. Indeed in Section 4 of [177], a detailed simplified analysis is provided and shows that the cross-term significantly determines the properties of the Jacobian matrix.

[2] In this section, we omit the index h to simplify the notation.

8.14.5 Block structure of the Jacobian, solution vector, and right hand side

Before we are able to design another Newton method, we want to understand the structure of the linear system (25) to be solved at each Newton iteration. For the spatial discretization, we use the previously introduced spaces $V_h \times W_h$ with vector-valued basis

$$\{\psi_i \,|i = 1, \dots, N\},$$

where the basis functions are primitive (they are only non-zero in one component), so we can separate them into displacement and phase-field basis functions and sort them accordingly:

$$\begin{aligned}\psi_i &= \begin{pmatrix}\chi_i^u\\ 0\end{pmatrix}, \text{ for } i = 1, \dots, N_u,\\ \psi_{(N_u+i)} &= \begin{pmatrix}0\\ \chi_i^\varphi\end{pmatrix}, \text{ for } i = 1, \dots, N_\varphi,\end{aligned}$$

where $N_u + N_\varphi = N$. This is now used to transform (25) into a system of the form

$$M\delta U = F, \tag{30}$$

where M is a block matrix (the Jacobian) and F the right hand side containing of the residuals. The unknown solution vector is δU. The block structures are

$$M = \begin{pmatrix}M^{uu} & M^{u\varphi}\\ M^{\varphi u} & M^{\varphi\varphi}\end{pmatrix}, \qquad F = \begin{pmatrix}F^u\\ F^\varphi\end{pmatrix}, \qquad \delta U = \begin{pmatrix}\delta U^u\\ \delta U^\varphi\end{pmatrix},$$

with entries coming from (28):

$$\begin{aligned}M_{i,j}^{uu} &= \Big(\big((1-\kappa)\varphi^2 + \kappa\big)\ \sigma^+(\chi_j^u), e(\chi_i^u)\Big) + (\sigma^-(\chi_j^u), e(\chi_i^u)),\\ M_{i,j}^{u\varphi} &= \Big(2\chi_j^\varphi(1-\kappa)\varphi\sigma^+(u), e(\chi_i^u)\Big) + 2\,(\chi_j^\varphi\ \varphi p, \text{div}\ \chi_i^u),\\ M_{i,j}^{\varphi u} &= 2(1-\kappa)(\varphi\ \sigma^+(\chi_j^u) : e(u), \chi_i^\varphi) + 2p(\varphi\ \text{div}(\chi_j^u), \chi_i^\varphi),\\ M_{i,j}^{\varphi\varphi} &= (1-\kappa)(\sigma^+(u) : e(u)\chi_j^\varphi, \chi_i^\varphi) + 2p(\text{div}(u)\chi_j^\varphi, \chi_i^\varphi)\\ &\quad + G_c\Big(\frac{1}{\varepsilon}(\chi_j^\varphi, \chi_i^\varphi) + \varepsilon(\nabla\chi_j^\varphi, \nabla\chi_i^\varphi)\Big) + \gamma(\chi_j^\varphi, \chi_i^\varphi)_{A(\varphi)}.\end{aligned}$$

The right hand side consists of the corresponding residuals (see Formulation 8.33 and therein (24)). In particular, we have

$$\begin{aligned}F_i^u &= -A(U^k)(\chi_i^u)\\ &= \Big(\big((1-\kappa){\varphi_k}^2 + \kappa\big)\ \sigma^+(u_k), e(\chi_i^u)\Big) + (\sigma^-(u_k), e(\chi_i^u)) + (\varphi_k^2 p, \text{div}\ \chi_i^u),\\ F_i^\varphi &= -A(U^k)(\chi_i^\varphi) = (1-\kappa)(\varphi_k\ \sigma^+(u_k) : e(u_k), \chi_i^\varphi) + 2(\varphi_k\ p\ \text{div}\ u_k, \chi_i^\varphi)\\ &\quad + G_c\Big(-\frac{1}{\varepsilon}(1-\varphi_k, \chi_i^\varphi) + \varepsilon(\nabla\varphi_k, \nabla\chi_i^\varphi)\Big) + ([\Xi_h + \gamma(\varphi_k - \varphi_k^{n-1}),]^+, \chi_i^\varphi).\end{aligned}$$

In the matrix, the degrees of freedom that belong to Dirichlet conditions (here only the displacements since we assume Neumann conditions for the phase-field variable) are strongly enforced by replacing the corresponding rows and columns as usually done in a finite element code.

8.15 A modified Newton method with Jacobian modification

We closely follow [177]. We suggest in the following a method that is inspired by two sources. First, a successfully-used algorithm for nonlinear flow problems is a modified Newton method with Jacobian modification [90, 111, 113]. This raises the question of how such a modification can be achieved in phase-field fracture. A hint can be found in [82] in which one block in the Jacobian was zero due to extrapolation in the phase-field variable, yielding an extremely robust and efficient method. The idea is to introduce a control parameter ω for this specific block in a fully monolithic setting.

Rather than employing line search in Step 3 in Algorithm 8.36, we introduce a control parameter $\omega \in [0,1]$ inside the Jacobian, which decides whether a full Newton system ($\omega = 1$), a Newton-like system with $0 < \omega < 1$ or even $\omega = 0$ is solved. The choice of this parameter is heuristic, but the key idea is quite simple. Based on several studies that have been performed for nonlinear flow [90, 111, 113], we further develop these concepts in the following.

As shown in Section 8.14.5, formally, the Jacobian reads at each Newton step k:

$$M = \begin{pmatrix} M^{uu} & M^{u\varphi} \\ M^{\varphi u} & M^{\varphi\varphi} \end{pmatrix}.$$

The critical block is $M^{u\varphi}$, particularly

$$(2\chi_j^{\varphi}(1-\kappa)\varphi\sigma^+(u), e(\chi_i^u)) \tag{31}$$

as it was already identified in [82] (see also the Remarks 8.40 below and 8.39 above). Thus the goal is to design a procedure in which this block is dynamically activated or disabled during a Newton iteration. Incorporating ω brings us to

$$M = \begin{pmatrix} M^{uu} & \omega M^{u\varphi} \\ M^{\varphi u} & M^{\varphi\varphi} \end{pmatrix} = \begin{pmatrix} M^{uu} & 0 \\ M^{\varphi u} & M^{\varphi\varphi} \end{pmatrix} + \omega \begin{pmatrix} 0 & M^{u\varphi} \\ 0 & 0 \end{pmatrix}. \tag{32}$$

Remark 8.40 (Extrapolated scheme). *In the extrapolated scheme, we replace φ^2 by a linear-in-time extrapolation $\tilde{\varphi}^2$ in the first line of the residual* (24). *When computing the Jacobian, the block $M_{i,j}^{u\varphi}$ is zero after differentiation with respect to φ. Therefore, the matrix M has always a triangular block structure:*

$$M = \begin{pmatrix} M^{uu} & 0 \\ M^{\varphi u} & M^{\varphi\varphi} \end{pmatrix}. \tag{33}$$

This pattern greatly facilitates the linear solution, in particular, the design of preconditioners when using an iterative technique, such as for instance GMRES. Evidence is shown in several studies for 2D and 3D problems where we could extend relatively easily the idea presented in [82] to parallel computations in 3D [106].

Remark 8.41. *Since the blocks $M^{\varphi u}$ and $M^{u\varphi}$ are identical since the matrix M is symmetric by construction, one may try to build a symmetric approximation by*

$$M = \begin{pmatrix} M^{uu} & \omega M^{u\varphi} \\ \omega M^{\varphi u} & M^{\varphi\varphi} \end{pmatrix} = \begin{pmatrix} M^{uu} & 0 \\ 0 & M^{\varphi\varphi} \end{pmatrix} + \omega \begin{pmatrix} 0 & M^{u\varphi} \\ M^{\varphi u} & 0 \end{pmatrix}. \tag{34}$$

We carried out some further numerical tests exploiting this idea, but found inferior performance of the Newton solver. From a numerical standpoint, this is clear because removing more terms in the matrix weakens the performance of the Newton scheme since the Jacobian and the residual fit less together. For this reason, we did not further pursue this idea in our current work and we worked rather with the decomposition in Equation (32).

8.15.1 Computing the control parameter ω

The choice of ω is done in a dynamic way depending on the previous two Newton residuals. Thus at each Newton step k, the parameter $\omega := \omega_k$ is updated if applicable.

We define the residual and reciprocal residual reductions, respectively:

$$Q_{k+1} = \frac{\|A(U^{k+1})(\Psi)\|}{\|A(U^k)(\Psi)\|}, \quad Q_{k+1}^{rec} = \frac{\|A(U^k)(\Psi)\|}{\|A(U^{k+1})(\Psi)\|}. \tag{35}$$

If $Q_{k+1} < 1$, the new residual is smaller and we classify this step as a 'good' step. Moreover, if $Q_{k+1} \to 0$, the better the current step. On the other hand, if $Q_{k+1} \geq 1$, the new residual is larger than the old one and we have the situation in which a monotonicity-based Newton method would fail and, for example, an error-oriented version may perform better [51].

We summarize the key ideas and the construction of the control parameter ω in the following:

Definition 8.42 (Computing ω_{k+1}). *At the Newton step k, let $0 \leq \omega_k \leq 1$ be given and let $S \in \mathbb{R}_+ \cup \{0\}$. We define*

$$\omega := \omega_{k+1} = S\omega_k. \tag{36}$$

Proposition 8.43 (Motivation of S). *The scaling parameter S is motivated as follows:*

1. *S must yield $\omega_{k+1} \in [0,1]$.*

2. *$S \gg 1$ should yield $\omega_{k+1} \to 1$ (full Newton).*

3. *$S \to 0$ should yield $\omega_{k+1} \to 0$ (Newton-like/fixed-point like scheme).*

Proposition 8.44 (A specific realization of S). *Given Proposition 8.43, the scaling parameter S can be realized, using the residuals Q_{k+1} and Q_{k+1}^{rec} defined in* (35)*, as follows:*

1. *For $Q_{k+1} \to 0$ (and $Q_{k+1}^{rec} \to \infty$) the matrix M defined in* (32) *has good properties and we can work with a full Newton step, i.e., $S \gg 1$ yielding $\omega_{k+1} = 1$.*

2. *On the other hand, for $Q_{k+1}^{rec} \to 0$ (and $Q_{k+1} \to \infty$), the matrix M becomes ill-conditioned, and $S \to 0$ yielding $\omega_{k+1} \ll 1$ should be employed.*

3. *These observations yield the following possible realization of S:*

$$S := \Big(\frac{a}{\exp\left(Q_{k+1}^{rec}\right)} + \frac{b}{\exp\left(Q_{k+1}\right)}\Big). \tag{37}$$

4. *The control parameter a is related to a fixed-point step with small ω_{k+1} and thus $a < 1$ should be chosen.*

5. *The control parameter b is related to a full Newton step with $\omega_{k+1} = 1$ and thus $b \geq 1$ should be chosen.*

6. *Both control parameters will be further explained and specified in Section 8.15.3.*

Corollary 8.45 (Further properties of S). *The scaling parameter S has the following properties:*

1. *S is bounded from below by zero: since $a, b, Q_{k+1}, Q_{k+1}^{rec} \geq 0$, we have $S \geq 0$ yielding $\omega_{k+1} \geq 0$.*

2. *S is not bounded from above. Consequently, it may easily happen that $\omega_{k+1} > 1$ in* (36)*. Therefore we use a simple projection:*

$$\textit{Set} \quad \omega_{k+1} := 1 \quad \textit{if} \quad \omega_{k+1} > 1.$$

8.15.2 The modified Newton algorithm

We define

$$\begin{aligned} A'_\omega(U)(\delta U, \Psi) = &\Big(\omega 2\delta\varphi(1-\kappa)\varphi\sigma^+(u) + \big((1-\kappa)\varphi^2 + \kappa\big)\ \sigma^+(\delta u), e(w)\Big) + (\sigma^-(\delta u), e(w)) + 2\,(\omega\delta\varphi\,\varphi p, \operatorname{div}\, w) \\ &+ (1-\kappa)\big(\delta\varphi\sigma^+(u) : e(u) + 2\varphi\ \sigma^+(\delta u) : e(u), \psi\big) + 2p(\delta\varphi\nabla\cdot u + \varphi\ \nabla\cdot\delta u, \psi) \\ &+ G_c\Big(\frac{1}{\varepsilon}(\delta\varphi, \psi) + \varepsilon(\nabla\delta\varphi, \nabla\psi)\Big) \\ &+ \gamma(\delta\varphi, \psi)_{A(\varphi)} \quad \forall\Psi := \{w, \psi\} \in V \times W, \end{aligned} \tag{38}$$

which is (28) except that the terms with $\delta\varphi$ multiplied by the w test function are scaled with ω and in particular, the very first term.

At a given time instance t_n, we shall find the time step solution U^n using:

Algorithm 8.46 (Modified Newton's method with Jacobian modification)**.** *Choose an initial Newton guess* U^0 *and an initial guess for the control parameter, i.e.,* $\omega_0 = 1$*. For the iteration steps* $k = 0, 1, 2, 3, \ldots$*:*

1. *Find* $\delta U^k := \{\delta u, \delta\varphi\} \in V \times W$ *such that*

$$A'_\omega(U^k)(\delta U^k, \Psi) = -A(U^k)(\Psi) \quad \forall \Psi \in V \times W, \tag{39}$$

$$U^{k+1} = U^k + \delta U^k. \tag{40}$$

2. *Compute:*

$$\omega := \omega_{k+1} = S\omega_k, \tag{41}$$

with S *determined by* (37)*.*

3. *Check*

$$\|A(U^{k+1})(\Psi)\| \leq TOL_N.$$

If this criterion is fulfilled, set $U^n := U^{k+1}$*. Else, we increment* $k \to k+1$ *and go to Step 1.*

Remark 8.47. *In this algorithm, we do not have any convergence monitor and it can happen that Newton's method diverges. Thus we also check in Step 3 whether*

$$\|A(U^{k+1})(\Psi)\| < TOL_N^{up}, \quad TOL_N^{up} = 10^{12},$$

otherwise we stop the algorithm because of divergence. In Section 9, we see that for backtracking line-search such a behavior is indeed detected, but in which the modified Newton method (without convergence monitor) yields excellent performance. We notice that TOL_N^{up} *seems very high, but there are examples in Section 9 where the residual goes up to* 10^7 *but nonetheless Newton's method will finally still converge.*

8.15.3 On the choice of a and b

The choices of a and b are heuristic. As outlined in Proposition 8.44, the parameter a controls the influence of block $M^{u\varphi}$. The parameter b controls the rate to go back to full Newton steps in case sufficient performance of the solver is detected. Therefore, we propose the following bounds:

$$0 \leq a < 1 \quad \text{and} \quad 1 \leq b < \infty.$$

Let us discuss the idea in more detail. If $Q_{k+1} \ll 1$ we had a good reduction and we can use a higher ω_{k+1} in the next step. Formula (37) yields

$$\lim_{Q_{k+1}^{rec} \to \infty} \lim_{Q_{k+1} \to 0} S \to b \quad \Rightarrow \quad \omega_{k+1} = b\omega_k \quad \Rightarrow \quad \omega_{k+1} \geq \omega_k.$$

On the other hand if $Q_{k+1} > 1$ or even $Q_{k+1} \gg 1$ (thus $Q_{k+1}^{rec} \to 0$) we want to eliminate the irregular terms in the Jacobian matrix and rather work with a Newton-like method in which the Jacobian is approximated by minimizing the influence of the term (31). Here:

$$\lim_{Q_{k+1}^{rec} \to 0} \lim_{Q_{k+1} \to \infty} S \to a \quad \Rightarrow \quad \omega_{k+1} = a\omega_k \quad \Rightarrow \quad \omega_{k+1} < \omega_k.$$

Since due to the construction, we cannot ensure a priori that S is bounded from above, the requirement $0 \leq \omega_{k+1} \leq 1$ may be violated. In this case, a projection is used (see Corollary 8.45).

Possible choices of a and b:

- Choice 1: $a = 0.001$ and $b = 10$ drastically tries to remove the entire block $M^{u\varphi}$ and moderately goes back to full Newton;
- Choice 2: $a = 0.1$ and $b = 2$ tries moderately to remove the influence of block $M^{u\varphi}$ and moderately goes back to full Newton;
- Choice 3: $a = 0$ and $b = 0$ resulting in $S = 0$ from which we obtain $\omega_k = 0$ for all k and thus never work with block $M^{u\varphi}$.

Obviously, the smaller $a \ll 1$ is chosen, the faster we obtain a Newton-like (fixed-point) scheme. Secondly, the larger we choose $b \geq 1$, the faster we go back to a full Newton scheme.

Remark 8.48 (Numerical results)**.** *For more options of a and b and their consequences in numerical examples, we refer to [177].*

8.16 Excursus: Modified Newton for functional minimization in $\mathbb{R}^2$

In this excursus we study some basic properties of the phase-field equations and the behavior of Newton method in terms of a simplified model problem. Specifically, we consider the minimization of a functional in $\mathbb{R}^2$. The calculations are motivated by [177] (we strongly notice: **they are not the same** since we modify the functional with an additional regularization term). Overall, this example originated from a student project at École Polytechnique (MAP 502, STEEM, winter term 2016/2017) [92].

8.16.1 Problem statement

Let $F : \mathbb{R}^2 \to \mathbb{R}$ be given by:

$$F(x,y) = (\kappa + y^2)x^2 + \varepsilon y^2, \quad \kappa, \varepsilon > 0, \quad (x,y) \in \mathbb{R}^2.$$

The function is visualized in Figure 19. We easily recognize the regularization parameters κ and ε from our original phase-field problem. Moreover, we relate x to u (displacements) and y to φ (phase-field).

Formulation 8.49. *Let $F(x,y)$ be given as before. Solve:*

$$\min_{(x,y)\in\mathbb{R}^2} F(x,y).$$

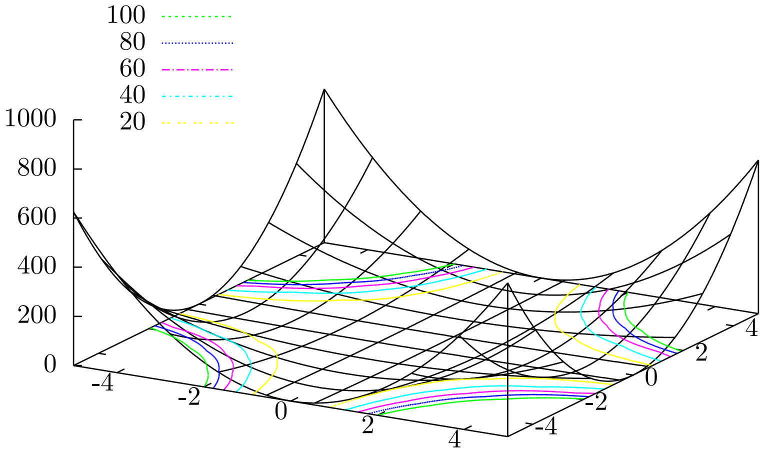

Figure 19: Visualization of $F(x,y)$ and its contour lines. Specifically, we easily see that the minimum is not unique. We also observe that by fixing one variable, the problem becomes strictly convex in the other unknown.

Remark 8.50 (Link to phase-field fracture)**.** *In phase-field fracture, the critical part of the underlying energy functional (see e.g., Formulation* (5.40)*) for $\mathbb{C}|e(u)|^2 = |\nabla u|^2$ in [27, 64], is:*

$$E_{crit}(u,\varphi) = \frac{1}{2}\int_B \underbrace{(\kappa + (1-\kappa)\varphi^2)}_{\sim(\kappa+y^2)} \underbrace{\mathbb{C}|e(u)|^2}_{\sim x^2} \, dx + \underbrace{\int_B \varepsilon |\nabla\varphi|^2 \, dx}_{\sim\varepsilon y^2} \tag{42}$$

where $\mathbb{C}e(u) = \sigma(u) := 2\mu e(u) + \lambda tr(e(u))I$.

Remark 8.51. *The function $F(x,y)$ represents the main term of the energy formulation of the fracture problem in a simplified fashion. Here the variable x represents the phase-field φ and y represents the displacements u (i.e., the stresses $\sigma(u)$). Clearly, we see that the minimal value of $F(x,y)$ is zero, however the solution (x,y) is not unique: any pair $(x,0)$ with $x \in \mathbb{R}$ yields $F(x,0) = 0$. The non-uniqueness is due to the non-convexity of $F(x,y)$.*

8.16.2 Derivatives

To solve the above problem numerically, we use the first order optimality condition to compute the stationary points:

Formulation 8.52. *The derivative of $F(x,y)$ is computed as*

$$F'(x,y) = (2(\kappa + y^2)x, 2yx^2 + 2y\varepsilon)^T.$$

Solve

$$F'(x,y) = 0.$$

In order to solve $F'(x,y) = 0$ to obtain a solution pair (x,y), we apply Newton's method. To this end, we need the second-order derivative, the so-called Hessian:

$$H_f := H_f(x,y) = F''(x,y) = \begin{pmatrix} 2(\kappa + y^2) & 4yx \\ 4yx & 2(\varepsilon + x^2) \end{pmatrix}.$$

Proposition 8.53. *The determinant of the Hessian H_f is given by:*

$$\det(H_f) = -12x^2y^2 + 4\kappa x^2 + 4\varepsilon y^2 + 4\kappa\varepsilon.$$

A graphical sketch is provided in Figure 20. Furthermore:

- *For κ and ε sufficiently small and x and y sufficiently large, it holds*

$$\det(H_f) < 0.$$

- *For $x = y = 0$ and $\kappa, \varepsilon > 0$, we have*

$$\det(H_f) > 0.$$

Since usually $\kappa \ll 1$ and $\varepsilon \ll 1$, a positive determinant only holds true in a small strip with $x \approx 0$ or $y \approx 0$.

Corollary 8.54. *From the previous result, we have in particular (see e.g., [98]):*

- *For $\det(H_f) < 0$, the Hessian H_f is indefinite.*
- *For $\det(H_f) > 0$, the Hessian H_f is positive definite because the first entry is strictly positive, i.e., $2(\kappa + y^2) > 0$ for $\kappa > 0$.*

As it is well-known, the sign of the determinant has important consequences for the solution of linear equation systems. In this context, we envisage to apply Newton's method in which a linear equation system using the Hessian must be solved.

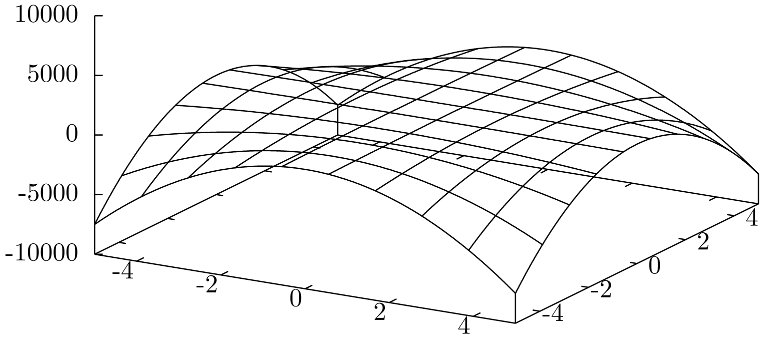

Figure 20: Visualization of $\det(H_f)$.

8.16.3 Newton's method

Let (x_0, y_0), be given and find (x_{k+1}, y_{k+1}) for $k = 0, 1, 2, 3, \ldots$,

$$H_f(x_k, y_k)(\delta x, \delta y)^T = -F'(x_k, y_k),$$
$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} \delta x \\ \delta y \end{pmatrix} + \begin{pmatrix} x_k \\ y_k \end{pmatrix}.$$

This Newton method might produce non-descending steps.

8.16.4 Numerical results and discussion

In the following, we carry out some computations with interesting results. The programming code is based on octave [55] and is a further extension of [92] and [177]. The online version is available here:

http://www.thomaswick.org/links/newton_octave_PFF_lecture_notes_Apr_3_2019.m

We take $\kappa = 0.001$ and $\varepsilon = 0.01$ and as the initial Newton guess, we take $(x_0, y_0) = (-5, 4)$. The only convergence monitoring criterion is whether the residual norm is smaller than a given tolerance, i.e.,

$$|F'(x_{k+1}, y_{k+1})| < TOL, \quad TOL = 10^{-8}.$$

8.16.4.1 Classical Newton's method without line search and no further modifications

In the first run, we obtain:

```
It x             y              dF(x)         dF(y)          ||(dF(x,y)||   F(x,y)
--------------------------------------------------------------------------------------
0 -5.000000e+00 4.000000e+00 | -1.600100e+02 2.000800e+02 | 2.561937e+02 | 4.099072e+05
1 -3.332514e+00 2.666911e+00 | -4.741111e+01 5.928889e+01 | 7.591433e+01 | 1.600987e+04
2 -2.220446e+00 1.778308e+00 | -1.404823e+01 1.757104e+01 | 2.249653e+01 | 6.260806e+02
3 -1.478451e+00 1.186091e+00 | -4.162760e+00 5.208876e+00 | 6.667905e+00 | 2.455273e+01
4 -9.828573e-01 7.915607e-01 | -1.233620e+00 1.545140e+00 | 1.977189e+00 | 9.693687e-01
5 -6.510517e-01 5.289746e-01 | -3.656490e-01 4.590107e-01 | 5.868475e-01 | 3.902773e-02
6 -4.276795e-01 3.546105e-01 | -1.084155e-01 1.368156e-01 | 1.745636e-01 | 1.721328e-03
7 -2.753187e-01 2.395625e-01 | -3.215183e-02 4.110912e-02 | 5.218908e-02 | 1.270542e-04
8 -1.678144e-01 1.652803e-01 | -9.504201e-03 1.261475e-02 | 1.579436e-02 | 3.068849e-05
9 -8.340052e-02 1.227063e-01 | -2.678298e-03 4.161130e-03 | 4.948564e-03 | 1.523658e-05
10 5.213122e-02 1.636034e-01 | 2.894959e-03 4.161307e-03 | 5.069246e-03 | 2.707421e-05
11 3.269704e-01 -3.686307e-01 | 8.951703e-02 -8.619295e-02 | 1.242680e-01 | 1.304937e-03
12 2.045163e-01 -2.524951e-01 | 2.648641e-02 -2.617208e-02 | 3.723584e-02 | 1.154943e-04
13 1.135300e-01 -1.813140e-01 | 7.691604e-03 -8.300216e-03 | 1.131611e-02 | 3.541126e-05
14 2.345261e-02 -1.620167e-01 | 1.278140e-03 -3.418561e-03 | 3.649685e-03 | 2.630864e-05
15 5.064687e-02 1.958866e-02 | 1.401617e-04 4.922671e-04 | 5.118323e-04 | 3.839196e-07
16 2.148634e-02 4.604862e-03 | 4.388390e-05 9.634902e-05 | 1.058722e-04 | 2.122405e-08
17 8.165437e-04 3.909709e-04 | 1.633337e-06 7.819939e-06 | 7.988694e-06 | 1.528849e-10
18 2.495594e-07 5.211606e-08 | 4.991189e-10 1.042321e-09 | 1.155661e-09 | 2.718575e-18
```

8.16.4.2 Modified Newton's method

In our second test, we work with Algorithm 8.46. Introducing ω, the Hessian matrix reads:

$$\begin{pmatrix} 2(\kappa + y^2) & \omega 4yx \\ 4yx & 2(\varepsilon + x^2) \end{pmatrix}.$$

Choosing $a = 0.01$ and $b = 10$, we obtain the result in 14, thus less iterations:

```
It x            y               dF(x)         dF(y)           ||(dF(x,y)||   F(x,y)
-------------------------------------------------------------------------------------
0 -5.000000e+00 4.000000e+00 | -1.600100e+02 2.000800e+02 | 2.561937e+02 | 4.099072e+05
1 -3.332514e+00 2.666911e+00 | -4.741111e+01 5.928889e+01 | 7.591433e+01 | 1.600987e+04
2 -2.220446e+00 1.778308e+00 | -1.404823e+01 1.757104e+01 | 2.249653e+01 | 6.260806e+02
3 -1.478451e+00 1.186091e+00 | -4.162760e+00 5.208876e+00 | 6.667905e+00 | 2.455273e+01
4 -9.828573e-01 7.915607e-01 | -1.233620e+00 1.545140e+00 | 1.977189e+00 | 9.693687e-01
5 -6.510517e-01 5.289746e-01 | -3.656490e-01 4.590107e-01 | 5.868475e-01 | 3.902773e-02
6 -4.276795e-01 3.546105e-01 | -1.084155e-01 1.368156e-01 | 1.745636e-01 | 1.721328e-03
7 -2.753187e-01 2.395625e-01 | -3.215183e-02 4.110912e-02 | 5.218908e-02 | 1.270542e-04
8 -1.678144e-01 1.652803e-01 | -9.504201e-03 1.261475e-02 | 1.579436e-02 | 3.068849e-05
9 -8.340052e-02 1.227063e-01 | -2.678298e-03 4.161130e-03 | 4.948564e-03 | 1.523658e-05
10 5.213122e-02 1.636034e-01 | 2.894959e-03 4.161307e-03 | 5.069246e-03 | 2.707421e-05
11 3.269704e-01 -3.686307e-01 | 8.951703e-02 -8.619295e-02 | 1.242680e-01 | 1.304937e-03
12 -5.184076e-04 -6.752678e-01 | -4.738107e-04 -1.350572e-02 | 1.351403e-02 | 4.560912e-04
13 -8.908034e-06 -3.567053e-05 | -1.781609e-08 -7.134106e-07 | 7.136330e-07 | 1.272390e-12
14 -1.951922e-12 -5.661131e-13 | -3.903843e-15 -1.132226e-14 | 1.197638e-14 | 3.206365e-28
```

8.16.4.3 Modified Newton's method - extreme case, positive definite Hessian

We also choose the extreme case $S = 0$ resulting in $\omega = 0$, thus

$$H_f(x,y) = \begin{pmatrix} 2(\kappa + y^2) & 0 \\ 4yx & 2(\varepsilon + x^2) \end{pmatrix}.$$

Here, the matrix H_f becomes positive semi-definite $det(H_f) = 4x^2y^2 + 4\kappa y^2$, even positive definite for any $y \neq 0$. Thus Newton can converge extremely fast in two steps, despite the fact that the modified H_f does not correspond to the true derivative of the right hand side residual $F'(x,y)$.

```
It x            y               dF(x)         dF(y)           ||(dF(x,y)||   F(x,y)
-------------------------------------------------------------------------------------
0 -5.000000e+00 4.000000e+00 | -1.600100e+02 2.000800e+02 | 2.561937e+02 | 4.099072e+05
1 0.000000e+00 7.996801e+00 | 0.000000e+00 1.599360e-01 | 1.599360e-01 | 6.394883e-02
2 0.000000e+00 -8.881784e-16 | 0.000000e+00 -1.776357e-17 | 1.776357e-17 | 7.888609e-34
```

8.16.5 Final comments

We first observe in the above computations that the local minimizer is not unique. Indeed, for all three examples, we obtain different final values for x and y.

Next, keep in mind that our problem of interest formulated in Chapter 5 is more complex since we do not seek a single point, but a solution (u^k, φ^k) minimizing $E(u^k, \varphi^k)$ in a function space setting. In general, we cannot expect that the Newton-like scheme with $\omega < 1$ converges as fast as in this example. However, these findings indicate us that the modified Newton scheme may work for the original problem at hand.

Recapitulating the key findings of this section, we found out that even for a pretended very simple optimization problem, Newton's method does not converge monotonically and may need many iterations. Secondly, the modification of the Jacobian according to Section 8.15 can yield a significant reduction of iteration steps. Thirdly, the line-search method with energy monitoring can specifically control the total energy, but should allow for working with negative line-search parameters.

8.17 A combined Newton algorithm and using a primal-dual active set strategy

In this section, we present a semi-smooth Newton method that combines the nonlinear iteration with the enforcement of the irreversibility condition. This Newton loop contains a back-tracking line search to improve the convergence radius.

8.17.1 Problem statement

We emphasize that the minimization problem (see Formulation 5.40)

$$\begin{aligned} &\min E_\varepsilon(u,\varphi),\\ &\text{subject to } \partial_t\varphi \le 0, \end{aligned}$$

is unusual since the forward problem is quasi-static without any explicit time derivatives and the time-dependence appears only in the inequality constraint. For the following, we set $U = (u,\varphi) \in V \times W$. Discretizing the time derivative via

$$\partial_t\varphi \approx \frac{\varphi(t_{n+1}) - \varphi(t_n)}{\delta t},$$

with a time step size $\delta t := t_{n+1} - t_n$, the problem can be rewritten as

$$\min E_\varepsilon(U) \tag{43}$$
$$\text{subject to } U \le \bar U \text{ on } \Phi, \tag{44}$$

where $\Phi = 0 \times W$, so that the constraint acts on the phase-field variable only, and $\bar U$ is the solution from the last time-step (or the initial condition).

Because one can show that the primal-dual active set method can be seen as a semi-smoothed Newton method (not just linear convergence, but superlinear), let us repeat the algorithm of a Newton method in the following.

We now briefly describe Newton's method for solving the unconstrained minimization problem $\min E_\varepsilon(U)$ in (43). We construct a sequence $U^0, U^1, \dots, U^N$ with

$$U^{k+1} = U^k + \delta U^k,$$

where the update δU^k is computed as the solution of the linear system:

$$E''_\varepsilon(U^k)\,\delta U^k = -E'_\varepsilon(U^k). \tag{45}$$

If we assume the constraints on the phase-field (44) hold for the initial guess U^0 (we will start with the solution from the last time step, which satisfies the constraint), the condition

$$\delta U^k \le 0 \text{ on } \Phi, \tag{46}$$

implies $U^{k+1} = U^k + \delta U^k \le U^k \le \cdots \le U^0 \le \bar U$ on Φ.

Let us now derive a primal-dual active set strategy for the linear system (45) with the constraint (46) to be solved in each Newton step:

$$\begin{aligned} E''_\varepsilon(U^k)\delta U^k &= -E'_\varepsilon(U^k),\\ \text{with } \delta U^k &\le 0 \text{ on } \Phi. \end{aligned}$$

Here, in contrast to the work originally proposed in [86], the constraint relates to time states (within a time-stepping scheme) rather than spatial constraints.

8.17.2 A primal-dual active set strategy

In the following, we shorten the notation by dropping the index k of Newton's method and setting $G := E''_\varepsilon(U^k)$, $F := -E'_\varepsilon(U^k)$ (this also highlights that our operator G is fixed in this section), and $\delta U := \delta U^k$. We note that ideally G is symmetric positive definite in order to employ a robust solution scheme. The previous system can be written as a minimization problem

$$\begin{aligned} &\min \frac{1}{2}(\delta U, G\delta U) - (F, \delta U), \\ &\text{with } \delta U \leq 0 \text{ on } \Phi. \end{aligned} \tag{47}$$

Following [86], the minimization problem (47) can be solved using a primal-dual active set strategy, which can also be viewed as a semi-smooth Newton method. We briefly recapitulate the most important steps since their understanding is crucial for our final algorithm. Using a Lagrange multiplier $\lambda \in 0 \times W^*$ (where W^* is the dual space of W), the minimization problem with constraint can be written as a system of equations:

$$\begin{aligned} (G\delta U, Z) + (\lambda, Z) &= (F, Z) \;\; \forall Z \in V \times W, \\ C(\delta U, \lambda) &= 0, \end{aligned}$$

where

$$C(\delta U, \lambda) = \lambda - \max(0, \lambda + c\delta U), \tag{48}$$

for a given $c > 0$. The *max* operation is understood in the point-wise sense. Since we require the Lagrange multiplier only for the phase-field variable φ, we can assume zero displacements (alternatively, one needs to restrict δU to the phase-field in the definition of C).

The primal-dual active set strategy replaces the condition $C(\delta U, \lambda) = 0$ by $\delta U = 0$ on the to be determined active set $\mathcal{A}$ and $\lambda = 0$ on the inactive set $\mathcal{N}$. In other words, the active set is the sub-domain in which the constraint applies and no PDE is solved. In the inactive set, the PDE is solved while the constraint is satisfied.

The active set algorithm then reads:

Algorithm 8.55. *Repeat for $k = 0, \dots$ until the active set $\mathcal{A}_k$ does not longer change:*

1. *Compute active set:*

$$\begin{aligned} \mathcal{A}_k &= \{x \mid \lambda^k(x) + c\delta U^k(x) > 0\}, \\ \mathcal{N}_k &= \{x \mid \lambda^k(x) + c\delta U^k(x) \leq 0\}. \end{aligned}$$

2. *Find $\delta U^{k+1} \in V \times W$ and $\lambda^{k+1} \in 0 \times W^*$*

$$\begin{aligned} (G\delta U^{k+1}, Z) + (\lambda^{k+1}, Z) &= (F, Z) && \forall Z \in V \times W, \\ (\delta U^{k+1}, \mu) &= 0 && \text{on } \mathcal{A}_k \quad \forall \mu \in 0 \times W^*, \\ \lambda^{k+1} &= 0 && \text{on } \mathcal{N}_k. \end{aligned}$$

Exercise 19. *Understand Algorithm 8.55 with the help of Algorithm 7.6.*

So far, the algorithm has been formulated on a continuous level. Next, we employ a finite element discretization by first subdividing the domain into quadrilateral elements. Displacements u and the phase-field variable φ are discretized using H^1-conforming bilinear elements, i.e., the Ansatz and test space uses Q_1^c-finite elements. Consequently, the discrete spaces are conforming such that $V_h \times W_h \subset V \times W$. A discretized version of Step 2 then results in a linear system with the following block structure:

$$\begin{pmatrix} G & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} \delta U_h^{k+1} \\ \lambda_h^{k+1} \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix}.$$

By using quadrature only in the support points of λ_h^k, B^T becomes diagonal and λ_h^k can be eliminated. The equations $B^T \delta U_h^{k+1} = 0$ will be handled via linear constraints used to eliminate equations in the G block where the phase-field is constrained (on $\mathcal{A}_k$). The eliminated equations are exactly those where the i-th entry of λ_h^{k+1} is non-zero. Therefore, the linear solver simplifies to

$$\hat{G}\delta U_h^{k+1} = \hat{F},$$

where $\hat{G}$ and $\hat{F}$ stem from G and F by removing the constrained rows from the system (we opt to restore symmetry by using Gaussian elimination on the columns in our implementation).

Finally, each entry of λ_h can be computed from U_h^{k+1} using

$$(B)_{ii}(\lambda_h^k)_i = (F)_i - (G\delta U_h^k)_i, \tag{49}$$

which is needed in the computation of the active set $\mathcal{A}$ in each step. The index i is in the active set $\mathcal{A}_k$ if

$$(B^{-1})_{ii}(F - G\delta U_h^k)_i + c(\delta U_h^k)_i > 0, \tag{50}$$

and in the inactive set $\mathcal{N}$ otherwise.

Remark 8.56. *Note that we require the matrix G and right-hand side F without constraints in the equations* (49) *and* (50)*.*

We recall that the condition for the active set

$$(B^{-1})_{ii}(F - G\delta U_h^k)_i + c(\delta U_h^k)_i > 0,$$

which reads (using the notation from before):

$$(B^{-1})_{ii}(-E'_\varepsilon(U_h^k) - E''_\varepsilon(U_h^k)\delta U_h^k)_i + c(\delta U_h^k)_i > 0.$$

We replace the linear residual $-E'_\varepsilon(U_h^k) - E''_\varepsilon(U_h^k)\delta U_h^k$ by the non-linear residual

$$R(U_h^{k+1}) = -E'_\varepsilon(U_h^{k+1}).$$

Remark 8.57. *Because we merge two Newton iterations it is no longer correct to just require $\delta U_h^k \le 0$ (pointwise) in each step since an intermediate active set allows a temporary violation of the crack growth condition during the Newton iteration. Therefore, we replace this condition by*

$$U_h^k + \delta U_h^k \le U_h^{old},$$

where U_h^{old} is the solution of the last time step.

8.17.3 A combined Newton algorithm

This gives us the algorithm:

Algorithm 8.58. *Repeat for $k = 0, \dots$ until the active set $\mathcal{A}_k$ does not change and $\widetilde{R}(U_h^k) <$ TOL:*

1. *Assemble the residual $R(U_h^k)$;*
2. *Compute the active set $\mathcal{A}_k = \{i \mid (B^{-1})_{ii}(R_k)_i + c(\delta U_h^k)_i > 0\}$;*
3. *Assemble the matrix $G = E''_\varepsilon(U_h^k)$ and the right-hand side $F = -E'_\varepsilon(U_h^k)$;*
4. *Eliminate rows and columns in $\mathcal{A}_k$ from G and F to obtain $\widetilde{G}$ and $\widetilde{F}$;*
5. *Solve the linear system $\widetilde{G}\delta U_k = \widetilde{F}$, i.e, find $\delta U_h^k \in V_h \times W_h$ with*
$$E''_\varepsilon(U_h^k)(\delta U_h^k, \Psi) = -E'_\varepsilon(U_h^k)(\Psi) \quad \forall \Psi \in V_h \times W_h, \tag{51}$$
where E''_ε and E'_ε are defined as before.
6. *Find a step size $0 < \omega \le 1$ using line search to get*
$$U_h^{k+1} = U_h^k + \omega\delta U_h^k,$$
with $\widetilde{R}(U_h^{k+1}) < \widetilde{R}(U_h^k)$.

Exercise 20. *Understand Algorithm 8.58 with the help of the Algorithms 8.55 and 7.6.*

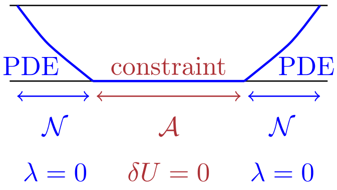

Figure 21: Determine the active set $\mathcal{A}$ and the nonactive set $\mathcal{N}$.

8.17.4 Solving the PDE in the Nonactive Set $\mathcal{N}$; including extrapolation of φ in the solid bulk term

On the nonactive set $\mathcal{N}$, the PDE is solved in a monolithic manner. Further, linearization of the critical terms is necessary.

- Monolithically-coupled variational formulation used to solve the forward PDE problem
- The Euler-Lagrange equations contain critical cross terms $\Big(\big((1-\kappa)\varphi^2 + \kappa\big)\,\sigma(u), e(u)\Big)$.
- Linearization of nonlinear terms by linear extrapolation:

$$\varphi \approx \tilde{\varphi} := \tilde{\varphi}^n = \varphi^{n-2}\frac{t_n - t_{n-1}}{t_{n-2} - t_{n-1}} + \varphi^{n-1}\frac{t_n - t_{n-2}}{t_{n-1} - t_{n-2}} \tag{52}$$

  to obtain a convex energy functional $E_\epsilon(u, \tilde{\varphi})$:

$$\begin{aligned} E_\epsilon(u, \tilde{\varphi}) = \frac{1}{2}\Big(\big((1-\kappa)\tilde{\varphi}^2 + \kappa\big)\,\sigma(u), e(u)\Big) \\ + G_C\left(\frac{1}{2\epsilon}\|1-\tilde{\varphi}\|^2 + \frac{\epsilon}{2}\|\nabla\tilde{\varphi}\|^2\right) \end{aligned} \tag{53}$$

Very briefly all fundamental solving steps are given.

Remark 8.59. *It is worth pointing out, that the residual $\widetilde{R}(U_h^k)$ might be far below the desired tolerance, however the active set can still change. Therefore, it is important to achieve both stopping criteria simultaneously:*

$$\mathcal{A}_{k+1} = \mathcal{A}_k \qquad \text{and} \qquad \widetilde{R}(U_h^k) < \mathrm{TOL}_{\mathrm{New}}\,.$$

Remark 8.60. *It is important to distinguish between the full residual $R(U_h^k)$ and the reduced residual $\widetilde{R}(U_h^k)$. The latter is the residual on the inactive set, which can be computed by eliminating the active set constraints from the former.*

Remark 8.61. *To address directly the numerical solution of the non-convex energy functional by minimization, the alternate minimization algorithm with backtracking line-search was suggested in [25, 28]. Here, it is utilized by noting that the energy functional is convex in each single variable when the other is kept fixed. The full convergence proof can be found in [35].*

Remark 8.62. *In optimization, it is well-known that active set and primal-dual active set methods may suffer from cycling and stalling of the algorithm, e.g., [76, 109]. Solutions to cope with these issues have been proposed in these theses.*

8.18 Linear solution inside Newton's method of monolithic phase-field fracture systems

We explain in this section the structure of the linear equation system when both solution variables u and φ are treated simultaneously, i.e., in a monolithic fashion. We recall:

$$\begin{aligned} A'(U_h^{n,j})(\delta U_h^n, \phi) &= -A(U_h^{n,j})(\phi) + F(\phi), \\ U_h^{n,j+1} &= U_h^{n,j} + \lambda \delta U_h^n. \end{aligned}$$

The linear equation system reads in matrix form:

$$M \delta U = B \tag{54}$$

where M has been defined before, B is the discretization of the residual:

$$B \sim A(u_h^{n,j})(\phi) + F(\phi)$$

and the solution vector δU is given by

$$\delta U = (\delta v_1, \ldots, \delta v_N, \delta u_1, \ldots, \delta u_M)^T.$$

8.18.1 Fully monolithic approximation

Since we dealt originally with two PDEs (now somewhat hidden in the semilinear form), it is often desirable to write (54) in block form:

$$\begin{pmatrix} M_{vv} & M_{vu} \\ M_{uv} & M_{uu} \end{pmatrix} \begin{pmatrix} \delta v \\ \delta u \end{pmatrix} = \begin{pmatrix} B_v \\ B_u \end{pmatrix},$$

where B_v and B_u are the residual parts corresponding to the first and second PDEs, respectively. Since in general such matrix systems are solved with iterative solvers, the block form allows a better view on the structure and construction of preconditioners. For instance, starting again from (54), a preconditioner is a matrix P^{-1} such that

$$P^{-1} M \delta U = P^{-1} B,$$

so that the condition number of $P^{-1}M$ is moderate. Obviously, the ideal preconditioner would be the inverse of A: $P^{-1} = A^{-1}$. In practice one tries to build P^{-1} in such a way that

$$P^{-1} M = \begin{pmatrix} I & * \\ 0 & I \end{pmatrix}$$

and where P^{-1} is a lower triangular block matrix:

$$P^{-1} = \begin{pmatrix} P_1^{-1} & 0 \\ P_3^{-1} & P_4^{-1} \end{pmatrix}$$

The procedure is as follows (see lectures for linear algebra in which the inverse is explicitly constructed):

$$\begin{aligned} M &= \begin{pmatrix} M_{vv} & M_{vu} \\ M_{uv} & M_{uu} \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix} \\ &= \begin{pmatrix} I & M_{vv}^{-1} M_{vu} \\ M_{uv} & M_{uu} \end{pmatrix} \begin{pmatrix} M_{vv}^{-1} & 0 \\ 0 & I \end{pmatrix} \\ &= \begin{pmatrix} I & M_{vv}^{-1} M_{vu} \\ 0 & \underbrace{M_{uu} - M_{uv} M_{vv}^{-1} M_{vu}}_{=S} \end{pmatrix} \begin{pmatrix} M_{vv}^{-1} & 0 \\ -M_{uv} M_{vv}^{-1} & I \end{pmatrix} \\ &= \begin{pmatrix} I & M_{vv}^{-1} M_{vu} \\ 0 & I \end{pmatrix} \underbrace{\begin{pmatrix} M_{vv}^{-1} & 0 \\ -S^{-1} M_{uv} M_{vv}^{-1} & S^{-1} \end{pmatrix}}_{=P^{-1}} \end{aligned}$$

where $S = M_{uu} - M_{uv}M_{vv}^{-1}M_{vu}$ is the so-called **Schur complement**. The matrix P^{-1} is used as (exact) preconditioner for M. Indeed we double-check:

$$\begin{pmatrix} M_{vv}^{-1} & 0 \\ -S^{-1}M_{uv}M_{vv}^{-1} & S^{-1} \end{pmatrix} \begin{pmatrix} M_{vv} & M_{vu} \\ M_{uv} & M_{uu} \end{pmatrix} = \begin{pmatrix} I & M_{vv}^{-1}M_{vu} \\ 0 & I \end{pmatrix}$$

Tacitly we assumed in the entire procedure that S and M_{vv} are invertible.

Using P^{-1} in a Krylov method, we only have to perform matrix-vector multiplications such as

$$\begin{pmatrix} X_{new} \\ Y_{new} \end{pmatrix} = \begin{pmatrix} P_1^{-1} & 0 \\ P_3^{-1} & P_4^{-1} \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix}$$

Now we obtain:

$$X_{new} = P_1^{-1}X \tag{55}$$

$$Y_{new} = P_3^{-1}X + P_4^{-1}Y \tag{56}$$

Remark 8.63. *Be careful, we deal with two iterative procedures in this example: Newton's method to compute iteratively the nonlinear solution. Inside Newton's method, we solve the linear equations systems with a Krylov space method, which is also an iterative method.*

8.18.2 A block-diagonal preconditioner for the extrapolated scheme

A block-diagonal preconditioner:

$$P^{-1} = \begin{pmatrix} (P^{uu})^{-1} & 0 \\ 0 & (P^{\varphi\varphi})^{-1} \end{pmatrix}$$

Since both blocks have an elliptic structure, it is simple to approximate the inverse matrices. One possibility is an algebraic multigrid solver. Approximately, we have:

$$P_{uu} \approx -\nabla \cdot (\varphi^2 \nabla u),$$

$$P_{\varphi\varphi} \approx \varepsilon \Delta \varphi - \frac{1}{\varepsilon}(1 - \varphi),$$

which both are of elliptic type and are therefore 'nice' terms to deal with. This preconditioner works very well as recently shown in [83].

8.18.3 A very short hint to parallel computing

The principle idea of parallel computing using deal.II ([13]), MPI, Trilinos ([84]), and p4est ([36]) is explained with the help of Figure 22 using four processors. In particular, this framework can be run on a cluster as demonstrated in [83].

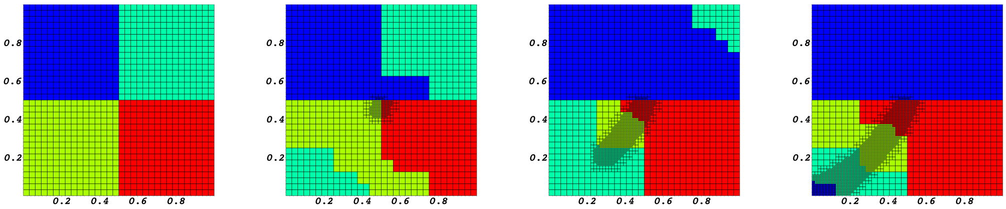

Figure 22: Exemplified visualization of parallel computing on 4 processors. The different sub-domains are associated with different processors. Depending on mesh refinement, the workload for each processor is adjusted dynamically at each time step. Figure taken from [5]. The original code was developed in [82] and recently extended in [83].

8.19 Excursus III: linear and nonlinear solver performances for the obstacle problem

In this excursus, we study the performance of the linear solver and the nonlinear Newton solver using a modified version of the code

http://www.thomaswick.org/links/Example_Obstacle_Simple_Penalization.zip

that we already used in Excursus II (Section 8.13).

8.19.1 Setup and goals

To this end, we study the behavior of the conjugate gradient (CG) method, without any preconditioner for simplicity. The CG method is used a linear solver within Newton's method. The initial mesh is five times uniformly refined yielding 1 089 degrees of freedom. The obstacle function g and the penalization parameter $\gamma = \frac{\bar{\gamma}}{h^2}$ are chosen as in Section 8.13.

The aim is to study the solver performance for a fixed mesh, but varying penalization parameter. This shall give some insight how the penalization parameters influnce the solvers. In addition, we measure how well the obstacle is approximated in $u(0.5, 0.5)$ indicated as `Point value in X`.

We choose:

$$\bar{\gamma} = 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 12.8, 25.6, 51.2, 102.4.$$

The solver tolerances are

```
TOL_New = 1e-10 // Newton
TOL_Lin = 1e-12 // CG
```

8.19.2 Results

The following results are obtained. For $\bar{\gamma} = 0.1$:

```
Newton step: 0  Residual (abs.):   9.7656e-04
Newton step: 0  Residual (rel.):   1.0000e+00
Newton step: 1  Residual (rel.):   8.7127e-01    55 CG iter  LineSearch {1}
Newton step: 2  Residual (rel.):   4.2824e-02    77 CG iter  LineSearch {0}
Newton step: 3  Residual (rel.):   5.9283e-05    77 CG iter  LineSearch {0}
Newton step: 4  Residual (rel.):   1.3756e-07    49 CG iter  LineSearch {0}
Newton step: 5  Residual (rel.):   3.1917e-10    43 CG iter  LineSearch {0}
Point value in X: -0.0266278
```

For $\bar{\gamma} = 0.2$:

```
Newton step: 0  Residual (abs.):   9.7656e-04
Newton step: 0  Residual (rel.):   1.0000e+00
Newton step: 1  Residual (rel.):   7.5000e-01   55 CG iter  LineSearch {2}
Newton step: 2  Residual (rel.):   3.7500e-01   77 CG iter  LineSearch {1}
Newton step: 3  Residual (rel.):   1.2817e-01   75 CG iter  LineSearch {0}
Newton step: 4  Residual (rel.):   7.4588e-04   74 CG iter  LineSearch {0}
Newton step: 5  Residual (rel.): < 1.0000e-11   48 CG iter  LineSearch {0}
Point value in X: -0.0192812
```

For $\bar{\gamma} = 0.4$:

```
Newton step: 0  Residual (abs.):   9.7656e-04
Newton step: 0  Residual (rel.):   1.0000e+00
Newton step: 1  Residual (rel.):   9.6854e-01   55 CG iter  LineSearch {2}
Newton step: 2  Residual (rel.):   4.8427e-01   74 CG iter  LineSearch {1}
Newton step: 3  Residual (rel.):   1.8561e-01   73 CG iter  LineSearch {0}
Newton step: 4  Residual (rel.):   3.6525e-03   71 CG iter  LineSearch {0}
Newton step: 5  Residual (rel.): < 1.0000e-11   46 CG iter  LineSearch {0}
Point value in X: -0.0148327
```

For $\bar{\gamma} = 0.8$:

```
Newton step: 0  Residual (abs.):   9.7656e-04
Newton step: 0  Residual (rel.):   1.0000e+00
Newton step: 1  Residual (rel.):   8.7500e-01   55 CG iter  LineSearch {3}
Newton step: 2  Residual (rel.):   8.2031e-01   53 CG iter  LineSearch {4}
Newton step: 3  Residual (rel.):   6.1523e-01   74 CG iter  LineSearch {2}
Newton step: 4  Residual (rel.):   3.0762e-01   71 CG iter  LineSearch {1}
Newton step: 5  Residual (rel.):   1.8146e-01   69 CG iter  LineSearch {0}
Newton step: 6  Residual (rel.):   2.3458e-03   67 CG iter  LineSearch {0}
Newton step: 7  Residual (rel.): < 1.0000e-11   42 CG iter  LineSearch {0}
Point value in X: -0.0124388
```

For $\bar{\gamma} = 1.6$:

```
Newton step: 0  Residual (abs.):   9.7656e-04
Newton step: 0  Residual (rel.):   1.0000e+00
Newton step: 1  Residual (rel.):   8.7500e-01   55 CG iter  LineSearch {3}
Newton step: 2  Residual (rel.):   8.4766e-01   53 CG iter  LineSearch {5}
Newton step: 3  Residual (rel.):   7.4170e-01   76 CG iter  LineSearch {3}
Newton step: 4  Residual (rel.):   5.5627e-01   72 CG iter  LineSearch {2}
Newton step: 5  Residual (rel.):   4.2296e-01   68 CG iter  LineSearch {1}
Newton step: 6  Residual (rel.):   2.1071e-01   66 CG iter  LineSearch {0}
Newton step: 7  Residual (rel.):   6.9472e-03   64 CG iter  LineSearch {0}
Newton step: 8  Residual (rel.): < 1.0000e-11   36 CG iter LineSearch {0}
Point value in X: -0.0112207
```

For $\bar{\gamma} = 3.2$:

```
... 11 Newton iter, linear iter 32 - 78

Point value in X: -0.0106104
```

For $\bar{\gamma} = 6.4$:

```
... 13 Newton iter, linear iter 34 - 80

Point value in X: -0.0103052
```

For $\bar{\gamma} = 12.8$:

```
... Newton iter 15, linear iter 55 - 90

Point value in X: -0.0101526
```

For $\bar{\gamma} = 25.6$:

```
... Newton iter 25, linear iter 55 - 117

Point value in X: -0.0100763
```

For $\bar{\gamma} = 51.2$:

```
... Newton iter 34, linear iter 55 - 155

Point value in X: -0.0100381
```

For $\bar{\gamma} = 102.4$:

```
... Newton iter 42, linear iter 55 - 210

Point value in X: -0.0100191
```

8.19.3 Discussion

We observe in our results that the nonlinear solver suffers much more from a higher penalization parameter than the linear solver. While we observe an increase from 77 (max.) to 90 (max.) linear iterations for $\bar{\gamma} = 0.1$ and $\bar{\gamma} = 12.8$, respectively, the Newton iterations increase from 5 to 15. This is the main reason why other nonlinear techniques such as augmented Lagrangian, active set and so forth have been proposed to improve the nonlinear solver performance. From $\bar{\gamma} = 25.6$, we also see a significant increase in linear iterations, indicating that the condition number of the matrices inside Newton's method increases (i.e., ill-conditioning due to the penalization).

8.20 The augmented Lagrangian loop and an inexact version

In order to enforce the penalty parameter, when working with Formulation 5.39 or 5.41, simply penalization leads to ill-conditioned systems as described earlier. In this section, we explain an update procedure to enforce this constraint. At each time $t_n, n \in \mathbb{N}$, the augmented Lagrangian loop constitutes the outer loop wherein at each step the Newton solver is adopted. Moreover, we propose a (heuristic) adaptive criterion in order to reduce the computational cost. This leads to an inexact augmented Lagrangian/Newton loop.

8.20.1 Augmented Lagrangian penalization

The iteration reads:

Algorithm 8.64 (Inexact augmented Lagrangian loop with inner Newton solver [175])**.** *At each $t_n, n = 0, 1, 2, \dots$ let $\Xi_{h,0}$ be given, e.g., $\Xi_{h,0} = 0$. Moreover, let $\gamma > 0$ be fixed and given for all t_n. At each time t_n iterate for $m = 0, 1, 2, \dots$*

1. *Given $\Xi_{h,m}$, we seek $U^n_{h,m+1} = \{u^n_{h,m+1}, \varphi^n_{h,m+1}\}$ by solving Formulation 8.35 with Newton's method via a Newton-type algorithm presented in Section 8.14 or 8.15.*
2. *Update*
$$\Xi_{h,m+1} = [\Xi_{h,m} + \gamma(\varphi_{h,m+1} - \varphi^{n-1}_h)]^+.$$
3. *Check the stopping criterion*
$$\{\|u^n_{h,m+1} - u^n_{h,m}\|_{L^2}, \|\Xi_{h,m+1} - \Xi_{h,m}\|_{L^2}\} \leq \mathrm{TOL}_{\mathrm{AL}}, \quad \mathrm{TOL}_{\mathrm{AL}} > 0. \qquad (57)$$
4. *a) If the stopping criterion is satisfied, set $U^n_h := U^n_{h,m^*}$ where m^* is the m that satisfies* (57)*.*
b) Else increment $m \to m + 1$ and go to Step 1.

8.20.2 An adaptive Newton stopping criterion for an inexact augmented Lagrangian method

In order to decrease the computational cost, we adaptively determine the stopping tolerance of Newton's method depending on the augmented Lagrangian norm. Such strategies are in particular well-known for (adaptive) inexact Newton methods, e.g., [37, 50, 51], in which the linear equations are only approximately solved at each stage. Related techniques using adaptive (or inexact) augmented Lagrangian realizations are discussed, for instance, in [139]. In all these adaptive inexact methods, the accuracy of the inner method should be chosen as such that the convergence pattern of the outer loop remains unperturbed.

Proposition 8.65 ([175])**.** *For $m = 0$ set $\Xi_{h,0}$ and, e.g., $TOL_N = 10^{-8}$. For $m = 1, 2, 3, \dots$ use Algorithm 8.64 and compute in Step 4b*
$$\Delta := \|\Xi_{h,m+1} - \Xi_{h,m}\|_{L^2}.$$
For each further augmented Lagrangian step m, we use in the inner Newton loop (part of Step 1) as stopping criterion
$$TOL_N := \alpha\Delta,$$
where $\alpha = 10^{-3} < 1$. The most important question in this respect is what level of accuracy of the inner solver (here Newton's method) is required to preserve convergence of the outer loop. As it is well-known the inner loop must be solved with a higher accuracy than the outer loop, i.e., $TOL_N < TOL_{AL}$. Therefore, $\alpha < 1$ is a necessary choice. From our practical observations in this paper, the largest α should be chosen as 10^{-3} in these types of problems.

8.21 Updating the strain history field when working with the variational equality system

As stated in Formulation 5.48, the strain history field concerns the second equation, namely the φ terms. Consequently, we can have two models:

- Monolithic treatment of φ in the u-equation.
- Using the φ extrapolation in the u-equation.

In particular, when the second model is employed, we can use work with the standard extrapolation (see Section 8.5) or the iteration on the extrapolation presented in Section 8.6. For the strain history we apply the same schemes: we can update either at each time step (yielding a relatively large approximation error in time) or we can iterate per time step a few times, which reduces the temporal error.

Updating strain history field

Algorithm 8.66. *1. Initialize $H(x, t_0) = H_0$*

2. For $n = 1, \ldots, N$ with

$$H = H_n = \begin{cases} \psi_E^+(e) & \psi_E^+(e) > H_{n-1} \\ H_{n-1} & otherwise \end{cases}$$

with the positive strain energy ψ_E^+ as defined before.

Remark 8.67. *H needs to be evaluated at each quadrature point in the inner FEM loop.*

Remark 8.68. *If the iteration on the extrapolation us used, then H_n can be updated as well at each extrapolation iteration step.*

9 Simulations I: Single edge notched shear test

We consider the single edge notched shear test, which has been often studied in the literature, e.g., [7, 24, 82, 117, 121, 175, 177]. In extension to published literature, we first demonstrate that this test case can be easily re-done by anybody. Then, we carefully describe the computational setting and finally present many computational results from which the most are not published in journal papers and are therefore novel findings. Here, the focus is on several comparisons on how to impose the crack irreversibility condition and secondly, to relax the nonlinearity in the first term of the u-equation by using extrapolation and an iterated extrapolation with only a few iterations.

9.1 Relation to reality

Everybody can re-do this test by his/her own as illustrated in Figure 23.

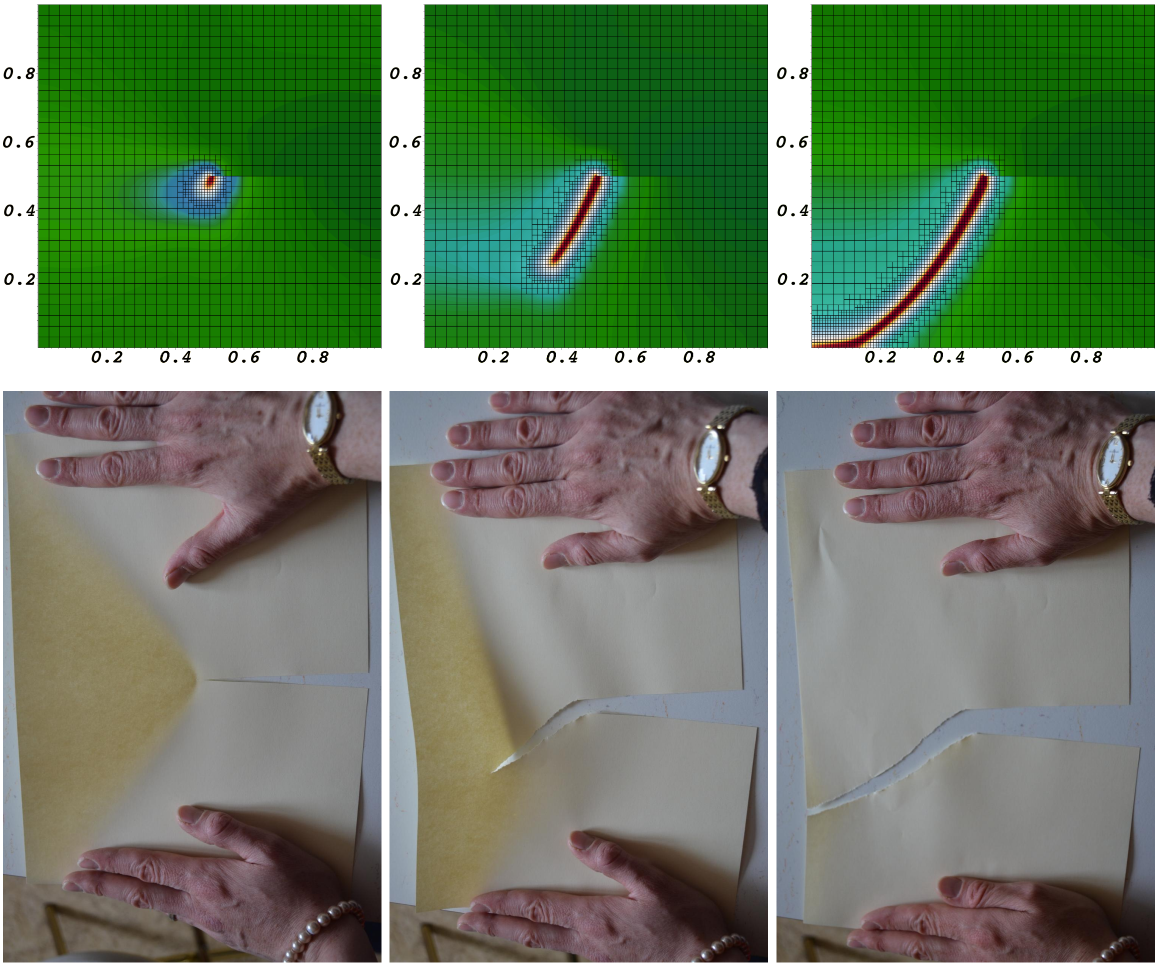

Figure 23: Comparison of experiment and numerical simulation of the single edge notched shear test.

9.2 Goals of our computations

We demonstrate the following numerical techniques:

- Computational analysis of different time step sizes (better to say: loading incremental steps).
- Extrapolation on the phase-field variable in the u equation.
- An iteration on the phase-field variable (new!).
- Comparison of the strain-history formulation (without explicit treatment of the inequality constraint) and the augmented Lagrangian penalization.
- Performance of the iteration on the extrapolation and augmented Lagrangian penalization.
- Performance of the iteration on the extrapolation and strain history formulation.
- Performance of the simultaneous iteration of both the extrapolation and strain history function.
- Computations using the primal-dual active set strategy with extrapolation and an iteration on the extrapolation.
- All simulations are analyzed in terms of
  - the time step when the crack reaches the lower boundary;
  - the load-displacement curve on the top boundary in shear direction;
  - Newton iteration numbers.

Table 1: Summary of all test cases. SH(n) = strain history with n iterations (together with extrapolation iterations), AL = augmented Lagrangian, PDAS = primal-dual active set, k time step size (loading step size).

Case	PFF var.	Iter. on extra.	Crack irr.	k
1	φ	–	SH(0)	$1e-4$
2	φ	–	AL	$1e-4$
3	$\tilde{\varphi}$	0	SH(0)	$1e-4$
4	$\tilde{\varphi}$	0	AL	$1e-4$
5	$\tilde{\varphi}$	3	SH(0)	$1e-4$
6	$\tilde{\varphi}$	3	AL	$1e-4$
7	$\tilde{\varphi}$	3	SH(3)	$1e-4$
8	$\tilde{\varphi}$	5	SH(5)	$1e-4$
9	$\tilde{\varphi}$	0	SH(0)	$5e-5$
10	$\tilde{\varphi}$	0	SH(0)	$2.5e-5$
11	$\tilde{\varphi}$	0	SH(0)	$1.25e-5$
12	$\tilde{\varphi}$	0	SH(0)	$1e-5$
13	$\tilde{\varphi}$	0	PDAS	$1e-4$
14	$\tilde{\varphi}$	3	PDAS	$1e-4$

Furthermore, in the previous sections, we already provided these findings:

- Figure 22: Partitioning of the computational domain using parallel computing (using the code published on

  https://github.com/tjhei/cracks

  see also [82].
- Figure 35 and Figure 36: Predictor-corrector mesh adaptivity (the mesh grows with the crack path) [82].

9.3 Configuration

The geometric and material properties are the same as used in [117]. In the single edge notched shear test, it is important to consider the correct boundary conditions and the spectral decomposition of the stress $\sigma(u)$ into tensile $\sigma^+(u)$ and compressive parts $\sigma^-(u)$. We refer to [11, 121] for a detailed physical motivation. A comparison highlighting the properties of one or the other splitting model has been published in [7]. In particular, the Miehe et al. splitting does not release all stresses once the fracture reaches the bottom part of the specimen. The characteristic feature of this test is that an initial crack is prescribed in the geometry rather than with phase-field and that the crack will slowly develop, followed by faster growth.

The geometry and boundary conditions are displayed in Figure 24. In particular, the initial domain has already a slit (fracture). The initial mesh is 4 times uniformly refined, leading to 1024 mesh cells, yielding 2210 DoFs for the solid, 1105 DoFs for the phase-field variable and 3315 DoFs in total. Here, $h = 0.044mm$. For mesh refinement studies we refer to published work, e.g., [82].

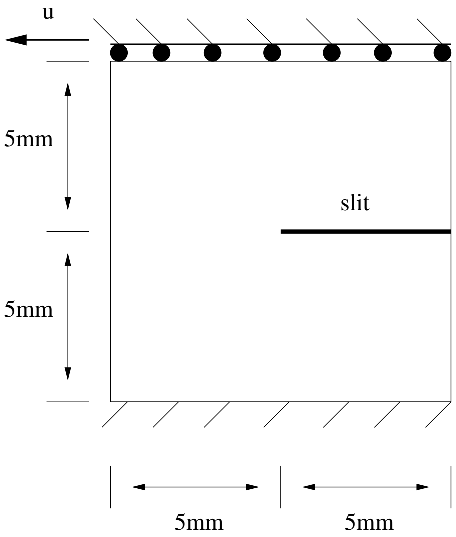

Figure 24: Example 1: Single edge notched shear test. We prescribe the following conditions: On the left and right boundaries as well as on the lower part of the slit, $u_y = 0mm$ and traction-free in x-direction. On the bottom part, we use $u_x = u_y = 0mm$ and on Γ_{top}, we prescribe $u_y = 0mm$ and u_x as stated in (58). Finally, the lower part of the slit is fixed in y-direction, i.e., $u_y = 0mm$. We notice that the initial crack is described in the geometry by doubling the degrees of freedom on the respective faces. Consequently, the initial phase-field is $\varphi^0 = 1$ in the entire domain.

9.4 Boundary conditions

We increase the displacement on Γ_{top} over time, namely we apply a time-dependent non-homogeneous Dirichlet condition:

$$u_x = t\bar{u}, \quad \bar{u} = 1 \text{ mm/s}, \tag{58}$$

where t denotes the total time. For phase-field, we prescribe homogeneous Neumann conditions (traction-free) on the entire boundary.

9.5 Initial conditions

The initial phase-field is given by $\varphi^0 = 1$ because the initial crack is represented in this example directly inside the geometry. This is simply done by doubling the number of degrees of freedom in the initial coarse mesh. For the displacements, no initial conditions need to be prescribed.

9.6 Parameters

Specifically, we use $\mu = 80.77kN/mm^2$, $\lambda = 121.15kN/mm^2$, and $G_c = 2.7N/mm$. In this example $p = 0$. The time step size is chosen as $k = 10^{-4}s$. Furthermore, we set $\kappa = 10^{-12}h[mm]$ and $\varepsilon = 2h$.

9.7 Quantities of interest

To check the solution, we observe the crack path and in particular, the time instant when the crack reaches the lower boundary. Secondly, we evaluate the surface load vector on $\Gamma_{top} := \{(x, y) \in B | \, 0mm \leq x \leq 10mm, \, y = 10mm\}$ as

$$\tau = \frac{1}{|\Gamma_{top}|}(F_x, F_y) := \int_{\Gamma_{top}} \sigma(u)n \, ds,$$

with normal vector n, and we are particularly interested in F_x. Usually, the integral is weighted with the length of the boundary of interest. In this case $|\Gamma_{top}|$ corresponds to a length of $10mm$, but our geometry is normalized to a size of 1×1.

9.8 Discussion of findings

The crack pattern at various time points and the final displacement field are shown in Figure 25 and is in good agreement to other results reported in the literature, e.g., [7, 24, 82, 117, 175, 177]. The first important observation can be found in Figure 26, which shows in the left and right sub-figures that independently of the specific Newton scheme, the load-displacement curve does not change.

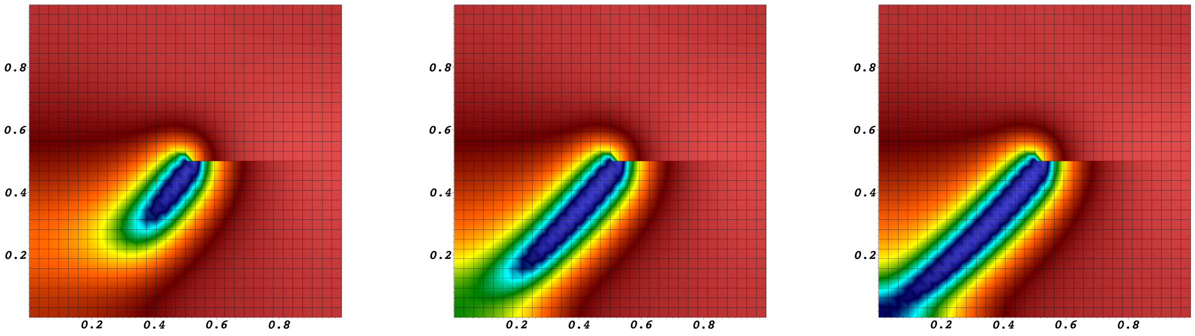

Figure 25: Case 14: Crack path using the primal-dual active set strategy with iterated extrapolation $m_{PC} = 3$ at $T = 0.011s, 0.015s$ and $T = 0.017s$.

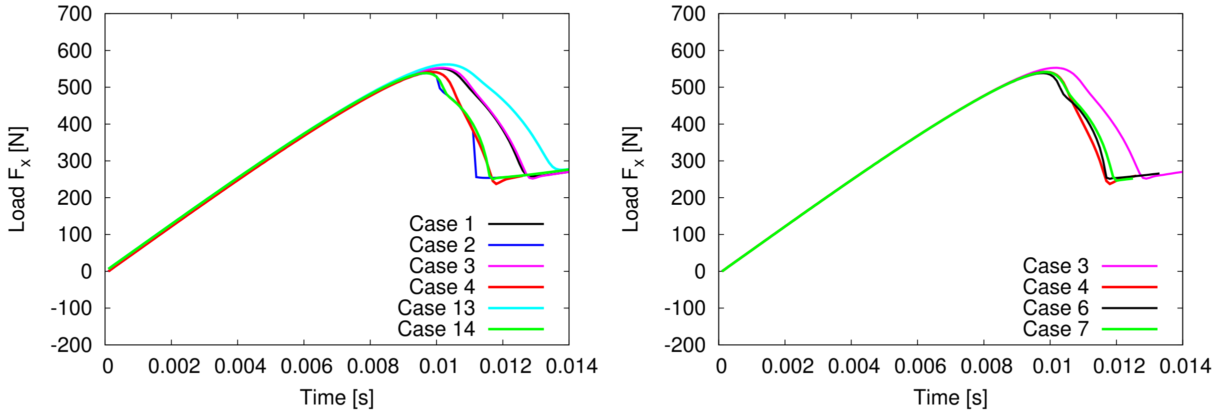

Figure 26: Load-displacement curves: time versus F_x.

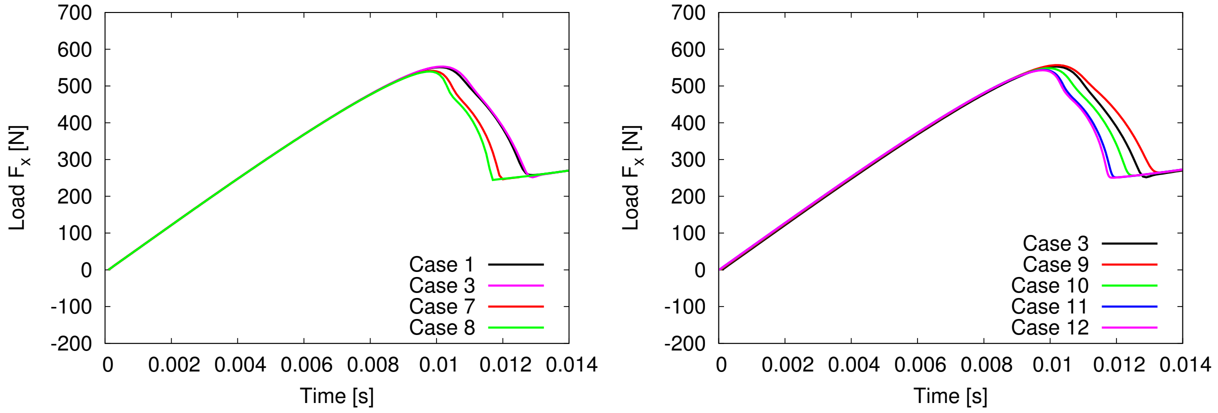

Figure 27: Load-displacement curves: time versus F_x.

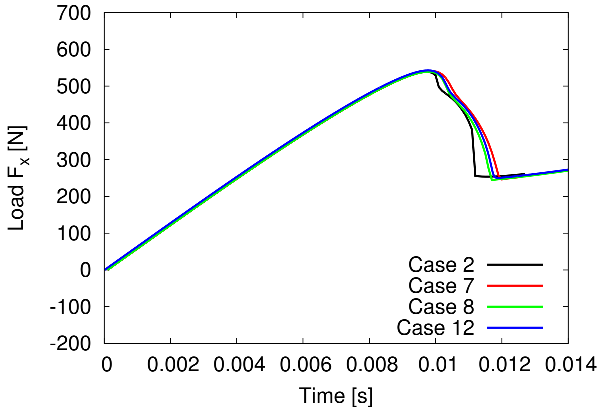

Figure 28: Load-displacement curves: time versus F_x.

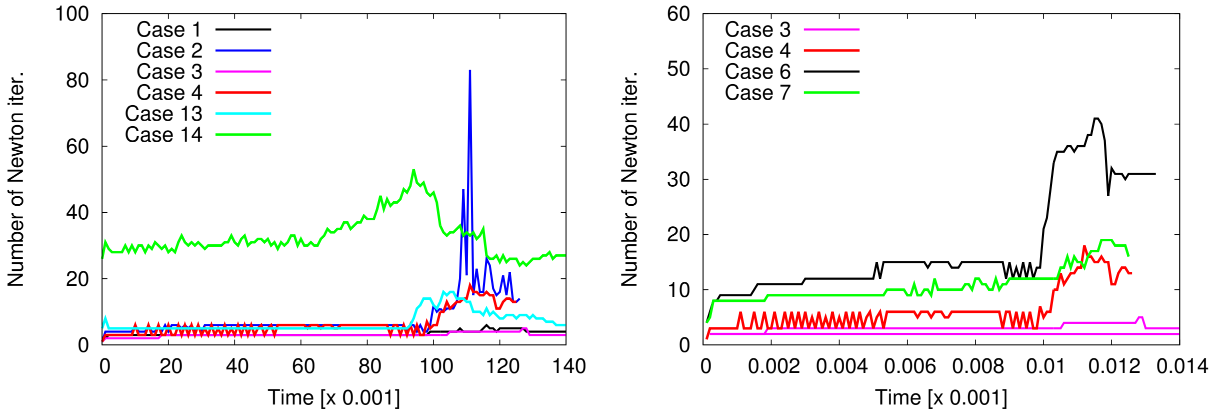

Figure 29: Newton iterations per time step.

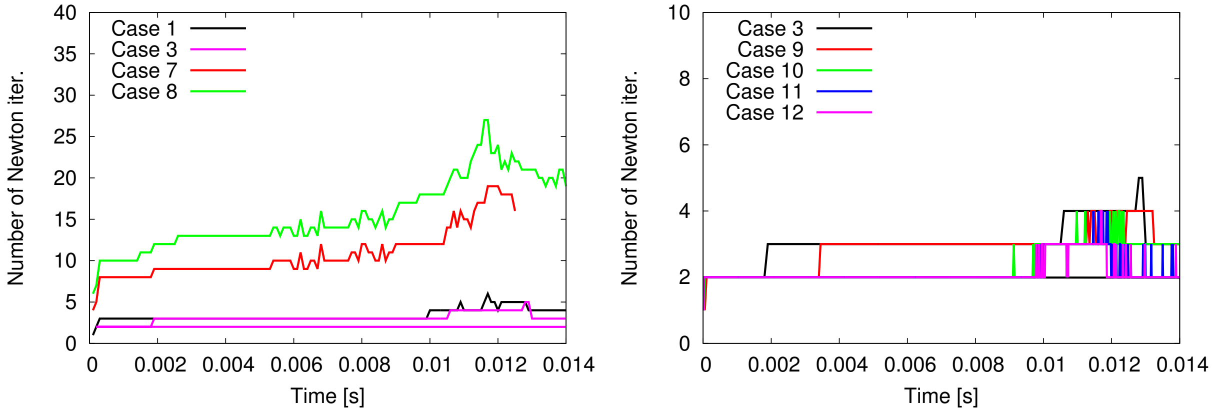

Figure 30: Newton iterations per time step.

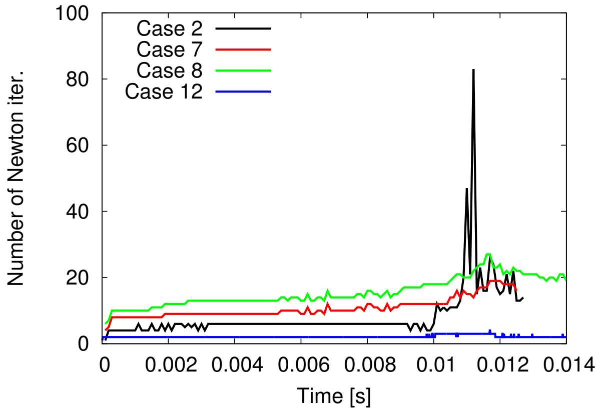

Figure 31: Newton iterations per time step.

10 Numerical modeling part III: Adaptivity and goal functionals

In this chapter, we concentrate on some special topics that extend classical modeling and standard numerical techniques.

10.1 Motivation of mesh adaptivity

In this very first subsection, we closely follow [144][Sect. 5.1] for motivating mesh adaptivity. The goal of any simulation is to obtain a numerical (discrete) solution u_h with **sufficient accuracy** at **minimal computational cost**. However, often, not the entire numerical solution u_h is of interest, but only a part. This 'part' can be expressed through a so-called **goal functional** $J(u_h)$. Often such 'parts of u_h' are motivated by requests from engineering, physics or other applications, and represent a **(physical) quantity of interest (QoI)**. The aim is then to estimate the error:

$$J(u) - J(u_h).$$

More precisely, for $h \to 0$, we hope for

$$|J(u) - J(u_h)| \to 0.$$

This will be exactly the discussion in the remainder of this chapter. In practice we rather aim for

$$|J(u) - J(u_h)| < TOL.$$

Let us briefly come back to the overall goals of adaptivity. It is the 'optimal use of computer resources' [144] according to either one of the two principles:

- Minimal work, h 'large' (coarse mesh) for a prescribed accuracy TOL.
- Best accuracy TOL for a given work, h is fixed.

These goals can be achieved with the following possibilities:

- Adaptive discretization;
- Adaptive design of stopping criteria for linear and nonlinear solvers;
- Model adaptivity.

In these lecture notes, we concentrate on the discretization error only. For the other techniques, specifically using goal-oriented techniques, we refer for instance to [31, 137] (model and discretization errors), [56, 148] (discretization and nonlinear iteration error), [18, 114] (discretization and linear error). But we also want to mention the work [58] and the recent extension to goal-oriented methods [110]. Finally, the recent Acta Numerica paper from J.T. Oden [136] is recommended; see Section 9.2 in [136][3]

10.2 Principles of error estimation

In general, we distinguish between **a priori** and **a posteriori error estimation**. In the first one, which we already discussed for finite differences and finite elements, the (discretization) error is estimated before we start a numerical simulation/computation. Often, there are, however, constants c and (unknown) higher-order derivatives $|u|_{H^m}$ of the (unknown) solution:

$$\|u - u_h\| \leq ch^m |u|_{H^m}.$$

Such estimates yield

- the asymptotic information of the error for $h \to 0$.
- In particular, they provide the expected order of convergence, which can be adopted to verify programming codes and the findings of numerical simulations for (simple?!) model problems.

[3] Just as remark to [136] for a general education point-of-view: it is worthy to read Sect. 1 and 2 in [136].

However, a priori estimations do not contain useful information during a computation. Specifically, the determination of better, reliable, adaptive step sizes h is nearly impossible (except for very simple cases).

On the other hand, a posteriori error estimation uses information of the already computed u_h during a simulation. Here, asymptotic information of u is not required:

$$\|u - u_h\| \leq c\eta(u_h)$$

where $\eta(u_h)$ is a computable quantity, because, as said, they work with the known discrete solution u_h. Such estimates cannot in general predict the asymptotic behavior, the reason for which a priori estimates remain important, but they can be evaluated during a computation with two main advantages:

- We can **control** the error during a computation;
- We can possibly **localize** the error estimator in order to obtain local error information on each element $K_i \in \mathcal{T}_h$. The elements with the highest error information can be decomposed into smaller elements in order to reduce the error.

10.3 Efficiency, reliability, basic adaptive algorithm

Following Becker/Rannacher [19, 20], Bangerth/Rannacher [17], and Rannacher [143], we derive a posteriori error estimates not only for norms, but for a general differentiable functional of interest $J : V \to \mathbb{R}$. This allows in particular, to estimate technical quantities arising from applications (mechanical and civil engineering, fluid dynamics, solid dynamics, etc.). Of course, norm-based error estimation, $\|u - u_h\|$ can be expressed in terms of $J(\cdot)$.

When designing a posteriori error estimates, we are in principle interested in the following relationship:

$$C_1\eta \leq |J(u) - J(u_h)| \leq C_2\eta \tag{59}$$

where $\eta := \eta(u_h)$ is the **error estimator** and C_1, C_2 are positive constants. Moreover, $J(u) - J(u_h)$ is the **true error**.

Definition 10.1 (Efficiency and reliability)**.** *A good estimator $\eta := \eta(u_h)$ should satisfy two bounds:*

1. *An error estimator η of the form* (59) *is called **efficient** when*

$$C_1\eta \leq |J(u) - J(u_h)|,$$

which means that the error estimator is bounded by the error itself.

2. *An error estimator η of the form* (59) *is called **reliable** when*

$$|J(u) - J(u_h)| \leq C_2\eta.$$

Here, the true error is bounded by the estimator.

Remark 10.2. *For general goal functionals $J(\cdot)$, it is much more simpler to derive **reliable** estimators rather than proving their efficiency.*

Remark 10.3 (AFEM)**.** *Finite element frameworks working with a posteriori error estimators applied to local mesh adaptivity, are called **adaptive FEM**.*

Definition 10.4 (Basic algorithm of AFEM)**.** *The basic algorithm for AFEM is always the same:*

1. ***Solve** the PDE on the current mesh $\mathcal{T}_h$;*
2. ***Estimate** the error via a posteriori error estimation to obtain η;*
3. ***Mark** the elements by localizing the error estimator;*
4. ***Refine/coarsen** the elements with the highest/lowest error contributions using a certain refinement strategy.*

A prototype situation on three meshes is displayed in Figure 32.

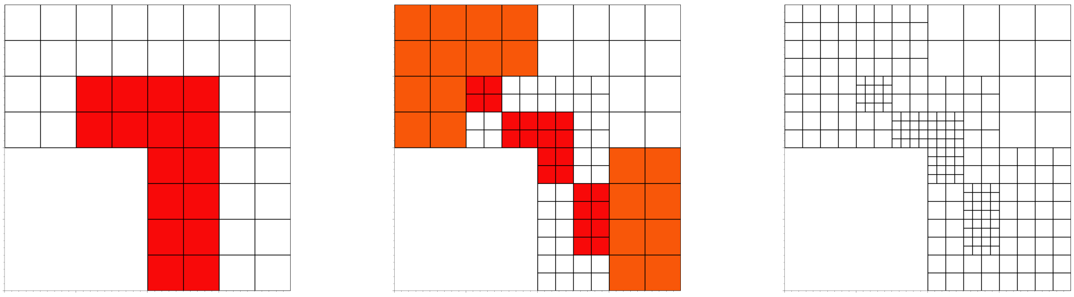

Figure 32: Meshes, say, on level 0, level 1 and 2. The colored mesh elements indicate high local errors and are marked for refinement using bisection and hanging nodes.

10.4 Goal-oriented error estimation using duality arguments: dual-weighted residuals (DWR)

10.4.1 Problem statement

The goal of this section is to derive an error estimator that is based on duality arguments and can be used for norm-based error estimation (residual-based) as well as estimating more general error functionals $J(\cdot)$. The key idea is based on numerical optimization; see for instance the classical works from Pontryagin et al. (1964) and Lions (1971) [108, 140].

From our previous considerations, our goal is to reduce the error in the functional of interest with respect to a given PDE:

Problem 10.5.

$$\min\big(J(u) - J(u_h)\big) \quad s.t. \quad a(u,\phi) = l(\phi), \tag{60}$$

where $J(\cdot)$ and $a(u,\phi)$ can be linear or nonlinear, but need to be differentiable (in Banach spaces).

Remark 10.6. *Of course for linear functionals, we can write*

$$J(u) - J(u_h) = J(u - u_h) = J(e), \quad e = u - u_h.$$

10.4.2 Lagrangian, primal, adjoint

Such minimization problems as in (60) can be treated with the help of the so-called **Lagrangian** $L : V \times V \to \mathbb{R}$ in which the functional $J(\cdot)$ is of main interest subject to the constraint $a(\cdot,\cdot) - l(\phi)$ (here the PDE in variational form). Here, we deal with the **primal variable** $u \in V$ and a **Lagrange multiplier** $z \in V$, which is the so-called **adjoint variable** and which is assigned to the constraint $a(u,z) - l(z) = 0$. We then obtain

Definition 10.7. *The Lagrangian $L : V \times V \to \mathbb{R}$ representing* (60) *is defined as*

$$L(u,z) = \big(J(u) - J(u_h)\big) - a(u,z) + l(z).$$

We provide some comments on the adjoint. In finite-dimensional spaces, we know that for $A \in \mathbb{R}^{m\times n}$, we have for $u \in \mathbb{R}^n$ and $v \in R^m$:

$$(Au, v) = (u, A^T v)$$

where A^T is the transposed matrix of A.

Remark 10.8. *In the discretization of the adjoint later, we exactly use A^T in order to compute the adjoint solution z.*

This concept can be extended to infinite-dimensional spaces as follows:

Definition 10.9 (Adjoint operator)**.** *Let U, V be normed spaces and let $A : U \to V$ a linear, bounded, mapping. We define the adjoint operator $A^* : V^* \to U^*$ via*

$$A^*(v^*)(u) = v^*(Au) = g(u),$$

where $g : U \to \mathbb{R}$. It holds

$$|g(u)| \leq \|v^*\|_{V^*} \|A\| \|u\|_U.$$

Consequently, g is bounded, which implies the continuity. Therefore, it indeed holds $g \in U^*$. For the norm holds furthermore:

$$\|g\|_{U^*} = \|A^* v^*\|_{U^*} \leq \|A\| \|v^*\|.$$

Since A is continuous, the last estimate shows that A^* is also bounded (i.e., continuous). If U and V are real Hilbert spaces, we have:

$$(Au, v)_V = (u, A^* v)_U$$

Remark 10.10 (Literature)**.** *A classical reference for optimal control problems with PDEs using exact/formal Lagrangians and adjoint states is Tröltzsch [160]. A very concise overview of the usage of the adjoint state in optimization and related problems is given by Allaire [3].*

Before we proceed, we briefly recapitulate an important property of the Lagrangian. Let V and Z be two Banach spaces. A Lagrangian is a mapping $L(v, q) : U \times Y \to \mathbb{R}$, where $U \times Y \subset V \times Z$.

Definition 10.11 (Saddle-point)**.** *A point $(u, z) \in U \times Y$ is a saddle point of L on $U \times Y$ when*

$$\forall y \in Y : \quad L(u, y) \leq L(u, z) \leq L(v, z) \quad \forall v \in U.$$

A saddle-point is also known as min-max point. Geometrically one may think of a horse saddle.

Remark 10.12. *One can show that a saddle point yields under certain (strong) conditions a global minimum of u of $J(\cdot)$ in a subspace of U.*

The Lagrangian has the following important property to clear away the constraint (recall the constraint is the PDE!):

Lemma 10.13. *The problem*

$$\inf_{u \in V, -a(u,z)+l(z)=0} \big(J(u) - J(u_h)\big)$$

is equivalent to

$$\inf_{u \in V, -a(u,z)+l(z)=0} = \inf_{u \in V} \sup_{z \in V} L(u, z)$$

Proof. If $a(u, z) + l(z) = 0$ we clearly have $J(u) - J(u_h) = L(u, z)$ for all $z \in V$. If $a(u, z) + l(z) \neq 0$, then $\sup_{z \in V} L(u, z) = +\infty$, which shows the result. □

Remark 10.14. *In the previous lemma, we prefer to work with the infimum (inf) since it is not clear a priori whether the minimum (min) is taken.*

10.4.3 Excursus: Variational principles and Lagrange multipliers in mechanics

Variational principles have been developed in physics and more precisely in **classical mechanics**, e.g., [63, 71]. This section shall serve two purposes:

1. it is another motivation of variational principles;
2. we give a mechanics-based motivation of Lagrange multipliers and constrained optimization problems as they form the background of the current chapter.

10.4.3.1 Variational principles in mechanics One of the first variational problems was designed by Jacob Bernoulli in the year 1696: how does a mass point reach from a point p_1 in the shortest time T another point p_2 under gravitational forces? In consequence, we seek a minimal time T along a curve $u(x)$:

$$\min J(u) = \min(T)$$

with respect to the boundary conditions $u(x_1) = u_1$ and $u(x_2) = u_2$. To obtain the solution $u(x)$, we start from energy conservation, which yields for the kinetic and the potential energies:

$$\frac{mv^2}{2} = mg(u_1 - u),$$

where m is the mass, v the velocity of the mass, g the gravitational force, u_1 the boundary condition, and $u = u(x)$ the sought solution curve. We have the functional:

$$J(u) = T = \int_1^2 \frac{ds}{v} = \int_{x_1}^{x_2} \sqrt{\frac{1 + u'(x)^2}{2g(u_1 - u(x))}}\, dx.$$

Proposition 10.15. *The solution to this problem is the so-called* ***Brachistochrone*** *formulated by Jacob Bernoulli in 1696 [63].*

More generally, we formulate the unconstrained problem:

Formulation 10.16. *Find $u = u(x)$ such that*

$$\min J(u)$$

with

$$J(u) = \int_{x_1}^{x_2} F(u, u', x)\, dx.$$

Here, we assume that the function $F(u, u', x)$ and the boundary values $u(x_1) = u_1$ and $u(x_2) = u_2$ are known.

The idea is that we vary $J(u + \delta u)$ with a small increment δu. Then, we arrive at the **Euler-Lagrange equations**:

Definition 10.17. *The (strong form of the) Euler-Lagrange equations are obtained as stationary point of the functional $J(u)$ and are nothing else, but the PDE in differential form:*

$$\frac{d}{dx}\frac{\partial F(u, u', x)}{\partial u'} = \frac{\partial F(u, u', x)}{\partial u}.$$

Remark 10.18 (Weak form of Euler-Lagrange equations)**.** *An equivalent statement is related to the weak form of the Euler-Lagrange equations in Banach spaces. Here, the functional $J(u)$ is differentiated w.r.t. u into the direction ϕ yielding $J'_u(u)(\phi)$.*

Example 10.19. *To compute the length of a curve $u(x)$ between two points $(x_1, u(x_1))$ and $(x_2, u(x_2))$, the following functional is used:*

$$J(u) = \int_1^2 ds = \int_{x_1}^{x_2} \sqrt{1 + (u')^2}\, dx.$$

This brings us to the question for which function $u(x)$ the functional $J(u)$ atteins a minimum, i.e., measuring the shortest distance between $(x_1, u(x_1))$ and $(x_2, u(x_2))$. In addition, this functional is the starting point to derive the clothesline problem.

Moreover, we identify $F(u, u', x) = \sqrt{1 + (u')^2}$. The Euler-Lagrange equation is then given by

$$\frac{d}{dx}\frac{u'(x)}{\sqrt{1 + (u')^2}} = 0.$$

When no external forces act (right hand side is zero), we can immediately derive the solution:

$$\frac{d}{dx}\frac{u'(x)}{\sqrt{1 + (u')^2}} = 0 \quad \Rightarrow \quad \frac{d}{dx}u'(x) = 0 \quad \Rightarrow \quad u'(x) = const \quad \Rightarrow \quad u(x) = ax + c.$$

The boundary conditions (not specified here) will determine the constants a and c. Of course, the solution, i.e., the shortest distance between two points, is a linear function.

10.4.3.2 Variational principles in mechanics subject to constraints - Lagrange multipliers We come now to the second goal and address a physical interpretation of **Lagrange multipliers**. We motivated the Poisson problem as the clothesline problem in [178]. The derivation based on first principles in physics, namely conservation of potential energy is as follows. Let a clothesline be subject to gravitational forces g. We seek the solution curve $u = u(x)$. A final equilibrium is achieved for minimal potential energy. This condition can be expressed as:

Formulation 10.20 (Minimal potential energy - an unconstrained variational problem)**.** *Find u such that*

$$\min J(u)$$

with

$$J(u) = E_{pot} = \underbrace{\int_1^2 gu\, dm}_{right\ hand\ side} = \underbrace{\rho g \int_{x_1}^{x_2} u\sqrt{1+(u')^2}\, dx}_{left\ hand\ side},$$

where g is the gravity, u the sought solution, dm mass elements, ρ the mass density. By variations δu we obtain solutions $u + \delta u$ and seek the optimal solution such that $J(u)$ is minimal. Of course, the boundary conditions u_1 and u_2 are not varied.

In the following, we formulate a constrained minimization problem. We ask that the length L of the clothesline is fixed:

$$K(u) = L = \int_{x_1}^{x_2} \sqrt{1+(u')^2}\, dx = \text{const.}$$

Formulation 10.21 (A constrained minimization problem)**.** *Find u such that*

$$\min J(u) \quad s.t. \quad K(u) = const.$$

The question is how to address Formulation 10.21 in practice? We explain the derivation in terms of a 1D situation in order to provide a basic understanding as usually done in these lecture notes.

The task is:

$$\min J(x, u(x)) \quad \text{s.t.} \quad K(x, u(x)) = 0. \tag{61}$$

Let us assume for a moment, we can explicitly compute $u = u_K(x)$ from $K(x, u(x)) = 0$ such

$$K(x, u_K(x)) = 0.$$

The minimal value of $J(x, u)$ on the curve $u_K(x)$ can be computed as minimum of $J(x, u_K(x))$. For this reason, we use the first derivative to compute the stationary point. With the help of the chain rule, we obtain:

$$0 = \frac{d}{dx} J(x, u_K) = J'_x(x, u_K) + J'_u(x, u_K)u'_K(x). \tag{62}$$

With this equation, we obtain the solution x_1. The solution u_1 is then obtain from $u_1 = u_K(x_1)$.

Using **Lagrange multipliers**, we avoid the explicit construction of $u_K(x)$ because such an expression is only easy to obtain for simple model problems. To this end, we introduce the variable z as a Lagrange multiplier.

We build the Lagrangian

$$L(x, u, z) = J(x, u) - zK(x, u)$$

and consider the problem:

$$\min L(x, u, z) \quad \text{s.t. } K(x, u) = 0.$$

Again, to find the optimal points, we differentiate w.r.t. to the three solution variables x, u, z:

$$\begin{aligned} J'_x(x, u) - zK'_x(x, u) &= 0 \\ J'_u(x, u) - zK'_u(x, u) &= 0 \\ K(x, u) &= 0 \end{aligned} \tag{63}$$

to obtain a first-order optimality system for determining x, u, z.

Proposition 10.22. *The formulations* (62) *and* (63) *are equivalent.*

Proof. We show (63) yields (62). We assume that we know $u = u_K(x)$, but we do not need the explicit construction of that $u_K(x)$. Then, $K(x,u) = 0$ is equivalent to

$$K(x,u) = u - u_K(x) = 0.$$

We differentiate w.r.t. x and u and insert the resulting expressions into the first two equations in (63):

$$J'_x(x,u) + zu'_K(x) = 0, \tag{64}$$

$$J'_u(x,u) - z = 0. \tag{65}$$

The second condition is nothing else, but

$$z = J'_u(x,u).$$

Here, we easily see that the Lagrange multiplier measures the sensitivity (i.e., the variation) of the solution curve u with respect to the functional $J(x,u)$. Inserting $z = J'_u(x,u)$ into (63) yields (62). The backward direction can be shown in a similar fashion. □

Remark 10.23. *The previous derivation has been done for a 1D problem in which $u(x)$ with $x \in \mathbb{R}$ is unknown. The method can be extended to $\mathbb{R}$ and Banach spaces, the latter one being addressed in Section 10.4.4.*

Remark 10.24. *We emphasize again that the use of Lagrange multipliers seems more complicated, but avoids the explicit construction of $u_K(x)$, which can be cumbersome. This is the main reason of the big success of **adjoint methods**, working with the **adjoint variable** z in physics and numerical optimization. Again, here, the Lagrangian $L(x,u,z)$ is minimized and the solution $u = u(x,z)$ contains a parameter z. This parameter is automatically determined such that the constraint $K(x,u) = 0$ is satisfied. The prize to pay is a higher computational cost since more equations need to be solved using* (63) *in comparison to* (63)*.*

10.4.4 First-order optimality system

We start with the Lagrangian stated in Definition 10.7. As motivated in the previous subsections, we seek minimal points and therefore we look at the first-order necessary conditions. Now we work in Banach spaces rather than $\mathbb{R}$. Differentiation with respect to $u \in V$ and $z \in V$ yields the optimality system:

Proposition 10.25 (Optimality system: Primal and adjoint problems)**.** *Differentiating (see Section 8.7) the Lagrangian in Definition 10.7 yields:*

$$L'_u(u,z)(\phi) = J'_u(u)(\phi) - a'_u(u,z)(\phi) \quad \forall \phi \in V,$$
$$L'_z(u,z)(\psi) = -a'_z(u,z)(\psi) + l'_z(\psi) \quad \forall \psi \in V.$$

*The first equation is called the **adjoint problem** and the second equation is nothing else than our PDE, the so-called **primal problem**. We also observe that the trial and test functions switch in a natural way in the adjoint problem.*

Proof. Trivial with the methods presented in Section 8.7. □

Remark 10.26 (on the notation)**.** *We abuse a bit the standard notation for semi-linear forms. Usually, all linear and nonlinear arguments are distinguished such that $a'_u(u,z)(\phi)$ would read $a'_u(u)(z,\phi)$ because u is nonlinear and z and ϕ are linear. We use in these notes, however, $a'_u(u,z)(\phi)$ in order to emphasize that u and z are the main variables.*

Corollary 10.27 (Primal and adjoint problems in the linear case)**.** *In the linear case, we obtain from the general formulation:*

$$L(\phi,z) = J(\phi) - a(\phi,z)$$
$$L(u,\psi) = -a(u,\psi) + l(\psi).$$

Definition 10.28. *When we discretize both problems using for example a finite element scheme (later more), we define the residuals for $u_h \in V_h$ and $z_h \in V_h$. The primal and adjoint residuals read, respectively:*

$$\begin{aligned}\rho(u_h,\cdot) &= -a'_z(u_h,z)(\cdot) + l'_z(\cdot)\\ \rho^*(z_h,\cdot) &= J'_u(u)(\cdot) - a'_u(u,z_h)(\cdot).\end{aligned}$$

In the linear case:

$$\begin{aligned}\rho(u_h,\cdot) &= -a(u_h,\cdot) + l(\cdot)\\ \rho^*(z_h,\cdot) &= J(\cdot) - a(\cdot,z_h).\end{aligned}$$

Proposition 10.29 (First-order optimality system)**.** *To determine the optimal points $(u,z) \in V \times V$, we set the first-order optimality conditions to zero:*

$$\begin{aligned}0 &= J'_u(u)(\phi) - a'_u(u)(z,\phi)\\ 0 &= -a'_z(u)(z,\psi) + l'_z(\psi).\end{aligned}$$

Here, we easily observe that the primal equation is nothing else than the bilinear form $a(\cdot,\cdot)$ we worked with so far. The adjoint problem is new (well known in optimization though; see the literature remark in Section 10.4.2) and yields ***sensitivity*** *measures z of the primal solution u with respect to the goal functional $J(\cdot)$.*

Example 10.30. *Let $a(u,\phi) = (\nabla u, \nabla\phi)$. Then: $a(u,z) = (\nabla u, \nabla z)$. Then,*

$$a'_u(u,z)(\phi) = (\nabla\phi, \nabla z),$$

and

$$a'_z(u,z)(\psi) = (\nabla u, \nabla\psi).$$

These derivatives are computed with the help of directional derivatives (Gâteaux derivatives) in Banach spaces.

10.4.5 Linear problems and linear goal functionals (Poisson)

We explain our developments in terms of the linear Poisson problem and linear goal functionals.

Formulation 10.31. *Let $f \in L^2(\Omega)$, and we assume that the problem and domain are sufficiently regular such that the trace theorem, e.g., [181]) holds true, i.e., $h \in H^{-\frac{1}{2}}(\Gamma_N)$, and finally $u_D \in H^{\frac{1}{2}}(\Omega)$. Find $u \in \{u_D + V\}$:*

$$a(u,\phi) = l(\phi) \quad \forall \phi \in V,$$

where

$$a(u,\phi) = (\alpha\nabla u, \nabla\phi)$$

and

$$l(\phi) := \int_\Omega f\phi\,dx + \int_{\Gamma_N} g\phi\,ds,$$

and the diffusion coefficient $\alpha := \alpha(x) \in L^\infty(\Omega)$. In this setting $\int_{\Gamma_N} g\phi\,ds$ has to be understood as duality product as for instance in [74]. If $g \in L^2(\Gamma_N)$ then it coincides with the integral.

As previously motivated, the aim is to compute a certain quantity of interest $J(u)$ with a desired accuracy at low computational cost.

Example 10.32. *Examples of goal functionals are mean values, line integration or point values:*

$$J(u) = \int_\Omega u\,dx, \quad J(u) = \int_\Gamma u\,ds, \quad J(u) = \int_\Gamma \partial_n u\,ds, \quad J(u) = u(x_0,y_0,z_0).$$

The first goal functional is simply the mean value of the solution. The third and fourth goal functionals are a priori not well defined. In case of the second functional we know the $\nabla u \in [L^2(\Omega)]^d$. Using the trace theorem, we can deduce that the trace in normal direction belongs to $H^{-\frac{1}{2}}(\partial\Omega)$. This leads to the problem that the second functional is not always well defined. Concerning the third functional, we remind the reader that H^1 functions with dimension $d > 1$, the solution u is not any more continuous and the last evaluation is not well defined, e.g., [32]. If the domain and boundaries are sufficiently regular in 2D, the resulting solution is, however, H^2 regular and thanks to Sobolev embedding theorems (e.g., [44, 59]) also continuous.

Example 10.33. *Let $J(u) = \int_\Omega u\,dx$. Then the Fréchet derivative is given by*

$$J(\phi) = J_u'(u)(\phi) = \int_\Omega \phi\,dx$$

and, of course $J_\lambda'(u)(\psi) \equiv 0$.

The above goal functionals are computed with a numerical method leading to a discrete version $J(u_h)$. Thus the key goal is to control the error $J(e) := J(u) - J(u_h)$ in terms of local residuals, which are computable on each mesh cell $K_i \in \mathcal{T}_h$.

Proposition 10.34 (Adjoint problem)**.** *Based on the optimality system, here Corollary 10.27, we seek the adjoint variable $z \in V$:*

$$a(\phi, z) = J(\phi) \quad \forall \phi \in V. \tag{66}$$

Specifically, the adjoint bilinear form for the Poisson problem is given by

$$a(\phi, z) = (\alpha \nabla \phi, \nabla z).$$

For symmetric problems, the adjoint bilinear form $a(\cdot,\cdot)$ is the same as the original one, but differs for non-symmetric problems like transport for example.

Proof. Apply the first-order necessary condition. □

Remark 10.35. *Existence and uniqueness of this adjoint solution follows by standard arguments provided sufficient regularity of the goal functional and the domain are given. The regularity of $z \in V$ depends on the regularity of the functional J. For $J \in H^{-1}(\Omega)$ it holds $z \in H^1(\Omega)$. Given a more regular functional like the L^2-error $J(\phi) = \|e\|^{-1}(e_h, \phi)$ (where $e := u - u_h$) with $J \in L^2(\Omega)^*$ (denoting the dual space), it holds $z \in H^2(\Omega)$ on suitable domains (convex polygonal or smooth boundary with C^2-parametrization).*

We now work with the techniques as adopted in the numerical analysis of FEM discretizations (see e.g., [178]). Inserting as special test function $\psi := u - u_h \in V$, recall that $u_h \in V_h \subset V$ into (66) yields:

$$a(u - u_h, z) = J(u - u_h),$$

and therefore we have now a representation for the error in the goal functional.

Next, we use the Galerkin orthogonality $a(u - u_h, \psi_h) = 0$ for all $\psi_h \in V_h$, and we obtain:

$$a(u - u_h, z) = a(u - u_h, z) - \underbrace{a(u - u_h, \psi_h)}_{=0} = a(u - u_h, z - \psi_h) = J(u - u_h). \tag{67}$$

The previous step allows us to choose ψ_h in such a way that $z - \psi_h$ can be bounded using interpolation estimates. Indeed, since ψ_h is an arbitrary discrete test function, we can for example use a projection $\psi_h := i_h z \in V_h$ in (67), which is for instance the nodal interpolation.

Definition 10.36 (Error identity)**.** *Choosing $\psi_h := i_h z \in V_h$ in (67) yields:*

$$a(u - u_h, z - i_h z) = J(u - u_h). \tag{68}$$

*Thus the error in the functional $J(u - u_h)$ can be expressed in terms of a residual, that is weighted by **adjoint sensitivity** information $z - i_h z$.*

However, since $z \in V$ is an unknown itself, we cannot yet simply evaluate the error identity because z is only known analytically in very special cases. In general, z is evaluated with the help of a finite element approximation yielding $z_h \in V_h$.

However, this yields another difficulty since we inserted the interpolation $i_h : V \to V_h$ for $z \in V$. When we approximate now z by z_h and use a linear or bilinear approximation $r = 1$: $z_h \in V_h^{(1)}$, then the interpolation i_h does nothing (in fact we interpolate a linear/bilinear function z_h with a linear/bilinear function $i_h z_h$, which is clearly

$$z_h - i_h z_h \equiv 0.$$

For this reason, we need to approximate z_h with a scheme that results in a higher-order representation: here at least something of quadratic order: $z_h \in V_h^{(2)}$.

10.4.6 Nonlinear problems and nonlinear goal functionals

In the nonlinear case, the PDE may be nonlinear (e.g., p-Laplace, nonlinear elasticity, Navier-Stokes) and also the goal functional may be nonlinear, e.g.,

$$J(u) = \int_\Omega u^2\, dx.$$

These nonlinear problems yield a semi-linear form $a(u)(\phi)$ (not bilinear any more), which is nonlinear in the first variable u and linear in the test function ϕ. Both the semi-linear form and the goal functional $J(\cdot)$ are assumed to be (Fréchet) differentiable.

We start from Definition 10.25. Assuming that we discretized both problems using finite elements (for further hints on the adjoint solution see Section 10.5). We suppose that all problems have unique solutions. We define:

Definition 10.37 (Primal and adjoint residuals)**.** *We define the primal and adjoint residuals, respectively:*

$$\begin{aligned}\rho(u_h)(\phi) &= l(\phi) - a(u_h)(\phi) \quad \forall \phi \in V,\\ \rho^*(z_h)(\phi) &= J'_u(u_h)(\phi) - a'_u(u_h)(\phi, z) \quad \forall \phi \in V.\end{aligned}$$

It holds:

Theorem 10.38 ([20])**.** *For the Galerkin approximation of the first-order necessary system Definition 10.25, we have the* ***combined*** *a posteriori error representation:*

$$J(u) - J(u_h) = \eta = \frac{1}{2}\min_{\phi_h \in V_h} \rho(u_h)(z - \phi_h) + \frac{1}{2}\min_{\phi_h \in V_h} \rho^*(z_h)(u - \phi_h) + R.$$

The remainder term is of third order in $J(\cdot)$ and second order in $a(\cdot)(\cdot)$. Thus for linear $a(\cdot)(\cdot)$ and quadratic $J(\cdot)$ the remainder term R vanishes. In practice the remainder term is neglected anyway and assumed to be small. However, in general this assumption should be justified for each nonlinear problem.

Proof. We refer the reader to [20]. □

Corollary 10.39 (Linear problems)**.** *In the case of linear problems the two residuals coincide. Then, it is sufficient to only solve the primal residual:*

$$\begin{aligned}J(u) - J(u_h) = J(u - u_h) &= \underbrace{\min_{\phi_h \in V_h} \rho(u_h)(z - \phi_h)}_{Primal}\\ &= \underbrace{\min_{\phi_h \in V_h} \rho^*(z_h)(u - \phi_h)}_{Adjoint}\\ &= \underbrace{\frac{1}{2}\min_{\phi_h \in V_h} \rho(u_h)(z - \phi_h) + \frac{1}{2}\min_{\phi_h \in V_h} \rho^*(z_h)(u - \phi_h)}_{Combined}\end{aligned}$$

Indeed the errors are exactly the same for linear goal functionals using the primal, adjoint or combined error estimator. The only difference are the resulting meshes. Several examples have been shown in Example 2 in [150].

Proof. It holds for the adjoint problem in the linear case:

$$J(\phi) = a(\phi, z).$$

Then:

$$
\begin{aligned}
J(u-u_h) = J(e) = {} & \underbrace{a(u-u_h, z)}_{\underbrace{= l(z) - a(u_h, z)}_{=\rho(u_h, z)}} \\
= {} & a(e,z) = a(e, z-z_h) = a(e, e^*) = a(u-u_h, e^*) \\
= {} & \underbrace{a(u, e^*)}_{=a(u,z)-a(u,z_h)=\underbrace{J(u) - a(u, z_h)}_{=\rho^*(z_h, u)}} \\
= {} & J(e^*),
\end{aligned}
$$

where $e^* := z - z_h$ and where $a(e,z) = a(e, z-z_h)$ and $a(u-u_h, e^*) = a(u, e^*)$ hold true thanks to Galerkin orthogonality. □

Definition 10.40 (A formal procedure to derive the adjoint problem for nonlinear equations)**.** *We summarize the previous developments. Based on Proposition 10.25, we set-up the adjoint problem as follows:*

1. *Given $a(u)(\phi)$, the residual (i.e., the PDE we want to solve)*
2. *Differentiate w.r.t. $u \in V$ such that*
$$a'_u(u)(\delta u, \phi)$$
with the direction $\delta u \in V$. Info: this object is required for Newton's method as well.
3. *Switch the trial function $\delta u \in V$ and the test function $\phi \in V$ such that:*
$$a'_u(u)(\phi, \delta u)$$
4. *Replace $\delta u \in V$ by the adjoint solution $z \in V$:*
$$a'_u(u)(\phi, z)$$

Remark 10.41. *This procedure also shows that the primal variable $u \in V$ enters the adjoint problem in the nonlinear case. However $u \in V$ is now given data and $z \in V$ is the sought unknown. This also means that in nonlinear problems, that primal solution needs to be stored in order to be accessed when solving the adjoint problem.*

Remark 10.42. *In practice, the adjoint is often not explicitly calculated by hand, but using the last Newton matrix (from the primal problem) and transpose it.*

Example 10.43 (Computing the adjoint: geometrical nonlinear elasticity)**.** *Given:*

$$-\nabla \cdot \sigma(u) = f$$

with $\sigma = 2\mu e(u)$ and $e(u) = \frac{1}{2}(\nabla u + \nabla u^T + \nabla u^T \nabla u)$. We do the steps as explained above:

1. $a(u)(\phi) = (\sigma(u), \nabla\phi) - (f, \phi) = \mu(\nabla u + \nabla u^T + \nabla u^T \nabla u, \nabla \phi) - (f, \phi)$
2. $a'(u)(\delta u, \phi) = (\sigma'(u)(\delta u), \nabla\phi) = \mu(\nabla\delta u + \nabla\delta u^T + \nabla\delta u^T \nabla u + \nabla u^T \nabla \delta u, \nabla\phi)$
3. $a'(u)(\phi, \delta u) = (\sigma'(u)(\phi), \nabla\delta u) = \mu(\nabla\phi + \nabla\phi^T + \nabla\phi^T \nabla u + \nabla u^T \nabla\phi, \nabla\delta u)$
4. $a'(u)(\phi, z) = (\sigma'(u)(\phi), \nabla z) = \mu(\nabla\phi + \nabla\phi^T + \nabla\phi^T \nabla u + \nabla u^T \nabla\phi, z)$

Example 10.44 (Computing the adjoint: obstacle problem with simple penalization)**.** *Given (formulation not complete!):*

$$-\Delta u - \gamma[g-u]^+ = f, \quad u \geq g$$

We do the steps as explained above:

1. $a(u)(\phi) = (\nabla u, \nabla\phi) - \gamma([g-u]^+, \phi) - (f, \phi)$
2. $a'(u)(\delta u, \phi) = (\nabla\delta u, \nabla\phi) - \gamma(\delta u, \phi)_{B(u)}$
3. $a'(u)(\phi, \delta u) = (\nabla\phi, \nabla\delta u) - \gamma(\phi, \delta u)_{B(u)}$
4. $a'(u)(\phi, z) = (\nabla\phi, \nabla z) - \gamma(\phi, z)_{B(u)}$

with $B(u) = \{x \in \Omega |\ g(x) > u(x)\}$.

10.5 Approximation of the adjoint solution for the primal estimator $\rho(u_h)(\cdot)$

In order to obtain a computable error representation, $z \in V$ is approximated through a finite element function $z_h \in V_h$, that is obtained from solving a discrete adjoint problem:

Formulation 10.45 (Discrete adjoint problem for linear problems)**.**

$$a(\psi_h, z_h) = J(\psi_h) \quad \forall \psi_h \in V_h. \tag{69}$$

Then the primal part of the error estimator reads:

$$a(u - u_h, z_h - i_h z_h) = \rho(u_h)(z_h - i_h z_h) \approx J(u) - J(u_h). \tag{70}$$

The difficulty is that if we compute the adjoint problem with the same polynomial degree as the primal problem, then $z_h - i_h z_h \equiv 0$, and thus the whole error identity defined in (70) would vanish, i.e., $J(u) - J(u_h) \equiv 0$. This is clear from a theoretical standpoint and can be easily verified in numerical computations.

In fact normally an interpolation operator i_h interpolates from infinite dimensional spaces V into finite-dimensional spaces V_h or from higher-order spaces into low-order spaces.

Thus:

$$\begin{aligned}
&i_h : V \to V_h^{(1)} && \text{so far...}\\
&i_h : V_h^{(1)} \to V_h^{(1)} && \text{first choice, but trivial solution, useless}\\
&i_h : V_h^{(2)} \to V_h^{(1)} && \text{useful choice with nontrivial solution}
\end{aligned}$$

From these considerations it is also clear that

$$i_h : V_h^{(1)} \to V_h^{(2)}$$

will even be worse (this would arise if we approximate the primal solution with $u_h \in V_h^{(2)}$ and $z_h \in V_h^{(1)}$).

In summary:

- the adjoint solution needs to be computed either with a global higher-order approximation (using a higher order finite element of degree $r+1$ when the primal problem is approximated with degree r),
- or a solution on a finer mesh,
- or local higher-order approximations using a patch-wise higher-order interpolation [17, 20, 150].

Clearly, the last possibility is the cheapest from the computational cost point of view, but needs some efforts to be implemented. For the convenience of the reader we tacitly work with a global higher-order approximation in the rest of this chapter.

We finally end up with the primal error estimator:

Definition 10.46 (Primal error estimator)**.** *The primal error estimator is given by:*

$$a(u - u_h, z_h^{(r+1)} - i_h z_h^{(r+1)}) = \rho(u_h)(z_h^{(r+1)} - i_h z_h^{(r+1)}) =: \eta \approx J(u) - J(u_h).$$

10.6 Approximation of the primal solution for the adjoint estimator $\rho^*(z_h)(\cdot)$

To evaluate the adjoint estimator $\rho(z_h)(\cdot)$, we need to construct

$$u - i_h u$$

with $i_h : V \to V_h$ for $u \in V$ and $u_h \in V_h$. Here we encounter the opposite problem to the previous section. We need to solve the primal problem with higher accuracy using polynomials of degree $r+1$ in order to construct a useful interpolation, yielding

$$u - i_h u \neq 0 \quad \text{a.e.}$$

Then:

$$J(u) - J(u_h) \approx \eta := \rho^*(z_h)(u_h^{(r+1)} - i_h u_h^{(r+1)}).$$

10.7 Measuring the quality of the error estimator η

As quality measure how well the estimator approximates the true error, we use the effectivity index I_{eff}:

Definition 10.47 (Effectivity index)**.** *The effectivity index is defined as:*

$$I_{eff} := I_{eff}(u_h, z_h) = \Big|\frac{\eta}{J(u) - J(u_h)}\Big|. \tag{71}$$

Problems with good I_{eff} satisfy asymptotically $I_{eff} \to 1$ for $h \to 0$. We say that

- *for $I_{eff} > 1$, we have an* ***over estimation*** *of the error,*
- *for $I_{eff} < 1$, we have an* ***under estimation*** *of the error.*

10.8 Localization techniques

In the previous sections, we derived an error approximation η with the help of duality arguments on the entire domain Ω. In order to use the error estimator for mesh refinement we need to localize the error estimator η to single mesh elements $K_j \in \mathcal{T}_h$ or degrees of freedom (DoF) i. We present two techniques:

- A classical procedure using integration by parts results in an element-based indicator η_K;
- A more recent way by employing a partition-of-unity (PU) yields PU-DoF-based indicators η_i.

We recall that the primal estimator starts with $z \in V$ from:

$$\rho(u_h)(z - i_h z) = a(u - u_h, z - i_h z) = a(u, z - i_h z) - a(u_h, z - i_h z) = l(z - i_h z) - a(u_h, z - i_h z),$$

and the adjoint estimator

$$\rho^*(z_h)(u - i_h u) = J(u - i_h u) - a(u - i_h u, z).$$

According to Theorem 10.38, we have for linear problems

$$\begin{aligned} J(u) - J(u_h) &= \frac{1}{2}\rho + \frac{1}{2}\rho^* + R \\ &= \frac{1}{2}\big(l(z - i_h z) - a(u_h, z - i_h z)\big) + \frac{1}{2}\big(J'(u - i_h u) - a'(u - i_h u, z_h)\big) + R \end{aligned}$$

where R is the remainder term. Furthermore on the discrete level, we have (for linear problems):

$$J(u) - J(u_h) \approx \eta := \frac{1}{2}\big(l(z_h - i_h z_h) - a(u_h, z_h - i_h z_h)\big) + \frac{1}{2}\big(J(u_h - i_h u_h) - a(u_h - i_h u_h, z_h)\big).$$

We know that the discrete solutions z_h in the first part and u_h in the second part have to be understood computed in terms of higher-order approximations $r+1$.

In the following we localize these error estimators on a single element $K_i \in \mathcal{T}_h$. Here, the influence of neighboring elements $K_j, j \neq i$ is important [40]. In order to achieve such an influence, we consider the error estimator on each cell and then either integrate back into the strong form (the classical way) or keep the weak form and introduce a partition-of-unity (a more recent way). Traditionally, there is also another way with weak form, proposed in [31], which has been analyzed theoretically in [150], but which we do not follow in these notes further.

In the following both localization techniques we present, we start from:

$$J(u-u_h) = l(z_h - i_h z_h) - a(u_h, z_h - i_h z_h)$$

For the Poisson problem, we can specify as follows:

$$J(u-u_h) = (f, z_h - i_h z_h) - (\nabla u_h, \nabla(z_h - i_h z_h))$$

The next step will be to localize both terms either on a cell $K \in \mathcal{T}_h$ (Section 10.8.1) or on a degree of freedom (Section 10.8.3).

10.8.1 The classical way of error localization of the primal estimator for linear problems

In the classical way, the error identity (68) is treated with integration by parts on every mesh element $K \in \mathcal{T}_h$, which yields:

Proposition 10.48. *It holds:*

$$J(u-u_h) \approx \eta = \sum_{K\in\mathcal{T}_h} (f + \nabla\cdot(\alpha\nabla u_h), z_h - i_h z_h)_K + (\alpha\partial_n u_h, z_h - i_h z_h)_{\partial K} \tag{72}$$

Proof. Let $\alpha = 1$ for simplicity. We start from

$$J(u-u_h) = (f, z_h - i_h z_h) - (\nabla u_h, \nabla(z_h - i_h z_h))$$

and obtain further

$$\begin{aligned} J(u-u_h) &= (f, z_h - i_h z_h) - (\nabla u_h, \nabla(z_h - i_h z_h)) \\ &= \sum_K (f, z_h - i_h z_h)_K - (\nabla u_h, \nabla(z_h - i_h z_h))_K \\ &= \sum_K (f, z_h - i_h z_h)_K + (\Delta u_h, z_h - i_h z_h)_K - (\partial_n u_h, z_h - i_h z_h)_{\partial K} \\ &= \sum_K (f + \Delta u_h, z_h - i_h z_h)_K - \frac{1}{2}([\partial_n u_h]_K, z_h - i_h z_h)_{\partial K} \end{aligned}$$

with $[\partial_n u_h] := [\partial_n u_h]_K = \partial_n u_h|_K + \partial_{n'} u_h|_{K'}$ where K' is a neighbor cell of K. On the outer (Dirichlet) boundary we set $[\partial_n u_h]_{\partial\Omega} = 2\partial_n u_h$. □

With the notation from the proof, we can define local residuals:

$$\begin{aligned} R_T &:= f + \Delta u_h, \\ r_{\partial K} &:= -[\partial_n u_h] \end{aligned}$$

Here, R_T are **element residuals** that measure the 'correctness' of the PDE. The $r_{\partial K}$ are so-called **face residuals** that compute the jumps over element faces and consequently measure the smoothness of the discrete solution u_h.

Further estimates are obtained as follows:

$$\begin{aligned} |J(u-u_h)| &\le \Big|\sum_{K\in\mathcal{T}_h} ...\Big| \\ &\le \sum_{K\in\mathcal{T}_h} |...| \end{aligned}$$

where we assume $z_h - \phi_h = 0$ on $\partial\Omega$. With Cauchy-Schwarz we further obtain:

$$
\begin{aligned}
|J(u-u_h)| \le \eta &= \sum_K \Big[\|f + \Delta u_h\|_K \|z_h - i_h z_h\|_K + \frac{1}{2} \|[\partial_n u_h]\|_{\partial K \setminus \partial\Omega} \|z_h - i_h z_h\|_{\partial K} \Big] \\
&= \sum_K \rho_K(u_h)\omega_K(z_h) + \rho_{\partial K}(u_h)\omega_{\partial K}(z_h).
\end{aligned}
$$

In summary we showed:

Proposition 10.49. *We have:*

$$
|J(u) - J(u_h)| \le \eta := \sum_{K \in \mathcal{T}_h} \rho_K \omega_K, \tag{73}
$$

with

$$
\rho_K := \|f + \nabla \cdot (\alpha \nabla u_h)\|_K + \frac{1}{2} h_K^{-\frac{1}{2}} \|[\alpha \partial_n u_h]\|_{\partial K}, \tag{74}
$$

$$
\omega_K := \|z - i_h z\|_K + h_k^{\frac{1}{2}} \|z - i_h z\|_{\partial K}, \tag{75}
$$

where by $[\alpha \partial_n u_h]$ we denote the jump of the u_h derivative in normal direction. The residual part ρ_K only contains the discrete solution u_h and the problem data. On Dirichlet boundaries Γ_D, we set $[\alpha \partial_n u_h] = 0$ and on the Neumann part we evaluate $\alpha \partial_n u_h = g_N$. Of course, we implicitly assume here that $g_N \in L^2(\Gamma_N)$ such that these terms are well-defined.

Remark 10.50. *In practice, this primal error estimator needs to be evaluated in the dual space. Here, we proceed as follows:*

- *Prolongate the primal solution u_h into the dual space;*
- *Next, we compute the interpolation $i_h z_h^{(r+1)} \in Q_r$ w.r.t. to the primal space;*
- *Then, we compute $z_h^{(r+1)} - i_h z_h^{(r+1)}$ (here, $i_h z_h^{(r+1)}$ is prolongated to Q_{r+1} in order to compute the difference);*
- *Evaluate the duality product $\langle \cdot, \cdot \rangle$ and face terms.*

Remark 10.51. *When $V_h = V_h^{(1)}$, then $\nabla \cdot \nabla u_h \equiv 0$. This also demonstrates heuristically that face terms are important.*

10.8.2 The classical way for the combined estimator

The combined estimator reads:

Proposition 10.52. *It holds:*

$$
J(u) - J(u_h) \approx \sum_{K \in \mathcal{T}_h} \frac{1}{2}\eta_K + \frac{1}{2}\eta_K^*
$$

with

$$
\begin{aligned}
\eta_K &= \big(\langle f + \nabla \cdot (\alpha \nabla u_h), z_h - i_h z_h \rangle_K + \int_{\partial K} \alpha \partial_n u_h \cdot (z_h - i_h z_h)\, ds \big) \\
\eta_K^* &= \big(J(u_h - i_h u_h) - (\int_K ... + \int_{\partial K} ...) \big)
\end{aligned}
$$

10.8.3 A variational primal-based error estimator with PU localization

An alternative way is a DoF-based estimator, which is the first difference to before. The second difference to the classical approach is that we continue to work in the variational form and do not integrate back into the strong form. Such an estimator has been developed and analyzed in [150]. This idea combines the simplicity of the approach proposed in [31] (as it is given in terms of variational residuals), which makes it particularly interesting for coupled and nonlinear PDE systems (see further comments below). Variational localizations are useful for nonlinear and coupled problems as we do not need to derive the strong form.

To this end, we need to introduce a partition-of-unity (PU), which can be realized in terms of another finite element function. The procedure is therefore easy to realize in existing codes.

Definition 10.53 (PU - partition-of-unity)**.** *The PU is given by:*

$$V_{PU} := \{\psi_1, \ldots, \psi_M\}$$

with $dim(V_{PU}) = M$. The PU has the property

$$\sum_{i=1}^{M} \psi_i \equiv 1.$$

Remark 10.54. *The PU can be simply chosen as the lowest order finite element space with linear or bilinear elements, i.e.,*

$$V_{PU} = V_h^{(1)}.$$

To understand the idea, we recall that in the classical error estimator the face terms are essential since they gather information from neighboring cells. When we work with the variational form, no integration by parts (fortunately!) is necessary. Therefore, the information of the neighboring cells is missing. Using the PU, we touch different cells per PU-node and consequently we gather know information from neighboring cells. Therefore, the PU serves as localization technique.

In the following, we now describe how the PU enters into the global error identity (68):

Proposition 10.55 (Primal error estimator)**.** *For the finite element approximation of Formulation 10.31, we have the a posteriori error estimate:*

$$|J(u) - J(u_h)| \le \eta := \sum_{i=1}^{M} |\eta_i|, \tag{76}$$

where

$$\eta_i = a(u - u_h, (z - i_h z)\psi_i) = l((z - i_h z)\psi_i) - a(u_h, (z - i_h z)\psi_i),$$

and more specifically for the Poisson problem:

$$\eta_i = \Big\{\langle f, (z - i_h z)\psi_i\rangle - (\alpha \nabla u_h, \nabla(z - i_h z)\psi_i)\Big\}. \tag{77}$$

10.8.4 PU localization for the combined estimator

Proposition 10.56 (The combined primal-dual error estimator)**.** *The combined estimator reads:*

$$|J(u) - J(u_h)| \le \eta := \sum_{i=1}^{M} \frac{1}{2}|\eta_i| + \frac{1}{2}|\eta_i^*|$$

with

$$\eta_i = l((z_h - i_h z_h)\psi_i) - a((u, z_h - i_h z_h)\psi_i)$$
$$\eta_i^* = J_u'((u_h - i_h u_h)\psi_i) - a_u'((u_h - i_h u_h)\psi_i, z)$$

10.9 Comments to adjoint-based error estimation

Adjoint-based error estimation allows to measure precisely at a low computational cost specific functionals of interest $J(u)$. However, the prize to pay is:

- We must compute a second solution $z \in V$.
- This second solution inside the primal estimator must be of higher order, which means more computational cost in comparison to the primal problem.
- For the full error estimator in total we need to compute four problems.
- From a theoretical point of view, we cannot proof convergence of the adaptive scheme for general goal functionals.

For nonlinear problems, one has to say that the primal problem is subject to nonlinear iterations, but the adjoint problem is always a linearized problem. Here, the computational cost may become less significant of computing an additional adjoint problem. Nonetheless, there is no free lunch.

10.10 Mesh refinement strategies

We have now on each element $K_j \in \mathcal{T}_h$ or each PU-DoF i an error value. It remains to set-up a strategy that tells us which elements shall be refined to enhance the accuracy in terms of the goal functional $J(\cdot)$.

Let an error tolerance (TOL) be given. Mesh adaption is realized using extracted local error indicators from the a posteriori error estimate on the mesh T_h. A cell-wise assembling reads:

$$|J(u) - J(u_h)| \leq \eta := \sum_{K \in T_h} \eta_K \quad \text{for all cells } K \in T_h.$$

Alternatively, the PU allows for a DoF-wise assembling:

$$|J(u) - J(u_h)| \leq \eta := \sum_i \eta_i \quad \text{for all DoFs } i \text{ of the PU.}$$

This information is used to adapt the mesh using the following strategy:

1. Compute the primal solution u_h and the adjoint solution u_h on the present mesh T_h.
2. Determine the cell indicator η_K at each cell K.
   Alternatively, determine the DoF-indicator η_i at each PU-DoF i.
3. Compute the sum of all indicators $\eta := \sum_{K \in T_h} \eta_K$.
   Alternatively, $\eta := \sum_i \eta_i$.
4. Check, if the stopping criterion is satisfied: $|J(u) - J(u_h)| \leq \eta \leq TOL$, then accept u_h within the tolerance TOL. Otherwise, proceed to the following step.
5. Mark all cells K_i that have values η_{K_i} above the average $\frac{\alpha\eta}{N}$ (where N denotes the total number of cells of the mesh $\mathbb{T}_h$ and $\alpha \approx 1$).
   Alternatively, all PU-DoFs are marked that are above, say, the average $\frac{\alpha\eta}{N}$.

Other mesh adaption strategies are discussed in the literature [17, 20]. For instance:

- Refining/coarsening a fixed fraction of elements. Here all elements are ordered with respect to their error values:
  $$\eta_1 \geq \eta_2 \geq \ldots \geq \eta_N.$$
  Then, for instance 30% of all elements are refined and for instance 2% of all cells are coarsened.
- Refining/coarsening according to a reduction of the error estimate (also known as bulk criterion or Dörfler marking [54]). Here, the error values are summed up such that a prescribed fraction of the total error is reduced. All elements that contribute to this fraction are refined.

Remark 10.57. *We emphasize that the tolerance TOL should be well above the tolerances of the numerical solvers. Just recall $TOL = 0.01$ would mean that the goal functional is measured up to a tolerance of 1%.*

When the DoF-based estimator is adopted, the error indicators η_i are node-wise contributions of the error. Mesh adaptivity can be carried out in two ways:

- in a node-wise fashion: if a node i is picked for refinement, all elements touching this node will be refined;
- alternatively, one could also first assemble element-wise for each $K \in \mathcal{T}_h$ indicators by summing up all indicators belonging to nodes of this element and then carry out adaptivity in the usual element-wise way.

On adaptive meshes with hanging nodes, the evaluation of the PU indicator is straightforward: First, the PU is assembled in (77) employing the basis functions $\psi_i \in V_{PU}$ for $i = 1, \dots, M$. In a second step, the contributions belonging to hanging nodes are condensed in the usual way by distribution to the neighboring indicators.

10.10.1 How to refine marked cells

It remains to explain how marked cells are finally refined.

10.10.1.1 Quads and hexs Using quadrilateral or hexahedral meshes, simply bisection can be used. Here, a cell is cut in the middle and split into 4 (in 2d) or 8 (in 3d) sub-elements. When the neighboring cell is not refined, we end up with so-called **hanging nodes**. These are degrees of freedom on the refined cells, but on the coarse neighboring cells, these nodes lie on the middle point of faces or edges and do not represent true degrees of freedom. Their values are obtained by interpolation of the neighboring DoFs. Consequently, conditions on the geometry are weakened in the presence of hanging nodes. For more details, we refer to Carey and Oden [38].

10.10.1.2 Triangles and prims For triangles or prisms, we have various possibilities how to split the elements into sub-elements. Here, common ways are red and green refinement strategies. Here a strategy is to use red refinement and apply green refinement when hanging nodes would occur in neighboring elements.

10.10.1.3 Conditions on the geometry While refining the mesh locally for problems in $n \geq 2$, we need to take care that the minimum angle condition (see e.g., [32]) is fulfilled.

10.10.2 Convergence of adaptive algorithms

The first convergence result of an adaptive algorithm was shown in [54] for the Poisson problem. The convergence of adaptive algorithms of generalized problems is subject to current research. Axioms of adaptivity have been recently formulated in [39].

10.11 Goal-oriented a posteriori (primal) error estimator for phase-field fracture

The primal error estimator ρ (thus neglecting ρ^*) can be stated as:

Proposition 10.58. *For the finite element approximation of the phase-field model with simple penalization with the discrete solution $U_h := (u_h, \varphi_h) \in \{u_D^h + V_h\} \times W_h$ we have the a posteriori error estimate:*

$$
\begin{aligned}
|J(U) - J(U_h)| &\leq \sum_i^N |\eta_i| \\
&= \sum_i^N \Big| \Big(-\big((1-\kappa)\varphi_h^2 - \kappa\big)\, \sigma^+(u_h), e(w_h)\Big) \\
&\quad - (\sigma^-(u_h), e(w_h)) \\
&\quad - (1-\kappa)(\varphi_h\, \sigma^+(u_h) : e(u_h), \psi_h) \\
&\quad - 2(\varphi_h\, p\, \nabla \cdot u_h, \psi_h) \\
&\quad - G_c\Big(-\frac{1}{\varepsilon}(1-\varphi_h, \psi_h) + \varepsilon(\nabla\varphi_h, \nabla\psi_h)\Big) \\
&\quad + R(\gamma, \varphi_h, \varphi_h^{n-1})\Big|
\end{aligned}
$$

where the weighting functions are defined as

$$
\begin{aligned}
w_h &:= (w_{2h}^{(2)} - z_h^u)\chi_h^i, \\
\psi_h &:= (\psi_{2h}^{(2)} - z_h^\varphi)\chi_h^i.
\end{aligned}
$$

The remainder term is defined as

$$
R(\gamma, \varphi_h, \varphi_h^{n-1}) := (\gamma[\varphi_h - \varphi_h^{n-1}]^+, \psi_h).
$$

The first factors $w_{2h}^{(2)} - z_h^u$ and $\psi_{2h}^{(2)} - z_h^\varphi$ of the weights are standard [20]. Here, $w_{2h}^{(2)}$ is a higher-order finite element approximation (i.e., Q_2^c) of the dual solution z^u, respectively for $\psi_{2h}^{(2)}$ and z^φ. The discrete dual solution $Z_h := \{z_h^u, z_h^\varphi\}$ is obtained by solving the corresponding dual problem. Of course, the dual problem is costly to solve. However in nonlinear problems, the (linearized) dual problem needs to be solved only once whereas several iterations are necessary for solving the primal problem. The second function χ_h^i is the novel PU-function.

Proof. From [20], we know the general error representation for the primal estimator:

$$
\begin{aligned}
J(U) - J(U_h) = B(Z - i_h Z) - A(U_h)(Z - i_h Z) \\
+ R^2(U - U_h, Z - Z_h),
\end{aligned}
$$

where i_h denotes an interpolation operator from $V \times W$ to $V_h \times W_h$. Moreover, R^2 denotes a remainder term that is quadratic in the error. The functional and semilinear forms are defined as

$$
\begin{aligned}
&B(Z - i_h Z) = -(\tilde\varphi^2 p, \nabla\cdot(z^u - i_h z^u)), \\
&A(U_h)(Z - i_h Z) \\
&\quad = \Big(\big((1-\kappa)\tilde\varphi_h^2 + \kappa\big)\, \sigma^+(u_h), e(z^u - i_h z^u)\Big) \\
&\qquad + (\sigma^-(u_h), e(z^u - i_h z^u)) \\
&\qquad + (1-\kappa)(\varphi_h\, \sigma^+(u_h) : e(u_h), z^\varphi - i_h z^\varphi) \\
&\qquad + G_c\Big(-\frac{1}{\varepsilon}(1-\varphi_h, z^\varphi - i_h z^\varphi) + \varepsilon(\nabla\varphi_h, \nabla(z^\varphi - i_h z^\varphi))\Big) \\
&\qquad + (\gamma[\varphi_h - \varphi_h^{n-1}]^+, z^\varphi - i_h z^\varphi).
\end{aligned}
$$

Taking the absolute value yields:

$$
\begin{aligned}
|J(U) - J(U_h)| \leq\ & |B(Z - i_h Z) - A(U_h)(Z - i_h Z)| \\
& + |R^2(U - U_h, Z - Z_h)|.
\end{aligned}
$$

Neglecting the remainder term and introducing the PU χ_h^i and summing over all degrees of freedom $i = 1, \dots, N$ brings us to:

$$
\begin{aligned}
& |J(U) - J(U_h)| \\
& \leq \sum_i^N |B((Z - i_h Z)\chi_h^i) - A(U_h)((Z - i_h Z)\chi_h^i)|.
\end{aligned}
$$

Inserting the definitions of $B(Z - i_h Z)$ and $A(U_h)(Z - i_h Z)$ and the approximation of the dual weights $Z - i_h Z$ by

$$
\begin{aligned}
z^u - i_h z^u &\approx w_{2h}^{(2)} - z_h^u, \\
z^\varphi - i_h z^\varphi &\approx \psi_{2h}^{(2)} - \psi_h^u,
\end{aligned}
$$

and using the short notations $w := (w_{2h}^{(2)} - z_h^u)\chi_h^i$ and $\psi := (\psi_{2h}^{(2)} - z_h^\varphi)\chi_h^i$ yields the statement. Q.E.D. □

10.12 Mesh refinement strategies for quasi-stationary (incremental) problems

In this section, two strategies for local mesh adaptivity on the basis of the previous error estimators are described. The key ingredient for mesh refinement is based on the error bound given by the total indicator:

$$
\eta := \sum_{i=1}^N |\eta_i|,
$$

in which local error indicators η_i are obtained by using Proposition 10.58. In the following, one strategy applies to stationary problems with a fixed fracture whereas the second mesh adaptation strategy can be used for quasi-stationary problems with propagating fractures.

Our philosophy of goal-oriented a posteriori error estimation for quasi-stationary problems follows [146, 147]. Rather than computing the dual problem backward in time, the time-evolving problems are split into a sequence of stationary problems (see again [145]). This means that time-dependent problems are discretized in time with a backward Euler scheme. In [146], Lemma 6.2, and Theorem 6.1, it has been shown that in the case of the quasi-stationary Prandtl-Reuss model, the error in each time/load step can accumulate at most linearly and does not become dominant, which is the argument to neglect a full backward-time dual problem. Specifically, meshes are kept fixed during a Newton iteration and are only refined during time steps. Therefore, the global spatial error is approximated through a sequence of stationary problems.

In more detail, for the loading steps $t = 0, 1, 2, \dots$, we perform at each t:

1 Compute the primal solution U_h on the current mesh $\mathbb{T}_h$;

2 Compute once the dual solution Z_h on the current mesh $\mathbb{T}_h$ using a higher-order method while taking the last Newton matrix of the primal problem;

3 Evaluate the error estimator (Proposition 10.58 and determine the indicator η_i at each PU-nodal point.

4 Compute the sum of all indicators $\eta := \sum_i \eta_i$.

5 Check, if the stopping criterion is satisfied: $|J(U) - J(U_h)| \leq \eta \leq TOL$, then accept U_h within the tolerance TOL. Otherwise, proceed to the following step.

6 Mark all nodes i that belong to PU-nodal indicators η_i above the average $\frac{\alpha\eta}{N}$ (where N denotes the total number of degrees of freedom of the PU and $\alpha \approx 1$). All cells that touch this node will then be refined.

7 Adapt the mesh.

8 Increment t and go to Step 1.

The marking strategy in Step 6 can be augmented with phase-field based refinement:

Remark 10.59. *In phase-field based refinement, all cells are refined in which $\eta_i := \varphi < c$ (for example $c = 0.4$). This is important since the fracture propagates through the domain and too high parameter fluctuations (recall that ε is coupled to h via $h = o(\varepsilon)$ which can be realized by choosing, for instance, $\varepsilon = ch^l$, $0 < l \leq 1$) should be avoided in the near fracture region. In coarse cells, ε remains large and influences the solution of the phase-field equation. A remedy has been presented in [82], where a predictor-corrector scheme has been applied in order to ensure the 'correct' size of $\varepsilon > h$ at each time step. Since we deal with a quasi-static problem, we do not want to refine the mesh in each loading step in order to keep the total number of cells reasonable. We might fix the finest mesh level that should not be exceeded.*

Rather than fixing the finest mesh level in order to keep the computational cost reasonable, another strategy of keeping the number of cells constant is based on the following strategy:

Remark 10.60. *Order all nodes according to their size η_i. A fixed portion of nodes (i.e., their touching cells) with the smallest contributions to the total indicator η is marked to be deleted. In the second step, the nodes (i.e., cells) with the largest η_i are refined such that the desired number of cells N_{max} is approximately achieved. This strategy has been successfully applied for Prandtl-Reuss models in perfect plasticity [145, 147].*

10.13 Excursus IV: DWR for the obstacle problem using simple penalization

We continue our investigations from the Sections 5.3 and 8.13 for the obstacle problem and develop a simple a posteriori DWR error estimator based on the primal formulation.

10.13.1 Problem statement

Please see Section 8.13.1.

10.13.2 A posteriori PU-DWR error estimator (primal versions)

Proposition 10.61 (DWR classical Poisson)**.** *For the finite element approximation of the classical Poisson problem, see Section 8.13.1, we have the a posteriori error estimate*

$$|J(u) - J(u_h)| \leq \eta(u_h) := \sum_{i=1}^{N} |\eta_i|$$

with

$$\eta_i = \langle f, (z - i_h z)\psi_i\rangle - (\alpha\nabla u_h, \nabla(z - i_h z)\psi_i). \tag{78}$$

with $\alpha = 1$ in this example.

Proposition 10.62 (DWR obstacle)**.** *For the finite element approximation of the obstacle problem, see Section 8.13.1, we have the a posteriori error estimate*

$$|J(u) - J(u_h)| \leq \eta(u_h) := \sum_{i=1}^{N} |\eta_i|$$

with

$$\eta_i = \langle f, (z - i_h z)\psi_i\rangle - (\alpha\nabla u_h, \nabla(z - i_h z)\psi_i) + (\gamma[g - u]^+, (z - i_h z)\psi_i) \tag{79}$$

with $\alpha = 1$ in this example and the penalization parameter as discussed in Section 8.13.

Proposition 10.63 (DWR obstacle, simplified version)**.** *For the finite element approximation of the obstacle problem, see Section 8.13.1, we have the a posteriori error estimate*

$$|J(u) - J(u_h)| \leq \eta(u_h) := \sum_{i=1}^{N} |\eta_i|$$

with the simplified version of local error indicators:

$$\eta_i = \langle f, (z - i_h z)\psi_i\rangle - (\alpha \nabla u_h, \nabla(z - i_h z)\psi_i) \tag{80}$$

with $\alpha = 1$ in this example.

10.13.3 Numerical results and discussion

Using the implemented setups from Section 8.13, we perform several tests using the previously stated goal-oriented PU-DWR error estimates. As goal functional, we choose:

$$J(u) = u(0.5, 0.5).$$

We are aware of the fact that point value evaluation are not well-posed in settings with dimension > 1 - nevertheless we do it. The material coefficient is $\alpha = 1$. For the obstacle problem, Config. $2 - 3$, the penalization parameter is chosen as $\gamma = \frac{\bar{\gamma}}{h^2}$ with $\bar{\gamma} = 0.1$, and the obstacle function is $g = -0.01$. For Config. $4 - 5$, we do not weight with the mesh size, and rather employ $\gamma = \bar{\gamma} = 100$.

On a sufficiently uniformly refined mesh, we computed the following reference values:

```
Config 1: u(0.5,0.5) = -7.3671574726468209e-02
          with 263169 DoFs (Ref. 9), classical Poisson
Config. 2-3: u(0.5,0.5) = -1.0305175780691633e-02
          with 66049  DoFs (Ref. 8), obstacle
Config. 4-5: u(0.5,0.5) = -1.9470939645319634e-02
          with 263169 DoFs (Ref. 9), obstacle without 1/h^2
```

10.13.3.1 Goals of these computations

The goals of our computations are to observe:

- the true error $J(u) - J(u_h)$;
- the estimator η;
- the effectivity index I_{eff} defined in (71).

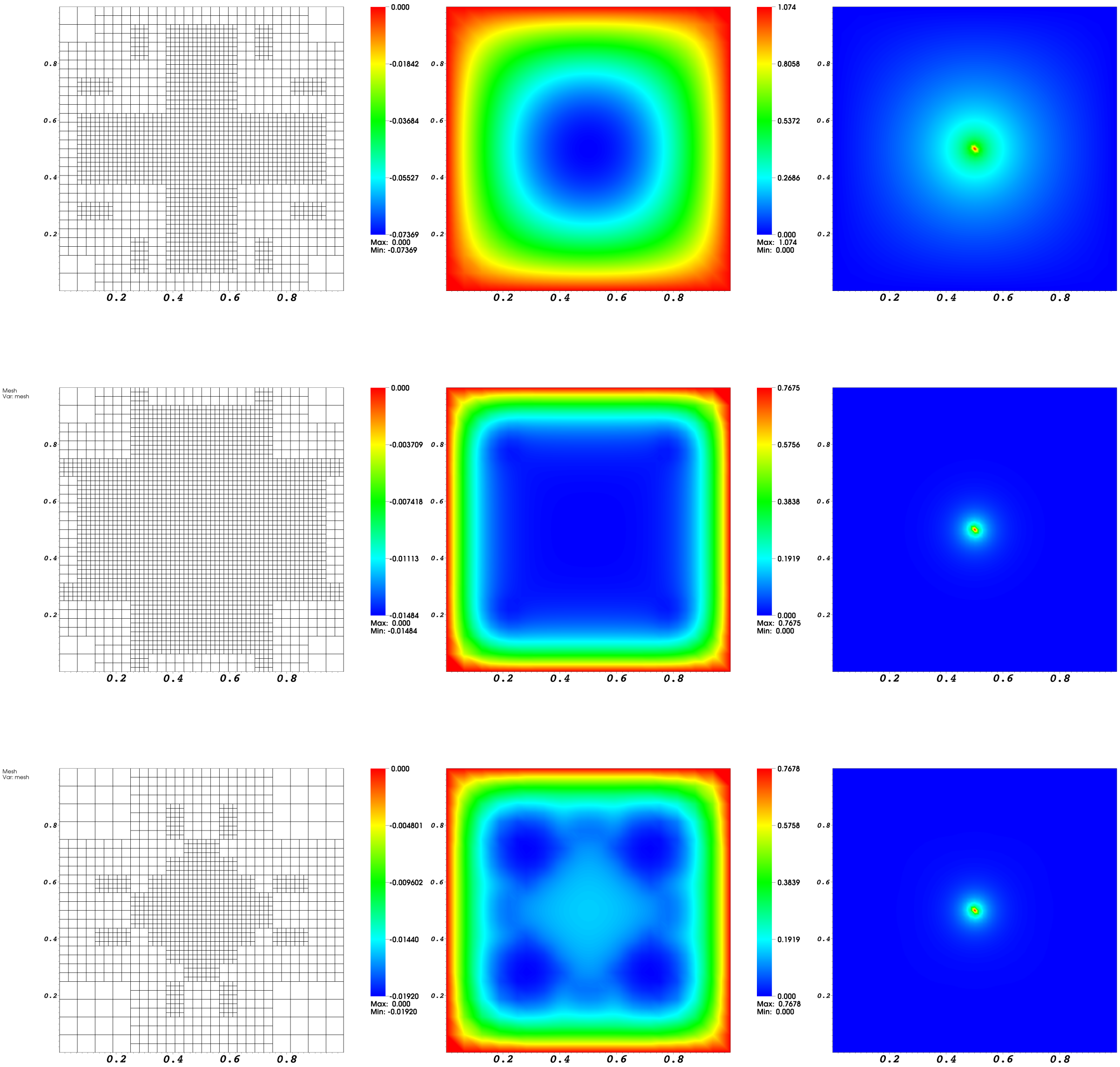

Figure 33: Config. $1-3$: Left column: adaptively refined meshes. Middle column: primal solutions. Right column: adjoint solution. Top row: classical Poisson problem. Middle row: obstacle problem with $g=-0.01$ and $\bar{\gamma}=0.1$ and the full error estimator. Bottom row: obstacle problem with $g=-0.01$ and $\bar{\gamma}=0.1$ and the simplified error estimator. we notice that the setups, numerical solutions and limiting values (i.e., $u(0.5,0.5)\approx -0.73$ and $u(0.5,0.5)\approx -0.0103$) coincide with those obtained for uniformly refined meshes presented in Section 8.13.

10.13.3.2 Config. 1: Classical Poisson For the classical Poisson problem, we obtain the values shown in Table 2. It can be nicely observed that the I_{eff} is around 1, which means that the error estimator η approximates very well the 'true' error $J(u)-J(u_h)$. Furthermore, the true error and η converge with about the order 2 (value is divided by 4 at each refinement step). This confirms the theory that for point value evaluations we should expect a convergence order $J(u)-J(u_h)=O(h^2)$.

DoFs	Error $J(u) - J(u_h)$	η	I_{eff}
27	$2.01e - 02$	$2.00e - 02$	$9.98e - 01$
75	$4.01e - 03$	$4.03e - 03$	$1.01e + 00$
243	$9.27e - 04$	$9.28e - 04$	$1.00e + 00$
723	$2.37e - 04$	$2.44e - 04$	$1.03e + 00$
2691	$5.79e - 05$	$5.91e - 05$	$1.02e + 00$
7467	$1.71e - 05$	$1.92e - 05$	$1.13e + 00$

Table 2: Configuration 1: classical Poisson

10.13.3.3 Config. 2: Obstacle, $\gamma = \frac{\bar{\gamma}}{h^2}$ For the obstacle problem, we notice that the goal functional is part of the active set zone in which the constraint is approximated and not anymore the PDE is solved. Here the proposed error estimator works much less well than before with a huge underestimation of the true error. Specifically, the true error converges at a much smaller rate than for the classical Poisson problem. But still, both $J(u)-J(u_h)$ and η decrease at each refinement step, so local mesh adaptivity still pays off. In conclusion: for quantitatively reliable results, the proposed error estimator for the obstacle needs to be improved or at least taken with care.

DoFs	Error $J(u) - J(u_h)$	η	I_{eff}
27	$8.28e - 02$	$2.01e - 02$	$2.43e - 01$
75	$6.48e - 02$	$4.04e - 03$	$6.23e - 02$
243	$5.51e - 02$	$9.13e - 04$	$1.66e - 02$
723	$3.80e - 02$	$2.12e - 04$	$5.58e - 03$
2691	$1.64e - 02$	$2.12e - 04$	$1.49e - 03$
9387	$4.53e - 03$	$7.46e - 07$	$1.65e - 04$

Table 3: Configuration 2. We observe a large underestimation of the true error.

10.13.3.4 Config. 3: Obstacle, simplified version Here, our results are very similar to the second example.

DoFs	Error $J(u) - J(u_h)$	η	I_{eff}
27	$8.28e - 02$	$2.04e - 02$	$2.46e - 01$
75	$6.48e - 02$	$4.75e - 03$	$7.33e - 02$
243	$5.51e - 02$	$1.70e - 03$	$3.08e - 02$
723	$3.80e - 02$	$8.30e - 04$	$2.19e - 02$
1803	$1.71e - 02$	$3.49e - 04$	$2.05e - 02$
4947	$5.03e - 03$	$1.06e - 04$	$2.11e - 02$

Table 4: Configuration 3. We observe a large underestimation of the true error.

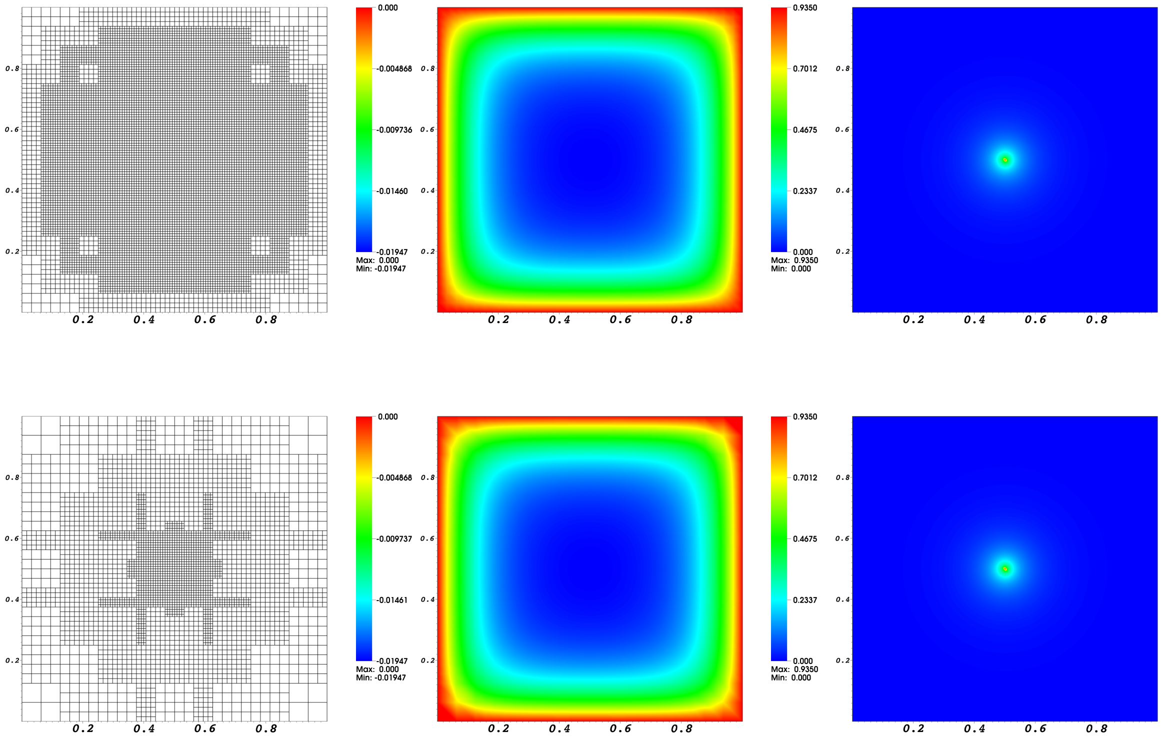

Figure 34: Config. $4-5$: Left column: adaptively refined meshes. Middle column: primal solutions. Right column: adjoint solution. Here, $\gamma = \bar{\gamma}$ is used without weighting with $1/h^2$. In the top row, the full error estimator is shown and in the bottom row, the simplified version is displayed. With respect to the meshes, the full version yields much more refinement, which is clear because of the high value of $\bar{\gamma}$. Nonetheless, the I_{eff} are nearly perfect. In the simplified version, less mesh elements are refined, and the true error is overestimated as shown in the corresponding tables.

10.13.3.5 Config. 4: Obstacle, $\gamma = \bar{\gamma}$ Our findings for Config. 4 are summarized in Table 5. In this version, we obtain nearly optimal I_{eff}.

DoFs	Error $J(u) - J(u_h)$	η	I_{eff}
27	$1.47e-02$	$1.72e-02$	$1.17e+00$
75	$5.53e-04$	$4.30e-04$	$7.78e-01$
243	$1.59e-04$	$1.67e-04$	$1.05e-00$
867	$4.12e-05$	$4.18e-05$	$1.01e-00$
2835	$1.06e-05$	$1.09e-05$	$1.03e-00$
10011	$2.64e-06$	$2.84e-06$	$1.08e-00$
37155	$6.35e-07$	$6.83e-07$	$1.08e-00$

Table 5: Configuration 4. Here, we obtain nearly optimal I_{eff}.

10.13.3.6 Config. 5: Obstacle, simplified version, $\gamma = \bar{\gamma}$ Our findings for Config. 5 are summarized in Table 6. Using this simplified version, we now observe an overestimation of the true error.

DoFs	Error $J(u) - J(u_h)$	η	I_{eff}
27	$1.47e-02$	$4.15e-02$	$2.83e+00$
75	$5.53e-04$	$1.95e-02$	$3.53e+01$
243	$1.59e-04$	$5.43e-03$	$3.42e+01$
723	$4.45e-05$	$1.41e-03$	$3.16e+01$
1491	$2.18e-05$	$3.88e-04$	$1.78e+01$
4227	$7.75e-06$	$1.08e-04$	$1.39e+01$
12603	$2.49e-06$	$3.02e-05$	$1.21e+01$

Table 6: Configuration 5. Here, we observe overestimation of the true error.

10.13.4 Literature on a posteriori error estimation for the obstacle problem

We briefly mention that a DWR version for the obstacle problem working directly with the variational inequality was developed in [155][Chapter 5]. Other approaches and findings for a posteriori error estimation for the obstacle problem can be found, for instance, in [1, 99, 165]

10.14 Predictor-corrector mesh adaptivity

This section is concerned with local mesh adaptivity with a focus on the crack path. Phase-field approaches require fine meshes around the interface (here, the fracture) in order to provide solutions of sufficient accuracy. Therefore, the goal is quite simple: we want to refine exclusively the fracture zone in order to work with a small regularization parameter ε. We notice that in comparison to the previous sections, we **only can provide mesh adaptivity, but no error estimator**.

There are several studies that have been investigated anisotropies introduced by the mesh [129], anisotropic adaptive mesh refinement [14], and pre-refined meshes when the crack path is known a priori [24]. Here, we are interested in a general treatment in which no a priori information of the crack path is required.

We are aware of attempts and studies that investigate the dependence of the crack path versus mesh refinement. In standard tests, e.g., single edge notched shear or three-point bending test, we could not find evidence of these dependencies.

10.14.1 The main algorithm

In the following we describe the predictor-corrector mesh refinement algorithm in more detail. The basic idea is to pick a single, small ε, and then decide on an adaptive refinement level r for the crack region that ensures $h < \varepsilon$. We then refine the mesh adaptively during the computation so that it is on level r in the crack region. To handle fast growing cracks with a priori unknown paths, we employ a predictor-corrector scheme that keeps repeating the current time step to guarantee the finest mesh level r in the crack region.

Algorithm 10.64 (At a single time step t_n)**.** *Let the solution to time step t_n be given; see Figure 36 top left.*

1. *Solve the full problem with a prediction of the new crack path at time step t_{n+1}; see Figure 36 top right.*

2. *If the crack is not resolved adequately, i.e., we have cells in which $h > \varepsilon$, we employ a predictor-corrector cycle:*

   a) *First, we refine the mesh based on the new solution and interpolate the old solution (at t_n) onto the new mesh (Figure 36, bottom left). The refinement is done using a chosen threshold $0 < C < 1$ ($C = 1$ corresponds to global mesh refinement) for the phase-field φ. Each cell that has at least one support point with value $\varphi(x_i) < C$ will be refined unless we are already at the maximum desired refinement level r.*

b) *Then we solve for the solution at t_{n+1} again, but on the refined mesh (Figure 36, bottom right).*

c) *Go to (a) and repeat refinement until maximum refinement level is reached.*

Remark 10.65. *The computational cost includes additional solves when the crack is growing, but the method is very robust and efficient as proven in several studies [82, 106].*

Remark 10.66. *So far, we could not find evidence that the crack path is influenced by the mesh refinement. In principle this may be possible, as motivated in the study of [14].*

In summary, our proposed predictor-corrector scheme - forcing the growing crack region to always be resolved with a fine mesh - reads:

Algorithm 10.67 (Predictor-corrector mesh adaptivity)**.** *Choose a fixed refinement level r for the crack region. On level r, determine $h_{max}^{(r)}$ and pick an appropriate $\varepsilon := \varepsilon^{(r)} > h_{max}^{(r)}$. Select a bound $0 < C < 1$ for φ to be considered inside the crack. For each time step do:*

1. *Solve for solution (u^{n+1}, φ^{n+1}) at t_{n+1}.*
2. *If cells need to be refined (cell with level $k < r$ has $\varphi^{n+1}(x) < C$):*
   refine and transfer solution from t_n, go to 1.

Remark 10.68. *The parameter ε needs to chosen relative to the largest cell size h that can appear on level r during the computation. For refinement of a quadrilateral mesh this quantity can be computed from the set of coarse cells $\mathcal{T}$ using*

$$h_{max}^{(r)} = \max_{T\in\mathcal{T}} 2^{-r} h_T$$

where h_T is the size of cell T.

Remark 10.69. *Another interpretation of this scheme is that we pick a-priori constant values ε and h for the computation and then coarsen cells away from the crack region where the solution is smooth.*

10.14.2 Goals and illustrations

Proposition 10.70. *The predictor-corrector algorithm asks for the following properties [82]:*

1. ***K**eep a single fixed, regularization parameter ε during the entire computation. Most importantly ε is a model parameter, and we do not want to change the fracture model during a computation! Decreasing ε (locally) during the computation will not allow for an increase in the accuracy of the solution. While reducing ε would result in a thinner crack mushy zone - thus changing the model. As a consequence, ε should not be changed during the computation.*

2. ***E**nsure $\varepsilon > h$ inside the crack region. It is required to have a sufficiently small mesh size h to resolve the transition of the phase field variable. The width of this zone is controlled by the choice of ε. Importantly, the ε-h relationship is only required to be satisfied inside or directly around the current crack region and not in the whole computational domain.*

3. ***E**rror is controlled by ε, not h: the interplay of model and discretization errors*

   In contrast to standard a-posteriori or goal-oriented adaptive mesh refinement, just refining the mesh does not reduce the discretization error significantly. This is because the choice of ε determines the width of the mushy zone around the crack path. Ideally, an adaptive method would try to minimize ε and pick an appropriate h to minimize the discretization error introduced by the mesh size.

4. ***N**o requirement of prior knowledge about the crack location(s). Typically, specifically for more realistic problems, the final location of the cracks is unknown. While it is an option to repeat the whole computation on a finer mesh that is determined using the first computation, this is too expensive to be practical. Therefore, the **predictor-corrector** algorithm should detect during the computation in which direction the cracks are growing.*

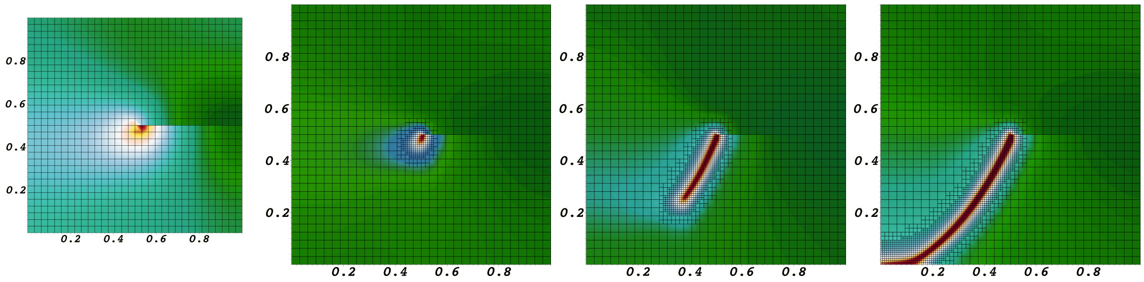

Figure 35: Functionality of predictor-corrector mesh refinement in two spatial dimensions: the mesh grows with the fracture. Here the transition zone with $0 < \varphi < 1$ determines the region in which the mesh has to be refined. If the fracture (in red) grows faster than the fine mesh, such that $\varepsilon > h$ is violated, we first refine the mesh (the predictor step) according to the transition zone and than perform a second computation to determine the precise fracture location inside the refined mesh.

5. ***H**andling fast growing cracks. Only adapting the mesh based on the current crack location before moving on to the next time step may result that the adaptive mesh lags behind in time and does not resolve the crack region adequately, in particular, if the crack is growing rapidly.*

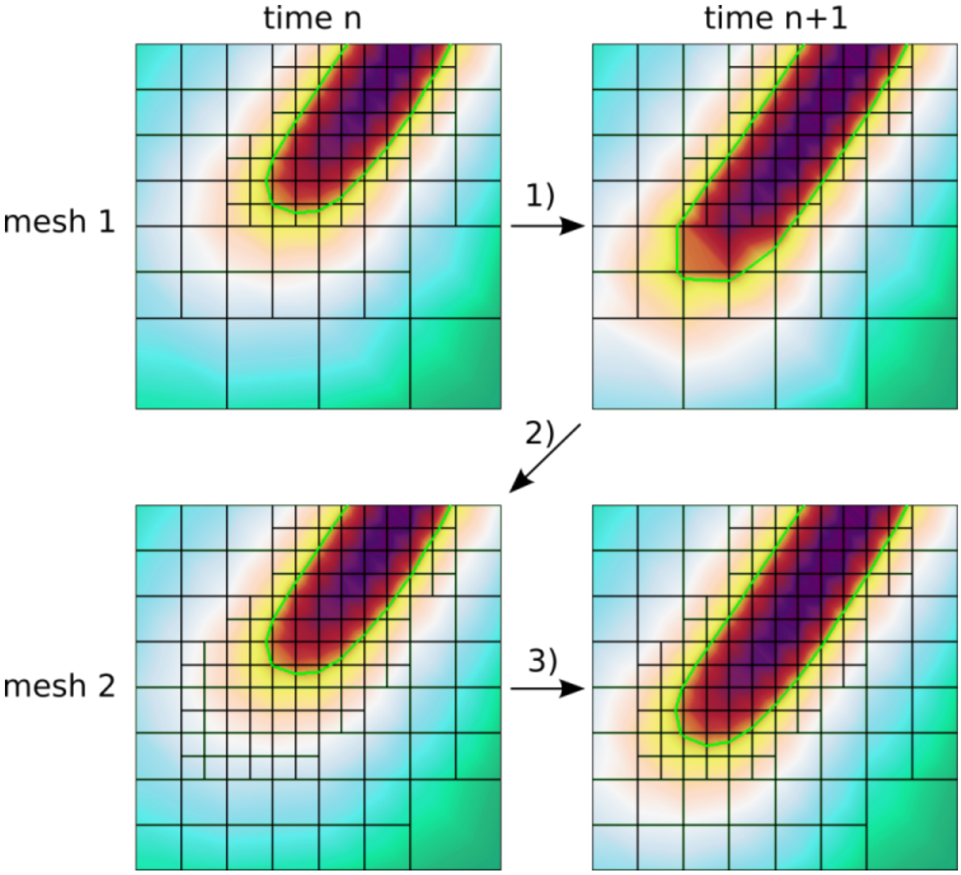

Figure 36: Predictor-corrector scheme: 1. advance in time, crack leaves fine mesh. 2. refine and go back in time (interpolate old solution). 3. advance in time on new mesh. Repeat until mesh doesn't change anymore. Refinement is triggered for $\varphi < C = 0.2$ (green contour line) here.

10.14.3 Performance

We re-copy the outcomes of [82]. For the single edge notched shear test - see Chapter 9 - we used an Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz machine with two processors. The following results were obtained:

Table 7: Single edge notched shear test: Comparison of computational cost for different refinement strategies. The numbers in the prec/corr strategy indicate the phase-field threshold value used for mesh refinement.

Strategy	Time [s]	Time [min]	Number of loading steps	DoFs: min/avg/max
global refinement	5036	84	151	50115
local prerefinement	1277	21	151	19746
predictor/corrector (0.8)	233	4	151+63	3315/4731/8286
predictor/corrector (0.6)	184	3	151+53	3315/4225/6666

Remark 10.71. *We notice that the accuracy of the different simulations was comparable as shown in [82]. Thus it is really worth to use predictor-corrector mesh refinement for this numerical example.*

10.15 Adaptive time step control

In this section, we present an algorithm for adaptive time step control. Since the presented phase-field model is only quasi-static it is questionable whether the algorithm works in all cases, however it makes sense to discuss this topic since in many fracture propagation problems we deal with basically three temporal regimes:

- $G < G_c$ and the fracture does not yet propagate (larger time steps/loading steps possible)
- $G = G_c$ fracture propagation (possibly smaller time steps necessary);
- $G > G_c$ or a complete damaged material (again larger time steps or even finishing the simulation due to total damage/fracture).

Our aim is to control the accuracy of a certain physical quantity of interest $J(U)$. The principle idea, e.g., [142], is based on ODE (ordinary differential equations) theory and utilizes the local truncation error. Thus the only assumptions we have to make is enough regularity of the truncation error and discrete stability of the underlying difference scheme used for discretizing the continuous time derivatives such as $\partial_t \hat{v}_f$.

The following description is adapted from an heuristic estimator developed in [60] in which time step control has been developed for the Fractional-Step-θ scheme for FSI multiphysics computations.

The entire approach is a combination of known results from the literature, which we briefly recapitulate to acknowledge the original authors. We follow an idea proposed by [161] but extend his criterion with additional terms, which are inspired by [75, 94]. The result is a (very) simple method but the price to pay are additional solves of the problem per time step. Specifically, per time step, we compute once the problem with $2k$, compute the goal functional $J(U)$, and then compute two times with time step size k again. After we compare $J(U_k)$ and $J(U_{2k})$ by evaluating the absolute error $abs_{err} := |J(U_{2k}) - J(U_k)|$.

Proposition 10.72. *Let the error tolerance TOL_{TS} be given. We evaluate:*

$$abs_{err} := |J(U_{2k}) - J(U_k)| \tag{81}$$

and then

$$\theta = \gamma * \left(\frac{TOL_{TS}}{abs_{err}}\right)^K, \tag{82}$$

where $\gamma \approx 1$ is a safety factor (in [75], $\gamma = 0.9$), $K = \frac{1}{15}$ as in [75]. We then compute the new time step size as

$$k_{new} = \begin{cases} k_{old} & 1 \le \theta \le 1.2, \\ k_{old} * \theta & \textit{otherwise.} \end{cases} \tag{83}$$

Moreover, we check whether $k_{min} \leq k_{new} \leq k_{max}$. If $k_{new} < 0.5k_{old}$, i.e., the new time step would be much smaller than the old one, we redo the current time step [161].

Remark 10.73. *The check* (83) *prevents high time step oscillations and just keeps the old time step if the difference is not too large. Several illustrating computations have been presented in [75].*

Remark 10.74. *The last check $k_{new} < 0.5k_{old}$ is very expensive since the entire calculation of the solution of this time step has to be repeated. However, in practice this happens rarely, but is necessary since a dramatic decrease of the time step size indicates that the current solution is by far not accurate enough.*

Algorithm 10.75. *The adaptive time step control for One-Step-θ schemes reads at t_n:*

1. *Set $k_{old} := k_{n-1}$*
2. *Perform*
   - *one computation with $2k_{old}$, evaluate $J(\hat{u}_{2k})$ at $t_n + 2k_{old}$;*
   - *compute two steps with k_{old}, evaluate at the end of the 2nd step $J(\hat{u}_k)$ at $t_n + k_{old} + k_{old}$;*
3. *Evaluate* (81) *and compute* (82)*;*
4. *Compute the new k_{new} with* (83)*;*
5. *If $k_{new} < 0.5k_{old}$, set $k_{old} := k_{new}$ and go to Step (ii);*
6. *Otherwise accept k_{new} and set $k_n = k_{new}$;*
7. *Increment $n \to n + 1$ and go to Step (i).*

10.16 Final comments to error estimation in numerical simulations

We give some final comments to error estimation. One should keep in mind several aspects. First:

- Numerical mathematics is its own discipline and we should aim to develop methods with the best accuracy as possible.
- On the other hand, numerical mathematics is a very useful tool for many other disciplines (engineering, natural sciences, etc.). Here, an improvement of a process (i.e., approximation) by several percents is already very important, while the total error may be still large though and in the very sense of numerical mathematics the result is not yet satisfying.

Second (w.r.t. the first bullet point): A posteriori error estimation, verification of the developed methods, and code validation are achieved in three steps (with increasing level of difficulty) after having constructed an a posteriori error estimator η that can be localized and used for local mesh adaptivity.

1. If possible, test your code with an acknowledged benchmark configuration and verify whether $J(u_h)$ matches the benchmark values in a given range and on a sequence of at least three meshes. This first step can be performed with uniform and adaptive mesh refinement.
2. Compute $J(u)$ either on a uniformly-refined super-fine mesh or even analytically. Compute the error $J(u) - J(u_h)$ and observe whether the error is decreasing. If a priori estimates are available, see if the orders of convergence are as expected. But be careful, often goal functionals are nonlinear, for which rigorous a priori error estimates are not available.
3. Compare η and $J(u) - J(u_h)$ in terms of the effectivity index I_{eff}.

In the very sense of numerical mathematics, we must go for all three steps, but in particular the last step No. 3 is often very difficult when not all theoretical requirements (smoothness of data and boundary conditions, the regularity of the goal functional, smoothness of domains and boundaries) are fulfilled. Also, keep in mind that all parts of error estimators are implemented and that still the remainder term R may play a role. For instance, in practice, very often when working with the DWR method, only the primal error estimator is implemented and the adjoint part neglected; see the following section.

Remark 10.76 ('Application' of these final comments)**.** *We refer to the plots tables presented in Section 10.13 (Excursus IV) for an 'application' of the above explained steps.*

1. *We approach obviously the correct reference values (please see the color bars of the primal solution)*

2. *Also we observe that the true error $J(u) - J(u_h)$ and the estimator η are both decreasing; with different orders depending on the setting. Therefore, we observe a first difference and need to be careful now. For mesh refinement, the results are still all reliable. For a quantitative error estimation, however, not!*

3. *The last step is indeed the most challenging: does the estimator cover the true error, i.e., can we obtain quantitative information from the proposed error estimator? Well, this is only true for Config.* 4, *Table 5, in which $I_{eff} \approx 1$.*

11 Simulations II: a PU-DWR method for a slit modeled by phase-field

This numerical test was originally inspired by [12, 22] in which a manufactured solution for a scalar-valued displacement field (i.e., Laplacian) was constructed. In [12, 22], a manufactured solution for the displacement field has been constructed. In our studies, the crack is represented by a phase-field function. Based on these works, in [173], the following studies were carried out:

- taking the solutions [12, 22] to study $\varepsilon \to 0$ for the corresponding phase-field fracture approximation;
- development of a PU-DWR error estimator for both the scalar-valued case and the vector-valued case (elasticity);
- studies of various h-ε relationships.

In this section, we revisit the entire setting. Here, we notice that the programming code was newly developed for these lecture notes since the version developed in [173] appeared non-practical for even further extensions.

11.1 Configuration and manufactured solution

The domain is $(-1,1)^2$ with a slit from $-1 \leq x \leq 0$ for $y = 0$. This slit is caused a discontinuity of the boundary conditions at $(-1, 0)$.

The analytical solution on the slit domain $(-1,1) \setminus \{(x,0)| -1 \leq x \leq 0\}$ is given by [22]:

$$(\lambda_{G_c} r^{1/2} \sin \phi/2; \{(x,0)| -\infty \leq x \leq 0\})$$

where polar coordinates with $r^2 = x^2 + y^2$ are used.

11.2 Boundary conditions

Employing the boundary function $u_D := g = \lambda_{G_c} \sin \phi/2$ on ∂B, we prescribe non-homogeneous Dirichlet conditions on all parts.

Specifically, transforming g into Cartesian coordinates we have

$$x \leq 0 \text{ and } y \geq 0 : g(x,y) = \lambda_{G_c}/\sqrt(2) * \sqrt{\sqrt{x^2+y^2} - x},$$
$$x \leq 0 \text{ and } y \leq 0 : g(x,y) = -\lambda_{G_c}/\sqrt(2) * \sqrt{\sqrt{x^2+y^2} - x},$$
$$x \geq 0 \text{ and } y \geq 0 : g(x,y) = \lambda_{G_c}/\sqrt(2) * \sqrt{\sqrt{x^2+y^2} - x},$$
$$x \geq 0 \text{ and } y \leq 0 : g(x,y) = -\lambda_{G_c}/\sqrt(2) * \sqrt{\sqrt{x^2+y^2} - x}.$$

For the phase-field variable, traction-free conditions (i.e., homogeneous Neumann conditions are applied).

11.3 Initial condition

For the phase-field function, we need an initial value φ^0 to prescribe the slit in the initial geometry. Here, we use:

$$\varphi^0 := \{(x,y) \in \Omega| \ -1 \leq x \leq 0; -h \leq y \leq h\},$$

where h is the mesh size parameter defined in the next subsection.

11.4 Parameters

The model and material parameters are given as: $\kappa = 10^{-12}$, $G_c = \lambda_{G_c}^2 \times \sqrt{\pi/2}$, $\lambda_{G_c} = 1.0$, and $\mu = 1.0$. Specifically $\varepsilon = 0.353553$, which corresponds to $\varepsilon = 2h$ on a four times uniformly refined grid with $h = 0.176777$. The penalization parameter is $\gamma = \bar{\gamma} = 1e + 3$.

11.5 Goal functional and reference value

As goal functional we choose again a point value:

$$J(u, \varphi) := u(0.75, 0.75)$$

The reference value is computed on an eight times uniformly refined grid:

$$J(u_{ref}, \varphi_{ref}) := u(0.75, 0.75) = 4.0369872630529557e - 01,$$

with ε and h given before.

11.6 PU-DWR error estimator in primal form

We only implemented the primal form of the error estimator. For this highly nonlinear problem this might be not sufficient for good I_{eff}. The complete extension is ongoing research and implementation.

Proposition 11.1 (DWR phase-field fracture, primal form)**.** *For the finite element approximation of the phase-field fracture problem, we have the a posteriori error estimate*

$$|J(U) - J(U_h)| \leq \eta(u_h) := \sum_{i=1}^{N} |\eta_i|$$

with

$$\eta_i = \langle f, (z - i_h z)\psi_i \rangle - a(U, (Z - i_h Z)\psi_i) \tag{84}$$

$$= \langle f, (z_u - i_h z_u)\psi_i \rangle - \Big([(1-\kappa)\varphi^2 + \kappa]\mu \nabla u, \nabla((z_u - i_h z_u)\psi_i) \Big) \tag{85}$$

$$- \Big((1-\kappa)\varphi\mu|\nabla u|^2, ((z_\varphi - i_h z_\varphi)\psi_i) \Big) + (\frac{G_C}{\epsilon}(1-\varphi), ((z_\varphi - i_h z_\varphi)\psi_i)) - (G_C \epsilon \nabla\varphi, \nabla((z_\varphi - i_h z_\varphi)\psi_i)) \tag{86}$$

$$+ \gamma([\varphi - \varphi^0]^+, ((z_\varphi - i_h z_\varphi)\psi_i)) \tag{87}$$

Here $U = (u, \varphi)$ and $Z = (z_u, z_\varphi)$ and we understand the we work here with the discrete versions. Moreover, in the present numerical test, $f \equiv 0$. The deformation is caused by non-homogeneous displacement boundary conditions are previously described.

Remark 11.2. *Due to the 'time' dependence of the phase-field variable, the problem is not purely stationary (see all of our discussions in the previous chapters). In this example, we imposed the initial condition φ^0 and computed just one single 'time' step.*

Remark 11.3. *For solving the primal problem, the extrapolation on φ was used as previously explained:*

$$\Big((1-\kappa)\varphi\mu|\nabla u|^2, ... \Big) \quad \to \quad \Big((1-\kappa)\tilde{\varphi}\mu|\nabla u|^2, ... \Big)$$

with $\tilde{\varphi} := \varphi^0$.

11.7 Code snippet of the PU-DWR loop implemented in deal.II

```
for ( ; cell!=endc; ++cell, ++cell_adjoint)
{

// Gather local error indicators while running
// of the degrees of freedom of the partition of unity
// and corresponding quadrature points.
for (unsigned int q=0; q<n_q_points; ++q)
{

// Run over all PU degrees of freedom per cell (namely 4 DoFs for Q1 FE-PU)
for (unsigned int i=0; i<dofs_per_cell; ++i)
{
 // Notice that the displacements are implemented in a vector-valued fashion
 // in order to easier extend to elasticity. In this example,
 // the 2nd component is zero though.
 Tensor<2,dim> grad_phi_psi;
 grad_phi_psi[0][0] = fe_values_pou[pou_extract].value(i,q) * grad_dw_v[0][0]
     + dw_v[0] * fe_values_pou[pou_extract].gradient(i,q)[0];
 grad_phi_psi[0][1] = fe_values_pou[pou_extract].value(i,q) * grad_dw_v[0][1]
     + dw_v[0] * fe_values_pou[pou_extract].gradient(i,q)[1];
 grad_phi_psi[1][0] = fe_values_pou[pou_extract].value(i,q) * grad_dw_v[1][0]
     + dw_v[1] * fe_values_pou[pou_extract].gradient(i,q)[0];
 grad_phi_psi[1][1] = fe_values_pou[pou_extract].value(i,q) * grad_dw_v[1][1]
     + dw_v[1] * fe_values_pou[pou_extract].gradient(i,q)[1];



 Tensor<1,dim> grad_phi_pf_psi;
 grad_phi_pf_psi[0] = fe_values_pou[pou_extract].value(i,q) * grad_dw_p[0] +
    dw_p * fe_values_pou[pou_extract].gradient(i,q)[0];
 grad_phi_pf_psi[1] = fe_values_pou[pou_extract].value(i,q) * grad_dw_p[1] +
    dw_p * fe_values_pou[pou_extract].gradient(i,q)[1];



  // Implement the error estimator
  // J(u) - J(u_h) \approx \eta := (f,...) - a({u,\varphi},...)
  //
  // First part: (f,...)
  local_err_ind(i) += (force  * dw_v * fe_values_pou[pou_extract].value(i,q)
        ) * fe_values_pou.JxW (q);

  // Second part: - (\nabla u, ...)
  local_err_ind(i) -=
    (scalar_product(((1.0-constant_k) * pf *  pf + constant_k) * stress_term,grad_phi_psi)
   + pf_minus_old_timestep_pf_plus * dw_p * fe_values_pou[pou_extract].value(i,q)
   + (1.0 - constant_k) * scalar_product(stress_term, E) * pf
     * dw_p * fe_values_pou[pou_extract].value(i,q)
   - G_c/alpha_eps * (1.0 - pf) * dw_p * fe_values_pou[pou_extract].value(i,q)
   + G_c * alpha_eps * grad_pf * grad_phi_pf_psi
   ) * fe_values_pou.JxW (q);

  }  // end loop DoFs i (PU FE)
 }  // end quadrature points of the PU DoFs
}  // end element (cell) loop using the PU degree of freedom handler
```

11.8 Numerical results

Using the previously stated PU-DWR error estimator, we obtain:

```
DoFs     True error       \eta             est ind          I_{eff}          ind_{eff}
867      3.19e-03         2.69e-04         1.78e-03         8.42e-02         5.56e-01
1959     1.43e-04         2.83e-05         5.51e-04         1.97e-01         3.84e+00
6249     7.25e-05         1.75e-05         1.63e-04         2.41e-01         2.25e+00
18693    3.85e-05         6.05e-06         5.44e-05         1.57e-01         1.41e+00
```

For the indicator index (last column) and its estimator (fourth column), we refer to [150].

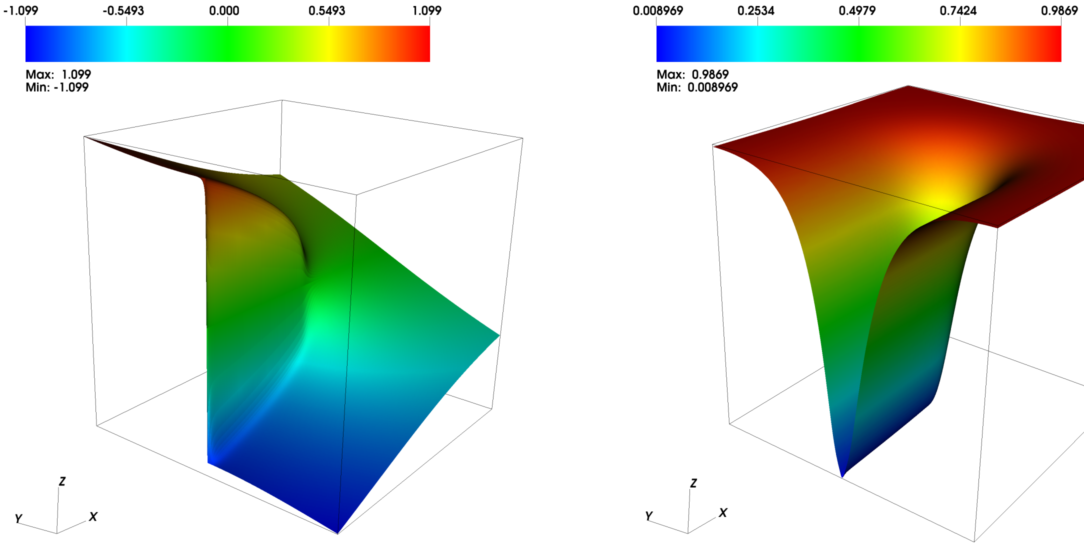

Figure 37: Surface plots of the scalar-valued displacement field (left) and phase-field approximation (right).

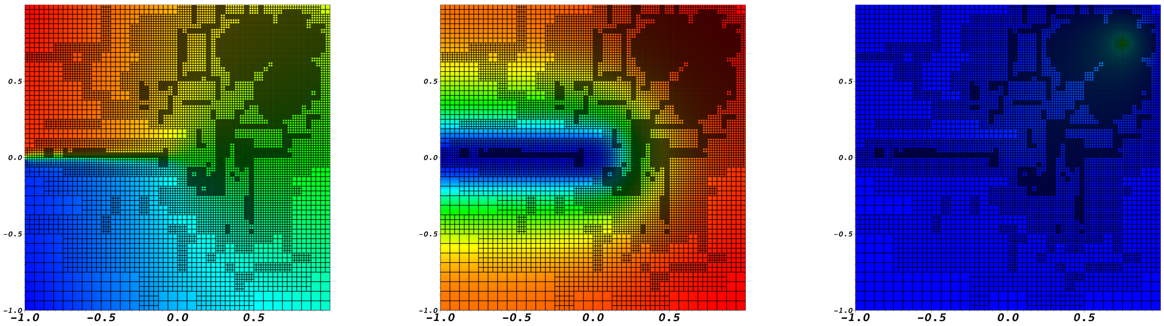

Figure 38: Plots of adaptively refined mesh with the point goal functional together with the primal solution, phase-field solution, and the adjoint solution. We observe localized mesh refinement towards the goal functional as well as mesh refinement inside the fracture and at the fracture tip. In particular, the latter two refinement are to be expected and show that a DWR method achieves both: refinement in the goal functional as well as refinements usually obtained by classical residual-based error estimation.

12 Simulations III: screw test: time adaptivity and iteration on extrapolation

In this chapter, our main aim is on two numerical aspects:

- Iteration on the extrapolation 8.6;
- Adaptive time step control 10.15.

From the mechanical point of view this test is interesting as well because no initial crack are prescribed. Rather fracture will develop in the regions with high stresses. These results were experimentally confirmed in [170].

12.1 Configuration

The geometric setting is displayed in Figure 39. The total length is $17.20mm$. The initial mesh has 3440 elements with $10800(7200+3600)$ DoFs. The corresponding minimal cell diameter is $h_{min} = 0.0689518$ We notice that the mesh is unstructured and not regular since the screw geometry was created using gmsh [69].

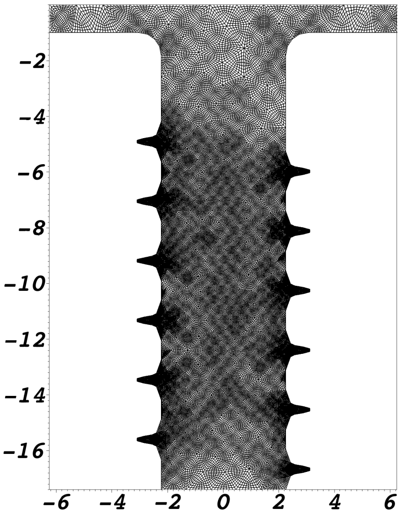

Figure 39: Mesh of screw simulations. The screw is fixed at the bottom, at top we have non-homogeneous Dirichlet conditions in y-direction (uniform tension). The units are in mm.

12.2 Boundary conditions

Crack growth is driven by a non-homogeneous Dirichlet condition for the displacement field u on Γ_{top}, the head of the screw at $y = 0.0$. We increase the displacement on Γ_{top} at each time step such that the head is pulled, namely

$$u_y = \delta t \times \bar{u}, \quad \bar{u} = 1.0 \text{ mm},$$

where $\delta t = 10^{-2}$s.

12.3 Initial condition

We prescribe all initial conditions to be zero. In particular, there is no initial fracture. Thus we will observe fracture nucleation.

12.4 Parameters

As model parameters, we choose $\gamma = 1$, $\kappa = 10^{-10}h, \epsilon = 2h$ mm. We notice that the maximum number of possible augmented Lagrangian iterations is 20 and TOL_{AL} is variable as shown below. The Newton tolerance is $TOL_N = 10^{-8}$. As material parameters, we use $\mu = 80.77kN/mm^2$, $\lambda = 121.15kN/mm^2$ and $G_c = 2.7N/mm$.

The initial time step size is $k = 0.01s$. For adaptive time step control we set the following parameters (see Section 10.15 for their meaning and final algorithm):

$$\begin{aligned} TOL_{TS} &= 1e-2 \quad \text{and} \quad 1e-3 \\ k_{max} &= 1e-2 \\ k_{min} &= 1e-4 \\ \gamma &= 1.0 \\ K &= 1/15 \end{aligned}$$

12.5 Quantity of interests

We study the following goal functionals:

- We observe the bulk energy

$$E_B = \int_\Omega ([1-\kappa]\varphi^2 + \kappa)\psi(e)\,dx, \tag{88}$$

and the crack energy

$$E_C(\varphi) = G_C \int_\Omega \Big(\frac{(1-\varphi)^2}{4\varepsilon} + \varepsilon|\nabla\varphi|^2 \Big)\,dx, \tag{89}$$

with the strain energy functional

$$\psi(e) := \mu \mathrm{tr}(e(u)^2) + \frac{1}{2}\lambda \mathrm{tr}(e(u))^2, \quad \text{with } e := e(u) := \frac{1}{2}(\nabla u + \nabla u^T),$$

and $|\nabla\varphi|^2 := \nabla\varphi : \nabla\varphi$.

As goal functional for adaptive time step control, we choose:

$$J(U) := E_C(\varphi).$$

12.6 Numerical results and discussion

In the Figures 40 and 41, graphical solutions are provided.

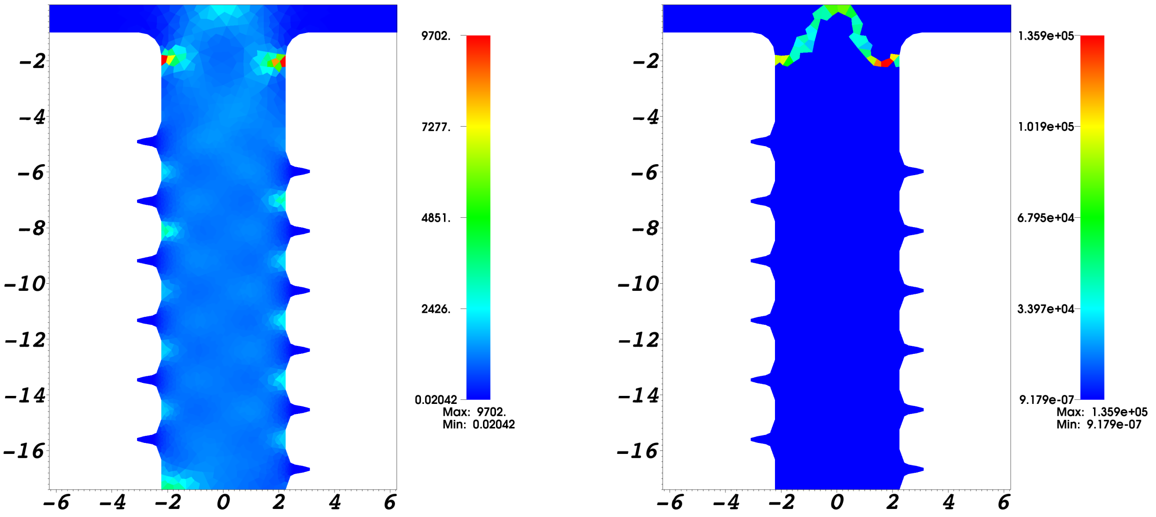

Figure 40: Screw simulations: numerical solutions at times $T = 9.92e - 2s$ and $T = 1.03e - 1s$. Here, the stresses are shown, which indicate high values in the corners between head and screw piston.

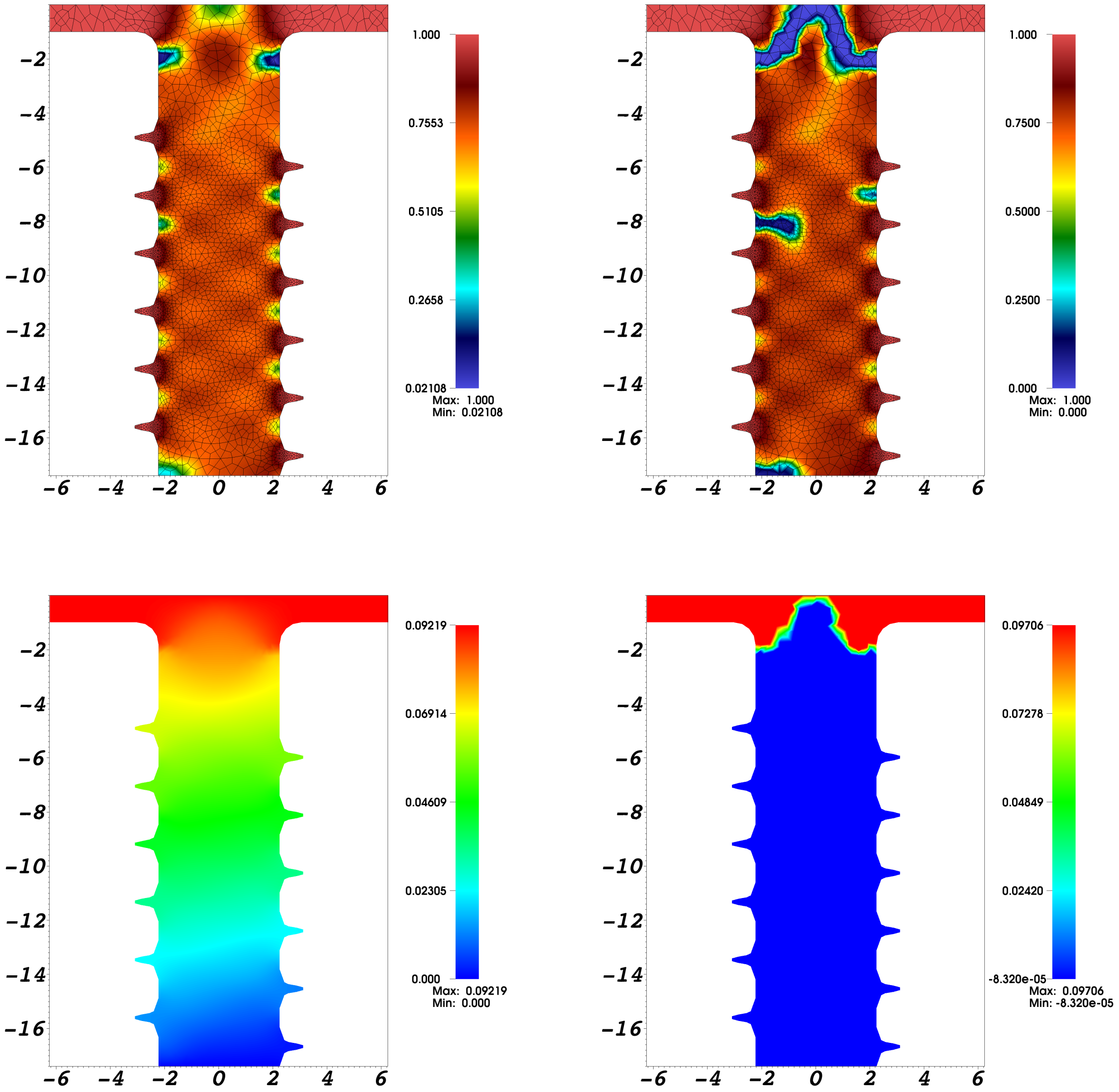

Figure 41: Screw simulations: numerical solutions at times $T = 9.92e - 2s$ and $T = 1.03e - 1s$. On top, the phase-field function is shown with a final cone-like crack in the head. At the bottom, the principal displacement direction u_y is displayed.

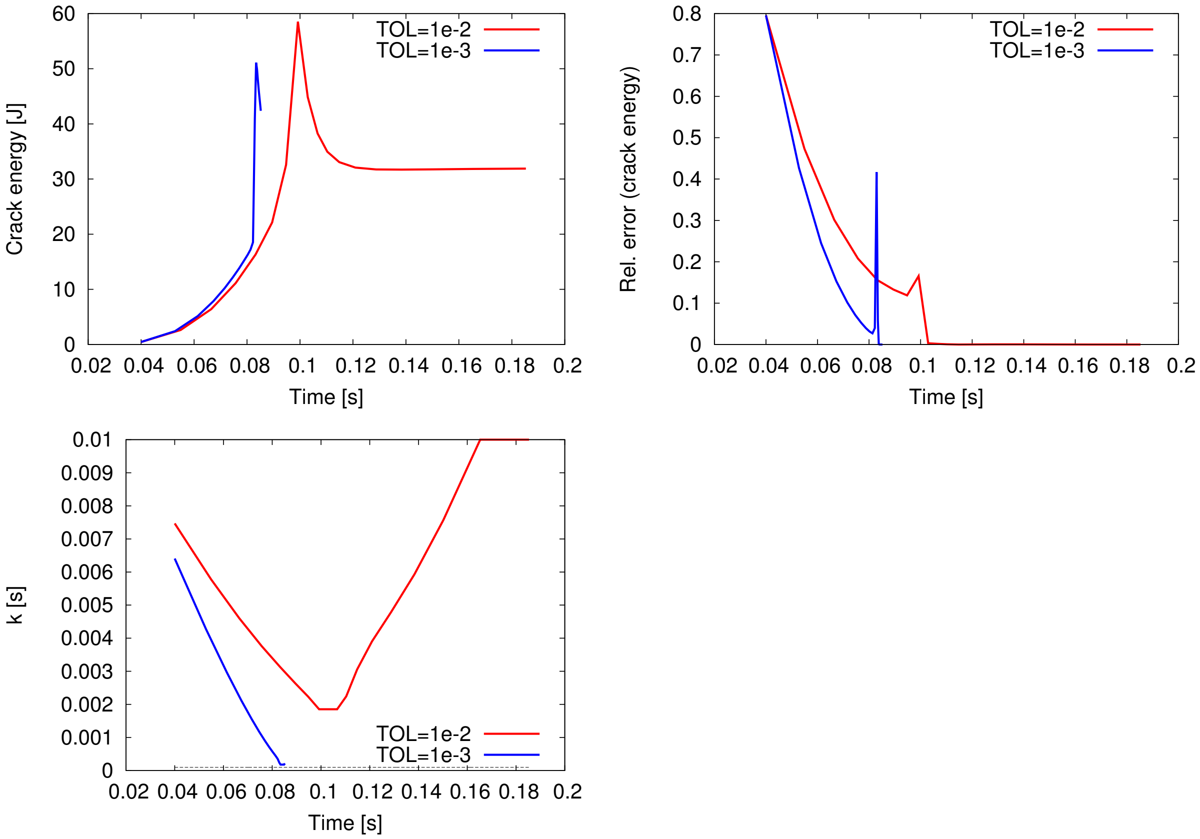

Figure 42: Screw simulations: evolution of the crack energy, its relative error and the corresponding time step sizes k in order to satisfy the prescribed tolerances. We observe an increasing crack energy until the screw is broken. Then, the crack energy remains constant and the relative error tends to zero as to be expected. Consequently, for the broken screw the time step sizes increase again because the relative error is small anyway.

13 Crack width, volume, interfaces - pressurized cracks

In this chapter, we concentrate on extensions and challenges of the classical phase-field fracture model that we studied before. Two main challenges that were outlined are:

- Crack width and volume computations;
- Realizing interface conditions on the fracture boundary.

We discuss both issues in this chapter even if these two topics are still subject to current research and even further modifications.

13.1 Computing the crack width

Not just the crack path, but also the crack width can be of interest, for example talking about medical diseases in the cardiovascular system or hydraulic fracture thinking about pipelines or groundwater flow. Let φ be the phase-field variable as before and φ_{ls} be the level-set variable introduced by Osher/Sethian in the 1980s and defined as $\varphi_{ls} := \varphi - C_{ls}$, where $\varphi_{ls} = 0$ in the fracture, so it follows $\varphi = C_{ls}$ in the fracture $\Gamma := \{x \in B | \varphi_{ls}(x) = 0\}$.

Definition 13.1. *Formally the crack width can be defined as*

$$w = [u \cdot n] = u^+ n^+ - u^- n^-. \tag{90}$$

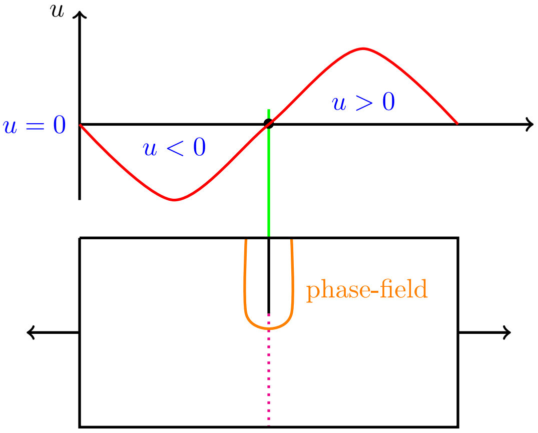

Figure 43: Profile of the deformation u and the phase-field variable φ observing a sheet of paper.

Now the question is how to compute the normal vectors. From calculus/analysis we know that $n_F := -\frac{\nabla \varphi_{ls}}{||\nabla \varphi_{ls}||}$. Let define w_D as the length width on the crack boundary. Under the assumption of symmetry $u^+ = u^-$ it holds $w_D = 2u^+ n^+$. In principle we can compute the fracture width via w_D. However there is an approximation error of size ϵ: This is why the following consideration thinks about how to obtain a *better*

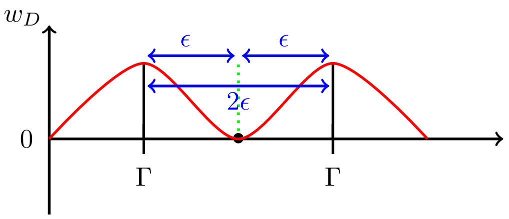

Figure 44: Profile of w_D using the assumption $u^+ = u^-$.

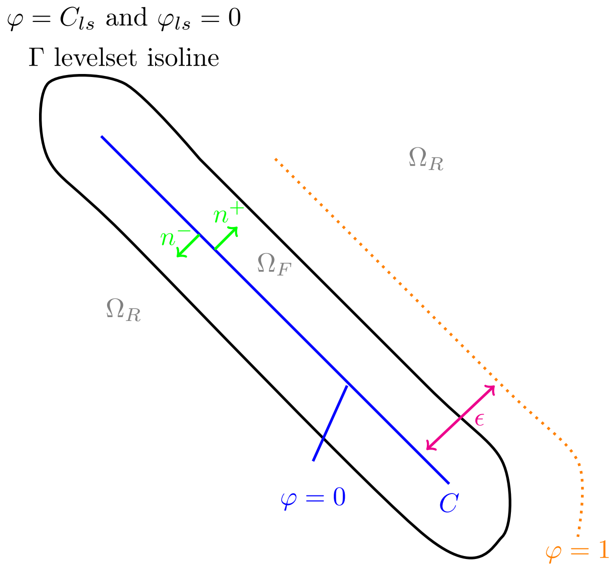

Figure 45: Pressurized fracture: phase field variable φ.

values very close to the true fracture C.

One idea can be to interpolate over the crack by means of a Laplacian-like additional PDE over the whole domain B. Such an additional problem can have the following formulation:

$$\begin{aligned} -\Delta w &= g \quad \text{in } B, \\ w &= w_D \quad \text{on } \Gamma, \\ w &= 0 \quad \text{on } \partial\Omega, \end{aligned}$$

with $g := \beta ||w_D||_{L^\infty}$ and $\beta > 0$. Actually this problem has just to be solved in the fracture but for simplicity it is solved on the whole domain B. β is an heuristic parameter and it is quite not said that this is the best approach.

13.2 Computing the total crack volume

The total crack volume can be computed as follows:

$$TCV_{h,\epsilon} = \int_\Omega u \cdot \nabla \phi dx, \tag{91}$$

13.3 Pressurized fractures

We now add a pressure inside the fracture. This pressure may come from a fluid (Darcy-type, Reynolds equation, Stokes, Navier-Stokes) flowing in the fracture. Possible applications are fractures in porous media and subsurface modeling, for instance groundwater flow.

13.3.1 Fracture and solid domains

We explicitly assume $\epsilon > 0$, that the fracture is in the same dimension as the surrounding material. According to the Figure 45, let us define the following domains:

$$\begin{aligned} \Omega_R &= \{x \in B | \varphi(y) \geq C\}, \\ \Omega_F &= \{x \in B | \varphi(x) \leq C\}, \\ \Gamma &= \partial\bar{\Omega}_F \cap \partial\bar{\Omega}_R. \end{aligned}$$

Here $0 < C < 1$ is a threshold number to determine the fracture domain. A possible choice is $C = 0.1$.

13.3.2 An interface law between the fluid and solid sub-domains

We start with an incompressible fluid of Stokes type in the fracture Ω_F: Find $v : \Omega_F \to \mathbb{R}^d$ and $p : \Omega_F \to \mathbb{R}$ such that

$$\begin{aligned} -\nabla \cdot \sigma_f &= 0 && (92)\\ \nabla \cdot v &= 0, && (93) \end{aligned}$$

equipped with suitable boundary conditions and with the fluid stress tensor $\sigma_f = -pI + \nu\rho(\nabla v + \nabla v^T)$ and the density ρ and kinematic viscosity ν.

Furthermore, it is assumed that the fracture height is much thinner than the fracture height (Reynold's lubrication equation, e.g., [156]). The leading order of stress σ in Ω_F is $\sigma = -pI$. By the presented mathematical modeling now we get an interface problem:

$$\sigma \cdot n = -pI \quad \text{on } \Gamma \quad \text{(interface law)} \tag{94}$$

under the assumption of a dynamic coupling condition: $\sigma_R n_R = -\sigma_F n_F$.

13.3.3 Modification of the energy functional

The goal now is to extend our VPFF to pressurized fractures to account for the additional force balances on Γ with homogeneous Neumann conditions. Hence, we obtain on the energy level:

$$E_R(u, \varphi) = \frac{1}{2}\int_B [(1-\kappa)\varphi^2 + \kappa] Ae(u) : e(u)\, dx - \int_\Gamma \tau \cdot u\, ds, \tag{95}$$

where τ is the fraction force given by a Neumann condition. The boundary term can be reformulated:

$$\begin{aligned} -\int_\Gamma \tau \cdot u\, ds &= -\int_\Gamma \sigma \cdot n \cdot u\, ds && (96)\\ &=_{\text{interface law}} \int_\Gamma pn \cdot u\, ds && (97)\\ &=_{\text{Gauss divergence}} \int_{\Omega_R} \nabla \cdot (pu)\, dx - \int_{\partial\Omega} p \cdot n \cdot u\, ds && (98)\\ &= \int_{\Omega_R} [u\nabla p + p\nabla \cdot u]\, dx - \int_{\partial\Omega} p \cdot n \cdot u\, ds. && (99) \end{aligned}$$

Assumption 13.2. *We make three assumptions for the pressurized fracture model:*

1) *The fracture does not reach the Neumann boundaries:* $\varphi = 1$ *on* $\partial\Omega$

2) $p \cdot n = 0$ *on* $\partial\Omega$, *which is physically plausible. With that the second term can be written as:*

$$-\int_{\partial\Omega} pnu\, ds \quad \to^{PDE\ level} -\int_{\partial\Omega} pnw\, ds, \tag{100}$$

when $w \in H_0^1$. *So it holds*

$$-\int_{\partial\Omega} pnw = 0. \tag{101}$$

3) *Assume that* p *has a low spatial variation:* $\nabla p \approx 0 \;\Rightarrow\; \int_{\partial\Omega} p\nabla \cdot u\, dx$.

Now we extend this integral from Ω_R to B:

$$\int_{\Omega_R} p\nabla\cdot u\; dx \quad \to \int_B [(1-\kappa)\varphi^2+\kappa]p\nabla\cdot u\; dx. \tag{102}$$

Now we modify the energy functional as:

$$E_{t,\epsilon}(u,\varphi) = \int_B g(\varphi)Ae(u) : e(u) + \int_B g(\varphi)p\nabla u\; dx + E_{S,\epsilon}(\varphi). \tag{103}$$

The last last term is the Ambrosio-Tortorelli functional. In a next stop we differentiate to obtain the Euler-Lagrangian equations as before Chapter 5.

Remark 13.3 (Simplification). 1) *Equation* (99) *can easily be extended with*

$$\int_B \varphi^2[u\nabla p + p\nabla\cdot u]\; dx. \tag{104}$$

2) *κ is mainly needed that at least one term contributes of entries for $\varphi\to 0$. For this reason, in the literature, for instance Mikelic, Wheeler, Wick 2013, it is often used $[(1-\kappa)\varphi^2+\kappa]\to\varphi^2$.*

3) *MWW 13 (ICES 13-15) (published as [126]) shows a link between both pressure formulations.*

4) *When normalized pressure is used, the quotient rule holds for the Euler-Lagrangian equations with the φ-equation.*

5) *A generalization to other interface laws of the form $\sigma_R n_R = \sigma_F n_F$ should be easily possible.*

Example 13.4 (Interface for other media). *The following cases could be modeled as well:*

1) *Pressurized fractures: $\sigma_F = -pI$*

2) *Poroelastic medium: $\sigma_R = \sigma_s - \alpha pI$ with $\alpha\in[0,1]$*

3) *Stokes in the fracture: $\sigma_F = -pI + \delta\nu(\nabla u + \nabla u^T)$*

13.3.4 A variational formulation of pressurized phase-field fractures

The energy functional including a given pressure $p:\Omega\to\mathbb{R}$ reads:

$$\begin{aligned} E_{\varepsilon,\varphi,p}(u,\varphi) = \frac{1}{2}\int_B [((1-\kappa)\varphi^2+\kappa)W(e(u))]\,dx + \int_B \varphi^2 p\nabla\cdot u\,dx \\ + \int_B G_c\big(\frac{1}{2\varepsilon}(1-\varphi)^2 + \frac{\varepsilon}{2}|\nabla\varphi|^2\big)\,dx. \end{aligned} \tag{105}$$

Here W is an energy storage functional - as before.

Remark 13.5. *The extension in which p is also an unknown was proposed and mathematically analyzed in [123, 124].*

We differentiate functional (105) with respect to both solution variables u and φ to obtain the Euler-Lagrange equations. The resulting variational formulation is stated in an incremental (i.e., time-discretized) formulation in which the continuous irreversibility constraint is approximated by

$$\varphi \le \varphi^{old}.$$

Here, φ^{old} will later denote the previous time step solution and φ the current solution. Let $V := H^1_0(B)$ and

$$W_{in} := \{w\in H^1(B)|\, w\le\varphi^{old}\le 1 \text{ a.e. on } B\}$$

be the function spaces we work with here; and for later purposes we also need $W := H^1(B)$.

The Euler-Lagrange system corresponding to the functional (105) then reads:

Formulation 13.6. *Let $p \in L^\infty(B)$ be given. Find $(u, \varphi) \in \{u_D + V\} \times W$ such that*

$$\Big(\big((1-\kappa)\varphi^2 + \kappa\big)\ \sigma(u), e(w)\Big) + (\varphi^2 p, \nabla \cdot w) = 0 \quad \forall w \in V, \tag{106}$$

and

$$\begin{aligned}(1-\kappa)(\varphi\ \sigma(u) : e(u), \psi{-}\varphi) + 2(\varphi\ p\ \nabla \cdot u, \psi{-}\varphi)& \\ + G_c\Big(-\frac{1}{\varepsilon}(1-\varphi, \psi{-}\varphi) + \varepsilon(\nabla\varphi, \nabla(\psi-\varphi))\Big) \geq 0&,\end{aligned} \tag{107}$$

for all $\psi \in W_{in} \cap L^\infty(B)$.

13.3.5 The strong form

In order to complete our derivations, we derive the strong form of Formulation 13.6 in this section. Using integration by parts, we obtain a quasi-stationary elliptic system for the displacements and the phase-field variable, where the latter one is subject to an inequality constraint in time:

Formulation 13.7. *Let $p : B \to \mathbb{R}$ be given. The displacement equation reads: Find $u : B \to \mathbb{R}^d$:*

$$\begin{aligned}-\nabla \cdot \Big(\big((1-\kappa)\varphi^2 + \kappa\big)\sigma\Big) - \nabla \cdot (\varphi^2 p) &= 0 \quad \text{in } B\\ u &= 0 \quad \text{on } \partial B,\\ \varphi^2 p n &= 0 \quad \text{on } \partial B.\end{aligned}$$

The phase-field system consists of three parts: the partial differential equation, the inequality constraint, and a compatibility condition (which is called Rice condition in the presence of fractures [149]): Find $\varphi : B \to [0, 1]$ such that

$$\begin{aligned}(1-\kappa)\sigma : e(u)\,\varphi - G_c\varepsilon\Delta\varphi - \frac{G_c}{\varepsilon}(1-\varphi) + 2\varphi p \nabla \cdot u &\leq 0 \quad \text{in } B,\\ \partial_t \varphi &\leq 0 \quad \text{in } B,\\ \Big[(1-\kappa)\sigma : e(u)\,\varphi - G_c\varepsilon\Delta\varphi - \frac{G_c}{\varepsilon}(1-\varphi) + 2\varphi p \nabla \cdot u\Big] \cdot \partial_t\varphi &= 0 \quad \text{in } B,\\ \partial_n \varphi &= 0 \quad \text{on } \partial B.\end{aligned}$$

14 Simulations IV: Parallel computing of a pressurized fracture

Another common example of a pressurized fracture is designed by Sneddon and Lowengrub [154]. Already in 1969 the y provided analytic solutions of a pressure-driven fracture in a two- or three-dimensional domain. In the following, results on parallel computations based on this Sneddon benchmark are given. Parallel computing is quite important on modern machines and it is important to check the scalability and efficiency of used numerical methods depending on the problem size or better the number of degrees of freedom (DoF).

14.1 Model problem

Using our derivations from Section 13.3, we are interested in working with the following energy functional:

$$\begin{aligned} E_\epsilon(u,\varphi) = \frac{1}{2}\Big(\big((1-\kappa)\varphi^2+\kappa\big)\,\sigma(u), e(u)\Big) + (\varphi^2 p, \text{div } u) + (\varphi^2 \nabla p, u) \\ + G_C \underbrace{\left(\frac{1}{2\epsilon}||1-\varphi||^2 + \frac{\epsilon}{2}||\nabla\varphi||^2\right)}_{\text{Hausdorff measure } \mathcal{H}^1(C)}. \end{aligned}$$

Here, we notice that we work with a given constant pressure $p : \Omega \to \mathbb{R}$, thus $\nabla p \equiv 0$. For convenience of the reader we recall that we consider the constraint

$$\partial_t \varphi \le 0, \quad \text{(crack irreversibility)}$$

where $\varphi : \Omega \to [0,1]$ is a continuous phase-field function with $\varphi = 0$ in the crack and $\varphi = 1$ in the unbroken material. Furthermore, we work with the following strain and stress tensors, respectively:

$$\begin{aligned} e(u) &:= \frac{1}{2}\left(\nabla u + \nabla u^T\right), \\ \sigma(u) &:= 2\mu e(u) + \lambda \text{tr}\left(e(u)\right) I. \end{aligned}$$

Due to the critical cross terms $\Big(\big((1-\kappa)\varphi^2+\kappa\big)\,\sigma(u), e(u)\Big)$ and $(\varphi^2 p, \text{div } u)$ we linearize (Section 8.5.2) as follows:

$$\varphi \approx \tilde\varphi := \tilde\varphi^n = \varphi^{n-2}\frac{t_n - t_{n-1}}{t_{n-2}-t_{n-1}} + \varphi^{n-1}\frac{t_n - t_{n-2}}{t_{n-1}-t_{n-2}},$$

to obtain a convex energy functional $E_\epsilon(u,\tilde\varphi)$:

$$\begin{aligned} E_\epsilon(u,\tilde\varphi) = \frac{1}{2}\Big(\big((1-\kappa)\tilde\varphi^2+\kappa\big)\,\sigma(u), e(u)\Big) + (\tilde\varphi^2 p, \text{div } u) \\ + G_C \left(\frac{1}{2\epsilon}||1-\tilde\varphi||^2 + \frac{\epsilon}{2}||\nabla\tilde\varphi||^2\right). \end{aligned}$$

14.2 Numerical solution algorithm

The numerical solution algorithm is based on our explanations in Chapter 7 and 8. This results into the algorithm sketched in Figure 46.

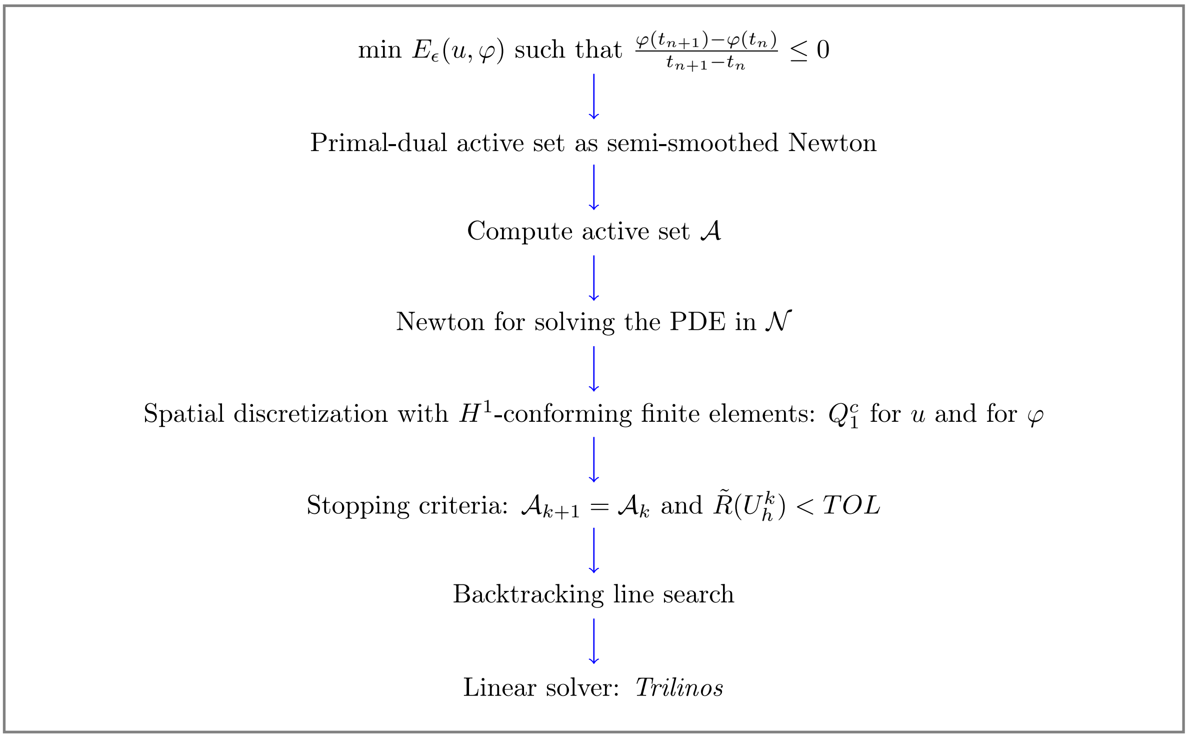
min $E_\epsilon(u,\varphi)$ such that $\frac{\varphi(t_{n+1})-\varphi(t_n)}{t_{n+1}-t_n} \leq 0$
Primal-dual active set as semi-smoothed Newton
Compute active set $\mathcal{A}$
Newton for solving the PDE in $\mathcal{N}$
Spatial discretization with H^1-conforming finite elements: Q_1^c for u and for φ
Stopping criteria: $\mathcal{A}_{k+1} = \mathcal{A}_k$ and $\tilde{R}(U_h^k) < TOL$
Backtracking line search
Linear solver: *Trilinos*

Figure 46: Solution process to simulate Sneddon's pressure-driven fracture.

To treat the variational inequality, we chose to use a primal-dual active set method which can be seen as a semi-smoothed Newton method. All details on the primal-dual active set strategy can be found in Section 8.17. Further, the used implementation is open-source code published in [83] based on deal.II [13]. To solve the PDE on the non-active set $\mathcal{N}$, a second Newton iteration is required. When necessary, globalization is achieved with a simple backtracking line search method. The final linear system is solved by a linear solver *Trilinos*.
For a stable discretization in space, we use H^1 conforming finite elements on quadrilaterals (2D). Specifically, we use bilinear elements Q_1^c for both, the displacements U and the phase-field variable φ. A detailed explanation of Q_1^c elements can be found in [45].

14.3 Problem setup - Sneddon/Lowengrub

Inspired by the analytical calculations made in [154], we propose a setting outlined in the following.

14.3.1 Configuration

The two-dimensional domain $\Omega = (0,4)^2$ contains an initial crack named C with $2l_0 = 0.4$ on $\Omega_c = (1.8, 2.2) \times (2-\delta, 2+\delta) \subset \Omega$, where $\delta \geq 0$ is a parameter to extend the initial crack slightly into the vertical direction in order to capture the initial crack with the corresponding discretization. We simply choose $\delta := h$, where h is the usual spatial discretization parameter.

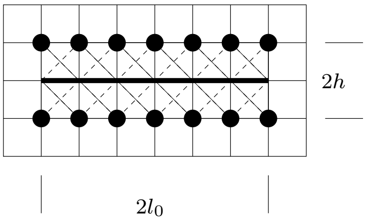

Figure 47: Zoom-in to the center of the domain Ω. The lower-dimensional crack with length $2l_0 = 0.4$ (middle line in black) is approximated as a volume by extending it with mesh size h in normal up- and down-directions.

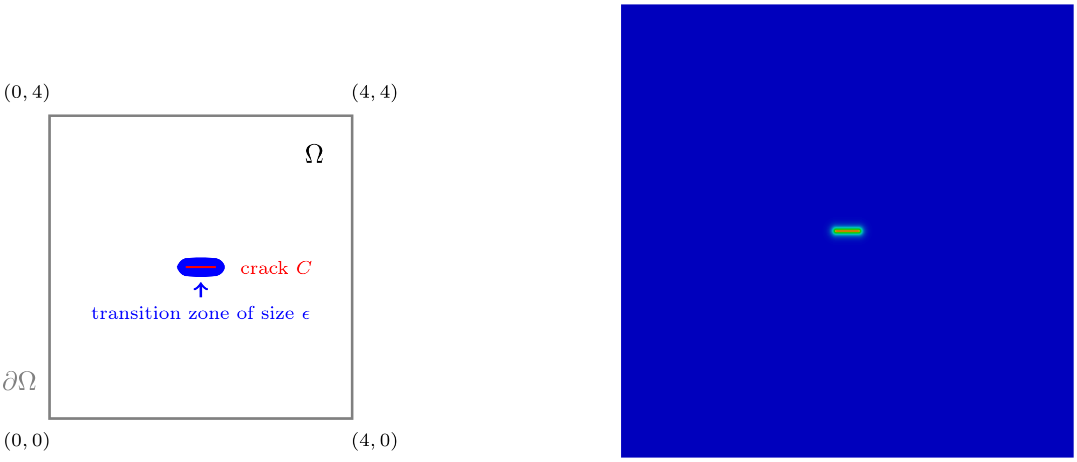

Figure 48: Left: Domain Ω (in 2D) with Dirichlet boundaries $\partial\Omega$, an initial crack C of length $2l_0$ and a transition zone of width ϵ between broken and unbroken material. Right: A snapshot of the phase-field function φ, $\varphi = 0$ in the crack (red colored) and $\varphi = 1$ in the unbroken material (blue colored).

14.3.2 Boundary conditions

On the boundary $\partial\Omega$, the displacements u are set to zero. For the phase-field variable φ, we use homogeneous Neumann conditions. So we set

$$\begin{aligned} u &= 0 \quad \text{on } \partial\Omega, \\ \partial_n \varphi &= 0 \quad \text{on } \partial\Omega \quad \text{(traction-free)}. \end{aligned}$$

14.3.3 Initial conditions

Because u is purely posed in a quasi-stationary manner, no initial condition is required. For the phase-field variable, we have a loading-dependency in the irreversibility constraint. As initial condition we set:

$$\begin{aligned} \varphi(0, x) &= 1 \quad \text{in } \Omega_c, \\ \varphi(0, x) &= 0 \quad \text{in } \Omega \setminus \Omega_c. \end{aligned}$$

14.3.4 Parameters

The mechanical and numerical parameter values are listed in Table 8.

14.3.5 Discretization and DoFs

As previously stated, we use Q_1/Q_1 elements on quadrilaterals. Specifically, we have five different test setups. Its problem size, the corresponding number of degrees of freedom (DoFs), the value of the discretization parameter h and the bandwidth of the crack ϵ are listed in Table 9.

Parameter	Definition	Value
Ω	Domain	$(0,4)^2$
l_0	Half crack length	0.2
α	Biot coefficient	0
G_c	Fracture toughness	1.0
E	Young's modulus	1.0
ν_s	Poisson's ratio	0.2
p_B	Injected pressure	10^{-3}
TOL_t	Tolerance time step loop	10^{-5}
ϵ	Bandwidth of the crack	$2h$
κ	Regularization parameter	$0.25\sqrt{h}$

Table 8: Setting of the benchmark parameters in 2D. The discretization parameter h varies depending on the current test, see Table 9

Test	DoFs	h	ϵ
global9	$789,507$	0.0110	0.0220
global10	$3,151,875$	0.0055	0.0110
global11	$12,595,203$	0.0028	0.0055
global12	$50,356,227$	0.0014	0.00280
adaptive (min.)	$3,045$	0.0110	0.0220
adaptive (max.)	$103,881$	0.0014	0.0027

Table 9: Number of degrees of freedom (DoFs) of each test case. global(n): n global refinement steps. Test case adaptive: 4 global, 5 local and 3 adaptive refinement steps.

14.3.6 Quantities of interest

We are interested in

- the crack opening displacement (COD):

$$\mathrm{COD}(x) := \int_0^4 u(x,y) \cdot \nabla\varphi(x,y)\; dy. \tag{108}$$

- the number of GMRES iterations;
- the number of Newton / active set iterations;
- performance of parallel computing in terms of scalability and speed-up.

14.3.7 Programming code

The programming code is based on slight and simply modifications of [83]. Therein, the implementation is realized in the open source software library deal.II ([13]), MPI, Trilinos ([84]), and p4est ([36])

14.3.8 Computing machine

The parallel runs are done on a single machine with four E7 v3 CPUs, so in total 64 cores are used.

14.4 Numerical results

In this section, we present numerical results with regard to the quantities of interest that we formulated in Section 14.3.6.

14.4.1 Solution plot and locally refined meshes

In Figure 49, an enlarged snapshot of the phase-field variable close to the fracture zone is given in the last pseudo-time step based on an adaptively refined mesh. The adaptive refinement is done by a predictor-corrector scheme as described in [82]. In Section 10.14, this strategy for local mesh adaptivity is discussed in detail.

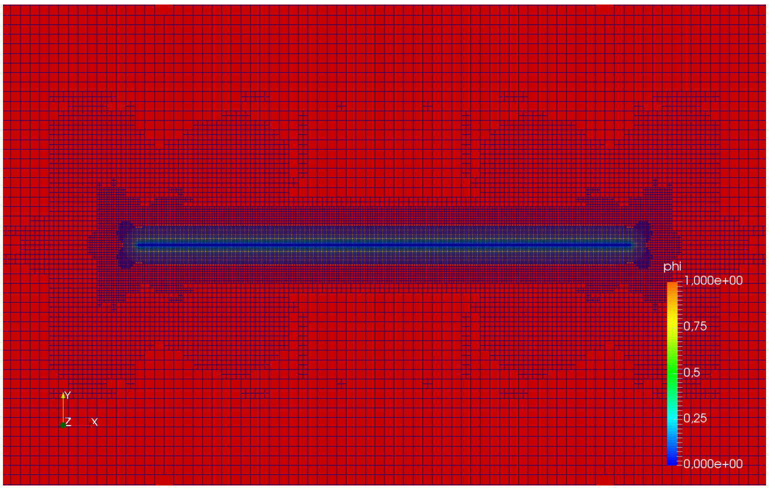

Figure 49: Test case adaptive: Last pseudo-time step with strong refinement close to the fracture. Predictor-corrector scheme for local mesh adaptivity.

14.4.2 Evaluation of the crack opening displacement

We first present our results for the COD defined in Equation (108). Figure 50 gives the crack opening displacement values for all tests listed in Table 9. The black line gives the exact crack opening displacement values calculated via the analytical solutions of the 2-dimensional problem given by Sneddon and Lowengrub [154].

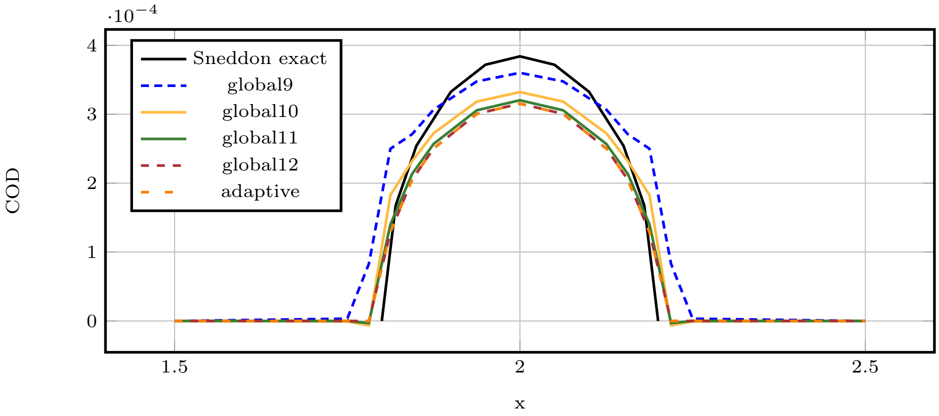

Figure 50: Crack opening displacement (COD) values for all different test cases listed in Table 9.

14.4.3 GMRES iterations per nonlinear (combined Newton) step

We want to discuss the number of required GMRES iterations per nonlinear step to show on one side, that parallel runs do not change the inner required iteration steps and on the other side, that via finer or adaptively refined meshes, the number of GMRES iterations or also the number of outer Newton iterations (see Section 14.4.4) can be reduced.

Test/Cores	1	2	4	8	16	32	64
global9	16.99	16.85	16.85	16.79	16.57	16.79	16.71
global10	16.53	16.53	16.47	16.47	16.33	16.31	16.31
global11	25.13	25.25	24.88	23.63	24.88	24.38	24.75
global12	–	38	37.33	38.83	38.67	39.17	40.16
adaptive (min.)	4.93	4.07	4.93	4.14	4	3.71	3.5
adaptive (max.)	18.33	18.83	18.66	17.83	17.67	17.5	17.67

Table 10: Average number of GMRES iterations per nonlinear Newton step for 5 different tests and different numbers of cores per loading step.

The average numbers of GMRES iterations per combined Newton step for all test cases of the Sneddon test with higher parallelization are given in Table 10 and plotted in Figure 51 in the left graph.

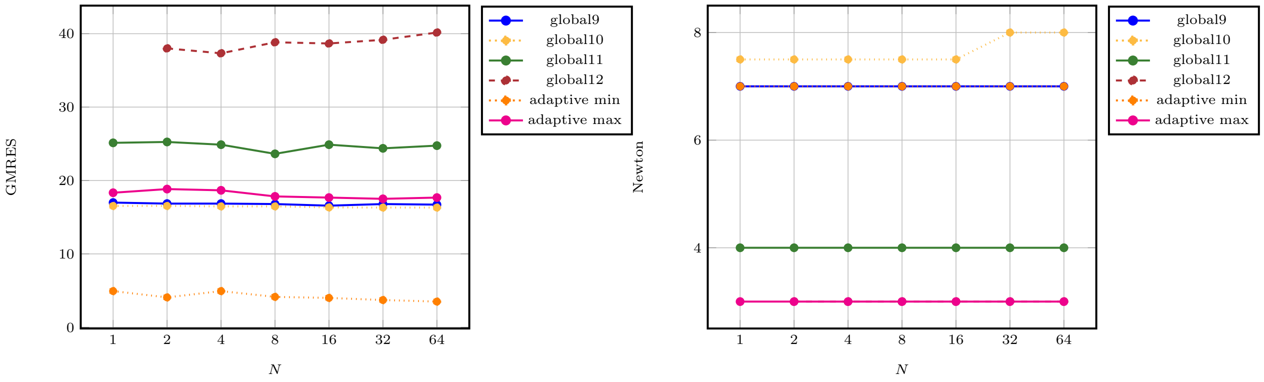

Figure 51: Left: Average number of GMRES iterations per nonlinear Newton step for 5 different tests and different numbers of cores per loading step. $N :=$ number of cores. Right: Average number of combined Newton iterations Newton step for 5 different tests and different numbers of cores per loading step. $N :=$ number of cores.

In Figure 51 in the left plot, the number of Newton intern GMRES iterations (listed in Table 10) is depicted.

14.4.4 Average number of combined Newton iterations per loading step

In Table 11 and Figure 51 on the right side, the average numbers of newton iterations per quasi-time step are given for all test setups and different levels of parallel computing.

We observe, that the number of necessary Newton iterations decreases with finer meshes while it seems to be independent of the current parallelization. Also in the test, based on adaptive refined mesh, we see that the via the locally refined mesh the number of Newton iterations can be decreased by more than the half (from 7 to 3).

14.4.5 Parallel computing

In the following we discuss the results of our parallel runs of Sneddon's test in 2D. The implementation mentioned above allows parallel computing.

Test/Cores	1	2	4	8	16	32	64
global9	7	7	7	7	7	7	7
global10	7.5	7.5	7.5	7.5	7.5	8	8
global11	4	4	4	4	4	4	4
global12	–	3	3	3	3	3	3
adaptive (min.)	7	7	7	7	7	7	7
adaptive (max.)	3	3	3	3	3	3	3

Table 11: Average number of combined Newton iterations Newton step for 5 different tests and different numbers of cores per loading step.

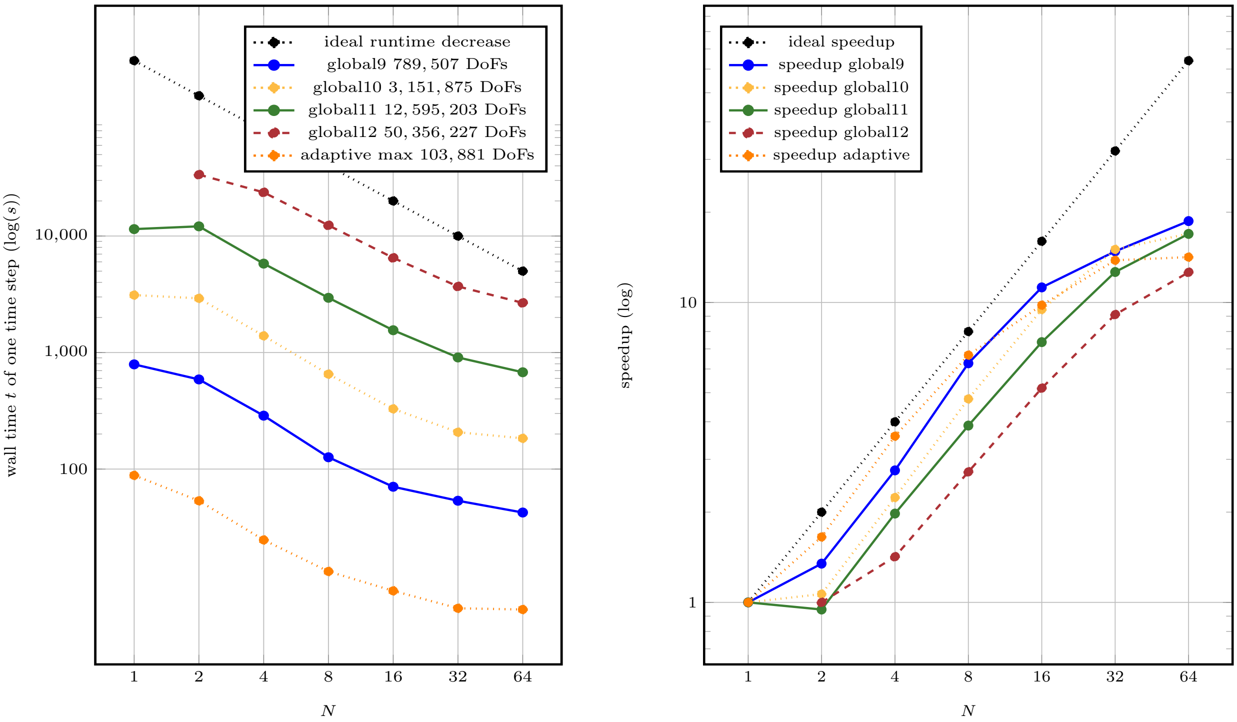

Figure 52: Left: $N :=$ number of cores, $t_N :=$ wall time of one time step per core with N used cores. Wall time of time step 1 per core plotted; in the test adaptive: Wall time of time step 7. Right: $N :=$ number of cores. Speedup is defined as $\frac{t_1}{t_N}$.

In Figure 52, on the left side, the total runtime of one pseudo-timestep of one core is given. We can observe, that the runtime decreases with a higher number of cores along the $x-$axis and the bigger the problem size, the worse the runtime per core. Further, it is notable that the bad run time difference of 1 to 2 cores can be explained by dynamic overclocking. And one has to keep in mind that we used 4 processors with 16 cores each, which can be the reason that the linear descent is slower on 32 and 64 cores because of the communication between the CPUs (coherent bus memory access).

Strong scalability is one quantity of interest, that want to be confirmed considering high performance computing. On the right side in Figure 52, the speedup is plotted in a logarithmic way.

In Figure 53, the efficiency is given as a percentage relative to the sequential runtime t_1. The red dashed line in the right plot looks very different, because t_1 is missing. One can say, that the speedup as well as the efficiency up to 8 or 16 cores is satisfying to assume strong scalablity, but with higher parallelization more

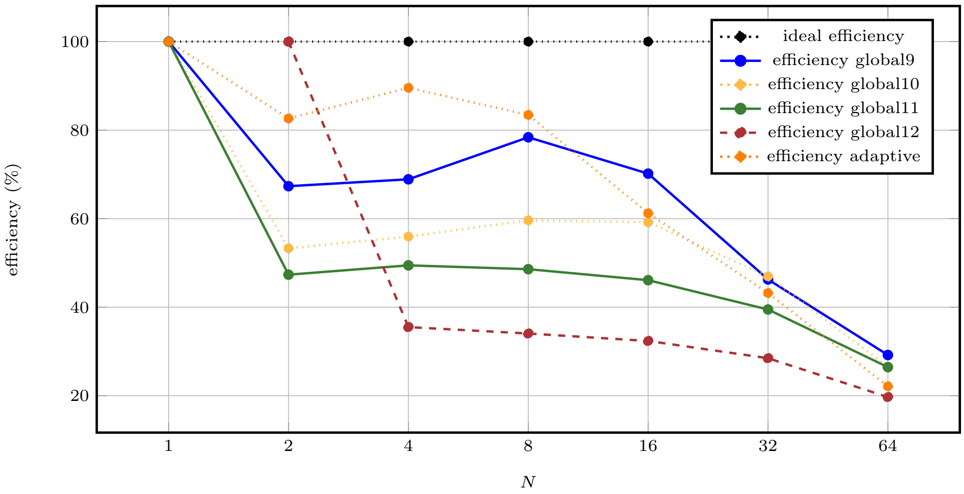

Figure 53: $N :=$ number of cores. Parallel scalability computed via $\frac{t_1}{N \cdot t_N} 100\%$.

tests and analysis of the used machine and memory accesses would be useful.

Let us shortly summarize the results of this section: Sneddon's benchmark is a practical test for fracture evolution using phase-field approach. Secondly, high performance computing is quite new in the context of phase-field fracture problems and the parallel results in the frame of this presentation confirm that the implementation of Sneddon's test in 2D is scalable and up to a explainable limit also efficient if the proposed solving strategy is used.

15 Practicing in DOpElib `Examples/PDE/InstatPDE/Example8`

In this chapter, we provide some hints and ideas for practical exercises. The underlying code is based on DOpElib [53, 72] `www.dopelib.net` `Examples/PDE/InstatPDE/Example8`. Within this example, the single edge notched shear test is computed; for the setup see Chapter 9 of the lecture notes on hand.

Remark 15.1. *Keep in mind, that practicing, programming, and testing will mainly start* ***after*** *the installation of deal.II and dopelib. So it is useful and important if you know what the software deal.II and DOpElib are used to, but the focus will be to understand one example in DOpElib in meticulous detail and to understand from there external methods or functions. Therefore we use the open source software as a very useful tool to apply what we learned in the last chapters and to see the numerical complexity behind the phase-field fracture model.*

15.1 Installation of deal.II

To install and use DOpElib, first deal.II [13, 16] needs to be installed. The most important installing steps are given in `https://www.dealii.org/9.0.0/readme.html` and the software can be downloaded here: `https://www.dealii.org/download.html`. Please download the current release version and not the most recent development version. If you have any trouble, pay attention, especially which compiler you use and which ones deal.II supports. Further, check the additional software requirements. If you want to deepen your understanding of deal.II, `http://www.math.colostate.edu/~bangerth/videos.html` provides a long list of video lectures by Prof. Wolfgang Bangerth. In particular, lecture 3 can be helpful for the installation of deal.II.

15.2 Installation of DOpElib

Please see `www.dopelib.net` `Documentation` and then `Manual` (version Aug 8, 2018). Therein, please see the Sections 2.6 - 2.8. There you also find a section about the installation of deal.II [13]. The version deal.II 9.0.0 has been successfully tested with DOpElib in September 2018. If you get any error, first check in Section 2.8 of the Manual (`http://wwwopt.mathematik.tu-darmstadt.de/dopelib/description_full.pdf`), if you recognize your error message in the listed errors. If you still have an error message, please do not hesitate to contact Katrin Mang `mang@ifam.uni-hannover.de`.

15.3 Building and running the example

The dirty and quick way is to go into `autobuild` and typing

```
> cmake ..
```

which creates a Makefile designed for an executable file in the parent subfolder. Compile via

```
> make
```

Then, we should obtain in the parent directory `DOpE-PDE-InstatPDE-Example8`, which can be run via typing

```
> ./DOpE-PDE-InstatPDE-Example8
```

Why now is this way dirty and quick? Because all paths are set and we can immediately run the example. The clean way would be NOT to TOUCH the directory `Example8` in order to keep a running and original version. Rather one would copy via

```
> cp -r Example8 /home/userNAME/Software/MySoftwareProjects/
```

into some local given directory, here `MySoftwareProjects/`, of the user.

In this case the path to DOpElib has to be set and one works with a Makefile directly in the main folder of the example. For detailed information, please see the DOpElib Manual Section 4.4.

Remark 15.2. *During the entire installation process of DOpElib and its examples, please make sure that the paths are set correctly. If the paths are only locally exported, in each new terminal window the paths need to be set again. This has been a frequent error source in the past and for this reason mentioned here. Setting the paths locally can be done with:*

```
export DOPE_DIR=path_to_main_directory_of_DOpElib
export DEAL_II_DIR=path_to_dealii/your_install_folder/
```

Remark 15.3 (Generating a Makefile using cmake)**.** *If no Makefile exists, we use the given* CMakeLists.txt *to generate a new local Makefile:*

```
MySoftwareProjects/Example8> cmake .
```

Be careful that the paths to deal.II and DOpElib are set correctly. In particular, make sure that you link to the same deal.II version (in case you have several) with which DOpElib was built. The latter can be found out by looking into CMakeCache.txt *either in one of the DOpElib examples or the main source files of the library.*

15.4 Files

In Example8 we find the following files:

```
autobuild       // The above mentioned autobuild folder
CMakeLists.txt  // File for cmake in order to create a new Makefile
Makefile        // Makefile (old DOpElib versions);
                // better remove it and create new one as described above
Test            // Needed for regression tests when changes in the library are done


localpde_quasi_monolithic.h  // 1st version of the PDE implementation
localpde_fully_implicit.h    // 2nd version of the PDE implementation
instat_step_modified_newtonsolver.h // A nonlinear Newton solver
stress_splitting.h           // Splitting the stress according to Miehe et al. 2010
problem_data.h               // Boundary data, etc.
functionals.h                // Evaluating functionals of interest, such as
                             // load-displacement curves
main.cc                      // The main file combining all data to run the example


dope.prm      // A parameter file
unit_slit.inp // The mesh file


DOpE-PDE-InstatPDE-Example8  // Executable file
Results                      // A results folder containing output data
```

15.5 Important files for working with the PDE and changing the program

```
instat_step_modified_newtonsolver.h // A nonlinear Newton solver

localpde_quasi_monolithic.h         // 1st version of the PDE implementation
localpde_fully_implicit.h           // 2nd version of the PDE implementation

problem_data.h                      // Boundary data, etc.
functionals.h                       // Evaluating functionals of interest, such as
                                    // load-displacement curves
main.cc                             // The main file combining all data to run the example

dope.prm                            // A parameter file
```

15.6 First changes to acquaint with the example

The easiest would be to change the mesh refinement level. Go into main.cc and search the line

```
triangulation.refine_global(4);
```

Change 4 to 3 or 5 or so. Another suggestion is to change the time step size in the line

```
GridGenerator::subdivided_hyper_cube(times, num_intervals, initial_time, end_time);
```

15.7 Examples of possible tasks in the course

15.7.1 Meeting 1

- Introduction, installation and first steps in DOpElib.
- How to run an example; see Section 15.3.
- Which files are in an example folder and how are they connected? See Section 15.4.

15.7.2 Meeting 2

- Running and understanding `Examples/StatPDE/Example4` (Laplace 2D) in DOpElib and/or `Examples/InstatPDE/Example4` (Heat 1D) to understand the numerical solution of simple partial differential equations.
- Studying in more detail the structure and contents in

  ```
  main.cc
  localpde_fully_implicit.h
  ```

- Trying to explain the equations implemented in the `localpde` in relation to Section 5.6.3.

15.7.3 Meeting 3

- Running Example 8 (phase-field fracture problem). Change parameters, the mesh size and time step size for example. Finding out where and how the stresses (functionals of interest are computed);
- Play with the penalization parameter γ; chose $\gamma = 0$ and (very) large $\gamma \gg 0$;
- Play with regularization parameter κ;
- Play with the regularization parameter ε;
- Use other time-stepping schemes.

15.7.4 Meeting 4 - 7 (Choice 1)

Copy Example 8 and change the parameter setting such that the problem setup is changed. Implementing another test case (e.g. l-shaped panel, e.g., [7] or single edge tension test, e.g., [117]).

15.7.5 Meeting 4 - 7 (Alternative 1, advanced)

Studying

```
instat_step_modified_newtonsolver.h // A nonlinear Newton solver
```

Implementation of the modified Newton scheme presented in Section 8.15.

15.7.6 Meeting 4 - 7 (Alternative 2, advanced)

Implementation of predictor-corrector mesh adaptivity as presented in Section 10.14. As alternative, a simple error estimator of Kelly-type could be implemented first as well; see e.g., `Examples/PDE/StatPDE/Example6`.

15.7.7 Meeting 4 - 7 (Alternative 3)

Pressurized fractures and total crack volume computations: a sequential version of Sneddon's test. The configuration and parameters are provided in Chapter 14.

15.7.8 Meeting 4 - 7 (Alternative 4, advanced)

Implement adaptive time step control as presented in Section 10.15.

15.7.9 Meeting 4 - 7 (Alternative 5, advanced)

Implement an alternate minimization algorithm, i.e., a partitioned approach to solve the displacement-phase-field system. Here, we solve for each variable separately and iterate between them. Alternating minimization has been widely used in the literature, e.g., [25, 35], and further instruction are given in the class if there is interest working on this exercise.

16 Computational convergence analysis

We provide some tools to perform a computational convergence analysis. In these notes we faced two situations of ‘convergence’:

- **Discretization error:** Convergence of the discrete solution u_h towards the (unknown) exact solution u;
- **Iteration error:** Convergence of an iterative scheme to approximate the discrete solution u_h through a sequence of approximate solutions $u_h^{(k)}, k = 1, 2, \ldots$.

In the following we further illustrate the terminologies ‘first order convergence’, ‘convergence of order two’, ’quadratic convergence’, ’linear convergence’, etc.

16.1 Discretization error

Before we go into detail, we discuss the relationship between the degrees of freedom (DoFs) N and the mesh size parameter h. In most cases the discretization error is measured in terms of h and all a priori and a posteriori error estimates are stated in a form

$$\|u - u_h\| = O(h^\alpha), \quad \alpha > 0.$$

In some situations it is, however, **better to create convergence plots in terms of DoFs vs. the error**. One example is when **adaptive schemes** are employed with different h. Then it would be not clear to which h the convergence plot should be drawn. But simply counting the total numbers of DoFs per refinement level is not a problem though.

16.1.1 Relationship between h and N (DoFs)

The relationship of h and N depends on the basis functions (linear, quadratic), whether a Lagrange method (only nodal points) or Hermite-type method (with derivative information) is employed. Moreover, the dimension of the problem plays a role.

We illustrate the relationship for a Lagrange method with linear basis functions in 1D,2D,3D:

Proposition 16.1. *Let d be the dimension of the problem: $d = 1, 2, 3$. It holds*

$$N = \Big(\frac{1}{h} + 1\Big)^d$$

where h is the mesh size parameter (length of an element or diameter in higher dimensions for instance), and N the number of DoFs.

Proof. Sketch. No strict mathematical proof. We initialize as follows:

- 1D: 2 values per line;
- 2D: 4 values per quadrilaterals;
- 3D: 8 values per hexahedra.

Of course, for triangles or prisms, we have different values in 2D and 3D. We work on the unit cell with $h = 1$. All other h can be realized by just normalizing h. By simple counting the nodal values, we have in 1D

```
h    N
=======
1    2
1/2  3
1/4  5
1/8  9
1/16 17
```

```
1/32 33
...
=======
```

We have in 2D

```
h    N
=======
1    4
1/2  9
1/4  25
1/8  36
1/16 49
1/32 64
...
=======
```

We have in 3D

```
h    N
=======
1    8
1/2  27
1/4  64
...
=======
```

□

16.1.2 Discretization error

With the previous considerations, we got a relationship between h and N that we can use to display the discretization error.

Proposition 16.2. *In the approximate limit it holds:*

$$N \sim \Big(\frac{1}{h}\Big)^d$$

yielding

$$h \sim \frac{1}{\sqrt[d]{N}}$$

These relationships allow us to replace h in error estimates by N.

Proposition 16.3 (Linear and quadratic convergence in 1D)**.** *If we say a scheme has a linear or quadratic convergence in 1D, (i.e., $d = 1$) respectively, we mean:*

$$O(h) = O\Big(\frac{1}{N}\Big)$$

or

$$O(h^2) = O\Big(\frac{1}{N^2}\Big)$$

In a linear scheme, the error will be divided by a factor of 2 when the mesh size h is divided by 2 and having quadratic convergence the error will decrease by a factor of 4.

Proposition 16.4 (Linear and quadratic convergence in 2D)**.** *When we say a scheme has a linear or quadratic convergence in 2D, (i.e., $d = 2$) respectively, we mean:*

$$O(h) = O\Big(\frac{1}{\sqrt{N}}\Big)$$

or

$$O(h^2) = O\Big(\frac{1}{N}\Big)$$

16.1.3 Computationally-obtained convergence order: example for time step sizes k

In order to calculate the convergence order α from numerical results, we make the following derivation. Let $P(k) \to P$ for $k \to 0$ be a converging process and assume that

$$P(k) - \tilde{P} = O(k^\alpha).$$

Here $\tilde{P}$ is either the exact limit P (in case it is known) or some 'good' approximation to it. Let us assume that three numerical solutions are known (this is the minimum number if the limit P is not known). That is

$$P(k), \quad P(k/2), \quad P(k/4).$$

Then, the convergence order can be calculated via the formal approach $P(k) - \tilde{P} = ck^\alpha$ with the following formula:

Proposition 16.5 (Computationally-obtained convergence order)**.** *Given three numerically-obtained values $P(k), P(k/2)$ and $P(k/4)$, the convergence order can be estimated as:*

$$\alpha = \frac{1}{log(2)} log\Big(\Big|\frac{P(k) - P(k/2)}{P(k/2) - P(k/4)}\Big|\Big). \tag{109}$$

The order α is an estimate and heuristic because we assumed a priori a given order, which strictly speaking we have to proof first.

Proof. We assume:

$$\begin{aligned} P(k) - P(k/2) &= O(k^\alpha), \\ P(k/2) - P(k/4) &= O((k/2)^\alpha). \end{aligned}$$

First, we have

$$P(k/2) - P(k/4) = O((k/2)^\alpha) = \frac{1}{2^\alpha} O(k^\alpha)$$

We simply re-arrange:

$$\begin{aligned} P(k/2) - P(k/4) &= \frac{1}{2^\alpha}\Big(P(k) - P(k/2)\Big) \\ \Rightarrow \quad 2^\alpha &= \frac{P(k) - P(k/2)}{P(k/2) - P(k/4)} \\ \Rightarrow \quad \alpha &= \frac{1}{log(2)} \frac{P(k) - P(k/2)}{P(k/2) - P(k/4)} \end{aligned}$$

□

In the following we present results (for all details, we refer the reader to [176]) for the (absolute) end time error of an ODE problem (but it could be any other PDE problem as well) on three mesh levels (different time step sizes k) with three schemes (FE - forward Euler, BE - backward Euler, CN - Crank-Nicolson):

```
Scheme         #steps  k     Absolute error at end time T
===========================================================
FE err.:       8       0.36  0.13786
BE err.:       8       0.36  0.16188
CN err.:       8       0.36  0.0023295
FE err.:       16      0.18  0.071567
BE err.:       16      0.18  0.077538
CN err.:       16      0.18  0.00058168
FE err.:       32      0.09  0.036483
BE err.:       32      0.09  0.037974
CN err.:       32      0.09  0.00014538
===========================================================
```

The absolute error at the end time T in the forth column is computed as

$$e_N = |y(T) - y_N|,$$

where $y(T)$ is the exact solution (which was known in this case) and y_N the numerical approximation at the end time value at the final numerical step N.

In order to compute numerically the convergence order α with the help of formula (109), we work with $k = k_{max} = 0.36$. Then we identify in the above table that $P(k_{max}) = P(0.36) = |y(T) - y_8|, P(k_{max}/2) = P(0.18) = |y(T) - y_{16}|$ and $P(k_{max}/4) = P(0.09) = |y(T) - y_{32}|$. We monitor that doubling the number of intervals (i.e., halving the step size k) reduces the error in the forward and backward Euler scheme by a factor of 2. This is (almost) linear convergence, which is confirmed by using Formula (109) yielding $\alpha = 0.91804$. The CN scheme is much more accurate (for instance using $n = 8$ the error is 0.2% rather than $13 - 16\%$) and we observe that the error is reduced by a factor of 4. Thus quadratic convergence is detected. Here the 'exact' order on these three mesh levels is $\alpha = 1.9967$.

16.1.4 Computationally-obtained convergence order: example for the spatial mesh parameter h

We explain two things in this section:

- How to construct a given right hand side f such that for a simple problem a manufactured solution holds true.
- As before: determining the convergence order via

$$\alpha = \frac{1}{log(2)} log\Big(\Big| \frac{P(h) - P(h/2)}{P(h/2) - P(h/4)} \Big| \Big). \tag{110}$$

Algorithm 16.6. *Given a PDE problem. E.g. $-\Delta u = f$ in Ω and $u = 0$ on the boundary $\partial\Omega$.*

1. *Construct by hand a solution u that fulfills the boundary conditions.*
2. *Insert u into the PDE to determine f.*
3. *Use that f in the finite element simulation to compute numerically u_h.*
4. *Compare $u - u_h$ in relevant norms (e.g., L^2, H^1).*
5. *Check whether the desired h powers can be obtained for small h.*

We demonstrate the previous algorithm for a $2D$ case in $\Omega = (0, \pi)^2$:

$$\begin{aligned} -\Delta u(x, y) &= f \quad \text{in } \Omega, \\ u(x, y) &= 0 \quad \text{on } \partial\Omega, \end{aligned}$$

and we construct $u(x, y) = \sin(x)\sin(y)$, which fulfills the boundary conditions (trivial to check, but please do it). Next, we compute the right hand side f:

$$-\Delta u = -(\partial_{xx} u + \partial_{yy} u) = 2sin(x)sin(y) = f(x, y).$$

We then use this f in the above program and evaluate the L^2 and H^1 norms. The results are:

```
Level   DoFs    h             L2 err          H1 err
===========================================================
2       16      1.11072       0.0948434       0.510265
3       64      0.55536       0.0238378       0.252641
4       256     0.27768       0.00596821      0.126015
5       1024    0.13884       0.00149261      0.0629697
6       4096    0.06942       0.000373189     0.0314801
7       16384   0.03471       9.32994e-05     0.0157395
```

```
8         65536    0.017355        2.3325e-05      0.00786965
9         262144   0.00867751      5.83126e-06     0.00393482
10        1048576  0.00433875      1.45781e-06     0.00196741
11        4194304  0.00216938      3.64451e-07     0.000983703
==============================================================
```

In this table we observe that we have quadratic convergence in the L^2 norm and linear convergence in the H^1 norm.

16.2 Iteration error

Iterative schemes are used to approximate the discrete solution u_h. This has a priori nothing to do with the discretization error. The main interest is how fast can we get a good approximation of the discrete solution u_h. One example can be found for solving implicit methods for ODEs in which Newton's method is used to compute the discrete solutions of the backward Euler scheme.

To speak about convergence, we compare two subsequent iterations:

Proposition 16.7. *Let us assume that we have an iterative scheme to compute a root z. The iteration converges with order p when*

$$\|x_k - z\| \leq c\, \|x_{k-1} - z\|^p, \quad k = 1, 2, 3, \ldots$$

with $p \geq 1$ and $c = const$. In more detail:

- *Linear convergence: $c \in (0, 1)$ and $p = 1$;*
- *Superlinear convergence: $c := c_k \to 0,\ (k \to \infty)$ and $p = 1$;*
- *Quadratic convergence $c \in \mathbb{R}$ and $p = 2$.*

Cupic and higher convergence are defined as quadratic convergence with the respective p.

Remark 16.8 (Other characterizations of superlinear and quadratic convergence)**.** *Other (but equivalent) formulations for superlinear and quadratic convergence, respectively, in the case $z \neq x_k$ for all k, are:*

$$\lim_{k\to\infty} \frac{\|x_k - z\|}{\|x_{k-1} - z\|} = 0,$$
$$\limsup_{k\to\infty} \frac{\|x_k - z\|}{\|x_{k-1} - z\|^2} < \infty.$$

Corollary 16.9 (Rule of thumb)**.** *A rule of thumb for quadratic convergence is: the number of correct digits doubles at each step. For instance, a Newton scheme to compute $f(x) = x - \sqrt{2} = 0$ yields the following results:*

```
Iter x            f(x)
==============================
0    3.000000e+00 7.000000e+00
1    1.833333e+00 1.361111e+00
2    1.462121e+00 1.377984e-01
3    1.414998e+00 2.220557e-03
4    1.414214e+00 6.156754e-07
5    1.414214e+00 4.751755e-14
==============================
```

17 Wrap-up - Quiz

In this final section, we consolidate what we learned through some questions:

1. Formulate a simplified phase-field fracture problem (Laplacian, scalar-valued).
2. Given the simplified strong form, derive a weak formulation.
3. How is the energy formulation given?
4. Describe in words the idea of phase-field fracture.
5. Please give some own examples in which damage or fracture arise in nature, research or daily life.
6. Interpret the crack irreversibility constraint physically and give a relationship to the obstacle problem.
7. What is the difference between quasi-static and dynamic fracture modeling?
8. What are solution algorithms to solve a coupled problem?
9. What are Euler-Lagrange equations?
10. What is a ‘stationary’ point?
11. How can we decouple the displacement/phase-field problem? What do we observe?
12. Characterize the phase-field fracture problem in terms of the classifications made in Chapter 6.
13. Formulate Newton’s method for solving the phase-field fracture problem.
14. Using a monolithic formulation for phase-field fracture, why do classical Newton methods have difficulties?
15. Take a monolithic formulation and do a mathematical manipulation in order to linearize the equations.
16. Give a simplified numerical analysis (key ideas) about the relationship between h and ε.
17. Which part(s) of the spatial discretization is(/are) tricky in phase-field fracture?
18. Formulate a fixed-point (staggered) iteration.
19. What are the possibilities to improve the efficiency of the numerical solution?
20. Why is goal-oriented mesh refinement useful?
21. Formulate the basic idea of goal-oriented mesh refinement using adjoint-based a posteriori error estimation (DWR).
22. Provide a brief idea of Griffth’s model.
23. What was the achievement of Francfort and Marigo in 1998?
24. Given a PDE on the continuous level, what are the key steps in the numerical solution process?
25. Explain one method to treat the crack irreversibility constraint numerically.
26. What are typical difficulties when using (simple) penalization for regularizing an inequality constraints?
27. What are the active and non-active sets in the obstacle problem?
28. What are challenges (or drawbacks) in using a phase-field method for fracture propagation? Please give one or two examples.
29. What is predictor-corrector mesh adaptivity?
30. What is a pressurized fracture?

18 Outlook

18.1 Extensions, applications and further literature

The main purpose of these lectures notes was to recapitulate the initial developments and arguments to design a variational model for treating fracture and damage in elasticity.

To date the approach has been extended in various directions. An exemplary list (stand June 2019) is:

1. Mathematical modeling and analysis [28, 41, 42, 47, 48, 64, 65]
2. Analysis and computations on crack nucleation [43, 164][theory], [170, 177][screw tests], [7, 66, 115, 175][L-shaped panel]
3. Other discretization techniques: IGA, e.g., [24], special basis functions [102], discontinuous Galerkin [57, 127]
4. Coupling VPFF with XFEM [70]
5. Shape optimization methods to formulate phase-field fracture [4].
6. Different formulations of the crack regularization (Ambrosio-Tortorelli) functional [30, 138] and [61][pages 649-650]
7. Investigation of degradation functions [151]
8. Optimal control, e.g., [130, 131]
9. Spatial mesh adaptivity [14, 15, 35, 82, 106, 173, 182]
10. Time step adaptivity and multirate coupling [174, 185] and [5]
11. Linear solvers, e.g., [61, 83, 93, 100]
12. Massive parallel solution, e.g., [83]
13. Nonlinear monolithic solvers (partially contained in these notes) [66, 175, 177] and partitioned (staggered) approaches/alternating minimization [25, 27, 34, 35, 115, 117].
14. Higher-order phase-field models [23, 85]
15. Dynamic fracture [24, 29, 88, 104, 105, 152]
16. Fracture with thermal / temperature interaction, e.g., [30, 120, 135].
17. Plasticity, ductile fracture, cohesive fracture, e.g., [2, 6, 8, 52, 103, 116, 153, 162, 166, 167].
18. Anisotropic phase-field fracture formulations [21, 157]
19. Towards phase-field formulations for nearly incompressible solids [112]
20. Crack growth along interfaces [78]
21. Crack width computations [107, 132]
22. Pressure-driven fracture, e.g., [26, 122, 125, 169].
23. Fluid-driven pressure, e.g., [80, 81, 118, 119, 124, 184].
24. Coupling to fluid-structure interaction [172] and [174].
25. VPFF as a small-scale model (a well model) inside large-scale setting [180] (preliminary studies; further studies necessary)
26. Towards multiscale formulations (global-local approach): [68, 134].

18.2 Some open questions

We finally list a few open questions:

1. Rigorous numerical analysis for $\varepsilon \to 0$ and $h \to 0$; and $\kappa \to 0$ and $h \to 0$.
2. Further validation computations and comparison between experiments and numerical simulations
3. Further analysis and simulations of limiting processes as for instance fracture nucleation
4. Further improvements of nonlinear solvers
5. Robust fracture width computations (some work done so far)
6. Further understanding of multiphysics fracture: how can we achieve accurate and robust realizations of interface conditions in the smoothed transition zone?

19 The end

The first version of these notes were developed in the beautiful spring in the year 2018 in Linz (Austria). While sitting and writing, watching out of the window of Raabheim's Turmzimmer, feelings of happiness[4, 5] can be best described by the following poem:

Eduard Mörike (1804-1875), deutscher Lyriker:

Er ist's

Frühling lässt sein blaues Band
Wieder flattern durch die Lüfte;
Süsse, wohlbekannte Düfte
Streifen ahnungsvoll das Land.
Veilchen träumen schon,
Wollen balde kommen.
–Horch, von fern ein leiser Harfenton!
Frühling, ja du bist's!
Dich hab' ich vernommen!

[4] Dear Jeremi (Mizerski), even that I knew works by Richard P. Feynman (e.g., the Feynman Lectures on Physics), I was not aware of this beautiful book 'The Pleasure of Finding Things Out' [62]. Feynman's 'pleasure' came to me while writing parts of these notes. Thanks for recommending when I was visiting you in Warsaw and Zamość! Best regards, Thomas W.

[5] M.D. Ph.D. Jeremi Mizerski, Warsaw/Poland, was the scientific supervisor from the Poland-side within the international graduate college IGK 710 Heidelberg-Warsaw (running until Dec 2009) during the PhD studies Nov 2008 - Dec 2011 of the second author, T. Wick.

References

[1] M. Ainsworth, J. Oden, and C. Lee. Local a posteriori error estimation for variational inequalities. *Numerical methods for partial differential equations*, 9:23–33, 1993.

[2] R. Alessi, J.-J. Marigo, C. Maurini, and S. Vidoli. Coupling damage and plasticity for a phase-field regularisation of brittle, cohesive and ductile fracture: One-dimensional examples. *International Journal of Mechanical Sciences*, 2017.

[3] G. Allaire. A review of adjoint methods for sensitivity analysis, uncertainty quantification and optimizarion in numerical codes. *Ingénieurs de l'Automobile, SIA*, pages 33–36, 2015.

[4] G. Allaire, F. Jouve, and N. V. Goethem. Damage and fracture evolution in brittle materials by shape optimization methods. *Journal of Computational Physics*, 230(12):5010 – 5044, 2011.

[5] T. Almani, S. Lee, M. Wheeler, and T. Wick. Multirate coupling for flow and geomechanics applied to hydraulic fracturing using an adaptive phase-field technique. SPE RSC 182610-MS, Feb. 2017, Montgomery, Texas, USA, 2017.

[6] M. Ambati, T. Gerasimov, and L. De Lorenzis. Phase-field modeling of ductile fracture. *Computational Mechanics*, 55(5):1017–1040, 2015.

[7] M. Ambati, T. Gerasimov, and L. De Lorenzis. A review on phase-field models of brittle fracture and a new fast hybrid formulation. *Computational Mechanics*, 55(2):383–405, 2015.

[8] M. Ambati and L. D. Lorenzis. Phase-field modeling of brittle and ductile fracture in shells with isogeometric nurbs-based solid-shell elements. *Computer Methods in Applied Mechanics and Engineering*, 312:351 – 373, 2016. Phase Field Approaches to Fracture.

[9] L. Ambrosio and V. Tortorelli. On the approximation of free discontinuity problems. *Bollettino dell'unione matematica italiana B.*, 6(1):105–123, 1992.

[10] L. Ambrosio and V. M. Tortorelli. Approximation of functional depending on jumps by elliptic functional via t-convergence. *Communications on Pure and Applied Mathematics*, 43(8):999–1036, 1990.

[11] H. Amor, J.-J. Marigo, and C. Maurini. Regularized formulation of the variational brittle fracture with unilateral contact: Numerical experiments. *J. Mech. Phys. Solids*, 57:1209–1229, 2009.

[12] J. Andersson and H. Mikayelyan. The asymptotics of the curvature of the free discontinuity set near the cracktip for the minimizers of the Mumford-Shah functional in the plain. a revision. arXiv: 1205.5328v2, 2015.

[13] D. Arndt, W. Bangerth, D. Davydov, T. Heister, L. Heltai, M. Kronbichler, M. Maier, J.-P. Pelteret, B. Turcksin, and D. Wells. The deal.II library, version 8.5. *Journal of Numerical Mathematics*, 2017.

[14] M. Artina, M. Fornasier, S. Micheletti, and S. Perotto. Anisotropic mesh adaptation for crack detection in brittle materials. *SIAM J. Sci. Comput.*, 37(4):B633–B659, 2015.

[15] H. Badnava, M. A. Msekh, E. Etemadi, and T. Rabczuk. An h-adaptive thermo-mechanical phase field model for fracture. *Finite Elements in Analysis and Design*, 138:31 – 47, 2018.

[16] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – a general purpose object oriented finite element library. *ACM Trans. Math. Softw.*, 33(4):24/1–24/27, 2007.

[17] W. Bangerth and R. Rannacher. *Adaptive Finite Element Methods for Differential Equations*. Birkhäuser, Lectures in Mathematics, ETH Zürich, 2003.

[18] R. Becker, C. Johnson, and R. Rannacher. Adaptive error control for multigrid finite element methods. *Computing*, 55:271–288, 1995.

[19] R. Becker and R. Rannacher. A feed-back approach to error control in finite element methods: basic analysis and examples. *East-West J. Numer. Math.*, 4:237–264, 1996.

[20] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica, Cambridge University Press*, pages 1–102, 2001.

[21] J. Bleyer and R. Alessi. Phase-field modeling of anisotropic brittle fracture including several damage mechanisms. *Computer Methods in Applied Mechanics and Engineering*, 2018. https://doi.org/10.1016/j.cma.2018.03.012.

[22] A. Bonnet and G. David. *Cracktip is a global Mumford-Shah minimizer.* Asterisque No. 274, 2001.

[23] M. J. Borden, T. J. Hughes, C. M. Landis, and C. V. Verhoosel. A higher-order phase-field model for brittle fracture: Formulation and analysis within the isogeometric analysis framework. *Computer Methods in Applied Mechanics and Engineering*, 273(0):100 – 118, 2014.

[24] M. J. Borden, C. V. Verhoosel, M. A. Scott, T. J. R. Hughes, and C. M. Landis. A phase-field description of dynamic brittle fracture. *Comput. Meth. Appl. Mech. Engrg.*, 217:77–95, 2012.

[25] B. Bourdin. Numerical implementation of the variational formulation for quasi-static brittle fracture. *Interfaces and free boundaries*, 9:411–430, 2007.

[26] B. Bourdin, C. Chukwudozie, and K. Yoshioka. A variational approach to the numerical simulation of hydraulic fracturing. SPE Journal, Conference Paper 159154-MS, 2012.

[27] B. Bourdin, G. Francfort, and J.-J. Marigo. Numerical experiments in revisited brittle fracture. *J. Mech. Phys. Solids*, 48(4):797–826, 2000.

[28] B. Bourdin, G. Francfort, and J.-J. Marigo. The variational approach to fracture. *J. Elasticity*, 91(1–3):1–148, 2008.

[29] B. Bourdin, C. Larsen, and C. Richardson. A time-discrete model for dynamic fracture based on crack regularization. *Int. J. Frac.*, 168(2):133–143, 2011.

[30] B. Bourdin, J.-J. Marigo, C. Maurini, and P. Sicsic. Morphogenesis and propagation of complex cracks induced by thermal shocks. *Phys. Rev. Lett.*, 112:014301, Jan 2014.

[31] M. Braack and A. Ern. A posteriori control of modeling errors and discretization errors. *Multiscale Model. Simul.*, 1(2):221–238, 2003.

[32] D. Braess. *Finite Elemente.* Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, vierte, überarbeitete und erweiterte edition, 2007.

[33] A. Braides. *Approximation of free-discontinuity problems.* Number 1694 in Lecture notes in Mathematics. Springer Science & Business Media, 1998.

[34] M. Brun, T. Wick, I. Berre, J. Nordbotten, and F. Radu. An iterative staggered scheme for phase field brittle fracture propagation with stabilizing parameters. arXiv preprint arXiv:1903.08717, March 2019.

[35] S. Burke, C. Ortner, and E. Süli. An adaptive finite element approximation of a variational model of brittle fracture. *SIAM J. Numer. Anal.*, 48(3):980–1012, 2010.

[36] C. Burstedde, L. C. Wilcox, and O. Ghattas. P4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM J. Sci. Comput.*, 33(3):1103–1133, May 2011.

[37] X.-C. Cai and D. E. Keyes. Nonlinearly preconditioned inexact newton algorithms. *SIAM Journal on Scientific Computing*, 24(1):183–200, 2002.

[38] G. F. Carey and J. T. Oden. *Finite Elements. Volume III. Compuational Aspects.* The Texas Finite Element Series, Prentice-Hall, Inc., Englewood Cliffs, 1984.

[39] C. Carstensen, M. Feischl, M. Page, and D. Praetorius. Axioms of adaptivity. *Computers and Mathematics with Applications*, 67(6):1195 – 1253, 2014.

[40] C. Carstensen and R. Verfürth. Edge residuals dominate a posteriori error estimates for low order finite element methods. *SIAM J. Numer. Anal.*, 36(5):1571–1587, 1999.

[41] A. Chambolle, S. Conti, and F. Iurlano. Approximation of functions with small jump sets and existence of strong minimizers of griffith's energy. *Journal de MathÃ©matiques Pures et AppliquÃ©es*, 128:119 – 139, 2019.

[42] A. Chambolle and V. Crismale. Existence of strong solutions to the Dirichlet problem for the Griffith energy. arXiv preprint arXiv:1811.07147v1, Nov 2018.

[43] A. Chambolle, A. Giacomini, and M. Ponsiglione. Crack initiation in brittle materials. *Arch. Ration. Mech. Anal.*, 188:309–349, 2008.

[44] P. G. Ciarlet. *Mathematical Elasticity. Volume 1: Three Dimensional Elasticity.* North-Holland, 1984.

[45] P. G. Ciarlet. *The finite element method for elliptic problems.* North-Holland, Amsterdam [u.a.], 2. pr. edition, 1987.

[46] P. G. Ciarlet. *Linear and Nonlinear Functional Analysis with Applications.* SIAM, 2013.

[47] G. dal Maso, G. A. Francfort, and R. Toader. Quasistatic crack growth in nonlinear elasticity. *Arch. Ration. Mech. Anal.*, 176:165–225, 2005.

[48] G. dal Maso and R. Toader. A model for the quasistatic growth of brittle fractures: existence and approximation results. *Arch. Ration. Mech. Anal.*, 162:101–135, 2002.

[49] R. Dautray and J.-L. Lions. *Mathematical Analysis and Numerical Methods for Science and Technology*, volume 5. Springer-Verlag, Berlin-Heidelberg, 2000.

[50] R. Dembo, S. Eisenstat, and T. Steihaug. Inexact newton methods. *SIAM J. Numer. Anal.*, 19(2):400–408, 1982.

[51] P. Deuflhard. *Newton Methods for Nonlinear Problems*, volume 35 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, 2011.

[52] M. Dittmann, F. Aldakheel, J. Schulte, P. Wriggers, and C. Hesch. Variational phase-field formulation of non-linear ductile fracture. *Computer Methods in Applied Mechanics and Engineering*, 342:71 – 94, 2018.

[53] The Differential Equation and Optimization Environment: DOpElib. `http://www.dopelib.net`.

[54] W. Dörfler. A convergent adaptive algorithm for poisson's equation. *SIAM J. Numer. Anal.*, 33(3):1106–1124, 1996.

[55] J. W. Eaton, D. Bateman, S. Hauberg, and R. Wehbring. *GNU Octave version 3.8.1 manual: a high-level interactive language for numerical computations.* CreateSpace Independent Publishing Platform, 2014. ISBN 1441413006.

[56] B. Endtmayer, U. Langer, and T. Wick. Multigoal-oriented error estimates for non-linear problems. *Journal of Numerical Mathematics*, 2018.

[57] C. Engwer and L. Schumacher. A phase field approach to pressurized fractures using discontinuous galerkin methods. *Mathematics and Computers in Simulation*, pages –, 2016.

[58] A. Ern and M. VohralÃk. Adaptive inexact newton methods with a posteriori stopping criteria for nonlinear diffusion pdes. *SIAM Journal on Scientific Computing*, 35(4):A1761–A1791, 2013.

[59] L. C. Evans. *Partial differential equations.* American Mathematical Society, 2010.

[60] L. Failer and T. Wick. Adaptive time-step control for nonlinear fluid-structure interaction. *Journal of Computational Physics*, 366:448 – 477, 2018.

[61] P. E. Farrell and C. Maurini. Linear and nonlinear solvers for variational phase-field models of brittle fracture. *Int. J. Numer. Meth. Engrg.*, 109:648–667, 2017.

[62] R. P. Feynman. *The Pleasure of Finding Things Out.* Helix Books, Perseus Books, Cambridge Massachusetts, 1999.

[63] T. Fliessbach. *Mechanik*. Spektrum Akademischer Verlag, 2007.

[64] G. Francfort and J.-J. Marigo. Revisiting brittle fracture as an energy minimization problem. *J. Mech. Phys. Solids*, 46(8):1319–1342, 1998.

[65] G. A. Francfort and C. J. Larsen. Existence and convergence for quasi-static evolution in brittle fracture. *Communications on Pure and Applied Mathematics*, 56(10):1465–1500, 2003.

[66] T. Gerasimov and L. D. Lorenzis. A line search assisted monolithic approach for phase-field computing of brittle fracture. *Computer Methods in Applied Mechanics and Engineering*, 312:276 – 303, 2016. Phase Field Approaches to Fracture.

[67] T. Gerasimov and L. D. Lorenzis. On penalization in variational phase-field models of brittle fracture. *Computer Methods in Applied Mechanics and Engineering*, 2019.

[68] T. Gerasimov, N. Noii, O. Allix, and L. De Lorenzis. A non-intrusive global/local approach applied to phase-field modeling of brittle fracture. *Advanced Modeling and Simulation in Engineering Sciences*, 5(1):14, May 2018.

[69] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *Int. J. Numer. Methods Engrg.*, 79(11):1309–1331, 2009.

[70] B. Giovanardi, A. Scotti, and L. Formaggia. A hybrid XFEM-phase field (xfield) method for crack propagation in brittle elastic materials. *Computer Methods in Applied Mechanics and Engineering*, 320:396 – 420, 2017.

[71] H. Goldstein, C. P. P. Jr., and J. L. S. Sr. *Klassische Mechanik.* Wiley-VCH, 3. auflage edition, 2006.

[72] C. Goll, T. Wick, and W. Wollner. DOpElib: Differential equations and optimization environment; A goal oriented software library for solving pdes and optimization problems with pdes. *Archive of Numerical Software*, 5(2):1–14, 2017.

[73] A. Griffith. The phenomena of flow and rupture in solids: Phil. *Trans. Roy. Soc. Lond. Ser. A*, 221:163–198, 1920.

[74] C. Großmann, H.-G. Roos, and M. Stynes. *Numerical Treatment of Partial Differential Equations.* Springer, 2007.

[75] K. Gustafsson, M. Lundh, and G. Soederlind. A PI stepsize control for the numerical solution of ordinary differential equations. *BIT*, 28(2):270–287, 1988.

[76] Z. Han. *Primal-dual active-set methods for convex quadratic optimization with applications.* PhD thesis, Lehigh University, 2015.

[77] M. Hanke-Bourgeois. *Grundlagen der numerischen Mathematik und des Wissenschaftlichen Rechnens.* Vieweg-Teubner Verlag, 2009.

[78] A. C. Hansen-Dörr, R. de Borst, P. Hennig, and M. Kästner. Phase-field modelling of interface failure in brittle materials. *Computer Methods in Applied Mechanics and Engineering*, 346:25 – 42, 2019.

[79] F. Hausdorff. Dimension und äußeres maß. *Mathematische Annalen*, 79(1):157–179, Mar 1918.

[80] Y. Heider and B. Markert. A phase-field modeling approach of hydraulic fracture in saturated porous media. *Mechanics Research Communications*, pages –, 2016.

[81] Y. Heider, S. Reiche, P. Siebert, and B. Markert. Modeling of hydraulic fracturing using a porous-media phase-field approach with reference to experimental data. *Engineering Fracture Mechanics*, 202:116 – 134, 2018.

[82] T. Heister, M. F. Wheeler, and T. Wick. A primal-dual active set method and predictor-corrector mesh adaptivity for computing fracture propagation using a phase-field approach. *Comp. Meth. Appl. Mech. Engrg.*, 290(0):466 – 495, 2015.

[83] T. Heister and T. Wick. Parallel solution, adaptivity, computational convergence, and open-source code of 2d and 3d pressurized phase-field fracture problems. *PAMM*, 18(1):e201800353, 2018.

[84] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski, E. T. Phipps, A. G. Salinger, H. K. Thornquist, R. S. Tuminaro, J. M. Willenbring, A. Williams, and K. S. Stanley. An overview of the trilinos project. *ACM Trans. Math. Softw.*, 31(3):397–423, 2005.

[85] C. Hesch, S. Schuss, M. Dittmann, M. Franke, and K. Weinberg. Isogeometric analysis and hierarchical refinement for higher-order phase-field models. *Computer Methods in Applied Mechanics and Engineering*, 303:185 – 207, 2016.

[86] M. Hintermüller, K. Ito, and K. Kunisch. The primal-dual active set strategy as a semismooth newton method. *SIAM Journal on Optimization*, 13(3):865–888, 2002.

[87] M. Hinze, R. Pineau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE constraints.* Mathematical Modelling: Theory and Applications, Vol. 23. Springer, 2009.

[88] M. Hofacker and C. Miehe. Continuum phase field modeling of dynamic fracture: variational principles and staggered FE implementation. *Int. J. Fract.*, 178:113–129, 2012.

[89] G. Holzapfel. *Nonlinear Solid Mechanics: A continuum approach for engineering.* John Wiley and Sons, LTD, 2000.

[90] J. Hron, A. Ouazzi, and S. Turek. A computational comparison of two FEM solvers for nonlinear incompressible flow. In *Lecture notes in Computational Science and Engineering, Vol. 35*, pages 87–109. Springer, 2003.

[91] K. Ito and K. Kunisch. Augmented lagrangian methods for nonsmooth, convex optimization in Hilbert spaces. *Nonlinear Analysis*, 41:591–616, 2000.

[92] L. Jansen and J. van Stappen. Newton's method for non-linear coupled systems inspired by load flow analysis. Project in numerical modeling, MAP 502, Ecole Polytechnique, March 2017.

[93] D. Jodlbauer, U. Langer, and T. Wick. Matrix-free multigrid solvers for phase-field fracture problems. arXiv preprint arXiv:1902.08112, Feb 2019.

[94] V. John and J. Rang. Adaptive time step control for the incompressible navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 199(9-12):514 – 524, 2010.

[95] N. Kikuchi and J. T. Oden. *Contact problems in elasticity: a study of variational inequalities and finite element methods*, volume 8. siam, 1988.

[96] D. Kinderlehrer and G. Stampacchia. *An introduction to variational inequalities and their applications*, volume 31. Siam, 1980.

[97] K. Koenigsberger. *Analysis 1.* Springer Lehrbuch. Springer, Berlin – Heidelberg – New York, 6. auflage edition, 2004.

[98] K. Koenigsberger. *Analysis 2*. Springer Lehrbuch. Springer, Berlin – Heidelberg – New York, 5. auflage edition, 2004.

[99] R. Kornhuber. A posteriori error estimates for elliptic variational inequalities. *Computers Math. Applic.*, 31:49–90, 1996.

[100] A. Krause and R. Krause. Recursive multilevel trust region method with application to fully monolithic phase-field models of brittle fracture. arXiv preprint arXiv:1903.00379, March 2019.

[101] C. Kuhn and R. Müller. A continuum phase field model for fracture. *Engineering Fracture Mechanics*, 77(18):3625 – 3634, 2010. Computational Mechanics in Fracture and Damage: A Special Issue in Honor of Prof. Gross.

[102] C. Kuhn and R. Müller. A new finite element technique for a phase field model of brittle fracture. *Journal of Theoretical and Applied Mechanics*, 49(4):1115–1133, 2011.

[103] C. Kuhn, T. Noll, and R. Müller. On phase field modeling of ductile fracture. *GAMM-Mitteilungen*, 39(1):35–54, 2016.

[104] C. J. Larsen. *IUTAM Symposium on Variational Concepts with Applications to the Mechanics of Materials: Proceedings of the IUTAM Symposium on Variational Concepts with Applications to the Mechanics of Materials, Bochum, Germany, September 22-26, 2008*, chapter Models for Dynamic Fracture Based on Griffith's Criterion, pages 131–140. Springer Netherlands, Dordrecht, 2010.

[105] C. J. Larsen, C. Ortner, and E. Süli. Existence of solutions to a regularized model of dynamics fracture. *Methods in Applied Sciences*, 20:1021–1048, 2010.

[106] S. Lee, M. F. Wheeler, and T. Wick. Pressure and fluid-driven fracture propagation in porous media using an adaptive finite element phase field model. *Computer Methods in Applied Mechanics and Engineering*, 305:111 – 132, 2016.

[107] S. Lee, M. F. Wheeler, and T. Wick. Iterative coupling of flow, geomechanics and adaptive phase-field fracture including level-set crack width approaches. *Journal of Computational and Applied Mathematics*, 314:40 – 60, 2017.

[108] J.-L. Lions. *Optimal Control of Systems Governed by Partial Differential Equations*, volume 170 of *Grundlehren Math. Wiss.* Springer, Berlin, 1971.

[109] C. M. Maes. *A regularized active-set method for sparse convex quadratic programming.* PhD thesis, Stanford University, 2010.

[110] G. Mallik, M. Vohralík, and S. Yousef. Goal-oriented a posteriori error estimation for conforming and nonconforming approximations with inexact solvers. working paper or preprint, Dec. 2018.

[111] S. Mandal, A. Ouazzi, and S. Turek. Modified Newton solver for yield stress fluids. In *Numerical Mathematics and Advanced Applications, ENUMATH 2015.* Springer, 2016.

[112] K. Mang, T. Wick, and W. Wollner. A phase-field model for fractures in incompressible solids. arXiv:1901.05378, accepted for publication in Computational Mechanics, July 2019.

[113] C. Mehlmann and T. Richter. A modified global newton solver for viscous-plastic sea ice models. *Ocean Modelling*, 116:96 – 107, 2017.

[114] D. Meidner, R. Rannacher, and J. Vihharev. Goal-oriented error control of the iterative solution of finite element equations. *Journal of Numerical Mathematics*, 17:143–172, 2009.

[115] A. Mesgarnejad, B. Bourdin, and M. Khonsari. Validation simulations for the variational approach to fracture. *Computer Methods in Applied Mechanics and Engineering*, 290:420 – 437, 2015.

[116] C. Miehe, M. Hofacker, L.-M. Schaenzel, and F. Aldakheel. Phase field modeling of fracture in multi-physics problems. part ii. coupled brittle-to-ductile failure criteria and crack propagation in thermo-elastic-plastic solids. *Computer Methods in Applied Mechanics and Engineering*, 294:486 – 522, 2015.

[117] C. Miehe, M. Hofacker, and F. Welschinger. A phase field model for rate-independent crack propagation: Robust algorithmic implementation based on operator splits. *Comput. Meth. Appl. Mech. Engrg.*, 199:2765–2778, 2010.

[118] C. Miehe and S. Mauthe. Phase field modeling of fracture in multi-physics problems. part iii. crack driving forces in hydro-poro-elasticity and hydraulic fracturing of fluid-saturated porous media. *Computer Methods in Applied Mechanics and Engineering*, pages –, 2015.

[119] C. Miehe, S. Mauthe, and S. Teichtmeister. Minimization principles for the coupled problem of darcy-biot-type fluid transport in porous media linked to phase field modeling of fracture. *Journal of the Mechanics and Physics of Solids*, 82:186 – 217, 2015.

[120] C. Miehe, L.-M. Schaenzel, and H. Ulmer. Phase field modeling of fracture in multi-physics problems. part i. balance of crack surface and failure criteria for brittle crack propagation in thermo-elastic solids. *Computer Methods in Applied Mechanics and Engineering*, 294:449 – 485, 2015.

[121] C. Miehe, F. Welschinger, and M. Hofacker. Thermodynamically consistent phase-field models of fracture: variational principles and multi-field fe implementations. *Int. J. Numer. Methods Engrg.*, 83:1273–1311, 2010.

[122] A. Mikelić, M. F. Wheeler, and T. Wick. A phase-field approach to the fluid filled fracture surrounded by a poroelastic medium. ICES Report 13-15, Jun 2013.

[123] A. Mikelić, M. F. Wheeler, and T. Wick. A phase-field method for propagating fluid-filled fractures coupled to a surrounding porous medium. *SIAM Multiscale Model. Simul.*, 13(1):367–398, 2015.

[124] A. Mikelić, M. F. Wheeler, and T. Wick. Phase-field modeling of a fluid-driven fracture in a poroelastic medium. *Computational Geosciences*, 19(6):1171–1195, 2015.

[125] A. Mikelić, M. F. Wheeler, and T. Wick. A quasi-static phase-field approach to pressurized fractures. *Nonlinearity*, 28(5):1371–1399, 2015.

[126] A. Mikelić, M. F. Wheeler, and T. Wick. Phase-field modeling through iterative splitting of hydraulic fractures in a poroelastic medium. *GEM - International Journal on Geomathematics*, 10(1), Jan 2019.

[127] P. Mital, T. Wick, M. Wheeler, and G. Pencheva. Discontinuous and enriched galerkin methods for phase-field fracture propagation in elasticity. In *Numerical Mathematics and Advanced Applications, ENUMATH 2015*. Springer, 2016.

[128] L. Modica and S. Mortola. Il limite nella γ-convergenza di una famiglia di funzionali ellittici. *Boll. Un. Mat. Ital. A (5)*, 14(3):526–529, 1977.

[129] M. Negri. The anisotropy introduced by the mesh in the finite element approximation of the mumford-shah functional. *Numerical Functional Analysis and Optimization*, 20(9-10):957–982, 1999.

[130] I. Neitzel, T. Wick, and W. Wollner. An optimal control problem governed by a regularized phase-field fracture propagation model. *SIAM Journal on Control and Optimization*, 55(4):2271–2288, 2017.

[131] I. Neitzel, T. Wick, and W. Wollner. An optimal control problem governed by a regularized phase-field fracture propagation model. part ii: The regularization limit. *SIAM Journal on Control and Optimization*, 57(3):1672–1690, 2019.

[132] T. Nguyen, J. Yvonnet, Q.-Z. Zhu, M. Bornert, and C. Chateau. A phase-field method for computational modeling of interfacial damage interacting with crack propagation in realistic microstructures obtained by microtomography. *Computer Methods in Applied Mechanics and Engineering*, 312:567 – 595, 2016.

[133] J. Nocedal and S. J. Wright. *Numerical optimization.* Springer Ser. Oper. Res. Financial Engrg., 2006.

[134] N. Noii, F. Aldakheel, T. Wick, and P. Wriggers. An adaptive global-local approach for phase-field modeling of anisotropic brittle fracture. arXiv preprint arXiv:1905.07519, May 2019.

[135] N. Noii and T. Wick. A phase-field description for pressurized and non-isothermal propagating fractures. *Computer Methods in Applied Mechanics and Engineering*, 351:860 – 890, 2019.

[136] J. Oden. Adaptive multiscale predictive modelling. *Acta Numerica, Cambridge University Press*, pages 353–450, 2018.

[137] J. Oden and S. Prudhomme. Estimation of modeling error in computational mechanics. *Journal of Computational Physics*, 182(2):496 – 515, 2002.

[138] K. Pham, H. Amor, J.-J. Marigo, and C. Maurini. Gradient Damage Models and Their Use to Approximate Brittle Fracture. *Int. J. of Damage Mech.*, pages 1–36, May 2011.

[139] G. Pillo, G. Liuzzi, S. Lucidi, and L. Palagi. A truncated newton method in an augmented lagrangian framework for nonlinear programming. *Computational Optimization and Applications*, 45(2):311–352, 2008.

[140] L. Pontryagin, V. Boltyanskii, and R. Gamkrelidze. *The mathematical theory of optimal processes.* Pergamon Press, Oxford, 1964.

[141] R. Rannacher. Analysis 2. Vorlesungsskriptum, 2010.

[142] R. Rannacher. *Numerik gewöhnlicher Differentialgleichungen.* Heidelberg University Publishing, 2017.

[143] R. Rannacher. *Numerik partieller Differentialgleichungen.* Heidelberg University Publishing, 2017.

[144] R. Rannacher. Special topics in numerics: II. Adaptivity in the FEM. Vorlesungsskriptum, 2017.

[145] R. Rannacher and F.-T. Suttmeier. A posteriori error control in finite element methods via duality techniques: Application to perfect plasticity. *Computational Mechanics*, 21(2):123–133, 1998.

[146] R. Rannacher and F.-T. Suttmeier. A posteriori error estimation and mesh adaptation for finite element models in elasto-plasticity. *Computer Methods in Applied Mechanics and Engineering*, 176(1-4):333 – 361, 1999.

[147] R. Rannacher and F.-T. Suttmeier. Error estimation and adaptive mesh design for FE models in elasto-plasticity. In E. Stein, editor, *Error-Controlled Adaptive FEMs in Solid Mechanics.* John Wiley, 2000.

[148] R. Rannacher and J. Vihharev. Adaptive finite element analysis of nonlinear problems: balancing of discretization and iteration errors. *Journal of Numerical Mathematics*, 21(1):23–61, 2013.

[149] J. Rice. *Mathematical analysis in the mechanics of fracture*, pages 191–311. Academic Press New York, chapter 3 of fracture: an advanced treatise edition, 1968.

[150] T. Richter and T. Wick. Variational localizations of the dual weighted residual estimator. *Journal of Computational and Applied Mathematics*, 279(0):192 – 208, 2015.

[151] J. M. Sargado, E. Keilegavlen, I. Berre, and J. M. Nordbotten. High-accuracy phase-field models for brittle fracture based on a new family of degradation functions. *Journal of the Mechanics and Physics of Solids*, 111:458 – 489, 2018.

[152] A. Schlüter, A. Willenbücher, C. Kuhn, and R. Müller. Phase field approximation of dynamic brittle fracture. *Comput. Mech.*, 54:1141–1161, 2014.

[153] M. Seiler, T. Linse, P. Hantschke, and M. Kästner. An efficient phase-field model for fatigue fracture in ductile materials. arXiv preprint arXiv:1903.06465, Mar 2019.

[154] I. N. Sneddon and M. Lowengrub. *Crack problems in the classical theory of elasticity.* SIAM series in Applied Mathematics. John Wiley and Sons, Philadelphia, 1969.

[155] F. Suttmeier. *Numerical solution of Variational Inequalities by Adaptive Finite Elements.* Vieweg+Teubner, 2008.

[156] A. Z. Szeri. *Fluid Film Lubrication.* Cambridge University Press, 2 edition, 2010.

[157] S. Teichtmeister, D. Kienle, F. Aldakheel, and M.-A. Keip. Phase field modeling of fracture in anisotropic brittle solids. *International Journal of Non-Linear Mechanics*, 97:1–21, 2017.

[158] R. Trémolières, J. Lions, and R. Glowinski. *Analyse numérique des inéquations variationnelles.* Dunod, 1976.

[159] R. Trémolières, J. Lions, and R. Glowinski. *Numerical Analysis of Variational Inequalities.* Studies in Mathematics and its Applications. Elsevier Science, 2011.

[160] F. Tröltzsch. *Optimale Steuerung partieller Differentialgleichungen - Theorie, Verfahren und Anwendungen.* Vieweg und Teubner, Wiesbaden, 2nd edition, 2009.

[161] S. Turek. *Efficient solvers for incompressible flow problems.* Springer-Verlag, 1999.

[162] H. Ulmer, M. Hofacker, and C. Miehe. Phase field modeling of brittle and ductile fracture. *PAMM*, 13:533–536, 2013.

[163] C. van Duijn, A. Mikelić, and T. Wick. A monolithic phase-field model of a fluid-driven fracture in a nonlinear poroelastic medium. *Mathematics and Mechanics of Solids*, 2018.

[164] N. van Goethem and A. Novotny. Crack nucleation sensitivity analysis. *Math. Methods Appl. Sci.*, 33(16), 2010.

[165] A. Veeser. Efficient and reliable a posteriori error estimators for elliptic obstacle problems. *SIAM Journal on Numerical Analysis*, 39(1):146–167, 2001.

[166] C. V. Verhoosel and R. de Borst. A phase-field model for cohesive fracture. *International Journal for Numerical Methods in Engineering*, 96(1):43–62, 2013.

[167] J. Vignollet, S. May, R. Borst, and C. V. Verhoosel. Phase-field models for brittle and cohesive fracture. *Meccanica*, 49(11):2587–2601, 2014.

[168] D. Werner. *Funktionalanalysis.* Springer, 2004.

[169] M. Wheeler, T. Wick, and W. Wollner. An augmented-Lagangrian method for the phase-field approach for pressurized fractures. *Comp. Meth. Appl. Mech. Engrg.*, 271:69–85, 2014.

[170] D. Wick, T. Wick, H. R. Hellmig, and H.-J. Christ. Numerical simulations of crack propagation in screws with phase-field modeling. *Computational Materials Science*, 109:367–379, 2015.

[171] T. Wick. Solving monolithic fluid-structure interaction problems in arbitrary Lagrangian Eulerian coordinates with the deal.II library. *Archive of Numerical Software*, 1:1–19, 2013.

[172] T. Wick. Coupling fluid-structure interaction with phase-field fracture. *Journal of Computational Physics*, 327:67 – 96, 2016.

[173] T. Wick. Goal functional evaluations for phase-field fracture using PU-based DWR mesh adaptivity. *Computational Mechanics*, 57(6):1017–1035, 2016.

[174] T. Wick. Coupling fluid-structure interaction with phase-field fracture: algorithmic details. In S. Frei, B. Holm, T. Richter, T. Wick, and H. Yang, editors, *Fluid-Structure Interaction: Modeling, Adaptive Discretization and Solvers*, Radon Series on Computational and Applied Mathematics. Walter de Gruyter, Berlin, 2017.

[175] T. Wick. An error-oriented Newton/inexact augmented Lagrangian approach for fully monolithic phase-field fracture propagation. *SIAM Journal on Scientific Computing*, 39(4):B589–B617, 2017.

[176] T. Wick. Introduction to numerical modeling. MAP 502: Lecture notes at Ecole Polytechnique, 2017.

[177] T. Wick. Modified Newton methods for solving fully monolithic phase-field quasi-static brittle fracture propagation. *Computer Methods in Applied Mechanics and Engineering*, 325:577 – 611, 2017.

[178] T. Wick. Numerical methods for partial differential equations. Institute for Applied Mathematics, Leibniz Universitaet Hannover, Germany, 2018.

[179] T. Wick, S. Lee, and M. Wheeler. 3D phase-field for pressurized fracture propagation in heterogeneous media. ECCOMAS and IACM Coupled Problems Proc., May 2015 at San Servolo, Venice, Italy, 2015.

[180] T. Wick, G. Singh, and M. Wheeler. Fluid-filled fracture propagation using a phase-field approach and coupling to a reservoir simulator. *SPE Journal*, 21(03):981–999, 2016.

[181] J. Wloka. *Partial differential equations.* Cambridge University Press, 1987.

[182] F. Zhang, W. Huang, X. Li, and S. Zhang. Moving mesh finite element simulation for phase-field modeling of brittle fracture and convergence of newton's iteration. *Journal of Computational Physics*, 356:127 – 149, 2018.

[183] S. Zhou, X. Zhuang, and T. Rabczuk. Phase-field modeling of fluid-driven dynamic cracking in porous media. *Computer Methods in Applied Mechanics and Engineering*, 350:169 – 198, 2019.

[184] S. Zhou, X. Zhuang, and T. Rabczuk. Phase-field modeling of fluid-driven dynamic cracking in porous media. *Computer Methods in Applied Mechanics and Engineering*, 350:169 – 198, 2019.

[185] V. Ziaei-Rad and Y. Shen. Massive parallelization of the phase field formulation for crack propagation with time adaptivity. *Computer Methods in Applied Mechanics and Engineering*, 312:224 – 253, 2016. Phase Field Approaches to Fracture.

Index

A posteriori error estimation, 123
A priori error estimation, 123
Active set, 31, 34, 107, 108
Adaptive FEM, 124
Adaptive finite elements, 124
Adaptivity, 64
  Spatial mesh, 148
  Time step, 151
Adjoint methods, 129
Adjoint problem, 131
Adjoint sensitivity, 130, 131
Adjoint variable, 125, 129
Advantages of phase-field fracture, 15
AFEM, 124
  Basic algorithm, 124
All variables, symbols, etc., 27
Ambrosio-Tortorelli, 29, 41, 168
Aperture, 174
Augmented Lagrangian, 114

Backtracking line-search, 101
backtracking line-search, 109
Bochner integral, 72
Bonnet-David
  Manufactured solution crack, 155
Brachistochrone, 127
Brittle fracture, 35, 39
Bulk energy, 37

Chain rule, 22
Change of variables, 22, 23
Classifications of PDEs, 51
COD, 175
Compact support, 24
Computational convergence, 183
Computational cost, 176
Computer program, 179
Constraint, 106
Contact zone, 34
Control parameter, 99, 100
Convergence order
  Computationally obtained, 185
Convex set, 33
Coupled PDE, 55
Coupling
  Interface, 57
  Volume, 56
Crack irreversibility, 43
Crack opening displacement, 174
Crack opening displacement, 175
Crack width, 165
Critical energy release rate, 35
Cross product, 20

deal.II, 111, 179
Decoupled vs. coupled, 55
Defect, 83
Deformation gradient, 23
Differential equation
  Classification, 53
  Linear vs. nonlinear, 53
Differentiation in Banach spaces, 80
Diffusive interface approaches, 17
Directional derivative, 97
Discretization error, 184
Divergence, 20
Divergence theorem, 23
DOpElib, 89, 179
Dual-weighted residual method, 125
Duality arguments, 125
DWR, 125, 155
  Obstacle problem, 143

Efficient error estimator, 124
Elastic membrane, 32
Element residual, 136
Energy
  Bulk, 37
  Surface, 36
Energy release rate, 36
Energy splitting, 44
Entropy condition, 43
Entropy condition, 46
Error estimation
  Goal-oriented, 125
Error localization, 135
Euler-Lagrange equations, 127
Extrapolation, 80, 115
extrapolation, 79, 99, 117

Face residuals, 136
Finite elements
  Adaptivity, 124
Fitted methods, 16
Fixed-point iteration, 77
Fracture irreversibility, 43
Free boundary, 34
Free boundary problem, 31
Fundamental lemma of calculus of variations, 24

GMRES, 176
Goal functionals, 174
Goal-oriented error estimation, 155

Goal-oriented mesh adaptivity, 155
Gradient, 20
Griffith, 35, 39, 40

Hanging nodes, 140
Hausdorff measure, 36

Ill-conditioning, 35, 114
Implementation, 89
    Pressurized fracture, 171
    PU-DWR phase-field, 155
    Single edge notched shear test, 117
inactive set, 107
Integration by parts, 23
Interface coupling, 57
Irreversibility, 29, 38, 39, 43, 46, 47
irreversibility, 117
Iteration error, 187

Jacobian, 20

Lagrange multiplier, 34, 125, 128
Lagrange multipliers, 128
Lagrangian, 125
Lamé parameters, 37
Length-scale parameter, 31
Line search algorithm, 94
Linear equation system
    Elasticity, stationary, 85
    Elasticity, nonlinear, non-stationary, 86
Linear operator, 53
Linear PDE, 53
Linear solver performance, 176
Linear vs. nonlinear, 53
Linear-in-time extrapolation, 79
Linearity
    Definition, 53
Linearization by Newton, 82
Loading increment parameter, 19
Lower-dimensional interface approaches, 17

Maximum norm, 21
Mesh adaptivity, 148
Mesh refinement studies, 93, 120, 158
Modified Newton
    Control parameter, 99
Modified Newton method, 105
Monolithic system, 110

Nabla-operator, 20
Newton
    Implementation, 95
Newton method, 106, 114
Newton's method, 82
    Modified version, 99
    Monotonicity-based criterion, 94
    Primal dual active set, 106
    Defect-correction, 83
    overview, 84
Non-active set, 31
Nonactive set, 109
Nonlinear PDE, 53
Norm, 21
Normed space, 21
Numerical analysis, 65

Obstacle problem, 29
Obstacle problem, 30, 32, 43, 89, 143
    Strong formulation, 34
ODE, 51
Ordinary differential equation, 51

Parallel computing, 171
Partial differential equation, 51
Partial integration, 23
PDE, 51
    Classifications, 51
    Coupled, 55
    Definition, 52
PDE system, 52
Penalization, 35, 62
Phase-field fracture, 3
    Lagrange multiplier formulation, 34
    Numerical analysis, 65
    Pressurized, 166
    Strong, simplified, formulation, 29
    Weak formulation, 43
    Weak formulation strain history, 48
    Weak formulation stress splitting, 46
Picard iteration, 77
Poisson's ratio, 37
Preconditioner, 111
Predictor-corrector mesh adaptivity, 148
Predictor-corrector scheme, 149
predictor-corrector scheme, 148
Pressurized fractures, 166
Primal dual active set, 63, 106
Primal variable, 125
Primal-dual active set, 63
Programming, 179
PU-DWR, 155

Quasi-linear, 31
Questions to these lecture notes, 189
Quiz, 189

Regularization parameter, 31, 42
Relationship h, ε, κ, 70
Relationship mesh size and DoFs, 183
Relationship Young, Poisson's ratio, Lamé, 37
Reliable error estimator, 124

Rotation, 20

Saddle point, 126
Saint-Venaint Kirchhoff model, 37
Schur complement, 111
Semi-linear, 56
Semi-linear DGL, 56
Semi-linear form, 43, 56
Semi-norm, 21
Semi-smooth Newton, 63
Semi-smoothed Newton method, 107
Sensitivity
    Adjoint, 130, 131
Shortcomings of phase-field fracture, 15
Simple penalization, 143
Simplified phase-field fracture, 29
Simulation
    Pressurized fracture, 171
    PU-DWR phase-field, 155
    Single edge notched shear test, 117
Single edge notched shear test, 117, 148
Smooth functions, 24
Sneddon, 171, 172
Software, 179
Spatial discretization, 94
Stability, 35
stopping criteria, 109
Strain history, 46, 47, 61
Strain history field, 115
Stress splitting, 44
Strong formulation
    Phase-field fracture, simplified, 29
    Simplified problem, 29
Strong formulation obstacle problem, 34
Substitution rule, 22
Summary
    All variables, symbols, etc., 27
Surface energy, 36

Tensile stress, 45
Test function, 43
Time step control, 151
Time-dependent, 31
Toughness, 35
Trace, 20
Transformation of integrals, 22
Trial function, 43

Unfitted methods, 17

Variational approach to fracture, 3
Variational formulation
    Obstacle problem, 33
Variational phase-field fracture, 3
Vector space, 21
Volume coupling, 56
Volume ratio, 23

Weak formulation, 43
    strain history, 48
    Stress splitting, 46
Width, 174

Young’s modulus, 37