Optimal Control of anisotropic Allen–Cahn equations

Dissertation zur Erlangung des Doktorgrades
der Naturwissenschaften (Dr. rer. nat.)
der Fakultät für Mathematik
der Universität Regensburg

vorgelegt von

Johannes Meisinger

aus Bayreuth

im Jahr 2022

Promotionsgesuch eingereicht am: 07.12.2021

Abstract

This thesis is concerned with the solution of an optimal control problem governed by an anisotropic Allen-Cahn equation as a model for, e.g., crystal growth.
The first part treats the analytical existence theory and first order optimality conditions of the in time continuous and of the time discretized versions. The state equation is discretized implicitly in time with piecewise constant functions. To this end, we consider a more general quasilinear parabolic equation, where the quasilinear term is strongly monotone and obeys a certain growth condition while the lower order term is potentially non-monotone. The existence of the control-to-state operator and its Lipschitz continuity is shown for the time discretized as well as for the time continuous problem. Then we present for both the existence of global minimizers as well as the convergence of a subsequence of time discrete optimal controls to a global minimizer of the time continuous problem. The results hold in arbitrary space dimensions. Under some further restrictions we are able to show Fréchet differentiability of the in time discretized problem and use this to rigorously set up the first order conditions. For this the anisotropies are required to be smooth enough, which in this thesis is achieved by a suitable regularization. Therefore, the convergence behavior of the optimal controls are studied for a sequence of (smooth) approximations of the former quasilinear term. In addition the simultaneous limit in the approximation and the time step size is considered. For a class covering a large variety of anisotropies we introduce a certain regularization and show the previously formulated requirements. Finally, we will show that the results cannot be straightforwardly transferred to a semi-implicit discretization scheme.
In the second part a trust region Newton method is presented, that eventually is used to numerically solve the optimal control problem. Different ways of preconditioning the involved Steihaug-CG solver are discussed and the limits of existing approaches in the present case are worked out. Then, several aspects of the implementation are examined, like the solver for the appearing partial differential equations, parallelization and the utility of adaptive meshes in the context of the control problem.
In the final part, various numerical results based on the previously mentioned choice of anisotropies are presented. These include convergence with respect to the regularization parameter, numerical evidence for mesh independent behavior and a thorough discussion of the simulation in several relevant settings. We concentrate on two choices for the anisotropies and in addition include the isotropic case for comparison. Among others, crystal formation and topology changes are addressed and we see that the algorithm is able to handle these. Furthermore, the behavior of various quantities over the course of the algorithm is investigated. Here we observe that the number of Steihaug steps, and therefore the execution time per trust region step, growths considerably towards the end of the algorithm. Finally, we look at the impact of some implementational aspects with respect to execution speed. We observe that the implicit and semi-implicit approaches perform comparably fast if the implementation is suitably optimized. We however conclude that the implicit approach is preferable since it is less sensitive with respect to the regularization and is supported by more theoretical results.

Zusammenfassung

In dieser Arbeit wird die Lösung eines Optimalsteuerungsproblems behandelt, welches eine anisotrope Allen-Cahn Gleichung als Modell für beispielsweise Kristallwachstum enthält.
Der erste Teil befasst sich mit der analytischen Existenztheorie und den Optimalitätsbedingungen erster Ordnung der zeitkontinuierlichen und zeitdiskreten Versionen. Die Zustandsgleichung wird implizit in der Zeit mit stückweise konstanten Funktionen diskretisiert. Dazu betrachten wir eine allgemeinere quasilineare parabolische Gleichung, bei welcher der quasilineare Term stark monoton ist und einer bestimmten Wachstumsbedingung gehorcht, während der Term niedrigerer Ordnung nicht unbedingt monoton ist. Die Existenz des Steuerungs-Zustands-Operators sowie dessen Lipschitz-Stetigkeit wird sowohl für das zeitdiskretisierte als auch für das zeitkontinuierliche Problem bewiesen. Anschließend zeigen wir sowohl die Existenz von globalen Minimierern als auch die Konvergenz einer Teilfolge von zeitdiskreten optimalen Steuerungen gegen einen globalen Minimierer des zeitkontinuierlichen Problems. Die Ergebnisse gelten für beliebige Raumdimensionen. Unter einigen zusätzlichen Einschränkungen sind wir in der Lage die Fréchet-Differenzierbarkeit des zeitdiskretisierten Problems zu zeigen und nutzen dies um die Bedingungen erster Ordnung rigoros aufzustellen. Hierfür müssen die Anisotropien glatt genug sein, was in dieser Arbeit durch eine geeignete Regularisierung sichergestellt wird. Daher wird das Konvergenzverhalten der optimalen Steuerungen für eine Folge von (glatten) Approximationen des ursprünglichen quasilinearen Terms untersucht. Zusätzlich wird der gleichzeitige Limes der Approximation und der Zeitschrittweite behandelt. Für eine Klasse, welche eine große Vielfalt an Anisotropien abdeckt, führen wir eine spezifische Regularisierung ein und zeigen die zuvor formulierten Voraussetzungen. Schließlich werden wir zeigen, dass die Ergebnisse nicht ohne Weiteres auf ein semi-implizites Diskretisierungsschema übertragen werden können.
Im zweiten Teil wird ein Trust-Region-Newton-Verfahren vorgestellt, das schließlich verwendet wird die numerische Lösung des optimalen Steuerungsproblems zu berechnen. Auf verschiedene Möglichkeiten der Vorkonditionierung des beteiligten Steihaug-CG-Lösers wird eingegangen und die Grenzen vorhandener Ansätze im vorliegenden Fall werden herausgearbeitet. Dann werden mehrere Aspekte der Implementierung beleuchtet, wie der Löser für die auftretenden partiellen Differentialgleichungen, Parallelisierung und der Nutzen von adaptiven Gittern im Zusammenhang mit dem Steuerungsproblem.
Im letzten Teil werden verschiedene numerische Ergebnisse präsentiert, die auf der zuvor erwähnten Wahl der Anisotropien basieren. Dazu gehören Konvergenz in Bezug auf den Regularisierungsparameter, numerische Evidenz für netzunabhängiges Verhalten und eine ausführliche Diskussion der Simulation in verschiedenen relevanten Situationen. Wir konzentrieren uns auf zwei Möglichkeiten für die Anisotropien und ergänzen zum Vergleich auch den isotropen Fall. Unter anderem werden die Kristallbildung und Topologieänderungen behandelt und wir werden sehen, dass der Algorithmus in der Lage ist diese zu bewältigen. Weiterhin wird das Verhalten verschiedener Größen im Verlauf des Algorithmus untersucht. Hier beobachten wir, dass die Anzahl der Steihaug-Schritte und damit die Ausführungszeit pro Trust-Region-Schritt, zum Ende des Algorithmus hin erheblich ansteigt. Schließlich betrachten wir die Auswirkungen einiger Implementierungsaspekte im Hinblick auf die Ausführungsgeschwindigkeit. Wir stellen fest, dass die impliziten und semi-impliziten Ansätze vergleichbar schnell sind, sofern die Implementierung entsprechend optimiert ist. Wir kommen jedoch zu dem Schluss, dass der implizite Ansatz zu bevorzugen ist, da er weniger empfindlich bezüglich der Regularisierung ist und durch mehr theoretische Ergebnisse gestützt wird.

Contents

1 Introduction

In many scientific areas the optimal control of an interface evolution towards a prescribed anisotropic shape is desired. For example, in chemistry or materials science one wishes to steer the solidification process of crystals and in medicine this supports the production of new pharmaceuticals. We refer to [27, 50, 59, 102] and references therein for examples. Concerning the time evolution of shapes, phase field models—that are often of Allen-Cahn or Cahn-Hilliard type—have shown great promise in many application areas. After the introduction of the Allen-Cahn equation in the late 70s [2], several attempts have been made to also formulate an anistropic version starting from the 90s [97, 125]. The use of the latter better meets the requirements of the above tasks as it allows for a more realistic modeling. The involved anisotropies are typically nonsmooth, which has to be taken care of when setting up optimality conditions. To the best of the author's knowledge there does not yet exist any mathematical treatment on the optimal control of anisotropic phase field models so far. Also the analysis of optimal control of quasilinear partial differential equations is still in its infancy.

For the phase field ansatz the interface is modeled by a diffuse interface layer. Therefore an order parameter y—the so called phase field—is introduced which reflects the pure phases with the values ± 1, e.g., the liquid phase for $y \approx 1$ and the solid phase when $y \approx -1$. The diffuse interface is then given by values between -1 and 1. The gradient flow of the Ginzburg-Landau energy of the form

$$\mathcal{E}(y) := \int_\Omega \varepsilon A(\nabla y) + \varepsilon^{-1}\psi(y)\,\mathrm{d}x \tag{1.1}$$

then determines the time evolution of the shape, in other words the state equation for the control problem. Here the first term represents the surface energy, where $A : \mathbb{R}^d \to \mathbb{R}$ is an (an-)isotropy function that typically is absolutely homogeneous[1] of degree 2. Hence, high values of $|\nabla y|$ are penalized, leading to moderate interface transitions. Also the amount of interfacial regions is kept small by this term. The effect of the additional anisotropy property of A is that the evolution prefers to build interfaces perpendicular to some directions. The potential ψ can be thought of being symmetric and to have its global minima at $\approx \pm 1$, forcing the constant phases to attain these values. The variable $\varepsilon > 0$ determines the interfacial thickness and can be considered as fixed throughout the whole thesis. Considering in particular the scaled L^2-gradient flow

$$\varepsilon \partial_t y = -\nabla_{L^2(\Omega)}\mathcal{E}(y) \tag{1.2}$$

of (1.1) with a sufficiently smooth potential and anisotropy, we obtain the anisotropic Allen-Cahn

[1]A function f is absolutely homogeneous of degree k if $f(rx) = |r|^k f(x)$ for all $x \in \mathbb{R}^d, r \in \mathbb{R}$.

equation, as will briefly be sketched in what follows. The L^2-gradient is given by the relation

$$(\nabla_{L^2(\Omega)}\mathcal{E}(y), v)_{L^2(\Omega)} = D\mathcal{E}(y)[v] \qquad \forall v \in L^2(\Omega). \tag{1.3}$$

By a formal variation of $\mathcal{E}(y)$ in direction v, we obtain for the directional derivative

$$D\mathcal{E}(y)[v] = \int_\Omega \varepsilon A'(\nabla y)\cdot\nabla v + \varepsilon^{-1}\psi'(y)v\,\mathrm{d}x. \tag{1.4}$$

If we further impose $A'(\nabla y)\nu = 0$ on $\partial\Omega$, we can identify $\nabla_{L^2(\Omega)}\mathcal{E}(y)$ after integrating by parts

$$D\mathcal{E}(y)[v] = \int_\Omega \underbrace{\left(-\varepsilon\nabla\cdot A'(\nabla y) + \varepsilon^{-1}\psi'(y)\right)}_{\nabla_{L^2(\Omega)}\mathcal{E}(y)} v\,\mathrm{d}x. \tag{1.5}$$

Hence, we may write the anisotropic Allen-Cahn equation as

$$\begin{aligned} \varepsilon\partial_t y - \varepsilon\nabla\cdot A'(\nabla y) + \frac{1}{\varepsilon}\psi'(y) &= 0 && \text{in } Q,\\ A'(\nabla y)\nu &= 0 && \text{on } \Sigma,\\ y(0) &= y_0 && \text{in } \Omega, \end{aligned} \tag{1.6}$$

where we defined the space-time cylinder by $Q := [0,T]\times\Omega$ and its lateral boundary by $\Sigma := [0,T]\times\partial\Omega$. For a further introduction to phase field models we refer to [48] and references therein.

In the literature there exist several ways to define the anisotropy A. We will give a small overview here and refer to [19, 45] as well as references therein for additional information. In the pioneering paper [86] the author considers convex anisotropies in order to obtain a well-posed problem. In [101, 113, 126, 136] various approaches are taken to enlarge this also to non-convex anisotropies by adding regularization terms or changing the structure of the energy functional. These approaches lead to higher dimensional terms which make the analysis complicated. The authors of [49] try to tackle the problem by convexifying the anisotropies, but also there mathematical difficulties appear. For the analysis part of this thesis we will keep the choice of A rather general, having in mind assumptions that are satisfied by a more concrete choice that will be treated later in Section 2.6 and that was first introduced by [16, 14]. This choice avoids most of the problems that the approaches from above comprise. Also for ψ we will demand some general requirements and hence the following analysis and numerical ansatz will not only be valid for the standard Allen-Cahn equations but can be applied in general to differential equations arising from a gradient flow of energies of the form (1.1).

The aim is now to determine the distributed control u driving the solution y of the anisotropic Allen-Cahn equation with source term

$$\varepsilon\partial_t y - \varepsilon\nabla\cdot A'(\nabla y) + \tfrac{1}{\varepsilon}\psi'(y) = u, \tag{1.7}$$

such that it minimizes a certain cost functional that penalizes deviations from a prescribed target function y_Ω at a given final time T. Hence the optimal control problem considered throughout this thesis may be described by the following setting:

Let $\Omega \subset \mathbb{R}^d$ be a bounded Lipschitz domain and $y_\Omega \in L^2(\Omega)$ be a given target function. Let a final time $0 < T < \infty$ be given and define Q and Σ as before. For a given initial state $y_0 \in H^1(\Omega)$ the objective is to find a solution to the optimal control problem

$$\min J(y,u) := \frac{1}{2}\|y(T) - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2\varepsilon}\|u\|^2_{L^2(Q)}, \tag{1.8}$$

subject to the quasilinear, possibly nonsmooth parabolic state equation

$$\begin{aligned} \int_Q \varepsilon\partial_t y\eta + \varepsilon A'(\nabla y)^T\nabla\eta + \frac{1}{\varepsilon}\psi'(y)\eta &= \int_Q u\eta \qquad && \forall\eta\in L^2(0,T;H^1(\Omega)),\\ y(0) &= y_0 && \text{in } \Omega, \end{aligned} \tag{1.9}$$

where $u \in L^2(Q) \cong L^2(0,T;L^2(\Omega))$ and $y \in L^2(0,T;H^1(\Omega)) \cap H^1(0,T;L^2(\Omega))$.

We want to stress that the results also hold for a target function $y_Q \in L^2(0,T;L^2(\Omega))$ that is given over the whole time horizon with small modifications (see Remark 2.3.4) and the cost functional (1.8) is studied rather exemplary. Note that J is well defined due to the embedding $L^2(0,T;H^1(\Omega)) \cap H^1(0,T;H^1(\Omega)') \hookrightarrow C([0,T];L^2(\Omega))$. The weighting with $\frac{1}{\varepsilon}$ in (1.8) is due to the fact that by numerical observations the main contributions of the control are located around the interface of the phase field with comparable width. Dividing by ε intends to compensate this behavior, such that the cost functional is invariant with respect to a rescaling thereof. Note again that ε is fixed, but the scale invariance will be helpful when considering the sharp interface limit in possible later studies.
The ultimate goal of this thesis is to study the problem presented above analytically as well as to supply a numerical solver and perform the necessary investigations on showcase simulations. For the optimal control we follow an approach where we first discretize the problem in time, then apply an optimization solver to this time discretized problem, and finally discretize in space for the implementation. For the time discretization we choose an implicit scheme which also allows for the viewpoint on the problem as a control problem of a series of quasilinear elliptic problems. A steepest descent algorithm as well as a trust region Newton solver were implemented as part of this work, where for the numerical investigations we stick to the latter due to its superior convergence properties in direct comparison.
For the numerics the first order conditions are required, that is we need to be able to differentiate the solution operator of the state equation (control-to-state operator). However for many common choices of the anisotropy A this is accompanied with difficulties. Usually the anisotropy is given by

$$A(p) = \tfrac{1}{2}|\gamma(p)|^2, \tag{1.10}$$

where the density function γ is assumed to be an absolutely 1-homogeneous function in $C^2(\mathbb{R}^d \setminus \{0\}) \cap C(\mathbb{R}^d)$ (see, e.g., [14, 45, 53, 64]) providing absolutely 2-homogeneity of A. Consequently A'' is absolutely 0-homogeneous and therefore it cannot be defined to be continuous at the origin unless γ is an energy norm. Hence the control-to-state operator is potentially non-differentiable. Since numerical methods for nonsmooth optimal control problems are still in its infancy, this is problematic for the search of efficient solvers. For a nonsmooth quasilinear elliptic control problem, a semismooth Newton method is applied to a relaxed optimality system in [39]. To the best of the author's knowledge globally convergent methods for parabolic equations without extra regularity requirements do not exist. To circumvent this problem, the present approach is to consider a regularized version of A by modifying the function γ.
In order to solve the problem (1.8)–(1.9) we will face several additional equations that are derived from (1.9) by linearization as well as discretizations of those equations in several stages. The next section tries to provide an overview of the approach pursued in this thesis by presenting the general discretization strategy and providing a formal derivation of just mentioned linearizations in the time continuous setting. With that, they can be consulted retrospectively whenever they appear later in the one or other form and can be placed in the proper context quickly. In the final section of this introductory chapter, we will specify the notation we use and collect the most important results needed throughout this thesis.
The outline of the remaining parts of this thesis is as follows:
In Part 2 we will discuss the analytical results that are related to problem (1.8)–(1.9). In particular, the existence of solutions for the state equation as well as for the control problem is considered in Sections 2.2 and 2.3—both for the time continuous as well as for the time discretized versions. Here we build on the results of [53] that allow for a treatment in arbitrary

space dimension. The hitherto results for the optimal control of the (isotropic) Allen-Cahn equation that are known to the author invoke the constraint $d \leq 3$ [141, 41] or some potential differing from ours [107]. Our results also hold for non-homogeneous anisotropies like the regularization introduced in Section 2.6. In Section 2.4 we will proceed with proving Fréchet differentiability of the solution operator of the state equation and derive the first order necessary conditions needed later for the numerics. Here we will restrict ourselves to the time discrete problem. There are several reasons why we do so. While there exists much literature about existence results for quasilinear parabolic problems and first order conditions of parabolic control problems separately, literature containing a treatment of both topics at once is sparse. A known result treating quasilinear parabolic control problems uses a nicer nonlinearity (in particular a monotone one) and requires spaces where a result is lacking for in our case [38]. Many other papers on quasilinear parabolic equations consider a quasilinear term that is of a different structure, like merely including dependence on y or $|\nabla y|$ [32, 26, 103]. With our time discretization we obtain a sequence of quasilinear elliptic problems. For such equations at least some results exist for distributed control problems with Dirichlet boundary [35, 37] or Neumann boundary controls [33, 34], which we can adapt to our case. We also provide results concerning the convergence of global minimizers with respect to the time discretization as well as with respect to the regularization parameter. Most of the results of this part so far were already published in [22, 23]. In Section 2.7, we will investigate how much of the analysis done so far can be carried over to the semi-implicit discretization introduced in [13, 15]. Although we will be faced with several issues, we will nevertheless give the formally derived first order condition and Hessian matrix to be able to compare this scheme to the implicit one in the numerics.
In Part 3 we are concerned with the algorithmic and implementational aspects of the optimal control problem. First, we will introduce the steepest descent method and trust region method with Steihaug-CG we implemented to tackle the problem on the computer in Section 3.1. Due to performance advantages we will stick to the latter in the numerical part. A similar method was applied in [24] for the isotropic Allen-Cahn equation. We will continue to discuss several preconditioning approaches for the full system as well as for the Steihaug-CG method applied to the reduced system in Section 3.2. Here, to the author's knowledge, only [20] considers the preconditioning of a comparable Allen-Cahn control problem. Then, in Section 3.3, several remarks about some important details of the implementation are given. We will explain how the partial differential equations were implemented in C++ and FEniCS [5], how the program was parallelized and the usefulness of mesh adaptivity in the present context.
The thesis is completed with a thorough investigation of the numerical solution of the control problem in Part 4. First, in Sections 4.1 and 4.2, we will verify previous theoretical findings concerning the dependence on the regularization parameter and the mesh. In Section 4.3 we will provide the results of simulations of several settings that show interesting behavior like topology changes or the formation of shapes with extreme excrescences. These will be further investigated with respect to the behavior of the algorithm in Section 4.4 and we will compare the implicit scheme to the semi-implicit one in Section 4.5.1. Generally, in Section 4.5 some of the implementational aspects from before will also be considered from the numerical point of view. Some of the numerical results from this part can also be found in [23].
Finally, in Part 5, the results of this thesis are summarized and an outlook on possible further investigations is given.

1.1 Overview of the occurring equations and quantities

As already mentioned in the previous section, the scope of this thesis comprises the analysis of (1.8)–(1.9) on both, a time discrete and a continuous level, as well as the provision of a numerical solver for the control problem. The present section tries to give an overview of the appearing equations by presenting them as they logically appear in the context of deriving first order conditions for the control problem using a Lagrangian-based point of view. The following discussion is meant for illustrative purposes only and we stress that the derivations are intended to be short, therefore are rather formal, and make no claim to mathematical rigor. Rigorous

derivations will be addressed in the subsequent chapters. We note that this section is inspired by the presentation in [79, Section 1.6].
We begin by considering an optimization problem in the more abstract form

$$\min_{y,u} J(y,u) \qquad \text{subject to} \qquad e(y,u)=0, \tag{1.11}$$

where $e(y,u)$ stands for a general state equation. In many cases, given u the equation $e(y,u)=0$ can be solved uniquely for the state y and we write $y(u)=Su$ in this case, where S denotes the solution operator of the state equation. Inserting this into the cost functional of (1.11) we can define the *reduced* control problem by

$$\min_{u} j(u) \coloneqq J(y(u),u). \tag{1.12}$$

For this control problem we can readily write down the first order condition in terms of the gradient of the *reduced cost functional* j, namely $\nabla j(u)=0$.
Since the definition of j contains the solution operator S of the state equation whose exact form is not known a priori (and is costly to compute for the numerics), the next task is to find a practical definition of said gradient. One quick way to derive such a representation is by means of the formal Lagrangian method. This gives an idea on how the gradient might be represented by introducing a so-called *adjoint state.* That this really gives a correct definition of the gradient and that the derived adjoint equation is well defined then still remains to be shown afterwards and will be done for our control problem in Section 2.4. As a starting point, let us define the Lagrangian associated with the problem (1.11)

$$L(u,y,p) \coloneqq J(y,u) - (p, e(y,u)), \tag{1.13}$$

where p is the Lagrange multiplier. The KKT-theory relates the first order conditions of (1.11) to the gradient of the Lagrangian (1.13). Inserting the solution of the state equation $y(u)$, one recovers the reduced cost functional

$$j(u) = J(y(u),u) = L(u,y(u),p). \tag{1.14}$$

We note that p is arbitrary so far but we are free to choose $p(u)$ satisfying some special condition, which we will do in the following to derive a useful expression for the gradient of j. Using the relation (1.14), we can write down the following expression for the derivative of j applied to a direction d

$$j'(u)[d] = D_uL(u,y(u),p(u))[d] + D_yL(u,y(u),p(u))[D_uy(u)[d]] + D_pL(u,y(u),p(u))[D_up(u)[d]]. \tag{1.15}$$

We will now look closer to each of the three terms. By the definition (1.13), we obtain

$$D_uL(u,y(u),p(u))[d] = J_u(y(u),u)[d] - (p(u), e_u(y(u),u)[d]). \tag{1.16}$$

This cannot be simplified further without additional knowledge of the special form of the cost functional J and the state equation defining $e(y,u)$. For the derivative with respect to y we get

$$\begin{aligned} D_yL(u,y(u),p(u))[\underbrace{D_uy(u)[d]}_{:=dy}] &= J_y(y(u),u)[dy] - (p(u), e_y(y(u),u)dy) \\ &= (J_y(y(u),u) - e_y(y(u),u)^*p(u), dy). \end{aligned} \tag{1.17}$$

Finally, the last contribution vanishes due to the definition of the state $y(u)$

$$D_pL(u,y(u),p(u))[D_up(u)[d]] = -(D_up(u)[d], \underbrace{e(y(u),u)}_{=0}) = 0. \tag{1.18}$$

$$\begin{tikzcd}
(\mathrm{OP}) \arrow[r, "\text{time discretization}"] \arrow[d] & (\mathrm{OP})_\tau \arrow[r] \arrow[d, "\text{optimization}"'] & (\mathrm{OP})_{\tau,h} \arrow[d] \\
(\mathrm{OP})^{(k)} \arrow[r] & (\mathrm{OP})_\tau^{(k)} \arrow[r, "\text{space discretization}"] & (\mathrm{OP})_{\tau,h}^{(k)}
\end{tikzcd}$$

Figure 1.1: Schematic description of the ways the optimization problem can be tackled with respect to discretization and optimization. The path we take in this thesis is indicated by the thick arrows.

In total we are left with the contributions from (1.16) and (1.17) that still are not of a particularly convenient form. At this point we should recall that so far we left open the choice of the multiplier $p(u)$ which now we are free to impose in order to eliminate the contribution from (1.17). As we want this term to vanish for all directions d and hence all dy (provided the linearization is onto), we obtain that $p(u)$ should satisfy the following *adjoint equation*

$$e_y(y(u),u)^*p(u) = J_y(y(u),u). \tag{1.19}$$

The derivative of j is then determined from this and (1.16). Summarized, we recover the state equation by demanding the derivative of the Lagrangian (1.13) with respect to p to vanish and likewise we obtain the adjoint equation (1.19) by differentiating with respect to y and the gradient of j by differentiating with respect to u. Let us further specify (1.16) for the most common cases. Typically we are confronted with functions of the form

$$J(u,y) = \tfrac{\alpha}{2}\|u\|^2 + q(y), \qquad e(y,u) = A(y) - Bu,$$

where q is a functional only depending on y, A is possibly a nonlinear operator and B is some linear operator. Then we have $J_u(y,u)[d] = \alpha(u,d)$ and $e_u(y,u)[d] = -Bd$ and by taking the Hilbert space representative, altogether one obtains

$$\nabla j(u) = \alpha u + B^*p(u), \tag{1.20}$$

with $p(u)$ solving

$$A_y(y(u))^*p = q_y(y(u)). \tag{1.21}$$

If A was linear, we would simply have $A_y(y) = A$. By a similar procedure as above one can also derive a representation of the Hessian of j. Since in this thesis this is only needed as part of an algorithm (see Algorithm 3), we will content ourselves with just stating an expression for the special choices given above, which can be thought of as a linearization of (1.20)

$$\nabla^2 j(u)\delta u = \alpha\delta u + B^*\delta p(u,\delta u). \tag{1.22}$$

The vector δp can be obtained by first solving a linearization of the state equation and subsequently a linearization of the adjoint equation. For more details consult, e.g., [79, Section 1.6.5]. What these equations look like for the control problem considered in this thesis is given below.

We are now in position to discuss the approach we pursue in this thesis to tackle problem (1.8)–(1.9). In principle we have discussed all the equations one needs to find a candidate for a local optimizer. Since we are out for the numerics, we have to discretize the appearing equations at a certain point. In our case, the state y is dependent on the spatial variable x and time t with regard to those it can potentially be discretized separately. All these possibilities are summarized in Figure 1.1. Arrows pointing to the right indicate discretization, first with respect to time (τ) and then with respect to space (h). Arrows pointing downwards indicate solving the first order system obtained by the just shown formalism ((k), representing some iteration number counter). For instance following all the arrows in the first line to $(\mathrm{OP})_{\tau,h}$ and then going down to $(\mathrm{OP})_{\tau,h}^{(k)}$ results in the approach *first-discretize-then-optimize* (DO), where the first order conditions are set up for the fully discrete system. Likewise, going via $(\mathrm{OP})^{(k)}$ is known as *first-optimize-then-discretize* (OD) and in this case the equations are not discretized before the optimization procedure stands. The approaches (DO) and (OD) may result in different algorithms, but in case the state equation is discretized implicitly in time as we do and appropriate space discretizations are chosen for y, u and p, they typically commute. In this case the resulting algorithm should be identical also for the mixed approach we take here. This one is indicated in the graphic by the thick arrows and consists of first discretizing the state equation only in time, then setting up the optimality conditions for the resulting control problem and finally conclude with the space discretization. Invoking said time discretization, in chapter 2 we are able to show existence of global solutions to the optimal control problem for both, the time discrete version as well as the time continuous counterpart. In Section 2.7 we will also discuss another possibility to discretize the equations in time given by [13, 15]. Here, due to the semi-implicit nature, the commutativity of (DO) and (OD) is no longer given then. The final space discretization will be done automatically by the tools of the FEniCS framework [5] we use for our implementation. As this is carried out by a straightforward final element approach, the fully discrete equations will not be discussed explicitly within the scope of this work.

That all said, we shall conclude this section by giving explicit expressions for the equations that arise from the previous discussion applied to the concrete problem (1.8)–(1.9). We will only give the time continuous equations in there classical formulation as they are the most memorable and we have not discussed the discretization yet. When the discrete versions are introduced we will back-reference to their counterparts here. First, let us repeat the state equation for the reader's convenience

$$\begin{aligned} \varepsilon\partial_t y - \varepsilon\nabla\cdot A'(\nabla y) + \frac{1}{\varepsilon}\psi'(y) &= u &&\text{in } Q,\\ A'(\nabla y)^T\nu &= 0 &&\text{on } \Sigma,\\ y(0) &= y_0 &&\text{in } \Omega. \end{aligned} \tag{1.23}$$

Existence of a unique weak solution for this and its discrete counterpart will be subject of Section 2.2. Inserting (1.23) and (1.8) into the Lagrangian (1.13), deriving with respect to y as in (1.17), one obtains for the adjoint equation (1.21) in this case

$$\begin{aligned} -\varepsilon\partial_t p - \varepsilon\nabla\cdot(A''(\nabla y)\nabla p) + \frac{1}{\varepsilon}\psi''(y)p &= 0 &&\text{in } Q,\\ \partial_{\nu_A} p &= 0 &&\text{on } \Sigma,\\ p(T) &= \frac{1}{\varepsilon}(y(T) - y_\Omega) &&\text{in } \Omega, \end{aligned} \tag{1.24}$$

with $\nu_A := A''(\nabla y)\nu$. At this point we should again emphasize that this equation may only hold on a formal level since A is not twice differentiable in general. In the course of this thesis, we will circumvent this problem by regularizing the function A. Having determined the adjoint state, one can readily compute the gradient from

$$\nabla j(u) = \tfrac{\lambda}{\varepsilon}u + p(u), \tag{1.25}$$

cf. (1.20). The representation (1.25) via the adjoint state (1.24) will be shown rigorously in more detail for the implicit time discretization at the end of Section 2.4.

Although relation (1.25), which is required for the gradient method, will be derived rigorously in the course of this thesis, for the numerics we will fall back on Newton's method, as the latter converges way earlier. Therefore, we supplementary give the linearized equations. However, also their time discretizations will only be derived formally later. The linearized state equation can be obtained by varying (1.23) with respect to u in direction δu and with the definition

$\delta y := D_u y(u)\delta u$ is given by

$$\begin{aligned} \varepsilon\partial_t\delta y - \varepsilon\nabla\cdot(A''(\nabla y)\nabla\delta y) + \frac{1}{\varepsilon}\psi''(y)\delta y &= \delta u && \text{in } Q,\\ \partial_{\nu_A}\delta y &= 0 && \text{on } \Sigma,\\ \delta y(0) &= 0 && \text{in } \Omega. \end{aligned} \tag{1.26}$$

Apart from the sign before ∂_t and the end point value that has become an initial value, it is of the same nature as the adjoint equation (1.24). Linearizing the latter results in

$$\begin{aligned} -\varepsilon\partial_t\delta p - \varepsilon\nabla\cdot(A''(\nabla y)\nabla\delta p) + \frac{1}{\varepsilon}\psi''(y)\delta p &= \varepsilon\nabla\cdot(A'''(\nabla y)[\nabla p,\nabla\delta y]) - \frac{1}{\varepsilon}\psi'''(y)p\delta y && \text{in } Q,\\ \partial_{\nu_A}\delta p &= 0 && \text{on } \Sigma,\\ \delta p(T) &= \frac{1}{\varepsilon}\delta y(T) && \text{in } \Omega. \end{aligned} \tag{1.27}$$

Note that the appearance of the third derivatives on the right-hand side results from the chain rule. It possibly diverges at the origin. The computed additional adjoint state δp can be used to determine the application of the Hessian to a vector δu by

$$\nabla^2 j(u)\delta u = \tfrac{\lambda}{\varepsilon}\delta u + \delta p. \tag{1.28}$$

This allows for the efficient use of an iterative method later (see Section 3.1.2), without the need of setting up the complete fully discretized Hessian.

1.2 Notation and auxiliary results

In this section we shall fix the notation that is used throughout the thesis and provide some important theorems that we will use in the following. Most statements can be found in the standard literature on linear and nonlinear functional analysis as [8, 139, 140, 117]. First, recall from the introduction that $\Omega \subset \mathbb{R}^d$ throughout the thesis denotes some bounded Lipschitz domain, where $d \geq 1$ is arbitrary for now and will be restricted later if necessary. A (real) *Banach space* V is a real vector space equipped with a norm $\|\cdot\|_V : V \to \mathbb{R}_{>0}$ with respect to which it is complete . The most common Banach spaces used in this thesis are the Lebesgue spaces $L^p(\Omega)$ with $1 \leq p \leq \infty$, the Sobolev spaces $W^{k,p}(\Omega)$ with $k \in \mathbb{N}$ and the Bochner spaces $L^p(0,T;X)$ where X denotes some other Banach space. Note that if $X = L^p(\Omega)$ the latter can be identified with $L^p(Q)$. Furthermore the space of Banach-valued L^p-functions $W^{k,p}(0,T;X)$ can be defined by demanding that the derivatives with respect to time up to the k-th order lie in $L^p(0,T;X)$. For a Banach space V we will denote its dual by V' and the duality product by $\langle\cdot,\cdot\rangle_V$. It is well-known that for the Lebesgue and Sobolev spaces, the dual spaces can be represented in terms of another one with exponent $p' = \frac{p}{p-1}$ if $1 \leq p < \infty$. The dual of the Bochner space is given by $L^{p'}(0,T;X')$. If a Banach space in addition is equipped with a scalar product that relates to its norm by $\|\cdot\|_V = \sqrt{(\cdot,\cdot)_V}$, then it is called a *Hilbert space.* For $p = 2$ and X being an Hilbert space the above spaces are Hilbert spaces. Here we use the common notation $W^{k,2}(\Omega) = H^k(\Omega)$. An important property of a Hilbert space is that it is isometrically isomorphic to its dual space (by Riesz representation theorem). If no subscripts are provided, by $(\cdot,\cdot)$ and $\|\cdot\|$ we denote the L^2- or l_2-scalar product and norm, respectively. The space should be clear from the context.
Next, we will briefly repeat the different notions of convergence on Banach spaces.

Definition 1.2.1. Let V be a Banach space.
A sequence $\{v_k\}_k \subset V$ is said to converge

1. *strongly* to $v \in V$ if $\lim_{k\to\infty}\|v_k - v\|_V = 0$, denoted by $v_k \to v$ in V,
2. *weakly* to $v \in V$ if $\langle v_k, v'\rangle_V \overset{k\to\infty}{\to} \langle v, v'\rangle_V$ for all $\forall v' \in V'$, denoted by $v_k \rightharpoonup v$ in V.

A sequence $\{v'_k\}_k \subset V'$ is said to converge

3. *weakly-$*$* to $v' \in V'$ if $\langle v, v'_k\rangle \overset{k\to\infty}{\to} \langle v, v'\rangle$ for all $v \in V$, denoted by $v'_k \overset{*}{\rightharpoonup} v'$ in V'.

In particular strong convergence implies weak and weak-$*$ convergence. For the next result we have to clarify two further terms. A Banach space is called *separable* if it contains a countable subset that is dense. It is called *reflexive* if the mapping $v \in V \mapsto \langle \cdot, v\rangle \in (V')'$ is surjective. Hilbert spaces are always reflexive. The Banach spaces given above are separable for $1 \leq p < \infty$ and if X is separable, and reflexive for $1 < p < \infty$ and if X is reflexive. The following theorem summarizes some important auxiliary results on weak convergence that we will use from time to time in this thesis (see, e.g., [8, chapter 8]).

Theorem 1.2.2. *Let V be a Banach space.*

1. *If V is reflexive, then for every bounded sequence $\{v_k\}_k \subset V$, there exists a weakly convergent subsequence.*
2. *If V is separable, then for every bounded sequence $\{v'_k\}_k \subset V'$, there exists a weakly-$*$ convergent subsequence.*
3. *Conversely, each weakly (-$*$) convergent sequence is bounded.*
4. *If V is a Hilbert space, then for every sequence $\{v_k\}_k \subset V$ the following equivalence holds*

$$v_k \rightharpoonup v,\ \|v_k\| \to \|v\| \qquad \Leftrightarrow \qquad v_k \to v \text{ in } V.$$

Since $L^1(\Omega)$ and $L^\infty(\Omega)$ are not reflexive, they are not weakly compact. However, as $L^1(\Omega)$ is separable, we have at least that bounded sets in $L^\infty(\Omega)$ posses a weakly-$*$-convergent subsequence. In the special case of Lebesgue spaces the following two results on weak convergence will prove to be useful.

Theorem 1.2.3.

1. *Let a sequence $f_k \in L^p(\Omega)$ with $1 \leq p \leq \infty$ be given. If $f_k \rightharpoonup f$ in $L^p(\Omega)$ and $f_k \to \tilde f$ almost everywhere for $k \to \infty$, then $f = \tilde f$ almost everywhere.*
2. *Let $f_k \to f$ in $L^p(\Omega)$ and $g_k \rightharpoonup g$ in $L^q(\Omega)$ with $1 < p < \infty$ and $q = \frac{p}{p-1}$.*
   Then $f_k g_k \rightharpoonup fg$ in $L^1(\Omega)$.

Proof:

1. See, e.g., [8, E8.1].
2. Using Hölder's inequality 1.2.5, one obtains for all $\phi \in L^\infty(\Omega)$

$$\begin{aligned}\left|\int_\Omega f_k g_k \phi \,\mathrm{d}x - \int_\Omega f g \phi \,\mathrm{d}x\right| &= \left|\int_\Omega (f_k - f) g_k \phi + f(g_k - g)\phi \,\mathrm{d}x\right| \\ &\leq \|f_k - f\|_{L^p(\Omega)} \|g_k\|_{L^q(\Omega)} \|\phi\|_{L^\infty(\Omega)} + \left|\int_\Omega (g_k - g) f \phi \,\mathrm{d}x\right|.\end{aligned} \tag{1.29}$$

The statement now follows from the assumed convergences (note that $f\phi \in L^p(\Omega)$) and the fact that weak convergent sequences are bounded. □

The following theorem finds frequent application in the theory of optimal control.

Theorem 1.2.4 (weak lower semicontinuity of convex functionals)**.** *Let V be a Banach space. Then any continuous and convex functional $F : V \to \mathbb{R}$ is weakly lower semicontinuous, i.e.*

$$v_k \rightharpoonup v \qquad \Rightarrow \qquad \liminf_{k\to\infty} F(v_k) \geq F(v).$$

We note that also norms are weakly (-$*$) lower semicontinuous.
In what follows, we collect several common inequalities and imbedding theorems.

Theorem 1.2.5 (Hölder's inequality, [8, Lemma 3.18]). *Let be $1 \leq p_i \leq \infty$ for $i = 1, \ldots, n$ and $1 \leq q \leq \infty$ such that*

$$\tfrac{1}{p_1} + \ldots + \tfrac{1}{p_n} = \tfrac{1}{q}.$$

Then given $f_i \in L^{p_i}(\Omega)$ for all $i = 1, \ldots, n$, the product $f_1 \ldots f_n$ lies in $L^q(\Omega)$ and it holds

$$\|f_1 \ldots f_n\|_{L^q(\Omega)} \leq \|f_1\|_{L^{p_1}(\Omega)} \cdots \|f_1\|_{L^{p_n}(\Omega)}.$$

Along the lines of the proof of [8, (3-11)] the following generalization to well-known Young's inequality can be shown.

Theorem 1.2.6 (scaled Young's inequality). *Let $a, b \in \mathbb{R}$ and $\varepsilon > 0$, then it holds*

$$ab \leq \tfrac{a^2}{2\varepsilon} + \tfrac{\varepsilon b^2}{2}.$$

We continue by recalling Gronwall's inequality (see, e.g., [87, A.1]).

Theorem 1.2.7 (Gronwall lemma).
Let $T > 0$ and $c \geq 0$. Let $u : [0, T] \to \mathbb{R}$ be measurable and bounded and $v : [0, T] \to \mathbb{R}$ be continuous and nonnegative. If

$$u(t) \leq c + \int_0^t v(s)u(s)\,\mathrm{d}s \qquad \forall t \in [0, T],$$

then

$$u(t) \leq c \exp\left(\int_0^t v(s)\,\mathrm{d}s\right) \qquad \forall t \in [0, T].$$

When considering the discrete equations, the following version of the discrete Gronwall lemma taken from [87, A.3] will be useful.

Theorem 1.2.8 (discrete Gronwall lemma).
Let $c \geq 0$ and $\{u_j\}_{j \geq 1}$ as well as $\{v_j\}_{j \geq 1}$ be nonnegative sequences. If

$$u_j \leq c + \sum_{i=1}^{j-1} v_i u_i \quad \textit{for } j \geq 1,$$

then

$$u_j \leq c \prod_{i=1}^{j-1} (1 + v_i) \leq c \exp\left(\sum_{i=1}^{j-1} v_i\right) \quad \textit{for } j \geq 1.$$

The next result is frequently used and can be found for instance in [44, Theorems 12.11 and 12.12].

Theorem 1.2.9 (Sobolev imbedding theorem). *Let Ω be as above. For $k, l \in \mathbb{N}_0$, $1 \leq p < \infty$, $1 \leq q$ as well as*

$$k - \tfrac{d}{p} \geq l - \tfrac{d}{q} \qquad \textit{and} \qquad k \geq l,$$

the imbedding

$$W^{k,p}(\Omega) \hookrightarrow W^{l,q}(\Omega) \tag{1.30}$$

holds.
Furthermore, if

$$k - \tfrac{d}{p} > l - \tfrac{d}{q} \qquad \textit{and} \qquad k > l,$$

then said imbedding is compact.

By a compact imbedding we understand that a sequence bounded in the former space has a strongly convergent subsequence in the other space. Here for $k = 0$, we define $W^{0,p}(\Omega) := L^p(\Omega)$. In particular for $d \le 3$ it holds $H^1(\Omega) \hookrightarrow L^6(\Omega)$ and for $d \le 2$ this imbedding is compact.
We continue with results on Bochner spaces. First, recall the following continuous imbedding

$$L^2(0,T;H^1(\Omega)) \cap H^1(0,T;H^1(\Omega)') \hookrightarrow C([0,T];L^2(\Omega)),$$

see, e.g., [139, Proposition 23.23] for a proof in a more general setting. This result is important as it renders the initial and final conditions of the appearing parabolic PDEs to be well defined. Concerning compact imbeddings, the following results are useful and can for instance be found in [28, Theorem II.5.16] or [120] (only for reflexive spaces).

Theorem 1.2.10 (Aubin-Lions-Simon). *Let $X \subset Y \subseteq Z$ be three Banach spaces with continuous imbedding $Y \hookrightarrow Z$ and compact imbedding $X \hookrightarrow Y$. Further let $1 \le p, q \le \infty$.*

1. *If $p < \infty$, then the imbedding*

$$L^p(0,T;X) \cap W^{1,q}(0,T;Z) \hookrightarrow L^p(0,T;Y)$$

is compact.

2. *If $q > 1$, then the imbedding*

$$L^\infty(0,T;X) \cap W^{1,q}(0,T;Z) \hookrightarrow C([0,T];Y)$$

is compact.

We note that without further information the imbedding $L^2(0,T;H^1(\Omega)) \hookrightarrow L^2(0,T;L^2(\Omega))$ is not compact, as for example for functions constant in space the norms of both function spaces coincide.
When considering first order conditions of our optimal control problem, we need a notion of differentiability on Banach spaces. The most common ones are summarized by the next definition.

Definition 1.2.11 (Gâteaux and Fréchet differentiability). Let X and Y be Banach spaces and $F : U \subseteq X \to Y$ with U being nonempty and open.

1. If for $u \in U$ and given $h \in X$ the limit

$$dF(u,h) = \lim_{t \searrow 0} \frac{F(u+th) - F(u)}{t} \in Y$$

exists, it is called the *directional derivative* of F in direction h.
If this limit exists for all $h \in X$, then the mapping $h \mapsto dF(u,h)$ is called *first variation* of F at u.

2. If in addition for given $u \in U$, the mapping $F'(u) : X \ni h \mapsto dF(u,h) \in Y$ is bounded and linear, i.e., $F'(u) \in \mathcal{L}(X,Y)$, then F is called *Gâteaux differentiable* at u.

3. If F is Gâteaux differentiable in every $v \in B_r(u) \subset U$ for a $r > 0$ and in addition it holds

$$\frac{\|F(u+h) - F(u) - F'(u)h\|_Y}{\|h\|_X} \to 0 \qquad \text{for} \qquad \|h\|_X \to 0,$$

then F is called *Fréchet differentiable* at u.

4. If for F one of the above three properties holds at every $u \in V$ with $V \subseteq U$ open, then F is called directionally, Gâteaux or Fréchet differentiable on V, respectively.

An important relation between Gâteaux and Fréchet differentiability is given by the following theorem (see, e.g., [117, Satz 2.5]).

Theorem 1.2.12. *Let X and Y be Banach spaces and $F : X \to Y$ be Gâteaux differentiable in $U \subseteq X$ open. If F' is continuous in $u_0 \in U$, then F is Fréchet differentiable in u_0.*

The next theorem demonstrates that Fréchet and Gâteaux differentiable functions satisfy a chain rule if they are combined appropriately.

Theorem 1.2.13 (chain rule)**.** *Let X, Y and Z be Banach spaces, $U \subseteq X$, $V \subseteq Y$ open subsets and $F : V \to Z$ as well as $G : U \to Y$, where $G(U) \subseteq V$.*

1. *If G is Fréchet differentiable in $x \in U$ and F is Fréchet differentiable in $G(x) \in V$, then $F \circ G$ is Fréchet differentiable.*
2. *If G is Gâteaux differentiable in $x \in U$ and F is Fréchet differentiable in $G(x) \in V$, then $F \circ G$ is Gâteaux differentiable.*

In both cases the derivative of $F \circ G : U \to Z$ can be written as

$$(F \circ G)'(x) = F'(G(x))G'(x), \tag{1.31}$$

where with F' and G' we denote the kind of derivative that was assumed, respectively.

Proof:

1. See, e.g., [117, Satz 2.7].
2. By the definition of the Gâteaux derivative we can write $G(x+th)-G(x) = tG'(x)[h]+o(t)$ and therefore we get

$$\begin{aligned}
d(F \circ G)(x,h) &= \lim_{t\searrow 0} \tfrac{1}{t}\left(F(G(x+th)) - F(G(x))\right)\\
&= \lim_{t\searrow 0} \tfrac{1}{t}\left(F(G(x) + tG'(x)[h] + o(t)) - F(G(x))\right)\\
&= \lim_{t\searrow 0} \tfrac{1}{t}\big(F(G(x) + t(G'(x)[h] + \tfrac{1}{t}o(t))) - F(G(x))\big)\\
&= \lim_{t\searrow 0} \tfrac{1}{t}\big(F(G(x)) + tF'(G(x))[G'(x)[h] + \tfrac{1}{t}o(t)] + o(\|tG'(x)[h] + o(t)\|_Y) - F(G(x))\big)\\
&= \lim_{t\searrow 0} \big(F'(G(x))[G'(x)[h] + \tfrac{1}{t}o(t)] + \tfrac{1}{t}o(t\|G'(x)[h] + \tfrac{1}{t}o(t)\|_Y)\big)\\
&= F'(G(x))[G'(x)[h]].
\end{aligned}$$

   For the limit we used in the first term the linearity of $F'(G(x))$ and for the second term that $\|G'(x)[h] + \frac{1}{t}o(t)\|_Y \le C$ for fixed h and small t. Noting that this representation holds for every h and $F'(G(x))G'(x)$ is linear and bounded the assertion follows. □

Note that F being Gâteaux differentiable is not enough in the second statement. The proof would not work out as the direction $tG'(x)[h]+o(t)$ cannot be written in the form tv with fixed v. Indeed there exist explicit counterexamples already for finite dimensional vector spaces [2]. However, if the Gâteaux derivative of $F \circ G$ exists (e.g. because it is actually Fréchet differentiable), it has to be of the form (1.31).

[2]Take $\mathcal{M} = \{(a,b) \in \mathbb{R}^2 \mid a^2 < b < 3a^2\}$ and $G : \mathbb{R} \to \mathcal{M} \subset \mathbb{R}^2, x \mapsto (x^2, 2x^4)$. Then G is Gâteaux (and even Fréchet) differentiable with $G'(0) = (0,0)$. Choose $F : \mathbb{R}^2 \to \mathbb{R}$ which has only support in $\mathcal{M}$, satisfies $F(s, 2s^2) = \sqrt{s}$ for $s \ge 0$ and is Gâteaux differentiable. Since for every line $\mathcal{L} \subset \mathbb{R}^2$ with $(0,0) \in \mathcal{L}$ there exists an $\varepsilon > 0$ such that it holds $\mathcal{L} \cap \mathcal{M} \cap B_\varepsilon(0) = \{(0,0)\}$, the last two requirements don't contradict each other and such a construction is indeed possible (with $dF(0,h) = 0$ for all $h \in \mathbb{R}^2$). However it holds $F(G(x)) = \sqrt{x^2} = |x|$ and thus the combination is not Gâteaux differentiable at the origin. The problem is that F is not Fréchet differentiable. This example was inspired by [83].

2

Analytical results

The first part of this thesis is dedicated to an analytical treatment of the optimal control problem (1.8)–(1.9) with the goal to provide an existence result for it, as well as providing the necessary first order conditions that are crucial for the numerics later. As a first step we will study the state equation (1.9). Therefore, in Section 2.2 we introduce an implicit time discretization by piecewise constant functions that will be utilized both, in the proof of the existence result as well as for the implementation. Then we discuss the existence and uniqueness of the solution of the discretized state equation as well as the Lipschitz continuity of the pertinent control-to-state operator. Furthermore, for a set of bounded controls we obtain bounds on the states that are independent of the discretization level. Using these results we consider the limit with respect to the time discretization and obtain the corresponding results also for the in time continuous state equation (1.9). Consequently, we also obtain convergence of the discretization. For the latter, we in addition show energy stability. As a next step, in Section 2.3 the existence of optimal solutions is shown in the time continuous and time discrete case. In addition the convergence of a subsequence of time discrete global optimal solutions to an optimal control of the original problem is obtained. These results do not only hold for the aiming at an end time state, but also for steering to a prescribed function over the whole time horizon. Furthermore, up to this point, the results hold in arbitrary space dimension. In Section 2.4 we study under stricter assumptions the Fréchet differentiability of the reduced cost functional for the time discretized problem. As a first step we analyze the differentiability of one time step of the state equation. Then, due to the implicit discretization, one can successively prove differentiability of the control-to-state operator and of the reduced cost functional. The corresponding time discrete adjoint equation is deduced rigorously. Subsequently, in Section 2.5 we give sufficient conditions on the regularization A_δ of A such that the corresponding states converge to the solution of the formerly given state equation in the limit $\delta \to 0$. Furthermore, also for global minimizers u_τ^δ the convergence of a subsequence with respect to the regularization parameter δ and the time discretization coarseness τ is addressed. All results hold under rather general assumptions on A. In the subsequent section 2.6 we study the class of anisotropies given in [16, 14], that we will also use later for the numerics. For this class we introduce a regularization that is done by modifying γ (cf. (1.10)). Furthermore, we show that A fulfills the formerly stated assumptions that were required for the existence results of the state equation and the control problem, and that the regularization A_δ fulfills additional assumptions that are needed for obtaining Fréchet differentiability. Finally, in Section 2.7 we discuss a semi-implicit time discretization of the state equation, which is based on that from the just given references where the special choice of A is used. We will see that, although for merely solving the forward problem this might be a reasonable choice, adapting above results for optimal control turns out to be difficult, since the

sensitivity analysis with respect to the right hand side is now more involved. This is one of the reasons why we prefer the implicit time discretization in this thesis. Let use remark that parts of the results of this chapter were already published in [22, 23].
To the best of the author's knowledge, there does not exist any mathematical treatment of the optimal control of anisotropic phase field models so far. Control problems involving isotropic phase field models in contrast are studied, e.g., in [81, 88], and such involving isotropic Allen-Cahn variational equations in [20, 24, 40, 56, 106] and for Cahn-Hilliard variational (in-)equalities consult [77, 42, 78] and references therein. Let us further mention results given in the context of anisotropic Allen-Cahn equations. One possible anisotropy was introduced in a pioneering paper by Kobayashi [86] and existence and uniqueness of a solution for some corresponding phase field equations are studied in [30, 101, 126]. The kind of anisotropy we will introduce in Section 2.6 was first utilized in [16, 14]. For quite general anisotropies the solution of Allen-Cahn equations with obstacle potential is analyzed in [53]. Among others, the authors use 2-homogeneity of A, an approximation of the potential similar as we will do in (2.1) below and an implicit time discretization (without showing convergence of the discretization). Explicit and semi-implicit approximations are discussed in the survey paper [45], where also many references are given. For convex Kobayashi anisotropies, several time discretizations are considered in [64]. In [15, 13], an efficient semi-implicit method using a particular linearization of A' and a convex/concave splitting is presented for the particular kind of anisotropies that also we use. Furthermore, energy stability is shown. Also several numerical experiments are shown comparing the anisotropies.
Literature covering optimal control of quasilinear parabolic equations of the form (1.9) is still in its infancy. Most literature known to the author treats quasilinearities with coefficients depending on x, t and on the function y, but not on its gradient [26, 80, 32, 99, 100]. For quasilinearities involving spatial derivatives of y, we refer the reader to [103, 57, 38]. In particular, it is worth mentioning that, according to the author's assessment, the latter reference addresses a control problem that is most similar to the one addressed in this thesis. The authors require a rather general quasilinearity with some particular polynomial growth condition. However, they demand the nonlinearity ψ' to be monotone. All the literature listed here assumes the quasilinear term to be rather well behaved, in particular none of its derivatives shall be singular at the origin. In the present context, to the author's knowledge, such difficulties have only been considered for elliptic equations [36].

2.1 Assumed properties of A, ψ and their derivatives

As already mentioned above, throughout this chapter we will not work directly with the smooth double-well potential and a concrete anisotropy. We rather give some general assumptions that are also fulfilled by the special choices we will use later, see also Remark 2.1.3 below. Under particular circumstances these assumptions need some enhancement which will then be given in the respective subsections, see for instance Assumptions 2.4.1 and the assumptions of Theorem 2.5.1.

Assumptions 2.1.1. Assume $A \in C^1(\mathbb{R}^d)$ with A' being strongly monotone, i.e.,

$$(A'(p) - A'(q), p - q) \geq C_A |p - q|^2 \qquad \forall p, q \in \mathbb{R}^d,$$

with $C_A > 0$ and fulfilling the growth condition $|A'(p)| \leq C|p|$.
Furthermore let $\psi \in C^1(\mathbb{R})$ be bounded from below and such that it can be approximated by f_n satisfying

$$f_n \in C^2(\mathbb{R}), \quad f_n \to \psi \quad \text{in } C^1_{\text{loc}}, \quad -c \leq f_n \leq c(\psi + 1), \quad f_n'' \geq -C_\psi, \quad |f_n''| \leq C_n, \tag{2.1}$$

with $c, C_\psi \geq 0$, $C_n > 0$ and $\psi(y_0) \in L^1(\Omega)$ for the given initial data $y_0 \in H^1(\Omega)$.

Some examples of A and ψ with respect to Allen-Cahn equations are mentioned below. Note that the assumptions on ψ in particular imply

Lemma 2.1.2. *Let Assumptions 2.1.1 hold. Then*

$$(\psi'(y_1) - \psi'(y_2), y_1 - y_2) \geq -C_\psi |y_1 - y_2|^2 \qquad \forall y_1, y_2 \in \mathbb{R}. \tag{2.2}$$

Proof: First we assume that $\psi \in C^2(\mathbb{R})$ and show that (2.2) is then equivalent to

$$\psi''(y) \geq -C_\psi \qquad \forall y \in \mathbb{R}. \tag{2.3}$$

For the implication (2.2) $\Rightarrow$ (2.3), we set $y_1 = y + tv$ (with $v \neq 0$) and $y_2 = y$. Inserting this into (2.2) we obtain

$$(\psi'(y+tv) - \psi'(y))\, tv \geq -C_\psi t^2 v^2 \qquad \Leftrightarrow \qquad \left(\tfrac{\psi'(y+tv)-\psi'(y)}{t}\right) v \geq -C_\psi v^2.$$

Taking the limit $t \to 0$ we arrive at

$$\psi''(y) v^2 \geq -C_\psi v^2,$$

which yields (2.3) after dividing by $v^2 > 0$.
For the inverse implication we note that by the mean value theorem there exists $\xi_{y_1,y_2} \in [y_1, y_2]$ ($y_1 < y_2$ w.l.o.g.) such that $\psi'(y_1) - \psi'(y_2) = \psi''(\xi_{y_1,y_2})(y_1 - y_2)$.
From this and (2.3) we obtain

$$(\psi'(y_1) - \psi'(y_2))\,(y_1 - y_2) = \psi''(\xi_{y_1,y_2})(y_1 - y_2)^2 \geq -C_\psi |y_1 - y_2|^2,$$

which is exactly (2.2).
For $\psi \in C^1(\mathbb{R})$ as in Assumptions 2.1.1 we use the fact that it can be approximated by $f_n \in C^2(\mathbb{R})$. Then, for y_1, y_2 fixed, it follows by adding zero and applying the previous to the assumed $f_n'' \geq -C_\psi$

$$\begin{aligned}(\psi'(y_1) - \psi'(y_2), y_1 - y_2) &= ((\psi'(y_1) - f_n'(y_1) + (f_n'(y_2) - \psi'(y_2)) + (f_n'(y_1) - f_n'(y_2)), y_1 - y_2) \\ &\geq ((\psi'(y_1) - f_n'(y_1) + (f_n'(y_2) - \psi'(y_2)), y_1 - y_2) - C_\psi |y_1 - y_2|^2.\end{aligned}$$

The first term vanishes in the limit $n \to \infty$ by using $f_n' \to \psi'$ in C_{loc}. Since y_1, y_2 were arbitrary this shows (2.2) also in this case. □

As promised above, let us now comment on possible choices for the function A of the quasilinear term and the function ψ—in particular with respect to the application to optimal control of anisotropic Allen-Cahn equations.

Remark 2.1.3.
The Assumptions 2.1.1 on A are fulfilled when

1. $A \in C^1(\mathbb{R}^d)$ is convex, 2-homogeneous and satisfies $A(p) > 0$ for $p \neq 0$ as in [53][1]. In this case the analysis can actually be done with only $A \in C^0(\mathbb{R})$ as described in the reference. If $A(p) = \frac{1}{2}\gamma(p)^2$ then sufficient conditions on γ can be found, e.g., in [64]. In Section 2.6 we will discuss optimal control including anisotropies arising for $\gamma(p) = \sum_{l=1}^{L} \left(p^T G_l p\right)^{1/2}$ with symmetric positive definite $G_l \in \mathbb{R}^{d \times d}$ and show the relevant properties of γ. For the Allen-Cahn equation such anisotropies are studied in [15, 13].

2. $A(p) = \frac{1}{2}\left(\sum_{l=1}^{L} \left(p^T G_l p + \delta\right)^{1/2}\right)^2$ with symmetric positive definite matrices $G_l \in \mathbb{R}^{d \times d}$ and $\delta > 0$, which is a regularization of A given above. A is not 2-homogeneous but in $C^2(\mathbb{R}^d)$ and hence suitable for numerical optimal control approaches. Details will be found in Section 2.6.

The Assumptions 2.1.1 on ψ are fulfilled in the following cases:

[1]The growth condition $|A'(p)| \leq C|p|$ follows since A' is then 1-homogeneous. Furthermore from homogeneity and $A(p) > 0$ it follows $A(p) > c|p|^2$ with $c > 0$. Together with convexity this implies the uniform convexity of A and finally the strong monotonicity of A'.

1. if $\psi \in C^2(\mathbb{R})$ is bounded from below, $\psi'' \geq -C_\psi$ for some $C_\psi \geq 0$ and $\lim_{t\to\pm\infty} \psi''(t) = +\infty$.
   Because in this case one can choose, e.g., $x_{-1} > 0$ large enough such that $\psi'' \geq 1, \psi' \geq 0$ on $[x_{-1}, \infty)$ and define with $x_n := \operatorname{argmin}_{x\in[x_{n-1}+1,\infty)} \psi''(x)$ the approximation on $[0, x_n]$ by $f_n := \psi$, for $x > x_n$ as $f_n(x) := \psi(x_n) + \psi'(x_n)(x - x_n) + \frac{1}{2}\psi''(x_n)(x - x_n)^2$. Then use this construction respectively on $(-\infty, 0]$. Further details will be given in Lemma 2.1.4.

2. for the double-well potential
$$\psi(y) = \tfrac{1}{4}(y^2 - 1)^2, \tag{2.4}$$
   since then the conditions in 1. hold.

   In addition to the regularity $y \in L^\infty(0, T; H^1(\Omega))$ for the solution of (2.10) shown in Theorem 2.2.5 this potential yields together with the estimate (2.37) also the regularity $y \in L^6(0, T; L^6(\Omega))$ for all space dimensions.

3. for regularizations of the obstacle potential ψ_{obst} which is
$$\psi_{obst} := \begin{cases} \frac{1}{2}(1 - x^2) & \text{on } [-1, 1], \\ \infty & \text{else,} \end{cases} \tag{2.5}$$
   as for example:
   - the regularization considered in [25] for analyzing the solution of the isotropic Allen-Cahn or Cahn-Hilliard variational inequalities. There ψ_{obst} is regularized to $\psi \in C^2$ by a smooth continuation with a cubic polynomial in a neighborhood $\pm(1, 1 + \delta)$ and then by a quadratic polynomial (cf. formula (2.9) there).
   - the Moreau-Yosida regularization of ψ_{obst}, i.e., $\psi \in C^1$ with $\psi(x) = \frac{1}{2}(1 - x^2) + s(\min\{x+1, 0\})^2 + s(\max\{x-1, 0\})^2$. It is for instance used in [77] to study the optimal control of isotropic Allen-Cahn inequalities and to obtain a numerical approach.

As mentioned in Remark 2.1.3.1 for ψ, we will conclude the section by showing in more detail how the construction given there works.

Lemma 2.1.4. *Let $\psi \in C^2(\mathbb{R})$ with $\psi(x) \geq -C_\psi$, $\psi''(x) \geq -C_\psi$ for all $x \in \mathbb{R}$ and $C_\psi \in \mathbb{R}$, $\lim_{x\to\pm\infty} \psi''(x) = \infty$. Then there exists a sequence f_n that satisfies (2.1), i.e.*
$$f_n \in C^2(\mathbb{R}),\ -c \leq f_n \leq c(\psi + 1),\ f_n'' \geq -C_\psi,\ |f_n''| \leq C_n,\ f_n \to \psi \text{ in } C^1_{loc}.$$

Proof: In what follows, we only modify ψ outside some ball around 0, i.e., we can consider the modifications of $\psi_{|\mathbb{R}^+}$ and $\psi_{|\mathbb{R}^-}$ separately. That is, we will only show how we modify $\psi_{|\mathbb{R}^+}$ to treat the case $x \to \infty$.
There exists $\tilde{\tilde{x}}_0 > 0$ with $\psi''_{|[\tilde{\tilde{x}}_0,\infty)} \geq 1$, otherwise contradicting the divergence. On using
$$\psi'(x) = \psi'(\tilde{\tilde{x}}_0) + \int_{\tilde{\tilde{x}}_0}^{x} \underbrace{\psi''(s)}_{\geq 1}\,\mathrm{d}s,$$
we can choose $\tilde{x}_0 \geq \tilde{\tilde{x}}_0 > 0$ large enough to obtain $\psi''_{|[\tilde{x}_0,\infty)} \geq 1$ and $\psi'_{|[\tilde{x}_0,\infty)} \geq 0$. We fix such an $\tilde{x}_0$ and define
$$x_0 := \operatorname{argmin}_{[\tilde{x}_0,\infty)} \psi''(x), \tag{2.6}$$
and
$$f_0(x) := \begin{cases} \psi(x) & \text{for } 0 \leq x \leq x_0, \\ \psi(x_0) + \psi'(x_0)(x - x_0) + \dfrac{1}{2}\psi''(x_0)(x - x_0)^2 & \text{for } x_0 < x. \end{cases} \tag{2.7}$$
Using
$$f_0''(x) := \begin{cases} \psi''(x) & \text{for } 0 \leq x \leq x_0, \\ \psi''(x_0) & \text{for } x_0 < x, \end{cases}$$

one can readily check the first four properties for f_0. Let us just comment that for the second property the lower bound follows from $\psi'(x_0), \psi''(x_0) \geq 0$ and that the construction of x_0 in (2.6) ensures that the upper bound holds. Next define x_n for $n = 1, 2, \dots$ by

$$x_n := \operatorname{argmin}_{[x_{n-1}+1,\infty)} \psi''(x),$$

f_n analogously to (2.7) with x_0 replaced by x_n and observe that the first four properties analogously hold for f_n possibly with another constant C_n. It remains to show the convergence in $C^1_{\mathrm{loc}}(\mathbb{R})$ for the whole sequence $(f_n)_{n\in\mathbb{N}}$. Let us choose an arbitrary compact set $S \subset \mathbb{R}$. By construction $x_n > x_{n-1}$, so there exists an $N_+ \in \mathbb{N}$ such that for all $n \geq N_+$ we have $S \cap [0, x_n] = S \cap [0, x_{N_+}]$. If there had been a need, we find a similar N_- for the construction on $\mathbb{R}^-$, otherwise just set $N_- := 1$. Then for $n \geq \max(N_+, N_-)$ we have $\psi_{|S} \equiv f_{n|S}$ and the convergence follows trivially. □

We now turn without further ado to the existence of solutions of the state equation.

2.2 Solution of the time discretized and time continuous state equations

The goal of this section is to obtain the existence of a solution to the state equation (1.9) as well as of an implicit discretization thereof. We also derive bounds on the solution that will become useful later when we head over to investigating existence of the associated control problem. The approach in this section is based on [53] where the authors show the existence of a solution to an anisotropic Allen-Cahn equation with an anisotropy that is assumed to be homogeneous. Following the reference, we start by introducing the time discretization and show a certain boundedness property. This one will be used to demonstrate the convergence of the solution of the discretized problem to the time continuous one. That also shows convergence of the discretization method. To obtain the result, the potential ψ is approximated by a sequence of functions f_n with bounded second derivatives (as, e.g., in [53] and [34]), such that the dominated convergence theorem can be used. Following the lines of [53] we do not have to restrict the space dimension d.

Our approach differs by the more general assumptions we impose on A as well as the uniqueness and Lipschitz continuity that we show in addition. Furthermore, in [53] first the limit in the time discretization and then in the approximation of ψ is taken. In our proceeding this is different since we keep the results for the time discrete equation separate, since we will also have to deal with it later.

In order to not overload the notation, let us set $\varepsilon = 1$ in this and the next subsection. For instance the Ginzburg-Landau energy (1.1) now reads as

$$\mathcal{E}(y) := \int_\Omega A(\nabla y) + \psi(y)\,\mathrm{d}x, \tag{2.8}$$

and the control problem (1.8)–(1.9) is given by

$$\min J(y,u) := \frac{1}{2}\|y(T) - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|u\|^2_{L^2(Q)}, \tag{2.9}$$

where $y \in L^2(0,T;H^1(\Omega)) \cap H^1(0,T;L^2(\Omega))$ and $u \in L^2(Q)$, subject to the state equation

$$\begin{aligned} \int_Q \partial_t y\eta + A'(\nabla y)^T\nabla\eta + \psi'(y)\eta &= \int_Q u\eta \qquad && \forall \eta \in L^2(0,T;H^1(\Omega)), \\ y(0) &= y_0 && \text{in } \Omega. \end{aligned} \tag{2.10}$$

For introducing the time discretization, we divide the interval $[0,T]$ into subintervals $I_j := (t_{j-1}, t_j]$ for $j = 1, \dots, N$ with $0 = t_0 < t_1 < \dots < t_N = T$ and define $\tau_j := t_j - t_{j-1}$ and $\tau := \max_j \tau_j$. The discrete scheme for the state equation we obtain by employing a discontinuous-Galerkin method (dG(0), see, e.g., [54]). Therefore, let us define the spaces

$$
\begin{aligned}
Y_\tau &:= \{y_\tau : Q \to \mathbb{R} \mid y_\tau(t,.) \in H^1(\Omega), y_\tau(.,x) \text{ constant in } I_j \text{ for } j = 1,\dots,N\},\\
U_\tau &:= \{u_\tau : Q \to \mathbb{R} \mid u_\tau(t,.) \in L^2(\Omega), u_\tau(.,x) \text{ constant in } I_j \text{ for } j = 1,\dots,N\},
\end{aligned}
\tag{2.11}
$$

and for each interval we label the constant by a subscript, e.g., $y_j := y_\tau|_{I_j}$. Occasionally we will also use the vector containing these constants, i.e., $(y_j)_{j=1,\dots,N} \in H^1(\Omega)^N$. Apparently it holds $Y_\tau \subseteq L^\infty(0,T;H^1(\Omega))$ and $U_\tau \subseteq L^\infty(0,T;L^2(\Omega))$. The time discretized variant of eqs. (2.9) and (2.10) is then given by

$$
\min_{Y_\tau \times U_\tau} J(y_\tau, u_\tau) = \frac{1}{2}\|y_N - y_\Omega\|^2 + \frac{\lambda}{2}\sum_{j=1}^{N} \tau_j \|u_j\|^2, \tag{2.12}
$$

subject to the time discretized state equation

$$
(y_j, \varphi) + \tau_j(A'(\nabla y_j), \nabla\varphi) + \tau_j(\psi'(y_j), \varphi) = \tau_j(u_j, \varphi) + (y_{j-1}, \varphi) \qquad \forall \varphi \in H^1(\Omega), \tag{2.13}
$$

with $j = 1,\dots,N$. The function $y_\tau(0,.) := y_0 \in H^1(\Omega)$ is given.

We note that the state equation could have arised equally well from an implicit Euler discretization and we will use the notation $\partial_t^{-\tau} y$ with

$$
\partial_t^{-\tau} y_{\tau|I_j} := \tfrac{1}{\tau_j}(y_j - y_{j-1}).
$$

One may favour an approximation of the quasilinear term A as in [13, 15] or a splitting approach for ψ instead of the fully implicit method. However, to our knowledge there exists no convergence proof for these discretizations of the state equation to the time continuous one in the limit $\tau \to 0$. Even more important is that for solving the control problem we aim at differentiability of the control to state operator, which will be addressed in Section 2.4. For a semi-implicit discretization of the quasilinear term it could not be shown that this property holds. The reader is referred to Section 2.7.3 for a discussion about the problems arising here. Moreover, for the trust region method used later, we will see in Section 4.4 that the additional computational cost for solving the state equation is nearly negligible. Therefore it is reasonable to stick with an implicit discretization thereof.

As mentioned in the beginning, we start by showing existence of a solution to the time discretization and some bound satisfied by that solution. First ψ will be replaced by better suited function f, i.e., one of the f_n from Assumptions 2.1.1 can be taken. Note that if $A \in C^1(\mathbb{R}^d)$, the variational inequality (2.14) is equivalent to (2.13) (apart from the replacement with f) using Remark 2.2.2.

Lemma 2.2.1. *Let A fulfill the conditions in Assumptions 2.1.1. Furthermore, let $y_0 \in H^1(\Omega)$ and $u_\tau \in U_\tau$.*
Then for every $f \in C^2(\mathbb{R})$ with f bounded from below, $|f''|$ bounded and $f(y_0) \in L^1(\Omega)$ there exists for $\tau < \frac{1}{C_\psi}$ a $y_\tau \in Y_\tau$ which is a solution of $y_\tau(0) = y_0$ in Ω and

$$
\int_\Omega \partial_t^{-\tau} y_j(\eta - y_j) + A(\nabla\eta) - A(\nabla y_j) + f'(y_j)(\eta - y_j) - u_j(\eta - y_j) \geq 0 \qquad \forall \eta \in H^1(\Omega), \tag{2.14}
$$

for all $j = 1,\dots,N$.
Moreover, given a $\Lambda > 0$ then for all u_τ, y_0, A, f fulfilling in addition

$$
\|u_\tau\|_{L^2(0,T;L^2(\Omega))},\ \|y_0\|_{L^2(\Omega)} \leq \Lambda,\ -\Lambda + \Lambda^{-1}|p|^2 \leq A(p) \text{ and } A'(p)^T p \leq \Lambda|p|^2,
$$

as well as

$$
\int_\Omega (A(\nabla y_0) + f(y_0)) \leq \Lambda \quad \text{and} \quad f \geq -\Lambda, \quad f'' \geq -\Lambda, \tag{2.15}
$$

for all $\tau < \frac{1}{C_\psi}$ there exist solutions y_τ of (2.14) satisfying

$$\|\partial_t^{-\tau} y_\tau\|_{L^2(0,T;L^2(\Omega))} + \|y_\tau\|_{L^\infty(0,T;H^1(\Omega))} + \|f'(y_\tau)\|_{L^2(0,T;L^2(\Omega))} \le C(\Lambda). \tag{2.16}$$

Note that we consider directly the bounds defined by Λ since for fixed u_τ, y_0, A and f this constant can be chosen appropriately. In particular for A the constant can be found since the growth condition induces $A'(p)^T p \le C|p|^2$ as well as $A'(0) = 0$ and therefore the strong monotonicity of A' implies $A'(p)^T p \ge C_A|p|^2$ which gives

$$A(p) = A(0) + \int_0^1 A'(sp)^T p \,\mathrm{d}s \ge A(0) + \int_0^1 C_A s|p|^2 \,\mathrm{d}s = A(0) + \tfrac{1}{2}C_A|p|^2.$$

Proof: We note that f and f' induce Nemytskii operators $f : L^2(\Omega) \to L^1(\Omega)$ and $f' : L^2(\Omega) \to L^2(\Omega)$ due to the bounds on f''.
Starting with $y_0 := y(0)$, we define $y_j \in H^1(\Omega)$ successively for $j \ge 1$ to be the unique minimizer of the functional

$$\Phi_{j,\tau}(\eta) := \int_\Omega \left(\tfrac{1}{2\tau_j}|\eta - y_{j-1}|^2 + A(\nabla\eta) + f(\eta) - u_j\eta \right), \tag{2.17}$$

where the integrands are strongly convex for $\frac{1}{\tau} + f''(s) \ge \frac{1}{\tau} - C_\psi > 0$. For $\eta \in H^1(\Omega)$, $\delta > 0$ we obtain with $\eta_\delta := y_j + \delta(\eta - y_j)$ and using the convexity of A as well as $\Phi_{j,\tau}(y_j) \le \Phi_{j,\tau}(\eta_\delta)$

$$\begin{aligned} &\int_\Omega (A(\nabla\eta) - A(\nabla y_j)) \ge \int_\Omega \frac{1}{\delta}(A(\nabla\eta_\delta) - A(\nabla y_j)) \\ &\ge -\int_\Omega \left(\tfrac{1}{2\tau_j\delta}(|\eta_\delta - y_{j-1}|^2 - |y_j - y_{j-1}|^2) + \tfrac{1}{\delta}(f(\eta_\delta) - f(y_j)) - u_j(\eta - y_j)\right) \\ &\overset{\delta\to 0}{\to} -\int_\Omega \left(\frac{y_j - y_{j-1}}{\tau_j}\right)(\eta - y_j) + f'(y_j)(\eta - y_j) - u_j(\eta - y_j). \end{aligned} \tag{2.18}$$

The limit in the last term is done by a similar argumentation as in Remark 2.2.2 using that $|f'(x)| \le C(1 + |x|)$. Therefore we obtain (2.14).
We now want to show (2.16). The summation of $\Phi_{l,\tau}(y_l) \le \Phi_{l,\tau}(y_{l-1})$ yields

$$\begin{aligned} \sum_{l=1}^{j} \int_\Omega \frac{1}{2\tau_l}|y_l - y_{l-1}|^2 + A(\nabla y_l) + f(y_l) &\le \sum_{l=1}^{j} \int_\Omega A(\nabla y_{l-1}) + f(y_{l-1}) + u_l(y_l - y_{l-1}) \\ &= \sum_{l=0}^{j-1} \int_\Omega A(\nabla y_l) + f(y_l) + \sum_{l=1}^{j} \int_\Omega u_l(y_l - y_{l-1}), \end{aligned}$$

from where we get

$$\underbrace{\sum_{l=1}^{j} \int_\Omega \frac{1}{2\tau_l}|y_l - y_{l-1}|^2}_{=\frac{1}{2}\int_0^{t_j}\int_\Omega |\partial_t^{-\tau} y_\tau|^2} + \int_\Omega A(\nabla y_j) + f(y_j) \le \int_\Omega A(\nabla y_0) + f(y_0) + \underbrace{\sum_{l=1}^{j} \int_\Omega u_l(y_l - y_{l-1})}_{=\int_0^{t_j}\int_\Omega u_\tau \partial_t^{-\tau} y_\tau},$$

and finally using (scaled) Young's inequality 1.2.6

$$\begin{aligned} &\int_0^{t_j}\int_\Omega \tfrac{1}{2}|\partial_t^{-\tau} y_\tau|^2 + \int_\Omega (A(\nabla y_j) + f(y_j)) \\ &\le \int_\Omega (A(\nabla y_0) + f(y_0)) + \int_0^{t_j}\int_\Omega u_\tau \partial_t^{-\tau} y_\tau \le C(\Lambda) + \tfrac{1}{4}\int_0^{t_j}\int_\Omega |\partial_t^{-\tau} y_\tau|^2. \end{aligned} \tag{2.19}$$

From this we obtain

$$\|\partial_t^{-\tau} y_\tau\|_{L^2(0,T;L^2(\Omega))} + \|\nabla y_\tau\|_{L^\infty(0,T;L^2(\Omega))} + \|y_\tau\|_{L^\infty(0,T;L^2(\Omega))} \leq C(\Lambda). \tag{2.20}$$

The boundedness of the first two terms directly follows from (2.19) using the assumptions $-\Lambda + \Lambda^{-1}|p|^2 \leq A(p)$ and $-f \leq \Lambda$. The last relation then can be deduced from $\|y_0\|_{L^2(\Omega)} \leq \Lambda$, $y_j = y_0 + \int_0^{t_j} \partial_t^{-\tau} y_\tau$ as well as the previously obtained bound on $\partial_t^{-\tau} y_\tau$.
It still remains to show the bound on $f'(y_\tau)$ in (2.16). For this, we choose $\eta := y_j - \delta f'(y_j), \delta > 0$ in (2.18) and obtain

$$\begin{aligned}\int_\Omega f'(y_j)^2 &\leq \int_\Omega -\tfrac{y_j - y_{j-1}}{\tau_j} f'(y_j) + u_j f'(y_j) - \frac{1}{\delta}(A(\nabla y_j) - A(\nabla(y_j - \delta f'(y_j))))\\ &= \int_\Omega -\tfrac{y_j - y_{j-1}}{\tau_j} f'(y_j) + u_j f'(y_j) - A'(\xi_\delta(y_j)\nabla y_j)\cdot \nabla y_j f''(y_j).\end{aligned} \tag{2.21}$$

To the third integral we applied the mean value theorem pointwisely almost everywhere, with the intermediate point of 1 and $1 - \delta f''(y)$ denoted by $\xi_\delta(y)$. Note that due to the boundedness of f'' also $\xi_\delta(\cdot)$ is bounded and $\xi_\delta \to 1$ for $\delta \to 0$ pointwise. Now we can use $0 \leq A'(p)^T p \leq \Lambda|p|^2$ and $-f'' \leq \Lambda$ as well as dominated convergence to obtain

$$\begin{aligned}\int_\Omega f'(y_j)^2 &\leq \int_\Omega -\tfrac{y_j - y_{j-1}}{\tau_j} f'(y_j) + u_j f'(y_j) + C(\Lambda)|\nabla y_j|^2\\ &\leq \int_\Omega \left(\tfrac{y_j - y_{j-1}}{\tau_j}\right)^2 + \tfrac{1}{4} f'(y_j)^2 + \tfrac{1}{2} u_j^2 + \tfrac{1}{2} f'(y_j)^2 + C(\Lambda)|\nabla y_j|^2.\end{aligned}$$

The last inequality follows from Young's and scaled Young's inequality. Estimating the appearing terms with (2.20) we get

$$\left(\sum_{j=1}^N \tau_j \|f'(y_j)\|^2\right)^{1/2} \leq C(\Lambda). \tag{2.22}$$

Together with (2.20) this yields (2.16). □

Remark 2.2.2. As mentioned before the proof above, the strong monotonicity of A' and the growth condition $|A'(p)| \leq C|p|$ induce $A'(0) = 0$, $A'(p)^T p \leq c|p|^2$ and $A(0) + \frac{1}{2}C_A|p|^2 \leq A(p)$. Furthermore also $A(p) \leq c(1 + |p|^2)$ holds. Hence $A(\nabla\eta) \in L^1(Q)$ and (using Young's inequality) we have $A'(\nabla\eta)^T \nabla\xi \in L^1(Q)$ for all $\eta, \xi \in L^2(0,T;H^1(\Omega))$. It also induces the pointwise estimate $|A'(\nabla y + s\delta\nabla\xi)^T \nabla\xi| \leq C'(|\nabla y|^2 + |\nabla\xi|^2)$, for $0 \leq s\delta \leq 1$ providing an integrable majorant, which allows to take the limit $\delta \searrow 0$ for the integral below. Hence we obtain

$$\lim_{\delta\searrow 0} \frac{1}{\delta}\int_Q (A(\nabla y + \delta\nabla\xi) - A(\nabla y)) = \lim_{\delta\searrow 0}\int_Q \int_0^1 A'(\nabla y + s\delta\nabla\xi)^T \nabla\xi \,\mathrm{d}s \quad = \int_Q A'(\nabla y)^T \nabla\xi. \tag{2.23}$$

The same holds respectively for integration over Ω. Together with the monotonicity of A', this enables the usual steps of the proof that solving the variational inequality (2.14) is equivalent to solving the variational equality (2.13) with f instead of ψ.

With Lemma 2.2.1 at hand, we are in the position to show the existence of a unique weak solution to the time discretized state equation (2.13).

Theorem 2.2.3. *Let Assumptions 2.1.1 be fulfilled. If $\tau = \max_j \tau_j < 1/C_\psi$ then for every $u_\tau \in U_\tau$ the time discretized state equation (2.13) has a unique solution $y_\tau \in Y_\tau$.*
The solution operator is denoted by $S_\tau : U_\tau \to Y_\tau$.
Furthermore, given a $\bar{c} > 0$ then for all $\|u_\tau\|_{L^2(0,T;L^2(\Omega))} \leq \bar{c}$ it holds independent of τ

$$\|\partial_t^{-\tau} y_\tau\|_{L^2(0,T;L^2(\Omega))} + \|y_\tau\|_{L^\infty(0,T;H^1(\Omega))} + \|\psi'(y_\tau)\|_{L^2(0,T;L^2(\Omega))} \leq C_{A,\psi,y_0}(\bar{c}). \tag{2.24}$$

Proof: We consider the approximation of ψ by f_n according to Assumption 2.1.1. Then, due to $\psi(y_0) \in L^1(\Omega)$ for given $y_0 \in H^1(\Omega)$ and $-c \leq f_n \leq c(\psi+1)$, $-C_\psi \leq f_n''$ one can find Λ large enough depending only on A, ψ and y_0, such that

$$\begin{gathered}\|y_0\|_{L^2(\Omega)}, \int_\Omega (A(\nabla y_0) + f_n(y_0)), -\inf_{t\in\mathbb{R}} f_n(t), -\inf_{t\in\mathbb{R}} f_n''(t) \leq \Lambda, \\ -\Lambda + \Lambda^{-1}|p|^2 \leq A(p) \text{ and } A'(p)^T p \leq \Lambda |p|^2.\end{gathered} \tag{2.25}$$

We denote by $y_{j,n}$ the solutions of (2.14) with $f = f_n$ which exist according to Lemma 2.2.1 and remark that they exist for $\tau < \frac{1}{C_\psi}$, where the integrands of (2.17) are strongly convex due to $\frac{1}{\tau} + f_n''(s) \geq \frac{1}{\tau} - C_\psi > 0$. Also Lemma 2.2.1 provides the estimates (2.16), i.e., for all τ and n it holds

$$\|\partial_t^{-\tau} y_{\tau,n}\|_{L^2(0,T;L^2(\Omega))} + \|y_{\tau,n}\|_{L^\infty(0,T;H^1(\Omega))} + \|f_n'(y_{\tau,n})\|_{L^2(0,T;L^2(\Omega))} \leq C(\Lambda, \bar{c}). \tag{2.26}$$

The second estimate implies $y_{\tau,n} \overset{*}{\rightharpoonup} y_\tau$ in $L^\infty(0,T;H^1(\Omega))$ for a subsequence and therefore also weak convergence in $L^2(0,T;H^1(\Omega))$. Next we choose $p > 2$ such that $H^1(\Omega) \hookrightarrow L^p(\Omega)$. Then it holds $L^\infty(0,T;H^1(\Omega)) \hookrightarrow L^p(0,T;H^1(\Omega)) \hookrightarrow L^p(0,T;L^p(\Omega)) \cong L^p(Q) \hookrightarrow L^2(Q) \cong L^2(0,T;L^2(\Omega))$ where the last imbedding is compact. Therefore we have $y_{\tau,n} \to y_\tau$ in $L^2(0,T;L^2(\Omega))$ and pointwise a.e. in Q. The first and the last estimate provide a further subsequence with $\partial_t^{-\tau} y_{\tau,n} \rightharpoonup \partial_t y_\tau$ and $f_n'(y_{\tau,n}) \rightharpoonup \psi'(y_\tau)$ in $L^2(0,T;L^2(\Omega))$, where the limits are identified due to pointwise a.e. convergence. For the latter, there entered the requirement $f_n' \to \psi'$ in C_{loc} as will briefly be sketched in what follows. First, we choose an arbitrary $x \in \Omega$ with $y_{\tau,n}(x) \to y_\tau(x)$. Further we take $N \in \mathbb{N}$ such that $y_{\tau,n}(x) \in [a,b]$ for all $n > N$ with $a, b \in \mathbb{R}$ and $a < b$. Then it holds

$$|f_n'(y_{\tau,n}(x)) - \psi'(y_\tau(x))| \leq \underbrace{|f_n'(y_{\tau,n}(x)) - \psi'(y_{\tau,n}(x))|}_{\leq \sup_{z\in[a,b]} |f_n'(z)-\psi'(z)| \to 0} + \underbrace{|\psi'(y_{\tau,n}(x)) - \psi'(y_\tau(x))|}_{\to 0, \text{ since } \psi' \text{ is continuous}} \to 0 \qquad \text{for } n \to \infty, \tag{2.27}$$

where in the first term we used $f_n' \to \psi'$ in C_{loc}. This shows the pointwise a.e. convergence of $f_n'(y_{\tau,n})$.
The just stated convergences together with the weak lower semicontinuity of A yield after taking the limit $n \to \infty$ for all terms in (2.26) and in (2.14) the estimate (2.24) and that for all $\eta \in H^1(\Omega)$ it holds

$$\int_\Omega (A(\nabla\eta) - A(\nabla y_j)) \geq \int_\Omega -\left(\frac{y_j - y_{j-1}}{\tau_j}\right)(\eta - y_j) - \psi'(y_j)(\eta - y_j) + u_j(\eta - y_j). \tag{2.28}$$

Finally we can go over to the equality (2.13) by the reasoning from Remark 2.2.2.
The uniqueness of the solution of (2.13) can be shown for each time step separately one after another. For this purpose assume the existence of two solutions $y_\tau^{(1)}$ and $y_\tau^{(2)}$ to the same initial data and right-hand side. Subtracting their defining equations, testing with their difference and using the strong monotonicity of A' and of $s + \tau_j\psi'(s)$ due to $\tau < 1/C_\psi$, we get for each time step

$$\begin{aligned}C\|y_j{}^{(1)} - y_j^{(2)}\|_{H^1(\Omega)}^2 &= C\|\nabla y_j^{(1)} - \nabla y_j^{(2)}\|_{L^2(\Omega)}^2 + C\|y_j^{(1)} - y_j^{(2)}\|_{L^2(\Omega)}^2 \\ &\leq \tau_j(A'(\nabla y_j^{(1)}) - A'(\nabla y_j^{(2)}), \nabla y_j^{(1)} - \nabla y_j^{(2)}) + (y_j^{(1)} + \tau_j\psi'(y_j^{(1)}) - y_j^{(2)} - \tau_j\psi'(y_j^{(2)}), y_j^{(1)} - y_j^{(2)}) \\ &= \tau_j(u_j - u_j, y_j^{(1)} - y_j^{(2)}) + (y_{j-1}^{(1)} - y_{j-1}^{(2)}, y_j^{(1)} - y_j^{(2)}) = 0.\end{aligned}$$

In the last line we used that both solutions were defined with the same right-hand side u_j and that we already know that the previous time step can be solved for uniquely (Note that for $j = 1$ it holds $y_{j-1}^{(1)} = y_{j-1}^{(2)} = y_0$). □
With a further (minor) restriction on the maximal time step τ we obtain Lipschitz continuity with a constant independent from τ.

Theorem 2.2.4. *Let Assumptions 2.1.1 and $\tau \le \frac{1}{1+2C_\psi}$ hold. Then the mapping $\tilde{S}_\tau : (y_0, u_\tau) \mapsto y_\tau$ where y_τ is the solution of equation* (2.13)*, is Lipschitz continuous, i.e., more precisely it holds*

$$\begin{aligned} &\|y_\tau^{(1)} - y_\tau^{(2)}\|_{L^\infty(0,T;L^2(\Omega))} + \|\nabla y_\tau^{(1)} - \nabla y_\tau^{(2)}\|_{L^2(0,T;L^2(\Omega))} \le \\ &\le C_{A,\psi,T} \left(\|y_0^{(1)} - y_0^{(2)}\|_{L^2(\Omega)} + \|u_\tau^{(1)} - u_\tau^{(2)}\|_{L^2(0,T;H^1(\Omega)')} \right), \end{aligned} \tag{2.29}$$

where $y_\tau^{(i)} = \tilde{S}_\tau(y_0^{(i)}, u_\tau^{(i)})$ for $i = 1, 2$.

Proof: We note down the differences by a prescript δ, e.g., $\delta y_\tau := y_\tau^{(1)} - y_\tau^{(2)}$. With $\frac{1}{2}(a^2 - b^2) \le (a-b)a$, testing the defining equalities (2.13) with δy_j and using that A' is strongly monotone as well as (2.2), we obtain

$$\begin{aligned} &\tfrac{1}{2}\left(\|\delta y_j\|^2 - \|\delta y_{j-1}\|^2\right) + \tau_j C_A \|\nabla \delta y_j\|^2 \\ &\quad \le (\delta y_j - \delta y_{j-1}, \delta y_j) + \tau_j \left(A'(\nabla y_j^{(1)}) - A'(\nabla y_j^{(2)}), \nabla \delta y_j \right) \\ &\quad = \tau_j (\delta u_j, \delta y_j) - \tau_j \left(\psi'(y_j^{(1)}) - \psi'(y_j^{(2)}), \delta y_j \right) \\ &\quad \le \tfrac{\tau_j}{2\epsilon} \|\delta u_j\|_{H^{1\prime}}^2 + \tfrac{\tau_j \epsilon}{2} \|\delta y_j\|_{H^1}^2 + \tau_j C_\psi \|\delta y_j\|^2. \end{aligned}$$

In the last step we used scaled Young's inequality (see Theorem 1.2.6) with $0 < \epsilon < \min(1, 2C_A)$. We now sum over $j = 1, \dots, J$ and get

$$\tfrac{1}{2}\|\delta y_J\|^2 + \tilde{C}_A \sum_{j=1}^{J} \tau_j \|\nabla \delta y_j\|^2 \le \tfrac{1}{2}\left(\|\delta y_0\|^2 + \sum_{j=1}^{J} \tfrac{\tau_j}{\epsilon} \|\delta u_j\|_{H^{1\prime}}^2 \right) + \tfrac{1}{2}\tilde{C}_\psi \sum_{j=1}^{J} \tau_j \|\delta y_j\|^2, \tag{2.30}$$

for all $1 \le J \le N_\tau$. Here we defined $\tilde{C}_A := C_A - \frac{\epsilon}{2}$ and $\tilde{C}_\psi := \epsilon + 2C_\psi$. Omitting the gradient term on the left and absorbing the J-th term from the right, we obtain

$$\begin{aligned} \|\delta y_J\|^2 &\le \frac{1}{(1 - \tilde{C}_\psi \tau_J)} \left(\|\delta y_0\|^2 + \sum_{j=1}^{J} \tfrac{\tau_j}{\epsilon} \|\delta u_j\|_{H^{1\prime}}^2 \right) + \frac{\tilde{C}_\psi}{1 - \tilde{C}_\psi \tau_J} \sum_{j=1}^{J-1} \tau_j \|\delta y_j\|^2 \\ &\le C_{\psi,\tau} \left(\|\delta y_0\|^2 + \sum_{j=1}^{N_\tau} \tfrac{\tau_j}{\epsilon} \|\delta u_j\|_{H^{1\prime}}^2 \right) + C_{\psi,\tau} \tilde{C}_\psi \sum_{j=1}^{J-1} \tau_j \|\delta y_j\|^2, \end{aligned}$$

where we had to make use of the requirement $C_{\psi,\tau} := \frac{1}{1 - \tilde{C}_\psi \tau} > 0$. To this we apply the discrete Gronwall Lemma 1.2.8 which yields

$$\begin{aligned} \|\delta y_J\|^2 &\le \left(\|\delta y_0\|^2 + \sum_{j=1}^{N_\tau} \tfrac{\tau_j}{\epsilon} \|\delta u_j\|_{H^{1\prime}}^2 \right) C_{\psi,\tau} \exp\left(C_{\psi,\tau} \tilde{C}_\psi \sum_{j=1}^{J-1} \tau_j \right) \\ &\le \left(\|\delta y_0\|^2 + \sum_{j=1}^{N_\tau} \tfrac{\tau_j}{\epsilon} \|\delta u_j\|_{H^{1\prime}}^2 \right) C_{\psi,\tau} \exp\left(C_{\psi,\tau} \tilde{C}_\psi T \right). \end{aligned} \tag{2.31}$$

Inserting this into (2.30) we finally get for all $J = 1, \dots, N_\tau$

$$\tilde{C}_A \sum_{j=1}^{J} \tau_j \|\nabla \delta y_j\|^2 \le \tfrac{1}{2} \left(\|\delta y_0\|^2 + \sum_{j=1}^{N_\tau} \tfrac{\tau_j}{\epsilon} \|\delta u_j\|_{H^{1\prime}}^2 \right) \left(1 + C_{\psi,\tau} \tilde{C}_\psi T \exp\left(C_{\psi,\tau} \tilde{C}_\psi T \right) \right), \tag{2.32}$$

which together with (2.31) and the boundedness of $C_{\psi,\tau}$ independently of τ yields the inequality (2.29). □

A similar result as that of Theorem 2.2.3 could also be obtained by using results on monotone

operators, see, e.g., [89]. Together with an argument formerly found by Stampacchia one would obtain the regularity $y_j \in L^\infty(\Omega) \cap H^1(\Omega)$ at each step of our time discretization [122, 34]. These results are applicable if τ is sufficiently small such that the term $y_j + \tau_j \psi'(y_j)$ becomes monontonic. However this regularity comes with restriction on the space dimension d.
Our approach also allows taking the limit $\tau \to 0$ providing the existence of a solution in the time continuous case, as is demonstrated in the following theorem.

Theorem 2.2.5.
Let Assumptions 2.1.1 hold. Then for every $u \in L^2(0,T;L^2(\Omega))$ there exists a unique weak solution $y \in L^\infty(0,T;H^1(\Omega)) \cap H^1(0,T;L^2(\Omega))$ to (2.10), i.e., to $y(0) = y_0$ a.e. in Ω and

$$\int_Q \partial_t y\eta + A'(\nabla y)^T \nabla\eta + \psi'(y)\eta = \int_Q u\eta \qquad \forall \eta \in L^2(0,T;H^1(\Omega)).$$

Moreover the solution depends Lipschitz continuously on (y_0, u). More precisely it holds

$$\|y_1 - y_2\|_{C([0,T];L^2(\Omega))\cap L^2(0,T;H^1(\Omega))} \le C_{\psi,A,T}\left(\|y_{1,0} - y_{2,0}\|_{L^2(\Omega)} + \|u_1 - u_2\|_{L^2(0,T;H^1(\Omega)')}\right), \tag{2.33}$$

where y_1, y_2 are the solutions to the data $(y_{1,0}, u_1)$ and $(y_{2,0}, u_2)$ respectively.

Proof: Given $u \in L^2(0,T;L^2(\Omega))$ we choose a sequence of discretizations $u_\tau \in U_\tau$ with $u_\tau \rightharpoonup u$ in $L^2(0,T;L^2(\Omega))$ for $\tau \to 0$. This allows for the choice of a $\bar{c} > 0$ with $\|u_\tau\| \le \bar{c}$. Let y_τ be the solution of (2.13) corresponding to u_τ. Then the estimates (2.24) hold. Hence, for $\tau \to 0$ there exists a (sub-)sequence satisfying

$$\begin{aligned}
y_\tau &\rightharpoonup y && \text{in } L^2(0,T;H^1(\Omega)),\\
y_\tau &\overset{*}{\rightharpoonup} y && \text{in } L^\infty(0,T;H^1(\Omega)),\\
y_\tau &\to y && \text{in } L^2(0,T;L^2(\Omega)),\\
\partial_t^{-\tau} y_\tau &\rightharpoonup \partial_t y && \text{in } L^2(0,T;L^2(\Omega)),\\
\psi'(y_\tau) &\rightharpoonup \psi'(y) && \text{in } L^2(0,T;L^2(\Omega)),
\end{aligned} \tag{2.34}$$

where the weak limits are identified using pointwise almost-everywhere convergence of y_τ and continuity of ψ'. With that we can take the limit in

$$\int_Q \partial_t^{-\tau} y_\tau(\eta - y_\tau) + A(\nabla\eta) - A(\nabla y_\tau) + \psi'(y_\tau)(\eta - y_\tau) - u_\tau(\eta - y_\tau) \ge 0 \quad \forall \eta \in L^2(0,T;H^1(\Omega)) \tag{2.35}$$

which is obtained by summing variational inequality (2.28) to obtain that y satisfies

$$\int_Q \partial_t y(\eta - y) + A(\nabla\eta) - A(\nabla y) + \psi'(y)(\eta - y) - u(\eta - y) \ge 0 \quad \forall \eta \in L^2(0,T;H^1(\Omega)). \tag{2.36}$$

Here also the weak lower semicontinuity of A entered. Remark 2.2.2 yields that y solves further the variational equality (2.10). Furthermore, using weak (-$*$) lower semicontinuity of the norms the solution y satisfies

$$\|\partial_t y\|_{L^2(0,T;L^2(\Omega))} + \|y\|_{L^\infty(0,T;H^1(\Omega))} + \|\psi'(y)\|_{L^2(0,T;L^2(\Omega))} \le C_{A,\psi,y_0}(\bar{c}). \tag{2.37}$$

The uniqueness follows from the Lipschitz continuity shown further below.
Recall that the choice of discretization (given by the choice of the intervals) was arbitrary. Furthermore, on the way to obtain y from $y_{j,n}$ we had to take subsequences twice. Whatever choice was made, we would have got a y satisfying the same variational inequality. However, this variational inequality has a unique solution, so the whole sequence has to converge. Summarized, for all discretization, we get a sequence $y_{j,n}$ that for $n \to \infty$ and then $\tau \to 0$ ($j \to \infty$) results in the same limit y satisfiying the variational inequality.
Finally, we will turn our attention to the Lipschitz continuity. Let $y_1, y_2 \in H^1(0,T;L^2(\Omega)) \cap$

$L^2(0,T;H^1(\Omega))$ be two solutions belonging to $(y_{0,1},u_1)$ and $(y_{0,2},u_2)$ and define $\delta y \coloneqq y_1 - y_2$, $\delta u \coloneqq u_1 - u_2$ as well as $\delta y_0 \coloneqq y_{0,1} - y_{0,2}$. Subtracting the defining equations and testing with δy, we obtain for almost all $t \in [0,T]$

$$\tfrac{1}{2}\frac{d}{dt}\|\delta y\|^2 + (A'(\nabla y_1) - A'(\nabla y_2), \nabla \delta y) = (\delta u, \delta y) - (\psi'(y_1) - \psi'(y_2), \delta y)\,.$$

Using strong monotonicity of A', $(\psi'(y_1) - \psi'(y_2), y_1 - y_2) \geq -C_\psi \|y_1 - y_2\|^2$ as well as Cauchy-Schwarz and scaled Young's inequality 1.2.6 yields

$$\tfrac{1}{2}\frac{d}{dt}\|\delta y\|^2 \leq \tfrac{1}{2}\frac{d}{dt}\|\delta y\|^2 + (C_A - \tfrac{\epsilon}{2})\|\nabla \delta y\|^2 \leq \tfrac{1}{2\epsilon}\|\delta u\|^2_{H^1(\Omega)'} + (C_\psi + \tfrac{\epsilon}{2})\|\delta y\|^2, \tag{2.38}$$

with $0 < \epsilon < 2C_A$. Integrating from 0 to $s \leq T$, we obtain (note that $y_1, y_2 \in C([0,T];L^2(\Omega))$ by imbedding)

$$\|\delta y(s)\|^2 \leq \left(\|\delta y(0)\|^2 + \tfrac{1}{\epsilon}\|\delta u\|^2_{L^2(0,T;H^1(\Omega)')}\right) + (2C_\psi + \epsilon)\int_0^s \|\delta y(t)\|^2 \,\mathrm{d}t, \tag{2.39}$$

and an application of Gronwall's inequality 1.2.7 gives

$$\|\delta y(s)\|^2 \leq \exp\left((2C_\psi + \epsilon)T\right)\left(\|\delta y(0)\|^2 + \tfrac{1}{\epsilon}\|\delta u\|^2_{L^2(0,T;H^1(\Omega)')}\right). \tag{2.40}$$

Hence $\|\delta y\|^2_{L^\infty(0,T;L^2(\Omega))} \leq C_{\psi,A,T}\left(\|\delta y(0)\|^2 + \tfrac{1}{\epsilon}\|\delta u\|^2_{L^2(0,T;H^1(\Omega)')}\right)$ and then (2.38) yields

$$\|\nabla \delta y\|^2_{L^2(0,T;L^2(\Omega))} \leq C_{\psi,A,T}\left(\|\delta y(0)\|^2 + \tfrac{1}{\epsilon}\|\delta u\|^2_{L^2(0,T;H^1(\Omega)')}\right), \tag{2.41}$$

with a generic constant $C_{\psi,A,T}$ depending only on ψ, A and T. Together the estimates imply (2.33). □

Note that (2.33) is the time continuous equivalent to (2.29) and the counterparts to (2.40) and (2.41) are (2.31) and (2.32), respectively. Especially note that the constants equal if one sets $\tau = 0$ there. However, since the convergence of y_τ is not shown in $L^\infty(0,T;L^2(\Omega))$, we cannot built the proof of (2.33) on (2.29).

Before we continue, let us compare the proceeding pursued so far in this section with approaches that can be found in the literature.

Remark 2.2.6. Consulting the literature of optimal control problems with state equations resembling one time step in (2.13), like, e.g., [33, 35, 34, 31], one notices that the authors work with an L^∞-regularity result for the state y. In contrast to this, we use the boundedness of $\|\psi'(y_j)\|_{L^2(\Omega)}$, which is less. However, for this we don't need a restriction on the space dimension d. In the next section, we will derive similar results as given in the above references, but only using said boundedness of $\psi'(y_j)$.

Let us compare the argumentation leading to Theorem 2.2.3 with the one leading to the L^∞-bound, where we exemplary have in mind [34, Theorem 3.1]. In both approaches a cutoff argument is used. As seen before, we truncate the function ψ whereas Casas and Fernández truncate a function which in the present setting would translate to $\psi'(s) + \frac{s}{\tau_j}$. In our case, the cutoff is required to be able to go over to the limit in (2.18), where we have to argue with dominated convergence (cf. Remark 2.2.2). Casas and Fernández directly apply a result from [89], which requires the boundedness assumptions on the nonlinearity. The main difference lies in the way the transition to the former equation with the uncut nonlinearity is done. Our truncated functions f_n approach ψ in C^1_{loc} and satisfy the bound (2.22), which allows taking the limit in the weak formulation. Since the bound holds independently from d, the existence of a solution for arbitrary space dimensions follows. In contrast to that, Casas and Fernández apply Stampacchia's method [122], that requires some conditions on d, but obtain an L^∞-bound in return. The limit is taken by the observation that the bound does not depend on the cutoff and by choosing the cutoff large enough such that the obtained solution does no longer change. Note

also, that for this approach the cutoff does not necessarily need to be done in a way yielding the resulting function differentiable.
Beyond the just presented differences, our approach in addition allows to take the limit $\tau \to 0$, due to the supplementary estimate (2.22). The authors of [53]—which our approach is based on—first take $\tau \to 0$ and then go over to ψ, since they are not interested in a discretization of the state equation. Also Casas et al. have considered continuous in time (i.e. parabolic) problems in [38], where existence of a solution to the state equation is shown by a similar cutoff argument as in the elliptic case. However no relation to a discrete in time equation is included there. Even if we were only interested in the time continuous problem, their results would not be applicable to our setting. The reason is that the requirements on the nonlinearity stay the same as in the elliptic case. Most notably a nonnegative derivative of the nonlinearity is required. In the discrete case we could apply their results by choosing τ_j small enough in $\psi'(s) + \frac{s}{\tau_j}$, which is no longer possible in the parabolic case.

Including results from the preceding proof also the following statement regarding the convergence of the time discretization holds.

Theorem 2.2.7. *Let Assumptions 2.1.1 and $u \in L^2(0,T;L^2(\Omega))$ hold. Then for every sequence of discretizations $u_\tau \in U_\tau$ with $u_\tau \rightharpoonup u$ in $L^2(0,T;L^2(\Omega))$, the corresponding solutions $y_\tau \in Y_\tau$ of the time discretized equation* (2.13) *converge to the solution $y \in H^1(0,T;L^2(\Omega)) \cap L^\infty(0,T;H^1(\Omega))$ of the continuous problem* (2.10) *with u in the spaces specified in (2.34). Furthermore we have as $\tau \to 0$*

$$\max_{t\in[0,T]} \|y_\tau(t) - y(t)\|_{L^2(\Omega)} \to 0. \tag{2.42}$$

Proof: It only remains to show the convergence in (2.42). Using the definitions from (2.11), for given y_τ we define its linear interpolant z_τ, i.e.

$$z_\tau(t)_{|I_j} = y_{j-1} + (t - t_{j-1})\partial_t^{-\tau} y_\tau(t_j). \tag{2.43}$$

Note that from (2.24) we have $\|z_\tau\|_{H^1(0,T;L^2(\Omega))\cap L^\infty(0,T;H^1(\Omega))} \leq C$ independent from τ. By the compact imbedding $L^\infty(0,T;H^1(\Omega)) \cap H^1(0,T;L^2(\Omega)) \hookrightarrow C([0,T];L^2(\Omega))$ (see Aubin–Lions–Simon compactness theorem 1.2.10) we deduce the existence of a z such that (possibly for a subsequence) $z_\tau \to z$ in $C([0,T];L^2(\Omega))$.
In addition, for $t = \beta t_j + (1-\beta)t_{j-1}$ with $\beta \in (0,1]$ we find

$$\begin{aligned}
\|y_\tau(t) - z_\tau(t)\|^2_{L^2(\Omega)} &= \|y_\tau(t_j) - (\beta z_\tau(t_j) + (1-\beta) z_\tau(t_{j-1}))\|^2_{L^2(\Omega)} \\
&= (1-\beta)^2 \|z_\tau(t_j) - z_\tau(t_{j-1})\|^2_{L^2(\Omega)} \\
&= (1-\beta)^2 \left\| \int_{t_{j-1}}^{t_j} \partial_t z_\tau(t)\,\mathrm{d}t \right\|^2_{L^2(\Omega)} \\
&= (1-\beta)^2 \int_\Omega \left(\int_{t_{j-1}}^{t_j} \partial_t^{-\tau} y_\tau \,\mathrm{d}t \right)^2 \mathrm{d}x \\
&\leq (1-\beta)^2 (t_j - t_{j-1}) \int_\Omega \int_{t_{j-1}}^{t_j} \left(\partial_t^{-\tau} y_\tau\right)^2 \mathrm{d}t\,\mathrm{d}x \\
&\leq (1-\beta)^2 \tau \|\partial_t^{-\tau} y_\tau\|^2_{L^2(0,T;L^2(\Omega))} \leq C_{A,\psi,y_0}\tau,
\end{aligned} \tag{2.44}$$

where $\tau := \max_j \tau_j$ as before. We have used that the integrand $\partial_t^{-\tau} y_\tau$ is constant on I_j as well as (2.24). Note that the constant C_{A,ψ,y_0} is independent from t. Consequently, together with the convergence of z_τ stated above, it follows $\max_{t\in[0,T]} \|y_\tau(t) - z(t)\|_{L^2(\Omega)} \to 0$ as $\tau \to 0$. Note that the limit z is uniquely given by y due to the pointwise a.e. convergence of y_τ (cf. (2.34)) and in addition the whole sequence converges. □

Using the fact that the linear interpolant z_τ from the previous proof actually is Hölder continuous, we can deduce the following supplementary theorem concerning the regularity of y. The idea of the proof is taken from [61, Lemma 3.4].

Theorem 2.2.8. *Let the assumptions of the previous theorem hold. Then in addition the solution y of (2.10) lies in $C^{0,\alpha}([0,T];L^2(\Omega))$ where $\alpha \in (0, \frac{1}{4})$.*

Proof: Again we consider the linear interpolant $z_\tau \in H^1(0,T;L^2(\Omega)) \cap L^\infty(0,T;H^1(\Omega))$ from (2.43). Since it holds $\partial_t z_\tau = \partial_t^{-\tau} y_\tau$ almost everywhere in $[0,T]$, we can rewrite (2.13) as

$$(\partial_t z_\tau(t,\cdot),\varphi) + (A'(\nabla y_\tau(t,\cdot)),\nabla\varphi) + (\psi'(y_\tau(t,\cdot)),\varphi) = (u_\tau(t,\cdot),\varphi) \quad \forall \varphi \in H^1(\Omega), \tag{2.45}$$

for almost all t. We now take $t_1, t_2 \in [0,T]$, $t_1 < t_2$, test with $\varphi = z_\tau(t_1,\cdot) - z_\tau(t_2,\cdot)$ and integrate from t_1 to t_2 to obtain

$$\begin{aligned}\int_\Omega |z_\tau(t_1,\cdot) - z_\tau(t_2,\cdot)|^2 &\le \int_{t_1}^{t_2} \left(\|A'(\nabla y_\tau)\|_{L^2(\Omega)} + \|\psi'(y_\tau)\|_{L^2(\Omega)} + \|u_\tau\|_{L^2(\Omega)}\right) \|z_\tau(t_1) - z_\tau(t_2)\|_{H^1(\Omega)} \\ &\le C \int_{t_1}^{t_2} \left(C_A \|\nabla y_\tau\|_{L^2(\Omega)} + \|\psi'(y_\tau)\|_{L^2(\Omega)} + \|u_\tau\|_{L^2(\Omega)}\right) \\ &\le C(t_2 - t_1)^{\frac{1}{2}} \left(\|y_\tau\|_{L^2(0,T;H^1(\Omega))} + \|\psi'(y_\tau)\|_{L^2(0,T;L^2(\Omega))} + \|u_\tau\|_{L^2(0,T;L^2(\Omega))}\right).\end{aligned} \tag{2.46}$$

For the second inequality we used the $L^\infty(0,T;H^1(\Omega))$ regularity of z_τ as well as the growth condition on A' and the last inequality follows from the Cauchy-Schwarz inequality. Together with (2.24) this in total yields

$$\|z_\tau(t_1,\cdot) - z_\tau(t_2,\cdot)\|_{L^2(\Omega)} \le C_{A,\psi,y_0} |t_1 - t_2|^{\frac{1}{4}}. \tag{2.47}$$

Since y_τ are uniformly bounded in $L^\infty(0,T;H^1(\Omega))$ (cf. (2.24)) also the z_τ are and together with the compact imbedding $H^1(\Omega) \hookrightarrow L^2(\Omega)$ and the equicontinuity given by (2.47) we infer from the Arzelà–Ascoli theorem for Banach space valued functions [120] (followed up by a reasoning as in [8, Theorem 10.6] involving (2.47)) that (possibly for a subsequence)

$$z_\tau \to z \quad \text{in } C^{0,\alpha}([0,T];L^2(\Omega)),$$

for some z. In the proof of Theorem 2.2.7 we have seen that the limit is uniquely given by y and so in fact the whole sequence converges to y. □

Since the state equation is given by the gradient flow of the energy $\mathcal{E}$ given in (2.8) (cf. also (1.2) and below), this functional decreases in time when there is no input, i.e., $u = 0$. The discretization shall inherit this property.

Theorem 2.2.9. *Let Assumptions 2.1.1 and $\tau \le 2/C_\psi$ hold. Then the scheme (2.13) for the state equation is energy stable, i.e., for $u_\tau = 0$ the energy functional $\mathcal{E}$ is decreasing in time.*

Proof: We set $u_j = 0$ in (2.13) and test with the difference $y_j - y_{j-1}$ to obtain

$$\frac{1}{\tau_j}\|y_j - y_{j-1}\|^2 + (A'(\nabla y_j), \nabla y_j - \nabla y_{j-1}) + (\psi'(y_j), y_j - y_{j-1}) = 0. \tag{2.48}$$

The convexity of A (recall A' is strongly monotone) yields

$$(A'(\nabla y_j), \nabla y_j - \nabla y_{j-1}) \ge A(\nabla y_j) - A(\nabla y_{j-1})$$

for the second term. The third term can be estimated by the following relation

$$\psi'(y_j)(y_j - y_{j-1}) \ge \psi(y_j) - \psi(y_{j-1}) - \tfrac{C_\psi}{2}(y_j - y_{j-1})^2. \tag{2.49}$$

To see this, we note that this holds for f_n approximating ψ as in Assumptions 2.1.1, i.e., it holds

$$f_n'(y_j)(y_j-y_{j-1}) = f_n(y_j)-f_n(y_{j-1})+\tfrac{1}{2}f_n''(s)(y_j-y_{j-1})^2 \geq f_n(y_j)-f_n(y_{j-1})-\tfrac{C_\psi}{2}(y_j-y_{j-1})^2, \tag{2.50}$$

where s is some appropriate intermediate point from the Taylor expansion. By adding zero and rearranging, from this we obtain

$$\begin{aligned}&\psi'(y_j)(y_j-y_{j-1}) \geq \psi(y_j)-\psi(y_{j-1})-\tfrac{C_\psi}{2}(y_j-y_{j-1})^2+\\ &[(\psi'(y_j)-f_n'(y_j))(y_j-y_{j-1})+(f_n(y_j)-\psi(y_j))-(f_n(y_{j-1})-\psi(y_{j-1}))].\end{aligned}$$

Taking the limit $n\to\infty$ using that $f_n\to\psi$ in C^1_{loc} the inequality (2.49) follows.
Collecting terms and using the definition of the Ginzburg-Landau energy (2.8) one finds

$$\left(\frac{1}{\tau_j}-\frac{C_\psi}{2}\right)\|y_j-y_{j-1}\|^2+[\mathcal{E}(y_j)-\mathcal{E}(y_{j-1})]\leq 0, \tag{2.51}$$

and thus $\mathcal{E}(y_j)\leq\mathcal{E}(y_{j-1})$ if $\tau\leq 2/C_\psi$. □

The result from Theorem 2.2.9 can be applied to the discretizations assumed in Theorem 2.2.3 and Theorem 2.2.4 since it provides a less restricting assumption on the step length τ.
For the numerics it is useful to know the boundedness of the solution at least in the uncontrolled case $u=0$. The following theorem shows that $|y|\leq C$ holds under certain circumstances for the anisotropic Allen-Cahn equation.

Theorem 2.2.10. *Let Assumptions 2.1.1 hold and y be the weak solution of the Allen-Cahn equation* (2.10) *(cf. Theorem 2.2.5) with $u=0$. Further assume that $\psi'(s)=0$ has finitely many real solutions that we denote by c_i. We set $\overline{c}_\psi := \max_i c_i$ and $\underline{c}_\psi := \min_i c_i$ and finally require that $\psi'(s)>0$ for $s>\overline{c}_\psi$ and $\psi'(s)<0$ for $s<\underline{c}_\psi$ (e.g., choose ψ like in Lemma 2.1.4). Then, if $\underline{c}_\psi\leq y_0(x)\leq\overline{c}_\psi$ almost everywhere in Ω, we have $\underline{c}_\psi\leq y(x,t)\leq\overline{c}_\psi$ almost everywhere in Q. That in particular means that $y\in L^\infty(0,T;H^1(\Omega))\cap H^1(0,T;L^2(\Omega))\cap L^\infty(Q)$.*

Proof: In the following we will only show $y(x,t)\leq\overline{c}_\psi$ almost everywhere, since the case $y(x,t)\geq\underline{c}_\psi$ works analogously. To this end, we define the following test function for (2.10)

$$\tilde\eta := \max(0, y-\overline{c}_\psi), \tag{2.52}$$

i.e., it holds $\tilde\eta=0$ where $y\leq\overline{c}_\psi$. Note that $\tilde\eta\in L^2(0,T;H^1(\Omega))\cap H^1(0,T;L^2(\Omega))$ and for the derivatives we obtain $\partial_t\tilde\eta=\chi_{\{y>\overline{c}_\psi\}}\partial_t y$ and $\nabla\tilde\eta=\chi_{\{y>\overline{c}_\psi\}}\nabla y$, where χ denotes the characteristic function. These facts follow, e.g., from [62, Lemma 7.6]. From this we further get almost everywhere the following inequalities

$$\begin{aligned}&\partial_t y\tilde\eta=\partial_t y\chi_{\{y>\overline{c}_\psi\}}y=\partial_t\tilde\eta\tilde\eta=\tfrac{1}{2}\partial_t|\tilde\eta|^2, \quad A'(\nabla y)^T\nabla\tilde\eta=\chi_{\{y>\overline{c}_\psi\}}A'(\nabla y)^T\nabla y\geq 0,\\ &\psi'(y)\tilde\eta=\chi_{\{y>\overline{c}_\psi\}}\psi'(y)(y-\overline{c}_\psi)\geq 0.\end{aligned}$$

Here we used the requirement $\psi'(s)>0$ for $s>\overline{c}_\psi$. With these identities it follows from (2.10) that

$$\frac{1}{2}\int_0^T\frac{d}{dt}\|\tilde\eta\|^2_{L^2(\Omega)}\leq 0,$$

and using the fundamental theorem of calculus and that $y_0\leq\overline{c}_\psi$ almost everywhere, we can conclude $\|\tilde\eta(t)\|_{L^2(\Omega)}\leq 0$ for almost all $t\in[0,T]$, i.e., $\tilde\eta\equiv 0$ almost everywhere in Q. From the definition (2.52) it finally follows that $y(x,t)\leq\overline{c}_\psi$ holds almost everywhere in Q. □

In case of the familiar double-well potential $\psi(s)=\frac{1}{4}(s^2-1)^2$, the previous theorem would yield $|y|\leq 1$ provided that $|y_0|\leq 1$ almost everywhere. Although the previous theorem does not apply to the controlled problem, we observed no strong deviation from this numerically. As can be seen from the plots in Section 4.3, we obtain values of u that reach into the two-digit range, but still the function values of y are very close to one.

2.3 Existence of the optimal control in the time discretized and in the continuous setting

Having shown the existence of solutions to the discretized equation (2.13) and time continuous state equation (2.10), we will further develop our results to show existence of solutions to the pertinent control problems (2.12)–(2.13) and (2.9)–(2.10). In the end we will show that global minimizers of a series of discrete problems converge to a global minimizer of the time continuous problem for $\tau \to 0$.

Theorem 2.3.1. *Let Assumptions 2.1.1 be fulfilled and* $\max_j \tau_j := \tau < \frac{1}{(1+2C_\psi)}$ *hold. Then for every* $y_\Omega \in L^2(\Omega)$ *the control problem* (2.12)–(2.13) *has at least one solution in* $U_\tau \times Y_\tau$.

Proof: The requirements assure that Theorem 2.2.3 is applicable and for every $u_\tau \in U_\tau$ we find a unique solution $S_\tau(u_\tau) = y_\tau \in Y_\tau$ of (2.13). Since the feasible set $\{(u_\tau, y_\tau) \mid y_\tau = S_\tau(u_\tau) \text{ for } u_\tau \in U_\tau\}$ is nonempty and the cost functional in (2.12) is bounded from below, we can deduce the existence of an infimum ι and of a minimizing sequence $((u_\tau^{(m)}, y_\tau^{(m)}))_m$ with $\iota := \lim_{m\to\infty} J(u_\tau^{(m)}, y_\tau^{(m)})$. If $u_\tau^{(m)} \in L^2(0,T;L^2(\Omega))$ was unbounded so would be J, which would contradict its convergence to an infimum. Hence there exists a constant $\bar{c}_\tau > 0$, possibly depending on τ with $\|u_\tau^{(m)}\|_{L^2(0,T;L^2(\Omega))} \le \bar{c}_\tau$ for all m, and we can extract a weakly convergent subsequence denoted in the same way $u_\tau^{(m)} \rightharpoonup u_\tau^*$ in $L^2(0,T;L^2(\Omega))$. From Theorem 2.2.3 we obtain independent from m

$$\|y_\tau^{(m)}\|_{L^\infty(0,T;H^1(\Omega))} + \|\psi'(y_\tau^{(m)})\|_{L^2(0,T;L^2(\Omega))} \le C_{A,\psi,y_0}(\bar{c}_\tau). \tag{2.53}$$

This yields $y_\tau^{(m)} \to {y_\tau}^*$ in $L^2(0,T;L^2(\Omega))$ and almost everywhere, as well as $\psi'(y_\tau^{(m)}) \rightharpoonup \psi'(y_\tau^*)$ in $L^2(0,T;L^2(\Omega))$ possibly for a subsequence. Since U_τ is finite dimensional in time and due to the compact imbedding $L^2(\Omega) \hookrightarrow H^1(\Omega)'$ we obtain $u_\tau^{(m)} \to u_\tau^*$ in $L^2(0,T;H^1(\Omega)')$. So the Lipschitz continuity stated in Theorem 2.2.4 in addition yields $y_\tau^{(m)} \to {y_\tau}^*$ in $L^2(0,T;H^1(\Omega))$. Now we can take the limit in the state equation and obtain

$$(y_j^* - y_{j-1}^*, \varphi) + \tau_j(A'(\nabla y_j^*), \nabla\varphi) + \tau_j(\psi'(y_j^*), \varphi) = \tau_j(u_j^*, \varphi) \qquad j = 1, \ldots, N. \tag{2.54}$$

The convergence of the second term arises from the fact that $A' : L^2(\Omega) \to L^2(\Omega)$ is a Nemytskii operator. From (2.54) we conclude that $y_\tau^* = S_\tau(u_\tau^*)$ and hence (u_τ^*, y_τ^*) is feasible and its optimality follows by using the weak lower semicontinuity of J. □

Similarly we can show the existence of the optimal control in the time continuous setting given the control-to-state operator $S : u \to y$ and the estimates (2.37) for y provided in the proof of Theorem 2.2.5.

Theorem 2.3.2. *If Assumptions 2.1.1 and* $y_\Omega \in L^2(\Omega)$ *hold, then there exists a solution to the optimization problem* (2.9)–(2.10).

Proof: From Theorem 2.2.5, for each $u \in L^2(0,T;L^2(\Omega))$ we get the existence of a unique weak solution $y = S(u)$ in $L^2(0,T;H^1(\Omega)) \cap H^1(0,T;L^2(\Omega))$ to (2.10). Since J is bounded from below, we deduce the existence of an infimum of J over the feasible set of solutions (y,u) of (2.10) and denote the minimizing sequence by $(y_m, u_m)_{m\in\mathbb{N}}$. The sequence $(u_m)_{m\in\mathbb{N}}$ is bounded in $L^2(Q)$ due to $J(y_m, u_m) \ge \frac{\lambda}{2}\|u_m\|^2$ and by reflexivity of $L^2(Q)$ we can extract a weakly convergent subsequence with limit $\bar{u}$. We now choose Λ at least large enough to bound the sequence $(u_m)_{m\in\mathbb{N}}$. Then the estimates (2.25) are fulfilled independently of m. Consequently the estimate (2.37) holds independently of m for the solutions $y_m = S(u_m)$ of (2.10) constructed in the existence proof. Hence, due to the uniqueness of the solutions, we have for y_m

$$\|\partial_t y_m\|_{L^2(0,T;L^2(\Omega))} + \|y_m\|_{L^2(0,T;H^1(\Omega))} + \|\psi'(y_m)\|_{L^2(0,T;L^2(\Omega))} \le \bar{c}. \tag{2.55}$$

From this we get a subsequence (u_m, y_m) with u_m converging weakly to a $\bar{u}$ in $L^2(0,T;L^2(\Omega))$, and y_m converging to a $\bar{y}$ weakly in $L^2(0,T;H^1(\Omega)) \cap H^1(0,T;H^1(\Omega)')$, by the compact imbedding 1.2.10.1 strongly in $L^2(Q)$ and pointwise almost everywhere in Q. Note that $\bar{y}(0) = y_0$ due to $L^2(0,T;H^1(\Omega)) \cap H^1(0,T;H^1(\Omega)') \hookrightarrow C([0,T];L^2(\Omega))$. Moreover, $\partial_t y_m$ and $\psi'(y_m)$ converge weakly to $\partial_t \bar{y}$, respectively to $\psi'(\bar{y})$ in $L^2(0,T;L^2(\Omega))$. That the limit actually equals $\psi'(\bar{y})$ is due to the pointwise a.e. convergence $y_m \to \bar{y}$ (from strong convergence in $L^2(Q)$) and continuity of ψ'.
In order to obtain $\bar{y} = S(\bar{u})$, we need to be able to pass to the limit also in the A'-term of (2.10). Given the fact that $A' : L^2(Q) \to L^2(Q)$ is a Nemytskii operator it is sufficient to show the strong convergence $\nabla y_m \to \nabla \bar{y}$ in $L^2(0,T;L^2(\Omega))$. Then finally, the weak lower semicontinuity of J (deduced from its continuity and convexity as a functional $L^2(0,T;H^1(\Omega)) \cap H^1(0,T;L^2(\Omega)) \times L^2(Q) \to \mathbb{R}$, where in particular the continuous imbedding $L^2(0,T;H^1(\Omega)) \cap H^1(0,T;H^1(\Omega)') \hookrightarrow C([0,T];L^2(\Omega))$ is used) provides $(\bar{y}, \bar{u})$ being a minimizer of J.
The time derivative is monotone if $y_m(0) - \bar{y}(0) = 0$, which follows from

$$\langle \partial_t y_m - \partial_t \bar{y}, y_m - \bar{y} \rangle = \tfrac{1}{2} \left(\|y_m(T) - \bar{y}(T)\|^2 - \|y_m(0) - \bar{y}(0)\|^2 \right) = \tfrac{1}{2} \|y_m(T) - \bar{y}(T)\|^2 \geq 0.$$

Hence we have $\langle \partial_t \bar{y}, y_m - \bar{y} \rangle \leq \langle \partial_t y_m, y_m - \bar{y} \rangle$, and $y_m = S(u_m)$ yields

$$(A'(\nabla y_m), \nabla y_m - \nabla \bar{y}) \leq (u_m, y_m - \bar{y}) - (\psi'(y_m), y_m - \bar{y}) - \langle \partial_t \bar{y}, y_m - \bar{y} \rangle .$$

Recalling the convergence properties of y_m together with $\|u_m\| + \|\psi'(y_m)\| \leq C$, the right-hand side vanishes in the limit $m \to \infty$. From strong monotonicity we obtain

$$\begin{aligned} C\|\nabla y_m - \nabla \bar{y}\|^2 &\leq (A'(\nabla y_m) - A'(\nabla \bar{y}), \nabla y_m - \nabla \bar{y}) \\ &\leq (A'(\nabla y_m), \nabla y_m - \nabla \bar{y}) - (A'(\nabla \bar{y}), \nabla y_m - \nabla \bar{y}), \end{aligned}$$

where the second term on the right hand side vanishes in the limit by weak convergence and we have just shown that the limit of the first one can be bounded by 0 from above. This finally yields the desired strong convergence of ∇y_m in $L^2(Q)$. □
Note that for the convergence $\nabla y_m \to \nabla y$ in $L^2(0,T;L^2(\Omega))$ we could not use (2.33) like we could use (2.29) for Theorem 2.3.1. The reason is that in the time continuous case we do not have the analogon to the compact imbedding $L^2(\Omega)^N \hookrightarrow H^1(\Omega)'^N$ ($L^2(0,T;L^2(\Omega)) \hookrightarrow L^2(0,T;H^1(\Omega)')$ is not compact). Therefore we had to show the convergence more directly.
Finally we obtain a convergence result for the discrete optimal controls like, e.g., in [132] or [74].

Theorem 2.3.3. *Let the previous assumptions on A and ψ hold. Consider a sequence of global optimal controls $(u_\tau, y_\tau)_\tau$ of* (2.12)–(2.13) *belonging to a sequence of discretizations with $\tau \to 0$. Then there exists a subsequence with $u_\tau \to u$ in $L^2(0,T;L^2(\Omega))$ where $(u, y(u))$ solves* (2.9)–(2.10).

Note that one can only expect to deduce the convergence of a subsequence. The whole sequence does not necessarily have to converge, since the global minimizers, whose existence was shown previously, do not have to be unique. Also note that, using the first order conditions, in practice one is usually only able to find local optima. Here one typically has to modify the cost functional to obtain convergence, unless one were to consider only strict local minimizers. The consideration of local minimizers also requires a generalization of Theorems 2.3.1 and 2.3.2 for problems where u or u_τ lie in some (possibly bounded) admissible set U_{ad} instead of $L^2(Q)$ as the idea is to transform the optimization problem to a new one that is minimized only over some ball containing the local optimizer. For further information on this we refer the interested reader to [132] where this analysis is done for another problem.
Proof of Theorem 2.3.3: First we choose an arbitrary $u^* \in L^2(0,T;L^2(\Omega))$ and a sequence $u^*_\tau \in U_\tau$ with $u^*_\tau \to u^*$ in $L^2(0,T;L^2(\Omega))$. Hence $y^*_\tau = S_\tau(u^*_\tau)$ is bounded in $L^\infty(0,T;H^1(\Omega))$ due to (2.24). Now let $(u_\tau)_\tau$ be the sequence of global minimizers to (2.12) subject to (2.13). It holds

$$J(y_\tau, u_\tau) \leq J(y^*_\tau, u^*_\tau) = \frac{1}{2}\|y^*_\tau(T) - y_\Omega\|^2 + \frac{\lambda}{2}\|u^*_\tau\|^2 \leq c,$$

where the first inequality follows from global optimality and in the last inequality we use the just shown boundedness and that u_τ^* is bounded due to convergence. This implies that $(u_\tau)_\tau$ is bounded in $L^2(0,T;L^2(\Omega))$ and we deduce a subsequence with $u_\tau \rightharpoonup u$ in $L^2(0,T;L^2(\Omega))$. Then Theorem 2.2.7 yields that for $y_\tau = S_\tau(u_\tau)$ and $y = S(u)$ we have the strong convergence $y_\tau(T,\cdot) \to y(T,\cdot)$ in $L^2(\Omega)$. Respectively, given some arbitrary sequence $\tilde{u}_\tau$ with $\tilde{u}_\tau \to \tilde{u}$ in $L^2(0,T;L^2(\Omega))$, we obtain the latter also for $\tilde{y}_\tau$ and $\tilde{y}$. This yields

$$J(y,u) \le \liminf_{\tau\to 0} J(y_\tau,u_\tau) \le \liminf_{\tau\to 0} J(\tilde{y}_\tau,\tilde{u}_\tau) = J(\tilde{y},\tilde{u}), \tag{2.56}$$

where in the first inequality we used the weak lower semicontinuity of the norm and strong convergence of $y_\tau(T)$, then the global optimality of (y_τ,u_τ) and finally the strong convergences of $\tilde{y}_\tau(T)$ in $L^2(\Omega)$ and $\tilde{u}_\tau$ in $L^2(0,T;L^2(\Omega))$, respectively. Since $\tilde{u}$ was arbitrary this yields the global optimality of u. Plugging in $\tilde{u} = u$ yields the convergence $\|u_\tau\| \to \|u\|$ and therefore with the weak convergence also the strong convergence $u_\tau \to u$ in $L^2(0,T;L^2(\Omega))$. □

Remark 2.3.4. If instead of the cost functionals (2.9) and (2.12) one considers the cost functionals with a target function $y_Q \in L^2(0,T;L^2(\Omega))$ given over the whole time horizon

$$J^Q(y,u) := \frac{1}{2}\|y-y_Q\|^2_{L^2(Q)} + \frac{\lambda}{2}\|u\|^2_{L^2(Q)}, \tag{2.57}$$

and its discrete counterpart

$$J^Q_\tau(y_\tau,u_\tau) := \frac{1}{2}\sum_{j=1}^N \tau_j\|y_j - y_{Q,j}\|^2 + \frac{\lambda}{2}\sum_{j=1}^N \tau_j\|u_j\|^2, \tag{2.58}$$

with $y_{Q,\tau} \in Y_\tau$ and $y_{Q,\tau} \to y_Q$ in $L^2(0,T;L^2(\Omega))$, the theorems of this section still hold true with proofs following the same lines.

2.4 Fréchet differentiability of the reduced cost functional for the time discretized problem

In this subsection we investigate the Fréchet differentiability of the cost functional $j_\tau(u_\tau) := J(y_\tau(u_\tau),u_\tau)$ for the time discretized optimal control problem reduced to the control u_τ under slightly stricter assumptions than given in 2.1.1. As the ultimate goal of this subsection is to deduce the first order optimality conditions that are also used for the numerical procedure, in contrast to the preceding two subsections we again work with the interface thickness ε. For this purpose, let us again state the time discretized control problem (2.12)–(2.13), which in this convention is given by

$$\min_{Y_\tau\times U_\tau} J(y_\tau,u_\tau) = \frac{1}{2}\|y_N - y_\Omega\|^2 + \frac{\lambda}{2\varepsilon}\sum_{j=1}^N \tau_j\|u_j\|^2, \tag{2.59}$$

subject to

$$\begin{aligned} \tfrac{\varepsilon}{\tau_j}(y_j,\varphi) + \varepsilon(A'(\nabla y_j),\nabla\varphi) + \tfrac{1}{\varepsilon}(\psi'(y_j),\varphi) = (u_j,\varphi) + \tfrac{\varepsilon}{\tau_j}(y_{j-1},\varphi) \quad &\forall \varphi \in H^1(\Omega),\\ &j = 1,\dots,N, \end{aligned} \tag{2.60}$$

and $y_0 \in H^1(\Omega)$ is given as initial condition like before. The spaces Y_τ and U_τ are defined as in (2.11).
In contrast to the preceding sections, the present discussion will need some stricter requirements that we shall summarize here. Note that the first part is a repetition of Assumptions 2.1.1 and contains the restriction on the time step size elaborated before (now including ε).

Assumptions 2.4.1.

a. Assume $A \in C^1(\mathbb{R}^d)$ with A' being strongly monotone and fulfilling the growth condition $|A'(p)| \leq C|p|$.
   Let $\psi \in C^1(\mathbb{R})$ be bounded from below and such that it can be approximated by f_n satisfying
   $$f_n \in C^2(\mathbb{R}), \quad f_n \to \psi \quad \text{in } C^1_{\text{loc}}, \quad -c \leq f_n \leq c(\psi + 1), \quad f_n'' \geq -C_\psi, \quad |f_n''| \leq C_n,$$
   with $c, C_\psi \geq 0$, $C_n > 0$ and $\psi(y_0) \in L^1(\Omega)$ for the given initial data $y_0 \in H^1(\Omega)$.
   In addition, for the time discretization the restriction $\max_j \tau_j < \varepsilon^2/C_\psi$ on the time steps $\tau_j := t_j - t_{j-1}$ holds.

b. Assume further that $A \in C^2(\mathbb{R}^d)$ with bounded A'' and let $\psi \in C^2(\mathbb{R})$, where the Nemytskii operator given by ψ'' is continuous from $H^1(\Omega)$ to $L^q(\Omega)$ for some $q > \max\{d/2, 1\}$.

Let us mention that one can find $p > 2$ with $H^1(\Omega) \hookrightarrow L^p(\Omega)$ and $\frac{1}{q} + \frac{2}{p} < 1$, e.g., when $d \in \{1, 2\}$ choose some $p \in (\frac{2q}{q-1}, \infty)$ and for $d \geq 3$ choose $p = \frac{2d}{d-2}$. In this section we assume p to be always chosen like this. Note that the assumptions imply that A'' is uniformly positive definite and $\psi'' \geq -C_\psi$ holds. Furthermore, the double-well potential (2.4) fulfills the condition if $d \leq 3$ since ψ'' induces a continuous Nemytskii operator from $L^{2q}(\Omega)$ to $L^q(\Omega)$ (see, e.g., [140, Proposition 26.6]) and the imbedding $H^1(\Omega) \hookrightarrow L^{2q}(\Omega)$ is only valid for $d \leq 3$.
As a first step the Fréchet differentiability of the discrete control-to-state operator $S_\tau : U_\tau \to Y_\tau$ of (2.60) is shown. Here, the idea is to prove it for a single time step and then to apply the chain rule Theorem 1.2.13.1. That is, we need to show Fréchet differentiability for each time step. This is in contrast to the case of a single elliptic equation (like, e.g., in [33, 35]), where Gâteaux differentiability would be enough to obtain the first order conditions due to Theorem 1.2.13.2 as long as the cost functional was Fréchet differentiable.
Let us recall, that the solution operator $S_\tau : U_\tau \to Y_\tau$ of (2.60) is given by mapping u_τ, correspondingly $(u_1, \ldots, u_N)$, to $y_\tau = S_\tau(u_\tau)$ determined by $(y_1, \ldots, y_N)$ with
$$y_j = \mathcal{S}(\tfrac{1}{\varepsilon} u_j + \tfrac{1}{\tau_j} y_{j-1}) \qquad \forall j = 1, \ldots N. \tag{2.61}$$
Here $\mathcal{S} : L^2(\Omega) \to H^1(\Omega)$, $g \mapsto \mathrm{y}$ is defined as the solution operator of the quasilinear elliptic problem
$$(A'(\nabla \mathrm{y}), \nabla \varphi) + (\zeta(\mathrm{y}), \varphi) = (g, \varphi) \qquad \forall \varphi \in H^1(\Omega), \tag{2.62}$$
with
$$\zeta(s) := \tfrac{1}{\varepsilon^2} \psi'(s) + \tfrac{1}{\tau_j} s. \tag{2.63}$$
Note that under Assumptions 2.4.1.a the left-hand side defines a strongly monotone operator. In Theorem 2.2.3 we have shown the unique existence of the solution. Let us recall from Remark 2.2.6 that with the restriction on the space dimension $d \leq 3$, a result from [34] for quasilinear elliptic equations with controls on the Neumann boundary provides solutions in $L^\infty(\Omega)$ if in addition $A'(0) = 0$. Here the restriction on d is due to the use of Stampacchia's method. We do not have such a constraint, but note from Assumptions 2.4.1.b that the choice of ψ becomes more limited if d gets bigger.
The next auxiliary lemma will be used several times in what follows. It is the analogue of Theorem 2.2.4 for $\mathcal{S}$ and in fact can be deduced from it (one can replace $L^2(0, T; \cdot)$ with $L^\infty(0, T; \cdot)$ there since the functions are defined to be piecewisely constant). However in this simplified setting there exists a more basic proof that we do not want to withhold.

Lemma 2.4.2. *Let Assumptions 2.4.1.a hold. Then the solution operator $\mathcal{S}$ for (2.62) is Lipschitz continuous with a constant independent of τ for small enough τ, i.e., to be more precise, for $g, \tilde{g} \in L^2(\Omega)$ and $\mathrm{y} = \mathcal{S}(g)$, $\tilde{\mathrm{y}} = \mathcal{S}(\tilde{g})$ it holds*
$$\|\mathrm{y} - \tilde{\mathrm{y}}\|_{H^1(\Omega)} \leq C\|g - \tilde{g}\|_{H^1(\Omega)'} \leq C\|g - \tilde{g}\|_{L^2(\Omega)}. \tag{2.64}$$

Proof: To make the constant independent of τ, we fix some $0 < \tilde{\tau} < \frac{\varepsilon^2}{C_\psi}$ and use that for all $\tau \le \tilde{\tau}$ it holds $\zeta'(s) \ge \frac{-C_\psi}{\varepsilon^2} + \frac{1}{\tau} \ge \frac{-C_\psi}{\varepsilon^2} + \frac{1}{\tilde{\tau}} := \tilde{C}$. Then, subtracting the defining equations, testing with $\mathrm{y} - \tilde{\mathrm{y}}$ and using strong monotonicity of A' and ζ, we obtain

$$\begin{aligned}\|\mathrm{y} - \tilde{\mathrm{y}}\|_{H^1(\Omega)}^2 &= \|\nabla \mathrm{y} - \nabla\tilde{\mathrm{y}}\|_{L^2(\Omega)}^2 + \|\mathrm{y} - \tilde{\mathrm{y}}\|_{L^2(\Omega)}^2 \\ &\le C(A'(\nabla \mathrm{y}) - A'(\nabla\tilde{\mathrm{y}}), \nabla \mathrm{y} - \nabla\tilde{\mathrm{y}}) + \tfrac{1}{\tilde{C}}(\zeta(\mathrm{y}) - \zeta(\tilde{\mathrm{y}}), \mathrm{y} - \tilde{\mathrm{y}}) \\ &= C\,\langle g - \tilde{g}, \mathrm{y} - \tilde{\mathrm{y}}\rangle_{H^{1\prime},H^1} \le C\|g - \tilde{g}\|_{H^1(\Omega)'}\|\mathrm{y} - \tilde{\mathrm{y}}\|_{H^1(\Omega)},\end{aligned}$$

and therefore the assertion after dividing by $\|\mathrm{y} - \tilde{\mathrm{y}}\|_{H^1(\Omega)}$. □

Now we turn our attention to the Fréchet-differentiability of the discrete control-to-state operator. Due to difficulties related to a required norm-gap for the differentiability of the A'-term, the implicit function theorem is not applicable directly (cf. [130]). Also the standard approach like, e.g., in [127, Theorem 4.17] cannot be applied without further ado due to this norm gap and the so far shown insufficient regularity properties of ∇y_j for that purpose. Instead of this, we follow the approaches in [35, 33, 34] and first show Gâteaux differentiability. Afterwards we will add some further effort in order to upgrade to Fréchet differentiability.

Theorem 2.4.3. *Let Assumptions 2.4.1 hold. Then the solution operator $\mathcal{S} : L^2(\Omega) \to H^1(\Omega)$ of (2.62) is Gâteaux differentiable and the directional derivative $\mathcal{S}'(g)v = z$ is given with $\mathrm{y} = \mathcal{S}(g)$ by $z \in H^1(\Omega)$ such that*

$$(A''(\nabla \mathrm{y})\nabla z, \nabla\varphi) + (\zeta'(\mathrm{y})z, \varphi) = (v, \varphi) \qquad \forall \varphi \in H^1(\Omega). \tag{2.65}$$

Furthermore there exists a C independent of $g \in L^2(\Omega)$ and τ with

$$\|z\|_{H^1(\Omega)} \le C\|v\|_{L^2(\Omega)}. \tag{2.66}$$

Proof: Due to the Assumptions 2.4.1 we have that A'' is uniformly positive definite and together with

$$\zeta'(s) = \tfrac{1}{\varepsilon^2}\psi''(s) + \tfrac{1}{\tau_j} \ge \tfrac{-C_\psi}{\varepsilon^2} + \tfrac{1}{\tau} > 0,$$

the bilinear form defined by the left-hand side of (2.65) is elliptic with an ellipticity constant independent of $\mathrm{y} = \mathcal{S}(g)$. Furthermore it is continuous given that

$$|(\zeta'(\mathrm{y})z, \varphi)| \le C\|\zeta'(\mathrm{y})\|_{L^q(\Omega)}\|z\|_{L^p(\Omega)}\|\varphi\|_{L^p(\Omega)} \le C\|\zeta'(\mathrm{y})\|_{L^q(\Omega)}\|z\|_{H^1(\Omega)}\|\varphi\|_{H^1(\Omega)},$$

and A'' is bounded. Hence the Lax-Milgram theorem provides existence and uniqueness of the solution of (2.65) for $v \in L^2(\Omega)$ and—using $\zeta'(s) \ge \tilde{C} > 0$ for small enough τ as in the previous proof—the estimate (2.66) holds for the solutions independently of g and τ. For $v \in L^2(\Omega)$ and $\rho > 0$ let us consider

$$(A'(\nabla \mathrm{y}_\rho), \nabla\varphi) + (\zeta(\mathrm{y}_\rho), \varphi) = (g + \rho v, \varphi)\ . \tag{2.67}$$

Subtracting the equation with $\rho = 0$ and dividing by ρ, we obtain

$$\left(\frac{A'(\nabla \mathrm{y}_\rho) - A'(\nabla \mathrm{y})}{\rho}, \nabla\varphi\right) + \left(\frac{\zeta(\mathrm{y}_\rho) - \zeta(\mathrm{y})}{\rho}, \varphi\right) = (v, \varphi) \qquad \forall \varphi \in H^1(\Omega). \tag{2.68}$$

Lemma 2.4.2 yields for $z_\rho := \frac{1}{\rho}(\mathrm{y}_\rho - \mathrm{y})$

$$\|z_\rho\|_{H^1(\Omega)} \le C\|v\|_{L^2(\Omega)} \qquad \forall \rho. \tag{2.69}$$

Therefore there exists a subsequence with $z_{\rho_n} \rightharpoonup z$ in $H^1(\Omega)$. We now show that this solves (2.65) by taking the limit in (2.68). This implies that z is in fact the desired Gâteaux derivative.

For the first term we have

$$\int_\Omega \tfrac{1}{\rho_n}\left(A'(\nabla \mathtt{y}_{\rho_n}) - A'(\nabla \mathtt{y})\right)\nabla\varphi \,\mathrm{d}x = \int_\Omega \nabla z_{\rho_n}^T A''(w_{\rho_n})\nabla\varphi\,\mathrm{d}x \overset{n\to\infty}{\longrightarrow} \int_\Omega \nabla z^T A''(\nabla \mathtt{y})\nabla\varphi\,\mathrm{d}x, \tag{2.70}$$

where $w_{\rho_n}(x) = \nabla \mathtt{y}(x) + s(x)(\nabla \mathtt{y}_{\rho_n}(x) - \nabla \mathtt{y}(x)))$ with $s(x) \in [0,1]$ is some intermediate point. Since $y_\rho \to y$ in $H^1(\Omega)$ as $\rho \to 0$ (due to Lemma 2.4.2) it holds $w_{\rho_n} \to \nabla \mathtt{y}$ in $L^2(\Omega)^d$. The convergence follows since ∇z_{ρ_n} converges weakly, and $A''(\cdot)\nabla\varphi$ is a continuous Nemytskii operator from $(L^2(\Omega))^d$ to $(L^2(\Omega))^d$ given $A'' : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is continuous and bounded. Therefore it holds $A''(w_{\rho_n})\nabla\varphi \to A''(\nabla \mathtt{y})\nabla\varphi$ in $L^2(\Omega)^d$. Analogously it holds for the second term

$$\int_\Omega \tfrac{1}{\rho_n}\left(\zeta(\mathtt{y}_{\rho_n}) - \zeta(\mathtt{y})\right)\varphi\,\mathrm{d}x = \int_\Omega z_{\rho_n}\zeta'(s_{\rho_n})\varphi\,\mathrm{d}x \overset{n\to\infty}{\longrightarrow} \int_\Omega z\zeta'(\mathtt{y})\varphi\,\mathrm{d}x, \tag{2.71}$$

with intermediate values s_{ρ_n} between $\mathtt{y}_{\rho_n}$ and $\mathtt{y}$ using that $\zeta' : H^1(\Omega) \to L^q(\Omega)$ is a continuous operator.
Hence z fulfills (2.65). Since the limit is given uniquely by the latter equation, the whole sequence $(z_\rho)_{\rho\geq 0}$ converges weakly to z in $H^1(\Omega)$.
It remains to show the strong convergence in $H^1(\Omega)$. Due to the compact imbedding into $L^2(\Omega)$ only the part $\nabla z_\rho \to \nabla z$ in $L^2(\Omega)^d$ is left. For this we consider the sequence $\{L_\rho^T \nabla z_\rho\}_{\rho>0}$ where L_ρ is the Cholesky-decomposition of the s.p.d.-matrix $A''(w_\rho)$. From boundedness and uniformly positive definiteness of A'', we obtain $c \leq \|L_\rho(x)\| \leq C$ with constants independent from ρ and x. Since $A''(w_\rho) \to A''(\nabla \mathtt{y})$ in $L^2(\Omega)^{d\times d}$, from the resulting almost everywhere convergence and just stated boundedness one can verify by dominated convergence that

$$L_\rho \to L \text{ and } L_\rho^{-1} \to L^{-1} \quad \text{in } L^2(\Omega)^{d\times d},$$

where L is the Cholesky-decomposition of $A''(\nabla \mathtt{y})$. Furthermore we have

$$\begin{aligned}\|L_\rho^T \nabla z_\rho\|^2_{L^2(\Omega)} &= \int_\Omega \nabla z_\rho^T A''(w_\rho)\nabla z_\rho\,\mathrm{d}x \leq \int_\Omega \nabla z_\rho^T A''(w_\rho)\nabla z_\rho\,\mathrm{d}x + (\underbrace{\zeta'(s_\rho)}_{>0} z_\rho, z_\rho)\\ &= (v, z_\rho)_{L^2(\Omega)} \leq C\|v\|^2_{L^2(\Omega)},\end{aligned}$$

using (2.68) in the intermediate value formulation and (2.69). So we can extract from the sequence $\{L_\rho^T\nabla z_\rho\}_{\rho>0}$ a weakly convergent subsequence in $L^2(\Omega)^d$. Since we have the strong convergence of L_ρ and the weak convergence of ∇z_ρ we know that $L_\rho^T\nabla z_\rho \rightharpoonup L^T\nabla z$ in $L^1(\Omega)^d$ (see Theorem 1.2.3.2) and since the limits in $L^1(\Omega)^d$ and $L^2(\Omega)^d$ have to coincide we conclude that the previous weak limit actually is $L^T\nabla z$. Due to the uniqueness of the limit also the whole sequence converges weakly in $L^2(\Omega)^d$. Furthermore, there exists a $p' < p$ with $\frac{1}{q} + \frac{2}{p'} \leq 1$. Hence the compact imbedding $H^1(\Omega) \hookrightarrow L^{p'}(\Omega)$ provides $z_\rho \to z$ in $L^{p'}(\Omega)$. Then, using $s_\rho \to \mathtt{y}$ in $H^1(\Omega)$ and given $\zeta' : H^1(\Omega) \to L^q(\Omega)$ is a continuous operator, we have

$$\begin{aligned}\|L^T\nabla z\|^2_{L^2(\Omega)} &\leq \liminf_{\rho\to 0} \|L_\rho^T \nabla z_\rho\|^2_{L^2(\Omega)} = \lim_{\rho\to 0}\,[(v, z_\rho) - (\zeta'(s_\rho)z_\rho, z_\rho)]\\ &= (v,z) - (\zeta'(\mathtt{y})z, z) = \|L^T\nabla z\|^2_{L^2(\Omega)},\end{aligned}$$

and with that we can even deduce $L_\rho^T\nabla z_\rho \to L^T\nabla z$ in $L^2(\Omega)^d$. Furthermore there exists some dominating function $m \in L^2(\Omega)$ with $|L_\rho^T\nabla z_\rho| \leq m$. Finally, from the pointwise relations

$$\nabla z_\rho = (L_\rho^{-T})(L_\rho^T\nabla z_\rho) \to (L^{-T})(L^T\nabla z) = \nabla z \quad \text{and} \quad |\nabla z_\rho| = |L_\rho^{-T}L_\rho^T\nabla z_\rho| \leq C|L_\rho^T\nabla z_\rho| \leq Cm,$$

we get by dominated convergence that $\nabla z_\rho \to \nabla z$ in $L^2(\Omega)^d$.
Ultimately, the desired continuity of $\mathcal{S}'(g) : L^2(\Omega) \to H^1(\Omega)$ needed for the Gâteaux differentiability is given by (2.66). ☐

The following theorem enhances the last result to Fréchet differentiability.

Theorem 2.4.4. *Let Assumptions 2.4.1 hold. Then the mapping $\mathcal{S} : L^2(\Omega) \to H^1(\Omega)$ is Fréchet differentiable.*

Proof: Due to Theorem 1.2.12 it remains to show that the mapping $g \mapsto \mathcal{S}'(g) \in \mathcal{L}(L^2(\Omega), H^1(\Omega))$ is continuous, i.e., that for $g_n \to g$ in $L^2(\Omega)$ it follows that

$$\sup_{v \in L^2(\Omega)} \frac{\|[\mathcal{S}'(g_n) - \mathcal{S}'(g)]v\|_{H^1(\Omega)}}{\|v\|_{L^2(\Omega)}} \to 0 \qquad \text{for } n \to \infty. \tag{2.72}$$

For $g_n \to g$ in $L^2(\Omega)$, Lemma 2.4.2 provides $\mathrm{y}_n := \mathcal{S}(g_n) \to \mathrm{y} := \mathcal{S}(g)$ in $H^1(\Omega)$ and given $v \in L^2(\Omega)$ we set $z_n := \mathcal{S}'(g_n)v$, $z := \mathcal{S}'(g)v$. Subtracting the defining equations for z_n and z (see (2.65)), testing with $(z_n - z) \in H^1(\Omega)$ and inserting 0 terms yields

$$\begin{aligned}
0 &= \int_\Omega \left(A''(\nabla y_n)\nabla z_n - A''(\nabla y)\nabla z\right)^T (\nabla z_n - \nabla z)\,\mathrm{d}x + (\zeta'(y_n)z_n - \zeta'(y)z, z_n - z) \\
&= \int_\Omega \left(A''(\nabla \mathrm{y}_n)\nabla z_n - A''(\nabla \mathrm{y}_n)\nabla z\right)^T (\nabla z_n - \nabla z)\,\mathrm{d}x \\
&\quad + \int_\Omega \nabla z^T \left(A''(\nabla \mathrm{y}_n) - A''(\nabla \mathrm{y})\right)(\nabla z_n - \nabla z)\,\mathrm{d}x \\
&\quad + (\zeta'(\mathrm{y}_n)z_n - \zeta'(\mathrm{y}_n)z, z_n - z) \\
&\quad + (\zeta'(\mathrm{y}_n)z - \zeta'(\mathrm{y})z, z_n - z) \\
&\geq C\|\nabla z_n - \nabla z\|^2_{L^2(\Omega)} \\
&\quad - \|A''(\nabla \mathrm{y}_n) - A''(\nabla \mathrm{y})\|\|\nabla z\|_{L^2(\Omega)}\|\nabla z_n - \nabla z\|_{L^2(\Omega)} \\
&\quad + C\|z_n - z\|^2_{L^2(\Omega)} \\
&\quad - C\|\zeta'(\mathrm{y}_n) - \zeta'(\mathrm{y})\|_{L^q(\Omega)}\|z\|_{L^p(\Omega)}\|z_n - z\|_{L^p(\Omega)},
\end{aligned}$$

where we used the ellipticity of $A''(s)$ and $\zeta'(s)$ with constants independent of s. Given the estimate (2.66) it holds $\|z\|_{H^1(\Omega)}, \|z_n - z\|_{H^1(\Omega)} \leq c\|v\|_{L^2(\Omega)}$ and by rearranging the previous equation it follows

$$\|A''(\nabla \mathrm{y}_n) - A''(\nabla \mathrm{y})\| + \|\zeta'(\mathrm{y}_n) - \zeta'(\mathrm{y})\|_{L^q(\Omega)} \geq C\frac{\|z_n - z\|^2_{H^1(\Omega)}}{\|v\|^2_{L^2(\Omega)}} \quad \forall v \in L^2(\Omega).$$

Since $A''(.)$ is a continuous operator from $(L^2(\Omega))^d \to (L^2(\Omega))^{d\times d}$ and $\zeta'(.)$ is a continuous operator from $L^p(\Omega) \to L^q(\Omega)$, $y_n \to y$ in $H^1(\Omega)$ provides that the left-hand side goes to 0 as $n \to \infty$. Since this convergence holds independent from v with (2.72) it follows that $\mathcal{S}$ is Fréchet differentiable. □

With the just shown results for the model problem (2.62), we are now in position to consider the solution operator $S_\tau : U_\tau \to Y_\tau$ for the whole time discretization (2.60). On each time interval I_j we have $y_\tau(u_\tau)|_{I_j} = y_j(u_1, \dots u_j) = \mathcal{S}(\frac{1}{\varepsilon}u_j + \frac{1}{\tau_j}y_{j-1}(u_1, \dots u_{j-1}))$. Using the previous shown theorem for $\mathcal{S}$ as well as the chain rule we obtain for $j = 1, \dots, N$

$$\begin{aligned}
z_j &:= \frac{dy_\tau(u_\tau)|_{I_j}}{du_\tau}v_\tau = \mathcal{S}'(\tfrac{1}{\varepsilon}u_j + \tfrac{1}{\tau_j}y_{j-1})(\tfrac{1}{\varepsilon}v_j + \tfrac{1}{\tau_j}\frac{dy_\tau(u_\tau)|_{I_{j-1}}}{du_\tau}v_\tau) \\
&= \mathcal{S}'(\tfrac{1}{\varepsilon}u_j + \tfrac{1}{\tau_j}y_{j-1})(\tfrac{1}{\varepsilon}v_j + \tfrac{1}{\tau_j}z_{j-1}),
\end{aligned} \tag{2.73}$$

where we used $z_0 := 0$ and by induction we can state the following theorem.

Theorem 2.4.5. *Let Assumptions 2.4.1 hold. Then the operator $S_\tau : U_\tau \to Y_\tau$ is Fréchet differentiable and consequently also the reduced cost functional*

$$j_\tau : U_\tau \to \mathbb{R} \quad \textit{with} \quad j_\tau(u_\tau) := J(S_\tau(u_\tau), u_\tau)$$

is Fréchet differentiable with

$$j'_\tau(u_\tau)v_\tau = (y_N - y_\Omega, z_N) + \tfrac{\lambda}{\varepsilon}(u_\tau, v_\tau),$$

where z_N *is given by the solution of the following sequence with* $z_0 := 0$

$$(\varepsilon A''(\nabla y_j)\nabla z_j, \nabla\varphi) + (\tfrac{1}{\varepsilon}\psi''(y_j)z_j + \tfrac{\varepsilon}{\tau_j}z_j, \varphi) = (v_j + \tfrac{\varepsilon}{\tau_j}z_{j-1}, \varphi) \qquad \forall j = 1, \dots N, \;\; \varphi \in H^1(\Omega). \tag{2.74}$$

We note that the z_j satisfy a linearized state equation given by $z_\tau = S'_\tau(y_\tau)v_\tau \in Y_\tau$. Furthermore, (2.74) is the dG(0) discretization (as used for the state equation) of the in time continuous, linearized state equation

$$\begin{aligned} \varepsilon(\partial_t z, \eta) + \varepsilon(A''(\nabla y)\nabla z, \nabla\eta) + \frac{1}{\varepsilon}(\psi''(y)z, \eta) &= (v, \eta) \qquad && \forall \eta \in L^2(0,T;H^1(\Omega)), \\ z(0) &= 0 && \text{in } \Omega, \end{aligned} \tag{2.75}$$

see also (1.26). Given Assumptions 2.4.1, the unique solvability of (2.75) is guaranteed by standard results on parabolic equations.
Let us mention that for the forward problem it may be more efficient to use the semi-implicit scheme of [13], where $A'(\nabla y_i)$ is approximated by $M(\nabla y_{j-1})\nabla y_j$. However, to show Fréchet differentiability with the above technique of applying the solution operator $\mathcal{S}$ recursively, higher regularity properties are required. In particular, to our best knowledge, the gradient of the previous time step solution—which appears in the ellipticity coefficient—has to be bounded in $L^\infty(\Omega)$ to obtain a convenient result [137]. For more information see Section 2.7.3.
Let us now define for given y the weak formulation of the adjoint equation in the time continuous setting (cf. (1.24)):

$$\begin{aligned} -\varepsilon(\eta, \partial_t p) + \varepsilon(A''(\nabla y)\nabla\eta, \nabla p) + \frac{1}{\varepsilon}(\psi''(y)\eta, p) &= 0 \qquad && \forall \eta \in L^2(0,T;H^1(\Omega)), \\ p(T) &= \tfrac{1}{\varepsilon}(y(T) - y_\Omega) && \text{in } \Omega. \end{aligned} \tag{2.76}$$

After the substitution $t \mapsto -t$, as for the linearized equation (2.75), the existence of a unique adjoint state p as a solution of (2.76) follows. In analogy to the discretization of the state equation, but taking into account the backward-in-time nature, we use the piecewise constant time discrete $p_\tau \in P_\tau$, where

$$P_\tau := \{p_\tau : Q \to \mathbb{R} \mid p_\tau(t,.) \in H^1(\Omega), p_\tau(.,x) \text{ constant in } \hat{I}_j \text{ for } j = 1, \dots, N\},$$

with $\hat{I}_j := [t_{j-1}, t_j)$ and we use the notation $p_{N+1} := p(T)$. The Galerkin scheme

$$\varepsilon(A''(\nabla y_j)\nabla\varphi, \nabla p_j) + (\tfrac{1}{\varepsilon}\psi''(y_j)\varphi + \tfrac{\varepsilon}{\tau_j}\varphi, p_j) = \tfrac{\varepsilon}{\tau_j}(\varphi, p_{j+1}) \qquad \text{for } j = N, \dots, 1, \tag{2.77}$$

starting with $p_{N+1} = \frac{1}{\varepsilon}(y_N - y_\Omega)$, then determines the approximation p_τ of p. Given (2.65) and the symmetry of $\mathcal{S}'(g_j)$ with $g_j = \frac{1}{\varepsilon}u_j + \frac{1}{\tau}_j y_{j-1}$ and $y_j = \mathcal{S}(g_j)$ we have

$$p_j = \tfrac{1}{\tau_j}\mathcal{S}'(g_j)p_{j+1} \qquad \text{for } j = N, \dots, 1.$$

With (2.73) this leads to

$$(z_j, p_{j+1}) = (\mathcal{S}'(g_j)(\tfrac{1}{\varepsilon}v_j + \tfrac{1}{\tau_j}z_{j-1}), p_{j+1}) = (\tfrac{1}{\varepsilon}v_j + \tfrac{1}{\tau_j}z_{j-1}, \mathcal{S}'(g_j)p_{j+1}) = \tfrac{\tau_j}{\varepsilon}(v_j, p_j) + (z_{j-1}, p_j), \tag{2.78}$$

for $j = N, \dots, 1$ and consequently we have

$$(y_N - y_\Omega, z_N) = \varepsilon(p_{N+1}, z_N) = \tau_j(p_N, v_N) + \varepsilon(p_N, z_{N-1}) = \dots = \sum_{j=1}^{N} \tau_j(p_j, v_j) = (p_\tau, v_\tau),$$

using $z_0 = 0$. Altogether, we have shown

Corollary 2.4.6. *Under Assumptions 2.4.1 the reduced cost functional $j_\tau : U_\tau \to \mathbb{R}$ is Fréchet differentiable and the derivative can be represented as*

$$\nabla j_\tau(u_\tau) = \frac{\lambda}{\varepsilon} u_\tau + p_\tau, \tag{2.79}$$

where p_τ is the solution of the discrete adjoint equation (2.77).

Remark 2.4.7. Note that there is some freedom in scaling the adjoint variable p_τ. By defining $\tilde{p}_{N+1} = y_N - y_\Omega = \varepsilon p_{N+1}$ and repeat the steps after (2.77), one would obtain

$$\nabla j_\tau(u_\tau) = \frac{1}{\varepsilon}(\lambda u_\tau + \tilde{p}_\tau).$$

This formulation is equivalent to (2.79) upon replacing $p_\tau \to \frac{1}{\varepsilon}\tilde{p}_\tau$. Numerically, however, there was observed no difference between both versions.

2.5 Convergence with respect to a regularization of A

In the previous section A had to fulfill Assumptions 2.4.1.a and b. However, as mentioned in the beginning, typical anisotropy functions A only fulfill Assumptions 2.4.1.a. In order to guarantee Fréchet differentiability for the numerical approach, we regularize such an A to A_δ, so that in addition Assumptions 2.4.1.b hold. An example of such a regularization is given and discussed in (2.98) and the paragraph before. In this subsection we consider the dependence on δ of the solutions of the in time discretized optimization problem (2.59)–(2.60). To consider convergence with $\delta \to 0$ we need that $A'_\delta \to A'$. However, the results of this section do not require Fréchet differentiability yet, such that Assumptions 2.4.1.a on A_δ are sufficient. We denote by $y_\tau \in Y_\tau$ the solution of (2.60) with A, while $y^\delta_\tau \in Y_\tau$ shall be given as the solution of the regularized equation

$$\varepsilon \frac{(y^\delta_j, \varphi) - (y^\delta_{j-1}, \varphi)}{\tau_j} + \varepsilon (A'_\delta(\nabla y^\delta_j), \nabla \varphi) + \frac{1}{\varepsilon}(\psi'(y^\delta_j), \varphi) = (u_j, \varphi) \quad \forall \varphi \in H^1(\Omega),\ j = 1, \dots, N, \tag{2.80}$$

and $y^\delta_\tau(0, \cdot) = y_0$. As before we define the reduced cost functional by

$$j_{\tau,\delta}(u_\tau) = \frac{1}{2}\|y^\delta_N(u_\tau) - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2\varepsilon}\|u_\tau\|^2_{L^2(Q)}. \tag{2.81}$$

We note that to not overload the notation, j_δ is used in place of $j_{\tau,\delta}$ as long as it is clear from the context that τ is considered fixed. The goal of this section is to derive a convergence result for minimizers of a sequence of $j_{\tau,\delta}$ to minimizers of $j_{\tau,0}$ in the limit $\delta \to 0$ and to minimizers of j when additionally $\tau \to 0$ holds. Therefore some convergence behavior of the δ-dependent solution y^δ_τ is needed that then is combined with results concerning $\tau \to 0$ from Section 2.3. The behavior of y^δ_τ will be covered by the following two auxiliary results.

Theorem 2.5.1. *Let the Assumptions 2.4.1.a hold and in addition let $A'_\delta : \mathbb{R}^d \to \mathbb{R}^d$ be strongly monotone with a constant C_A independent of δ, i.e.,*

$$(A'_\delta(p) - A'_\delta(q), p - q) \geq C_A |p - q|^2 \qquad \forall \delta > 0, p, q \in \mathbb{R}^d,$$

and furthermore let

$$|A'_\delta(p) - A'(p)| \leq \eta(\delta) \qquad \forall p \in \mathbb{R}^d.$$

Then, for fixed y_0, u_τ and $\max_j \tau_j$ *the solutions $y_\tau(u_\tau)$ and $y^\delta_\tau(u_\tau)$ of* (2.60) *and* (2.80), *respectively, satisfy the following estimate*

$$\|y_\tau(u_\tau) - y^\delta_\tau(u_\tau)\|_{L^\infty(0,T;L^2(\Omega))} + \|\nabla y_\tau(u_\tau) - \nabla y^\delta_\tau(u_\tau)\|_{L^2(0,T;L^2(\Omega))} \leq C_{A,\psi,T}\,\eta(\delta). \tag{2.82}$$

The proof uses the same idea as the proof of Theorem 2.2.4. Proceeding similarly as in Lemma 2.4.2 is possible, but only gives an estimate $\|y^\delta_\tau(u_\tau) - y_\tau(u_\tau)\|_{H^1(\Omega)^N} \leq C_\tau \eta(\delta)$, so the limit $\tau \to 0$ that we consider later on would not be possible without further effort.

Proof: We note down the differences by a prescript Δ, e.g., $\Delta y_\tau := y_\tau - y^\delta_\tau$. With $\frac{1}{2}(a^2 - b^2) \leq (a-b)a$, testing the defining equations (2.60) and (2.80) with Δy_j and using that A'_δ is strongly monotone as well as $(\psi'(x) - \psi'(y), x - y) \geq -C_\psi |x-y|^2$, we obtain

$$\begin{aligned}
&\tfrac{1}{2}\left(\|\Delta y_j\|^2 - \|\Delta y_{j-1}\|^2\right) + \tau_j C_A \|\nabla \Delta y_j\|^2 \\
&\quad \leq (\Delta y_j - \Delta y_{j-1}, \Delta y_j) + \tau_j \left(A'_\delta(\nabla y_j) - A'_\delta(\nabla y^\delta_j), \nabla \Delta y_j\right) \\
&\quad \leq (\Delta y_j - \Delta y_{j-1}, \Delta y_j) + \tau_j \left(A'(\nabla y_j) - A'_\delta(\nabla y^\delta_j), \nabla \Delta y_j\right) + \tau_j \left(A'_\delta(\nabla y_j) - A'(\nabla y_j), \nabla \Delta y_j\right) \\
&\quad = -\tfrac{\tau_j}{\varepsilon^2}\left(\psi'(y_j) - \psi'(y^\delta_j), \Delta y_j\right) + \tau_j \underbrace{\left(A'_\delta(\nabla y_j) - A'(\nabla y_j), \nabla \Delta y_j\right)}_{\leq \|A'_\delta(\nabla y_j) - A'(\nabla y_j)\| \|\nabla \Delta y_j\| \leq |\Omega| \eta(\delta) \|\nabla \Delta y_j\|} \\
&\quad \leq \tfrac{\tau_j}{\varepsilon^2} C_\psi \|\Delta y_j\|^2 + \tfrac{\tau_j C_A}{2}\|\nabla \Delta y_j\|^2 + \tfrac{|\Omega|^2 \tau_j}{2 C_A}\eta(\delta)^2.
\end{aligned}$$

In the last step we used scaled Young's inequality (see Theorem 1.2.6) with the scaling C_A. We now sum over $j = 1, \ldots, J$ and get

$$\tfrac{1}{2}\|\Delta y_J\|^2 + \tfrac{C_A}{2}\sum_{j=1}^{J} \tau_j \|\nabla \Delta y_j\|^2 \leq \tfrac{1}{2}\tfrac{|\Omega|^2}{C_A}\eta(\delta)^2 \sum_{j=1}^{J}\tau_j + \tfrac{1}{2}\tilde{C}_\psi \sum_{j=1}^{J}\tau_j \|\Delta y_j\|^2, \tag{2.83}$$

for all $1 \leq J \leq N_\tau$. Here we defined $\tilde{C}_\psi := \frac{C_\psi}{\varepsilon^2}$. Omitting the gradient term on the left, absorbing the J-th term from the right and using $\frac{1}{(1 - \tilde{C}_\psi \tau_J)} \leq c$, we obtain

$$\|\Delta y_J\|^2 \leq \frac{c|\Omega|^2}{C_A} T \eta(\delta)^2 + c\tilde{C}_\psi \sum_{j=1}^{J-1} \tau_j \|\Delta y_j\|^2.$$

To this we apply the discrete Gronwall Lemma 1.2.8 which yields

$$\|\Delta y_J\|^2 \leq \frac{c|\Omega|^2}{C_A} T \eta(\delta)^2 \exp\left(c\tilde{C}_\psi T\right). \tag{2.84}$$

Inserting this into (2.83) we finally get for all $J = 1, \ldots, N_\tau$

$$C_A \sum_{j=1}^{J} \tau_j \|\nabla \Delta y_j\|^2 \leq \frac{|\Omega|^2}{C_A} T \eta(\delta)^2 \left(1 + c\tilde{C}_\psi T \exp\left(c\tilde{C}_\psi T\right)\right),$$

which together with (2.84) yields the inequality (2.82). □

With this at hand, together with results from Section 2.2, we obtain the following convergence result.

Corollary 2.5.2. *Let the assumptions of Theorem 2.5.1 be fulfilled and $u_\tau, \tilde{u}_\tau \in U_\tau$ be given. Then the estimate*

$$\begin{aligned}
&\|y_\tau(u_\tau) - y^\delta_\tau(\tilde{u}_\tau)\|_{L^\infty(0,T;L^2(\Omega))} + \|\nabla y_\tau(u_\tau) - \nabla y^\delta_\tau(\tilde{u}_\tau)\|_{L^2(0,T;L^2(\Omega))} \\
&\qquad \leq C_{A,\psi,T}\big(\eta(\delta) + \|u_\tau - \tilde{u}_\tau\|_{L^2(0,T;H^1(\Omega)')}\big)
\end{aligned} \tag{2.85}$$

holds. Hence, given a sequence $(u_\tau)_\tau$ with $u_\tau \rightharpoonup u$ in $L^2(0,T;L^2(\Omega))$ for $\tau \to 0$, there exists $\sigma(\tau)$ with $\lim_{\tau \to 0}\sigma(\tau) = 0$ such that

$$\max_{t \in [0,T]} \|y(u)_{|t} - y^\delta_\tau(u_\tau)_{|t}\|_{L^2(\Omega)} \leq C\left(\eta(\delta) + \sigma(\tau)\right). \tag{2.86}$$

Proof: Estimate (2.85) follows by zero completion with $y_\tau(\tilde{u}_\tau)$, triangle inequality and esti-

mating the resulting terms by Theorem 2.5.1 and Theorem 2.2.4, respectively (2.29).
For the second estimate we recall from Theorem 2.2.7 that if $u_\tau \rightharpoonup u$ in $L^2(0,T;L^2(\Omega))$ there exists $\sigma(\tau)$ with $\lim_{\tau\to 0}\sigma(\tau)=0$ such that $\max_{t\in[0,T]}\|y(u)_{|t} - y_\tau(u_\tau)_{|t}\|_{L^2(\Omega)} \le C\sigma(\tau)$. By inserting $y_\tau(u_\tau)$ and using the triangle inequality together with the first estimate one obtains (2.86). □

Finally, we conclude this section with the following convergence result of global minimizers. The second statement is a generalization of Theorem 2.3.3.

Theorem 2.5.3. *Let the assumptions of Theorem 2.5.1 be fulfilled and* $\lim_{\delta\to 0}\eta(\delta)=0$. *Denote by* u_τ^δ *a global minimizer of* $j_{\tau,\delta}$. *Then it holds:*

1. *Considering* $\delta\to 0$ *for fixed* $\tau>0$, *there exists a subsequence such that it holds* $u_\tau^\delta \to \underline{u}_\tau$ *in* U_τ, $y_\tau^\delta(u_\tau^\delta)\to y_\tau(\underline{u}_\tau)$ *in* Y_τ *and* $j_{\tau,\delta}(u_\tau^\delta)\to j_\tau(\underline{u}_\tau)$ *for* $\delta\to 0$. *Furthermore,* $\underline{u}_\tau$ *is a global minimizer of* j_τ.

2. *Considering* $\tau,\delta\to 0$, *there exists a subsequence such that it holds* $u_\tau^\delta\to\underline{u}$ *in* $L^2(0,T;L^2(\Omega))$, $y_\tau^\delta(u_\tau^\delta)\to y(\underline{u})$ *in* $L^2(0,T;L^2(\Omega))$ *and* $j_{\tau,\delta}(u_\tau^\delta)\to j(\underline{u})$. *Furthermore,* $\underline{u}$ *is a global minimizer of* j.

Proof:

1. Take $\bar{u}_\tau\in U_\tau$ fixed. From Theorem 2.5.1 we obtain $y_\tau^\delta(\bar{u}_\tau)\to y_\tau(\bar{u}_\tau)$ in Y_τ for $\delta\to 0$ and therefore from the boundedness of this sequence and the fact that the u_τ^δ are optimal, we obtain
$$\frac{\lambda}{2\varepsilon}\|u_\tau^\delta\|^2_{L^2(Q)} \le j_{\tau,\delta}(u_\tau^\delta)\le j_{\tau,\delta}(\bar{u}_\tau)\le C.$$
Hence $u_\tau^\delta \rightharpoonup \underline{u}_\tau\in U_\tau$ for a subsequence, which is considered in the following, and consequently $y_N^\delta(u_\tau^\delta)\to y_N(\underline{u}_\tau)$ in $L^2(\Omega)$, see Corollary 2.5.2. Using the definition of $j_{\tau,\delta}$ in (2.81) leads to
$$j_\tau(\underline{u}_\tau)\le\liminf_{\delta\to 0} j_{\tau,\delta}(u_\tau^\delta)\le\limsup_{\delta\to 0} j_{\tau,\delta}(u_\tau^\delta)\le\lim_{\delta\to 0} j_{\tau,\delta}(u_\tau)=j_\tau(u_\tau)\qquad\forall u_\tau\in U_\tau. \quad (2.87)$$
Hence $\underline{u}_\tau$ is a minimizer. Since we can also choose $\underline{u}_\tau$ on the righter part of (2.87), in addition we obtain $j_{\tau,\delta}(u_\tau^\delta)\to j_\tau(\underline{u}_\tau)$.
Since we already have $u_\tau^\delta\rightharpoonup\underline{u}_\tau$, to obtain the strong convergence $u_\tau^\delta\to\underline{u}_\tau$ in U_τ it remains to check that the norms converge. This follows from
$$\tfrac{\lambda}{2\varepsilon}\|u_\tau^\delta\|^2_{L^2(Q)} = j_{\tau,\delta}(u_\tau^\delta)-\tfrac12\|y_N^\delta(u_\tau^\delta)-y_\Omega\|^2_{L^2(\Omega)}\to j_\tau(\underline{u}_\tau)-\tfrac12\|y_N(\underline{u}_\tau)-y_\Omega\|^2_{L^2(\Omega)}=\tfrac{\lambda}{2\varepsilon}\|\underline{u}_\tau\|^2_{L^2(Q)}.$$

2. First we choose an arbitrary but fixed $\bar{u}\in L^2(0,T;L^2(\Omega))$ and a sequence $\bar{u}_\tau\in U_\tau$ with $\lim_{\tau\to 0}\bar{u}_\tau=\bar{u}$ in $L^2(0,T;L^2(\Omega))$. Hence $\lim_{\delta,\tau\to 0} y_\tau^\delta(\bar{u}_\tau)_{|T}=y(\bar{u})_{|T}$ in $L^2(\Omega)$ due to (2.86). As above it holds
$$\frac{\lambda}{2\varepsilon}\|u_\tau^\delta\|^2_{L^2(Q)}\le\frac12\|y_\tau^\delta(\bar{u}_\tau)_{|T}-y_\Omega\|^2_{L^2(\Omega)}+\frac{\lambda}{2\varepsilon}\|\bar{u}_\tau\|^2_{L^2(Q)}\le C.$$
Hence we can deduce a subsequence denoted in the same way with $u_\tau^\delta\rightharpoonup\underline{u}$ in $L^2(0,T;L^2(\Omega))$. Then Corollary 2.5.2 yields that $y_\tau^\delta(u_\tau^\delta)_{|T}\to y(\underline{u})_{|T}$ in $L^2(\Omega)$ and hence
$$j(\underline{u})\le\liminf_{\tau,\delta\to 0} j_{\tau,\delta}(u_\tau^\delta).$$
Respectively, given some arbitrary $\tilde{u}\in L^2(0,T;L^2(\Omega))$ and a sequence $\tilde{u}_\tau$ with $\tilde{u}_\tau\to\tilde{u}$, we obtain $\lim_{\tau,\delta\to 0} j_{\tau,\delta}(\tilde{u}_\tau)=j(\tilde{u})$. Then the assertions follows as in 1. □

Remark 2.5.4. Like in Remark 2.3.4, the previous theorem also holds true if instead of the cost functional (2.81) one considers the whole time horizon ($y_{Q,\tau}\in Y_\tau$)
$$j^Q_{\tau,\delta}(u_\tau)=\frac12\|y_\tau^\delta(u_\tau)-y_{Q,\tau}\|^2_{L^2(Q)}+\frac{\lambda}{2\varepsilon}\|u_\tau\|^2_{L^2(Q)}, \qquad (2.88)$$

with a proof along the same lines. The resulting limit is a minimizer of the reduced problem belonging to (2.57) or (2.58), respectively (with $\frac{1}{\varepsilon}$ inserted). This case even is a little bit easier as one does not need (2.86).

Remark 2.5.5. As in Theorem 2.3.3 we could only show the convergence of a subsequence. This is the case because there might be more than one minimizer. The convergence to a certain minimizer $\underline{u}_\tau^*$ or $\underline{u}^*$ can be achieved by modifying the cost functional by

$$j_{\tau,\delta}^*(u_\tau) = \frac{1}{2}\|y_N^\delta(u_\tau) - y_\Omega\|_{L^2(\Omega)}^2 + \frac{\lambda}{2\varepsilon}\|u_\tau\|_{L^2(Q)}^2 + \frac{1}{2}\|\underline{u}_\tau^* - u\|_{L^2(Q)}^2, \tag{2.89}$$

or an analogous expression for $\underline{u}^*$. The new term measures the distance to the desired optimum. For more information we refer to, e.g., [33, 35, 36, Sections 4], where some other regularization on quasilinear equations—resembling one of our time steps—is considered. The goal there is however different to ours. The authors consider a broader class of anisotropic terms with different growing conditions. In certain cases the optimality system cannot be deduced directly since the solution operator is not Gâteaux differentiable. So they apply a certain regularization and use their analogous result to our Theorem 2.5.1 to take the limit in the regularized optimality conditions that hold in any specific minimizer. In our case this is not possible, since the limiting optimality condition is expected to be of another form since A' is not differentiable and hence the adjoint equation is not well defined. As a remedy one can consider only regions where $|\nabla y| > 0$, cf. the last reference given. However, in that paper the limit procedure works, because already for the regularized equation they are treating this kind of optimality system.
For the numerical application we are interested in, convergence to a specific minimizer is not of interest nor can be expected. This is because we do not know the optimizer a priori, so we cannot include it in the cost functional as in (2.89). Also, by solving merely the first order conditions, we can only find local minimizers. Furthermore, we usually fix one δ and do not consider the limit numerically, since the algorithm is expected to break down for δ too small due to the ill-definedness of the first order conditions in the case $\delta = 0$. We only want to be sure that if we have found a global solution by solving the optimality system of the regularized problem, that then it is close to at least some optimizer of the unregularized system.

2.6 The regularization of a class of anisotropies

In order to be able to perform the numerical simulations in Part 4 later, we have to specify the anisotropy function A. As already mentioned in the previous sections, this function typically can be thought of being absolutely 2-homogeneous. In general however this conflicts with the requirement of A being twice continuously differentiable. Our approach is to apply a regularization, but without further information on A it is unclear how exactly this should be achieved. Therefore this section's goal is to specify the employed A, to introduce an appropriate regularization A_δ and to show that A satisfies Assumptions 2.4.1a. and A_δ fulfills in addition Assumptions 2.4.1b. This guarantees that the results from the preceding chapters can be applied.
First, recall from the introduction on page 9 that A can be written as

$$A(p) = \tfrac{1}{2}|\gamma(p)|^2 \qquad \forall p \in \mathbb{R}^d, \tag{2.90}$$

where the so-called density function $\gamma : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ with $\gamma \in C^2(\mathbb{R}^d \setminus \{0\}) \cap C(\mathbb{R}^d)$ shall be absolutely 1-homogeneous. The terminology 'density function' goes back to the study of sharp interface models, where the surface energy of the interface between a solid and liquid phase, say, is given by $|\Gamma|_\gamma = \int_\Gamma \gamma(\nu)\,\mathrm{d}s$. In the isotropic case $\gamma(p) = |p|$ this would reduce to the area of the interface $|\Gamma|_\gamma = |\Gamma|$. The authors of [1, 53] show for the Allen-Cahn equation (1.9)—with A defined as in (2.90)—that in the limit $\varepsilon \to 0$ the zero level sets converge to a sharp interface Γ moving with $V = \gamma(\nu)\kappa_\gamma$ if $u = 0$. While there exist several approaches to define γ, like, e.g., in [86] or in [45], we constrain ourselves to a class of anisotropies for which the density function γ is introduced in [16, 14]. The corresponding phase field ansatz is studied for instance in [13, 15]. In the following they are referred to as BGN-anisotropies. They allow for the modelling and approximation of a large class of common anisotropies. Also they are well suited to model crystal growth, since crystals build characteristical faces. The basic observation is that for the metric $(\cdot,\cdot)_{\tilde{G}}$ defined by symmetric positive definite $\tilde{G}$, the surface area element can be expressed as $\gamma(\nu) = \sqrt{\nu^T G \nu}$ with $G = \det(\tilde{G})^{\frac{1}{d-1}} \tilde{G}^{-1}$ (see [14]). This motivates the choice of the class of density functions γ given by

$$\gamma(p) = \sum_{l=1}^{L} \gamma_l(p), \qquad \text{where } \gamma_l(p) = \sqrt{p^T G_l p}, \tag{2.91}$$

and $G_l \in \mathbb{R}^{d\times d}$ are symmetric and positive definite. Note that for $p \neq 0$ the gradient of A can then be computed as

$$A'(p) = \gamma(p)\gamma'(p) = \sum_{l,m} \frac{\gamma_m(p)}{\gamma_l(p)} G_l p, \tag{2.92}$$

and A' is continuous also at $p = 0$ with $A'(0) = 0$.
The second derivative exists for $p \neq 0$ and is given by

$$A''(p) = \gamma(p)\gamma''(p) + \gamma'(p)\gamma'(p)^T, \tag{2.93}$$

where

$$\gamma''(p) = \sum_l \left(\frac{G_l}{\gamma_l(p)} - \frac{G_l p (G_l p)^T}{\gamma_l(p)^3} \right).$$

We note that $A'' : \mathbb{R}^d \setminus \{0\} \to \mathbb{R}^{d\times d}$ is continuous as a combination of continuous functions. Furthermore, using arguments from [64] we can show the following properties.

Lemma 2.6.1. *The function $A'' : \mathbb{R}^d \setminus \{0\} \to \mathbb{R}^{d\times d}$ is uniformly positive definite and bounded, i.e., it holds*

$$c_1|q|^2 \leq q^T A''(p) q \leq c_2 |q|^2 \qquad \forall p \in \mathbb{R}^d \setminus \{0\},\ q \in \mathbb{R}^d, \tag{2.94}$$

where the constants $c_1, c_2 > 0$ are independent from p.
Moreover, $A' : \mathbb{R}^d \to \mathbb{R}^d$ is Lipschitz continuous and strongly monotone.

Proof: First we note that for the density functions the following relation holds:

$$v^T \gamma''(p) v \geq c_0 |v|^2 \qquad \forall p, v \in \mathbb{R}^d \text{ with } p^T v = 0 \text{ and } |p| = 1. \tag{2.95}$$

This follows by an application of the Cauchy-Schwarz inequality

$$v^T \gamma''(p) v = \sum_l \left(\frac{v^T G_l v}{\gamma_l(p)} - \frac{(v^T G_l p)^2}{\gamma_l(p)^3} \right) \geq \sum_l \left(\frac{v^T G_l v}{\gamma_l(p)} - \frac{\gamma_l(p)^2 \gamma_l(v)^2}{\gamma_l(p)^3} \right) = 0,$$

where equality does not hold for $p^T v = 0$. Further note that the set

$$\{(p, v) \in \mathbb{R}^d \times \mathbb{R}^d \ \mid\ |p| = |v| = 1,\ p^T v = 0\}$$

is compact as a closed subset of the compact set $S^{d-1} \times S^{d-1}$, where S^{d-1} is the $d-1$ dimensional unit sphere. Hence there exists the minimum of $v^T \gamma''(p) v$ on this set and a rescale argument yields the desired relation (2.95).
To show the positive definiteness of $A''(p)$, for some fixed $p \neq 0$ with $|p| = 1$ we decompose $q \in \mathbb{R}^d$ as $q = \alpha p + p^\perp$, where $p^T p^\perp = 0$. From the 1-homogeneity of γ it follows $\gamma''(p)p = \frac{\partial}{\partial a}\gamma'(ap)_{|a=1} = \frac{\partial}{\partial a}\gamma'(p)_{|a=1} = 0$, i.e. $\gamma''(p)q = \gamma''(p)p^\perp$. Having this in mind, we now distinguish two cases.
If $p^\perp \neq 0$, it follows using (2.95) and neglecting the nonnegative second term

$$q^T A''(p) q = \gamma(p)(q^T \gamma''(p) q) + (q^T \gamma'(p))^2 \geq \gamma(p) c_0 |p^\perp|^2 > 0.$$

On the other, hand if $p^\perp = 0$ the first term vanishes and from $p^T\gamma'(p) = \frac{\partial}{\partial a}\gamma(ap)_{|a=1} = \frac{\partial}{\partial a}a\gamma(p)_{|a=1} = \gamma(p)$ we obtain

$$q^T A''(p)q = \alpha^2\gamma(p)(p^T\gamma''(p)p) + \alpha^2(p^T\gamma'(p))^2 \geq \alpha^2\gamma(p)^2 > 0,$$

since $p \neq 0$.
Finally, the uniform positive definiteness and boundedness again follow from a compactness argument noting that the function $(p,q) \mapsto q^T A''(p)q$ is continuous on the compact set $S^{d-1} \times S^{d-1}$ and absolutely 2-homogeneous in q as well as absolutely 0-homogeneous in p.

Now we go for the Lipschitz continuity of $A' : \mathbb{R}^d \to \mathbb{R}^d$. For this, let $p, q \in \mathbb{R}^d$. We denote the straight line between p and q by $[p,q]$.
If $0 \notin [p,q]$, then $\gamma_{|[p,q]} \in C^2$ and using the boundedness of A'' obtained from the just shown relation (2.94), one obtains from the mean value theorem that

$$|A'(p) - A'(q)| = |A''(\xi_{p,q})(p-q)| \leq c_2|p-q|,$$

where $\xi_{p,q} \in [p,q]$ is an intermediate point.
The Lipschitz continuity in the case $[0,q]$ follows from taking the limit $r \to 0$ in the above case for $[rp,q]$ and the continuity of A'. The same can be done for $[p,0]$.
Finally, if $0 \in [p,q]$ we have $p = \alpha q$ with a certain $\alpha \leq 0$ and using that $|p-q| = |(\alpha-1)q| = (-\alpha+1)|q| = |\alpha q| + |q| = |p| + |q|$, we obtain

$$|A'(p)-A'(q)| \leq |A'(p)-A'(0)| + |A'(0)-A'(q)| \leq c_2|p-0| + c_2|0-q| = c_2(|p|+|q|) = c_2|p-q|,$$

and the Lipschitz continuity is shown.

With the same case distinction we can show the strong monotonicity of A'.
This time, in the case $0 \notin [p,q]$ we can use the lower estimate in (2.94) of A'' to obtain from the mean value theorem that

$$(A'(p) - A'(q), p-q) = (A''(\xi_{p,q})(p-q), p-q) \geq c_1|p-q|^2.$$

The cases $[0,q]$ and $[p,0]$ once more follow from continuity.
Ultimately, if $0 \in [p,q]$ we can again write $p = \alpha q$ with $\alpha \leq 0$ and using the previous it follows

$$\begin{aligned}(A'(p) - A'(q), p-q) &= (A'(p) - A'(0), \underbrace{p-q}_{=(1-1/\alpha)p}) + (A'(0) - A'(q), \underbrace{p-q}_{=(\alpha-1)q}) \\ &= \big(1 - \tfrac{1}{\alpha}\big)(A'(p) - A'(0), p-0) + (1-\alpha)(A'(0) - A'(q), 0-q) \\ &\geq \big(1 - \tfrac{1}{\alpha}\big)c_1|p-0|^2 + (1-\alpha)c_1|0-q|^2 \\ &= c_1\left((\alpha^2 - \alpha)|q|^2 + (1-\alpha)|q|^2\right) = c_1|(\alpha-1)q|^2 = c_1|p-q|^2.\end{aligned}$$

With this all properties are shown. □

Our goal for regularizing A is that $A_\delta \in C^2(\mathbb{R}^d)$ shall fulfill the requirements for the existence of an optimal control (Assumptions 2.4.1), that the derivative shall be simple to evaluate and that the influence on the interfacial region (i.e., $\nabla y \not\approx 0$) shall be little. The approach is to modify the γ_l, but one could also think of regularizing, e.g., the quotient appearing in the sum in (2.92). Among various choices we considered (see also Remark 2.7.2), the most promising was to alter the functions γ_l by a small shift of δ, i.e.,

$$\gamma_l^\delta := \sqrt{\gamma_l^2 + \delta}, \tag{2.96}$$

where $\delta > 0$. This we use in the following and, like in Section 2.5, we denote the resulting regularizations by A_δ and γ_δ. Both lie in $C^\infty(\mathbb{R}^d)$ now. A very convenient property for this choice is that

$$\gamma_\delta(p) = \tilde{\gamma}((p, \sqrt{\delta})^T), \tag{2.97}$$

where $\tilde{\gamma}$ is defined employing the matrices $\tilde{G}_l := \begin{pmatrix} G_l & \\ & 1 \end{pmatrix}$. Hence one can also view the regularized anisotropy A_δ on $\mathbb{R}^d$ as an unregularized BGN-anisotropy $\tilde{A}$ on $\mathbb{R}^{d+1}$ for which above properties hold.
The derivatives still have the same structures as in (2.92) and (2.93), namely

$$A'_\delta(p) = \sum_{l,m=1}^{L} \frac{\gamma_m^\delta(p)}{\gamma_l^\delta(p)} G_l p, \tag{2.98}$$

$$A''_\delta(p) = \gamma_\delta(p)\gamma''_\delta(p) + \gamma'_\delta(p)\gamma'_\delta(p)^T, \tag{2.99}$$

$$\text{with} \quad \gamma''_\delta(p) = \sum_{l=1}^{L} \left(\frac{G_l}{\gamma_l^\delta(p)} - \frac{G_l p (G_l p)^T}{\gamma_l^\delta(p)^3} \right), \tag{2.100}$$

though these hold in the regularized version for all $p \in \mathbb{R}^d$. Note that for $L = 1$, i.e., for $A(p) = \frac{1}{2} p^T G p$ which includes the isotropic case, it holds $A'_\delta = A'$. As in this case A is already smooth by itself and hence there is no need for regularization anyway, this is a particularly convenient property of this kind of regularization. Using the handy relations

$$A_\delta(p) = \tilde{A}\left(\begin{smallmatrix} p \\ \sqrt{\delta} \end{smallmatrix}\right), \quad A'_\delta(p) = \left(\tilde{A}'\left(\begin{smallmatrix} p \\ \sqrt{\delta} \end{smallmatrix}\right)\right)_{1,\dots,d}, \quad A''_\delta(p) = \left(\tilde{A}''\left(\begin{smallmatrix} p \\ \sqrt{\delta} \end{smallmatrix}\right)\right)_{\substack{1,\dots,d \\ 1,\dots,d}}, \tag{2.101}$$

we can proof the following theorem.

Lemma 2.6.2. *The mappings $A_\delta : \mathbb{R}^d \to \mathbb{R}$ for $\delta \geq 0$ (with $A_{\delta=0} := A$ as shorthand notation) have the following properties:*

a) A_δ fulfill the growth condition $c|p|^2 \leq A_\delta(p) \leq C_\delta + C|p|^2$ for all p with positive constants c, C, C_δ, where only C_δ may depend on δ.

b) A'_δ are Lipschitz continuous and strongly monotone on $\mathbb{R}^d$ with constants independent of δ and $A'_\delta(0) = 0$.

c) A''_δ induce uniformly equivalent norms on $\mathbb{R}^d$ for $\delta > 0$, i.e., there exist constants c_0, C such that

$$c_0 \|q\|^2 \leq q^T A''_\delta(p) q \leq C \|q\|^2 \qquad \forall p, q \in \mathbb{R}^d, \delta > 0,$$

and $A''_\delta(0) = L \sum_{l=1}^{L} G_l$. Furthermore, if $\delta = 0$, the same holds true for all $p \neq 0$.

d) $A'_{(.)}(p)$ is Hölder continuous with exponent $1/2$ and with a constant independent of p. It especially holds

$$|A_\delta{}'(p) - A'(p)| \leq C\sqrt{\delta} \qquad \forall p \in \mathbb{R}^d, \delta > 0. \tag{2.102}$$

In particular the Assumptions 2.4.1 are fulfilled if $\delta > 0$, Assumptions 2.4.1a. hold for A and finally the estimate assumed in Theorem 2.5.1 holds.

Proof: For $\delta = 0$ property a) with $C_\delta = 0$ is an immediate consequence of the 2-homogeneity. Noting (2.101) we can use this special case to deduce for arbitrary δ

$$A_\delta(p) = \tilde{A}\left(\begin{smallmatrix} p \\ \sqrt{\delta} \end{smallmatrix}\right) \leq C \left|\left(\begin{smallmatrix} p \\ \sqrt{\delta} \end{smallmatrix}\right)\right|^2 = C\left(|p|^2 + \delta\right) \quad \text{and} \quad A_\delta(p) = \tilde{A}\left(\begin{smallmatrix} p \\ \sqrt{\delta} \end{smallmatrix}\right) \geq c\left(|p|^2 + \delta\right) \geq c|p|^2,$$

from which the desired estimate follows.
For the Lipschitz continuity and strong monotonicity in the unregularized case we recall the result from Lemma 2.6.1. For $\delta > 0$ these follow again from (2.101) as follows. Using that an additional $(d+1)$th entry can only make the norm bigger, one obtains

$$|A'_\delta(p) - A'_\delta(q)| \leq \left|\tilde{A}'\left(\left(\begin{smallmatrix} p \\ \sqrt{\delta} \end{smallmatrix}\right)\right) - \tilde{A}'\left(\left(\begin{smallmatrix} q \\ \sqrt{\delta} \end{smallmatrix}\right)\right)\right| \leq L \left| \begin{pmatrix} p \\ \sqrt{\delta} \end{pmatrix} - \begin{pmatrix} q \\ \sqrt{\delta} \end{pmatrix} \right| = L|p - q| \qquad \forall p, q \in \mathbb{R}^d,$$

and hence the Lipschitz continuity holds also for A'_δ. The strong monotonicity follows from

$$\begin{aligned}(A'_\delta(p) - A'_\delta(q), p-q) &= \left(\tilde{A}'\left(\left(\begin{smallmatrix} p \\ \sqrt{\delta}\end{smallmatrix}\right)\right) - \tilde{A}'\left(\left(\begin{smallmatrix} q \\ \sqrt{\delta}\end{smallmatrix}\right)\right), \begin{pmatrix} p-q \\ 0 \end{pmatrix}\right) \\ &= \left(\tilde{A}'\left(\left(\begin{smallmatrix} p \\ \sqrt{\delta}\end{smallmatrix}\right)\right) - \tilde{A}'\left(\left(\begin{smallmatrix} q \\ \sqrt{\delta}\end{smallmatrix}\right)\right), \begin{pmatrix} p-q \\ \sqrt{\delta} - \sqrt{\delta} \end{pmatrix}\right) \\ &\geq C\left|\begin{pmatrix} p \\ \sqrt{\delta}\end{pmatrix} - \begin{pmatrix} q \\ \sqrt{\delta}\end{pmatrix}\right|^2 = C|p-q|^2 \qquad \forall p,q \in \mathbb{R}^d.\end{aligned}$$

Note that in both cases only the constants for $\tilde{A}'$ appear which are independent from δ. Moreover, since $\tilde{A}''$ induces uniformly equivalent norms on $\mathbb{R}^{d+1}$ the same holds for A''_δ on $\mathbb{R}^d$. Hence $A''_\delta(p)$ is bounded independent from p and δ and we obtain assertion c).
Finally, Hölder continuity of $A'_\delta(p)$ with respect to δ is a consequence of the Lipschitz continuity of $\tilde{A}'$ as can be seen from

$$\begin{aligned}|A'_{\delta_1}(p) - A'_{\delta_2}(p)| &= \left|\left(\tilde{A}'\left(\begin{smallmatrix} p \\ \sqrt{\delta_1}\end{smallmatrix}\right) - \tilde{A}'\left(\begin{smallmatrix} p \\ \sqrt{\delta_2}\end{smallmatrix}\right)\right)_{1,\dots,d}\right| \leq \left|\tilde{A}'\left(\begin{smallmatrix} p \\ \sqrt{\delta_1}\end{smallmatrix}\right) - \tilde{A}'\left(\begin{smallmatrix} p \\ \sqrt{\delta_2}\end{smallmatrix}\right)\right| \\ &\leq C\left|\left(\begin{smallmatrix} p \\ \sqrt{\delta_1}\end{smallmatrix}\right) - \left(\begin{smallmatrix} p \\ \sqrt{\delta_2}\end{smallmatrix}\right)\right| = C\left|\sqrt{\delta_1} - \sqrt{\delta_2}\right| \leq C\sqrt{|\delta_1 - \delta_2|},\end{aligned}$$

where for the last inequality we used the known fact that the square root function is $\frac{1}{2}$-Hölder continuous. □

Remark 2.6.3. Note that the property d) seems to be suboptimal at first sight if compared to property b) that holds with respect to the variable p. Recalling the definition (2.96) however, one notices that actually the square of p (or rather a function behaving like that) appears under the square root compared to δ that enters only as it is. So this actually is an artifact of the way we defined δ to show up in the regularization. Replacing $\delta \to \tilde{\delta}^2$ one would obtain Lipschitz continuity with respect to $\tilde{\delta}$. For the discussion in Section 2.5, the actual rate of convergence was not of importance however.

2.7 A semi-implicit splitting scheme for the anisotropy

So far we have considered an implicit discretization (see, for instance (2.60)) of the state equation (1.9). Regarding the fact that solving such an implicit discretization requires a nonlinear solver, it might be tempting to consider a semi-implicit discretization for the state equation instead. In the following subsection we will introduce such a semi-implicit discretization and discuss some results that find their analogues in the previous sections. After that, we will give the adjoint equation that is required to formulate the first order condition as well as the linearized state and the additional adjoint equation which are required to compute the action of the Hessian on a vector. The derivation of these equations will be done on a formal level, since there are some serious problems hindering a more rigorous treatment, as will be explained by the end of Section 2.7.2. In the final subsection we will point out one of these problems in more detail—namely finding proper spaces for a continuity result for the chained time step problems. Nevertheless, the formally derived equations can still be fed into the optimization solver to obtain numerical solutions of the control problem. The results for this and a comparison between the implicit and semi-implicit schemes is presented in Section 4.5.1.

2.7.1 Existence and stability result

Let us now introduce the concrete form of the semi-implicit splitting scheme. We will maintain the same partition of the interval $[0,T]$ into I_j as in Section 2.2. Note that in contrast to there we prefer not to use the notion of y_τ and u_τ living in the spaces Y_τ and U_τ defined in (2.11), respectively, since the scheme discussed in this section can no longer be obtained by applying a

discontinuous-Galerkin method on (1.9). Rather we prefer to denote the state and control as $(y_j)_{j=1,\dots,N} \in H^1(\Omega)^N$ and $(u_j)_{j=1,\dots,N} \in L^2(\Omega)^N$, respectively.
The general idea is to seek a splitting of the anisotropy A' that is given by a linearization of it as follows

$$A'(\nabla\varphi_j) \approx M(\nabla\varphi_{j-1})\nabla\varphi_j + b(\nabla\varphi_{j-1}), \tag{2.103}$$

where M and b are still to be determined. Here we already indicated the place where each time step enters. Note that with this convention the known previous time step solution appears in the coefficients and the seeked solution φ_j only appears linearly. As we have to set up the first order conditions for the numerical approach, the functions M and b should be regularized, preferably using a regularization related to a regularization of A'.
Let us get more concrete at this point by explicitly considering the anisotropy and regularizations given in Section 2.6. Considering the expressions from (2.92) and (2.98), we observe that we can write

$$A'(p) = M(p)p \qquad \text{or} \qquad A'_\delta(p) = M_\delta(p)p, \tag{2.104}$$

respectively, with

$$M(p) = \begin{cases} \displaystyle\sum_{l,m=1}^{L} \frac{\gamma_m(p)}{\gamma_l(p)} G_l & \text{if } p \neq 0, \\ \displaystyle L\sum_{l=1}^{L} G_l & \text{if } p = 0, \end{cases} \tag{2.105}$$

or

$$M_\delta(p) = \sum_{l,m=1}^{L} \frac{\gamma_m^\delta(p)}{\gamma_l^\delta(p)} G_l. \tag{2.106}$$

Note that

$$M_\delta(0) = \sum_{l,m=1}^{L} \frac{\sqrt{\delta}}{\sqrt{\delta}} G_l = L\sum_{l=1}^{L} G_l,$$

motivating the definition of M for $p = 0$ in (2.105). The unregularized version M is not continuous at the origin, in contrast to $A'(p)$, which vanishes there due to the extra factor of p. The splitting in (2.104) and (2.105) which is of the form (2.103) with $b \equiv 0$, was first proposed and analyzed in [13, 15]. In particular, the authors showed the following stability inequality

$$(M(q)p)^T(p-q) \geq A(p) - A(q) \qquad \forall p, q \in \mathbb{R}^d, \tag{2.107}$$

which is essential for their scheme to provide energy stability. Note that, in contrast to here, they use the double obstacle potential. Our first step after proposing (2.106) is to check if (2.107) still holds true in the regularized case. After that we will specify our scheme and check if it also fulfills energy stability using said inequality.

Lemma 2.7.1. *Let $\delta > 0$. Then for M_δ defined in (2.106), there holds the following stability inequality:*

$$(M_\delta(q)p)^T(p-q) \geq A_\delta(p) - A_\delta(q) \qquad \forall p, q \in \mathbb{R}^d. \tag{2.108}$$

Furthermore, M_δ is uniformly positive definite, bounded and Lipschitz continuous.

Proof: The proof for (2.108) is an extension of the proof of [13, Lemma 2.2]. Inserting the definition of M_δ, γ_l and γ_l^δ (cf. eqs. (2.91) and (2.96)) one obtains

$$(M_\delta(q)p)^T(p-q) = \gamma^\delta(q)\sum_l \gamma_l^\delta(q)^{-1}(\gamma_l(p)^2 - q^T G_l p).$$

The trick now is to insert $0 = \delta - \delta$ to obtain

$$(M_\delta(q)p)^T(p-q) = \gamma^\delta(q)\sum_l \gamma_l^\delta(q)^{-1}\gamma_l^\delta(p)^2 - \gamma^\delta(q)\sum_l \gamma_l^\delta(q)^{-1}(q^T G_l p + \delta).$$

Applying Cauchy-Schwarz backwards it is easy to verify that the first term can be estimated by

$$\gamma^\delta(q)\sum_l \gamma_l^\delta(q)^{-1}\gamma_l^\delta(p)^2 \geq \left(\sum_l \gamma_l^\delta(q)^{\frac{1}{2}}\frac{\gamma_l^\delta(p)}{\gamma_l^\delta(q)^{\frac{1}{2}}}\right)^2 = \gamma^\delta(p)^2.$$

Also using Cauchy-Schwarz one obtains

$$q^T G_l p + \delta \leq \gamma_l^\delta(q)\gamma_l^\delta(p).$$

Putting together the previous results and applying Young's inequality one finally gets

$$(M_\delta(q)p)^T(p-q) \geq \gamma^\delta(p)^2 - \gamma^\delta(q)\gamma^\delta(p) \geq \tfrac{1}{2}\gamma^\delta(p)^2 - \tfrac{1}{2}\gamma^\delta(q)^2.$$

Recalling the definition of A_δ this is exactly (2.108).
Finally, uniformly positive definiteness and boundedness for instance follow from (2.97) and the fact that the $\tilde{\gamma}_l$ define norms. The Lipschitz continuity is a consequence of the fact that the derivative, which for instance is given later in (2.116), is bounded in q. □

Remark 2.7.2. If one considers a regularization of the type

$$\gamma_l^\delta(q) := \gamma_l(q)^{1+\delta},$$

the stability inequality (2.108) is not satisfied and several other problems appear.
First, a short computation reveals that

$$A_\delta'(p) = (1+\delta)\sum_{l,m=1}^{L}\frac{\gamma_m(p)^{1+\delta}}{\gamma_l(p)^{1-\delta}}G_l p.$$

We note that due to the different growth condition $|A_\delta'(p)| \leq C|p|^{1+2\delta}$ the Nemytskii operator belonging to A_δ' no longer provides a mapping $L^2(\Omega) \to L^2(\Omega)$ (yielding several issues in the considerations from the previous chapters). To see that the natural splitting $A_\delta'(p) = M_\delta(p)p$ with

$$M_\delta(p) = (1+\delta)\sum_{l,m=1}^{L}\frac{\gamma_m(p)^{1+\delta}}{\gamma_l(p)^{1-\delta}}G_l$$

does not satisfy (2.108), we insert $q = 0$, since then

$$0 = (M_\delta(0)p)^T p \geq A_\delta(p) - A_\delta(0) = A_\delta(p) > 0$$

for all $p \neq 0$, which is a contradiction. Furthermore, uniformly positive definiteness and boundedness is clearly not satisfied.
Numerically we observed several issues that arise apart from the above also from the fact that the denominator with this regularization is actually allowed to vanish. Also with the altered power dependency, regions with high values for p (that is, ∇y) are influenced strongly by this regularization, which are precisely represented by the vicinity of the interface that mainly drives the evolution of the phase field.

We now want to introduce the discrete scheme that we are going to consider in the remainder of this chapter and for which we can show energy stability under certain conditions using the previous result. It is given by

$$(y_j,\varphi) + \tau_j(M_\delta(\nabla y_{j-1})\nabla y_j, \nabla\varphi) = (y_{j-1},\varphi) + \frac{\tau_j}{\varepsilon}(u_j,\varphi) - \frac{\tau_j}{\varepsilon^2}(\psi'(y_{j-1}),\varphi) \qquad \forall\varphi \in H^1(\Omega) \tag{2.109}$$

for $j = 1,\dots,N$ and where $y_0 \in H^1(\Omega)$ shall be given. This defines time step wise the mapping $S_{\text{semi}} : L^2(\Omega)^N \to H^1(\Omega)^N, (u_j)_{j=1,\dots,N} \mapsto (y_j)_{j=1,\dots,N}$. Note that we have set $b_\delta = 0$ as suggested by the splittings from the previous discussion. The considerations that follow would also hold true for other choices in (2.103) as long as no specific form of M_δ is assumed, but we will keep $b_\delta = 0$ for sake of presentation, since the equations quickly become clumsy. It is not difficult however to add the suitable terms if needed, since the more complicated ones arise from M_δ. Let us therefore state at this place the assumptions on M_δ that will be used in the following.

Assumption 2.7.3. We assume that $M_\delta \in C^2(\mathbb{R}^d)^{d\times d}$ satisfies (2.108) and is bounded and uniformly positive definite, i.e.

$$c_1^M |q|^2 \le q^T M_\delta(p) q \le c_2^M |q|^2 \qquad \forall p, q \in \mathbb{R}^d.$$

For M_δ as given in (2.106) these properties were shown in Lemma 2.7.1. Note that we need the first and second derivative of M_δ to define all equations that are needed for the numerics (see below).
We also note that in (2.109) we chose to take the potential ψ' explicitly, so it appears on the right-hand side as it only depends on the previous time step. From a numerical point of view this choice may seem questionable, since explicit schemes for ψ' are known to lead to stability issues and severe time step restrictions [119]. Let us give some reasons why we nevertheless focus on this kind of discretization here.
First, recall from the previous sections that in the existence theorems it was mostly the term involving ψ' that led to the biggest issues, as on A we imposed suitable growing conditions. Due to the lack of a direct correspondence to the continuous case via discontinuous Galerkin we cannot use the same arguments as done in Section 2.2 to show boundedness of the ψ' term here. Since we would like to concentrate on the novel issues the M_δ term brings along, we decided to include the potential explicitly to simplify the discussion. As a drawback, in the theorems that follow we will then have to assume $d \le 3$ and restrict ourselves to the smooth double-well potential

$$\psi(s) = \tfrac{1}{4}(1-s^2)^2 \qquad \text{and hence} \qquad \psi'(s) = s^3 - s, \tag{2.110}$$

to be able to use the imbedding $H^1(\Omega) \hookrightarrow L^6(\Omega)$, such that $\psi' : H^1(\Omega) \to L^2(\Omega)$ defines a continuous Nemytskii operator.
A second issue is that the reason we introduced the splitting (2.103) was to circumvent a costly application of a nonlinear solver for the state equation. Reintroducing a term as $\psi'(y_j)$ in the state equation would practically nullify this advantage. At this point we should note that there are some schemes that are probably better suited and still linear in y_j (see for instance [67, 68]) but they involve a certain splitting of the potential which leads to additional terms that we want to avoid in the discussion due to the reasons stated above. Finally let us comment that since we do optimal control, we are only interested in evolving the state until a fixed end time point T. Given that we have to evaluate the appearing PDEs many times we cannot afford T being as big as in usual considerations, where one only considers the state equation. Hence we did not observe any numerical issues arising from the explicit discretization in our settings.

Before we go on with discussing the optimal control issues, we have to face the question regarding the solvability of the scheme (2.109). Fortunately this can be dealt with rather quickly (compared to the implicit one (2.13)), since it is of a form that allows us to use standard results from the theory of elliptic PDEs.

Theorem 2.7.4. *Let $d \le 3$ and Assumption 2.7.3 hold. Furthermore, take ψ to be the smooth double-well potential* (2.110). *Then for every $u := (u_j)_{j=1,\dots,N} \in L^2(\Omega)^N$ and $y_0 \in H^1(\Omega)$ there exists a unique solution $y := (y_j)_{j=1,\dots,N} \in H^1(\Omega)^N$ that satisfies the scheme* (2.109).

Proof: The assertion can be shown time step wise. Given $y_{j-1} \in H^1(\Omega)$, from the imbedding $H^1(\Omega) \hookrightarrow L^6(\Omega)$ it follows that the right-hand side of (2.109) is an element of $L^2(\Omega)$. Then, since $M_\delta(\nabla y_{j-1}) \in L^\infty(\Omega)^{d\times d}$ is uniformly positive definite we can apply the lemma of Lax-Milgram to deduce the unique existence of $y_j \in H^1(\Omega)$. □

Remark 2.7.5. Note that although we can consider each time step as a linear elliptic equation, the full system that treats all time steps simultaneously is still quasilinear. In particular, the solution operator S_{semi} (defined below (2.109)) continues to be nonlinear and hence a nonlinear solver is needed to tackle the optimization problem. Furthermore this means that uniqueness of a minimizer is still not provided.

Now we demonstrate how one can use the inequality (2.108) to derive the stability of the splitting scheme for the state equation under certain conditions (most notably we have to cut ψ). This theorem is the analogue of Theorem 2.2.9 in the implicit case.

Theorem 2.7.6. *Let the previous assumptions hold, but replace the double-well potential ψ by a potential $\tilde{\psi}$ that obeys $|\tilde{\psi}''| \le R$. Then provided $\tau_j \le \frac{2\varepsilon^2}{R}$ the scheme for the state equation* (2.109) *is energy stable if $u_j = 0$ for all $j \in \mathbb{N}$, i.e., the energy functional $\mathcal{E}$ is decreasing in time.*

Proof: We test (2.109) (with $\tilde{\psi}'$) with the difference $y_j - y_{j-1}$ and obtain

$$\frac{1}{\tau_j}\|y_j - y_{j-1}\|^2 + (M_\delta(\nabla y_{j-1})\nabla y_j, \nabla y_j - \nabla y_{j-1}) + \frac{1}{\varepsilon^2}\left(\tilde{\psi}'(y_{j-1}), y_j - y_{j-1}\right) = 0. \tag{2.111}$$

The second term we estimate by the stability inequality (2.108) and the third term by the relation

$$\tilde{\psi}'(y_{j-1})(y_j - y_{j-1}) = \tilde{\psi}(y_j) - \tilde{\psi}(y_{j-1}) - \tfrac{1}{2}\tilde{\psi}''(s(\cdot))(y_j - y_{j-1})^2 \ge \tilde{\psi}(y_j) - \tilde{\psi}(y_{j-1}) - \tfrac{R}{2}(y_j - y_{j-1})^2, \tag{2.112}$$

where s is some appropriate intermediate point.
Collecting terms and using the definition of the Ginzburg-Landau energy (2.8) one finds

$$\left(\frac{1}{\tau_j} - \frac{R}{2\varepsilon^2}\right)\|y_j - y_{j-1}\|^2 + \frac{1}{\varepsilon}\left[\mathcal{E}(y_j) - \mathcal{E}(y_{j-1})\right] \le 0 \tag{2.113}$$

and thus $\mathcal{E}(y_j) \le \mathcal{E}(y_{j-1})$ if $\tau_j \le \frac{2\varepsilon^2}{R}$ □

Remark 2.7.7.

1. Note that in comparison to Theorem 2.2.9 we had to cut the potential here. The reason is that the signs in front of the second derivative differ in (2.50) and (2.112). That is, here we have to estimate ψ'' from above instead of below and therefore we have to cut the potential. This theorem would hold analogously without cutting if we included the ψ'-term implicitly.

2. In the numerics we still use the former version of the double-well potential. Recall from Theorem 2.2.10 that we expect y to be restricted to the interval $[-1, 1]$ if y_0 almost everywhere lies therein. In numerical simulations we did not observe strong deviations from this also in the controlled case. Therefore we do not see a problem in leaving ψ' in the equation, as its behavior outside some compact interval becomes irrelevant.

3. The authors of [13] do not need to cut their potential as they use the obstacle potential (2.5) instead.

4. A possible choice for $\tilde{\psi}$ is given by f_0 in (2.7) with x_0 suitably chosen.

2.7.2 First order conditions and linearized equations

In contrast to the implicit case we are faced with several problems when trying to deduce the existence of an optimal solution as well as setting up the first order conditions here. This prevents us from doing such a rigorous consideration as done there. Instead we will derive the first order conditions by means of the formal Lagrangian method (cf. Section 1.1; see also [127, 84]). The arising equations are the time discrete counterparts to (1.23)–(1.27) for the semi-implicit scheme. In the upcoming section we will then add further information on the

problems that we mentioned above. We will show that there are already serious restrictions when proving the continuous dependence of the solution y_j on the previous solution y_{j-1}.
The Lagrangian for problem (2.59) subject to (2.109) is given by

$$\begin{aligned} L(y,p,u) &= \frac{1}{2}\|y_N - y_\Omega\|^2 + \frac{\lambda}{2\varepsilon}\sum_{j=1}^{N}\tau_j\|u_j\|^2 \\ &+ \sum_{j=1}^{N}\Big[-\varepsilon(y_j,p_j) + \varepsilon(y_{j-1},p_j) - \varepsilon\tau_j(M_\delta(\nabla y_{j-1})\nabla y_j,\nabla p_j) - \frac{\tau_j}{\varepsilon}(\psi'(y_{j-1}),p_j) + \tau_j(u_j,p_j)\Big], \end{aligned} \tag{2.114}$$

where the adjoint states $p_j,\ j = 1,\dots N$ act as Lagrange multipliers.
As a quick check, one can verify that by variation with respect to p_j one recovers the semi-implicit state equation (2.109). The gradient of the reduced cost functional $j_\tau(u) = J(S_{\text{semi}}(u),u)$ with J from (2.59) is obtained by deriving the Lagrangian with respect to the control variable

$$\partial_{(u_j)_j}L(y,p,u)[(v_j)_j] = \sum_{j=1}^{N}\tau_j\frac{\lambda}{\epsilon}(u_j,v_j)_{L^2(\Omega)} + \tau_j(p_j,v_j)_{L^2(\Omega)} \overset{!}{=} (\nabla j_\tau(u),v),$$

i.e., we can write $\nabla j_\tau(u) = \left(\frac{\lambda}{\epsilon}u_j + p_j\right)_{j=1\dots,N}$ in terms of the discrete space-time scalar product $(\cdot,\cdot) \coloneqq \sum_{j=1}^{N}\tau_j(\cdot,\cdot)_{L^2(\Omega)}$. Note that this is of the same form as (2.79), however the adjoint state will be defined differently here.
As usual the appearing adjoint state $(p_j)_{j=1,\dots,N}$ can be determined by solving an adjoint equation. It is deduced by varying the Lagrangian with respect to the state variable y and reads as

$$\begin{aligned} (p_N,\varphi) + \tau_N(M_\delta(\nabla y_{N-1})\nabla p_N,\nabla\varphi) &= \frac{1}{\varepsilon}(y_N - y_\Omega,\varphi), \\ (p_j,\varphi) + \tau_j(M_\delta(\nabla y_{j-1})\nabla p_j,\nabla\varphi) &= (p_{j+1},\varphi) - \tau_{j+1}(M'_\delta(\nabla y_j)[\nabla\varphi]\nabla y_{j+1},\nabla p_{j+1}) \\ &\quad - \frac{\tau_{j+1}}{\varepsilon^2}(\psi''(y_j)p_{j+1},\varphi) \qquad j = N-1,\dots,1. \end{aligned} \tag{2.115}$$

The appearing derivative of M_δ is a 3-tensor and, if it is defined as in (2.106), its action to a vector $w \in \mathbb{R}^d$ is given by

$$M'_\delta(q)[w] \coloneqq \sum_{l,m=1}^{L}\frac{1}{\gamma_m^\delta(q)\gamma_l^\delta(q)}w^T G_m q\ G_l - \sum_{l=1}^{L}\frac{\gamma^\delta(q)}{[\gamma_l^\delta(q)]^3}w^T G_l q\ G_l. \tag{2.116}$$

Note in particular that although M_δ is symmetric, its derivative is not symmetric under exchange of all indices, so we cannot simplify by freely swapping around $\nabla\varphi$ to the right of the scalar product. Due to our regularization scheme, the limit $\|q\| \to 0$ is well defined for $\delta > 0$. For $\delta = 0$ the limit $\|q\| \to 0$ becomes divergent.
Finally we will deduce the linearizations of the state and adjoint equation. They are needed to determine the application of the Hessian of $j(u)$ to the direction δu, which formally is given by $\nabla^2 j(u)\delta u = \left(\frac{\lambda}{\epsilon}\delta u_j + \delta p_j\right)_{j=1\dots,N}$. The linearized state equation may be obtained by deriving the state equation (2.109) with respect to u in direction δu and setting $(D_u y_j)\delta u \coloneqq \delta y_j$ and $(D_u u_j)\delta u = \delta u_j$. One finds

$$\begin{aligned} (\delta y_j,\varphi) + \tau_j(M_\delta(\nabla y_{j-1})\nabla\delta y_j,\nabla\varphi) &= \frac{\tau_j}{\varepsilon}(\delta u_j,\varphi) + (\delta y_{j-1},\varphi) - \tau_j(M'_\delta(\nabla y_{j-1})[\nabla\delta y_{j-1}]\nabla y_j,\nabla\varphi) \\ &\quad - \frac{\tau_j}{\varepsilon^2}(\psi''(y_{j-1})\delta y_{j-1},\varphi) \qquad j = 1,\dots,N, \end{aligned} \tag{2.117}$$

and $\delta y_0 = 0$.
For the additional adjoint equation, we analogously derive the adjoint equation with respect to

u in direction δu and in addition to the earlier replacements, we set $(D_u p_j)\delta u := \delta p_j$. Doing so, we obtain

$$\begin{aligned}(\delta p_N,\varphi)+\tau_N(M_\delta(\nabla y_{N-1})\nabla\delta p_N,\nabla\varphi) &= \frac{1}{\varepsilon}(\delta y_N,\varphi)-\tau_N(M'_\delta(\nabla y_{N-1})[\nabla\delta y_{N-1}]\nabla p_N,\nabla\varphi)\\ (\delta p_j,\varphi)+\tau_j(M_\delta(\nabla y_{j-1})\nabla\delta p_j,\nabla\varphi) &= (\delta p_{j+1},\varphi)-\tau_{j+1}(M''_\delta(\nabla y_j)[\nabla\varphi,\nabla\delta y_j]\nabla y_{j+1},\nabla p_{j+1})\\ &\quad -\tau_{j+1}(M'_\delta(\nabla y_j)[\nabla\varphi]\nabla y_{j+1},\nabla\delta p_{j+1})\\ &\quad -\tau_{j+1}(M'_\delta(\nabla y_j)[\nabla\varphi]\nabla\delta y_{j+1},\nabla p_{j+1})\\ &\quad -\tau_j(M'_\delta(\nabla y_{j-1})[\nabla\delta y_{j-1}]\nabla p_j,\nabla\varphi)\\ &\quad -\frac{\tau_{j+1}}{\varepsilon^2}(\psi'''(y_j)\delta y_j\, p_{j+1},\varphi)-\frac{\tau_{j+1}}{\varepsilon^2}(\psi''(y_j)\delta p_{j+1},\varphi)\\ &\qquad j=1,\dots,N.\end{aligned}\tag{2.118}$$

Note that there appears the 4-tensor M''_δ. Like M'_δ in general it is not symmetric. We will not give the expression for M''_δ explicitly, since it gives no new insights. In the implementation it is automatically determined by the FEniCS Form Compiler. Like M'_δ it possesses a divergence at $q=0$ in the case $\delta=0$ that now behaves like $\sim q^{-2}$ (as M is 0-homogeneous).
Equations (2.115), (2.117) and (2.118) mainly differ by the right-hand sides that contain terms arising from the chain rule. They consist of terms that contain among others products of up to four times the gradient of the appearing variables φ, y_j, p_j, δy_j and δp_j. For these terms being well defined, an affiliation to $H^1(\Omega)$ is not sufficient. This already is a sign that a rigorous treatment of the derivation even of eqs. (2.115) and (2.117) will be more involved. In order for the solutions to be better than $H^1(\Omega)$, one will in general need better regularity both for the right-hand side and the coefficient M'_δ. Since both depend on previous time step solutions, there are some further restricting requirements on the previous time step and so on. To obtain continuity or even Fréchet differentiability, these requirements might be even stronger. Eventually one typically ends up with a sequence of regularities that the solutions have to fulfill for the single time steps. Such an analysis is for example done in [133, Section 3.2] in order to obtain Fréchet differentiability that is required to rigorously set up the first order conditions. Similar to our problem, the previous time step appears in a coefficient on the left-hand side there, however not as a gradient. Also unique to our case, said gradient ∇y_{j-1} appears in the ellipticity coefficient M_δ which by its boundedness provides a mapping $L^s(\Omega)\to L^\infty(\Omega)$ for some $s\geq 1$. We will investigate below how this s has to be chosen to provide at least continuity. We will see that s can be chosen to be finite, but nevertheless the argument of M_δ has to be bounded in $L^\infty(\Omega)$ in the most convenient case. For Fréchet differentiability these conditions will only become stricter. We emphasize that $\nabla y_{j-1}\in L^\infty(\Omega)$ is a very strong requirement that involves much regularity on the data that lead to y_{j-1}.

2.7.3 On the continuous dependence of the state

As promised before, in this subsection we will supply a derivation of the continuity of the solution operator of the state equation with respect to the control variable. It will turn out that the main bottleneck is given by ∇y_{j-1} that appears as an argument for the coefficient M_δ. As already mentioned, to the best of our knowledge this requires that ∇y_{j-1} is bounded in $L^\infty(\Omega)$ to obtain a convenient result as it is the case in [137]. Also matching the regularities to apply the presented theorems in chain constitutes a serious problem. The latter reference is also the basis of this chapters discussion.
First let us simplify the appearance of the problem by writing (2.109) as

$$(M(\nabla y_{j-1})\nabla y_j,\nabla\varphi)+(y_j,\varphi)=(f(y_{j-1},u_j),\varphi),\tag{2.119}$$

with $f(y_{j-1},u_j)=y_{j-1}+u_j-\psi'(y_{j-1})$. Here, for presentational purposes, we set $\tau_j=\varepsilon=1$ and dropped the index δ, i.e., we expect M to fulfill the Assumption 2.7.3. Since $y_{j-1}\in H^1(\Omega)$, the right-hand side is an element of $L^2(\Omega)$ using the imbedding $H^1(\Omega)\hookrightarrow L^6(\Omega)$.
In this chapter we shall assume that Ω is regular in the sense of Gröger [66, Definition 2]. We

note that for the case of a square domain $[-1,1]^2$ this condition is fulfilled and also generally for bounded Lipschitz domains if we consider Neumann boundary conditions on the whole boundary as we do. We refer to [71, Section 5] for a proof of this fact and more information. Now the results from [66] yield:

$$\exists \bar{p} > 2: \quad y_j \in W^{1,p}(\Omega) \text{ solves (2.119)} \quad \text{and} \quad \|y_j\|_{W^{1,p}(\Omega)} \le C\|f\|_{(W^{1,p'})'} \quad \forall p \in [2,\bar{p}], \tag{2.120}$$

where $\frac{1}{p}+\frac{1}{p'}=1$ as usual. The number $\bar{p}$ is dependent on the domain and already for simple ones not easy to determine.

Essentially, the dependence on the control in (2.119) occurs at two places. One time explicitly on the right-hand side but on the other hand also implicitly through y_{j-1} on both sides, since the previous time step itself depends on the control as it satisfies a similar equation. We consider both dependencies—on the right-hand side and through M on the left-hand side—separately. For the former, we first observe that for fixed ∇y_{j-1} the equation simply has the form of a linear elliptic one and the continuous dependence of the solution in space on f in space immediately follows from a standard result of PDE theory. Furthermore, f depends continuously on the control and the solution from the previous time step due to the appearance of the Nemytskii operator ψ' (see (2.110) and below). Now we consider fixed right-hand side and look at the dependence on ∇y_{j-1} on the left-hand side. This case is a bit more complicated and restricting and will be discussed in the remainder of this section.

We first address the regularities needed to deal with continuity in the $H^1(\Omega)$-norm for one time step. We will see in the next theorem that ∇y_{j-1} needs to belong to a space better than $L^2(\Omega)$, so this theorem is not applicable to the previous time step anew. That means we have to think of different spaces to get a result for the whole system of equations. This motivates the more general treatment done afterwards. We also note that the required Lipschitz continuity holds for $M = M_\delta$ given in (2.106), as shown in Lemma 2.7.1.

Theorem 2.7.8. *Let Assumption 2.7.3 hold for M and in addition let M be Lipschitz continuous. Let $\bar{p} \ge p \ge 2$ with $\bar{p}$ from* (2.120) *and $y_{j-1}, \tilde{y}_{j-1} \in W^{1,s}(\Omega)$ with $s \in [\frac{2p}{p-2}, \infty]$. Denote by y_j and $\tilde{y}_j$ the solutions of*

$$(M(g)\nabla y, \nabla\varphi) + (y,\varphi) = \langle f, \varphi\rangle \qquad \forall \varphi \in H^1(\Omega), \tag{2.121}$$

with fixed right-hand side $f \in (W^{1,p'}(\Omega))'$ and coefficients $g = \nabla y_{j-1}$ and $g = \nabla \tilde{y}_{j-1}$, respectively.
Then there exists a constant C_f such that

$$\|y_j - \tilde{y}_j\|_{H^1(\Omega)} \le C_f \|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^s(\Omega)}, \tag{2.122}$$

i.e., the solution of the next step is Lipschitz continuously dependent on the gradient of the solution from the previous step as a mapping from $L^s(\Omega) \to H^1(\Omega)$.

The proof is based on [137, Theorem 2.12 (i)].

Proof: From simple manipulations one obtains

$$\begin{aligned}
&(M(\nabla y_{j-1})(\nabla\tilde{y}_j - \nabla y_j), \nabla\varphi) + (\tilde{y}_j - y_j, \varphi)\\
&\quad = (M(\nabla y_{j-1})\nabla\tilde{y}_j, \nabla\varphi) + (\tilde{y}_j,\varphi) - (M(\nabla y_{j-1})\nabla y_j, \nabla\varphi) - (y_j,\varphi)\\
&\quad = (M(\nabla y_{j-1})\nabla\tilde{y}_j, \nabla\varphi) + (\tilde{y}_j,\varphi) - (M(\nabla \tilde{y}_{j-1})\nabla \tilde{y}_j, \nabla\varphi) - (\tilde{y}_j,\varphi)\\
&\quad = (M(\nabla y_{j-1})\nabla\tilde{y}_j, \nabla\varphi) - (M(\nabla \tilde{y}_{j-1})\nabla \tilde{y}_j, \nabla\varphi)\\
&\quad = ([M(\nabla y_{j-1}) - M(\nabla\tilde{y}_{j-1})]\nabla\tilde{y}_j, \nabla\varphi),
\end{aligned} \tag{2.123}$$

where between the second and third line we used the fact that the right-hand side of (2.121) is fixed. Testing with $\tilde{y}_j - y_j$ this gives

$$(M(\nabla y_{j-1})(\nabla\tilde{y}_j - \nabla y_j), \nabla\tilde{y}_j - \nabla y_j) + (\tilde{y}_j - y_j, \tilde{y}_j - y_j) = ([M(\nabla y_{j-1}) - M(\nabla\tilde{y}_{j-1})]\nabla\tilde{y}_j, \nabla\tilde{y}_j - \nabla y_j).$$

The left-hand side can be estimated from below by

$$C\|\tilde{y}_j - y_j\|^2_{H^1(\Omega)} \leq (M(\nabla y_{j-1})(\nabla\tilde{y}_j - \nabla y_j), \nabla\tilde{y}_j - \nabla y_j) + (\tilde{y}_j - y_j, \tilde{y}_j - y_j),$$

using the uniformly positive definiteness of M. Invoking the Lipschitz continuity of M we obtain for the right-hand side

$$([M(\nabla y_{j-1}) - M(\nabla\tilde{y}_{j-1})]\nabla\tilde{y}_j, \nabla\tilde{y}_j - \nabla y_j) \leq C\|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^s(\Omega)}\|\tilde{y}_j\|_{W^{1,p}(\Omega)}\|\tilde{y}_j - y_j\|_{H^1(\Omega)}, \tag{2.124}$$

where we have used that $\frac{1}{p} + \frac{1}{2} + \frac{p-2}{2p} = 1$. The term $\|\tilde{y}_j\|_{W^{1,p}(\Omega)}$ can be estimated using (2.120) and is contributing the f-dependence to the constant. Combining the last three equations one obtains the desired inequality (2.122). □

Since the guaranteed regularity of the previous time step is also restricted by $\bar{p}$ from Gröger's result, we should assume $s \leq \bar{p}$. This leads to the requirement $\bar{p} \geq 4$ as can be seen from the following consideration: Assume $p < 4$, then it always holds $s > 4$, hence $\bar{p} \geq s > 4$. On the other side, to be able to choose $s \leq 4$, it has to hold $p \geq 4$ and hence $\bar{p} \geq p \geq 4$.
Unfortunately, it is not clear (if not even rather unlikely already for the most common cases) that this choice can be made, since the regularity result cited at the beginning of this section only assures the existence of *some* $\bar{p} > 2$ that supplies the index for the desired higher regularity. Also recall that this number is dependent on the domain Ω and no sufficient conditions for $\bar{p} \geq 4$ are known to the author.
Anyway, the just shown theorem cannot be applied in chain, since for this we would need the continuity of y_{j-1} in the $W^{1,s}$-norm, which is not covered by employing the same theorem to the previous time step. Therefore, the next theorem treats the continuity with respect to a more regular space. As a downside we now require L^∞-regularity of ∇y_{j-1}, to obtain a convenient result.

Theorem 2.7.9. *Let the previous assumptions hold. In addition, let* $2 < p < \min\{\bar{p}, 4\}$ *and* $s \in [\frac{2p}{p-2}, \infty]$. *Denote by* y_j *and* $\tilde{y}_j$ *the solutions of*

$$(M(g)\nabla y, \nabla\varphi) + (y, \varphi) = \langle f, \varphi\rangle \qquad \forall\varphi \in H^1(\Omega), \tag{2.125}$$

with fixed right-hand side $f \in (W^{1,p'}(\Omega))'$ *and coefficients* $g = \nabla y_{j-1}$ *and* $g = \nabla\tilde{y}_{j-1}$, *respectively. Furthermore, we require* $y_{j-1}, \tilde{y}_{j-1} \in Y_{ad}$, *where*

$$Y_{ad} := \left\{ y \in W^{1,s}(\Omega) \;\middle|\; |\nabla y| \leq \hat{C} \text{ almost everywhere in } \Omega \right\},$$

with $\hat{C} > 0$. *Then it holds*

$$\|y_j - \tilde{y}_j\|_{W^{1,p}(\Omega)} \to 0 \qquad \text{for} \qquad \|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^s(\Omega)} \to 0. \tag{2.126}$$

The proof is based on [137, Theorem 2.12 (ii)].

Proof: We recall from (2.123) that

$$(M(\nabla y_{j-1})(\nabla\tilde{y}_j - \nabla y_j), \nabla\varphi) + (\tilde{y}_j - y_j, \varphi) = ([M(\nabla y_{j-1}) - M(\nabla\tilde{y}_{j-1})]\nabla\tilde{y}_j, \nabla\varphi). \tag{2.127}$$

From this, $\tilde{y}_j - y_j$ can be interpreted as the weak solution for the right-hand side defined by

$$\langle F, \varphi\rangle := ([M(\nabla y_{j-1}) - M(\nabla\tilde{y}_{j-1})]\nabla\tilde{y}_j, \nabla\varphi) \qquad \forall\varphi \in H^1(\Omega). \tag{2.128}$$

The idea is now to apply Gröger's result (2.120) in order to obtain continuity in the space $W^{1,p}(\Omega)$. In what follows we therefore show that $F \in (W^{1,p'}(\Omega))'$ with $p' = \frac{p}{p-1}$. In contrast to (2.124) we need to assume less regularity for the last term at the expense of the first two terms.

To proceed with this, first let us choose

$$0 < \epsilon < \min\left\{\bar{p} - p, \frac{p^2 - 2p}{4 - p}\right\},$$

which is possible due to the requirements on p. We can apply Hölder's inequality and the Lipschitz continuity of M to arrive at

$$\begin{aligned} |\langle F, \varphi\rangle| &\le C\|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^{\frac{p(p+\epsilon)}{\epsilon}}(\Omega)} \|\tilde{y}_j\|_{W^{1,p+\epsilon}(\Omega)}\|\varphi\|_{W^{1,p'}(\Omega)} \\ &\le C\|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^{\frac{p(p+\epsilon)}{\epsilon}}(\Omega)} \|f\|_{(W^{1,\bar{p}'})'}\|\varphi\|_{W^{1,p'}(\Omega)}, \end{aligned} \tag{2.129}$$

for all $\varphi \in W^{1,p'}(\Omega)$. The center term was again estimated by Gröger's result. For the well definedness of the first term the assumed L^∞-regularity enters. We will now use the boundedness in the latter space to get rid of the explicit ϵ-dependence. Our assumption on ϵ yields that

$$\bar{s} := \frac{2p}{p-2} < \frac{p(p+\epsilon)}{\epsilon} \tag{2.130}$$

(this is the minimum possible exponent given from the previous theorem). We can now pull out some power of $\nabla y_{j-1} - \nabla\tilde{y}_{j-1} \in Y_{\mathrm{ad}}$ and absorb it into a constant due to the required pointwise almost everywhere boundedness and obtain

$$\|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^{\frac{p(p+\epsilon)}{\epsilon}}(\Omega)} \le C_\infty \|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^{\bar{s}}(\Omega)}^{\frac{\bar{s}\epsilon}{p(p+\epsilon)}}. \tag{2.131}$$

Since from what has been shown so far it follows $F \in (W^{1,p'})'$ as desired, we can now use (2.120) in (2.127) as announced above and obtain in total

$$\begin{aligned} \|y_j - \tilde{y}_j\|_{W^{1,p}(\Omega)} &\overset{(2.120)}{\le} C\|F\|_{(W^{1,p'})'} \overset{(2.129)}{\le} C_f\|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^{\frac{p(p+\epsilon)}{\epsilon}}(\Omega)} \\ &\overset{(2.131)}{\le} C_{f,\infty}\|\nabla y_{j-1} - \nabla\tilde{y}_{j-1}\|_{L^{\bar{s}}(\Omega)}^{\frac{\bar{s}\epsilon}{p(p+\epsilon)}}. \end{aligned} \tag{2.132}$$

From this the claim follows first for $\bar{s}$ and then by imbedding also for $s > \bar{s}$. □

Applying this theorem to our sequence of elliptic problems provides several problems. For the discussion, let us add a subscript j to the regularity indices of Theorem 2.7.9 that are related to the time step j we are considering in the following. First, in contrast to Theorem 2.7.8, we are now forced to set $p_j < 4$ and, as discussed before, this implies $s_j > 4$ yielding the same issues as already there. In addition, if we want to apply this theorem in chain, we now also need continuity of y_{j-1} with respect to $W^{1,s_j}(\Omega)$, i.e., $p_{j-1} \ge s_j$, which contradicts the requirement $p_{j-1} < 4$. As already mentioned, a further issue is the required regularity assumption on ∇y_j. From Gröger's result there is no hope to get $\nabla y_j \in L^\infty(\Omega)$. Note that the requirement is even stronger as we need to find a $\hat{C}$ such that the bound $\|\nabla\tilde{y}_{j-1}\|_{L^\infty(\Omega)} \le \hat{C}$ holds for all $\nabla\tilde{y}_{j-1}$ that are sufficiently close to ∇y_{j-1} in $L^s(\Omega)$. The author does not know any results that would yield the desired regularity, also having in mind that such results probably need better regularity of the data, i.e., also of y_{j-1} on the right-hand side, which satisfies a similar equation.

Remark 2.7.10. Note that in principle the restrictions $y_{j-1}, \tilde{y}_{j-1} \in Y_{\mathrm{ad}}$ and $p < 4$ could be avoided by concluding the proof after (2.129) (i.e. after the second inequality in (2.132)) and leaving ϵ a free parameter satisfying $0 < \epsilon < \bar{p} - p$. In (2.126) we will then require the convergence in $L^{\frac{p(p+\epsilon)}{\epsilon}}(\Omega)$ instead of $L^s(\Omega)$ and then also (2.129) is automatically well defined. However $\bar{p}$ will probably be close to 2 and hence ϵ small yielding $\frac{p(p+\epsilon)}{\epsilon} \gg 1$, so still (too) much regularity is needed. Further, applying the theorem in chain then requires $p_{j-1} \ge \frac{p_j(p_j+\epsilon)}{\epsilon}$, which applied recursively grows very strongly.

Ultimately, we want to recall that on all accounts the final goal is to apply an optimization algorithm to the problem and to discretize the resulting equations also in space. For this it is favorable to work in Hilbert spaces which we would be abandoning by above results. For Fréchet differentiability we can expect even stricter requirements. As the issues could not be solved satisfactorily already for the consideration of continuity, the rigorous treatment of the semi-implicit scheme will stop at this point. We will nevertheless discuss numerical results obtained from the purely formal considerations in Section 2.7.2 and compare them to those from the implicit scheme in Section 4.5.1.

3 Algorithms and implementation

The following part of this thesis is devoted to a discussion of different aspects that belong to the task of solving (1.8)–(1.9) numerically. First, in Section 3.1 we will motivate and present the algorithms that have been implemented for this. We will discuss the differences between approaches using the full and the reduced system and give an overview of the solvers and preconditioners that already exist in the literature. For our purposes, we will continue with the reduced approach. For this, we will introduce a steepest descent method with line search and a trust region Newton algorithm with the Steihaug-CG method in Sections 3.1.1 and 3.1.2. The latter is a globally convergent extension to ordinary Newton's method and due to its preferable convergence properties will almost exclusively be considered in the remainder of this thesis. In Section 3.1.3 we will enlighten the connection of the Steihaug method formulated in function space with a preconditioned algorithm in the Euclidean space after discretization. Due to this, we can conclude that the convergence properties of the reduced problem should already be mesh independent. As this does not seem to suffice for a reasonably fast solution process, a more general discussion about how to precondition the full and reduced systems is presented in Section 3.2. We will look at the dependency of the condition number on certain parameters and variables of the control problem. We will further work out the issues that come up when using state of the art preconditioning techniques. There exist more approaches for the full system here, but they only provide mesh independence and have not been observed to be faster than the pure Steihaug method in our experiments.
In Section 3.3 some concrete details, thoughts and technical aspects, that made its way into the code, will be given. How the quasilinear parabolic partial differential equations were solved using the FEniCS framework will be explained in Section 3.3.1. By the end of this subsection, we will present a way to speed up the Steihaug method by storing reusable quantities in the memory. The actual benefits of this implementation aspect will be demonstrated later in this thesis in Section 4.5.2. We will also explain how the data has to be stored in memory such that the parallelization capabilities of FEniCS can be harnessed, see Section 3.3.2. Finally, in Section 3.3.3, we will investigate to what extent using an adaptive mesh is reasonable for our control problem.

3.1 Presentation of the algorithms

In the following subsections we want to introduce the numerical methods that were implemented to compute a solution to the optimal control problem (1.8)–(1.9). Recall from Section 1.1—in particular Figure 1.1—that there are different ways to tackle the problem with regard to the discretization procedure. The same holds true also for the optimization algorithm to choose and the formulation of the problem we want to apply it to. As indicated in the section mentioned above, we will solve the first order condition for the reduced cost functional using second order information. However, we have not discussed there why we selected this approach. Before we start to treat the algorithm we implemented in more detail, we shall therefore become clear about what reflections our choice is based upon. In principle there are two formulations of the problem that we could build on for optimization—the reduced and the unreduced problem. To illustrate this, let us consider the abstract control problem

$$\min J(y,u) \qquad \text{subject to} \qquad A(y) = u, \tag{3.1}$$

with possibly nonlinear state equation. Recall that if the state equation is uniquely solvable, we can define the reduced cost functional

$$j(u) \coloneqq J(y(u),u). \tag{3.2}$$

In Section 1.1 we showed that then the first order optimality condition for the cost functional (1.8) reads as

$$\nabla j(u) = \tfrac{\lambda}{\epsilon} u + p \overset{!}{=} 0, \tag{3.3}$$

where the variable p is given by the solution of the adjoint equation. In principle the gradient is already sufficient to implement a steepest descent method (see the following section), but this is known to typically possess a slow convergence behavior. Therefore it makes sense to implement a Newton-like method that utilizes second order information. For problem (1.8)–(1.9), the Hessian is of the form

$$\nabla^2 j(u)\delta u = \tfrac{\lambda}{\varepsilon}\delta u + \delta p = \left(\tfrac{\lambda}{\varepsilon} I + \mathcal{K}(y)^{-*}\mathcal{P}(y,p)\mathcal{K}(y)^{-1}\right)\delta u, \tag{3.4}$$

where $\mathcal{K}(y)$ is the operator associated to the linearized state equation (1.26), i.e. (for the smooth double-well potential (2.4))

$$\mathcal{K}(y) = \varepsilon\partial_t - \varepsilon\nabla\cdot(A''(\nabla y)\nabla\cdot) + \tfrac{1}{\varepsilon}(3y^2-1), \tag{3.5}$$

and $\mathcal{K}^*(y)$ given by

$$\mathcal{K}^*(y) = -\varepsilon\partial_t - \varepsilon\nabla\cdot(A''(\nabla y)\nabla\cdot) + \tfrac{1}{\varepsilon}(3y^2-1) \tag{3.6}$$

is associated to the additional adjoint equation (1.27).
Then $w = \mathcal{K}(y)^{-1}v$ is given by the solution of

$$\mathcal{K}(y)w = v \;\text{ in } Q, \quad \partial_{\nu_A} w = 0 \;\text{ on } \Sigma, \quad w(0) = 0 \;\text{ in } \Omega, \tag{3.7}$$

and $w = \mathcal{K}(y)^{-*}v$ by

$$\mathcal{K}(y)^*w = v \;\text{ in } Q, \quad \partial_{\nu_A} w = 0 \;\text{ on } \Sigma, \quad w(T) = \tfrac{1}{\varepsilon}v(T) \;\text{ in } \Omega. \tag{3.8}$$

The operator $\mathcal{P}(y,p)$ converts the solution $\delta y = \mathcal{K}(y)^{-1}\delta u$ into the right-hand side of the additional adjoint equation by

$$\mathcal{P}(y,p)\delta y = -\tfrac{6}{\varepsilon}yp\delta y + \varepsilon\nabla\cdot(A'''(\nabla y)[\nabla p, \nabla\delta y]). \tag{3.9}$$

In the subsequent discussion we will drop the arguments of $\mathcal{K}(y)$, $\mathcal{K}^*(y)$ and $\mathcal{P}(y,p)$ for the sake of readability. Working with (3.4) is known as the reduced Hessian approach (see, e.g., [46] and the references therein). For later convenience, let us mention that in discretized form the reduced Hessian can be written as

$$\nabla^2 j(u) = \left(\tfrac{\lambda}{\varepsilon}M + MK^{-T}PK^{-1}M\right), \tag{3.10}$$

where M shall be a mass matrix and K and P are discretized versions of $\mathcal{K}$ and $\mathcal{P}$.
Often it suffices to find a good approximation of the Hessian, but only solving the appearing PDEs inexactly might not be adequate. The reason is that it is unclear how an outer iterative solver for the whole operator in (3.4) behaves if its application varies slightly among iterations due to uncertainties. Literature about the influence of such kind of inexactness in inner iterations is sparse as this is still an active topic of research (see, e.g., [76, 46, 47, 121]). To apply the Hessian to a vector δu one is therefore more or less obliged to solve two linear parabolic equations *exactly* (or rather to high accuracy). This can be considered as a big disadvantage of this approach, since the application of the Hessian typically has to be computed numerous times which due to above is very costly. On the other hand, globalization does not comprise many problems with this strategy as there exist well-established methods like the trust region Newton method we will present later. Globalization is necessary, since in some cases $\mathcal{P}$ might be singular (e.g. if $p \equiv 0$) or have negative eigenvalues and therefore the operator in (3.4) might not be positive definite.
Another topic we would want to touch on is the accessibility to preconditioning techniques, see Section 3.2. First let us comment that, as will be discussed in Section 3.2.2 (see also [96]), (3.4) should already be mesh independent by itself, which would eliminate the main reason for preconditioning in the context of PDEs. On the other hand, if there nonetheless arises the need for preconditioning due to other sources of bad condition, the structure of (3.4) provides more issues than for the unreduced problem that will be considered next (see also Section 3.2.3).

In contrast to (3.3), one could also try to find a saddle point of the Lagrangian defined in (1.13), which under some conditions provides a local minimum of the original task. In this case the first order necessary condition is given by

$$\nabla L(y,u,p) = \nabla[J(y,u) - (A(y) - u, p)] = \begin{pmatrix} J_y(y,u) - A_y(y)^* p \\ J_u(y,u) + p \\ -A(y) + u \end{pmatrix} \overset{!}{=} 0. \tag{3.11}$$

Also here it is preferable to invoke second order information. For instance, in order to apply Newton's method to this problem, in each step one has to solve

$$\nabla^2 L(y,u,p)(\delta y, \delta u, \delta p) = -\nabla L(y,u,p),$$

which in matrix form reads as

$$\begin{pmatrix} \mathcal{P} & 0 & -\mathcal{K}^* \\ 0 & \frac{\lambda}{\varepsilon} I & I \\ -\mathcal{K} & I & 0 \end{pmatrix} \begin{pmatrix} \delta y \\ \delta u \\ \delta p \end{pmatrix} = - \begin{pmatrix} L_y \\ L_u \\ L_p \end{pmatrix}. \tag{3.12}$$

Note that the operators $\mathcal{K}$ and $\mathcal{P}$ are the same as appearing in (3.4). In fact, the relation between eqs. (3.4) and (3.12) becomes apparent by determining the Schur complement of (3.12) with respect to δu. Doing this, one obtains

$$\mathcal{S}\delta u = \Big(\tfrac{\lambda}{\varepsilon} I - \begin{pmatrix} 0 & I \end{pmatrix} \underbrace{\begin{pmatrix} \mathcal{P} & -\mathcal{K}^* \\ -\mathcal{K} & 0 \end{pmatrix}^{-1}}_{\begin{pmatrix} \star & \star \\ \star & -\mathcal{K}^{-*}\mathcal{P}\mathcal{K}^{-1} \end{pmatrix}} \begin{pmatrix} 0 \\ I \end{pmatrix} \Big) \delta u = \big(\tfrac{\lambda}{\varepsilon} I + I \mathcal{K}^{-*}\mathcal{P}\mathcal{K}^{-1} I \big)\, \delta u,$$

where we used a formula for the inversion of 2×2 block matrices, see, e.g., [94]. Note that for computing the gradient of L in (3.11), neither a state nor an adjoint equation has to be solved, but only applied. All variables are updated by the step sizes determined in (3.12). Hence the main effort lies in solving (3.12) in each Newton step. Here one has much more freedom concerning the tolerance in case of an iterative solver, since the variables δy, δu, δp no longer depend on each other, but are rather computed together and no inner iterations appear as before. In contrast to the other approach, in each iteration y and p do not necessarily fulfill the state or adjoint equation since otherwise $L_y = L_p = 0$ would have to hold in every step. As by Newton's method we seek a root of ∇L this is only the case for the solutions found in the end. However, if the state and adjoint equation were fulfilled, due to the vanishing of the respective derivatives of L the system (3.12) would reduce to a Newton system for $j(u)$ after taking the

Schur complement also on the right-hand side.
The unreduced approach also has disadvantages. On the one side, due to the lack of related results, one can no longer expect the condition number of the system (3.12) to be mesh independent. This means that preconditioning becomes important. In the literature, preconditioning a block system of that form has successfully been studied in [21, 124, 112], but most of the considered systems only involve linear state equations like the Laplace equation, heat equation or alike. In the context of the Allen-Cahn equation only the paper [20] is known to the author. On the other side, for nonlinear problems a globally convergent method is required, since one is not guaranteed to start with a positive definite system matrix. Here one should note that as far the author knows, literature on the just mentioned preconditioning does not consider globalization strategies, but rather relies on the initial value to be suitably close to the solution. One common approach for a globalization of the unreduced problem (3.1) is the trust region SQP method [43]. Here one applies the trust region method to the optimization subproblems that have to be solved in each iteration of the SQP method. The trust region subproblems that arise there in context of optimal control are well suited to be approximated by splitting the iterates into a normal and a tangential step [73].
As we have seen, both the reduced as well as the unreduced problem have their advantages and disadvantages. For this thesis, we have decided to apply a trust region method to the reduced system, as for the given structure the trust region subproblem can be solved efficiently by the Steihaug-CG method. In context of the Allen-Cahn equation this ansatz has shown promising results [24]. Finally the mesh independence is already provided, whereby the need for further preconditioning due to other reasons cannot be excluded.

3.1.1 Steepest descent and trust region method

As discussed introductory, in the following we will consider the reduced problem. Hence in this section, let us generally consider an unconstrained optimization problem of the following form

$$\min_{u \in U} j(u), \tag{3.13}$$

where $j : U \to \mathbb{R}$ and U denotes some Hilbert space. By virtue of the Riesz representation theorem, we identify the gradient $\nabla j(u)$ with an element of U. In case of the optimal control of the anisotropic Allen-Cahn equation we have $U = L^2(Q)$. In general, numerically we are only able to find local minima by solving for the first order condition, that is, for a local minimizer $u^* \in U$ it holds

$$\nabla j(u^*) = 0. \tag{3.14}$$

In this section we treat two algorithms: the steepest descent method and the trust region method. Both methods are based on solving the problem iteratively by starting at the current iterate u_k with function value $j(u_k)$ and finding a direction δu_k such that the next iterate $u_{k+1} \coloneqq u_k + \delta u_k$ fulfills $j(u_{k+1}) < j(u_k)$. For the stopping criteria they use (3.14), i.e., the norm of $\nabla j(u_{k+1})$ is checked against a tolerance. In principle, both methods differ only by the choice of δu_k, which however has strong impact on the computational cost of one step as well as the convergence properties of the whole algorithm.

Steepest descent method

Let us start with discussing the steepest descent method. Here one makes use of the fact that the negative gradient $-\nabla j(u)$ points into the direction of the steepest descent. So one simply takes $\delta u_k = -\alpha \nabla j(u_k)$ with some $\alpha > 0$. The scaling with α is necessary as the descent is only guaranteed to be local and the function might increase again if going to far. On the other hand α must not be chosen too small since the method might not converge then. To determine α, there are several conditions and algorithms one can prescribe. One simple rule to demand is the so called Armijo condition, which in case of the steepest descent method reads as

$$j(u_k + \alpha \nabla j(u_k)) \leq j(u_k) - c\alpha \|\nabla j(u_k)\|_U^2, \tag{3.15}$$

with some $c \in (0,1)$. If one reduces α successively by a constant factor, i.e., $\alpha_i = \kappa^i \alpha_0$ with $\kappa \in (0,1)$, one can show that this conditions is fulfilled after a finite number of iterates. For more information about the choice of α and convergence results we refer to [104] in the finite dimensional case and [79] on Banach spaces. Algorithm 1 provides a pseudo code of the just described method. The numerical effort per step lies in evaluating $\nabla j(u_k)$ in line 3 as well as evaluating $j(u_k + \alpha_i \nabla j(u_k))$ in line 9 several times. In terms of PDE solves per iteration the latter requires the evaluation of a nonlinear parabolic PDE (the state equation) and the former the evaluation of a linear parabolic PDE (the adjoint equation), since starting with the second iteration over k, the state equation that is required to evaluate $\nabla j(u_k)$ is already known from line 9 in the last iteration.

Algorithm 1 steepest descent method with line search

```
Input: initial control $u_0$
Output: local minimizer $u$ of $j$
 1: $k := 0$
 2: while $k \leq k_{\max}$ do
 3:    determine search direction $\delta u_k := -\nabla j(u_k)$
 4:    if $\|\nabla j(u_k)\|_{L^2(Q)} \leq$ tol then
 5:       return
 6:    end if
 7:    while $i \leq i_{\max}$ do
 8:       compute $j(u_k + \alpha_i \nabla j(u_k))$
 9:       if $j(u_k + \alpha_i \nabla j(u_k)) < j(u_k) - c\alpha_i \|\nabla j(u_k)\|_U^2$ then
10:          $\alpha := \alpha_i$
11:          break
12:       end if
13:       $\alpha_{i+1} := \kappa \alpha_i$
14:       $i := i + 1$
15:    end while
16:    $u_{k+1} := u_k + \alpha \delta u_k$
17:    $k := k + 1$
18: end while
```

Trust region method

As is common knowledge and as we will later also verify for our case (see Figure 4.1), the convergence properties of the steepest descent method are not optimal. Efforts were therefore made to improve the performance by finding a better search direction than $\nabla j(u)$. As long as it is a descent direction, that is it satisfies

$$(\delta u, \nabla j(u))_U < 0, \tag{3.16}$$

the value of j will decrease in the next iterate provided the step size is sufficiently small. To find such a direction, one usually starts with the gradient and modifies it in an appropriate manner. For example the prescription

$$\delta u := -B \nabla j(u)$$

fulfills (3.16) if the operator $B : U \to U$ is positive definite. Typically one sets $B = [\nabla^2 j(u)]^{-1}$ or some reasonable approximation of it. By choosing the Hessian of j one obtains the so-called Newton's method. In the vicinity of a local optimizer the second order sufficient condition ensures that $\nabla^2 j(u)$ is positive definite and so δu is a descent direction. It can be shown to locally converge superlinearly or even quadratic if j is sufficiently smooth. Again we refer to [104, 79] for further information.

The last statement already hinted towards a problem with Newton's method: we can only expect the convergence properties to hold locally. If we start too far away from the optimizer, then we are not even guaranteed to obtain a descent direction at all as the Hessian could be no longer positive definite. To remedy this issue one has to apply a globalization strategy. One common approach for this is the so-called class of trust region methods. The idea goes back to the obervation that one step of Newton's method is equivalent to solving the quadratically approximated problem

$$\min_{\delta u \in U} m(\delta u) := j(u) + (\nabla j(u), \delta u)_U + \tfrac{1}{2}(\nabla^2 j(u)\delta u, \delta u)_U, \tag{3.17}$$

which can be checked by writing out the first order necessary (and in this case also sufficient) conditions. Since m is just an approximation of the function j around a specific point, one can 'trust' it to be sufficiently similar only in some small environment. Therefore, instead one can propose to consider solving the problem

$$\inf_{\|\delta u\|_U \leq \sigma} m(\delta u), \tag{3.18}$$

where $\sigma > 0$ is sufficiently small. In principle also other choices than the ball of radius σ are possible, but we will stick to this most common choice here. Note that now the problem is also well defined for indefinite $\nabla^2 j(u)$. Since in infinite dimensional spaces the unit ball is not compact, we had to replace the minimum by an infimum in this case. This means that we might not be able to find a δu such that the infimum is attained, but in practice this is irrelevant, since we only seek an approximate solution of problem (3.18), i.e., we seek a δu such that $m(\delta u)$ is close to the infimal value. In each iteration the trust region method updates the current iterate by δu given as the approximate solution of (3.18). The main concern is now to choose and adapt the trust region radius σ in each iteration. How it is altered depends on how good j is approximated by m in the region specified by the current value of σ. To estimate this one can consider

$$\rho := \frac{j(u + \delta u) - j(u)}{m(\delta u) - m(0)}, \tag{3.19}$$

which should be close to the value 1 if the approximation is good. Based on the value of ρ there are different updating strategies for the radius σ. Our choice is given in Algorithm 2, for other ones we refer to [104, 43]. In principle one should increase the radius if the approximation is good and decrease it otherwise. It can be shown that being near the optimizer, the radius of the trust region eventually becomes inactive and Newton's method is recovered in the final few steps. The only question that is still open is how to solve the trust region subproblem in line 8. Since this can be very costly, one usually only approximates the solution, which will be the subject of the next chapter. Note that the first term in (3.17) does not need to be computed as it is just a constant that does not affect the location of the minimum.

We conclude this section by a discussion of the numerical effort of Algorithm 2. The trust region subproblem in line 8 contributes strongly to this, but also depends on how it is actually approximated. Further, we have to consider the function evaluations. In line 3 we have to compute the gradient of j that corresponds to the evaluation of the nonlinear state equation and the adjoint equation. Like before the solution of the state equation is known at this point from the second iteration onwards, since it is also needed in line 9. Further the Hessian or rather its application to δu is needed in line 10 which requires the solution of the linearized state equation and the additional adjoint equation. In contrast to the gradient the application of the Hessian and j itself have to be recomputed each time σ is adapted. Depending on its nature, the approximation in line 8 might be recomputed in a cheaper way then.

3.1.2 The Steihaug-CG method

As promised we will now address the problem of determining a solution to the trust region subproblem in line 8 of Algorithm 2. We already pointed out in the last section that an exact solution in general is too costly and also not practical since in infinite dimensions the infimum will not always be attained. Therefore the solution will only be approximated. Before we do so,

Algorithm 2 trust region Newton method

Input: initial control u_0, trust region radius σ, $\kappa_1 > 1$, $\kappa_2 \in (0,1)$, $\eta \in [0, \frac{1}{2})$, $\sigma_{\max}$
Output: local minimizer u of j

```
1: $k := 0$
2: while $k \leq k_{\max}$ do
3:   compute $\nabla j(u_k)$
4:   if $\|\nabla j(u_k)\|_U \leq \mathrm{tol}$ or $\|\nabla j(u_k)\|_U \leq \mathrm{reltol}\|\nabla j(u_0)\|_U$ then
5:     return
6:   end if
7:   repeat
8:     approximate $\inf_{\|\delta u\|_U \leq \sigma} m(\delta u)$
9:     compute $j(u_k + \delta u) = J(S(u_k + \delta u), u_k + \delta u)$
10:    compute $m(\delta u) - m(0) = (\nabla j(u_k), \delta u)_U + \frac{1}{2}(\nabla^2 j(u_k)\delta u, \delta u)_U$
11:    determine $\rho := \frac{j(u_k+\delta u) - j(u_k)}{m(\delta u) - m(0)}$
12:    if $\rho \approx 1$ then
13:      $\sigma := \min\{\kappa_1 \sigma, \sigma_{\max}\}$
14:    else
15:      $\sigma := \kappa_2 \sigma$
16:    end if
17:    if $\rho \geq \eta$ then
18:      accepted := true
19:    else
20:      accepted := false
21:    end if
22:  until accepted
23:  $u_{k+1} := u_k + \delta u$
24:  $k := k + 1$
25: end while
```

let us reflect a moment on how the exact solution of (3.18) does look like if the minimum exists. If $\nabla^2 j(u)$ is positive definite, then there exists also a global minimum of the unconstrained model problem and there are two possibilities. Either the unconstrained minimizer lies outside or inside the trust region. If it lies inside, the minimizer of the constrained problem will coincide and can be computed by $[\nabla^2 j(u)]^{-1}\nabla j(u)$. Otherwise the minimizer will lie on the boundary of the trust region and the exact determination would be more involved. In case the Hessian has a negative eigenvalue, the minimizer may lie on the boundary of the trust region. Considering the minimizer as a function of the boundary radius σ, one typically observes that it describes an arc-like shape starting at the center of the trust region and terminating at the unconstrained minimizer, at least if the Hessian is positive definite. Typical approaches for approximating the solution of the trust region subproblem therefore try to approximate this path by an easier version, where the intersection with the boundary can be computed by a reasonable computational effort. Also the Steihaug-CG method [123] we ultimately want to present in this chapter can be understood as an approximation of the arc by iterates of the CG method.

Before we go on let us quickly introduce the notion of the Cauchy point which is important for the convergence properties of the trust region method. The Cauchy point is defined as the minimizer of the model function (3.17) on the set $\{-\alpha\nabla j(u) \mid \alpha \geq 0\} \cap B_\sigma(0)$ and can be given explicitly by

$$p^C := -\tau\sigma \frac{\nabla j(u)}{\|\nabla j(u)\|_U}, \tag{3.20}$$

where

$$\tau = \begin{cases} 1 & \text{if } (\nabla^2 j(u)\nabla j(u), \nabla j(u))_U \leq 0, \\ \min\left(\frac{\|\nabla j(u)\|_U^3}{\sigma(\nabla^2 j(u)\nabla j(u), \nabla j(u))_U}, 1\right) & \text{else.} \end{cases}$$

It is either given by the unconstrained minimizer ($\tau < 1$) or lies on the boundary of the trust region, which is the case if the unconstrained minimizer is to large or the curvature in direction $\nabla j(u)$ is negative. The importance of the Cauchy point lies in the fact that it provides a condition on the sufficient reduction per step. That is because one can show that the trust region algorithm converges globally if its steps δu_k provide a reduction in the model function that is at least some fixed positive multiple of the decrease obtained by the Cauchy step p^C.

Of course only taking the Cauchy point as next iterate wouldn't yield an efficient algorithm since one effectively falls back to the steepest descent method with a certain step size then. Therefore one should improve the Cauchy point. Common ideas are the dogleg method or double-dogleg method, where one approximates the above described arc by line segments between the origin, the Cauchy point, dependingly a third intermediate point and the unconstrained solution. Another idea is the two-dimensional subspace minimization where one minimizes the model function over the subspace spanned by $\nabla j(u)$ and $[\nabla^2 j(u)]^{-1}\nabla j(u)$. For a more detailed discussion on these briefly mentioned ideas we refer to [104]. All these methods have in common that they require an (approximate) solution of $[\nabla^2 j(u)]^{-1}\nabla j(u)$. Since the Hessian typically is high dimensional, this might be very expensive—also with an iterative method. Another problem is the fact that both methods require the Hessian to be positive definite, since otherwise the computed "unconstrained minimizer" would not be valid. Since this cannot easily be checked in general, one can only apply these methods in regions where one knows that the Hessian is positive definite or use an approximation where one has this property. In the subspace minimization one can also add a multiple of the identity to the Hessian but nevertheless the multiplication constant still has to be estimated.

These problems are elegantly faced by the Steihaug-CG method [123]. The idea is that during solving $[\nabla^2 j(u)]^{-1}\nabla j(u)$ iteratively (by the CG method) one checks at each step if the curvature in direction of the current iterate is positive and if one is still inside the trust region. Otherwise the algorithm is aborted with a proposed step obtained from the current iterate. Therefore, one only needs to fully iterate in the vicinity of the final solution, when the trust region radius is inactive. A full pseudo code of the Steihaug method is given in Algorithm 3. Apart from the lines following the if-conditions in line 5 and 11 one recognizes the regular CG method. For instance in lines 9 and 10 the solution of the line search of the model problem (3.17) in the direction of d_i is computed and in lines 16 and 17 the previous search direction is projected out

Algorithm 3 Steihaug-CG

```
Input: control $u$, gradient $\nabla j(u)$, trust region radius $\sigma$
Output: the solution $\delta u$ of $\min_{\|\delta u\|_U \leq \sigma} m(\delta u)$
1: $\delta u_0 := 0,\ r_0 := \nabla j(u),\ d_0 := -\nabla j(u)$
2: $i := 0$
3: while $i \leq i_{\max}$ do
4:     compute $\nabla^2 j(u) d_i$
5:     if $(d_i, \nabla^2 j(u) d_i)_U \leq 0$ then
6:         $\delta u_{i+1} := \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_{i+1}\|_U = \sigma$
7:         return $\delta u_{i+1}$
8:     end if
9:     $\alpha_i := (r_i, r_i)_U / (d_i, \nabla^2 j(u) d_i)_U$
10:    $\delta u_{i+1} := \delta u_i + \alpha_i d_i$
11:    if $\|\delta u_{i+1}\|_U \geq \sigma$ then
12:        $\delta u_{i+1} := \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_i + \tau d_i\|_U = \sigma$
13:        return $\delta u_{i+1}$
14:    end if
15:    $r_{i+1} := r_i + \alpha_i \nabla^2 j(u) d_i$
16:    $\beta_{i+1} := (r_{i+1}, r_{i+1})_U / (r_i, r_i)_U$
17:    $d_{i+1} := -r_{i+1} + \beta_{i+1} d_i$
18:    if $\|r_{i+1}\|_U <$ tol or $\|r_{i+1}\|_U <$ reltol$\|r_0\|_U$ then
19:        return $\delta u_{i+1}$
20:    end if
21:    $i := i + 1$
22: end while
```

of the residual such that the next direction is orthogonal to the previous ones with respect to the scalar product induced by $\nabla^2 j(u)$. As these steps only make sense if the matrix is positive definite, it is checked in line 5 if d_i is a direction with negative curvature and the algorithm is stopped in this case. After having updated the proposal for the next iterate of the trust region Newton method in line 10 it is checked in line 11 if this proposal surpasses the trust region radius and if that is the case the algorithm is also terminated. In both of the just described early stopping scenarios the current search direction is pursued until the boundary. This makes sense vividly in both cases. In case of negative curvature one knows that the model function can only get smaller in direction d_i, so the solution should lie on the boundary. In the second case taking the intersection with the boundary of the line between current and next iterate is an evident choice. If neither of the just discussed cases occur, the Steihaug-CG method reduces to the standard CG method and it iterates out until $[\nabla^2 j(u)]^{-1} \nabla j(u)$ is solved with desired accuracy.
Apart from the two modifications discussed above there is also another minor difference to the CG method. Note that in line 1 the initial value is constrained to be $\delta u_0 = 0$. This is important in order for the method to produce reasonable iterates in context of the trust region problem. For instance, consider the first iterate. If neither of lines 5 or 11 apply, it is given by

$$\delta u_1 = \delta u_0 + \alpha_0 d_0 = 0 + \tfrac{(r_0, r_0)_U}{(d_0, \nabla^2 j(u) d_0)_U} d_0 = -\tfrac{(\nabla j(u), \nabla j(u))_U}{(\nabla j(u), \nabla^2 j(u) \nabla j(u))_U} \nabla j(u).$$

Otherwise, one obtains, since $\|\delta u_1\| = \sigma$ has to hold that

$$\delta u_1 = -\sigma \tfrac{\nabla j(u)}{\|\nabla j(u)\|_U}.$$

Taken together one recognizes that this exactly produces the Cauchy point defined in (3.20). This means that we have sufficient decrease as it holds $\Delta m(\delta u^{\mathrm{ST}}) \geq \Delta m(p^C)$ (with $\Delta m(p) := m(0) - m(p)$ and δu^{ST} is the solution returned by the Steihaug method) which follows from the next theorem.

Theorem 3.1.1. *The sequence of iterates $\{\delta u_i\}_{i=0,\ldots,n}$ produced by Algorithm 3 satisfies*

$$m(\delta u_0) > \ldots > m(\delta u_i) > m(\delta u_{i+1}) > \ldots > m(\delta u_{n-1}) \geq m(\delta u_n),$$

i.e., the model function is strictly decreasing apart from possibly the last iterate.
Proof: See [123, Theorem 2.1], where the steps also hold true for the scalar product on U. □
If the Hessian is positive definite, one can even show that $\Delta m(\delta u^{\mathrm{ST}}) \geq \frac{1}{2}\Delta m(\delta u^*)$, where δu^* is the exact solution of (3.18) (see [43, Theorem 7.5.9]).
The next theorem justifies the termination of the algorithm after line 11.

Theorem 3.1.2. *Except for the last one, the iterates $\{\delta u_i\}_{i=0,\ldots,n}$ are independent of σ. Furthermore it holds $\|\delta u_i\|_U < \|\delta u_{i+1}\|_U$ for all iterates.*

Proof: This is shown in [104, Theorem 7.3] or also in [123, Theorem 2.1], where the steps of the proof also work with the scalar product on U. We point out that the choice $\delta u_0 = 0$ enters here. □
As the norm of the iterates only becomes larger the longer the algorithm iterates, it makes sense to abort the algorithm as soon as the trust region radius is reached since the iterates won't reenter the trust region. Furthermore note that after shrinking the trust region radius without accepting the step (i.e. $\rho < \eta < \frac{1}{2} \not\approx 1$ in Algorithm 2), the result of this theorem implies that the iterates do not have to be recomputed except for the last one given by the intersection with the boundary.
We note that the solution to $\|\delta u_i + \tau d_i\|_U = \sigma$ with $\tau > 0$ from lines 5 and 11 is given by

$$\tau^2\|d_i\|_U^2 + 2\tau(\delta u_i, d_i)_U + \left(\|\delta u_i\|_U^2 - \sigma^2\right) = 0, \quad \tau > 0,$$

or equivalently

$$\tau = \frac{-(\delta u_i,d_i)_U + \sqrt{(\delta u_i,d_i)_U^2 + \|d_i\|_U^2\left(\sigma^2 - \|\delta u_i\|_U^2\right)}}{\|d_i\|_U^2}.$$

The appearing norms and scalar products can be determined efficiently by the recurrence relations

$$\|d_{i+1}\|_U^2 = \|r_{i+1}\|_U^2 + \beta_{i+1}^2\|d_i\|_U^2$$

following from line 17 and the relation $(r_{i+1}, d_i)_U = 0$ known from the ordinary CG method [75, Theorem 4.1], and

$$(\delta u_{i+1}, d_{i+1})_U = (\delta u_i + \alpha_i d_i, -r_{i+1} + \beta_{i+1} d_i)_U = \beta_{i+1}\left((\delta u_i, d_i)_U + \alpha_i\|d_i\|_U^2\right),$$

using again the previous relation and the fact that due to $\delta u_0 = 0$ the vector δu_{i+1} is a linear combination of the directions d_i. The appearing norms and scalar products are either known directly from the algorithm or from the previous recursion step.
Finally let us turn back to the discussion of the numerical effort from the end of the last section, now that we have presented the ansatz we will use to approximate the trust region subproblem. Also for the Steihaug-CG method the main effort lies in solving PDEs and with that the computation of scalar products and norms in general can be neglected. Let us nevertheless briefly comment that like in the ordinary CG method in principle one gets along by computing the scalar products $(r_i, r_i)_U$ and $(d_i, \nabla^2 j(u) d_i)_U$ once per step. But as just indicated the main effort lies rather in computing the application of the Hessian $\nabla^2 j(u) d_i$ appearing in the last scalar product. In the case of optimal control, for this one has to solve two linear parabolic PDEs—the linearized state equation and the additional adjoint equation—that themselves depend on the state and adjoint state. Note however that the latter two are known from the trust region algorithm and are fixed during the whole Steihaug method, so they do not need to be recomputed. As also mentioned previously, recomputing the solution to the model problem after having shrunk the trust region radius can be executed in a much cheaper way, as the directions d_i (and their norms indicating excession of the boundary) are already known. So one only has to solve for the crossing with the boundary which as discussed above can be computed in a cheap manner. The total number of iterates performed by the Steihaug-CG method also

has a strong influence on the effort one trust region step takes. However they are not known a priori and depend on whether the algorithm stops earlier or not. As one expects from the intuition of the trust region method and as we will also see in the numerical section later, the Steihaug method iterates longer towards the end of the algorithm. When the trust region radius is inactive we observed that the Steihaug method iterates until convergence or stops due to exceeding $i_{\max}$.

3.1.3 A note on the preconditioned Steihaug-CG method

In what follows we want to discuss a preconditioned formulation of the Steihaug-CG method 3. We will also see that by discretizing the algorithm one will actually end up with a preconditioned method formulated in $\mathbb{R}^n$, where n is the number of the degrees of freedom after discretization. This indicates that the algorithm is mesh independent. Further the different viewpoints allow to find an implementation that saves a matrix multiplication.
Preconditioning can be viewed as a change of the underlying scalar product. To this end we take a positive definite operator $C : U \to U$ and define the new scalar product by

$$(u, v)_C := (u, Cv)_U \qquad \forall u, v \in U. \tag{3.21}$$

In terms of this scalar product we can reformulate the trust region subproblem (3.18) as

$$\min_{\delta u \in U, \|\delta u\|_C \leq \sigma} m(\delta u) := (\nabla j(u), \delta u)_U + \tfrac{1}{2}(\nabla^2 j(u)\delta u, \delta u)_U = (\nabla_C j(u), \delta u)_C + \tfrac{1}{2}(\nabla_C^2 j(u)\delta u, \delta u)_C. \tag{3.22}$$

Note that we freely omitted the constant $j(u)$ not affecting the minimizer of the model function. The only difference to (3.18) is that the trust region is now given in terms of the norm induced by C. The model function m is identical apart from now being formulated in terms of the C-scalar product using the adapted quantities

$$\begin{aligned} \nabla_C j(u) &= C^{-1}\nabla j(u), \\ \nabla_C^2 j(u) &= C^{-1}\nabla^2 j(u). \end{aligned} \tag{3.23}$$

The new Hessian $\nabla_C^2 j(u)$ is now symmetric with respect to the C-scalar product. Using these quantities one can simply translate the Algorithm 3 by using the C-scalar product and norm instead of the U counterparts and using the quantities h_i belonging to $\nabla_C j(u)$ instead of r_i. The resulting method is presented in Algorithm 4. Note that the steps in this algorithm are identical to those in Algorithm 3 as can easily be verified line by line. The only difference lies in the stopping criterion where we now use a quantity that is computed in the course of this version of the algorithm.
Often one does not want to work with the transformed problem but rather with the former quantities. Using relations (3.23) and defining the new vectors $h_i := C^{-1}r_i$ one obtains the standard form of the preconditioned Steihaug-CG method that is given in Algorithm 5. Also here the stopping criterion was adapted to use already computed quantities. Comparing this to the unpreconditioned version in Algorithm 3 one may observe that the difference in the numerical effort lies in solving line 16. In return one is rewarded with better convergence properties (at least if the algorithm does not stop earlier) that are given by the condition number of $C^{-\frac{1}{2}}\nabla^2 j(u)C^{-\frac{1}{2}}$, which should be smaller than the former one if C is in some sense a good approximation of $\nabla^2 j(u)$.
As we will see in the following, sometimes a mixture of the formulations 4 and 5 is more useful in our context. Therefore we look at the algorithm resulting from a discretization of Algorithm 3. Upon discretizing we would like the appearing vectors living in, say, $L^2(Q)$ to be replaced by elements from $\mathbb{R}^n$. Doing so, the scalar products $(\cdot,\cdot)_{L^2(Q)}$ would become the discrete scalar products $(\cdot,\cdot)_M$ given by some positive definite matrix $M \in \mathbb{R}^{n\times n}$. This means that we will replace the Hilbert space $(L^2(Q), (\cdot,\cdot)_{L^2(Q)})$ with $(\mathbb{R}^n, (\cdot,\cdot)_M)$. The natural question that arises is how this relates to the usual formulation of the algorithm that is given in $(\mathbb{R}^n, (\cdot,\cdot)_{\ell_2})$. Since

Algorithm 4 Steihaug-CG (C-scalar product)

```
Input: control $u$, gradient $\nabla j(u)$, trust region radius $\sigma$
Output: the solution $\delta u$ of $\min_{\|\delta u\|_C \le \sigma} m(\delta u)$
 1: $\delta u_0 \coloneqq 0,\ h_0 = \nabla_C j(u),\ d_0 \coloneqq -\nabla_C j(u)$
 2: $i \coloneqq 0$
 3: while $i \le i_{\max}$ do
 4:     compute $\nabla_C^2 j(u) d_i$
 5:     if $(d_i, \nabla_C^2 j(u) d_i)_C \le 0$ then
 6:         $\delta u_{i+1} \coloneqq \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_{i+1}\|_C = \sigma$
 7:         return $\delta u_{i+1}$
 8:     end if
 9:     $\alpha_i \coloneqq (h_i, h_i)_C / (d_i, \nabla_C^2 j(u) d_i)_C$
10:     $\delta u_{i+1} \coloneqq \delta u_i + \alpha_i d_i$
11:     if $\|\delta u_{i+1}\|_C \ge \sigma$ then
12:         $\delta u_{i+1} \coloneqq \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_i + \tau d_i\|_C = \sigma$
13:         return $\delta u_{i+1}$
14:     end if
15:     $h_{i+1} \coloneqq h_i + \alpha_i \nabla_C^2 j(u) d_i$
16:     $\beta_{i+1} \coloneqq (h_{i+1}, h_{i+1})_C / (h_i, h_i)_C$
17:     $d_{i+1} \coloneqq -h_{i+1} + \beta_{i+1} d_i$
18:     if $\|h_{i+1}\|_C <$ tol or $\|h_{i+1}\|_C <$ reltol$\|h_0\|_C$ then
19:         return $\delta u_{i+1}$
20:     end if
21:     $i \coloneqq i + 1$
22: end while
```

Algorithm 5 preconditioned Steihaug-CG

```
Input: control $u$, gradient $\nabla j(u)$, trust region radius $\sigma$
Output: the solution $\delta u$ of $\min_{\|\delta u\|_C \le \sigma} m(\delta u)$
 1: $\delta u_0 \coloneqq 0,\ r_0 \coloneqq \nabla j(u),\ h_0 = C^{-1} \nabla j(u),\ d_0 \coloneqq -h_0$
 2: $i \coloneqq 0$
 3: while $i \le i_{\max}$ do
 4:     compute $\nabla^2 j(u) d_i$
 5:     if $(d_i, \nabla^2 j(u) d_i)_U \le 0$ then
 6:         $\delta u_{i+1} \coloneqq \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_{i+1}\|_C = \sigma$
 7:         return $\delta u_{i+1}$
 8:     end if
 9:     $\alpha_i \coloneqq (r_i, h_i)_U / (d_i, \nabla^2 j(u) d_i)_U$
10:     $\delta u_{i+1} \coloneqq \delta u_i + \alpha_i d_i$
11:     if $\|\delta u_{i+1}\|_C \ge \sigma$ then
12:         $\delta u_{i+1} \coloneqq \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_i + \tau d_i\|_C = \sigma$
13:         return $\delta u_{i+1}$
14:     end if
15:     $r_{i+1} \coloneqq r_i + \alpha_i \nabla^2 j(u) d_i$
16:     $h_{i+1} \coloneqq C^{-1} r_{i+1}$
17:     $\beta_{i+1} \coloneqq (r_{i+1}, h_{i+1})_U / (r_i, h_i)_U$
18:     $d_{i+1} \coloneqq -h_{i+1} + \beta_{i+1} d_i$
19:     if $(r_{i+1}, h_{i+1})_U <$ tol or $(r_{i+1}, h_{i+1})_U <$ reltol$(r_0, h_0)_U$ then
20:         return $\delta u_{i+1}$
21:     end if
22:     $i \coloneqq i + 1$
23: end while
```

Algorithm 6 preconditioned Steihaug-CG (computationally efficient version formulated in $\mathbb{R}^n$)

Note that $M^{-1}\nabla^2 j(u)d_i$ can be cheaply computed by simply omitting the multiplication with M in (3.10) (and similarly for $M^{-1}\nabla j(u)$). In particular note the difference in line 17 compared to line 16 of Algorithm 5.

Input: control u, gradient $M^{-1}\nabla j(u)$, trust region radius σ

Output: the solution δu of $\min_{\|\delta u\|_M \le \sigma} m(\delta u)$

```
1: $\delta u_0 := 0,\ h_0 = M^{-1}\nabla j(u),\ r_0 := Mh_0,\ d_0 := -h_0$
2: $i := 0$
3: while $i \le i_{\max}$ do
4:   $y_i := M^{-1}\nabla^2 j(u)d_i$
5:   $z_i := My_i$
6:   if $(d_i, z_i)_{\ell_2} \le 0$ then
7:     $\delta u_{i+1} := \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_{i+1}\|_M = \sigma$
8:     return $\delta u_{i+1}$
9:   end if
10:  $\alpha_i := (r_i, h_i)_{\ell_2}/(d_i, z_i)_{\ell_2}$
11:  $\delta u_{i+1} := \delta u_i + \alpha_i d_i$
12:  if $\|\delta u_{i+1}\|_M \ge \sigma$ then
13:    $\delta u_{i+1} := \delta u_i + \tau d_i$ with $\tau > 0$ s.t. $\|\delta u_i + \tau d_i\|_M = \sigma$
14:    return $\delta u_{i+1}$
15:  end if
16:  $r_{i+1} := r_i + \alpha_i z_i$
17:  $h_{i+1} := h_i + \alpha_i y_i$
18:  $\beta_{i+1} := (r_{i+1}, h_{i+1})_{\ell_2}/(r_i, h_i)_{\ell_2}$
19:  $d_{i+1} := -h_{i+1} + \beta_{i+1} d_i$
20:  if $(r_{i+1}, h_{i+1})_{\ell_2} <$ tol or $(r_{i+1}, h_{i+1})_{\ell_2} <$ reltol$(r_0, h_0)_{\ell_2}$ then
21:    return $\delta u_{i+1}$
22:  end if
23:  $i := i + 1$
24: end while
```

the discretization does not result immediately in the ℓ_2-formulation, we could try to apply the results obtained so far in this section to see if we can identify it with a preconditioned algorithm. Looking at (3.21) we may try relating U to ℓ_2 and C to the matrix M there, i.e., we suspect the discretized version to be preconditioned with the matrix M. Then the resulting algorithm after the discretization would resemble Algorithm 4 (identifying $\nabla_C j(u)$ and $\nabla_C^2 j(u)$ with the quantities obtained from discretization) and from the above discussion we would obtain the equivalence to Algorithm 5 which is now formulated in terms of ℓ_2. Hence the discretized algorithm is indeed equivalent to a preconditioned Steihaug-CG method with preconditioner M formulated in terms of the standard scalar product of $\mathbb{R}^n$.

To better understand the fact that the discretized formulation is related to a preconditioned algorithm, let us for a moment return to Algorithm 3 in the undiscretized version. Note that on Banach spaces derivatives are actually elements of the dual space, i.e., we have $j'(u) \in U'$ and $j''(u) : U \to U'$. Since U was assumed to be a Hilbert space we implicitly used the Riesz mapping $R_U : U \to U'$ to identify $\nabla j(u) = R_U^{-1} j'(u) \in U$ and $\nabla^2 j(u) = R_U^{-1} j''(u) : U \to U$ such that the algorithm could be formulated in terms of the U-scalar product instead of using the duality product $\langle \cdot, \cdot \rangle_{U',U}$. However, in formulating the algorithm, we would not be able to proceed without using the Riesz representative $\nabla j(u)$, since in line 10 we add the search direction d_i to the iterate $\delta u_i \in U$. As d_i is related to the gradient (see line 1) at the latest at this point we would have had to make the just stated identifications to obtain $d_i \in U$ and a well defined sum. In some sense the preconditioning step in the discrete version corresponds to this identification that admittedly is somewhat hidden in the former Hilbert space formulation. The operator R_U is in the discrete version given by the matrix M. In Algorithm 4 where we apply the 'scalar' product we have to use the Riesz representatives given by the relation (3.23) and in Algorithm 5 where we apply the 'duality' product the quantity r_{i+1} has to be 'converted' in line 16 to its Riesz representative such that d_{i+1} is in the correct space.

As mentioned above, Algorithm 5 can be implemented more efficiently in the context of optimal control by also using the formulation in terms of $\nabla_C j(u)$ and $\nabla_C^2 j(u)$ given in Algorithm 4. The reason is that the application of the Hessian is in our case performed by evaluating several operators in a row (cf. also (3.10)). First we would have to solve the linearized state equation and then the additional adjoint equation. At this point we would already have computed $\nabla^2 j(u)$ and obtaining $j''(u)$ would correspond to applying R_U (or M in the discrete case). But since we are actually interested in the former, there would be an unnecessary application of this operator that would be removed again by the subsequent preconditioning step. Instead, one can store both quantities h_i and r_i in parallel, first computing h_i using $\nabla_C^2 j(u) \mathrel{\hat{=}} j''(u)$ and then after applying $C \mathrel{\hat{=}} M$ the residual r_i using $\nabla^2 j(u)$. r_i can then be used to cheaply apply the scalar product ($(r_i, h_i)_{\ell_2}$ is favorable to $(h_i, h_i)_M$ as no matrix multiplication is involved) and h_i can be used to update the search direction d_i. In this case no inversion of M is needed, just its application. The just described combination of the Algorithms 4 and 5 is listed in Algorithm 6 formulated for the discrete case. Let us conclusively summarize the main aspects of this section.

Conclusion 3.1.3. *As is known, the Steihaug-CG algorithm in the C-scalar product (Algorithm 4) is equivalent to a preconditioned Steihaug-CG method with preconditioner C (Algorithm 5). From this we can deduce that upon discretizing the Steihaug-CG algorithm in $(L^2(Q), (\cdot,\cdot)_{L^2(Q)})$ one obtains an algorithm formulated in $(\mathbb{R}^n, (\cdot,\cdot)_{\ell_2})$ with preconditioner M. For the reduced Hessian approach, mesh independence is assured as a consequence. Further, in this case one can write down Algorithm 6 that per iteration saves one matrix-vector application in Algorithm 4 and the linear solve in line 16 of Algorithm 5 by combining these two algorithms.*

In fact, the mesh independence will be checked numerically later, see Section 4.2. Nevertheless, as also mentioned earlier, the algorithm still takes forbiddingly many iterations in some situations which hints to a bad condition number due to other reasons than mesh dependence. The question arises if one further could apply other kinds of preconditioning to remedy this. This will be discussed in the next section.

3.2 Preconditioning

As hinted at the end of the previous section, numerical investigations suggest that the Newton system for the considered optimization problem is ill-conditioned. As an indicator one may consider the amount of iterations the Steihaug-CG method takes closely before the trust region Newton method converges. It does not uncommonly take a total count of 300 to 500 for most relevant settings and sometimes even exceeds the given maximum amount of 800 iterations, see Section 4.5. Furthermore, these trust region steps occupy a significant amount of the total running time of the algorithm. This is despite the fact that, due to the function space formulation of the Steihaug algorithm, its convergence properties should be relatively independent of the space and time discretization. As discussed in the previous section, this formulation belongs to an effective preconditioning by the mass matrix, which however might be insufficient as in principle it approximately only induces a scaling of the system matrix. Also the authors of [24], where the case of the isotropic Allen-Cahn equation is considered, conclude that already this problem is ill-conditioned with comparable iteration counts. Hence, it is unlikely that the anisotropy contributes the dominating part to the deficiencies although it might still provide some residual effect. For sake of an investigation of the main issues, we will therefore mainly concentrate on that simpler case in this section.
To this end, let us briefly repeat the Hessian's form given in eqs. (3.4) to (3.9) in the isotropic case for the reader's convenience. It reads as

$$\nabla^2 j(u) = \left(\tfrac{\lambda}{\varepsilon} I + \mathcal{K}(y)^{-*}\mathcal{P}(y,p)\mathcal{K}(y)^{-1}\right), \tag{3.24}$$

where eqs. (3.5) and (3.6) in this case reduce to

$$\mathcal{K}(y) = \varepsilon\partial_t - \varepsilon\Delta + \tfrac{1}{\varepsilon}(3y^2-1), \qquad \mathcal{K}^*(y) = -\varepsilon\partial_t - \varepsilon\Delta + \tfrac{1}{\varepsilon}(3y^2-1), \tag{3.25}$$

and the application of $\mathcal{K}(y)^{-1}$ and $\mathcal{K}(y)^{-*}$ is given as in eqs. (3.7) and (3.8) (with $\nu_A = \nu$). The right-hand side of the additional adjoint equation is now provided by

$$\mathcal{P}(y,p)\delta y = -\tfrac{6}{\varepsilon} yp\delta y. \tag{3.26}$$

We point out that the main difference to the anisotropic case is the lack of a further term in $\mathcal{P}(y,p)\delta y$, that would render its behavior even less predictable as described in the subsequent subsection. As before, we will drop the arguments of $\mathcal{K}(y)$, $\mathcal{K}^*(y)$ and $\mathcal{P}(y,p)$ for the sake of readability. It is difficult to make precise statements about the condition number of (3.24). From a theoretical point of view, the dependence on the a priori unknown state y and the adjoint state p provides serious obstacles. Their appearance of both as a product in the operator $\mathcal{P}$ and the appearance of y in the operator $\mathcal{K}$—that in addition enters by its inverse—make an analytical treatment nearly impossible. Also the need to explicitly compute this inverse to determine the condition number provides limitations in the sense that realistic system sizes cannot be investigated numerically.
In the following subsections we nevertheless try to shed further light on the issues and discuss the dependence on several quantities as well as some problems that arise for the classical preconditioning ansatzes. First, we will treat the just mentioned dependence on the functions y and p. Then we will once again have a look at the mesh independence, in particular we will address the time discretization. Finally, we will discuss preconditioning strategies for the reduced and in the end also for the all-at-once approach.

3.2.1 Dependence of the condition number on y and p

When solving the Newton system, the functions y and p are given functions that were computed as solutions of the state and adjoint equation, respectively. One difficulty is that from the point of preconditioning they are rather arbitrary functions whose behavior can barely be predicted. This is due to the 'arbitrariness' of the control $u(t,x)$ provided in intermediate steps of the algorithm. Also this control might differ in orders of magnitude both with respect to t (at the

beginning the control is empirically more moderate than at the end) and with respect to x (for given t the support mainly lies on the interface). The same can be said about the adjoint state that, due to the first order optimality condition (3.3), should fulfill $p \approx -\frac{\lambda}{\varepsilon} u$ in vicinity of the optimal solution. Fortunately all this is not reflected by the state y which (at least during the converging phase) is observed to still obey $|y| \lesssim 1$ in case of the smooth double-well potential (2.4) (compare to Theorem 2.2.10 and the discussion below). Here, the values $y \approx 0$ are only attained at the interface. This should render the operator $\mathcal{K}$ to be a perturbation of the heat equation, but note that the additional term has a relative scaling of $\frac{1}{\varepsilon^2}$.
More problems should be provided by the operator $\mathcal{P}$ which prepares the right-hand side of the additional adjoint equation. For linear problems as the heat equation this operator would be positive semidefinite which would yield the whole $\nabla^2 j$ to be positive definite as a perturbation of a multiple of the identity. This is not guaranteed for the present nonlinear case and in fact this is the reason why we had to use Steihaug-CG instead of CG. As it is a requirement for the convergence of Newton's method, the Hessian should be positive definite in vicinity of an optimizer. So if the algorithm converges this should be fulfilled at least in the last few iterations. As already hinted, it is usually not until then that we require preconditioning, as before the Steihaug-CG method typically stops due to other reasons. But nevertheless, we sometimes observed large iteration counts already at the middle of the algorithm (see also Section 4.5) and we do not know a priory when the algorithm has entered the region of convergence. Also, we still cannot exclude that some eigenvalues are close to zero at the end.
As indicated before, the whole trouble originates from the operator $\mathcal{P}$. For the present nonlinear case, it in addition contains a term involving the product of y and p that we even do know less about than both individually. However, there are still some vague statements one can make about it. Most importantly, we expect the product to be small on most part of the domain Ω. This arises from the fact that $y \approx 0$ in vicinity of the interface and $p \approx 0$ away from the interface where almost no control takes place. It is hard to predict however of what order of magnitude the product will be in vicinity of the interface, where one factor is rather large and the other rather small.
Anyway, as there is no guarantee for y and p to have the same sign, the resulting operator $\mathcal{P}$ is expected to have both positive and negative eigenvalues. The negative ones might lead to small eigenvalues for the total $\nabla^2 j$. On the other hand also (positive) eigenvalues with large magnitude might lead to an ill-conditioned Hessian. For certain values of y (for instance $y \approx \pm\sqrt{\frac{1}{3}}$, i.e., the roots of ψ'') we might get eigenvalues of the order $\lambda(\mathcal{K}) \sim \varepsilon$ that together with the $\frac{1}{\varepsilon}$-scaling of $\mathcal{P}$ yields an amplification of $\sim \frac{1}{\varepsilon^3}$. In any case we can conclude that due to the explicit dependence of $\mathcal{K}$ and $\mathcal{P}$ on the state and adjoint variables, clustering of the eigenvalues is not granted. This was not the case already for the heat equation but for the Allen-Cahn the situation is definitely worse. In case of the anisotropic Allen-Cahn equation—where the operator $\mathcal{P}$ contains an additional term arising from the chain rule for A (cf. (3.9))—the described situation is even more unclear.

Conclusion 3.2.1. *The product yp appearing in the operator $\mathcal{P}$ is the reason for negative and/or possibly small eigenvalues of* (3.24)*. Although it should always hold $|y| \lesssim 1$, the value of p highly depends on the present setting. As a consequence it is hard to make any statements about yp. As the support of p is expected to lie around the interface, yp should be small nearly everywhere. For the anisotropic Allen-Cahn equation an additional term appears (cf. (3.9)) that makes the investigation even more obscure.*

Finally, let us briefly comment on the influence of the weight parameter λ. By choosing λ large enough one can always tune the system (3.24) to be positive definite and have a good condition number. This however leads to unpleasing qualitative results. As λ weights the penalization of the control cost, by choosing a value too large the algorithm eventually does not care about reaching the proper end state. An optimal solution would then be to just control virtually nothing at all. The standard value for λ used for the numerical results later is the biggest choice possible before such effects become dominant. For this value the influence of the second term in (3.24) still shows the problems described in this subsection.

3.2.2 Dependence of the condition number on the mesh and T

Due to the strong intervention of effects stemming from y and p that were discussed previously, it is difficult to deduce results on the mesh dependence of the condition number for the reduced Hessian arising in the optimal control of the anisotropic Allen-Cahn equation. To the author's knowledge there exists only literature in case of linear parabolic control problems, see, e.g., [96, 124, 112], where a similar system has to be solved for the reduced gradient that is already linearly dependent from the control in this case. In the present subsection we will therefore treat this simplified situation and suggest that the emerging mesh independence also holds in our case—which we at least can verify numerically (see Section 4.2). In all of the cited references the authors conclude that the condition number is independent of the coarseness of the space discretization. For the implicit time discretization the first reference contains an estimate that goes like $\frac{1}{\tau}$, whereas the authors of the second reference conclude only a small dependence from their bound. Let us note however that there is a flaw in the argument leading to the latter result, that starts after equation (5.9) in [124]. The deduced expression $(c_1 + \tau c_2 - d_1)$ is generally not positive for small τ, so one is not allowed to take the square as is done there. To obtain a definite result, we performed an own investigation that uses similar arguments as in the proof of Theorem 2.2.4 and that lead to a bound independent from the time step size τ and with at most linear growth in the end time point T. This will be discussed in the remainder of this subsection.

As a model problem we consider

$$\min J(y,u) := \frac{1}{2}\|y(T) - y_\Omega\|^2_{L^2(\Omega)} + \frac{\beta}{2}\|u\|^2_{L^2(Q)},$$

where $u \in L^2(Q)$ and $y \in H^1(0,T;L^2(\Omega)) \cap L^2(0,T;H^1(\Omega))$ shall fulfill the linear parabolic partial differential equation

$$\begin{aligned} (\partial_t y, \eta) + \alpha(\nabla y, \nabla \eta) + c_0(y,\eta) &= (u,\eta) && \forall \eta \in L^2(0,T;H^1(\Omega)), \\ y(0) &= y_0 && \text{in } \Omega, \end{aligned}$$

with $\alpha > 0$ and $c_0 \in \mathbb{R}$. Here $y_0 \in H^1(\Omega)$ and $\Omega \subset \mathbb{R}^d$ is again a bounded Lipschitz domain. The corresponding implicit time discretization of the state equation reads as

$$\tfrac{1}{\tau_j}(y_j,\varphi) + \alpha(\nabla y_j, \nabla\varphi) + c_0(y_j,\varphi) = (u_j,\varphi) + \tfrac{1}{\tau_j}(y_{j-1},\varphi) \qquad \forall \varphi \in H^1(\Omega),\ j = 1,\dots,N, \tag{3.27}$$

with given $y_0 \in H^1(\Omega)$. The last time step can then be understood as a function of the control variable and the initial state, i.e. $y_N(u_\tau, y_0)$. The adjoint equation corresponding to (3.27) reads as

$$\tfrac{1}{\tau_j}(\varphi,p_j) + \alpha(\nabla\varphi, \nabla p_j) + c_0(\varphi,p_j) = \tfrac{1}{\tau_j}(\varphi,p_{j+1}) \qquad \forall \varphi \in H^1(\Omega),\ j = N,\dots,1, \tag{3.28}$$

and $p_{N+1} := y_N - y_\Omega$. In the following, we use the same notation as given in (2.11). Again, we can understand the adjoint state as a function of the just given p_{N+1} and it further can be splitted due to linearity as

$$p_\tau = p_\tau(y_N(u_\tau,y_0) - y_\Omega) = p_\tau(y_N(u_\tau,0) + y_N(0,y_0) - y_\Omega) = p_\tau(y_N(u_\tau,0)) + \underbrace{p_\tau(y_N(0,y_0) - y_\Omega)}_{:=\bar{p}_\tau}, \tag{3.29}$$

such that the first order condition in essence reads as

$$F_\tau u_\tau := \beta u_\tau + p_\tau(y_N(u_\tau,0)) = -\bar{p}_\tau. \tag{3.30}$$

The linear operator $F_\tau : U_\tau \to U_\tau$ has the same structure as the operator $\nabla^2 j(u)$ (see (1.28)) in our case and hence we are interested in its condition number.

Theorem 3.2.2. *The condition number of the operator F_τ defined in (3.30) can be estimated by*

$$\kappa(F_\tau) \leq \begin{cases} 1 + 1.82\,\frac{T}{\beta} & \text{for } c_0 \geq 0, \\ 1 + \sqrt{e}\frac{T}{\beta}(\exp(-2c_0 T) + s(\tau)) & \text{for } c_0 < 0, \end{cases} \tag{3.31}$$

for τ being small enough and where $s(\tau) \to 0$ for $\tau \to 0$.

So if τ is small enough, the condition number of F_τ is bounded independently of the time discretization and for $c_0 \geq 0$ we have a bound with linear growth in T.

Proof: We will estimate the eigenvalues of the linear operator F_τ in order to deduce a bound for its condition number. Since $p_\tau = S_\tau^* S_\tau u_\tau$ where $S_\tau : U_\tau \to H^1(\Omega), u_\tau \mapsto y_N$, the map $u_\tau \mapsto p_\tau$ is positive semi-definite and hence the smallest eigenvalue can simply be estimated by $\mu_{\min} \geq \beta$. For the largest eigenvalue we have to make some further considerations. As in the proof of Theorem 2.2.4 one obtains for the present time discrete state equation as an analogue to (2.31)

$$\|y_J\|^2 \leq \left(\|y_0\|^2 + \tfrac{1}{\epsilon}\sum_{j=1}^{N_\tau}\tau_j\|u_j\|^2\right)\frac{1}{1-\tau_J(\epsilon - 2c_0)}\max\left\{1, \exp\left(\frac{\epsilon - 2c_0}{1-\tau_J(\epsilon-2c_0)}T\right)\right\}. \tag{3.32}$$

Here the additional max-function arises from a case distinction with respect to the sign of c_0 and ϵ is an arbitrary parameter from scaled Young's inequality satisfying $1 > \tau_N(\epsilon - 2c_0)$ and that we will specify later.
We also need a similar inequality for the time discrete adjoint equation (3.28) that in forward notation (i.e. $\tilde{p}_j := p_{N+1-j}$ for $j = 1, \ldots, N$) reads as

$$\tfrac{1}{\tau_{N+1-j}}(\tilde{p}_j - \tilde{p}_{j-1}, \varphi) + \alpha\,(\nabla\tilde{p}_j, \nabla\varphi) + c_0\,(\tilde{p}_j, \varphi) = 0 \qquad \forall\varphi \in H^1(\Omega),$$

and $\tilde{p}_0 := y_N - y_\Omega$. The same analysis as above yields after transforming back to backward p_τ the bound

$$\|p_\tau\|^2_{L^2(Q)} \leq \|y_N - y_\Omega\|^2\frac{T}{1+\underline{\tau}2c_0}\max\left\{1, \exp\left(\frac{-2c_0}{1+\bar{\tau}2c_0}T\right)\right\}, \tag{3.33}$$

where we define for now and later $\underline{\tau} := \min_j \tau_j$ and $\bar{\tau} := \max_j \tau_j$ or vice versa if $c_0 < 0$. With τ we still denote $\tau := \max_j \tau_j$ as in the chapters before. Combining (3.32) and (3.33), we get

$$\|p_\tau(y_N(u_\tau, 0))\|^2_{L^2(Q)} \leq k(\tau, T, c_0, \epsilon)\|u_\tau\|^2_{L^2(Q)},$$

with

$$\begin{aligned} k(\tau, T, c_0, \epsilon) := \frac{T}{\epsilon(1-\tau_N(\epsilon-2c_0))(1+\underline{\tau}2c_0)}&\max\left\{1, \exp\left(\frac{\epsilon-2c_0}{1-\tau_N(\epsilon-2c_0)}T\right)\right\} \\ &\cdot\max\left\{1, \exp\left(\frac{-2c_0}{1+\bar{\tau}2c_0}T\right)\right\} \end{aligned}$$

(remember that we set $y_0 := 0, y_\Omega := 0$ due to the splitting (3.29)). It remains to estimate $k(\tau, T, c_0, \epsilon)$ and for this we have to distinguish between the two possible signs of c_0.
For the case $c_0 \geq 0$ and assuming $\tau_N \leq T/N$ with $N \geq 11$ we can deduce by choosing $\epsilon = \frac{1}{T}$ (which is optimal in the limit with $\tau \to 0$)

$$k(\tau, T, c_0, \tfrac{1}{T}) \leq k(\tau, T, 0, \tfrac{1}{T}) \leq \tfrac{N}{N-1}\exp\left(\tfrac{N}{N-1}\right)T^2 \leq 1.1\exp(1.1)\,T^2,$$

and hence bounded independently of τ_N. So in this case we can conclude that for τ small enough

$$\beta \leq \mu \leq \beta + T\sqrt{1.1\exp(1.1)} < \beta + 1.82\,T. \tag{3.34}$$

On the other hand if $c_0 < 0$ we again set $\epsilon = \frac{1}{T}$ and by observing that the maxima are always attained by the exponentials we obtain

$$k(\tau, T, c_0, \tfrac{1}{T}) \quad \rightarrow \quad T^2 \exp\left(1 - 4c_0 T\right),$$

for $\tau \rightarrow 0$. Summarized, we have shown that the condition number can be estimated by (3.31). □

Note that we derived our result by using the Hilbert spaces $L^2(\Omega)$ and $H^1(\Omega)$ for the spatial variable and did not make any use of the properties of these spaces and their scalar products. Therefore, if we discretize the problem by a conforming finite element method, for the condition number we obtain the same estimate (3.31), meaning that it is also independent from the space discretization. Finally, let us again mention that the mesh independence was checked numerically to also hold for solving the Newton problem arising in optimal control of the anisotropic Allen-Cahn equation in Section 4.2.

3.2.3 Factorizing the Hessian

Having talked about the potential dependencies of the condition number as well as possible sources of bad condition, we now want to turn our attention to the issues that arise in applying known preconditioners to our problem. In this section, we will again consider the reduced Hessian approach, whereas in the subsequent one the all-at-once system is treated. Let us look at a general operator of the form $(I + AB)$. One typical idea would be to split it up as

$$(I + AB) \approx (I + A)(I + B), \tag{3.35}$$

cf., e.g., [12] or [118]. If for example $A = B = \Delta$ (i.e. the Laplacian) one could use the fact that there are well-established (fast) solvers for $(I + \Delta)$ like, e.g., multilevel solvers [29, 70, 110] that can be used to approximate the right-hand side. This does also work for more general elliptic operators. For parabolic equations, (parareal) time domain decomposition methods can be used (see, e.g., [129, 96, 72]). So how does (3.35) apply to our case? To keep the presentation clear, we set $\frac{\lambda}{\varepsilon} = 1$. The first question concerns the definition of the operators A and B. In fact, in (3.24) there appears the product of three operators $\mathcal{K}^{-*}\mathcal{P}\mathcal{K}^{-1}$. One obvious idea would be to take the square root of the central operator $\mathcal{P}$ (which in matrix representation would be block diagonal). As already seen above, however, in general $\mathcal{P}$ has negative eigenvalues and so the root would have to be approximated by, e.g., $\sqrt{|\mathcal{P}|}$, which would yield

$$\nabla^2 j(u) \approx (I + \mathcal{K}^{-*}\sqrt{|\mathcal{P}|})(I + \sqrt{|\mathcal{P}|}\mathcal{K}^{-1}). \tag{3.36}$$

Here, an efficient way to obtain the square root of the diagonal blocks of $\mathcal{P}$ is essential. Also it is not clear how well the approximation by the absolute value mimics the real eigenvalue distribution. As this seemingly already leads to open questions here, in the following we leave $\mathcal{P}$ as it is and take $A = \mathcal{K}^{-*}$ and $B = \mathcal{P}\mathcal{K}^{-1}$, i.e.

$$\nabla^2 j(u) \approx (I + \mathcal{K}^{-*})(I + \mathcal{P}\mathcal{K}^{-1}). \tag{3.37}$$

The next question arising would be how to efficiently solve $(I + \mathcal{K}^{-*})^{-1}$. The author does not know any results on this. Note that here the action of the operator A is not explicitly known as it belongs to an inverse that typically is not set up explicitly. Also note that the second term still contains a product which deteriorates the situation (this is also a problem for the square root ansatz). This stands in contrast to the Laplacian situation that we mentioned earlier and is more standard in the discussion of such preconditioners. Another problem is that the operator—or in any case $(I + \mathcal{P}\mathcal{K}^{-1})$—might even not be invertible due to nonpositive eigenvalues. We only know that $(I + \mathcal{K}^{-*}\mathcal{P}\mathcal{K}^{-1})$ has an inverse (at least in vicinity of the optimal solution). Although it is quite unlikely that we get an eigenvalue that exactly matches zero, already small values might have a great impact on the stability of the involved solvers. Also, if we have negative eigenvalues, it is not obvious which efficient solvers can be used. For the analysis it is problematic if we cannot exclude explicitly that the operator may be singular.

Finally, another idea would be to rewrite the just discussed approach as

$$\nabla^2 j(u) \approx \mathcal{K}^{-*}(\mathcal{K}^* + I)(\mathcal{K} + \mathcal{P})\mathcal{K}^{-1}. \tag{3.38}$$

In contrast to the other cases, no products and inverses appear in the factors that contain the sums. That is, for the terms $(\mathcal{K}^* + I)$ and $(\mathcal{K} + \mathcal{P})$ there may exist again efficient solvers. However, also here the problem remains that these two factors could potentially be singular or at least have negative eigenvalues and some which are close to zero. Another observation and potential drawback is that we now have to apply four operators in a row and the question arises if this would still be efficient enough for a preconditioner. The answer to this probably would have to be tested out in practice so we cannot say much at this point. If one wants to compute the approximate inverse of above expression it at least turns out that the additional factors are only applications of an operator belonging to a parabolic PDE and involve no solves. Note also that transforming the problem to $\mathcal{K}^*\nabla^2 j(u)\mathcal{K}$ probably provides no further advantage as one actually does not intend to solve the remaining problem iteratively.

Conclusion 3.2.3. *We discussed different approaches to apply the splitting* (3.35) *to our setting. The following issues were elaborated:*

(3.36)*: It is unclear if this is a good approximation and no efficient solver is known.*

(3.37)*: The right operator is not guaranteed to be invertible and no efficient solver is known.*

(3.38)*: The third operator is not guaranteed to be invertible and it is unclear if applying the product of four operators is efficient enough.*

3.2.4 Considering the full system as an alternative

As we have seen, the reduced system comes with several problems concerning preconditioners other than the simple mass matrix. Many researchers therefore prefer to consider the full system instead (see, e.g., [124, 20, 114]). Hence, we conclude this section by discussing the problems that are present there in context of the (anisotropic) Allen-Cahn equation. For convenience let us repeat its expression initially given in (3.12):

$$\begin{pmatrix} \mathcal{P} & 0 & -\mathcal{K}^* \\ 0 & \frac{\lambda}{\varepsilon}I & I \\ -\mathcal{K} & I & 0 \end{pmatrix} \begin{pmatrix} \delta y \\ \delta u \\ \delta p \end{pmatrix} = - \begin{pmatrix} L_y \\ L_u \\ L_p \end{pmatrix}. \tag{3.39}$$

Here, $\mathcal{K}$ and $\mathcal{P}$ are given by (3.25) and (3.26) for the isotropic case, or by (3.5) and (3.9) for the anisotropic case, respectively. We recall that in contrast to the reduced approach, mesh independence is not granted here, so preconditioning is in fact required already to achieve this.

Symmetrical preconditioner

A typical candidate for a preconditioner of (3.39) would be the block diagonal operator

$$\hat{\mathcal{P}}_{\text{sym}} := \begin{pmatrix} \mathcal{P} & 0 & 0 \\ 0 & \frac{\lambda}{\varepsilon}I & 0 \\ 0 & 0 & -\tilde{\mathcal{S}} \end{pmatrix}, \tag{3.40}$$

where $\tilde{\mathcal{S}}$ should be an approximation to the Schur complement $\mathcal{S}$ of (3.39) with respect to δp, i.e. $\mathcal{S} = -[\mathcal{K}\mathcal{P}^{-1}\mathcal{K}^* + I(\frac{\lambda}{\varepsilon}I)^{-1}I]$. This is a different operator than (3.24) which resulted from the Schur complement with respect to δu. For the approximation of $\mathcal{S}$ one typically leaves away the second term [114], so for instance $-\tilde{\mathcal{S}} = \text{AMG}(\mathcal{K})\mathcal{P}^{-1}\text{AMG}(\mathcal{K}^*)$, where by $\text{AMG}(\mathcal{O})$ we want to note down that the inverse of the operator $\mathcal{O}$ shall be approximated by an algebraic multigrid method [115, 55] (combined with forward/backward substitution in the parabolic case). Also parareal algorithms may be used to (approximately) solve the linearized state and adjoint equation [46, 128]. The identity I (i.e. the mass matrix after discretization) is usually treated by the Chebyshev semi-iteration [135, 63] or can simply be inverted if approximated by a lumped mass matrix. The difficulties for this approach again are related to the operator $\mathcal{P}$ that potentially is not positive definite and even could be singular. Therefore one might use an invertible approximation or simply replace it by an appropriately scaled mass matrix.

Nonsymmetrical preconditioner

Another idea would be to use a nonsymmetric preconditioner instead of (3.40) (see, e.g., [111] or [21]), as this avoids the need for the $(1,1)$-block to be positive definite. Here the idea is to first permute the system matrix (3.39) by exchanging the first and last row to obtain the matrix

$$\left(\begin{array}{cc|c} -\mathcal{K} & I & 0 \\ 0 & \frac{\lambda}{\varepsilon} I & I \\ \hline \mathcal{P} & 0 & -\mathcal{K}^* \end{array}\right). \tag{3.41}$$

The indicated block structure suggests the following block triangular preconditioner

$$\hat{\mathcal{P}}_{\text{nonsym},\Pi} := \begin{pmatrix} -\mathcal{K} & I & 0 \\ 0 & \frac{\lambda}{\varepsilon} I & 0 \\ \mathcal{P} & 0 & -\tilde{\mathcal{S}}_\Pi \end{pmatrix}, \tag{3.42}$$

where $\tilde{\mathcal{S}}_\Pi$ should be an approximation of the Schur complement $\mathcal{S}_\Pi$ of (3.41) with respect to the $(2,2)$-block, i.e.

$$\tilde{\mathcal{S}}_\Pi \approx \mathcal{S}_\Pi = -\mathcal{K}^* - \mathcal{P}\mathcal{K}^{-1} I \left(\tfrac{\lambda}{\varepsilon} I\right)^{-1} I. \tag{3.43}$$

Here and in the following we do not simplify the term $\left(\frac{\lambda}{\varepsilon} I\right)^{-1}$ such that the reader can better follow the derivation. The inverse of $\hat{\mathcal{P}}_{\text{nonsym},\Pi}$ is then given by

$$\hat{\mathcal{P}}^{-1}_{\text{nonsym},\Pi} = \begin{pmatrix} -\mathcal{K}^{-1} & \mathcal{K}^{-1} I \left(\frac{\lambda}{\varepsilon} I\right)^{-1} & 0 \\ 0 & \left(\frac{\lambda}{\varepsilon} I\right)^{-1} & 0 \\ -\tilde{\mathcal{S}}_\Pi^{-1}\mathcal{P}\mathcal{K}^{-1} & \tilde{\mathcal{S}}_\Pi^{-1}\mathcal{P}\mathcal{K}^{-1} I \left(\frac{\lambda}{\varepsilon} I\right)^{-1} & -\tilde{\mathcal{S}}_\Pi^{-1} \end{pmatrix}, \tag{3.44}$$

which can be verified by direct computation. Permuting this back one finally obtains the following preconditioner for the system (3.39)

$$\hat{\mathcal{P}}^{-1}_{\text{nonsym}} := \begin{pmatrix} 0 & \mathcal{K}^{-1} I \left(\frac{\lambda}{\varepsilon} I\right)^{-1} & -\mathcal{K}^{-1} \\ 0 & \left(\frac{\lambda}{\varepsilon} I\right)^{-1} & 0 \\ -\tilde{\mathcal{S}}_\Pi^{-1} & \tilde{\mathcal{S}}_\Pi^{-1}\mathcal{P}\mathcal{K}^{-1} I \left(\frac{\lambda}{\varepsilon} I\right)^{-1} & -\tilde{\mathcal{S}}_\Pi^{-1}\mathcal{P}\mathcal{K}^{-1} \end{pmatrix}. \tag{3.45}$$

The application

$$v = \hat{\mathcal{P}}^{-1}_{\text{nonsym}} w \qquad \text{with } v := \left(\begin{smallmatrix} v_1 \\ v_2 \\ v_3 \end{smallmatrix}\right) \text{ and } w := \left(\begin{smallmatrix} w_1 \\ w_2 \\ w_3 \end{smallmatrix}\right)$$

can be computed as follows. From the second row we get

$$v_2 = \left(\tfrac{\lambda}{\varepsilon} I\right)^{-1} w_2. \tag{3.46}$$

Inserting this into the first row, we obtain

$$v_1 = \mathcal{K}^{-1}\left(I \left(\tfrac{\lambda}{\varepsilon} I\right)^{-1} w_2 - w_3\right) = \mathcal{K}^{-1}\left(I v_2 - w_3\right). \tag{3.47}$$

Finally, after some straightforward rearrangement, v_3 can be computed from the third row by

$$v_3 = \tilde{\mathcal{S}}_\Pi^{-1}\left(\mathcal{P} v_1 - w_1\right). \tag{3.48}$$

For eqs. (3.46) and (3.47) the same approximation ideas as discussed in the previous subsection can be applied. The efficient solvability of (3.48) relies on a good approximation of $\tilde{\mathcal{S}}_\Pi$. Here one could use for example

$$\tilde{\mathcal{S}}_\Pi = -\left(\mathcal{K}^* + \mathcal{P}\right)\mathcal{K}^{-1}\left(\mathcal{K} + I\left(\tfrac{\lambda}{\varepsilon}I\right)^{-1}I\right), \tag{3.49}$$

as proposed in [111]. Unfortunately, as before, the first factor is not guaranteed to be invertible, since in our case the operator $\mathcal{P}$ can also assume negative eigenvalues. To test the preconditioner $\hat{\mathcal{P}}_{\text{nonsym}}$ we tried to set $\mathcal{P} = 0$, since in Section 3.2.1 we argued that the product yp should be of small magnitude in most parts of the domain. Also simply setting $\tilde{\mathcal{S}}_\Pi = -\mathcal{K}^*$ (i.e. neglecting the second term in (3.43)) yielded comparable results.

Comparison between the preconditioners

The question that remains is whether preconditioning the full system provides an advantage opposed to using the reduced system that the main focus lies on in this thesis. Therefore we implemented a test program for the full system with the symmetric preconditioner (3.40) and the nonsymmetric one (3.45). To deal with the block matrix structure, the implementation was done in Python using the *cbc.block* library [95] building on FEniCS (see [5] and also the next chapter). For the AMG solver we resorted to *pyamg* [109]. Note that the reduced and full approach are difficult to compare, due to their rather different nature. For instance, the state y produced after each Newton step of the full approach does not necessarily have to fulfill the state equation with the current iterate u as right-hand side. To test the all-at-once approach we looked at one step of a Newton solver (i.e. one solve of (3.39)) that we fed with a configuration obtained from the reduced approach and then considered the next iteration for both solvers. The result is presented in Table 3.1. We tested different linear solvers provided by *cbc.block* combined with both the nonsymmetric and the symmetric preconditioner (with $\mathcal{P}$ approximated by I). The best result was obtained by the nonsymmetric preconditioner (3.45) with $\tilde{\mathcal{S}}_\Pi = -\mathcal{K}^*$ using an LGMRES solver with 10 inner iterations, which we present exemplary for comparison in Table 3.1. As a setting we considered the isotropic case with 50 timesteps of size 10^{-4} on a 129×129 grid for keeping a circle of radius 0.5 constant (for more details on the other standard parameters chosen consider the introduction of Part 4). To compare both ansatzes, we measured the running times the solvers required to reduce the residual by six orders of magnitudes. As can be seen, solving the full system lasted about an order of magnitude longer than the reduced approach. That is mainly due to the higher iteration count the full problem required even with preconditioning. Also the times per iterative solver iteration are better for the reduced system but should be taken with a grain of salt due to the different implementation frameworks. Nevertheless, the difference could be partially explained by the higher amount of degrees of freedom for the full system ((y, u, p) instead of just u) and the overhead of the preconditioner. Furthermore, for the preconditioner we could not observe notable differences when dropping y- and p-dependent terms or replacing A' by the Laplacian in the anisotropic case. Also using only the identity as an approximation of $\mathcal{P}$ seemed to be a good approach. As the involved preconditioner is not sensitive to such details, together with the performance properties discussed above, it seems that only mesh independence can be reduced which however is a natural property of the reduced Hessian ansatz.
Finally let us note that only using Newton's method for the full system is not enough in general as only its local convergence is guaranteed. For global convergence, strategies as sequential quadratic programming (SQP) usually have to be involved. In contrast, the Steihaug-CG method, which we used for the comparison with the reduced system, is already part of the globalization via the trust region method as explained in Sections 3.1.1 and 3.1.2. For the numerical example discussed above, we ensured that the Steihaug-CG method does not have to stop earlier and in principle it is similar to a pure CG method.

Conclusion 3.2.4. *The full system* (3.39) *requires preconditioning to be mesh independent. We discussed a symmetric* (3.40) *and a nonsymmetric* (3.45) *preconditioner that are commonly used in the literature. For the symmetric preconditioner, as $\mathcal{P}$ can become singular it has to be*

	reduced problem	unreduced problem with preconditioner
degrees of freedoms	832,050	2,496,150
iteration count	32	114
time per iteration	5-6s	12-13s
time total	193s	1455s

Table 3.1: Comparison of one step of the solvers for the reduced as well as the all-at-once system. The latter was preconditioned with a nonsymmetric preconditioner. The iteration counts belong to the Steihaug-CG solver and a LGMRES solver, respectively, and are given for a reduction of six orders of magnitude. Here, the isotropic Allen-Cahn equation was considered. For further details consult the explanations in the pertinent subsection.

replaced. This is remedied by the nonsymmetric one, but also here the indefiniteness of $\mathcal{P}$ might exclude some common ansatzes for the approximation of the appearing Schur complement (3.43)*, as, e.g.,* (3.49)*. Further it was observed that dropping y-dependent terms in $\mathcal{K}$ usually has no negative impact on the effectiveness of the preconditioner. In numerical experiments we observed that the mesh dependence could in fact be removed, but nonetheless the reduced approach* (3.24) *performed better. This also means that the other sources of bad condition could not be met.*

3.3 Comments on the implementation

Before we discuss the numerical results obtained from the methods described in Section 3.1, let us give some comments on various implementational details. We note that the algorithm was implemented in C++, where the appearing partial differential equations were solved using the finite element toolbox FEniCS [5]. In the following section we shall give further information on how the solutions to the PDEs were computed. Later, we will comment on the use of parallelization in our code and issues that arise when applying mesh adaptivity.

3.3.1 Solving the PDEs

As mentioned introductory, in order to solve the appearing partial differential equations we resort to the finite element toolbox FEniCS [5] or rather its C++ interface DOLFIN [91, 92]. For more information than we will discuss here, we refer the reader to [90] for the general aspects of this framework, or [60] for an introduction closer related to optimal control. The components of FEniCS comprise an all-in-one solution for the problem of solving PDEs on the computer, starting from providing a simple domain-specific language to formulate the variational problem in a notation similar to the usual one used in the mathematical context up to finally supplying convenient interfaces to various PDE-solvers provided by common linear algebra packages as PETSc, Trilinos or Eigen. Let us briefly describe how a typical workflow for including a PDE solver in the code does look like using FEniCS. First of all the variational forms are coded into a UFL file. The Unified Form Language (UFL, [6, 3]) is a Python based domain-specific language for representing weak formulations of PDEs. In a next step this is compiled using the FEniCS Form Compiler (FFC, [85, 93, 108]), which outputs a C++ header file that is conform to the UFC specification. The Unified Form-assembly Code (UFC, [7, 4]) defines functions in C++ that perform the assembly of variational forms and classes that then finally can be used in the C++ (or Python) code via the DOLFIN library. Before we discuss in more detail how the parabolic (quasilinear and linear) PDEs needed for the outer solver were implemented, let us give an overview of the inner solvers from the FEniCS framework we utilized. Figure 3.1 provides a quick overview of the trust region algorithm we described previously with respect to the elliptic problems that have to be solved. With N we denote again the numbers of steps

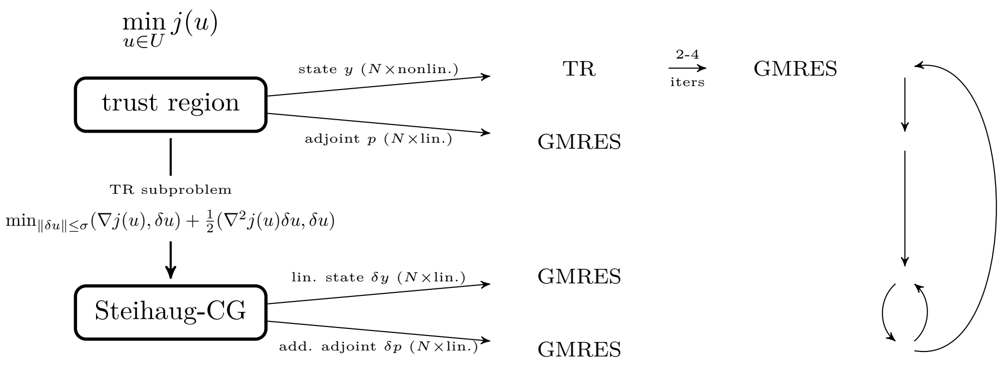

$\min_{u \in U} j(u)$
trust region
state y ($N\times$nonlin.)
adjoint p ($N\times$lin.)
TR
2-4 iters
GMRES
GMRES
TR subproblem
$\min_{\|\delta u\| \le \sigma} (\nabla j(u), \delta u) + \frac{1}{2}(\nabla^2 j(u)\delta u, \delta u)$
Steihaug-CG
lin. state δy ($N\times$lin.)
add. adjoint δp ($N\times$lin.)
GMRES
GMRES

Figure 3.1: The trust region Newton method with Steihaug-CG and the solvers that were used to compute solutions to the appearing PDEs. The value of 2-4 inner Newton steps is an empirical estimate from numerical observations.

the time discrete problems consist of[1] (see text below (2.10)). First we have to solve the state and adjoint equations, each of which requires the successive solution of N elliptic PDEs. To treat each linear elliptic PDE in case of the adjoint equation, we will use a GMRES (generalized minimal residual) solver as it showed the best convergence behavior. Each time step of the state equation is of nonlinear nature and hence we apply the built-in Newton trust region solver of FEniCS to it. From experiment we can say that the number of linear solves this requires lies around 4. The latter are again performed by the GMRES method. Note that in case of the semi-implicit discretization presented in Section 2.7, solving the trust region subproblem requires the successive solution of the linearized and additional adjoint equation. Each of them again consist of N linear equations to that we apply the GMRES method. The latter is in all cases provided by the backend PETSc [11, 10, 9] and is preconditioned by an incomplete LU factorization (ILU). It needs about 10 iterations per time step with a relative tolerance of 10^{-10}. In what follows we discuss how the state and adjoint equation were implemented. The linearized state and additional adjoint equation are computed in an analogous way as the adjoint equation (because for each of those three equations the elliptic subproblems are linear) and therefore will only be discussed briefly at the end. We decided to start with the adjoint equation, as for the nonlinear state equation we in addition have to set up a Newton solver. Note that for sake of clarity the presented code snippets are modified versions of the actual implementation in terms of leaving out some commands and comments that are insignificant or distracting. Also some variable and function names were changed slightly to better fit the notations used in the thesis. The full code that was used to produce the results of Part 4 can be found on the CD attached to this dissertation.

Adjoint equation

First let us have a look a the UFL codes which specify the PDEs in their variational form. We would like to adopt the general definition of A' as given in (2.98). Therefore we first look at a helper code written in Python which can be called in UFL files (recall that UFL is a domain specific language on the basis of Python). The following code defines a function that returns $A'_\delta(p)$ for some $p \in \mathbb{R}^d$ and $\delta > 0$.

```
from ufl import *
nl = 8

def A_prime(p, E, delta):
    return dot(M_rg(p, E, delta), p)
```

[1]This number is the same for each of the four equations as we do not resort to adaptivity, see also Section 3.3.3.

```
 6
 7  def M_rg(p, E, delta):
 8      G = [as_matrix([[E[i*4],E[i*4+1]],[E[i*4+2], E[i*4+3]]]) \
 9          for i in range(len(E)/4)]
10      GAM = [sqrt(inner(dot(Gl,p), p)+delta) for Gl in G]
11      gamma = sum(GAM)
12      return sum(gamma*inv(GAM[l])*G[l] for l in range(len(G)))
```

Note that in the code we made use of the splitting (2.104) by calling a second function which implements $M_\delta(p)$ as defined in (2.106). With that we can reuse the code above also for the implementation of the splitting scheme (2.109). The main issue in defining A'_δ is to leave the choice of G_l as general as possible. In our case this is accomplished by storing its entries in the `VectorConstant E`. The latter will appear as an attribute in the forms we will define soon. Let's assume we have a `LinearForm` called `L` in the C++ code, then the matrix entries can be handed over by

```
L.E = std::make_shared<Constant>(std::vector<double>{1,0,0,1e-2,1e-2,0,0,1});
```

where in this case we have defined the exemplary 2×2 matrices

$$G_1 = \begin{pmatrix} 1 & 0 \\ 0 & 10^{-2} \end{pmatrix}, \qquad G_2 = \begin{pmatrix} 10^{-2} & 0 \\ 0 & 1 \end{pmatrix}.$$

This can be seen by looking at the extraction in line 8. Note that with this approach the dimension is fixed to $d = 2$ and also the number of matrices L has to be defined before compiling. In the Python code this is the hard coded number `nl` $= L \cdot d^2$ defined in line 2. Nevertheless, increasing d or L can be done straightforwardly if needed, but requires to recompile. The next listing demonstrates how we can use the previous definition to implement the forms needed for specifying the discrete adjoint equation (2.77).

```
 1  import imp
 2  def_A = imp.load_source('', './def_A.py')
 3
 4  E  = VectorConstant("triangle", def_A.nl)
 5  P1 = FiniteElement("Lagrange", "triangle", 1)
 6
 7  v   = TestFunction(P1)
 8  p   = TrialFunction(P1)  # current solution
 9  p0  = Coefficient(P1)    # solution from previous time step
10  y   = Coefficient(P1)    # y(T-t)
11
12  tau       = Constant(triangle)
13  epsilon   = Constant(triangle)
14  delta     = Constant(triangle)
15
16  # second derivative of the potential
17  ddphi = 3*y*y-1
18
19  # define forms
20  a = tau*dot(derivative(def_A.A_prime(grad(y), E, delta), y, v), grad(p))*dx \
21      + v*p*dx + tau/(epsilon*epsilon)*ddphi*p*v*dx
22  L = p0*v*dx
```

The most important part is the definition of the `BilinearForm a` and the `LinearForm L` in lines 20 and 22. Here `a` is the left-hand side of (2.77) with the unknown p_j represented by `p` and the right-hand side is given by `L` where the previous computed solution p_{j+1} appears as `p0`. Note that the derivative A''_δ is computed automatically by the `derivative` function provided by FEniCS in line 20. The purpose of the previous bunch of lines is to define the quantities appearing in the definition of the forms. For instance, the scalar values τ_j, ε and δ are represented by variables of type `Constant` and will be defined in the C++ code similar to `E` above (no need for a `std::vector` here). Furthermore the appearing functions are defined in lines 7 to 10. Both forms contain a `TestFunction` and the `BilinearForm a` differs from the `LinearForm L` by also including the `TrialFunction p` after which is solved. Functions without further purpose like the solution of the previous time step or of the state equation enter as `Coefficient`. The above specifications can now be compiled on the command line by

```
ffc -l dolfin -O Adjoint_i.ufl
```

where the option `-l` specifies for which interface the output should be generated and `-O` instructs the form compiler to produce optimized code. The next listing demonstrates how the defined forms can be used in the C++ code via the DOLFIN library to solve the adjoint equation.

```cpp
void solve_adjoint(std::shared_ptr<const Mesh> mesh, double* y, double* p_out)
{
        // initialize forms and set the required parameters
        auto V = std::make_shared<Adjoint_i::FunctionSpace>(mesh);

        Adjoint_i::BilinearForm a(V, V);
        Adjoint_i::LinearForm L(V);

        auto p = std::make_shared<Function>(V);
        auto p0 = std::make_shared<Function>(V);
        auto y_t = std::make_shared<Function>(V);

        auto c_tau = std::make_shared<Constant>(pms.tau);
        auto c_epsilon = std::make_shared<Constant>(pms.epsilon);
        auto c_delta = std::make_shared<Constant>(pms.delta);

        a.tau = c_tau;
        a.epsilon = c_epsilon;
        a.y = y_t;
        a.delta = c_delta;
        a.E = E;
        L.p0 = p0;

        // set up intitial condition
        TargetState y_omega;
        auto y_T = std::make_shared<Function>(V);
        auto y_omega_i = std::make_shared<Function>(V);
        y_omega_i->interpolate(y_omega);
        set_vector(*(y_T->vector()), y, pms.n_timesteps, mesh);
        p = y_T;
        *(p->vector()) -= *(y_omega_i->vector());
        *(p->vector()) *= 1/pms.epsilon;

        // set up GMRES solver (default preconditioner is ILU)
        KrylovSolver solver("gmres", "default");
        solver.parameters["nonzero_initial_guess"] = true;
        solver.parameters["relative_tolerance"] = pms.lin_tol;
        solver.parameters["absolute_tolerance"] = pms.lin_globaltol;

40          Vector b;
41          Matrix S;
42  
43          // solve the time discrete problem
44          int i=pms.n_timesteps;
45          for (double t = pms.T; t >= pms.tau-__EPS__; t -= pms.tau)
46          {
47                  *(p0->vector()) = *(p->vector());
48                  set_vector(*(y_t->vector()), y, i, mesh);
49  
50                  assemble(b, L);
51                  assemble(S, a);
52                  solver.solve(S, *(p->vector()), b);
53  
54                  set_data(*(p->vector()), p_out, i, mesh);
55                  i--;
56          }
57  
58          set_data(*(p->vector()), p_out, i, mesh);
59  }
```

The function `solve_adjoint` takes as parameters the grid, a vector containing the solution of the state equation that appears as final condition and a vector where the computed solution is stored. In lines 4 to 22 the `BilinearForm` and `LinearForm` defined in a file called `Adjoint_i.h` outputted by the previous `ffc` command are initialized. `pms` is a global `struct` containing several parameters that appear quite commonly in the whole simulation. Also `E` which contains the definition of the G_l's is defined globally as it appears in every function that solves one of the PDEs. Next, the right-hand side of the first equation given by p_{N+1} (see below (2.77)) is defined. The GMRES solver is initialized in line 35 and its parameters are set up until line 38. The computationally expensive part is done in the for-loop between lines 45 and 56, where each time step of (2.77) is computed successively. The value of `__EPS__` is 10^{-15} and it is used to compensate rounding errors accumulated during the subtractions. The GMRES solver is called in line 52, where the parameters are a `Vector` and `Matrix` containing the previously assembled forms and a `Vector` where the obtained solution for the current time step is stored. The collection of all time steps is stored in the variables `y` and `p_out` that are of type `double*` and have dimension $(N+1) \times N_h^2$, where with N we denote the number of time steps like in Section 2.2 and by N_h we denote the number of spatial grid points in one direction. The additional time step is for the initial value that for convenience is also stored in these variables. Finally we note that line 50 could equally well have been written as

```
M.mult(*(p0->vector()), b);
```

with a mass matrix `M` defined once. This might be cheaper as an explicit assembly, but in the example above we wanted to demonstrate how the right-hand side can generally be obtained by defining a `LinearForm`.

State equation

Now we head on to the state equation. In principle the appearing files and functions are of the same structure as for the adjoint equation. However one has to take care that, due to its nonlinear nature one has to set up a corresponding solver. In C++ the `NonlinearProblem` needed by the Newton solver is defined as follows.

```
class AllenCahnNLP : public NonlinearProblem
{
```

```
3      public:
4              AllenCahnNLP(AllenCahn_i::LinearForm& L,
5                           AllenCahn_i::BilinearForm& a):  _a(a), _L(L) {}
6
7              virtual void F(GenericVector& b, const GenericVector& x)
8              {
9                      assemble(b, _L);
10             }
11
12             virtual void J(GenericMatrix& A, const GenericVector& x)
13             {
14                     assemble(A, _a);
15             }
16
17     private:
18             AllenCahn_i::BilinearForm& _a;
19             AllenCahn_i::LinearForm&   _L;
20 };
```

The class AllenCahnNLP needs to override the virtual functions F and J provided by its base class. Here F should assemble in the vector b the equation whose root is looked for (in our case the left-hand side minus the right-hand side of the discrete Allen-Cahn equation) and J should set up its derivative. The forms we use to setup these are handed over in the constructor in line 4. Their definitions can be found in the following UFL code that to most part is similar to the one from the adjoint equation.

```
1  import imp
2  def_A = imp.load_source('', './def_A.py')
3
4  E = VectorConstant("triangle", def_A.nl)
5  P1 = FiniteElement("Lagrange", "triangle", 1)
6
7  dy = TrialFunction(P1)
8  v  = TestFunction(P1)
9
10 y   = Coefficient(P1)  # current solution
11 y0  = Coefficient(P1)  # solution from previous time step
12 u   = Coefficient(P1)  # control u(t)
13
14 tau       = Constant(triangle)
15 epsilon   = Constant(triangle)
16 delta     = Constant(triangle)
17
18 # first and second derivative of the potential
19 dphi  = -y*(1-y*y)
20 ddphi = 3*y*y-1
21
22 L = tau*dot(def_A.A_prime(grad(y), E, delta), grad(v))*dx + v*y*dx \
23     + tau/(epsilon*epsilon)*dphi*v*dx - v*(y0 + u*tau/epsilon)*dx
24 a = derivative(L, y, dy)
```

In line 22 we define the LinearForm that belongs to one time step of the discrete Allen-Cahn equation (2.60). Analogously to before, the solution from the previous time step y_{j-1} appears as the Coefficient y0. Note that also y is only a Coefficient as this form only appears on the right-hand side of the Newton equation. In contrast, the BilinearForm defined in line 24 contains the `TrialFunction dy`. It is simply given by the directional derivative of `L` into direction `dy` which is computed by the means of the FEniCS toolbox. With these definitions the built-in Newton solver is able to iteratively compute a solution to the nonlinear state equation. The complete code to solve the state equation for a given right-hand side u is given in the following listing.

```
void solve_state(std::shared_ptr<const Mesh> mesh, double* u, double* y_out)
{
        // initialize forms and set the required parameters
        auto V = std::make_shared<AllenCahn_i::FunctionSpace>(mesh);

        AllenCahn_i::BilinearForm a(V, V);
        AllenCahn_i::LinearForm L(V);

        auto y = std::make_shared<Function>(V);
        auto y0 = std::make_shared<Function>(V);
        auto u_t = std::make_shared<Function>(V);

        auto c_tau = std::make_shared<Constant>(pms.tau);
        auto c_epsilon = std::make_shared<Constant>(pms.epsilon);
        auto c_delta = std::make_shared<Constant>(pms.delta);

        a.tau = c_tau;
        a.delta = c_delta;
        a.E = E;
        a.epsilon = c_epsilon;
        a.y = y;
        L.tau = c_tau;
        L.delta = c_delta;
        L.E = E;
        L.epsilon = c_epsilon;
        L.y = y;
        L.y0 = y0;
        L.u = u_t;

        AllenCahnNLP NLP(L, a);

        // set up nonlinear solver
        PETScSNESSolver newton_solver(MPI_COMM_WORLD);
        newton_solver.parameters["method"] = "newtontr";
        newton_solver.parameters["linear_solver"] = "gmres";
        newton_solver.parameters["maximum_iterations"] = 200;
        newton_solver.parameters["relative_tolerance"] = pms.nonlin_tol;
        newton_solver.parameters["absolute_tolerance"] = pms.nonlin_globaltol;

        // set up intitial condition
        InitialState y_initial;
        *y = y_initial;
        set_data(*(y->vector()), y_out, 0, mesh);

        int i=1;
        for (double t = pms.tau; t <= pms.T+__EPS__; t += pms.tau)
        {
                *(y0->vector()) = *(y->vector());
                set_vector(*(u_t->vector()), u, i, mesh);
50
51          auto [nnorm,nconv] = newton_solver.solve(NLP, *(y->vector()));
52          if (!nconv)
53              printf("Newton did not converge!\n");
54
55          set_data(*(y->vector()), y_out, i, mesh);
56          i++;
57      }
58  }
```

Apart from the mesh, the function takes the control `u` and the array `y_out`, where the solution is stored. Note that the latter is typically identical to the array `y` handed over in the function `solve_adjoint` discussed previously. As before, first the `LinearForm` and `BilinearForm` are set up starting at line 3 until line 28. The `class AllenCahnNLP` defined in the previous listing is initialized in line 30 and used later in line 51 to solve the Newton equation appearing in each time step. The trust region solver used for this is defined in line 33 and configured in the lines that follow. The state equation is solved between lines 45 and 57, where each iteration of the for-loop corresponds to one time step.

Linearized and additional adjoint equation

The linearized and additional adjoint equation are implemented in functions that are called `solve_linearized` and `solve_additional_adjoint`. As the implementation is very similar to the adjoint equation, they will not be listed completely here. Note however that the operators $\mathcal{K}$, $\mathcal{K}^*$ and $\mathcal{P}$ (cf. (3.4)) are the same in each iteration of the Steihaug-CG solver. As their assembly needs to load the previously computed functions y and p as well as to compute the quadrature of nonlinear integrals, it makes sense to store them for later reuse. For the operator $\mathcal{P}$ (that only has to be computed in `solve_additional_adjoint`), it is not clear a priori if there will be any benefit from this, as it has to be combined with the solution from the previous time step anyway owed to the stepwise forward/backward substitution we implemented. This will be further investigated in Section 4.5. To conclude this subsection let us briefly take a look at how the function `solve_linearized` has implemented the discussed feature.

```
1   void solve_linearized(std::shared_ptr<const Mesh> mesh, double* y, double* du,
2                  double* dy_out, int iter)
3   {
4       static std::vector<Matrix> Mat_store_S(pms.n_timesteps+1);
5       [...]
6       for (t = pms.tau; t <= pms.T+__EPS__; t += pms.tau)
7       {
8           [...]
9           if (iter == 1)
10          {
11              set_vector(*(y_t->vector()), y, i, mesh);
12              assemble(Mat_store_S[i-1], a);
13          }
14          [...]
15          solver.solve(Mat_store_S[i-1], *(dy->vector()), b);
16          [...]
17      }
18    [...]
19  }
```

First of all, we have to pass the iteration count of the Steihaug-CG method in line 2. This is needed later in line 9 as the matrix has only to be assembled in the first iteration. For all other iterations we can readily use the matrix in line 15 as it is already stored in `Mat_store_S` then.

The latter is initialized in line 4 and is of type `static std::vector<Matrix>`. The attribute `static` ensures that it is only initialized once at startup and available at every function call with the modifications previously applied.
Finally note that the third derivative of A appearing in (1.27) is computed by automatic differentiation, i.e., in the ufl code we write

```
(derivative(derivative(defM.A_prime(grad(y),E,delta),y,v),y,dy),grad(p))*dx
```

Note that the order of the application of the various variables is important here as A''' is not guaranteed to be symmetric.

3.3.2 Parallelization

As we have seen, the algorithms presented in this chapter are computationally expensive as they require the repeated solution of several PDEs. Although we believe to have chosen the most suitable methods for our problem and also made use of the built-in preconditioning options of our PDE solving backend, due to the complexity of the present control problem computations may still take long time. Another possibility of speeding up the code in addition on the hardware level is to allow the use of multiple processing cores which will be subject of this section. For doing so we use the *Message Passing Interface* (MPI, [58]) that is also supported by FEniCS and is commonly used in scientific computing. This standard defines routines that coordinate communication between processes on a multiple instruction multiple data (MIMD) machine. In principle MPI is fully supported by FEniCS and the code will automatically run in parallel if it is started via

```
mpirun -n 4 ./a.out
```

where in this example the executable file `a.out` is run by 4 processes. In the case of optimal control we also would like to temporarily save the solutions of every time step of the appearing PDEs since they might reappear in another PDE or just to avoid repeated computations (for example in case the trust region radius is reduced). Further we would like to be able to store the computed solutions on a hard drive as a backup copy or for later resumption. Note that the vtu files produced by FEniCS are only meant for plotting and cannot be read back into the program offhand. Therefore, we pursue a different approach. Instead of storing the results in terms of the vector data type provided by the FEniCS framework we stored the binary data of the node values in an array. The latter is sufficient as we use the same uniform mesh in every time step, so the mesh information does not have to be stored separately. The data can be stored in a binary file by simply calling the C routine `fwrite`. Since every process only holds the data it uses, each of them has to write its own file in order to avoid unnecessary communication. The code snippet

```
sprintf(filename, "example_file_%i.bin", MPI::rank(mesh->mpi_comm()));
```

can be applied to avoid name collisions. The FEniCS interface however only allows to modify its built-in vector class component-wise or by passing a `std::vector<double>`. On the other side we store all time steps of a quantity in one array, so we only have the starting address of the storage belonging to a specific time step. To be able to efficiently use the `std::vector` interface of FEniCS, we use the following wrapper class template adapted from [82].

```
template <class T>
class vectorWrapper : public std::vector<T>
{
    public:
        vectorWrapper(T* sourceArray, int arraySize)
        {
            this->_M_impl._M_start  = sourceArray;
            this->_M_impl._M_finish = this->_M_impl._M_end_of_storage
                                    = sourceArray + arraySize;
        }

        ~vectorWrapper()
        {
            this->_M_impl._M_start = this->_M_impl._M_finish
                                   = this->_M_impl._M_end_of_storage = NULL;
        }
};
```

This allows the construction of a class compatible to `std::vector` out of existing storage without clearing it upon destruction. We note that its definition depends on the specific implementation of `std::vector` and might have to be adapted for other compilers or future revisions of the C++ standard library (we use the `g++` compiler in version 7.5.0). Neththeless this allows to seamlessly communicate with the interface of the `GenericVector` class without reallocating memory and invoking unnecessary copies. The following code demonstrates how to get the data of a `GenericVector` derivative and save it in an array of type `double*`.

```
1  void set_data(GenericVector& v, double* data, int step,
2                std::shared_ptr<const Mesh> mesh, int nTimesteps)
3  {
4          double* y_t = get_func_at_timestep(data, step, mesh, nTimesteps);
5          vectorWrapper<double> tmp(y_t, v.local_size());
6          v.get_local(tmp);
7  }
```

The code assumes that the array `data` consecutively stores every time step after another. Since the correct advancement of the pointer is a recurring task, it was outsourced into a function called `get_func_at_timestep`. The data is fetched in line 6 using the instance of `vectorWrapper` defined in the previous line from the computed memory address. We note that FEniCS makes sure that only the local memory of each process is read out here and apart from calling `GenericVector::local_size` we do not have to care about the local index structure. Furthermore, we remark that the above approach works because `GenericVector::get_local` does not change the memory allocated by the passed `std::vector`, since otherwise the `vectorWrapper` was corrupted. In a similar fashion we can also set the data of a `GenericVector`.

```
1  void set_vector(GenericVector& v, double* data, int step,
2                  std::shared_ptr<const Mesh> mesh, int nTimesteps)
3  {
4          double* y_t = get_func_at_timestep(data, step, mesh, nTimesteps);
5          const vectorWrapper<double> tmp(y_t, v.local_size());
6          v.set_local(tmp);
7          v.apply("");
8  }
```

Also here the call to `GenericVector::set_local` with the `vectorWrapper` argument in line 6 is crucial. The final invocation of `GenericVector::apply` is necessary to finalize the vector construction.

3.3.3 About mesh refinement

Another technique that is commonly applied to improve computational efficiency in context of numerics for PDEs is the use of an adaptive mesh strategy. By doing so, one is typically able to drastically decrease the number of degrees of freedom and hence computation time. This is especially amenable for phase field models, where in general only the interface has to be resolved sufficiently well, since in the remaining regions the solution stays almost constant. However, in context of optimal control there arise several additional challenges, which will be discussed in this subsection.

Generally speaking, mesh adaptivity requires a good estimator of the local discretization error based on which the decision regarding a possible refinement is taken. Like before, the space and time discretizations are usually considered separately. In case of phase field models, we refer to [17, 105, 52] for presentations of different approaches to adaptive space discretization. With respect to time, in this particular context no special adaptivity strategies are known to the author. The cited approaches work well as long as merely the forward problem without control is considered, since then the interfaces are expected to only change slowly between the time steps and topology changes are foreseeable at an early stage. This might no longer be true for non-vanishing right-hand sides. For instance, in the controlled case, new phases can appear out of nowhere which is not predicted by the above approaches early enough. For optimal control, the usual strategy therefore is to try to reduce errors in a special cost functional (that does not need to be $J(y, u)$), rather than looking at the state equation. We refer to [98] for a discussion in the parabolic case where space as well as time adaptivity is considered. These strategies generally apply to optimal control problems and are not specifically adapted for phase field models.

In any case, the question that arises is whether to use different meshes for the different quantities. For all PDEs most of the action is expected to take place in vicinity of the interfaces. Nevertheless, there still can be strong deviations from this. For instance consider the adjoint state which satisfies $p(T) = \frac{1}{\varepsilon}(y(T) - y_\Omega)$ by definition. At the beginning of the control procedure, $y(T)$ is expected to be significantly different from y_Ω and as a consequence an important part of the evolution of the adjoint equation will take place outside of the interface region of the state. This can be remedied by using a different mesh for the adjoint equation. The same holds true for the linearized equation and the additional adjoint equation if one were to use the trust region Newton method with Steihaug-CG presented in Sections 3.1.1 and 3.1.2. Using different meshes also has disadvantages however. One drawback is that the approaches first-discretize-then-optimize and first-optimize-then-discretize (cf. Section 1.1) may no longer coincide as the discretization spaces differ. Further, for the semi-implicit scheme that was introduced in Section 2.7, the discretization of the equations could not be derived from a discontinuous Galerkin approach. So it would not be clear how a reasonable discretization of the adjoint equation would look like in case the time steps differ from those of the state equation, since then the approach from Section 2.7.2 does no longer work and (1.24) would have to be discretized directly. Another problem is that the solutions have to be projected or interpolated into the appropriate spaces when appearing in another equation. On the one hand this leads to additional computational effort that might nullify the gains obtained from the reduction of degrees of freedom aimed at by adaptivity. On the other hand it might not always be clear how the interpolation should be done. As an example, we again mention the semi-implicit scheme where no clear correspondence of the solutions y_j to a specific time interval I_j is possible. This then also complicates the mapping between solutions of forward and backward problems.

The other approach would be to use the same mesh for all four appearing PDEs. Then one might run into the incompatibility problems described above, but after the optimization algorithm has established for a certain time, it might be a good approach to assume that the area of interest is the same for all equations. Instead of considering the state equation time step wise, one could also consider to successively solve the optimization problem for increasingly refined meshes as is also pursued in [98]. If one interpolates the old solution as a starting configuration for the refined run, the adjoint state should already live in the vicinity of the interface from the beginning. Some test cases performed for the isotropic Allen-Cahn equation indicated that also in this case the adaptive version performs inferior to the non-adaptive one, see [116]. The latter allows for various tweaks in the implementation with respect to efficiency that can be implemented instead then.

Finally, let us note that another question concerns the choice of the discretization of the control space. A common choice is to take the same space as that for the adjoint state motivated by the form of the gradient (1.25). However, in context of adaptivity this implies that one might

have a different discrete control space in each step of the optimization procedure. With that, it is in general not clear that the optimization algorithm converges at all.
Due to the significant amount of extra work one has to put into the implementation and the questionable increase in performance that can be expected, we ultimately decided not to employ mesh refinement in our programs. Instead we resort to parallelization as described in Section 3.3.2 and implemented various optimizations that are possible for a non-adaptive version, like reusing some already computed quantities. As mentioned before, we do not expect any drawbacks according to the observations of [116]. The situation could be different if we considered smaller ε as then also the space discretization has to be finer, leading to much more degrees of freedom if we would not use an adaptive mesh. Also for possible simulations in three space dimensions one should think about adaptivity again.

4

Numerical results

In the final part of this thesis we want to report the numerical results for the optimal control of the anisotropic Allen-Cahn equation that were achieved with the presented method. Let us note that in contrast to chapter 2, here we fix ψ to consistently be the smooth double-well potential $\psi(s) = \frac{1}{4}(1-s^2)^2$ which corresponds to $C_\psi = 1$ in Assumptions 2.1.1 and 2.4.1. For the anisotropy A we consider various choices that will be specified below, but are all of the form introduced in Section 2.6. Furthermore, in constrast to Section 2.5, to keep the presentation lucid we drop the index δ in the relevant quantities as long as we consider this parameter fixed. In fact, all simulations in this chapter, apart from the isotropic ones, were computed with a regularized anisotropy.
Preliminary numerical results have been obtained by a line search method based on the gradient ∇j_τ as described in Section 3.1.1. This however performed rather poor for the present optimization problem. For instance, the simulation from Figure 4.18b later only took 37 iterations with the trust region method. In Figure 4.1, we see the first 5000 steps obtained from applying the

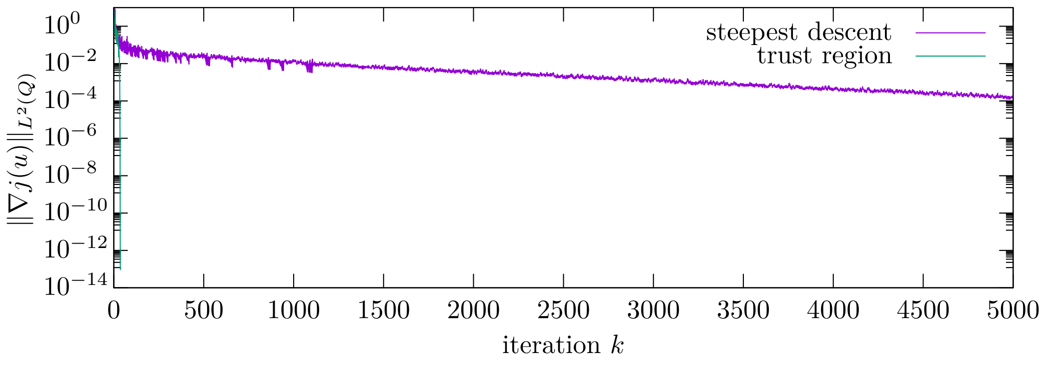


Figure 4.1: Comparison of the convergence of the steepest descent method and the trust region Newton method for the setting from Figure 4.18b.

steepest descent method to the very same setting. As expected we observe linear convergence, but the rate of error reduction is very small. In case of the isotropic Allen-Cahn equation, for a more detailed comparison we refer to [116] (see also [24]). Due to these performance deficiencies, we stayed with the trust region solver as we have not seen any relevant differences in the computed controls and states.

Before we will present the actual numerical results, we will therefore first introduce the additional equations that are needed to incorporate the second order information that is needed for the trust region Newton method. In contrast to the analysis in Section 2.4 they are only given formally here. After that we will specify the matrices G_l from Section 2.6 that are needed to define the utilized anisotropies and will also provide the values of the remaining parameters that are used for the simulations. Most of them stay fixed throughout the simulations if not stated otherwise. In the following sections we will then discuss various outcomes of the numerical simulations. First we will give support for the convergence result of Theorem 2.5.1 concerning the regularization. Then, we will provide numerical evidence for mesh independent behavior in the solution process and we present optimal control results for different anisotropies and different desired states, including star like objects and topology changes. This is followed by a discussion of some of the quantities that give some information about the solution process of the algorithm. In the end, we compare some of the implementational issues that were brought up in course of the thesis, especially with regard to execution speed. For instance, we will compare our implicit scheme to the semi-implicit one from Section 2.7, discuss the benefit of storing quantities from the linearized equations as discussed by the end of Section 3.3.1 and investigate the usefulness of parallelization in our context.

Linearized equations

Let us briefly summarize the main formulas that are needed to understand the trust region Newton method with Steihaug-CG from Sections 3.1.1 and 3.1.2 in context of the optimal control of the anisotropic Allen-Cahn equation. As a quick overview one might also consult the pseudocodes from listings 2 and 3, where in our case $U = L^2(Q)$.
In this thesis the trust region method is used to find a local optimizer of the reduced cost functional

$$j_\tau : U_\tau \to \mathbb{R}, \quad j_\tau(u_\tau) = \frac{1}{2}\|y_\tau(T) - y_\Omega\|^2 + \frac{\lambda}{2\varepsilon}\|u_\tau\|^2,$$

by solving the first order condition $\nabla j_\tau(u_\tau) = 0$ with

$$\nabla j_\tau(u_\tau) = \frac{\lambda}{\varepsilon}u_\tau + p_\tau.$$

Here $y_\tau = S_\tau(u_\tau)$ is given by the state equation (2.60) and $p_\tau = (S'_\tau(u_\tau))^*(y_\tau(T) - y_\Omega)$ is the solution of the time discrete adjoint equation (2.77). Recall that the trust region method extends the established Newton method by a globalization strategy that is needed due to the nonlinearity of the state equation, and hence the non-convexity of $j_\tau(u_\tau)$.
The Steihaug-CG method [123] determines in each step the approximate solution $\delta\bar{u}_\tau$ of the quadratic subproblem at the current iterate u_τ given by

$$\min_{\|\delta u_\tau\| \le \sigma} (\nabla j_\tau(u_\tau), \delta u_\tau) + \tfrac{1}{2}(\nabla^2 j_\tau(u_\tau)\delta u_\tau, \delta u_\tau). \tag{4.1}$$

The new iterate is then set to $u_\tau + \delta\bar{u}_\tau$.
The second derivative that we require here, in particular its action on an $L^2(Q)$-function, can be deduced formally like in Section 2.7.2. Since we do not need it at other places, we will be content with that. As expected, its general form is given by

$$\nabla^2 j_\tau(u_\tau)\delta u_\tau = \frac{\lambda}{\varepsilon}\delta u_\tau + \delta p_\tau. \tag{4.2}$$

Here, for the given solution $p_\tau(u_\tau)$ of (2.77), the derivative

$$\delta p_\tau := \frac{dp_\tau(u_\tau)}{du_\tau}\delta u_\tau$$

shall fulfill the additional adjoint equation, which in time discrete form is given by $\delta p_{N+1} := \frac{1}{\varepsilon}\delta y_N$

and

$$\varepsilon(A_\delta''(\nabla y_j)\nabla\varphi,\nabla\delta p_j)+(\tfrac{1}{\varepsilon}\psi''(y_j)\varphi+\tfrac{\varepsilon}{\tau_j}\varphi,\delta p_j)=(\varphi,\tfrac{\varepsilon}{\tau_j}\delta p_{j+1})$$
$$-\varepsilon(A_\delta'''(\nabla y_j)[\nabla\varphi,\nabla\delta y_j],\nabla p_j)-\tfrac{1}{\varepsilon}(\psi'''(y_j)\varphi\delta y_j,p_j)\qquad\forall j=1,\dots,N,\ \ \varphi\in H^1(\Omega). \tag{4.3}$$

As for (2.77) the unique existence of the solution to (4.3) is guaranteed in each time step as long as the right-hand side is well defined. The time continuous counterpart can be found in (1.27) and reads in variational formulation as

$$\begin{aligned}&-\varepsilon(\eta,\partial_t\delta p)+\varepsilon(A_\delta''(\nabla y)\nabla\eta,\nabla\delta p)+\frac{1}{\varepsilon}(\psi''(y)\eta,\delta p)=\\&\qquad-\varepsilon(A_\delta'''(\nabla y)[\nabla\eta,\nabla\delta y],\nabla p)-\frac{1}{\varepsilon}(\psi'''(y)\eta\delta y,p)\qquad\forall\eta\in L^2(0,T;H^1(\Omega)),\\&\delta p(T)=\frac{1}{\varepsilon}\delta y(T)\quad\text{in }\Omega.\end{aligned}\tag{4.4}$$

The appearing linearized state δy_τ is determined as the unique solution to the linearized state equation (2.74) with $v=\delta u_\tau$, i.e.

$$\varepsilon(A_\delta''(\nabla y_j)\nabla\delta y_j,\nabla\varphi)+(\tfrac{1}{\varepsilon}\psi''(y_j)\delta y_j+\tfrac{\varepsilon}{\tau_j}\delta y_j,\varphi)=(\delta u_j,\varphi)+\tfrac{\varepsilon}{\tau_j}(\delta y_{j-1},\varphi)\qquad\forall j=1,\dots N,\ \ \varphi\in H^1(\Omega). \tag{4.5}$$

Analogous to before, this equation in essence is given by the derivative of y_τ in direction of δu_τ, i.e.

$$\delta y_\tau:=\frac{dy_\tau(u_\tau)}{du_\tau}\delta u_\tau.$$

The time continuous counterpart reads in variational formulation as (cf. eqs. (1.26) and (2.75))

$$\begin{aligned}\varepsilon(\partial_t\delta y,\eta)+\varepsilon(A_\delta''(\nabla y)\nabla\delta y,\nabla\eta)+\frac{1}{\varepsilon}(\psi''(y)\delta y,\eta)&=(\delta u,\eta)\qquad&&\forall\eta\in L^2(0,T;H^1(\Omega)),\\\delta y(0)&=0&&\text{in }\Omega.\end{aligned}\tag{4.6}$$

Note that eqs. (4.5) and (4.6) as well as eqs. (4.3) and (4.4) are related by the discontinuous Galerkin time discretization as for the state and adjoint equations. With that we have that each of the discretized equations has a time continuous counterpart (cf. eqs. (1.9), (2.76), (4.4) and (4.6)). As described in Section 1.1, eqs. (2.76), (4.4) and (4.6) are the equations you would expect to get as the adjoint and corresponding linearized equations for (1.9). Thus at least on a formal level it holds that the approaches *first discretize then optimize* and *first optimize then discretize* commute for the implicit time discretization in the sense of [79, chapters 3.2.2 and 3.2.3]. Also discretization and optimization are interchangeable for the spatial discretization if one chooses the same ansatz spaces for y and p (see also the discussion in Sections 1.1 and 3.3.3). Consequently, one expects to obtain for the optimization solver iteration numbers independent of the discretization level. This is also strengthened by the numerical observations in Section 4.2. Note that in contrast to A_δ'', in general the 3-tensor A_δ''' appearing in (4.4) is not symmetric, so we had to keep care of order in the corresponding term. In the implementation, A_δ''' is determined by automatic differentiation (cf. the very last code snipped in Section 3.3.1)—stating an explicit formula does give no new insight. Moreover, since $\nabla y=0$ in the pure phases one is particularly interested in the behavior of $A_\delta'''(q)$ when $q\to 0$. However, $\tilde{A}'''(\sqrt{\delta}\binom{q}{1})$ behaves like $1/\sqrt{\delta}$ due to the 2-homogeneity of $\tilde{A}$ (see (2.101)). Hence $\lim_{q\to 0}A_\delta'''(q)$ cannot be bounded independently of δ. Numerically we face this problem for values $\delta<10^{-13}$.

Standard choice of the parameters

In the following we will expose the standard setup that we used as a basis for all simulations. If not mentioned otherwise, the values given here are used in the considered example. To keep the computational cost moderate we set $d=2$ in all experiments. Furthermore, throughout this section we use as a spatial domain the square $\Omega=(-1,1)^2$. For the time horizon $[0,T]$ we

choose $T = 1.625 \cdot 10^{-2}$ and we set the parameters $\varepsilon = \frac{1}{14\pi}$, $\lambda = 0.01$ and the regularization parameter $\delta = 10^{-7}$. We choose the constant time step size $\tau = 1.625 \cdot 10^{-4}$, which fulfills the condition $\tau \leq \varepsilon^2 / C_\psi$ and Ω is uniformly discretized with 129×129 grid points. Moreover by numerical evidence we know that the interface is resolved sufficiently with 6-14 mesh points across its expansion.
For the optimization solver the settings are as follows. As stopping criterion we use an absolute tolerance of 10^{-13} for the trust region method and for the Steihaug-CG solver we use a relative tolerance of 10^{-10} and an absolute tolerance of 10^{-13}. Furthermore, we allow for a maximum amount of 800 CG steps per trust region iteration. The initial trust region radius is set to $\sigma = 1$ and the upper bound is given by $\sigma_{\max} = 100$. Finally, each computation started with $u^{(0)} \equiv 0$. Throughout this section, we will look at various setups that mainly vary by y_0, y_Ω and the final time T. For the anisotropies determined by G_l (cf. eqs. (2.91) and (2.92)) we use three different choices, that are listed in the following.

1. isotropic case: $\gamma(p) = \|p\|_2$ this would belong to the choice $L = 1$ with
$$G_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{4.7}$$
Note that in this case regularization is not necessary. In addition, the regularization would cancel out as can be seen in (2.98) and as it is also discussed in the corresponding text thereunder.

2. regularized l_1-norm: This is given by $L = 2$ and
$$G_1 = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix}, \qquad G_2 = \frac{1}{2}\begin{pmatrix} \epsilon & 0 \\ 0 & 1 \end{pmatrix}, \tag{4.8}$$
with some small parameter ϵ that we set to $\epsilon = 0.01$ (not to be confused with the interface parameter ε). For $\epsilon = 0$ and without the scaling of $\frac{1}{2}$, this would reduce to the ordinary l_1-norm.

3. form of a smoothed hexagon: Here $L = 3$ and
$$G_l = \frac{1}{3}\begin{pmatrix} \cos(\alpha_l) & -\sin(\alpha_l) \\ \sin(\alpha_l) & \cos(\alpha_l) \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix}\begin{pmatrix} \cos(\alpha_l) & \sin(\alpha_l) \\ -\sin(\alpha_l) & \cos(\alpha_l) \end{pmatrix}, \tag{4.9}$$
where $\alpha_l = \frac{\pi}{3} l$, $l = 1, 2, 3$ and ϵ as before.

Remark 4.0.1. Note that in general there can be more than one combination of matrices that lead to the same outcome. For example, for arbitrary L the first choice is equivalent to using
$$G_l = \begin{pmatrix} \frac{1}{L^2} & 0 \\ 0 & \frac{1}{L^2} \end{pmatrix} \qquad \text{for } l = 1, \dots, L,$$
as can quickly be verified by inserting this into (2.91). Of course one should not use this version as it provides much more computational overhead since the algorithm does not simplify automatically. Nevertheless, this formulation can be used to check the correctness of the equations that were enhanced to contain the anisotropy. Note however, that the results might differ slightly though as the versions for different L are no longer completely the same after discretization.

In contrast to the choices in [14] we divided the matrices by their total amount L. By this scaling the costs between the different anisotropies becomes more comparable, since by numerical observation the velocity of the shrinkage is approximately equal. This can also be seen by looking at the Wulff shapes, which are defined as
$$\mathcal{W} := \left\{ q \in \mathbb{R}^d : \sup_{p \in \mathbb{R}^d \setminus \{0\}} \frac{p^T q}{\gamma(p)} \leq 1 \right\},$$

see [138], and whose boundary can be parametrized as

$$z_{\mathcal{W}}(z) := \gamma'(z) = \sum_{l=1}^{L} \frac{G_l z}{\gamma_l(z)} \qquad \text{for} \qquad z \in S^{d-1},$$

see [69]. In Figure 4.2 they are visualized for the above choices of γ. Without the rescaling the Wulff shape of the hexagon anisotropy would extend approximately to the label 2.0 on each axis, with the rescaling they have about the same extension as can be observed in the figure. For completeness, we give also the corresponding Frank diagrams that are defined by

$$\mathcal{F} := \left\{ q \in \mathbb{R}^d : \gamma(q) \leq 1 \right\},$$

and whose boundary can be parametrized by

$$z_{\mathcal{F}}(z) := \tfrac{z}{\gamma(z)} \qquad \text{for} \qquad z \in S^{d-1},$$

see again [69]. They can be looked at in Figure 4.3.

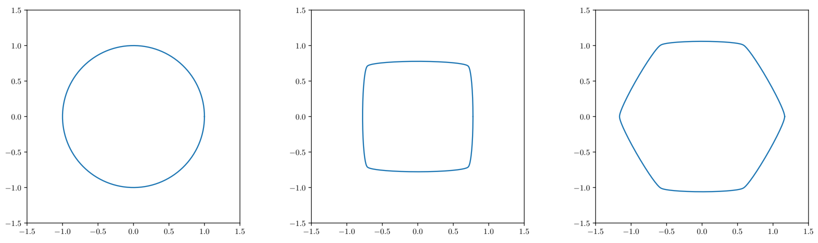


Figure 4.2: Wulff shapes of isotropic, l_1- and hexagon-anisotropy settings.

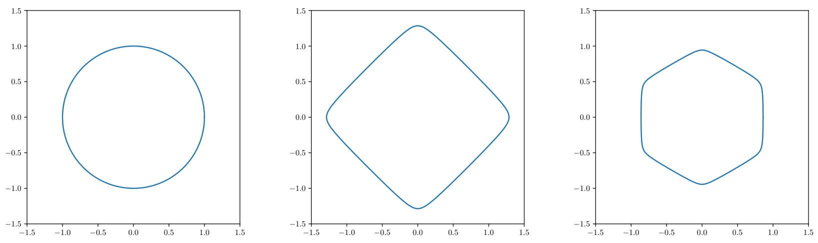
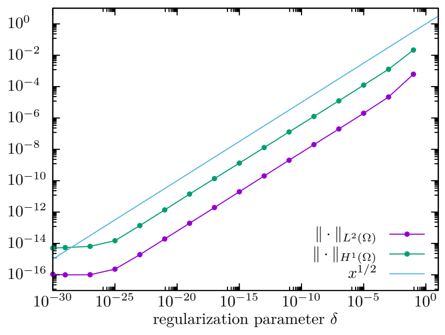


Figure 4.3: Frank diagrams of isotropic, l_1- and hexagon-anisotropy settings.

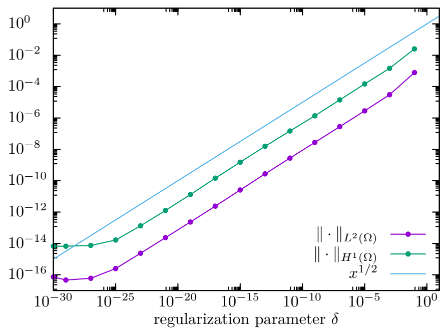
Recall from Section 3.3.1 that as computing framework we used FEniCS [5] or rather its C++ interface DOLFIN [91]. The version number thereof was 2019.1.0. The simulations were carried out on an HP EliteDesk 800 G4 workstation containing an Intel Core i7-8700 CPU with 12 cores à 3.20GHz and 16 GB of RAM.

4.1 Dependence on the regularization parameter δ

Here we analyze numerically the dependence of the solution of the state equation on the parameter δ. As a setting we start from a circle of radius 0.5 and look at the evolution of the state only using $u = 0$. Here T and the remaining parameters remain the same as given in the introductory section before. A plot containing the results for the choices of both anisotropies respectively is given in Figure 4.4. Recall from above that for the isotropic case the regularization drops out, so it does not make sense to discuss it here. We have plotted both the difference of the states in $L^2(\Omega)$ as well as $H^1(\Omega)$ at the end point T. Since errors accumulate during the time evolution, the deviation at T should be a good metric for comparison. As one would expect, the values for the $H^1(\Omega)$-norm are steadily bigger than those for the $L^2(\Omega)$-norm. Apart from that, the values between both anisotropies seem to be very similar. With the additionally plotted function $f(x) = x^{1/2}$ (i.e. a line with slope 1/2 on the log-log-plot) the figure clearly exhibits the convergence order 1/2 which is expected according to eqs. (2.82) and (2.102), i.e., it equals the approximation order of A'_δ to A'. One can observe deviations from the straight line for values below 10^{-25}. Here errors arising from computational inaccuracies dominate. However, the optimization algorithm may break down earlier as it also involves the adjoint and linearized equations.



(a) Results for the regularized l_1-norm.



(b) Results for the hexagon anisotropy.

Figure 4.4: Comparison of $\|y^\delta_\tau(T) - y_\tau(T)\|$ in the $L^2(\Omega)$- and $H^1(\Omega)$-norms for different values of δ.

Furthermore, also with regard to the next section, let us mention that according to our experience there is at most weak dependence of the number of the trust region as well as of the Steihaug-CG iterations when varying δ. They stay nearly the same as in Tables 4.1 and 4.2 and are therefore not listed there. If δ is such small that rounding errors accumulate for A''_δ and even more for A'''_δ (and consequently for the solutions of the respective equations) the algorithm may not converge. However, for $\delta \gtrsim 10^{-10}$ the algorithm was always robust.

4.2 Mesh independent behavior

In this section we investigate numerically if the problem solver depends on the granularity of the mesh. Recall from Sections 3.1.3 and 3.2.2 that the Steihaug-CG method should be independent of the time steps size and the spatial discretization. For the trust region solver we do not have a theoretical result. More concretely, in this section we look at the number of trust region iterations, called *TR steps* in the following tables, as well as at the number of Steihaug-CG steps that are needed to solve the quadratic subproblems. Since the Steihaug algorithm consists of early stopping criteria given by the trust region algorithm, the amount of Steihaug-CG steps might vary drastically over the progress of the algorithm. Also, given that we have a fixed absolute tolerance, this amount can depend strongly on how close we are to the actual solution, since this has an influence on the initial residual. During the end of the outer trust region method, the Steihaug-CG algorithm already starts with a small residual. Therefore only looking at the average amount of Steihaug steps is no good metric to measure performance of the method. We rather look at the average amount of steps, called *mean CG* in the following tables, that are needed to decrease the residual by 6 orders of magnitude. This is only done for trust region steps where this kind of measurement is possible (i.e. span a proper amount of magnitudes). In addition we also take the maximum amount, called *max CG* as an indicator. These numbers of CG iterations reflect on the conditioning of the linear systems corresponding to the quadratic subproblems. To avoid confusion we point out that the results of the numerics section were computed without extra preconditioning. The mesh independence observed here is rather a consequence of considering the reduced Hessian approach (see Sections 3.1.3 and 3.2.2). As final time we choose in these experiments $T = 2 \times 10^{-3}$. The deviating choice for T just comes from the fact that it is easier to vary the time step size with it in the following. The remaining parameters are left unchanged. We inspect the dependence on the space discretization by fixing $\tau = 10^{-4}$ and varying $h = 2/N$. For analyzing the dependence on the step size τ we fix the spatial mesh size using $N = 128$.

As model problem for the isotropic case, we consider the control of a circle from radius $r = 0.5$ to $r = 0.55$. The corresponding results can be found in Table 4.1. One cannot observe a clear tendency that would suggest dependency of the maximal or mean number of CG iterations and of the trust region steps on the granularity, as is expected by the discussion from earlier in this numerics section. Only the amount of total computing time clearly increases with the number of unknowns which is owed to the growing computational costs. Let us mention that in case of $\tau = 10^{-4}$ and $N = 516$ the reduced optimization problem has around 5.4 million unknowns given by the amount of discretization points of u. If $\tau = 10^{-6}$ and $N = 128$ the number of unknowns is roughly 33.3 million. Note that the column associated with $N = 128$ of the left table presents the same data as the column associated with $\tau = 10^{-4}$ of the right table. For reproducibility we point out that due to the parallelization of the algorithm and the non-commutativity of floating point operations the results might vary slightly among runs sharing the same configuration.

N	64	128	256	512
max CG	38	48	38	39
mean CG	21.2	22.7	18.8	20.4
TR steps	12	16	11	12
time (s)	17	72	235	1196

τ	10^{-4}	$10^{-4.5}$	10^{-5}	$10^{-5.5}$	10^{-6}
max CG	48	60	34	34	34
mean CG	22.7	22.2	18.4	18.0	18.5
TR steps	16	11	9	8	8
time (s)	72	105	213	706	2032

Table 4.1: Dependence on N and τ for the isotropic case.

Next we do the same analysis for the anisotropic Allen-Cahn equation with the regularized l_1-norm. Here we choose y_0 and y_Ω to be identical, i.e., we try to keep a square constant. The outcomes are listed in Table 4.2. Again, almost no dependence on the discretization parameters is observed. Numbers for the average Steihaug steps as well as for the total trust region steps rather seem to ameliorate for more accurate computations. The values for the Steihaug iterations are comparable to the isotropic case—the table comparing different τ may show less tendencies but the second columns of Table 4.1 may just be a fluctuation. Only the amount of trust region steps is generally lower (but still shows a relatively constant behavior across the row). However, since we chose another configuration that is better adapted to the present anisotropy, a deviation here is expected.

N	64	128	256	512
max CG	60	40	40	39
mean CG	30.0	22.0	21.3	21.0
TR steps	10	6	6	6
time (s)	24	66	194	1193

τ	10^{-4}	$10^{-4.5}$	10^{-5}	$10^{-5.5}$	10^{-6}
max CG	40	40	39	35	35
mean CG	22.0	21.8	24.0	20.0	20.7
TR steps	6	6	7	5	5
time (s)	66	161	537	1127	3306

Table 4.2: Dependence on N and τ for the regularized l_1-norm where $y_\Omega = y_0$.

Finally, in Table 4.3 we list the outcomes for the control of a circle to a star with four fingers, as we want to see if the results also hold for more involved simulations. To get an impression of the utilized initial and final state, the reader can consult Figure 4.5. In this case a more fine-grained control has to be computed which is also reflected by the solution process which now takes significantly more trust region steps. Also, compared to Table 4.2 the number of CG iterations is increased. While this again indicates a dependency on the control configuration, the results concerning the CG iterations still hint at a behavior independent of the discretization level. A slight increase in the number of trust region steps towards finer meshes is present.

N	64	128	256	512
max CG	185	207	243	167
mean CG	129.0	127.7	139.7	117.0
TR steps	71	83	106	125
time (s)	95	641	4068	23657

τ	10^{-4}	$10^{-4.5}$	10^{-5}	$10^{-5.5}$	10^{-6}
max CG	207	180	175	181	175
mean CG	127.7	153.0	148.0	130.5	137.3
TR steps	83	95	146	126	108
time (s)	641	1462	5476	13753	46297

Table 4.3: Dependence on N and τ for the simulation circle to 4-star for the regularized l_1-norm.

4.3 Numerical examples for different desired states and anisotropies

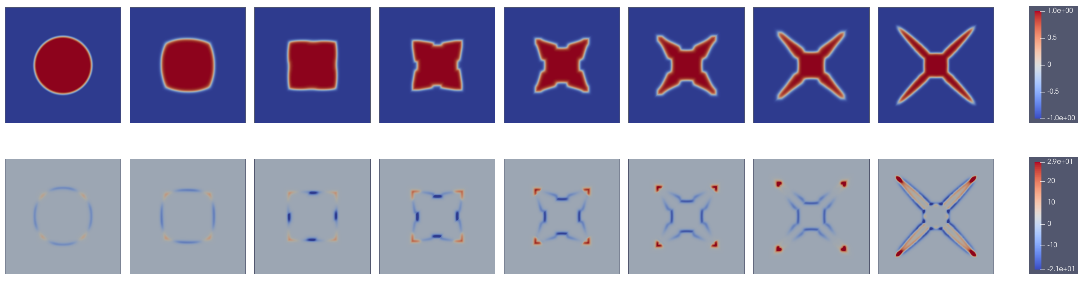
In the following subsection we present the solutions for four different objectives: the evolution to star-like structures, the splitting as well as the merging of geometries and finally the filling of the entire domain. For the figures showing the evolution of the control u we employed a scaling of the color that was adjusted to the values at $t \approx T/2$. Hence this allows to see where in Ω the control is present although its values may be clipped on some images. However adjusting the range for each image individually would lead to a very confusing presentation. Rather we in addition include figures showing the $L^2(\Omega)$-norm of the control over time in order to see how much the system is controlled at which time step.

4.3.1 Evolution to star-like structures

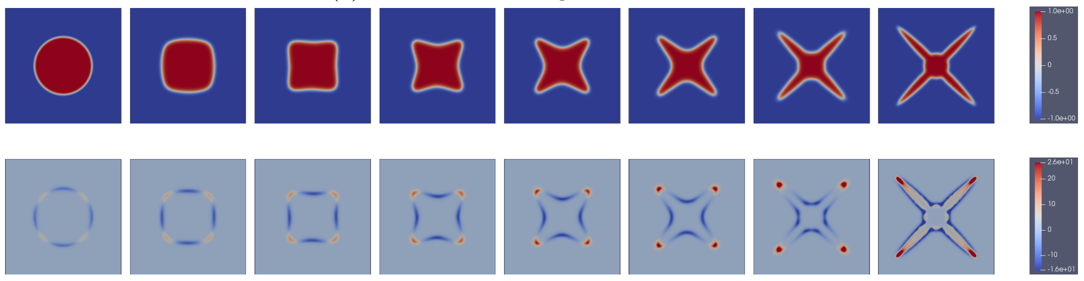
In the first experiment we start from a circle of radius 0.5 and try to steer it to a star-like structure with 4 or 6 fingers respectively. The images of the time evolution of the corresponding states and controls can be found in Figures 4.5 and 4.6. For the '4-star' target, the solution of the state equation is given by the Allen-Cahn equation with the regularized l_1-norm anisotropy, and for the '6-star' target with the 'hexagon' anisotropy. In addition we present the results in both cases for the isotropic evolution equation for comparison. In practice the choice of the version of the Allen-Cahn equation is given by the model equation and not by the desired state. Qualitatively the main observation is that in all cases the control takes place in a neighborhood of the interface. Moreover, the evolution is controlled essentially in the second half of the time interval. This can be particularly seen in Figure 4.7 where the $L^2(\Omega)$-norms of the control are plotted over the time t. On the t-axis we indicated the times at which the states and controls were sampled for Figures 4.5 and 4.6. One can observe that the control cost gets bigger during the time evolution. While for the isotropic case the middle part seems to grow nearly linearly, for the cases of the regularized l_1-norm and the hexagon anisotropy one observes bigger jumps intersected by approximately constant parts. The first plateau comes from the fact that at the first part the evolution follows the nearly uncontrolled Allen-Cahn flow to get a square-like, respectively a hexagon-like shape. Only then the control truly enters to initiate the development of the petals with the strongly non-convex parts. From that point onwards more control is needed for the anisotropic cases than for the isotropic case but towards the end they approximately overlap. The last sharp increase arises from the fact that much of the control is spent to form the details of the excrescences in the last few time steps. Finally we observe that the intermediate states show strong characteristics of the underlying anisotropy. For the isotropic cases the contours have a much smoother appearance than for the anisotropic ones. In comparison to that, the 'hexagon' states in Figure 4.6a for instance have small tips on their petals. These show exactly into the direction of the corners of the Wulff shape.

	4 star		6 star	
	iso	$l1$	iso	hexa
$j(u)$	0.102184	0.107378	0.115034	0.12248
$j_1 + j_2$	0.0115987 + 0.0905854	0.0108916 + 0.0964865	0.0122366 + 0.102798	0.0156595 + 0.106821

Table 4.4: Values of the cost functional.

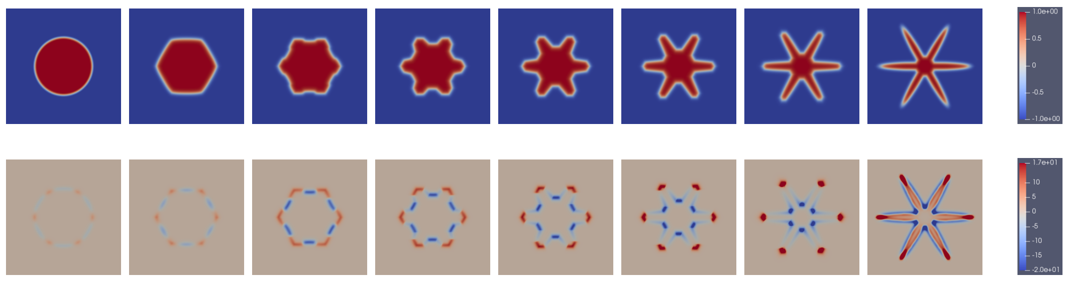


(a) Results for the regularized l_1-norm.

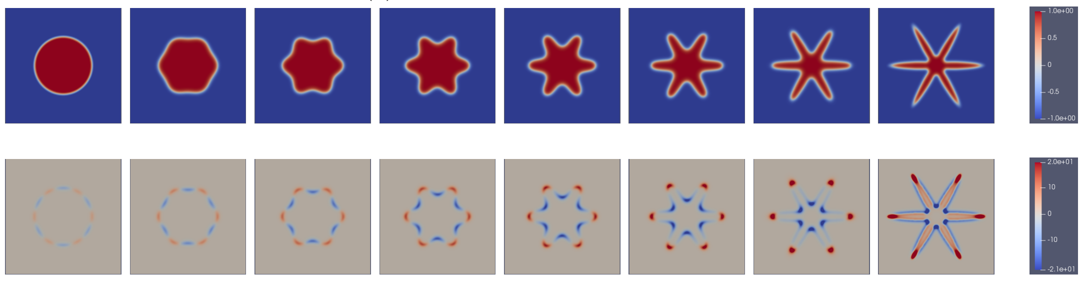


(b) Results for the isotropic case.

Figure 4.5: ‘circle to 4-star’ solutions: states in the first and third row and controls in the second and fourth row .

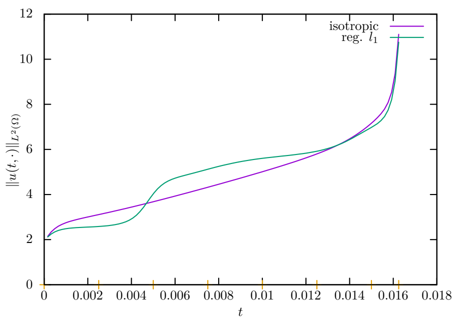


(a) Results for the hexagon case.

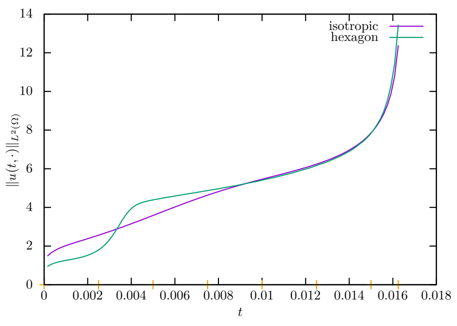


(b) Results for the isotropic case.

Figure 4.6: ‘circle to 6-star’ solutions: states in the first and third row and controls in the second and fourth row.

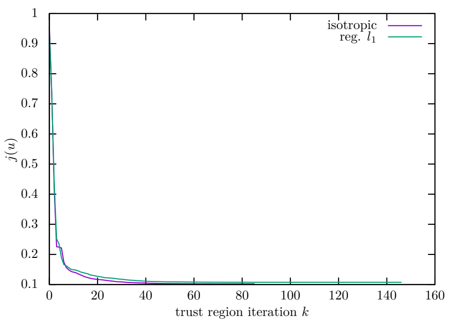


(a) Results for 'circle to 4-star', cf. Figure 4.5.

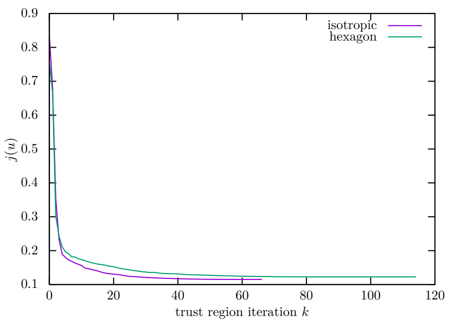


(b) Results for 'circle to 6-star', cf. Figure 4.6.

Figure 4.7: Time evolution of $\|u(t,\cdot)\|_{L^2(\Omega)}$.



(a) Results for 'circle to 4-star', cf. Figure 4.5.



(b) Results for 'circle to 6-star', cf. Figure 4.6.

Figure 4.8: Evolution of $j(u)$ over the trust region iterations k.

In Table 4.4 the values of the computed (local) minima are listed together with their single constituents—the difference of the optimal state to the desired state $j_1 = \frac{1}{2}\|y(T) - y_\Omega\|^2_{L^2(\Omega)}$ as well as the contribution of the control $j_2 = \frac{\lambda}{2\varepsilon}\|u\|^2_{L^2(Q)}$. We note that the (local) optima for the isotropic case are slightly below their anisotropic counterparts. We also observe that in all cases the values for j_2 are about a magnitude higher than those of j_1. As for u the norm over the whole time horizon is considered, this seems nevertheless a balanced ratio. Further, for our purposes it is important that the error in the end time point is small so we don't matter about having a little bit more cost on the control side.

In Figure 4.8 the corresponding plots of $j(u)$ against the trust region iterations k are given. One can observe that a value close to the optimum is soon reached after a rather steep decrease in the first few iterations. So most of the running time is seemingly taken to optimize some rather small details in the control of the state equation. Further, the lines nearly overlap as one would also expect from watching Table 4.4, as the local minima are very comparable in magnitude. At most one can say that after some time the line for the isotropic case lies constantly below the other, but this is a consequence of the fact that the algorithm generally converged earlier for the isotropic case here.

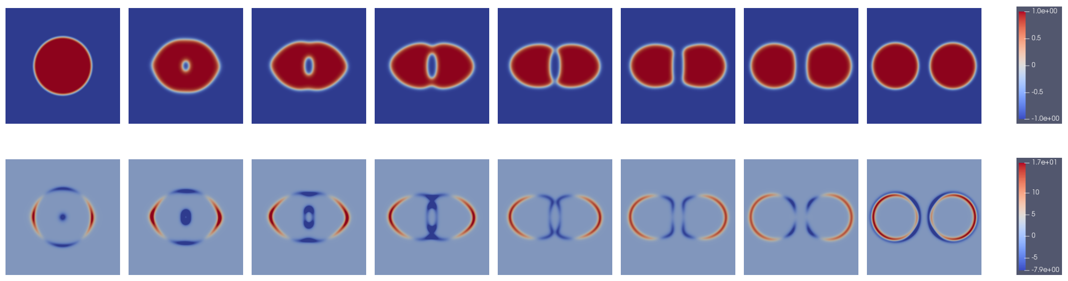
4.3.2 Splitting and merging geometries

Next we take a look at examples where topology changes are necessary to aim at the target. In Figure 4.9 we present the results for a simulation where we try to split a circle, a square and a hexagon into two of such respectively. The underlying model equation uses the (an-)isotropy with the Wulff shape corresponding to the initial value. In Figure 4.10 the solutions of merging two of these objects into one are given. Here the target objects are the initial states of the splitting examples and vice versa. The norms of the corresponding controls over the time can be seen in Figure 4.11. To avoid potential confusion, we point out that the scales of the ordinates are adapted to better fit the plots.

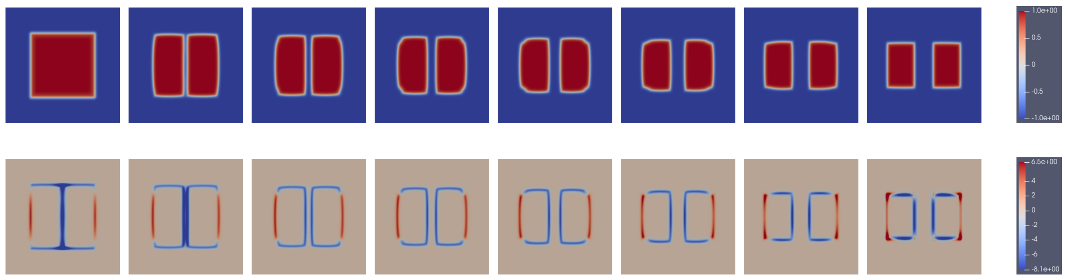
While the hexagon is split by squeezing it together vertically, the circle is controlled to develop first a hole in the middle and then to increase the hole until the split is present. The square is divided at the whole middle line simultaneously—as far as we could see visually. The controls are largest at times where they force topology changes as can be observed in Figure 4.11a.

Considering the examples for 'merging' (see Figure 4.10) we observe a very similar behavior of the states as for the 'splitting' solutions, only backwards in time. There is less control necessary which is indicated by the values for j_2 in Tables 4.5 and 4.6 where the cost functionals are given as for the star like examples before. For the isotropic and hexagon example the splitting cost is higher by a factor of approximately 1.5. That the difference is not bigger is probably due to the short time interval that forces the evolution of the gradient flow to be accelerated to obtain the target in time—a phenomenon present for splitting as well as for merging, with comparable impact. This also leads to the nearly constant time behavior for a long period that can be seen in Figure 4.11b. Further note that the value of $j(u)$ is mainly driven by j_2, since the value of j_1 is comparatively small. This is pleasing as we would like the algorithm to approximate the end time point equally well independent of the special situation. Therefore the costs should only differ on the intermediate way that can vary strongly between different settings. The reduction at the end of Figure 4.11b, that is most notable for the square example, is due to the agreement of the merging with the uncontrolled Allen-Cahn evolution. The steep peak at the end is explained by the adjustment of small details as was also the case in the earlier examples. Also notable is the almost linear increase for the isotropic example in the beginning starting at about 0.002. Comparing with Figure 4.10 lets us conclude that this is due to the deformation of the circle to build two prongs at the top and bottom, which strongly opposes the natural Allen-Cahn flow. As soon as the two circles touch at these two points (see the fourth snapshot) the control cost decreases again.

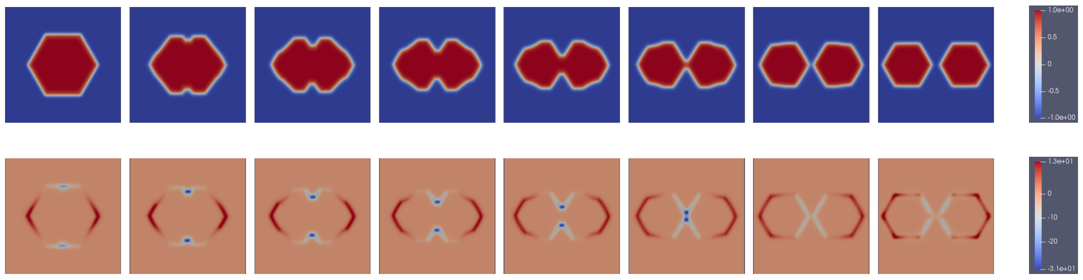
For completeness we also give the values of $j(u)$ over the course of the trust region algorithm in Figure 4.12. The behavior is the same as already explained in the previous subsection. In Figure 4.12a the lines are a little bit more distinguishable but also the optimum is in Table 4.5.



(a) Result for the ‘splitting circle’ in the isotropic case.



(b) Result for the ‘splitting square’ with the regularized l_1-norm.



(c) Result for the ‘splitting hexagon’ using the hexagon anisotropy.

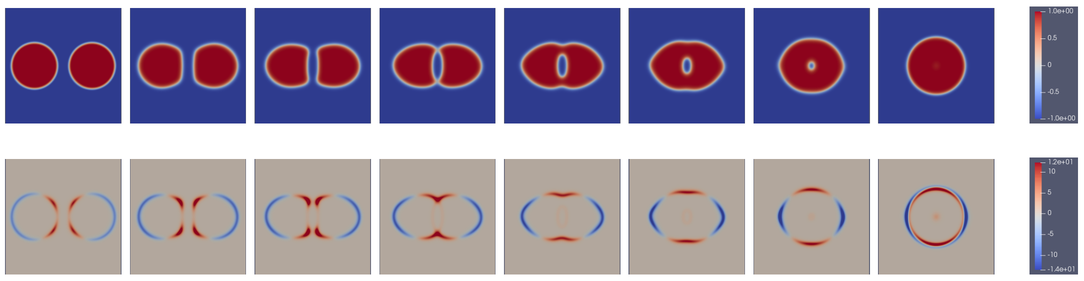
Figure 4.9: States (above) and corresponding controls (below) for the solution of splitting geometries.

	iso	l_1	hexa
$j(u)$	0.103955	0.0562286	0.0884921
$j_1 + j_2$	0.00409254 + 0.0998625	0.000728494 + 0.0555001	0.00115402 + 0.0873381

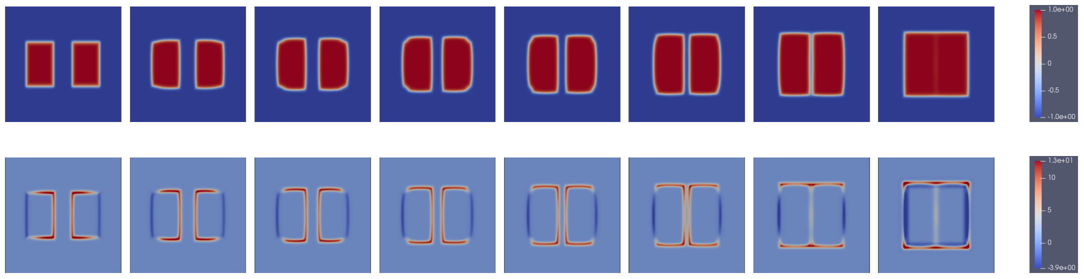
Table 4.5: Values of the cost functional for splitting geometries.

	iso	l_1	hexa
$j(u)$	0.0666414	0.0496374	0.0588482
$j_1 + j_2$	0.00274715 + 0.0638943	0.0010941 + 0.0485433	0.000905395 + 0.0579428

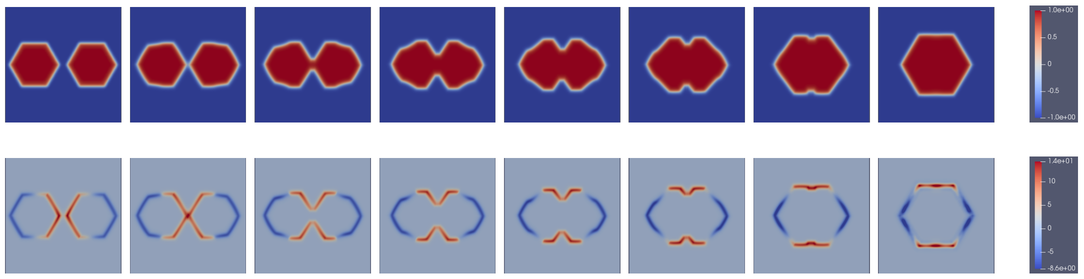
Table 4.6: Values of the cost functional for merging geometries.



(a) Result for ‘merge circle’ in the isotropic case.

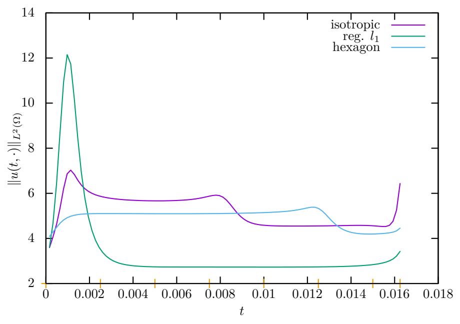


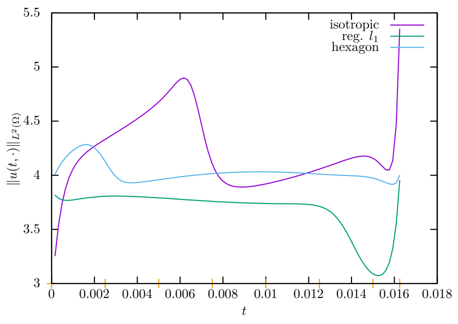
(b) Result for ‘merge square’ with the regularized l_1-norm.



(c) Result for ‘merge hexagon’ using the hexagon anisotropy.

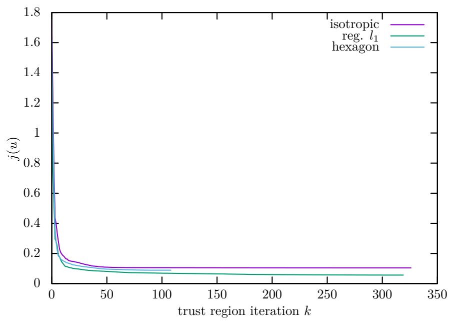
Figure 4.10: State (above) and corresponding control (below) for the solution of merge geometries.



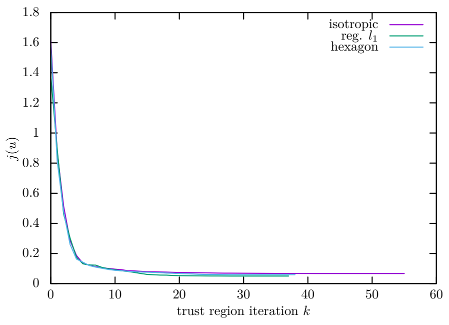


(a) Results for ‘splitting’, cf. Figures 4.9a to 4.9c. (b) Results for ‘merging’, cf. Figures 4.10a to 4.10c.

Figure 4.11: Time evolution of $\|u(t,\cdot)\|_{L^2(\Omega)}$ when topology changes are present.



(a) Results for 'splitting', cf. Figures 4.9a to 4.9c.



(b) Results for 'merging', cf. Figures 4.10a to 4.10c.

Figure 4.12: Evolution of $j(u)$ over the trust region iterations k when topology changes are present.

4.3.3 Filling the whole domain

As final example we want to consider a set of simulations where we seek filling the whole domain with one phase. Also here a topology change is present. As a starting configuration we choose a square with rounded corners that already covers a fair amount of the area. On the one hand this saves much computation time and on the other hand this allows a more detailed investigation of the final boundary occupying phase, in which we are mainly interested. As we have the same initial and final conditions for all anisotropies considered here, this setting also allows for a more direct comparison, exposing also subtle qualitative differences that we may have missed during the prior subsections. The associated results are depicted in Figure 4.13.

As in essence the control does not have much freedom, at first glance the outcomes look very similar for all three cases. The starting configuration soon begins to grow constantly until the whole domain is covered. Also the control costs plotted in Figure 4.14a resemble each other very much among the different anisotropies. However, as already mentioned, there are some differences in the details. Let us first have a look at the images of the controls in Figures 4.13a to 4.13c. As expected, also in this setting they have their support located in vicinity of the interface. By looking at the values that are assumed thereon, we find that for the isotropic and hexagon cases the cost expended at the corners is higher than at the rest, whereas for the square situation it is much more equally distributed. The reason for this behavior is that the form of the whole domain also is that of a square. Hence the most straightforward strategy, that was also pursued by the three simulations, is to fill the area by just growing a square—at least as long as no topology changes take place. As for the regularized l_1 simulation the actual Wulff shape is similar to this square, the force that has to be applied is comparable in all directions. This is in contrast to the isotropic and hexagon cases. For instance, in the isotropic case without control, a square would soon be deformed into a circle by first contracting the protruding corners. So this has to be counteracted by the control u, which explains why it attains its highest value at the vertices. Also for the hexagon anisotropy the control is strongest there. Note that in contrast to the isotropic case the attaching angle is somewhat skew.

The next difference concerns the final phase in which the boundary is reached. Also here the regularized l_1 case stands out, as being the only one where the contact takes place everywhere at the same time. In the isotropic case, the control prefers to initially build small excrescences that touch the boundary first. As it seems, these appear primarily next to the corners of the domain. Furthermore it is striking that this is not a symmetric solution. In repeated simulations one observes these excrescences in other places (but still around the corners). In Figure 4.15 such an alternative solution is presented. So it appears that there exists more than one local minimizer, all lying symmetrically around the starting guess $u^{(0)} \equiv 0$. The decision for one of those is made at the beginning based on some small but random rounding errors (recall that we are computing

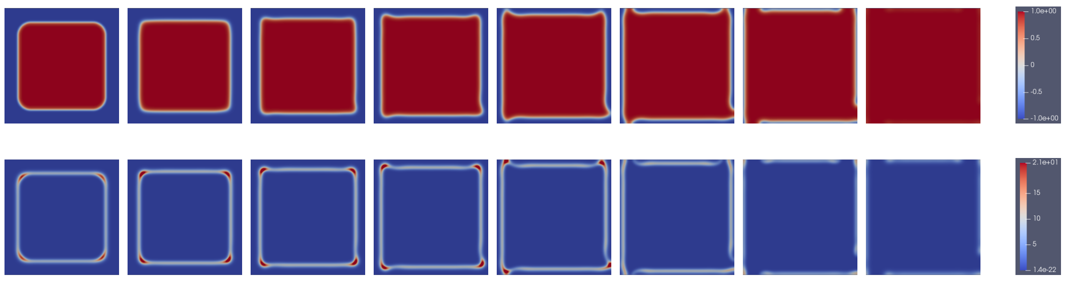
in parallel and that the floating point summation is not commutative). Finally, the hexagon anisotropy seems to provide a mixture of the two settings described before. The top and bottom parts approach the boundary as a whole like for the regularized l_1 anisotropy. In contrast to that, the left and right boundaries are again first touched by small excrescences. However there appear more than for the isotropic case and moreover they are arranged more regularly. The first and also biggest outgrowths are again built close to the corners. The reason for this twofold behavior can once more be explained by the Wulff shape. On the top and bottom it has a face that is more or less parallel to the boundary of the domain, just as is the case for the square anisotropy. In contrast to that, on the sides it tapers and is no longer parallel, which explains the behavior similar to the isotropic case.
The control costs of the regularized l_1 case are also qualitatively a bit different from the other two cases, see Figure 4.14a. In the middle section, the expense is more or less constant in the former case, whereas for the other two anisotropies it begins to vary at about $t = 0.009$. Comparing this to the related images it becomes apparent that this is exactly the moment when the excrescences are built. The control costs for the last segment are equivalent for all three settings. As soon as the boundary is reached, the expense rapidly decreases. Only in the end it goes up again by a small amount.
As already for the settings from the previous subsections, we provide the values for the cost functionals of the final solutions, which can be seen in Table 4.7. Again the main contribution results from the control cost term $\frac{\lambda}{2\varepsilon}\|u\|^2_{L^2(Q)}$. For the regularized l_1 norm it has the highest value. It therefore seems that for the ‘fill all’ simulation a strategy where not the whole boundary is reached at once is preferable, as was done for the isotropic and hexagon cases. Since the shape of the domain resembles a square, the optimal solution for the regularized l_1 case still approaches the boundary with the interface parallel to it, but this compares worse to the other two cases. Additionally in Figure 4.14b the values of $j(u)$ are given over the course of the trust region algorithm, but the behavior is the same as for the other settings.

	iso	l_1	hexa
$j(u)$	0.121633	0.144512	0.11084
$j_1 + j_2$	0.000737542 + 0.120896	0.000918727 + 0.143593	0.000668872 + 0.110171

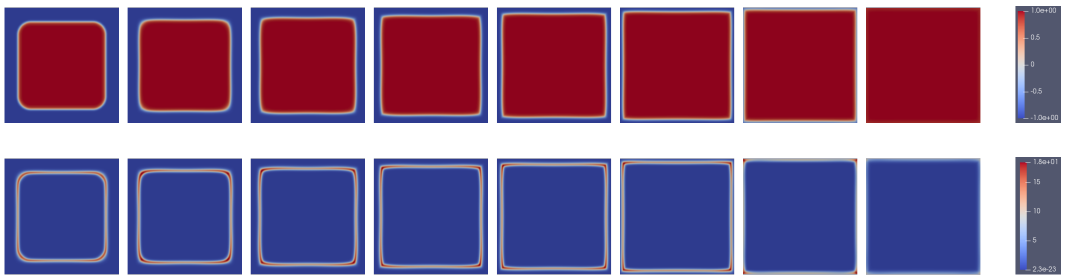
Table 4.7: Values of the cost functional for ‘fill all’.

4.4 Monitoring quantities

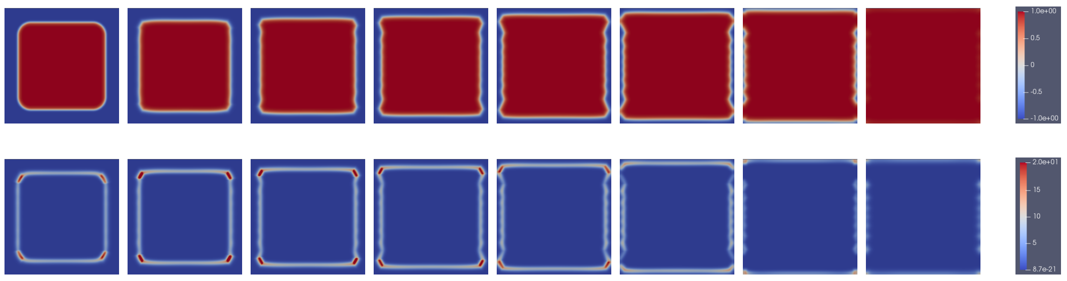
In this section we will investigate a bit more in detail how the algorithm behaves in course of the numerical simulations. Therefore, for each trust region step k we will look at several quantities providing information about this, including the residual $\|\nabla j(u_k)\|_{L^2(Q)}$, the amount of Steihaug-CG iterations, the current trust region radius σ and finally the the amount of seconds required for it. The corresponding results are presented in Figures 4.16 to 4.19. For each of the simulations discussed in the previous section, the corresponding plot of the monitoring quantities is included (see also the references given in the description of the figures). Before we start to treat the previously listed aspects in the following subsections, let us have a look at the general reading of the data presented. In each of the single graphs all four quantities that we mentioned above are given at once. Their respective color assignments are defined in the descriptions below. The residual is plotted in log-scale with the values given on the left ordinate. The values for the Steihaug iterations and the time spent in each trust region step can be read off the right ordinate. The latter is given in terms of seconds. Note that the scale for the trust region radius σ is not drawn in. The value at the first step is always initialized to 1.0 and can be taken as a reference. Further note that although σ can traverse several orders of magnitude, the most vivid representation was found to be achieved with a linear scale, like already for the latter two cases. For sake of comparability, for all presented plots we decided to use the same scale on the y-axis. That has as a consequence that in rare cases some values might be cut off. When



(a) Result for ‘fill all’ in the isotropic case.

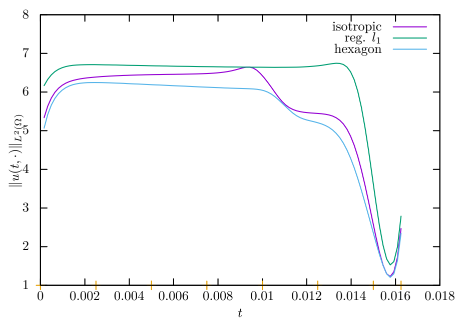


(b) Result for ‘fill all’ with the regularized l_1-norm.

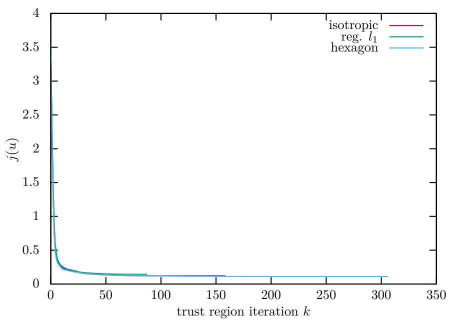


(c) Result for ‘fill all’ using the hexagon anisotropy.

Figure 4.13: State (above) and corresponding control (below) for the solution of ‘fill all’.



(a) Time evolution of $\|u(t,\cdot)\|_{L^2(\Omega)}$.



(b) Evolution of $j(u)$ over k.

Figure 4.14: $\|u(t,\cdot)\|_{L^2(\Omega)}$ and $j(u)$ for ‘fill all’, cf. Figures 4.13a to 4.13c.

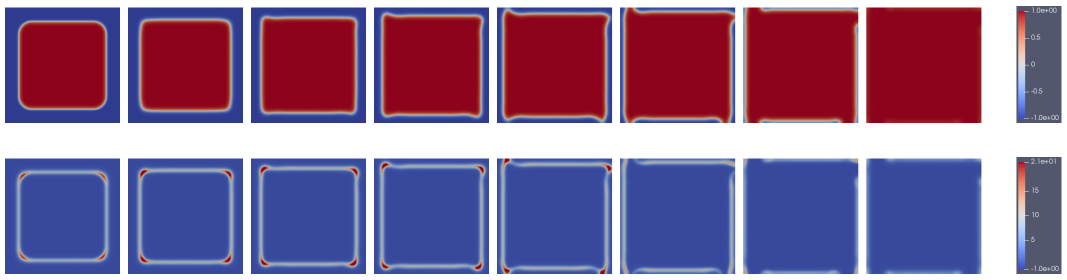


Figure 4.15: Another run of the simulation ‘fill all’ in the isotropic setting. Here, in contrast to Figure 4.13a, a different local minimizer has been determined, but the minimum is still $j(u) = 0.121633$.

they are important for the understanding, they are given in the corresponding discussion. In most cases, though, they also can easily be guessed. However note that the scale of the x-axis is different in all graphs. It is adapted to the respective total amount of trust region iterations.

4.4.1 Residual

Let us first discuss the residual, which is given by $\|\nabla j(u_k)\|_{L^2(Q)}$. Recall from Algorithm 2 that we terminate the algorithm if this value falls below some fixed threshold (we only use the absolute tolerance in the simulations). As given introductory and can also be read off the graphs, the tolerance is set to 10^{-13}. In all simulations, the initial residual is approximately of order 10^0 or 10^1. By looking at the history, one notices that it can essentially be split in two phases. In the majority of the simulations, most part of the running time is governed by the first phase which is characterized by the fact that there is barely a reduction of the residual. In many cases it rather oscillates around some value and only on a larger scale one can spot a small reduction. The length of this phase varies strongly between the different settings, as in most cases it roughly matches the total amount of trust region steps the algorithm took. The second phase corresponds to the superlinear convergence of the trust region Newton method close to the minimizer. This typically can be observed in the last 5 to 10 steps. Here the residual drastically decreases compared to the rest of the algorithm. Note that in some plots, e.g., those in Figure 4.18, the reduction in the very last step is smaller compared to the iterates before. However this is only an artifact of the algorithm. We stop as soon as the tolerance is reached, although in theory the residual could still be decreased at that point by the Steihaug method.

4.4.2 Steihaug-CG iterations and time per step

Next let us investigate the number of Steihaug-CG iterations and the time that was expended in each step of the trust region Newton algorithm. We will discuss these two quantities together as they share the same behavior. In fact, if one assumes that each Steihaug iteration approximately takes the same time and neglects further contributions, then as a rule of thumb one would expect that the time required for each iteration is given by a fixed multiple out of the corresponding Steihaug iteration. This can indeed be observed in the simulations. For instance, in Figure 4.17a, from step 50 onwards the amount of time is a bit below the Steihaug iteration count, implying that one iteration thereof needs somewhat less than a second. In the other cases the situation is similar for the most part. In Figure 4.17b, between steps 50 and 270, it seems that more time was needed. The reason is that here the trust region radius σ often had to be adjusted, forcing the Steihaug method to reiterate. For all simulations, at the beginning, the line corresponding to the time passed is clearly above the line for the Steihaug iterations, see, e.g., steps 0 until 50 in Figure 4.17a. Indeed, the latter is barely visible there. That means that other contributions, like the solution processes of the state and adjoint equations, dominate. This phase is however short for most of the simulations which took many trust region iterations, like, e.g., the two examples discussed above.

As we have just seen, when we include the Steihaug iterations, in principle we can now observe three phases in the most cases. In the following, we will refer to them as is described next. They can best be seen in Figure 4.17a and we give the corresponding trust region iterations for this example as a reference. The first phase is characterized by a small amount of Steihaug iterations and almost no error reduction (0–50). In the second phase, the latter is still the case, although the Steihaug iterations now have significantly increased (50–315). In third phase the superlinear error reduction takes place and often the amount of Steihaug iterations again have slightly increased in comparison to phase 2 (315–328).
The time passed in phases 2 and 3 is one or two orders of magnitude higher than that in phase 1. In simulations where phase 1 is short, we can therefore conclude that the main bottleneck of the algorithm lies in the solution of the Steihaug-CG method. For instance, let us check this for Figure 4.17a (splitting setting for isotropic case). Considering only the contributions from step 50 onwards and taking an average of about 150 seconds for each step, one obtains an estimated total running time of about $(330-50) \times 150s/3600\frac{s}{h} \approx 11.7h$ which coincides pretty well with the value of $11.98h$ given later in Table 4.8. This demonstrates that in cases where phases 2 and 3 dominate, the total running time indeed can be estimated by the Steihaug-CG method.
Now let us discuss in more detail the Steihaug iterations only. Recall from before that in phase 1 their amount is fairly small. The reason is that in this phase the Steihaug method does not iterate to the end, but rather breaks down due to the two other stopping criteria discussed in Section 3.1.2. That is, either a direction with negative curvature has been found, or the trust region boundary has been reached. As soon as the current iterate gets closer to the local minimizer, the amount of CG iterations increases. Ideally, at that point one is already very close to the optimizer, such that the Steihaug method only iterates to completion for a small amount of trust region steps, see, e.g., Figures 4.16a, 4.18b and 4.19c. This is not always the case, however, as can especially be observed when the algorithm resides for a long time in phase 2, like for instance in Figures 4.17a and 4.17b. Note also that in Figure 4.17a one can clearly observe that sometimes the time spent strongly surpasses the amount of CG steps that is very low (e.g. in vicinity of $k = 250$). The reason is that the algorithm first did a full pass of the Steihaug method, then realized that the final solution did not approximate the model problem in a satisfying way, shrank the radius and then eventually stopped with a much fewer amount of Steihaug iterations. In Figure 4.16a, at about trust region iteration 60, the number of CG iterations sharply increased for a short time, although the algorithm was far from converging. All in all, it became apparent that an increasing amount of Steihaug iterations does not necessarily indicate that the algorithm is close to the end (i.e. in phase 3), although in many of our simulated situations this was the case.
Finally, let us briefly comment on the maximum number of Steihaug steps that were attained. Recall that in the simulations, we do not allow for more than 800 iterations. For instance, this limit was indeed reached in Figure 4.16a. Also in Figure 4.19a, the amount of iterations is close to that several times. In most other simulations, the number of iterations in phase 3 lies at about 300. For slightly different situations, the maximum amount of 800 can be surpassed quite commonly. For example, if the end states of Figure 4.5a and Figure 4.6a would not contain the little square or circle in the middle, the situation would be more involved. Also the end time point T has a huge impact on this (cf. Section 3.2.2).

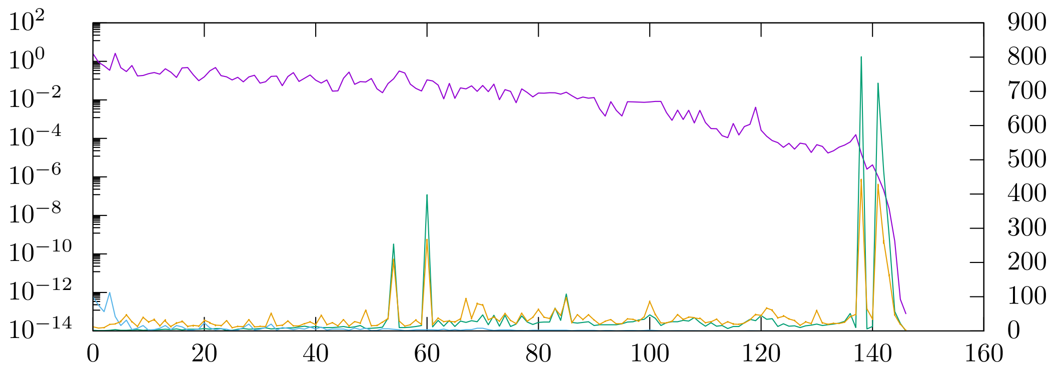
4.4.3 Trust region radius

The final quantity we want to look at is σ, the trust region radius. Recall from Section 3.1.1 that it determines how far we 'trust' the quadratic approximation (3.17) to be sufficiently well and therefore enters in the model problem (3.18). Roughly speaking, we would expect σ to be small in the beginning and increase towards the end of the algorithm. Indeed, this can be observed in the plots. As we do not know what 'small' means in a specific setting, we just initialize it with a value of 1.0. Typically this is too high and the radius soon shrinks to a smaller value. With the latter it stays for a long time and only in the last few iterations it increases strongly. For example, in Figure 4.17b, by the end a radius of roughly 43 is reached. At step 50, σ is only about 0.045. In Figures 4.17a and 4.17b one observes that the trust region radius oscillates in phase 2. Here the way of how σ is steered is probably not optimal and could be improved by tweaking the parameters κ_1,κ_2 and η in Algorithm 2. However, it is difficult to know that this is required in advance, as for the other situations the same values seemed to be robust choices. If the trust region radius became some orders of magnitude larger compared to its previous values, this was always a sign that the algorithm has been close to the end, i.e., that it entered phase 3. The converse is not true as can be seen in Figure 4.17a, where σ's final value has already been attained earlier. Sometimes σ is also observed to decrease again in the end. This can best be seen in Figure 4.18c. In Figure 4.16a the final radius is so small that it cannot be read off from the plot. In fact it is of the order 10^{-5} there. If the cost functional is approximated very well, the nominator and denominator in (3.19) can consist of fairly small numbers. Due to rounding errors, ρ is then no longer guaranteed to be a reliable estimator in Algorithm 2 and hence the radius can also decrease in this case. As the updates computed in the last few steps are also quite small, the size of the boundary however did not have an influence on the algorithm's superlinear error reduction.

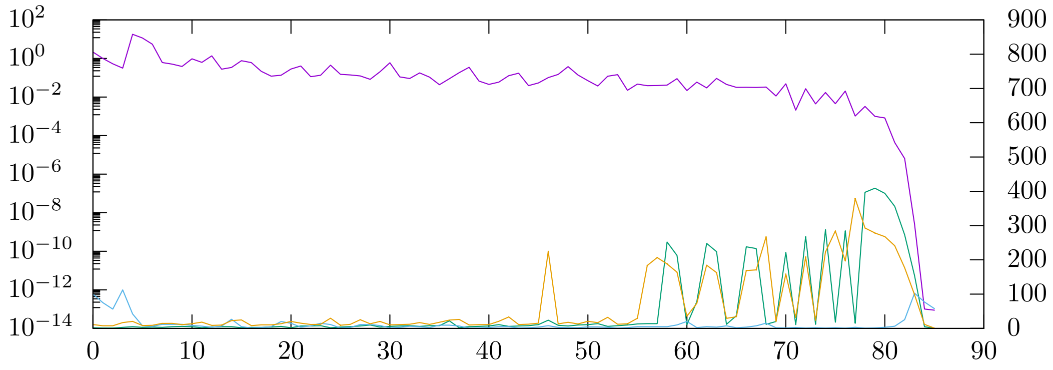
Conclusion 4.4.1. *In most cases, the algorithm can be roughly divided into three phases. They are characterized by the amount of Steihaug-CG steps as well as the error reduction. In phase 1 both of these characteristics are low. When the superlinear error reduction of Newton's method shows up (phase 3), the amount of Steihaug iterations typically increases by some orders of magnitude. This growth can also enter earlier (phase 2), but this phase is not always so pronounced. The total running time is dominated in the first phase by its length, in the second phase additionally by the amount of Steihaug iterations and in the third phase only by the Steihaug iteration count. When the trust region radius σ increases strongly, the algorithm is typically in phase 3. There are however situations where σ does not show this behavior.*

4.5 Efficiency discussions

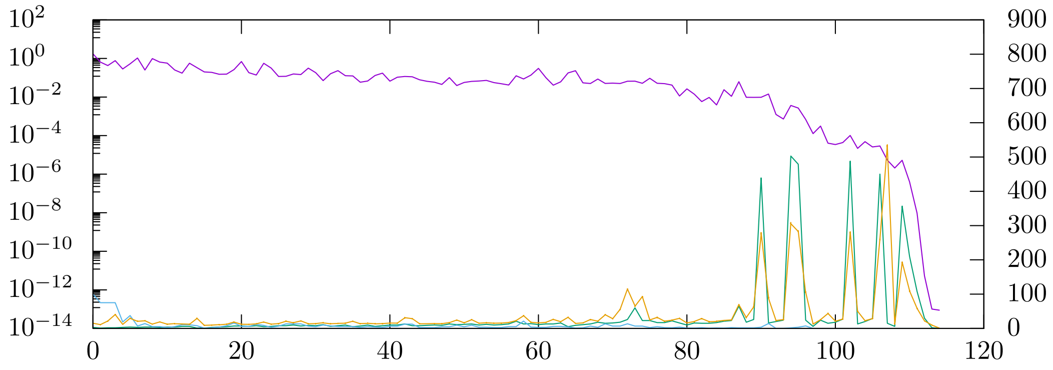
The last section of this thesis is devoted to the investigation of the algorithm's characteristics with respect to some implementational aspects that were raised throughout this thesis. This is in contrast to the last chapters, where all simulations were executed using the same program (differing only between the (an-)isotropies). Topics that we will consider in this section include the comparison to the semi-implicit scheme that we formally derived in Section 2.7.2, the effect of using `Mat_store_S` introduced by the end of Section 3.3.1, as well as the benefits of the parallelization discussed in Section 3.3.2.
As we will mostly concentrate on runtimes in the following, let us quickly compare the performance of the algorithm on the settings discussed so far. An overview is given in Table 4.8, where we compare the total iteration count and the total time the trust region solver took, as well as the median, maximum and minimum of the times that were spent at the single trust region steps. Note that instead of the mean we give the median as the former would not be meaningful in this context due to the two phases with rather different behavior, see also the last section. Concerning the iteration count we can state that the behavior varies a lot between the different configurations. For example in the 'star' settings, the isotropic case in both cases required less iterations than the anisotropies, whereas in the 'merge' and 'split' settings it took the most. The iteration count can also serve as an indicator for the total running time, but this has to be taken with a grain of salt, since the amount of Steihaug-CG iterations can also vary a lot (cf., e.g., Figure 4.17a). For the merging simulations, the regularized l_1 and hexagon cases seem to be very comparable however. This is also strengthened by a glimpse at Figures 4.18b and 4.18c. Note that the total runtime is a bit higher for the hexagon anisotropy, but this is expected because there is an additional matrix in (4.9) and hence more computational effort. In most cases other aspect are more important however. For instance, for the splitting setting the isotropic simulation took about twice as much time as with the regularized l_1-norm, although the total iteration counts are comparable. By looking at Figures 4.17a and 4.17b, we recognize that this is potentially due to the single Steihaug run-throughs taking more iterations and therefore more time. Note also that the median strongly differs from the others in the former



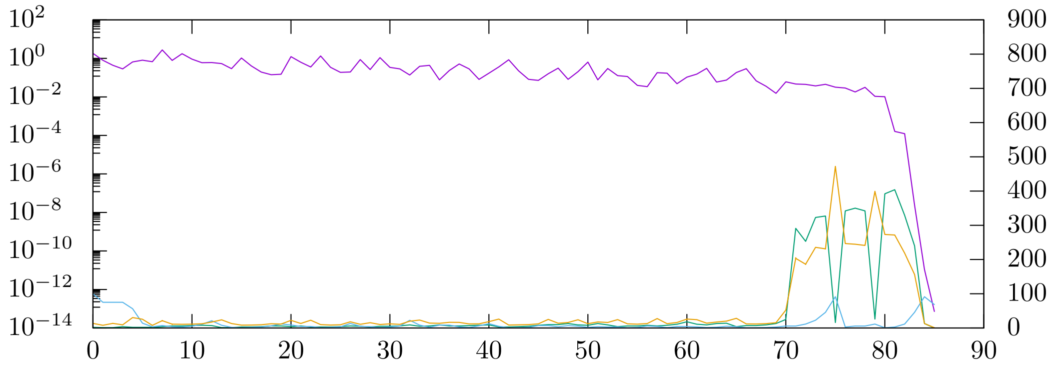
(a) 4-star: the regularized l_1 case, cf. Figure 4.5a.



(b) 4-star: the isotropic case, cf. Figure 4.5b.



(c) 6-star: the hexagon case, cf. Figure 4.6a.

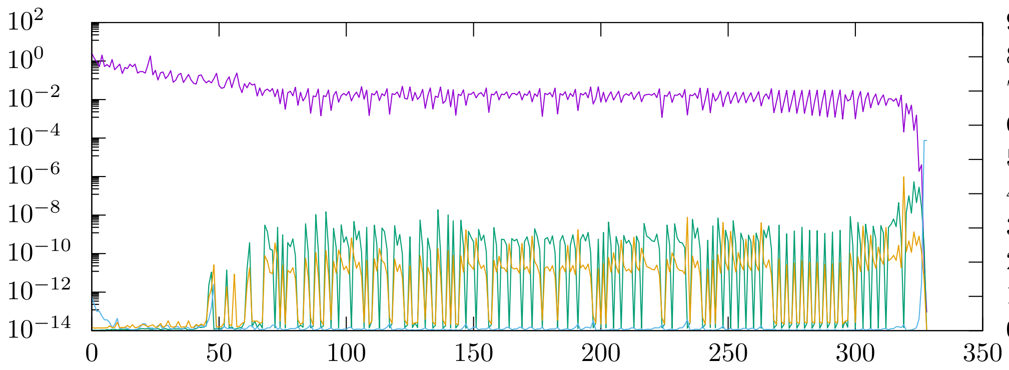


(d) 6-star: the isotropic case, cf. Figure 4.6b.

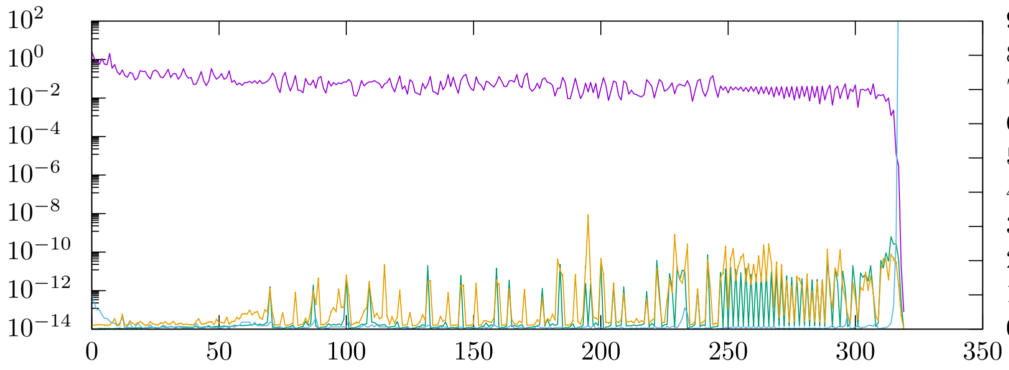
Figure 4.16: Corresponding to Figures 4.5 and 4.6, the following quantities are plotted against the *trust region iteration* k:

- residual $\|\nabla j(u_k)\|$ (left axis)
- St-CG iterations (right axis)
- TR radius σ (see below)
- seconds passed (right axis)

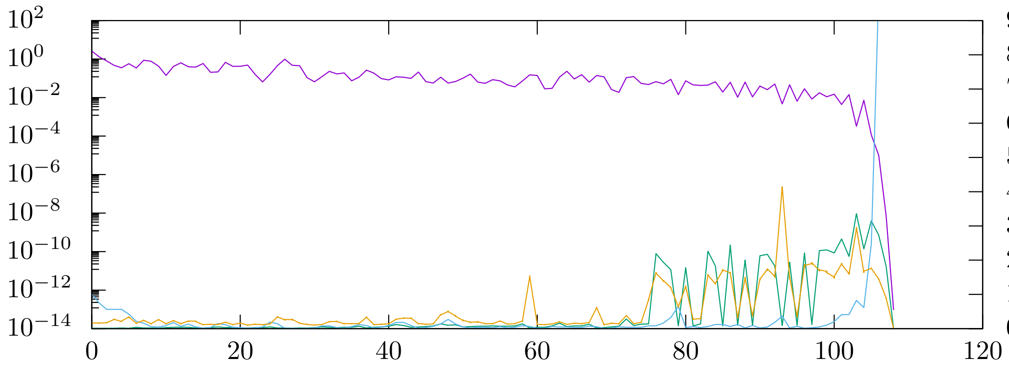
Note that the values belonging to the linear scale of σ are not given, but as a reference one can use that we always initialize it to 1.0.



(a) ‘splitting circle’, cf. Figure 4.9a.



(b) ‘splitting square’, cf. Figure 4.9b.

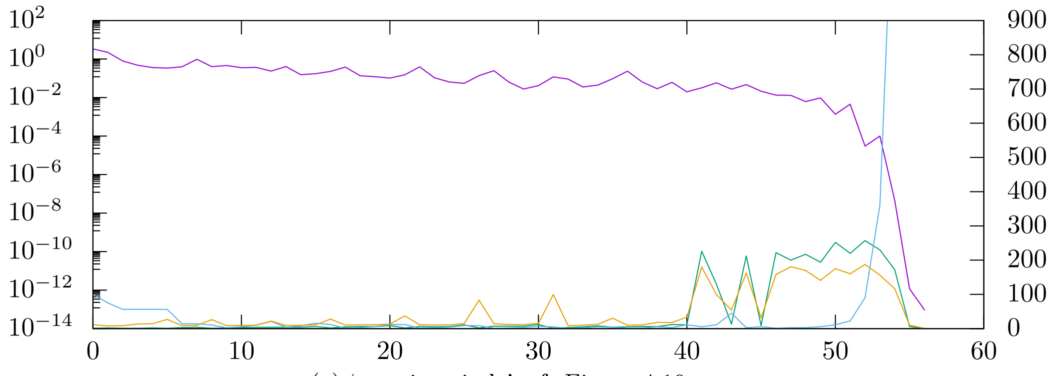


(c) ‘splitting hexagon’, cf. Figure 4.9c.

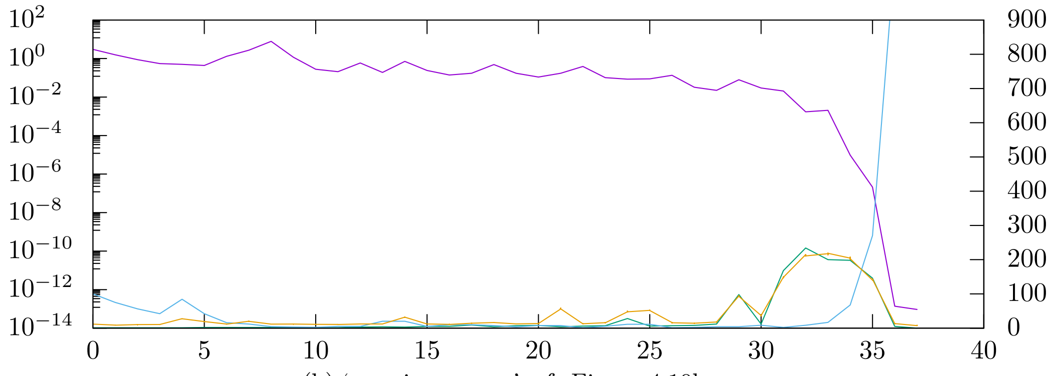
Figure 4.17: Corresponding to Figure 4.9, the following quantities are plotted against the *trust region iteration* k:

- residual $\|\nabla j(u_k)\|$ (left axis)
- TR radius σ (see below)
- St-CG iterations (right axis)
- seconds passed (right axis)

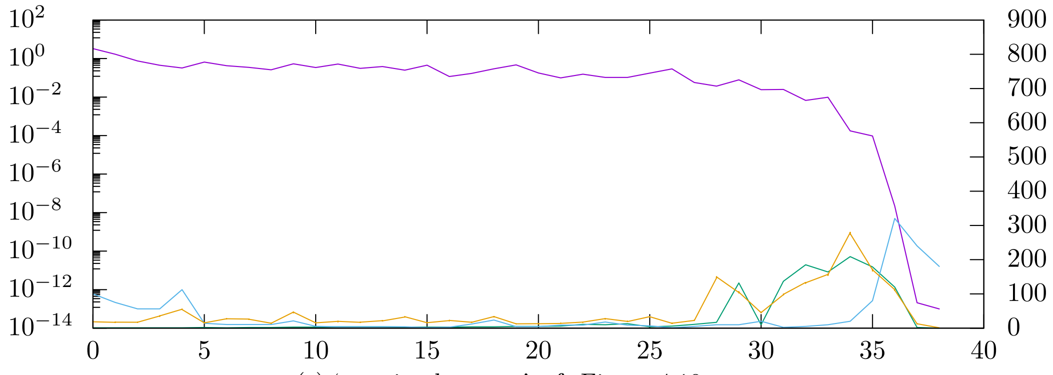
Note that the values belonging to the linear scale of σ are not given, but as a reference one can use that we always initialize it to 1.0.



(a) ‘merging circle’, cf. Figure 4.10a.



(b) ‘merging square’, cf. Figure 4.10b.

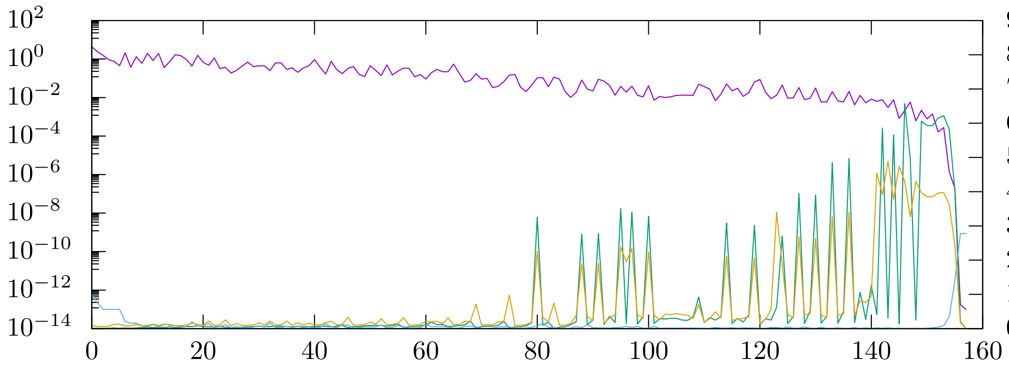


(c) ‘merging hexagon’, cf. Figure 4.10c.

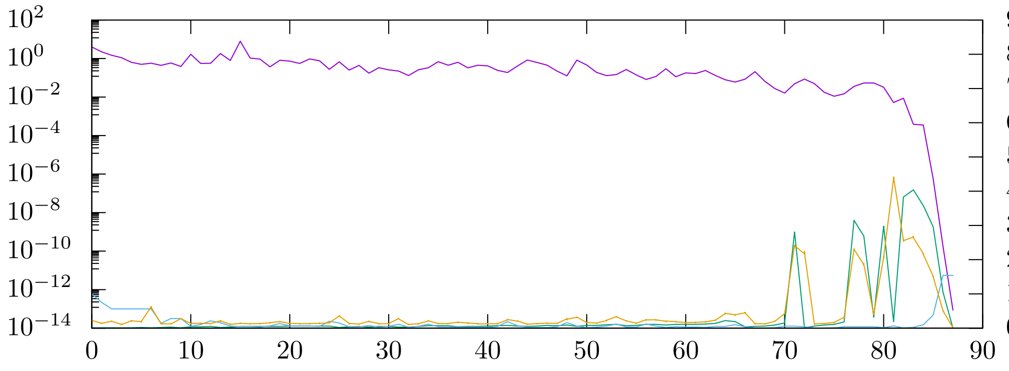
Figure 4.18: Corresponding to Figure 4.10, the following quantities are plotted against the *trust region iteration* k:

- residual $\|\nabla j(u_k)\|$ (left axis)
- St-CG iterations (right axis)
- TR radius σ (see below)
- seconds passed (right axis)

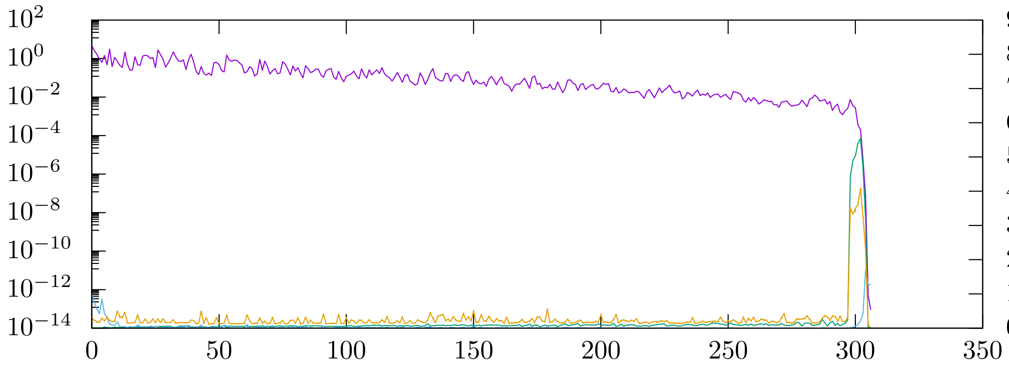
Note that the values belonging to the linear scale of σ are not given, but as a reference one can use that we always initialize it to 1.0.



(a) ‘fill all’ (isotropic case), cf. Figure 4.13a.



(b) ‘fill all’ (regularized l_1-norm), cf. Figure 4.13b.



(c) ‘fill all’ (‘hexagon’ anisotropy), cf. Figure 4.13c.

Figure 4.19: Corresponding to Figure 4.13, the following quantities are plotted against the *trust region iteration* k:

- residual $\|\nabla j(u_k)\|$ (left axis)
- TR radius σ (see below)
- St-CG iterations (right axis)
- seconds passed (right axis)

Note that the values belonging to the linear scale of σ are not given, but as a reference one can use that we always initialize it to 1.0.

	4-star		6-star		fill		
	l_1	iso	hexa	iso	iso	l_1	hexa
iterations	146	85	114	84	157	87	306
total time [h]	1.71	1.62	1.36	1.26	3.49	1.10	2.42
median [s]	28.55	19.37	20.63	14.89	21.97	16.95	18.85
max [s]	442.95	378.69	535.31	471.90	489.46	441.61	408.84
min [s]	1.95	0.97	1.05	0.92	0.94	0.97	1.28

	split			merge		
	iso	l_1	hexa	iso	l_1	hexa
iterations	328	319	108	56	37	38
total time [h]	11.98	5.85	1.76	0.76	0.45	0.53
median [s]	171.11	28.73	21.87	14.09	14.36	22.35
max [s]	449.18	333.98	414.34	187.58	219.15	279.74
min [s]	0.95	1.02	1.03	0.94	7.31	1.11

Table 4.8: Comparison of the total iteration count of the trust region Newton solver, its total time consumption, as well as the median, maximum and minimum times required by the Steihaug-CG solver.

	4-star*	split*	merge	fill
iterations	241	399	82	1228
total time [h]	3.37	2.43	0.38	4.07
median [s]	16.55	12.85	12.15	10.48
max [s]	718.70	167.63	123.91	209.99
min [s]	1.53	1.72	1.77	1.44

Table 4.9: The analogue to Table 4.8 with the semi-implicit discretization for comparison. Here, as anisotropy we only considered the regularized l_1-norm. The asterisk indicates that the setting was computed with $\delta = 0.1$. As a comparison let us mention that in the implicit case the 4-star setting took about 3 hours and 129 trust region iterations if run with $\delta = 10^{-1}$. The reason it takes more time compared to Table 4.8 is that towards the end there are more Steihaug run-throughs that run to completion.

case. Finally, observe that for the 'filling' simulations the iteration counts (and hence also the total time) is very different between the settings.

4.5.1 Semi-implicit scheme

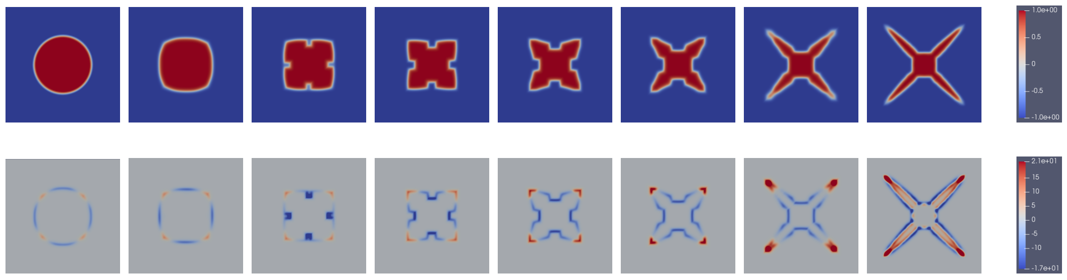
In this subsection we will briefly compare the results from Sections 4.3 and 4.4 with those that are achieved by using the semi-implicit discretization scheme discussed in Section 2.7. The semi-implicitly discretized versions of eqs. (1.24) to (1.27) were derived formally in Section 2.7.2. Recall from Section 2.7.3 that a rigorous treatment of the optimality conditions turned out to come up with serious difficulties in the semi-implicit case. Nevertheless, the trust region algorithm from Section 3.1.1 can still be applied using the formally derived equations. In order to compare this to the implicit discretization, we will exemplary look at the regularized l_1-norm (4.8) and recompute the four settings we considered before, now with the semi-implicit equations. The pictures of the resulting state and control can be seen in Figure 4.20. Hardly any difference can be spotted by eye. The values of the cost functionals are slightly different however, cf. Table 4.10.
Let us make some remarks about the simulation at that point. We first tried the standard choice $\delta = 10^{-7}$ from before. For this, only the settings 'merging square' and 'fill all' converged

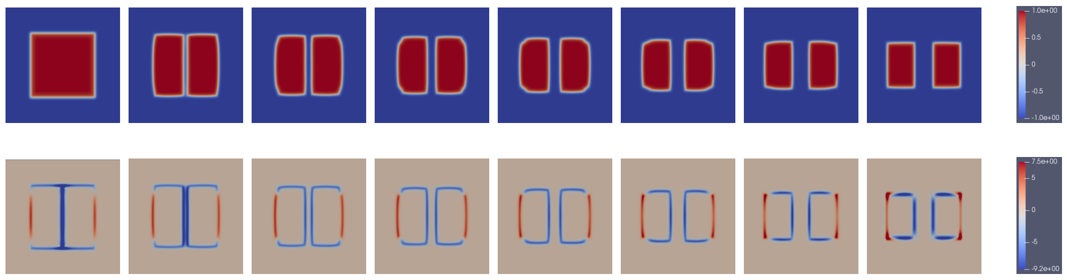
successfully. But they, too, were already showing signs of instability. The latter setting required 1228 trust region iterations, which is of orders of magnitude higher than the 87 taken by the implicit scheme. With 82 iterations, the merging simulation required less. The total number of iterations is observed to be rather sensitive with respect to rounding errors however. In successive simulations we got total iteration counts between 75 and 101. Note that these are the simplest settings that also for the implicit approach required the fewest iterations. For the simulations 'circle to 4-star' and 'splitting square' also after 3000 iterations the minimum has not been reached and there was no sign that this would be the case in the foreseeable future. The situation was no different for the choice $\delta = 10^{-4}$. We eventually tried the rather sophisticated choice $\delta = 10^{-1}$, where finally a solution could be found. This one will be considered in the following for the settings that did not work with $\delta = 10^{-7}$. We asterisked the name of the settings with $\delta = 10^{-1}$ in the description of the plots to remind of this fact. The difference in the determined minimum of the cost functionals can at most partially be explained by using another regularization parameter, as also for the merging and filling simulations it is different in Table 4.10. In fact, for the 4-star setting with $\delta = 10^{-4}$ and $\delta = 10^{-7}$, we respectively obtained the values $j = 0.123919$ and $j = 0.124101$ at the point we aborted the simulation, which compare pretty well to the values given in Table 4.10. We suspect the cause of these instabilities to be the matrix M_δ and its derivatives that appear in the semi-implicit equations, cf. eqs. (2.115) to (2.118). As explained in the pertinent chapter, the matrix tensor M''_δ behaves like $\sim q^{-2}$ at the origin for $\delta = 0$. In contrast to that, A'''_δ only behaves like $\sim q^{-1}$ for $\delta = 0$. This also explains the strong dependence of the stability on δ in the semi-implicit case. The explicit appearance of the double-well potential may render the evaluation of the state equation more unstable, but this typically gives a restriction on the time step size which was chosen small enough to provide sufficient stability. This fact alone was not observed to have an effect on the behavior of the optimization algorithm.

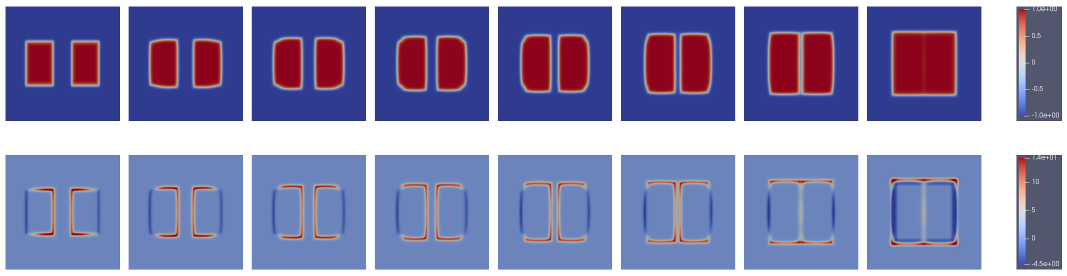
Now let us further discuss the results we obtained. In Figure 4.21 the corresponding plots of $\|u(t,\cdot)\|_{L^2(\Omega)}$ against t are presented. Qualitatively, we can only make out a difference from the implicit case for Figure 4.21a, where now two hills appear at the beginning of the deformation and when the excrescences grow the most. In general, in all situations the costs are slightly higher as in the implicit case as we also can observe for $j(u)$ in Table 4.10. The meanwhile familiar graphs of the monitoring quantities are plotted in Figure 4.22. Note that with the more moderate regularization, the algorithm needs less trust region iterations than for the setting 'fill all'. In the implicit case, where δ was left untouched (i.e. $\delta = 10^{-7}$), this was the other way round. It is noticeable that in Figure 4.22a, the maximum amount of 800 Steihaug steps was reached quite often. Overall, the behavior of the algorithm is comparable to the implicit case. Finally, in Table 4.9 we give the analogue of Table 4.8, now for the semi-implicit simulations. Again we observe that the total amount of iterations is not necessarily proportional to the total time, since also the behavior of the Steihaug method enters here. The remaining observables show minor differences to the implicit case. Only the maximum time in the first column stands out a bit. Furthermore, the median for the last two columns is a little bit better compared to the implicit case, but this is probably due to the fact that the algorithm is stuck for a long time with very few Steihaug iterations. So in total we can state that the semi-implicit discretization runs comparable to the implicit one, but needs significantly more regularization. In the following subsection, the benefit of a certain implementation aspect is addressed, also with respect to the semi-implicit approach.

4.5.2 Keeping assembled terms in memory

In this subsection we want to numerically investigate an implementational aspect that we initially discussed in the part 'Linearized and additional adjoint equation' of Section 3.3.1 on page 90. Recall from there that for the linearized equations being solved in turn during the Steihaug-CG method, for a fixed trust region step most terms don't change as they include the same state and adjoint variable. This fact can be harnessed to speed up the simulation by storing the reusable terms, as was demonstrated there. We now want to corroborate the assertion that this in fact leads to an acceleration of the simulation. For respectively storing the left-hand sides of both


(a) ‘circle to 4-star’*


(b) ‘splitting square’*


(c) ‘merging square’

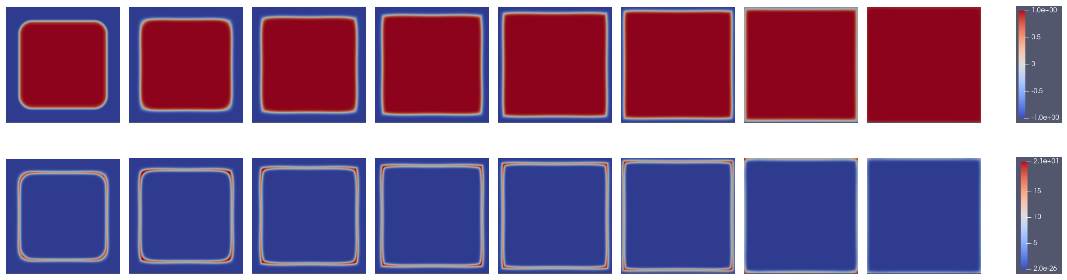
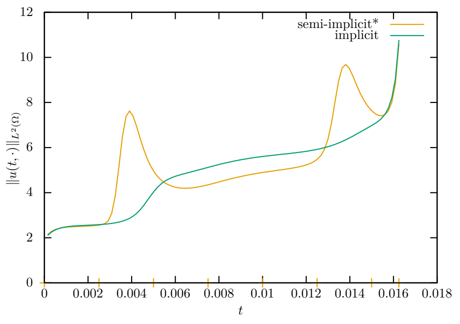

(d) ‘fill all’

Figure 4.20: State (above) and corresponding control (below) for the considered settings for the semi-implicit discretization. The asterisk indicates that the setting was computed with $\delta = 0.1$.

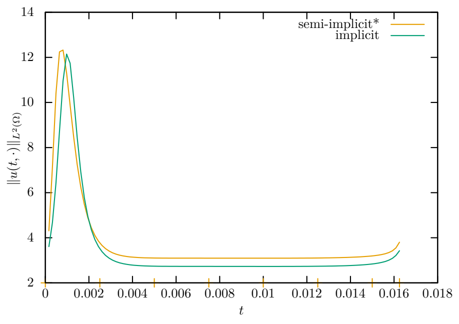
semi	4-star*	split*	merge	fill
$j(u)$	0.123892	0.0660538	0.0600625	0.178357
$j_1 + j_2$	0.01190 + 0.11200	0.00097 + 0.06509	0.00128 + 0.05878	0.00056 + 0.17780

implicit	4-star	split	merge	fill
$j(u)$	0.107378	0.0562286	0.0496374	0.144512
$j_1 + j_2$	0.01089 + 0.09649	0.00073 + 0.05550	0.00109 + 0.04854	0.00092 + 0.14359

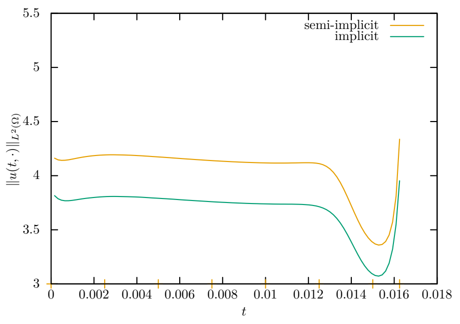
Table 4.10: Values of the cost functional for the semi-implicit scheme (above). The values for the implicit scheme (below) are given for comparison and were collected from Tables 4.4, 4.5, 4.6 and 4.7.



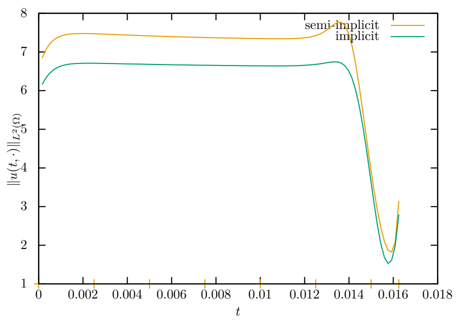
(a) Results for ‘circle to 4-star’*, cf. Figure 4.20a.



(b) Results for ‘splitting square’*, cf. Figure 4.20b.

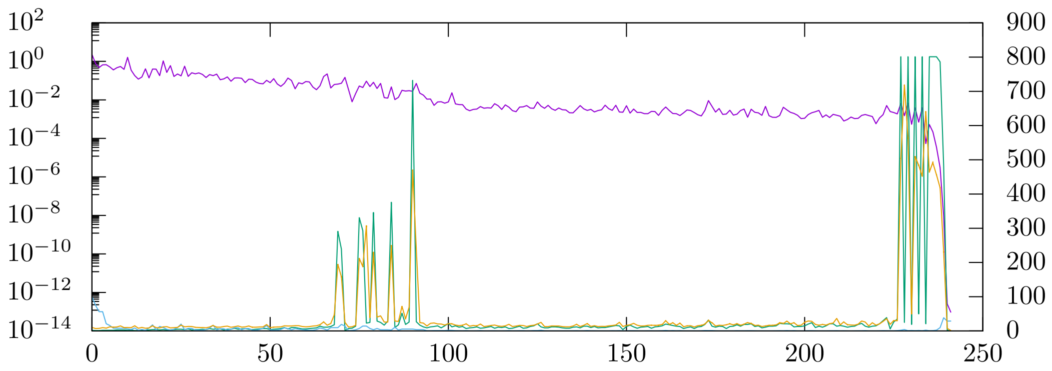


(c) Results for ‘merging square’, cf. Figure 4.20c.

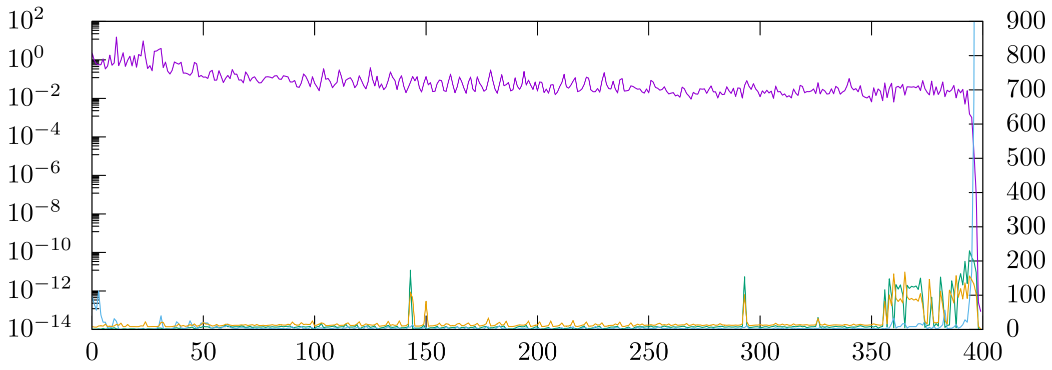


(d) Results for ‘fill all’, cf. Figure 4.20d.

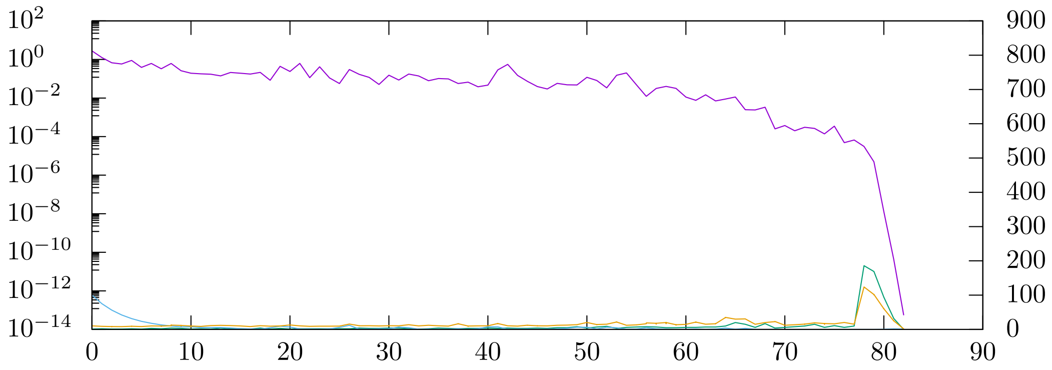
Figure 4.21: Evolution of $\|u(t,\cdot)\|_{L^2(\Omega)}$ over the time t for the semi-implicit settings from Figure 4.20. Their implicit counterparts are plotted for comparison and were taken from Figures 4.7a, 4.11a, 4.11b and 4.14a.



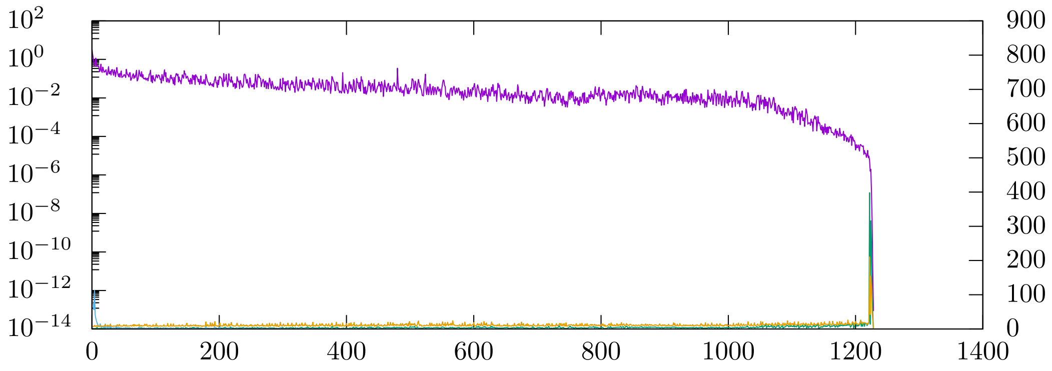
(a) 'circle to 4-star'*, cf. Figure 4.20a



(b) 'splitting square'*, cf. Figure 4.20b.



(c) 'merging square', cf. Figure 4.20c.



(d) 'fill all', cf. Figure 4.20d.

Figure 4.22: For the semi-implicit results corresponding to Figure 4.20, the following quantities are plotted against the *trust region iteration* k:

- residual $\|\nabla j(u_k)\|$ (left axis)
- TR radius σ (see below)
- St-CG iterations (right axis)
- seconds passed (right axis)

Note that the values belonging to the linear scale of σ are not given, but as a reference one can use that we always initialize it to 1.0.

equations, one expects a benefit offhand, as they are *exactly* the same in each iteration. For the right-hand sides this is not that clear as they also contain terms that have to be assembled in any case, as they rely on variables that are updated in each iteration, like δu_j, δy_j and the solution from the previous time step.
Let us first concentrate on the implicit case that is mainly considered in this thesis. For the linearized state equation (4.5) the right-hand side only contains terms with changing quantities, so it has to be reassembled anyway. The additional adjoint equation (4.3) contains terms involving A''' and ψ''', which apart from the factor δy_j remain the same in each iteration. Hence there could be a benefit from storing the matrix that results from these terms if one leaves away δy_j, then apply that matrix to the latter and add it to the first term. However it is not clear a priory if this is worth the effort, as the assembly typically consists of only one for-loop over the elements which we now exchange for several for-loops for the matrix applications and the subsequent vector addition. We therefore consider two cases, one where we only store the left-hand side (S) and another where in addition also the right-hand side is saved as just described (S & L).
For the semi-implicit discretization from Section 2.7 the situation is a little bit different. Here also for the linearized state equation (2.117) the right-hand side contains non-trivial terms. So it is reasonable to apply the just described procedure also for this equation, and hence for the semi-implicit discretization, the case (S & L) additionally contains this improvement. Furthermore another issue can appear in the semi-implicit case that is best understood by looking at the additional adjoint equations (4.3) and (2.118). Note that for the semi-implicit discretization this equation consists of a larger amount of individual terms. However, the number of floating point operations that are needed to set it up are the same as in the implicit case, as we will reason in the following. For this, recall the relation (2.104) between A' and the matrix M. Given this, for the implicit case we are actually free to express the derivatives A', A'' and A''' also in terms of M and its derivatives. Due to the product rule that has to be applied when using the latter, there appear additional terms that each can be related to one of those in the semi-implicit equation. Since in the latter case quantities at different time steps are incorporated, one does not have the freedom to express everything in terms of A' and its derivatives there, yielding the cumbersome looking additional terms. Note also that, compared to the implicit scheme, some parts of the equation appear on another side, leading to a slightly different balancing of the costs. There are however other sources of additional expense for the semi-implicit case. First of all, additional variables, namely y_{j-1}, y_{j+1}, p_{j+1}, δy_{j-1} and δy_{j+1}, have to be loaded out of the memory now. Further note that not only δy_j appears on the right-hand side, but also δy_{j-1} and δy_{j+1}. Hence one now has to separately store four terms (belonging to δy_{j-1}, δy_j, δy_{j+1} and δp_{j+1}) that eventually have to be added in each iteration of the Steihaug-CG method. In summary, for the semi-implicit discretization it is even less clear than for the implicit one if storing the right-hand side comes up with a benefit.

The answer to all these questions is given in Table 4.11. Here we present the results that were achieved with the just described optimizations for both, the implicit and semi-implicit discretizations. For the first column no optimization was done, for the second column we only store the left-hand side and for the third column we in addition save parts of the right-hand sides as described above. To compute the values shown, we took exemplary the first two Steihaug-CG steps of the first trust region step of the setting where we split a square in the regularized l_1-norm (Figure 4.9b). In other situations the times could be different, but the observed relative behavior should be the same. By looking at the first two steps, we are able to catch the behavior we are interested in. All values are given in seconds and in terms of the wall clock time that was spent to solve the given equation. The first value is the value of the *second* Steihaug step and represents the expected behavior of all the remaining steps of the Steihaug method. Here the benefits of the optimization will show up. The value in brackets belongs to the *first* Steihaug step and is given for comparison. This step will take longer since here the assembly and subsequent storage is performed. As expected, for the first columns both times are the same, apart from some natural fluctuations. At that point let us briefly mention the times that were taken for the state and adjoint equations. For comparison, in the implicit case the adjoint equation took about 2.5 seconds and in the semi-implicit case about 4.2 seconds, which is in good agreement

	implicit		
	standard	store S	store S & L
linearized equation	2.59 (2.56)	0.49 (3.14)	0.48 (3.07)
add. adjoint equation	4.96 (4.88)	2.80 (5.45)	0.49 (6.06)

	semi-implicit		
	standard	store S	store S & L
linearized equation	4.26 (4.25)	2.62 (5.09)	0.45 (5.57)
add. adjoint equation	5.79 (5.86)	4.14 (6.41)	0.50 (14.03)

Table 4.11: Comparison of the wall clock times between different ways of solving the linearized and additional adjoint equation for the implicit (above) and semi-implicit (below) discretizations. In the column titled 'standard' no storage was done. With S we refer to the bilinear form (left-hand side) and with L to the linear form (right-hand side). The first value given is the time required for one iteration of the Steihaug-CG method. In brackets we give the time of the first iteration, where the storage is performed. All times are given in seconds.

with the times presented for the linearized equations. This is reasonable as for both in essence the same things have to be computed. Note also that here the semi-implicit approach takes more time, which probably lies in the fact that for the right-hand side the additional variable y_{j-1} has to be loaded. For the state equation the situation is different. In the semi-implicit case it also took about 4.2 seconds, whereas for the implicit case it required about 15 seconds. The latter however is not surprising, as a nonlinear equation has to be solved now. For the semi-implicit approach the equation to solve is still of linear nature. The second column, where we saved the left-hand side for later reuse, is more interesting. Note that in all cases the first iteration (the time in brackets) now took significantly more time than the other iterations. Compared to the first column, the first iteration has taken longer but the other steps are computed quicker now. The relative speed up is better for the implicit case, since as discussed above, the right-hand side is quite trivial here. In comparison to this, in the semi-implicit case some terms moved from the left-hand side to the right-hand side. As it seems, there is actually a good benefit from storing in addition the right-hand side, as can be seen in the last column. Note that for the linearized equation in the implicit case, apart from fluctuations, there is no difference to the previous case. This is because we could not make any further improvement on the implementation. Especially for the semi-implicit case we obtain a great speed up. The times now compare to the implicit case—only the first step takes longer than before. In particular this can be observed for the additional adjoint equation, that now lasts about 14 seconds. The reason is that we have to assemble several parts of the right-hand side separately, each for a different variable and time step. Formerly, the right-hand side was assembled at once. But also here the storing approach is preferable, as the overhead is compensated after three Steihaug iterations ($(14.03 + 0.50 + 0.50)/5.79 < 3$). This amount is typically attained also very early in the trust region algorithm when the Steihaug-CG method stops earlier. So choosing this approach should give a benefit in any case. In summary we have demonstrated that the most benefit can in fact be achieved by storing both parts of the equations.

4.5.3 Parallelization

Let us finally comment on the effects of the utilized parallelization that was discussed in Section 3.3.2. As explained there, we use the built-in parallelization of the FEniCS solvers to speed up the solution process of the equations. With this also the whole algorithm will terminate earlier. However, the general behavior of the trust region method in terms of the iteration count is not affected by this, although due to the non-commutativity of the floating point addition there can appear minor differences on the long run. To explore the effect of the parallelization,

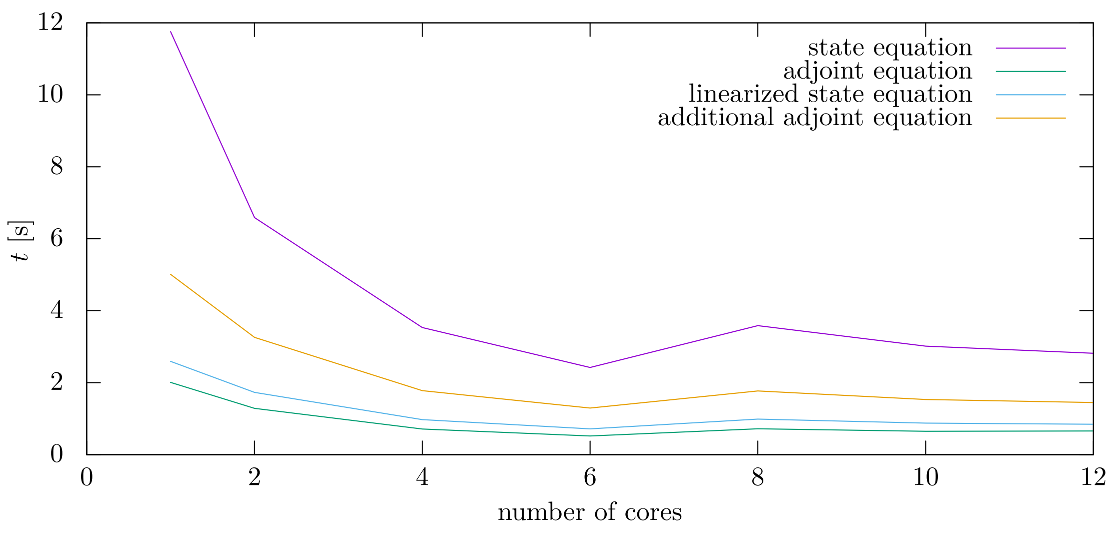


Figure 4.23: Plot of the average computation times for one solve of each of the equations.

we will look at the equations being solved. As an example we again measured the times of the first trust region step of the simulation belonging to the splitting of a square in the regularized l_1-norm. The results are plotted in Figure 4.23. Considering another trust region step or a different setting could yield different times, but the relation among them should be the same. To be more concrete, the presented figure demonstrates the wall clock time in seconds for the four appearing equations. Here we measured the time that is needed to solve the full parabolic equation, that itself consists of a sequence of several elliptic equations. The number of processes is chosen between one and twelve, where the latter is the maximum amount of cores our machine provides. As can be observed, the state equation takes the most time to solve. The reason for this is that in each time step we have to solve a nonlinear equation. The adjoint and linearized equation both require approximately the same amount of time. This is expected as in essence the same equation with different right-hand sides has to be solved. Also for the additional adjoint equation, only linear elliptic equations have to be solved. However the operator differs from that of the equations just discussed. Further the assembly of the right-hand side, which also contributes to the times, takes longer. This explains why the line for this equation is a little bit above the other two. However, all four graphs in principle show the same behavior. The runtimes decrease until an amount of six cores is reached. The times between one and two cores as well as two and four cores approximately are halved for the state equation. After the minimum at six cores the times stagnate and are even slightly worse. That we reach the optimum so early is potentially a consequence of the relatively small space discretization we use. With $129 \times 129 = 16641$ grid points and six cores, each process has to handle about 2774 degrees of freedom in each time step. For a computer this is not much and probably the costs of communication overweight soon. With a finer space discretization one will be able to take more profit of the parallelization. As for optimal control one has to solve a high amount of equations, this is not always suitable. Also note that these equations have to be solved in sequence, so the use of parallelization is only limited for our purposes.

Conclusion 4.5.1. *If the implementation is suitably optimized, the time spent to solve the linearized state equation and the additional adjoint equation is comparable for the implicit and semi-implicit approaches. In the former case, the state equation takes longer to solve because of its nonlinear nature. This fact is negligible in our simulations, as typically the execution is dominated by running the successive Steihaug iterations. We also found that, unlike the implicit approach, the semi-implicit one frequently does not converge if the regularization parameter is* *too small. The use of parallelization is limited by the coarseness of the discretization as well as the overall sequential nature of the algorithm.*

5 Conclusion and outlook

In this thesis we have applied the framework of optimal control to an anisotropic Allen-Cahn equation, comprising analytical results as well as numerical considerations. While there already exists plenty literature about the analysis of phase field models, quasilinear optimal control problems and numerical methods for optimization problems, to the author's knowledge a combination of all these different facets is still pending. A first step was done in this thesis by providing as a new contribution the associated analysis and numerical results as well as pointing out the issues that arise. In this concluding chapter, we summarize the work and review the enhancements that were made to the existing literature. Later we will point out some open problems that could not be resolved within the scope of this thesis and could be the subject of future investigations.

The general form of the anisotropic Allen-Cahn equation that has been used in this thesis is similar to the one given in [53]. Our result concerning the existence of a weak solution was formulated based on this, but the approach differed by the assumptions we put on the anisotropy and the double-well potential. While for the latter the intention was just to obtain a slightly more general result, the altered assumptions on the anisotropy were actually necessary in view of the regularization we had to invoke later in order to formulate the first order conditions. A further difference was that we delayed taking the limit in the time discretization to the end in order to simultaneously obtain results also for the implicit in time formulation and its convergence pursued in this thesis. Hereby the inequalities (2.24) and (2.37) were of big interest as they also appeared later in the proof of the existence of an optimal control for the optimization problem. They further allowed the existence results to be independent of the space dimension. We also could show that a subsequence of global minimizers of the regularized and time discrete problems converge for the regularization parameter $\delta \to 0$ and/or time step $\tau \to 0$ where the limit is again a global minimizer of the pertinent problem. Furthermore we specified the appearing anisotropy to be of a certain form that was first introduced in [16, 14]. Therefor an efficient time discretization was applied by the same authors in [13, 15], but in Section 2.7 we saw that this was unsuitable for optimal control as continuity and differentiability results turned out to provide some serious issues. The special choice of anisotropies allowed to specify some regularization prescription that rendered the quasilinear term to be sufficiently smooth and moreover suitable for the numerics.

We also derived first order necessary conditions for our control problem. Here we restricted ourselves to the time discrete version. We could adapt results from [33, 35] and applied this to the single time steps with a generalization concerning the space dimension, although being a little bit more restrictive now depending on the actual choice of the potential (see Assumptions 2.4.1). The linearized equations needed for the later used Newton type solver were

derived formally. Due to the term $(A_\delta'''(\nabla y_j)[\nabla\varphi, \nabla\delta y_j], \nabla p_j)$ in (4.3), the associated rigorous treatment is expected to require more than L^2-regularity for the gradient terms. This would also force all the previous considerations to be altered leading to a regularity-'cascade' similar to the semi-implicit discretization as in Section 2.7.3. This remains as an open problem.
The next part of the thesis covered the implementation of an optimization solver and eventually the provision of numerical results. As solver we ultimately decided to use a trust region Newton method with a Steihaug-CG algorithm to solve the quadratic subproblems. We weighted up several arguments to decide between solving the full or reduced system and different algorithms therefor. We will briefly repeat the main points here. The full problem does not require the single PDEs to be satisfied, but on the other hand has to handle thrice as much degrees of freedom at once. Also, for the full system there exist more preconditioning approaches in the literature, however these are primarily applied to provide mesh independence which is already included in the reduced approach (see Section 3.1.3). Also a test implementation favored the reduced approach in this regard. Finally, the existing globalization approaches are more involved for the full system, whereby for the reduced system, which represents an unconstrained problem, the globalization using a trust region method can be applied quite naturally. All these arguments led our choice to consider the *reduced* problem in this chapter. With a look at the numerics, the reduced system still seemed to be quite ill-conditioned. In Section 3.2 we pointed out the possible reasons for this as well as problems that conventional preconditioning techniques provide in the present context. We concentrated on the isotropic case as the driving problems were already believed to appear here. To the author's knowledge there exists no literature about the preconditioning of the reduced Hessian of a nonlinear parabolic control problem. For the present case, the analysis is mainly hindered by a term $-\frac{6}{\varepsilon}py$ appearing in the reduced operator that also possibly causes negative and/or small eigenvalues. In the anisotropic case a similar term would also appear for the gradients. Also the end time point T has an influence on the condition number, whereas the discretization—both with respect to space and time—does not. Finally we saw that an approach where the operator is factorized cannot be transferred straightforwardly to this case as it is unclear whether there exist efficient (approximate) solvers for the resulting terms if they are invertible at all.
In the end, we presented various numerical results. First we supported the previous theoretical considerations about mesh independence and convergence with respect to the regularization parameter. Then we demonstrated the applicability of the method to several settings of interest. These included crystal formation and topology changes. Also the behavior of some of the algorithm's monitoring quantities was discussed. Here we observed that towards the end of the algorithm, the number of Steihaug steps grew considerably, and with that also the execution time per trust region step. Finally, we compared the efficiency between the implicit and semi-implicit schemes and some further implementational aspects. We observed that if the implementation is suitably optimized, both discretization schemes performed comparably fast. As the implicit approach is less sensitive with respect to the regularization and as it is supported by more theoretical results, we however preferred it over the semi-implicit one.

We want to conclude this thesis with a discussion of several possible future research topics built on some open problems that could not be resolved within the scope of this work.
First, recall that our approach to deal with the nonsmoothness was to regularize the problematic term. Another way would be to leave the term as it is, adopt the analysis accordingly and use nonsmooth solvers to handle the problem numerically. The advantages would be that the quasilinear term stays homogeneous, expiring the need for some unhandy assumptions, and the fact that one is closer to the original problem, not leading to approximation errors due to the regularization (that were quite small in our case though). However optimal control with such kinds of nonsmoothness is still in its infancy and probably some concepts would have to be tested on simpler problems first. The closest work known to the author is [36], where only an elliptic problem is considered. The quasilinearity also is of another form however, but includes a singularity at the origin. Furthermore, the term that would correspond to the double-well potential is required to be monotone there. The resulting first order conditions are of a form that omits areas with vanishing state in the formulation of the adjoint equation. Most other literature

that exists, as, e.g., [41, 78, 65], treats nonsmoothness in the potential term, as for instance the obstacle potential (2.5). Semismooth Newton methods seem to be good candidates as they can be applied when the singularity only appears at countably many points [39]. They have to be extended by a globalization strategy in order to be suitable for our purposes. The cited reference merely treats elliptic equations with the nonsmoothness appearing in the potential term. Furthermore, for our problem, A''' is not only undefined at the origin, but also diverges by approaching it (unless A is the square of an energy-norm). Note that this problem was implicitly remedied by our regularization. So it could be the case that one needs regularization anyway at least for a numerical treatment. A further problem that is present for the Allen-Cahn equation is that $\nabla y \approx 0$ due to the phase separation correspondence. Analytical problems are only expected to hold at places where $\nabla y = 0$, but as hinted above, potential numerical solvers can break down earlier. Also, for other choices of the potential (e.g. for the double obstacle potential), the gradient of the state will in fact vanish on some set with positive measure.
The next point concerns the analysis of eqs. (1.23) to (1.28) at different discretization levels. In course of the thesis, we treated the state equation (1.23) both—in the time continuous as well as in the time discrete setting. The adjoint equation (1.24) was derived for the time discretized control problem. In a future project one might also do the same analysis on the time continuous level. Recall from above that [38] is the closest reference to this topic known to the author, however with the drawbacks discussed before.
We only derived formally the equations related to the second order condition (i.e. (1.26) and (1.27))—also for the time discrete problem. In our case this was reasonable as the Hessian appeared as second order information in a Newton type algorithm, where it is common to only involve approximations. If one were to do a more rigorous treatment for the time continuous case, one is expected to encounter the same problems as for the first order conditions discussed above. But note that there are already issues for the discrete equations. Recall again that the third derivative of A can be divergent. To the author's knowledge there does not exist any literature treating this problem for the present kind of quasilinearities—even for the elliptic case. We refer to [26, 31] as some exemplary selection of second order treatments of other kinds of quasilinear terms.
Instead of a deeper analytical treatment of the equations considered throughout the thesis, one may enhance the present discussion by coupling further equations. For example, to model dendritic growth, the control variable u might be replaced by the solution of a heat equation [51]. The new control variable could then enter for instance at the Dirichlet boundary of that newly coupled equation. This will probably yield more realistic numerical results. Also here only the forward problem has been investigated in the literature so far. For instance, in [15] the authors enhance their splitting scheme from [13] to this new setting. As one is expected to get the same problems as we encountered in Section 2.7, one might again try an implicit discretization first. Provided that one wants to derive the first and possibly also second order conditions, the new equation will render the analysis more difficult of course. On the other hand, the most problematic terms will stay the anisotropy and the double-well potential that both appeared already in the mere Allen-Cahn equation.
Finally, also to the numerical algorithm some extensions or modifications could be applied. The biggest potential seen by the author is to speed up the solution process of the trust region subproblem (3.18). This was also attempted at some points in the thesis and in the implementation. For instance, in Section 3.2 we addressed the problem of preconditioning the employed Steihaug-CG method. As already summarized above this turned out to be rather involved for the present problem and we were not able to give a final answer here. Probably some new approaches should be developed here, possibly also for a simpler model problem first. Another idea would be to simply replace the exact evaluation of the Hessian performed by the Steihaug-CG ansatz with some approximation that can be computed in a cheaper way. Here the BFGS or rather L-BFGS method [104] seems to be suitable. Also some simplifications of the appearing PDEs, like dropping costly terms, may be considered. Of course, also completely different algorithms could be tried out.
As we have seen, the optimal control of anisotropic phase field models is a rich field of research with large future potential and the contributions of this thesis will hopefully provide a good starting point for prospective projects.

Bibliography

[1] M. Alfaro et al. “Motion by anisotropic mean curvature as sharp interface limit of an inhomogeneous and anisotropic Allen–Cahn equation”. *Proceedings of the Royal Society of Edinburgh: Section A Mathematics* 140.4 (2010), 673–706.

[2] S. M. Allen and J. W. Cahn. “A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening”. *Acta metallurgica* 27.6 (1979), pp. 1085–1095.

[3] M. S. Alnæs. “UFL: a Finite Element Form Language”. *Automated Solution of Differential Equations by the Finite Element Method, Volume 84 of Lecture Notes in Computational Science and Engineering.* Ed. by A. Logg, K.-A. Mardal, and G. N. Wells. Springer, 2012. Chap. 17.

[4] M. S. Alnæs, A. Logg, and K.-A. Mardal. “UFC: a Finite Element Code Generation Interface”. *Automated Solution of Differential Equations by the Finite Element Method, Volume 84 of Lecture Notes in Computational Science and Engineering.* Ed. by A. Logg, K.-A. Mardal, and G. N. Wells. Springer, 2012. Chap. 16.

[5] M. S. Alnæs et al. “The FEniCS Project Version 1.5”. *Archive of Numerical Software* 3.100 (2015).

[6] M. S. Alnæs et al. “Unified Form Language: A domain-specific language for weak formulations of partial differential equations”. *ACM Transactions on Mathematical Software* 40.2 (2014).

[7] M. S. Alnæs et al. “Unified Framework for Finite Element Assembly”. *International Journal of Computational Science and Engineering* 4.4 (2009), pp. 231–244.

[8] H. W. Alt and R. Nürnberg. *Linear Functional Analysis: An Application-Oriented Introduction.* Universitext. Springer London, 2016.

[9] S. Balay et al. “Efficient Management of Parallelism in Object Oriented Numerical Software Libraries”. *Modern Software Tools in Scientific Computing.* Ed. by E. Arge, A. M. Bruaset, and H. P. Langtangen. Birkhauser Press, 1997, pp. 163–202.

[10] S. Balay et al. *PETSc Users Manual.* Tech. rep. ANL-95/11 - Revision 3.15. Argonne National Laboratory, 2021.

[11] S. Balay et al. *PETSc Web page.* https://petsc.org/. 2021.

[12] E. Bänsch, P. Morin, and R. H. Nochetto. “Preconditioning a class of fourth order problems by operator splitting”. *Numerische Mathematik* 118.2 (2011), pp. 197–228.

[13] J. W. Barrett, H. Garcke, and R. Nürnberg. “On the stable discretization of strongly anisotropic phase field models with applications to crystal growth”. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik* 93.10–11 (2013), pp. 719–732.

[14] J. W. Barrett, H. Garcke, and R. Nürnberg. “A variational formulation of anisotropic geometric evolution equations in higher dimensions”. *Numerische Mathematik* 109.1 (2008), pp. 1–44.

[15] J. W. Barrett, H. Garcke, and R. Nürnberg. “Stable phase field approximations of anisotropic solidification”. *IMA Journal of Numerical Analysis* 34.4 (2014), pp. 1289–1327.

[16] J. W. Barrett, H. Garcke, and R. Nürnberg. “Numerical approximation of anisotropic geometric evolution equations in the plane”. *IMA Journal of Numerical Analysis* 28.2 (2007), pp. 292–330. eprint: `http://oup.prod.sis.lan/imajna/article-pdf/28/2/292/1945147/drm013.pdf`.

[17] J. W. Barrett, R. Nürnberg, and V. Styles. “Finite Element Approximation of a Phase Field Model for Void Electromigration”. *SIAM Journal on Numerical Analysis* 42.2 (2004), pp. 738–772. eprint: `https://doi.org/10.1137/S0036142902413421`.

[18] S. Bartels. *Numerical Methods for Nonlinear Partial Differential Equations.* Springer Series in Computational Mathematics. Springer International Publishing, 2015.

[19] G. Bellettini and M. Paolini. “Anisotropic motion by mean curvature in the context of finsler geometry”. *Hokkaido Mathematical Journal* 25.3 (1996), pp. 537–566.

[20] P. Benner and M. Stoll. *Optimal control for Allen–Cahn equations enhanced by model predictive control.* Vol. 1. IFAC, 2013, pp. 139–143.

[21] P. Benner et al. “Low-rank solvers for unsteady Stokes-Brinkman optimal control problem with random data”. *Computer Methods in Applied Mechanics and Engineering* 304 (2016), pp. 26–54.

[22] L. Blank and J. Meisinger. *Optimal control of a quasilinear parabolic equation and its time discretization.* 2021. arXiv: `2102.02616 [math.OC]`.

[23] L. Blank and J. Meisinger. *Optimal control of anisotropic Allen–Cahn equations.* 2021. arXiv: `2105.13310 [math.OC]`.

[24] L. Blank et al. “Optimal control of Allen–Cahn systems”. *Trends in PDE Constrained Optimization.* Ed. by G. Leugering et al. Springer, 2014, pp. 11–26.

[25] J. F. Blowey and C. M. Elliott. “The Cahn–Hilliard gradient theory for phase separation with non-smooth free energy Part I: Mathematical analysis”. *European Journal of Applied Mathematics* 2.3 (1991), pp. 233–280.

[26] L. Bonifacius and I. Neitzel. “Second order optimality conditions for optimal control of quasilinear parabolic equations”. *Mathematical Control & Related Fields* 8.2156-8472_2018_1_1 (2018), pp. 1–34.

[27] C. Borchert et al. “On the prediction of crystal shape distributions in a steady-state continuous crystallizer”. *Chemical Engineering Science* 64.4 (2009), pp. 686–696.

[28] F. Boyer and P. Fabrie. *Mathematical Tools for the Study of the Incompressible Navier-Stokes Equations and Related Models.* Applied Mathematical Sciences. Springer New York, 2012.

[29] J. Bramble. *Multigrid Methods.* CRC Press, 2019.

[30] E. Burman and J. Rappaz. “Existence of solutions to an anisotropic phase-field model”. *Mathematical methods in the applied sciences* 26.13 (2003), pp. 1137–1160.

[31] E. Casas and F. Tröltzsch. “First- and second-order optimality conditions for a class of optimal control problems with quasilinear elliptic equations”. *SIAM Journal on Control and Optimization* 48 (2009), pp. 688–718.

[32] E. Casas and K. Chrysafinos. “Analysis and optimal control of some quasilinear parabolic equations”. *Mathematical Control & Related Fields* 8.2156-8472_2018_3-4_607 (2018), pp. 607–623.

[33] E. Casas and L. A. Fernández. *Boundary control of quasilinear elliptic equations.* Research Report RR-0782. INRIA, 1988.

[34] E. Casas and L. A. Fernández. “Dealing with Integral State Constraints in Boundary Control Problems of Quasilinear Elliptic Equations”. *SIAM Journal on Control and Optimization* 33.2 (1995), pp. 568–589.

[35] E. Casas and L. A. Fernández. “Distributed Control of Systems Governed by a General Class of Quasilinear Elliptic Equations”. *Journal of Differential Equations* 104.1 (1993), pp. 20–47.

[36] E. Casas and L. A. Fernández. “Optimal control of quasilinear elliptic equations with non differentiable coefficients at the origin”. *Revista Matemática de la Universidad Complutense de Madrid* 4 (1991), pp. 227–250.

[37] E. Casas and L. A. Fernández. “Optimal control of quasilinear multistate elliptic systems”. *Analysis and Optimization of Systems*. Springer, 1988, pp. 393–406.

[38] E. Casas, L. A. Fernández, and J. Yong. “Optimal control of quasilinear parabolic equations”. *Proceedings of The Royal Society A: Mathematical, Physical and Engineering Sciences* 125.3 (1995), pp. 545–565.

[39] A. R. Christian Clason Vu Huu Nhu. “Optimal control of a non-smooth quasilinear elliptic equation”. *Mathematical Control and Related Fields* 11.3 (2021), pp. 521–554.

[40] P. Colli, M. H. Farshbaf-Shaker, and J. Sprekels. “A deep quench approach to the optimal control of an Allen–Cahn equation with dynamic boundary conditions and double obstacles”. *Applied Mathematics and Optimization* 71 (2015), pp. 1–24.

[41] P. Colli and J. Sprekels. “Optimal control of an Allen–Cahn equation with singular potentials and dynamic boundary condition”. *SIAM Journal on Control and Optimization* 53.1 (2015), pp. 213–234.

[42] P. Colli et al. “Optimal boundary control of a viscous Cahn–Hilliard system with dynamic boundary condition and double obstacle potentials”. *SIAM Journal on Control and Optimization* 53.4 (2015), pp. 2696–2721.

[43] A. Conn, N. Gould, and P. Toint. *Trust Region Methods*. MPS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, 2000.

[44] B. Dacorogna. *Direct Methods in the Calculus of Variations*. Applied Mathematical Sciences. Springer Berlin Heidelberg, 2012.

[45] K. Deckelnick, G. Dziuk, and C. M. Elliott. “Computation of geometric partial differential equations and mean curvature flow”. *Acta numerica* 14 (2005), pp. 139–232.

[46] X. Du et al. “Inexact and truncated parareal-in-time Krylov subspace methods for parabolic optimal control problems”. *Electronic Transactions on Numerical Analysis* 40 (2013), pp. 36–57.

[47] X. Du et al. “Varying iteration accuracy using inexact conjugate gradients in control problems governed by pde’s”. *Proceedings of the 2nd Annual International Conference on Computational Mathematics, Computational Geometry and Statistics (CMCGS 2013)*. 2008, pp. 29–38.

[48] C. Eck, H. Garcke, and P. Knabner. *Mathematical Modeling*. Springer Undergraduate Mathematics Series. Springer International Publishing, 2017.

[49] J. Eggleston, G. B. McFadden, and P. Voorhees. “A phase-field model for highly anisotropic interfacial energy”. *Physica D: Nonlinear Phenomena* 150.1 (2001), pp. 91–103.

[50] H. Eisenschmidt, A. Voigt, and K. Sundmacher. “Face-Specific Growth and Dissolution Kinetics of Potassium Dihydrogen Phosphate Crystals from Batch Crystallization Experiments”. *Crystal Growth & Design* 15.1 (2014), pp. 219–227.

[51] C. M. Elliott and A. R. Gardiner. “Double obstacle phase field computations of dendritic growth”. CMAIA University of Sussex Report 1996-19 (1996).

[52] C. M. Elliott, A. R. Gardiner, and T. Kuhn. *Generalized double obstacle phase field approximation of the anisotropic mean curvature flow*. Tech. rep. Technical report, University of Sussex CMAIA Research report, 1996.

[53] C. M. Elliott and R. Schätzle. “The limit of the anisotropic double-obstacle Allen–Cahn equation”. *Proceedings of the Royal Society of Edinburgh: Section A Mathematics* 126 (1996), pp. 1217–1234.

[54] K. Eriksson et al. *Computational differential equations.* Cambridge University Press, 1996.

[55] R. D. Falgout. “An Introduction to Algebraic Multigrid”. *Computing in Science and Engineering* 8.6 (2006), pp. 24–33.

[56] M. H. Farshbaf-Shaker and C. Hecht. “Optimal control of elastic vector-valued Allen–Cahn variational inequalities”. *SIAM Journal on Control and Optimization* 54.1 (2016), pp. 129–152. eprint: `https://doi.org/10.1137/130937354`.

[57] J. Fischer. “Optimal Control Problems Governed by Nonlinear Partial Differential Equations and Inclusions”. PhD thesis. Bayreuth, 2010. eprint: `https://eref.uni-bayreuth.de/4377/`.

[58] M. P. Forum. *MPI: A Message-Passing Interface Standard.* Tech. rep. USA, 1994.

[59] K. Fujiwara et al. “Growth of structure-controlled polycrystalline silicon ingots for solar cells by casting”. *Acta Materialia* 54.12 (2006), pp. 3191–3197.

[60] S. W. Funke and P. E. Farrell. *A framework for automated PDE-constrained optimisation.* 2013. arXiv: `1302.3894 [cs.MS]`.

[61] H. Garcke. “On Cahn-Hilliard systems with elasticity”. *Royal Society of Edinburgh - Proceedings A* 133.2 (2003), pp. 307–331.

[62] D. Gilbarg and N. Trudinger. *Elliptic Partial Differential Equations of Second Order.* Classics in Mathematics. Springer Berlin Heidelberg, 2015.

[63] G. Golub and R. Varga. “Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods - Part I”. *Numerische Mathematik* 3 (1961).

[64] C. Gräser, R. Kornhuber, and U. Sack. “Time discretizations of anisotropic Allen–Cahn equations”. *IMA Journal of Numerical Analysis* 33.4 (2013), pp. 1226–1244.

[65] C. Gräßle et al. *Simulation and Control of a Nonsmooth Cahn-Hilliard Navier-Stokes System with Variable Fluid Densities.* 2019. arXiv: `1907.04285 [math.OC]`.

[66] K. Gröger. “A $W^{1,p}$-estimate for solutions to mixed boundary value problems for second order elliptic differential equations”. *Mathematische Annalen* 283 (1989), pp. 679–687.

[67] F. Guillén-González and G. Tierra. “On linear schemes for a Cahn–Hilliard diffuse interface model”. *Journal of Computational Physics* 234.1 (2013), pp. 140–171.

[68] F. Guillén-González and G. Tierra. “Second order schemes and time-step adaptivity for Allen–Cahn and Cahn–Hilliard models”. *Computers and Mathematics with Applications* 68.8 (2014), pp. 821–846.

[69] M. E. Gurtin. *Thermomechanics of Evolving Phase Boundaries in the Plane.* Oxford mathematical monographs. Clarendon Press, 1993.

[70] W. Hackbusch. *Multi-Grid Methods and Applications.* Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2013.

[71] R. Haller-Dintelmann et al. “Hölder continuity and optimal control for nonsmooth elliptic problems”. *Applied Mathematics and Optimization* 60.3 (2009), pp. 397–428.

[72] M. Heinkenschloss. “A time-domain decomposition iterative method for the solution of distributed linear quadratic optimal control problems”. *Journal of Computational and Applied Mathematics* 173.1 (2005), pp. 169–198.

[73] M. Heinkenschloss and D. Ridzal. “A Matrix-Free Trust-Region SQP Method for Equality Constrained Optimization”. *SIAM Journal on Optimization* 24.3 (2014), pp. 1507–1541.

[74] R. Herzog and C. Meyer. “Optimal control of static plasticity with linear kinematic hardening”. *ZAMM Zeitschrift fur Angewandte Mathematik und Mechanik* 91.10 (2011), pp. 777–794.

[75] M. R. Hestenes and E. Stiefel. “Methods of conjugate gradients for solving linear systems”. *Journal of research of the National Bureau of Standards* 49 (1952), pp. 409–436.

[76] J. E. Hicken. “Inexact Hessian-vector products in reduced-space differential-equation constrained optimization”. *Optimization and Engineering* 15.3 (2014), pp. 575–608.

[77] M. Hintermüller, M. Hinze, and C. Kahle. “An adaptive finite element Moreau–Yosida-based solver for a coupled Cahn–Hilliard/Navier–Stokes system”. *Journal of Computational Physics* 235 (2013), pp. 810–827.

[78] M. Hintermüller and D. Wegner. “Distributed optimal control of the Cahn–Hilliard system including the case of a double-obstacle homogeneous free energy density”. *SIAM Journal on Control and Optimization* 50.1 (2012), pp. 388–418.

[79] M. Hinze et al. *Optimization with PDE Constraints.* Mathematical Modelling: Theory and Applications. Springer Netherlands, 2008.

[80] F. Hoppe and I. Neitzel. “Convergence of the SQP method for quasilinear parabolic optimal control problems”. *Optimization and Engineering* (2020), pp. 1–47.

[81] W. Horn, J. Sokolowski, and J. Sprekels. “Control Problems with State Constraints for the Penrose-Fife Phase-field model”. *Control and Cybernetics* 25 (1996), pp. 1137–1153.

[82] Ethereal (https://stackoverflow.com/users/1546600/ethereals). *C++ - pointer array to Vector?* Stack Overflow. URL: https://stackoverflow.com/a/15203325 (version: 2018-04-25). eprint: `https://stackoverflow.com/a/15203325`.

[83] R. Israel (https://math.stackexchange.com/users/8508/robert israel). *Counterexample for the Chain rule for the Gateaux-derivative.* Mathematics Stack Exchange. URL:https://math.stackexchange.com/q/705876 (version: 2014-03-09). eprint: `https://math.stackexchange.com/q/705876`.

[84] K. Ito and K. Kunisch. *Lagrange Multiplier Approach to Variational Problems and Applications.* Advances in Design and Control. Society for Industrial and Applied Mathematics, 2008.

[85] R. C. Kirby and A. Logg. “A Compiler for Variational Forms”. *ACM Transactions on Mathematical Software* 32.3 (2006).

[86] R. Kobayashi. “Modeling and numerical simulations of dendritic crystal growth”. *Physica D: Nonlinear Phenomena* 63.3 (1993), pp. 410–423.

[87] R. Kruse. *Strong and Weak Approximation of Semilinear Stochastic Evolution Equations.* Vol. 2093. 2014.

[88] C. Lefter and J. Sprekels. “Optimal boundary control of a phase field system modeling nonisothermal phase transitions”. *Advances in Mathematical Sciences and Applications* 17 (2007), p. 181.

[89] J. L. Lions. *Quelques Méthodes de Résolution des Problèmes aux Limites Non Linéaires.* Etudes mathématiques. Paris: Dunod, 1969.

[90] A. Logg, K.-A. Mardal, G. N. Wells, et al. *Automated Solution of Differential Equations by the Finite Element Method.* Springer, 2012.

[91] A. Logg and G. N. Wells. “DOLFIN: Automated Finite Element Computing”. *ACM Transactions on Mathematical Software* 37.2 (2010).

[92] A. Logg, G. N. Wells, and J. Hake. “DOLFIN: a C++/Python Finite Element Library”. *Automated Solution of Differential Equations by the Finite Element Method, Volume 84 of Lecture Notes in Computational Science and Engineering.* Ed. by A. Logg, K.-A. Mardal, and G. N. Wells. Springer, 2012. Chap. 10.

[93] A. Logg et al. “FFC: the FEniCS Form Compiler”. *Automated Solution of Differential Equations by the Finite Element Method, Volume 84 of Lecture Notes in Computational Science and Engineering.* Ed. by A. Logg, K.-A. Mardal, and G. N. Wells. Springer, 2012. Chap. 11.

[94] T. T. Lu and S. H. Shiou. “Inverses of 2 × 2 block matrices”. *Computers and Mathematics with Applications* 43.1-2 (2002), pp. 119–129.

[95] K.-A. Mardal and J. B. Haga. “Block preconditioning of systems of PDEs”. *Automated solution of differential equations by the finite element method.* Springer, 2012, pp. 643–655.

[96] T. P. Mathew, M. Sarkis, and C. E. Schaerer. “Analysis of block parareal preconditioners for parabolic optimal control problems”. *SIAM Journal on Scientific Computing* 32.3 (2010), pp. 1180–1200.

[97] G. B. McFadden et al. “Phase-field models for anisotropic interfaces”. *Physical Review E* 48.3 (1993), pp. 2016–2024.

[98] D. Meidner and B. Vexler. “Adaptive Space-Time Finite Element Methods for Parabolic Optimization Problems”. *SIAM Journal on Control and Optimization* 46.1 (2007), pp. 116–142. eprint: `https://doi.org/10.1137/060648994`.

[99] H. Meinlschmidt, C. Meyer, and J. Rehberg. “Optimal Control of the Thermistor Problem in Three Spatial Dimensions, Part 1: Existence of Optimal Solutions”. *SIAM Journal on Control and Optimization* 55.5 (2017), pp. 2876–2904. eprint: `https://doi.org/10.1137/16M1072644`.

[100] H. Meinlschmidt, C. Meyer, and J. Rehberg. “Optimal Control of the Thermistor Problem in Three Spatial Dimensions, Part 2: Optimality Conditions”. *SIAM Journal on Control and Optimization* 55.4 (2017), pp. 2368–2392. eprint: `https://doi.org/10.1137/16M1072656`.

[101] A. Miranville. “On an anisotropic Allen–Cahn system”. *Cubo (Temuco)* 17.2 (2015), pp. 73–88.

[102] K. Nakajima and N. Usami. *Crystal Growth of Si for Solar Cells.* Springer, 2009.

[103] S. Nicaise and F. Tröltzsch. “Optimal control of some quasilinear Maxwell equations of parabolic type”. *Discrete and Continuous Dynamical Systems - Series S* 10.6 (2017), pp. 1375–1391.

[104] J. Nocedal and S. Wright. *Numerical Optimization.* Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.

[105] R. Nochetto, M. Paolini, and C. Verdi. “A Dynamic Mesh Algorithm for Curvature Dependent Evolving Interfaces”. *Journal of Computational Physics* 123.2 (1996), pp. 296–310.

[106] T. Ohtsuka, K. Shirakawa, and N. Yamazaki. “Optimal control problem for Allen–Cahn type equation associated with total variation energy”. *Discrete and Continuous Dynamical Systems - Series S* 5 (2012), pp. 159–181.

[107] T. Ohtsuka, K. Shirakawa, and N. Yamazaki. “Optimal control problem for Allen–Cahn type equation associated with total variation energy”. *Discrete and Continuous Dynamical Systems - Series S* 5.1 (2012), pp. 159–181.

[108] K. B. Ølgaard and G. N. Wells. “Optimisations for Quadrature Representations of Finite Element Tensors Through Automated Code Generation”. *ACM Transactions on Mathematical Software* 37 (2010).

[109] L. N. Olson and J. B. Schroder. *PyAMG: Algebraic Multigrid Solvers in Python v4.0.* Release 4.0. 2018.

[110] P. Oswald. *Multilevel Finite Element Approximation: Theory and Applications.* Teubner Skripten zur Numerik. Vieweg+Teubner Verlag, 1994.

[111] J. W. Pearson, M. Porcelli, and M. Stoll. “Interior-point methods and preconditioning for PDE-constrained optimization problems involving sparsity terms”. *Numerical Linear Algebra with Applications* 27.2 (2020), e2276. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/nla.2276`.

[112] J. W. Pearson, M. Stoll, and A. J. Wathen. “Regularization-robust preconditioners for time-dependent pde-constrained optimization problems”. *SIAM Journal on Matrix Analysis and Applications* 33.4 (2012), pp. 1126–1152.

[113] A. Rätz and A. Voigt. “Higher order regularization of anisotropic geometric evolution equations in three dimensions”. *Journal of Computational and Theoretical Nanoscience* 3.4 (2006), pp. 560–564.

[114] T. Rees, H. S. Dollar, and A. J. Wathen. “Optimal solvers for PDE-constrained optimization”. *SIAM Journal on Scientific Computing* 32.1 (2010), pp. 271–298.

[115] J. W. Ruge and K. Stüben. “Algebraic multigrid”. *Multigrid methods.* SIAM, 1987, pp. 73–130.

[116] C. Rupprecht. “Numerische Lösung optimaler Steuerung der Allen–Cahn Gleichung inklusive Adaptivität”. Diploma thesis. Regensburg, 2011.

[117] M. Ruzicka. *Nichtlineare Funktionalanalysis: Eine Einführung.* Springer-Lehrbuch Masterclass. Springer Berlin Heidelberg, 2004.

[118] A. Schiela and S. Ulbrich. “Operator preconditioning for a class of inequality constrained optimal control problems”. *SIAM Journal on Optimization* 24.1 (2014), pp. 435–466.

[119] J. Shen and X. Yang. “Numerical approximations of Allen–Cahn and Cahn–Hilliard equations”. *Discrete & Continuous Dynamical Systems* 28.4 (2010), p. 1669.

[120] J. Simon. “Compact sets in the space $L^p(0,T;B)$”. *Annali di Matematica Pura ed Applicata* 146 (1986), pp. 65–96.

[121] V. Simoncini and D. B. Szyld. “Theory of Inexact Krylov Subspace Methods and Applications to Scientific Computing”. 25.2 (2003), pp. 454–477.

[122] G. Stampacchia. “Le problème de Dirichlet pour les équations elliptiques du second ordre à coefficients discontinus”. *Annales de l’institut Fourier* 15.1 (1965), pp. 189–257.

[123] T. Steihaug. “The Conjugate Gradient Method and Trust Regions in Large Scale Optimization”. *SIAM Journal on Numerical Analysis* 20.3 (1983), pp. 626–637.

[124] M. Stoll and A. J. Wathen. “All-at-once solution of time-dependent PDE-constrained optimization problems” (2010), pp. 24–29.

[125] J. E. Taylor and J. W. Cahn. “Linking anisotropic sharp and diffuse surface motion laws via gradient flows”. *Journal of Statistical Physics* 77.1-2 (1994), pp. 183–197.

[126] S. Torabi et al. “A new phase-field model for strongly anisotropic systems”. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 465 (2009), pp. 1337–1359.

[127] F. Tröltzsch and J. Sprekels. *Optimal Control of Partial Differential Equations: Theory, Methods, and Applications.* Graduate studies in mathematics. American Mathematical Society, 2010.

[128] S. Ulbrich. “Generalized SQP methods with “parareal” time-domain decomposition for time-dependent PDE-constrained optimization”. *Real-time PDE-constrained optimization.* SIAM, 2007, pp. 145–168.

[129] S. Ulbrich. “Preconditioners based on “parareal” time-domain decomposition for time-dependent PDE-constrained optimization”. *Multiple Shooting and Time Domain Decomposition Methods.* Springer, 2015, pp. 203–232.

[130] G. Wachsmuth. “Differentiability of implicit functions: Beyond the implicit function theorem”. *Journal of Mathematical Analysis and Applications* 414.1 (2014), pp. 259 –272.

[131] G. Wachsmuth. “Optimal control of quasistatic plasticity”. PhD thesis. Chemnitz, 2011.

[132] G. Wachsmuth. “Optimal Control of Quasistatic Plasticity with Linear Kinematic Hardening, Part I: Existence and Discretization in Time”. *SIAM Journal on Control and Optimization* 50 (2012).

[133] G. Wachsmuth. “Optimal Control of Quasistatic Plasticity with Linear Kinematic Hardening, Part II: Regularization and Differentiability”. *Zeitschrift für Analysis und ihre Anwendungen* 34 (2015), pp. 391–418.

[134] G. Wachsmuth. “Optimal Control of Quasistatic Plasticity with Linear Kinematic Hardening, Part III: Optimality Conditions”. *Zeitschrift für Analysis und ihre Anwendungen* 35 (2016).

[135] A. J. Wathen and T. Rees. “Chebyshev semi-iteration in preconditioning for problems including the mass matrix”. *Electronic Transactions on Numerical Analysis* 34 (2009), pp. 125–135.

[136] S. M. Wise, J. Kim, and J. S. Lowengrub. “Solving the regularized, strongly anisotropic Cahn-Hilliard equation by an adaptive non-linear multigrid method”. *J. Comp. Phys.* 226 (2007), pp. 441–446.

[137] M. Wolff and M. Böhm. “On parameter identification for general linear elliptic problems of second order”. Berichte aus der Technomathematik 18-01. Universität Bremen, Zentrum für Technomathematik, Fachbereich 3-Mathematik und Informatik, 2018.

[138] G. Wulff. “XXV. Zur Frage der Geschwindigkeit des Wachsthums und der Auflösung der Krystallflächen”. *Zeitschrift für Kristallographie - Crystalline Materials* 34 (1901), pp. 449 –530.

[139] E. Zeidler and L. Boron. *Nonlinear Functional Analysis and Its Applications: II/ A: Linear Monotone Operators*. Springer New York, 2012.

[140] E. Zeidler and L. Boron. *Nonlinear Functional Analysis and its Applications: II/B: Nonlinear Monotone Operators*. Springer New York, 2013.

[141] X. Zhang, H. Li, and C. Liu. “Optimal Control Problem for the Cahn–Hilliard/Allen–Cahn Equation with State Constraint”. *Applied Mathematics and Optimization* 82.2 (2020), pp. 721–754.

Acknowledgments

First of all I want to mention the following persons to which I want to express my gratitude:

- my supervisor Prof. Dr. Luise Blank for her steady advice during the work on this thesis and providing the topic
- my co-supervisor Prof. Dr. Eberhard Bänsch for discussions on preconditioning and tracking my progress
- Prof. Dr. Harald Garcke, Dr. Christian Kahle and Prof. Dr. Martin Stoll for useful discussions
- Prof. Dr. Christian Meyer for providing me the PhD thesis of his former student Prof. Dr. Daniel Wachsmuth [131] which also gave much inspirations and directions to this thesis
- Kira Bangert and Jonas Bierler for useful discussions and social interaction
- Dennis Trautwein, Paul Hüttl and Michael Kelly for proofreading

I also thank my family and friends for their constant support.

Finally, I gratefully acknowledge the financial support by the RTG 2339 "Interfaces, Complex Structures, and Singular Limits" of the German Science Foundation (DFG).