Optimization Methods and Software
Vol. 00, No. 00, Month 200x, 1–27

RESEARCH ARTICLE

An Inexact ℓ_1 Penalty SQP Algorithm for PDE Constrained Optimization with an Application to Shape Optimization in Linear Elasticity

Wolfgang Hess* and Stefan Ulbrich

Department of Mathematics, Technische Universität Darmstadt, Dolivostr. 15, 64293 Darmstadt, Germany

(*Received 10 October 2010; in final form 00 Month 200x*)

We develop an optimization algorithm which is able to deal with inexact evaluations of the objective function. The proposed algorithm employs sequential quadratic programming with a line search that uses the ℓ_1 penalty function for an Armijo-like condition. Both the objective gradient computations for the quadratic subproblems and the objective function computations for the line search admit some inexactness which is controlled by the algorithm. We prove convergence results for the presented algorithm under suitable assumptions.

The kind of optimization problems handled by the algorithm arises, e. g., when we apply iterative solvers in the evaluation of the reduced objective function of a shape optimization problem subject to the linear elasticity equations. In the second part of this paper, we present an application of the algorithm to the shape optimization of steel profiles using nested iterations with adaptively refined meshes. It is shown how the inexactness in the algorithm translates to an acceptable residual of the state and adjoint state solutions in the iterative solver. We look at three different stopping criteria for the outer iterations. Numerical results are presented.

Keywords: sequential quadratic programming; line search; inexact objective function evaluations; shape optimization; iterative solution; engineering application

AMS Subject Classification: 49M37; 49Q10; 74P10; 90C55

1. Introduction

In product development, simulation models are often used to predict the behavior of a product prior to manufacturing a prototype. This simulation models can be used not only for the examination of several different designs, but also in shape optimization to generate optimal designs for a given objective function and given design constraints. As an example, the optimal shape of steel profiles with respect to their mechanical properties can be computed for a given load case by utilizing linear elasticity. In this way the application of optimization algorithms can be introduced as part of a product development approach.

In this paper, we examine an inexact SQP algorithm for a certain class of optimization problems. Quite often in the context of PDE constrained optimization, there is only a finite number of control or design parameters which have to be optimized. In contrast, the solution to the PDE which depends on the control or design parameters is an element of some function space. Here, we consider optimization problems in which the solution of the PDE appears in the formulation of the objective function, but in no constraint besides the PDE constraint itself, i. e.,

*Corresponding author. Email: wohess@gmail.com

ISSN: 1055-6788 print/ISSN 1029-4937 online
© 200x Taylor & Francis
DOI: 10.1080/03081080xxxxxxxxx
http://www.informaworld.com

optimization problems of the form

$$\begin{aligned} \min\ & J(y,\mathbf{u}) \\ \text{s. t.}\ & C(y,\mathbf{u}) = 0 \\ & \mathbf{w}(\mathbf{u}) \le \mathbf{0} \\ & \mathbf{u} \in \mathbb{R}^n \\ & y \in Y(\mathbf{u}) \end{aligned} \tag{1}$$

Here, the solution of the (possibly discretized) PDE constraint $C(y,\mathbf{u}) = 0$ is denoted by $y \in Y(\mathbf{u})$, where $Y(\mathbf{u})$ is a function space which possibly depends on the design parameters $\mathbf{u}$ since we want to include shape optimization problems. Only a finite number of design parameters $\mathbf{u}$ exists, which are subject to nonlinear constraints $\mathbf{w}(\mathbf{u}) \le \mathbf{0}$. In contrast to the objective function J, these constraints do not depend on the state y. Assume that unique solutions $y(\mathbf{u})$ exist. Removing the explicit occurrence of the PDE constraint and the state, and rewriting the objective function as $j(\mathbf{u}) = J(y(\mathbf{u}),\mathbf{u})$ yields the corresponding reduced problem. In an actual implementation of a PDE constrained optimization algorithm, approximate solutions of the PDE must be obtained, e. g., using a finite element discretization. This typically leads to large linear systems which need to be solved. Using an iterative solver for these state equations gives rise to inexact evaluations.

In the following, we look at the nonlinear program with an inexact objective function (IO-NLP) given as

$$\begin{aligned} \min\ & j(\mathbf{u}) \\ \text{s. t.}\ & \mathbf{w}(\mathbf{u}) \le \mathbf{0} \\ & \mathbf{u} \in \mathbb{R}^n \end{aligned} \tag{IO-NLP}$$

where the objective function $j : \mathbb{R}^n \to \mathbb{R}$ and the inequality constraints $\mathbf{w} : \mathbb{R}^n \to \mathbb{R}^m$ are assumed to be twice continuously differentiable. We restrict our presentation to inequality constraints although treatment of equality constraints can arguably be added in a straightforward manner. Moreover, we assume that exact evaluations of the constraints and their derivatives are efficient, whereas only expensive and inexact evaluations of the objective function are available.

An inexact SQP algorithms for equality constrained problems using trust-regions is proposed by Heinkenschloss and Vicente [17], an inexact trust-region SQP approach for problems with a state equation and bounds on the control variables is given in a paper by Dennis, Heinkenschloss, and Vicente [11]. For exact evaluations of the objective function and its gradient, SQP methods exist for general inequality constrained problems, see, e. g., [12] by Fletcher, Gould, Leyffer, Toint, and Wächter for a trust-region based approach. We cite the book by Geiger and Kanzow [13] for an exact SQP method using the ℓ_1 penalty function together with a line search, for an overview of SQP methods see also Boggs and Tolle [3], for the original paper proposing globalization of SQP using the ℓ_1 penalty function as a merit function see Han [15]. Usage of the ℓ_1 penalty function in the quadratic subproblems and as a merit function is discussed in [10] by Conn, Gould, and Toint in the context of trust-region SQP methods. Jäger and Sachs [20] look at an SQP method for equality constrained problems that allows for inexact evaluations of an iterative solver. They analyze the global convergence in a Banach space setting and employ a line search based on the ℓ_1 penalty function. The local convergence properties of an SQP algorithm, which solves the quadratic subproblems only inexactly using an interior point approach, are considered in [21] by Leibfritz and Sachs. They present numerical results for an optimal control problem with inequality constraints on the state. Since no globalization is used, an approximate solution is computed as a starting point of the local algorithm. Lately, Cartis, Gould, and Toint proposed globalizing Newton-like optimization iterations by a cubic overestimator – as opposed to a line search or trust-region – for unconstrained optimization problems [8]. An extension to convex feasible domains has been suggested. In [7], Byrd, Curtis, and Nocedal look at the update of the penalty parameter in an SQP algorithm using the ℓ_1 penalty function in the subproblems and the line search.

Here, we are concerned with an ℓ_1 penalty SQP algorithm to solve PDE constrained optimization problems subject to nonlinear inequality constraints that employs a line search and allows for inexact evaluations in both the reduced objective function and its gradient. Hence, we are able to use an iterative solver for the state and adjoint state equations.

This paper is organized as follows. We begin Section 2 with a short motivation and the statement of the algorithm, followed by the convergence results and their proofs. Section 3 contains the presentation of an application of the algorithm to shape optimization of steel profiles subject to linear elasticity. In Section 4, we present numerical results for two example problems – the optimization of a multi-chamber profile and an overhead conveyor – that were solved using the described methods. Finally, we conclude in Section 5 and give a short outlook of possible directions of future research.

2. An Inexact ℓ_1 Penalty SQP Algorithm

We begin this section with a short motivation followed by the statement of the algorithm. We base our algorithm on the globalized SQP method from [13] which employs the ℓ_1 penalty function both in the subproblems and as a merit function for an Armijo-like line search. This algorithm handles nonlinear inequality constraints which is important for the problems we have in mind.

We want to apply this approach to reduced problems (IO-NLP) which are derived from optimization problems (1) with a PDE constraint and have to cope with iterative solvers for the discretized PDE constraint. Hence, we are only able to compute inexact evaluations of the reduced objective function $j(\mathbf{u})$ and its gradient $\nabla j(\mathbf{u})$. This is important in two places: The quadratic subproblems (4) depend on the gradient of the reduced objective function, and the line search inequality (5) depends on the objective function itself. The following algorithm uses inexactness criteria in order to deal with inexact function values and gradients in an appropriate way. In the presented engineering application, outer iterations are added and the algorithm will be applied to a sequence of successively refined problems.

2.1. *Statement of the Algorithm*

We define the ℓ_1 penalty function, which will appear as a merit function in the following algorithm, as

$$P_\rho(\mathbf{u}) = j(\mathbf{u}) + \rho \sum_{i=1}^{m} \max\{0, w_i(\mathbf{u})\}. \tag{2}$$

This is an exact penalty function, i. e., under suitable assumptions and a sufficiently large finite penalty parameter ρ, feasible first-order critical points of (IO-NLP) are stationary points of the penalty function. The ℓ_1 penalty function was suggested by Han [15] as a merit function for the globalization of the SQP method. A detailed discussion of the ℓ_1 penalty function can be found, e.g., in [3] and [13].

In this paper, we propose the following inexact SQP algorithm. First, we state the method and then give a short motivation of its steps.

Algorithm 1 Inexact SQP method

```
1: Initialization
   Choose $(\mathbf{u}^0, \lambda^0) \in \mathbb{R}^n \times \mathbb{R}^m$, $\mathbf{H}_0 \in \mathbb{R}^{n\times n}$ symmetric positive definite, $\rho, \varepsilon_0 > 0$,
   $\beta, \gamma, \sigma \in (0,1)$, $\alpha > 1$, $-2 \le \nu < 0$, $k \leftarrow 0$.
2: Inexact objective gradient
   Compute an approximation $\widetilde{\nabla j}^k$ to the objective gradient at $\mathbf{u}^k$ for which
       $$\|\widetilde{\nabla j}^k - \nabla j(\mathbf{u}^k)\| \le c\varepsilon_k \qquad (3)$$
   holds for some constant $c$ independent of $k$.
3: Quadratic subproblem
   Compute a solution $(\mathbf{s}^k, \xi^k)$ and corresponding Lagrange multipliers $(\check{\lambda}^k, \mu^k)$
   to the inexact quadratic subproblem
       $$\begin{array}{ll} \min & (\widetilde{\nabla j}^k)^T \mathbf{s} + \rho \sum_{i=1}^m \xi_i + \frac{1}{2}\mathbf{s}^T \mathbf{H}_k \mathbf{s} \\ \text{s. t.} & \mathbf{w}(\mathbf{u}^k) + \nabla \mathbf{w}(\mathbf{u}^k)^T \mathbf{s} \le \xi \\ & \xi \ge 0 \\ & \mathbf{s} \in \mathbb{R}^n, \xi \in \mathbb{R}^m \end{array} \qquad (4)$$
4: Vanishing search direction
   If $\mathbf{s}^k = \mathbf{0}$ and $\widetilde{\nabla j}^k = \nabla j(\mathbf{u}^k)$, the algorithm is allowed to stop. Otherwise if
   $\mathbf{s}^k = \mathbf{0}$, then the step fails and we use $(\mathbf{u}^{k+1}, \lambda^{k+1}) = (\mathbf{u}^k, \lambda^k)$, $\mathbf{H}_{k+1} = \mathbf{H}_k$,
   $\varepsilon_{k+1} = \gamma\varepsilon_k$, we set $k \leftarrow k+1$ and continue with Step 2.
5: Inexact line search
   Compute, if it exists, the maximal step size $\delta_k \in \{\beta^0, \beta^1, \beta^2, \dots\} \cap \{\delta \in \mathbb{R} : \delta \ge$
   $\varepsilon_k \|\mathbf{s}^k\|^\nu\}$ with
       $$\widetilde{P_\rho}^k(\mathbf{u}^k + \delta_k \mathbf{s}^k) \le \widetilde{P_\rho}^k(\mathbf{u}^k) - \sigma\delta_k (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k \qquad (5)$$
   where both approximate evaluations $\widetilde{P_\rho}^k(\mathbf{u}^k)$ and $\widetilde{P_\rho}^k(\mathbf{u}^k + \delta_k \mathbf{s}^k)$ of $P_\rho$ are
   chosen to satisfy
       $$|\widetilde{P_\rho}^k(\mathbf{u}) - P_\rho(\mathbf{u})| \le c\varepsilon_k^\alpha \qquad (6)$$
   for some constant $c$ independent of $k$; if the last step was successful, we use
   $\widetilde{P_\rho}^k(\mathbf{u}^k) = \widetilde{P_\rho}^{k-1}(\mathbf{u}^{k-1} + \delta_{k-1}\mathbf{s}^{k-1})$ which already is a suitable approximation,
   otherwise we can use any $\widetilde{P_\rho}^k(\mathbf{u}^k)$ which satisfies (6).
6: Failed step
   If no valid step size $\delta_k \ge \varepsilon_k \|\mathbf{s}^k\|^\nu$ exists, the step fails and we use $(\mathbf{u}^{k+1}, \lambda^{k+1}) =$
   $(\mathbf{u}^k, \lambda^k)$, $\mathbf{H}_{k+1} = \mathbf{H}_k$, $\varepsilon_{k+1} = \gamma\varepsilon_k$, we set $k \leftarrow k+1$ and continue with Step 2.
7: Successful step
   The step is successful and we use $(\mathbf{u}^{k+1}, \lambda^{k+1}) = (\mathbf{u}^k + \delta_k \mathbf{s}^k, \check{\lambda}^k)$, choose $\mathbf{H}_{k+1} \in$
   $\mathbb{R}^{n\times n}$ symmetric positive definite, we use $\varepsilon_{k+1} = \varepsilon_k$, set $k \leftarrow k+1$ and continue
   with Step 2.
```

We now discuss the ingredients of the algorithm. First, we observe that the initial choice $\varepsilon_0 = 0$ would lead to a standard SQP method with globalization by

Table 1. List of symbols

j	(reduced) objective function
$\widetilde{\nabla j}^k$	inexact objective gradient evaluation
$\mathbf{H}_k$	symmetric positive definite approximation of the Hessian of the Lagrangian which is used in the quadratic subproblems
k, l	inner and outer iteration numbers
P_ρ	ℓ_1 penalty function
$\widetilde{P_\rho}^k$	inexact penalty function evaluation
$\mathbf{s}^k$	step computed using the QP subproblem
$\mathbf{u}$	design parameters, an element of $\mathbb{R}^n$
$\mathbf{w}$	inequality constraints only depending on $\mathbf{u}$
y	state in the optimization problem; the displacement in the application to linear elasticity
α	exponent in the inexactness condition for the objective gradient
β	adjustment factor of the line search
γ	adjustment factor of the inexactness tolerance for failed steps
δ_k	step size computed using a line search
ε_k	inexactness tolerance in the inexact SQP method
$\varepsilon'_l, \varepsilon''_l, \varepsilon'''_l$	parameters of the stopping criteria in outer iteration l of the nested iterations
$\check{\lambda}, \lambda$	Lagrange multipliers $\check{\lambda}$ for the QP, λ for the SQP iterations
ν	exponent controlling the relation between step, step size and inexactness
ρ	penalty parameter
σ	parameter of the Armijo-like condition in the SQP method

an ℓ_1 penalty function as considered, e. g., in [13], since then $\widetilde{\nabla j}^k = \nabla j(\mathbf{u}^k)$ and $\widetilde{P_\rho}^k = P_\rho$.

In Step 2 an approximate gradient is computed which satisfies the tolerance criterion (3). In our application, we will compute the reduced gradient $\widetilde{\nabla j}^k$ by using adjoint techniques. Then (3) will allow us to work with inexact solutions of the adjoint equation.

In Step 3 the inexact gradient $\widetilde{\nabla j}^k$ is used in the QP subproblem (4). If the subproblem is solved by $\mathbf{s}^k = 0$, the algorithm stops in Step 4 if $\widetilde{\nabla j}^k = \nabla j(\mathbf{u}^k)$, otherwise the accuracy requirement for $\widetilde{\nabla j}^k$ is strengthened by reducing ε_k.

In Step 5 an Armijo-like line search is performed that is based on inexact evaluations $\widetilde{P_\rho}^k$ of the ℓ_1 penalty function, which have to satisfy the accuracy condition (6) at the current point $\mathbf{u}^k$ and the new trial points. In the context of PDE constrained optimization, the inexactness in $\widetilde{P_\rho}^k$ is caused by inexact function evaluations $\widetilde{j}(\mathbf{u})$ resulting from inexact solutions of the PDE constraint $C(y, \mathbf{u}) = 0$. The tolerance condition (6) can therefore be achieved by a sufficiently accurate solution of the PDE constraint.

If the step size δ_k becomes too small compared to ε_k, then the step $\mathbf{s}^k$ is rejected in Step 6 and ε_k is reduced. Otherwise, the step is accepted in Step 7. It is necessary to choose $\nu < 0$ to ensure that the inexactness tolerance ε_k is decreased when the steps $\mathbf{s}^k$ become too small. On the other hand, for larger steps the choice $\nu \geq -2$ ensures that $\delta_k \|\mathbf{s}^k\|^2$ is large enough such that (5) ensures sufficient decrease in the penalty function. Furthermore, choosing $\alpha > 1$ makes sure that if $\|\mathbf{s}^k\|$ is bounded from below and $\delta_k \to 0$, then the inexactness tolerance ε_k^α becomes small compared to the decrease ensured by (5) and thus (5) eventually ensures sufficient decrease also for the exact penalty function P_ρ.

We will discuss in Section 3 the implementation of Algorithm 1, especially of the accuracy conditions (3) and (6), for shape optimization governed by the linear elasticity equations.

For convenience, the constants of the algorithm and other notations are listed in Table 1.

2.2. *The Quadratic Programming Subproblem*

Before we can prove the desired convergence results, we need to introduce some properties of the quadratic subproblem and its solutions in view of our choice of a merit function, i. e., the ℓ_1 penalty function. We begin by defining

$$\begin{aligned} \mathcal{G}(\mathbf{u}) &= \{i \in \{1,\dots,m\} : w_i(\mathbf{u}) > 0\}, \\ \mathcal{E}(\mathbf{u}) &= \{i \in \{1,\dots,m\} : w_i(\mathbf{u}) = 0\}, \\ \mathcal{N}(\mathbf{u}) &= \{i \in \{1,\dots,m\} : w_i(\mathbf{u}) < 0\}, \end{aligned}$$

and can now write the directional derivative of the ℓ_1 penalty function (2) in the direction $\mathbf{d} \in \mathbb{R}^n$ as

$$\begin{aligned} P'_\rho(\mathbf{u};\mathbf{d}) &= \lim_{h\to 0+} \frac{P_\rho(\mathbf{u}+h\mathbf{d}) - P_\rho(\mathbf{u})}{h} \\ &= \nabla j(\mathbf{u})^T\mathbf{d} + \rho \sum_{i\in\mathcal{G}(\mathbf{u})} \nabla w_i(\mathbf{u})^T\mathbf{d} + \rho \sum_{i\in\mathcal{E}(\mathbf{u})} \max\{0, \nabla w_i(\mathbf{u})^T\mathbf{d}\}. \end{aligned} \tag{7}$$

Next, we state the KKT system of the quadratic subproblem (4) with Lagrange parameters $(\check{\lambda}, \mu)$ as

$$\begin{array}{rrl} \widetilde{\nabla j}^k + \mathbf{H}_k\mathbf{s} + \nabla\mathbf{w}(\mathbf{u}^k)\check{\lambda} = \mathbf{0}, & \rho - \check{\lambda}_i - \mu_i = 0, & \text{(stationarity)} \\ w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T\mathbf{s} - \xi_i \le 0, & \xi_i \ge 0, & \text{(feasibility)} \\ \check{\lambda}_i \ge 0, & \mu_i \ge 0, & \text{(dual feasibility)} \\ \check{\lambda}_i(w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T\mathbf{s} - \xi_i) = 0, & \xi_i\mu_i = 0, & \text{(complementarity)} \\ & \text{for } i \in \{1,\dots,m\}. & \end{array} \tag{8}$$

Note that for every optimal solution of the subproblem (4), it holds that

$$\xi_i = \max\{0, w_i(\mathbf{u}) + \nabla w_i(\mathbf{u})^T\mathbf{s}\} \tag{9}$$

as a ξ_i greater than that implies that there is a descent direction which decreases ξ_i and holds all other variables fixed, while a ξ_i lesser than that implies infeasibility. We can give an unconstrained problem that is equivalent to the quadratic programming problem (4). First, we define a function Φ by

$$\Phi(\mathbf{s};\mathbf{u},\widetilde{\nabla j}) = j(\mathbf{u}) + \widetilde{\nabla j}^T\mathbf{s} + \rho\sum_{i=1}^m \max\{0, w_i(\mathbf{u}) + \nabla w_i(\mathbf{u})^T\mathbf{s}\} \tag{10}$$

and then use it to rewrite problem (4) as

$$\begin{array}{l} \min \Phi(\mathbf{s};\mathbf{u}^k,\widetilde{\nabla j}^k) + \frac{1}{2}\mathbf{s}^T\mathbf{H}_k\mathbf{s} \\ \quad \mathbf{s} \in \mathbb{R}^n. \end{array} \tag{11}$$

Problem (11) is strictly convex since $\mathbf{H}_k$ is symmetric positive definite, and therefore has a unique optimal solution $\mathbf{s}$. With ξ chosen by (9), $(\mathbf{s}, \xi)$ is the unique optimal solution of the convex problem (4).

Lemma 2.1 *Let $(\mathbf{s}, \xi)$ be a solution of the quadratic programming problem (4). Then it holds that*

$$\Phi(\mathbf{s}; \mathbf{u}^k, \widetilde{\nabla j}^k) \le P_\rho(\mathbf{u}^k) - \mathbf{s}^T \mathbf{H}_k \mathbf{s}.$$

Proof This is similar to [13, Lemma 5.44]. For convenience, the proof is adapted to our setting and notation. Let $(\check{\lambda}, \mu)$ be corresponding Lagrange multipliers. Observe that from the definition (2) of the ℓ_1 penalty function, we have

$$j(\mathbf{u}^k) = P_\rho(\mathbf{u}^k) - \rho \sum_{i=1}^m \max\{0, w_i(\mathbf{u}^k)\}, \tag{12}$$

and from the KKT system (8), we can see that

$$(\widetilde{\nabla j}^k)^T \mathbf{s} = -\mathbf{s}^T \mathbf{H}_k \mathbf{s} - \check{\lambda}^T \nabla \mathbf{w}(\mathbf{u}^k)^T \mathbf{s}, \tag{13}$$

$$\check{\lambda}_i \nabla w_i(\mathbf{u}^k)^T \mathbf{s} = \check{\lambda}_i (\xi_i - w_i(\mathbf{u}^k)), \tag{14}$$

$$\max\{0, w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T \mathbf{s}\} \le \xi_i, \tag{15}$$

$$-\check{\lambda}_i \xi_i + \rho \xi_i = (\rho - \check{\lambda}_i - \mu_i)\xi_i = 0, \tag{16}$$

$$\check{\lambda}_i - \rho = -\mu_i. \tag{17}$$

Using the above, we get

$$\begin{aligned}
\Phi(\mathbf{s}; \mathbf{u}^k, \widetilde{\nabla j}^k) &\overset{\substack{(10)\\(12)\\(13)}}{=} P_\rho(\mathbf{u}^k) - \rho \sum_{i=1}^m \max\{0, w_i(\mathbf{u}^k)\} - \mathbf{s}^T \mathbf{H}_k \mathbf{s} - \check{\lambda}^T \nabla \mathbf{w}(\mathbf{u}^k)^T \mathbf{s} \\
&\quad + \rho \sum_{i=1}^m \max\{0, w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T \mathbf{s}\} \\
&\overset{\substack{(14)\\(15)}}{\le} P_\rho(\mathbf{u}^k) - \mathbf{s}^T \mathbf{H}_k \mathbf{s} + \sum_{i=1}^m \big(\min\{0, -\rho w_i(\mathbf{u}^k)\} - \check{\lambda}_i(\xi_i - w_i(\mathbf{u}^k)) + \rho \xi_i \big) \\
&\overset{(16)}{=} P_\rho(\mathbf{u}^k) - \mathbf{s}^T \mathbf{H}_k \mathbf{s} + \sum_{i=1}^m \big(\min\{0, -\rho w_i(\mathbf{u}^k)\} + \check{\lambda}_i w_i(\mathbf{u}^k) \big) \\
&\overset{(17)}{=} P_\rho(\mathbf{u}^k) - \mathbf{s}^T \mathbf{H}_k \mathbf{s} + \sum_{i=1}^m \min\{\check{\lambda}_i w_i(\mathbf{u}^k), -\mu_i w_i(\mathbf{u}^k)\} \\
&\le P_\rho(\mathbf{u}^k) - \mathbf{s}^T \mathbf{H}_k \mathbf{s}
\end{aligned}$$

since it cannot be that both $\check{\lambda}_i w_i(\mathbf{u}^k)$ and $-\mu_i w_i(\mathbf{u}^k)$ are positive as $\check{\lambda}_i, \mu_i \ge 0$. This proves the lemma. ■

Theorem 2.2 *Let $(\mathbf{s}, \xi)$ be a solution of the exact quadratic programming problem (4), i. e., $\widetilde{\nabla j}^k = \nabla j(\mathbf{u}^k)$, with $\mathbf{s} \neq \mathbf{0}$. Then $\mathbf{s}$ is a descent direction of the ℓ_1 penalty* *function, i. e.,*

$$P'_\rho(\mathbf{u}^k;\mathbf{s}) < 0.$$

Proof This is analogous to [13, Theorem 5.45]. For convenience, we give an adapted version of the proof. Its first part is [13, Lemma 5.43 a]. By the definitions (2), (7) of P_ρ and P'_ρ, we have

$$\begin{aligned}
P_\rho(\mathbf{u}^k) + P'_\rho(\mathbf{u}^k;\mathbf{s}) &= j(\mathbf{u}^k) + \nabla j(\mathbf{u}^k)^T\mathbf{s} + \rho \sum_{i\in\mathcal{G}(\mathbf{u}^k)} \big(w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T\mathbf{s}\big) \\
&\quad + \rho \sum_{i\in\mathcal{E}(\mathbf{u}^k)} \max\{0, \nabla w_i(\mathbf{u}^k)^T\mathbf{s}\} \\
&\le j(\mathbf{u}^k) + \nabla j(\mathbf{u}^k)^T\mathbf{s} + \rho\sum_{i=1}^m \max\{0, w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T\mathbf{s}\} \\
&= \Phi(\mathbf{s};\mathbf{u}^k,\widetilde{\nabla j}^k).
\end{aligned}$$

Using this and Lemma 2.1, we get

$$P'_\rho(\mathbf{u}^k;\mathbf{s}) \le -\mathbf{s}^T\mathbf{H}_k\mathbf{s}$$

where the right-hand side is negative since $\mathbf{s} \neq \mathbf{0}$ and $\mathbf{H}_k$ is positive definite. This proves the proposition. ■

2.3. *Convergence Results*

Now we will examine three different cases and prove a convergence result for each of them:

(1) the algorithm stops — see Theorem 2.3
(2) the algorithm eventually fails for every step — see Theorem 2.4
(3) the algorithm produces an infinite subsequence of successful steps — see Theorem 2.5

THEOREM 2.3 *If Algorithm 1 stops at Step 4,* $\mathbf{u}^k$ *is a stationary point of the* ℓ_1 *penalty function, i. e.,*

$$P'_\rho(\mathbf{u}^k;\mathbf{d}) \ge 0 \qquad \forall \mathbf{d}\in\mathbb{R}^n.$$

Proof We have $\widetilde{\nabla j}^k = \nabla j(\mathbf{u}^k)$ and $\mathbf{s}^k = \mathbf{0}$ by assumption. Moreover since $(\mathbf{s}^k, \xi^k)$ with corresponding Lagrange multipliers $(\check{\lambda}^k, \mu^k)$ was chosen as a solution to (4), the system (8) has a solution $(\mathbf{s}, \xi, \check{\lambda}, \mu)$ with $\mathbf{s} = \mathbf{0}$. This leads to

$$\begin{aligned}
&\nabla j(\mathbf{u}^k)^T\mathbf{d} = -\check{\lambda}^T\nabla\mathbf{w}(\mathbf{u}^k)^T\mathbf{d}, \\
&P'_\rho(\mathbf{u}^k;\mathbf{d}) = -\check{\lambda}^T\nabla\mathbf{w}(\mathbf{u}^k)^T\mathbf{d} + \rho\sum_{i\in\mathcal{G}(\mathbf{u}^k)} \nabla w_i(\mathbf{u}^k)^T\mathbf{d} + \rho\sum_{i\in\mathcal{E}(\mathbf{u}^k)} \max\{0, \nabla w_i(\mathbf{u}^k)^T\mathbf{d}\}.
\end{aligned} \tag{18}$$

Again from (8), we get for $w_i(\mathbf{u}^k) < 0$ that $w_i(\mathbf{u}^k) - \xi_i < 0$ and thus $\check{\lambda}_i = 0$ and can rewrite (18) as

$$P'_\rho(\mathbf{u}^k; \mathbf{d}) = \sum_{i \in \mathcal{G}(\mathbf{u}^k)} (\rho - \check{\lambda}_i) \nabla w_i(\mathbf{u}^k)^T \mathbf{d} \\ + \sum_{i \in \mathcal{E}(\mathbf{u}^k)} \max\{-\check{\lambda}_i \nabla w_i(\mathbf{u}^k)^T \mathbf{d}, (\rho - \check{\lambda}_i) \nabla w_i(\mathbf{u}^k)^T \mathbf{d}\}.$$

Now, it suffices to show that for each $i \in \mathcal{G}(\mathbf{u}^k) \cup \mathcal{E}(\mathbf{u}^k)$ the corresponding summand is nonnegative.

For $i \in \mathcal{G}(\mathbf{u}^k)$, we get $\xi_i \geq w_i(\mathbf{u}^k) > 0$, thus $\mu_i = 0$ and $\rho - \check{\lambda}_i = 0$. Therefore, all corresponding summands are zero.

For $i \in \mathcal{E}(\mathbf{u}^k)$, we have $-\check{\lambda}_i \leq 0$ and $\rho - \check{\lambda}_i = \mu_i \geq 0$. Thus if $\nabla w_i(\mathbf{u}^k)^T \mathbf{d} < 0$, we have that $-\check{\lambda}_i \nabla w_i(\mathbf{u}^k)^T \mathbf{d} \geq 0$, if $\nabla w_i(\mathbf{u}^k)^T \mathbf{d} > 0$, we have that $(\rho - \check{\lambda}_i) \nabla w_i(\mathbf{u}^k)^T \mathbf{d} \geq 0$, and if $\nabla w_i(\mathbf{u}^k)^T \mathbf{d} = 0$, then both are zero. Thus, it holds that $\max\{-\check{\lambda}_i \nabla w_i(\mathbf{u}^k)^T \mathbf{d}, (\rho - \check{\lambda}_i) \nabla w_i(\mathbf{u}^k)^T \mathbf{d}\} \geq 0$, and we are done. ∎

Theorem 2.4 *If Algorithm 1 does not stop and does not generate infinitely many successful steps, then the sequence will eventually remain at a stationary point of the ℓ_1 penalty function, i. e., for some $k_0 \in \mathbb{N}$, we have that $\mathbf{u}^{k_0}$ is a stationary point of the penalty function and $\mathbf{u}^k = \mathbf{u}^{k_0}$ for all $k \geq k_0$.*

Proof From the KKT system (8), we get that $0 \leq \check{\lambda}_i^k, \mu_i^k \leq \rho$, i. e., $\check{\lambda}^k$ and μ^k are bounded.

Since there are only finitely many successful steps, there exists a $k_0 \in \mathbb{N}$ such that all steps $k \geq k_0$ fail. For $k \geq k_0$, we have $\mathbf{u}^k = \mathbf{u}^{k_0}$ and thus $\nabla \mathbf{w}(\mathbf{u}^k) = \nabla \mathbf{w}(\mathbf{u}^{k_0})$. Furthermore, we have $\varepsilon_{k+1} = \gamma \varepsilon_k$ for every failed step, and thus we get for $k \to \infty$ that $\varepsilon_k \to 0$ and thus $\widetilde{\nabla j}^k \to \nabla j(\mathbf{u}^{k_0})$. We get from the KKT system (8) that

$$\mathbf{H}_k \mathbf{s}^k = -\widetilde{\nabla j}^k - \nabla \mathbf{w}(\mathbf{u}^k) \check{\lambda}^k,$$

and using the facts from above, we observe that the right-hand side is bounded. Since $\mathbf{H}_k^{-1} = \mathbf{H}_{k_0}^{-1}$, it follows that the $\mathbf{s}^k$ are bounded as well.

Again using the KKT system (8), we have that either $\xi_i^k = 0$ or we have that $\mu_i^k = 0$, $\check{\lambda}_i^k = \rho$ and thus $\xi_i^k = w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T \mathbf{s}^k$. Since for $k \geq k_0$ we have that $\mathbf{w}(\mathbf{u}^k) = \mathbf{w}(\mathbf{u}^{k_0})$ and $\nabla \mathbf{w}(\mathbf{u}^k) = \nabla \mathbf{w}(\mathbf{u}^{k_0})$ and $\mathbf{s}^k$ is bounded, it follows that ξ_i^k is bounded as well.

Now we can pass to a subsequence K for which

$$\lim_{\substack{k \to \infty \\ k \in K}} \check{\lambda}^k = \check{\lambda}^\star, \qquad \lim_{\substack{k \to \infty \\ k \in K}} \mu^k = \mu^\star, \qquad \lim_{\substack{k \to \infty \\ k \in K}} \xi^k = \xi^\star, \qquad \lim_{\substack{k \to \infty \\ k \in K}} \mathbf{s}^k = \mathbf{s}^\star.$$

Since we have

$$\lim_{\substack{k \to \infty \\ k \in K}} \widetilde{\nabla j}^k = \nabla j(\mathbf{u}^{k_0}),$$

we can look at the subsequence K of KKT systems (8) and arrive at the KKT system of the exact subproblem with $(\mathbf{s}^\star, \xi^\star)$ being a solution with corresponding Lagrange multipliers $(\check{\lambda}^\star, \mu^\star)$. If $\mathbf{s}^\star \neq \mathbf{0}$, then we can pass to a subsequence K' of K with $\|\mathbf{s}^k\| \neq 0$ and we have by Theorem 2.2 that $\mathbf{s}^\star$ is a descent direction of P_ρ at $\mathbf{u}^{k_0}$. Thus, Step 6 would eventually succeed since $\lim_{k \to \infty, k \in K'} \varepsilon_k \|\mathbf{s}^k\|^\nu = 0$, i. e., an arbitrarily small, positive step size suffices. This contradicts our assumptions and therefore it must be that $\mathbf{s}^\star = \mathbf{0}$ and the stationarity of $\mathbf{u}^{k_0}$ in the ℓ_1 penalty function follows analogously to Theorem 2.3. ■

THEOREM 2.5 *If Algorithm 1 does not stop and with constants* $c_1, c_2 > 0$ *the assumption*

$$c_1\|\mathbf{s}\|^2 \le \mathbf{s}^T\mathbf{H}_k\mathbf{s} \le c_2\|\mathbf{s}\|^2 \qquad \forall \mathbf{s} \in \mathbb{R}^n, \forall k \in \mathbb{N} \tag{19}$$

holds, then each accumulation point of the generated sequence $\{\mathbf{u}^k\}$ *is a stationary point of the* ℓ_1 *penalty function* P_ρ.

Proof If there are only finitely many successful steps, the desired result follows directly from Theorem 2.4. We now assume that there is an infinite number of successful steps.

Let us first look at the case of only finitely many failed steps. Then there exists a $k_0 \in \mathbb{N}$ such that for $k \ge k_0$ the steps are successful and thus $\varepsilon_k = \varepsilon_{k_0}$ holds. We observe that since we explicitly reuse the approximations $\widetilde{P_\rho}$ for $k \ge k_0$, we get together with $\delta_k \ge \varepsilon_{k_0}\|\mathbf{s}^k\|^\nu$ that

$$\begin{aligned}\widetilde{P_\rho}^{k+1}(\mathbf{u}^{k+1}) &= \widetilde{P_\rho}^{k}(\mathbf{u}^k + \delta_k\mathbf{s}^k)\\ &\le \widetilde{P_\rho}^{k}(\mathbf{u}^k) - \sigma\delta_k(\mathbf{s}^k)^T\mathbf{H}_k\mathbf{s}^k\\ &\le \widetilde{P_\rho}^{k}(\mathbf{u}^k) - \sigma\delta_k c_1\|\mathbf{s}^k\|^2\\ &\le \widetilde{P_\rho}^{k}(\mathbf{u}^k) - \sigma c_1\varepsilon_{k_0}\|\mathbf{s}^k\|^{2+\nu}.\end{aligned}$$

In the case that $\sum_{k\ge k_0}\|\mathbf{s}^k\|^{2+\nu} = \infty$, it follows that $\widetilde{P_\rho}^{k}(\mathbf{u}^k) \to -\infty$ and thus $P_\rho(\mathbf{u}^k) \to -\infty$ with $k \to \infty$ since $\sigma c_1 \varepsilon_{k_0}$ is a positive constant. Thus $\{\mathbf{u}^k\}$ does not have an accumulation point in this case.

If on the other hand we have $\sum_{k\ge k_0}\|\mathbf{s}^k\|^{2+\nu} < \infty$, it follows that $\|\mathbf{s}^k\| \to 0$ with $k \to \infty$ since $2+\nu \ge 0$. Thus, we get from $\nu < 0$ that $\varepsilon_{k_0}\|\mathbf{s}^k\|^\nu \to \infty$ and therefore a step must eventually fail. As this is a contradiction, this case cannot occur.

We have proven the proposition for finitely many failed steps and will now consider the cases with infinitely many failed steps. Let $\mathbf{u}^\star$ be an accumulation point and K be a convergent subsequence. In these cases, we have

$$\lim_{k\to\infty} \varepsilon_k = 0, \tag{20}$$

$$\lim_{\substack{k\to\infty\\ k\in K}} \mathbf{u}^k = \mathbf{u}^\star. \tag{21}$$

Let us first consider the case that $\liminf_{k\to\infty, k\in K}\|\mathbf{s}^k\| = 0$. Then we can pass to a subsequence K' of K such that $\lim_{k\to\infty, k\in K'}\|\mathbf{s}^k\| = 0$. From the KKT system (8) we get that $0 \le \check{\lambda}_i^k, \mu_i^k \le \rho$, i. e., $\check{\lambda}^k$ and μ^k are bounded. Using the KKT system, we have that either $\xi_i^k = 0$ or we have that $\mu_i^k = 0$, $\check{\lambda}_i^k = \rho$ and thus $\xi_i^k = w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T\mathbf{s}^k$. Observe that since w_i and ∇w_i are continuous, we have from (21) that

$$\lim_{\substack{k\to\infty\\ k\in K'}} w_i(\mathbf{u}^k) + \nabla w_i(\mathbf{u}^k)^T\mathbf{s}^k = w_i(\mathbf{u}^\star)$$

and thus $\{\xi^k\}_{K'}$ is bounded. We can therefore pass to a subsequence K'' of K' for which

$$\lim_{\substack{k\to\infty \\ k\in K''}} \check{\lambda}^k = \check{\lambda}^\star, \qquad \lim_{\substack{k\to\infty \\ k\in K''}} \mu^k = \mu^\star, \qquad \lim_{\substack{k\to\infty \\ k\in K''}} \xi^k = \xi^\star, \qquad \lim_{\substack{k\to\infty \\ k\in K''}} \mathbf{H}_k = \mathbf{H}_\star.$$

Furthermore due to (20), (21) and the fact that ∇j is continuous, we have

$$\lim_{\substack{k\to\infty \\ k\in K''}} \widetilde{\nabla j}^k = \nabla j(\mathbf{u}^\star)$$

and so the subsequence K'' of KKT systems (8) converges to the KKT system of the exact subproblem at $\mathbf{u}^\star$ using the symmetric positive definite matrix $\mathbf{H}_\star$ with the solution $(\mathbf{0}, \xi^\star)$ and Lagrange multipliers $(\check{\lambda}^\star, \mu^\star)$. Now the stationarity of $\mathbf{u}^\star$ in the ℓ_1 penalty function follows as in Theorem 2.3.

We now turn to the case that $\liminf_{k\to\infty, k\in K} \|\mathbf{s}^k\| > 0$. As before, we get from the KKT system (8) that $\{\check{\lambda}^k\}_K$ is bounded and that

$$\mathbf{H}_k \mathbf{s}^k = -\widetilde{\nabla j}^k - \nabla \mathbf{w}(\mathbf{u}^k)\check{\lambda}^k.$$

Again using (20), (21) and continuity of ∇j and $\nabla \mathbf{w}$, we observe that the right-hand side is bounded for $k \in K$. Since $\mathbf{H}_k^{-1}$ is bounded by assumption, it follows that $\{\mathbf{s}^k\}_K$ is bounded as well and we can thus pass to a convergent subsequence K' and have

$$\lim_{\substack{k\to\infty \\ k\in K'}} \mathbf{s}^k = \mathbf{s}^\star \neq \mathbf{0} \tag{22}$$

and we can assume w. l. o. g. that $\|\mathbf{s}^k\| \neq 0$ for $k \in K'$ and therefore

$$\lim_{\substack{k\to\infty \\ k\in K'}} \varepsilon_k \|\mathbf{s}^k\|^\nu = 0.$$

This implies in the case of infinitely many unsuccessful steps in K' that there exists a subsequence K'' of steps that fail at Step 6 but were checked for step sizes $\{\tau_k\}_{K''} \to 0$ for which (5) with τ_k in place of δ_k does not hold but $\tau_k \geq \varepsilon_k \|\mathbf{s}^k\|^\nu$ does. Next, we show that the same is true in the case of only finitely many unsuccessful steps in K'. In this case, there must be infinitely many successful steps in K'. Moreover, it must be the case that for a subsequence K'' of successful steps we have

$$\lim_{\substack{k\to\infty \\ k\in K''}} \delta_k = 0$$

because otherwise there exists a subsequence $K^\star$ of K'' of steps with step sizes $\delta_k \geq \delta_\star > 0$ which gives us that

$$\begin{aligned} \widetilde{P_\rho}^{k+1}(\mathbf{u}^{k+1}) - \widetilde{P_\rho}^k(\mathbf{u}^k) &\leq -\sigma\delta_k (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k \\ &\leq -\sigma\delta_\star c_1 \|\mathbf{s}^k\|^2. \end{aligned}$$

Since we have $\{\mathbf{s}^k\}_{K^\star} \to \mathbf{s}^\star$, we get an infinite amount of decrease in $\widetilde{P_\rho}$ by these steps. For all other successful steps that the algorithm takes, we have

$$\widetilde{P_\rho}^{k+1}(\mathbf{u}^{k+1}) - \widetilde{P_\rho}^{k}(\mathbf{u}^{k}) \le -\sigma\delta_k (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k \le 0.$$

For the i-th failed step, we get

$$\begin{aligned}\widetilde{P_\rho}^{k+1}(\mathbf{u}^{k+1}) - \widetilde{P_\rho}^{k}(\mathbf{u}^{k}) &= \widetilde{P_\rho}^{k+1}(\mathbf{u}^{k+1}) - P_\rho(\mathbf{u}^{k+1}) - \widetilde{P_\rho}^{k}(\mathbf{u}^{k}) + P_\rho(\mathbf{u}^k)\\ &\le |\widetilde{P_\rho}^{k+1}(\mathbf{u}^{k+1}) - P_\rho(\mathbf{u}^{k+1})| + |\widetilde{P_\rho}^{k}(\mathbf{u}^{k}) - P_\rho(\mathbf{u}^k)|\\ &\le c\varepsilon_{k+1}^\alpha + c\varepsilon_k^\alpha = c\gamma^{i\alpha}\varepsilon_0^\alpha(1-\gamma^{-\alpha}),\end{aligned}$$

and we can thus give as an upper bound on the increase in $\widetilde{P_\rho}$ by the failed steps

$$\sum_{i=1}^{\infty} c\gamma^{i\alpha}\varepsilon_0^\alpha(1-\gamma^{-\alpha}) = c\varepsilon_0^\alpha(1-\gamma^{-\alpha})\sum_{i=1}^{\infty}(\gamma^\alpha)^i$$

which is finite since $0 < \gamma^\alpha < 1$. Thus we would have $\{\widetilde{P_\rho}^{k}(\mathbf{u}^k)\} \to -\infty$ and therefore $\{P_\rho(\mathbf{u}^k)\} \to -\infty$ and no accumulation point of $\{\mathbf{u}^k\}$ since P_ρ is continuous. This is a contradiction.

Thus, we can assume w. l. o. g. that $\delta_k < 1$ for $k \in K''$ and therefore that the step sizes $\tau_k = \delta_k\beta^{-1}$ were checked but rejected and satisfy $\{\tau_k\}_{K''} \to 0$ and $\tau_k \ge \varepsilon_k \|\mathbf{s}^k\|^\nu$ but not (5) with τ_k in place of δ_k.

Now we proceed as in [13, pages 276–278], and we look at the steps $k \in K''$. We have by the Mean Value Theorem that

$$\begin{aligned}j(\mathbf{u}^k + \tau_k\mathbf{s}^k) &= j(\mathbf{u}^k) + \tau_k \nabla j(\mathbf{u}^k + \theta_k\tau_k\mathbf{s}^k)^T\mathbf{s}^k\\ &= j(\mathbf{u}^k) + \tau_k\nabla j(\mathbf{u}^k)^T\mathbf{s}^k + \underbrace{\tau_k\big(\nabla j(\mathbf{u}^k + \theta_k\tau_k\mathbf{s}^k) - \nabla j(\mathbf{u}^k)\big)^T\mathbf{s}^k}_{\eta_{0,k}}\\ &= j(\mathbf{u}^k) + \tau_k\nabla j(\mathbf{u}^k)^T\mathbf{s}^k + \eta_{0,k}\end{aligned}$$

with $0 < \theta_k < 1$, and since ∇j is continuous, we have

$$\lim_{\substack{k\to\infty\\ k\in K''}} \frac{\eta_{0,k}}{\tau_k} = 0.$$

Likewise, we get for $i \in \{1, \dots, m\}$ that

$$w_i(\mathbf{u}^k + \tau_k\mathbf{s}^k) = w_i(\mathbf{u}^k) + \tau_k\nabla w_i(\mathbf{u}^k)^T\mathbf{s}^k + \eta_{i,k}$$

holds with

$$\lim_{\substack{k\to\infty\\ k\in K''}} \frac{\eta_{i,k}}{\tau_k} = 0.$$

Thus, we obtain for the ℓ_1 penalty function

$$\begin{aligned}P_\rho(\mathbf{u}^k + \tau_k \mathbf{s}^k) &= j(\mathbf{u}^k + \tau_k \mathbf{s}^k) + \rho \sum_{i=1}^m \max\{0, w_i(\mathbf{u}^k + \tau_k \mathbf{s}^k)\} \\ &= j(\mathbf{u}^k) + \tau_k \nabla j(\mathbf{u}^k)^T \mathbf{s}^k + \eta_{0,k} \\ &\quad + \rho \sum_{i=1}^m \max\{0, w_i(\mathbf{u}^k) + \tau_k \nabla w_i(\mathbf{u}^k)^T \mathbf{s}^k + \eta_{i,k}\}.\end{aligned} \tag{23}$$

We define the function Φ as before by

$$\Phi(\mathbf{s}; \mathbf{u}, \nabla j(\mathbf{u})) = j(\mathbf{u}) + \nabla j(\mathbf{u})^T \mathbf{s} + \rho \sum_{i=1}^m \max\{0, w_i(\mathbf{u}) + \nabla w_i(\mathbf{u})^T \mathbf{s}\},$$

and since $\tau \mapsto \Phi(\tau \mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k))$ is convex, we get

$$\begin{aligned}\Phi(\tau_k \mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) &\le (1-\tau_k)\Phi(\mathbf{0}; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) + \tau_k \Phi(\mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) \\ &= P_\rho(\mathbf{u}^k) + \tau_k \Big(\Phi(\mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) - P_\rho(\mathbf{u}^k) \Big).\end{aligned} \tag{24}$$

We write (23) with the function Φ as

$$P_\rho(\mathbf{u}^k + \tau_k \mathbf{s}^k) = \Phi(\tau_k \mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) + \eta_k \tag{25}$$

where

$$\lim_{\substack{k\to\infty \\ k \in K''}} \frac{\eta_k}{\tau_k} = 0. \tag{26}$$

Since τ_k does not satisfy (5) with τ_k in place of δ_k, we get

$$\widetilde{P_\rho}^k(\mathbf{u}^k + \tau_k \mathbf{s}^k) > \widetilde{P_\rho}^k(\mathbf{u}^k) - \sigma \tau_k (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k$$

and thus by (6)

$$P_\rho(\mathbf{u}^k + \tau_k \mathbf{s}^k) + c\varepsilon_k^\alpha > P_\rho(\mathbf{u}^k) - c\varepsilon_k^\alpha - \sigma \tau_k (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k. \tag{27}$$

Now, we see that

$$\begin{aligned}&P_\rho(\mathbf{u}^k) + \tau_k \Big(\Phi(\mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) - P_\rho(\mathbf{u}^k) \Big) + \eta_k \\ &\overset{(24)}{\ge} \Phi(\tau_k \mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) + \eta_k \\ &\overset{(25)}{=} P_\rho(\mathbf{u}^k + \tau_k \mathbf{s}^k) \\ &\overset{(27)}{>} P_\rho(\mathbf{u}^k) - 2c\varepsilon_k^\alpha - \sigma \tau_k (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k\end{aligned}$$

and therefore

$$\Phi(\mathbf{s}^k; \mathbf{u}^k, \nabla j(\mathbf{u}^k)) - P_\rho(\mathbf{u}^k) + \frac{\eta_k + 2c\varepsilon_k^\alpha}{\tau_k} > -\sigma (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k. \tag{28}$$

But from Lemma 2.1, we have

$$\Phi(\mathbf{s}^k; \mathbf{u}^k, \widetilde{\nabla j}^k) \leq P_\rho(\mathbf{u}^k) - (\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k$$

which together with (28) gives us

$$\frac{\eta_k + 2c\varepsilon_k^\alpha}{\tau_k} > (1-\sigma)(\mathbf{s}^k)^T \mathbf{H}_k \mathbf{s}^k + \zeta_k,$$

with

$$\zeta_k = \left(\widetilde{\nabla j}^k - \nabla j(\mathbf{u}^k)\right)^T \mathbf{s}^k, \qquad \lim_{\substack{k\to\infty \\ k\in K''}} \zeta_k = 0,$$

and using the assumption (19), we have

$$\frac{\eta_k + 2c\varepsilon_k^\alpha}{\tau_k} > (1-\sigma) c_1 \|\mathbf{s}^k\|^2 + \zeta_k. \tag{29}$$

Because of $\{\mathbf{s}^k\}_{K''} \to \mathbf{s}^\star$, the sequence $\{\|\mathbf{s}^k\|^{-\nu}\}_{K''}$ is bounded. Thus, it follows from $\tau_k \geq \varepsilon_k \|\mathbf{s}^k\|^\nu$ that

$$\lim_{\substack{k\to\infty \\ k\in K''}} \frac{2c\varepsilon_k^\alpha}{\tau_k} = 0.$$

Together with (26), we see that the limit with respect to K'' of (29) is

$$0 \geq (1-\sigma) c_1 \|\mathbf{s}^\star\|^2$$

where the right-hand side is positive since $\mathbf{s}^\star \neq \mathbf{0}$. This is a contradiction, and hence (22) cannot possibly hold. This completes the proof. ■

This concludes our discussion of the global convergence of Algorithm 1. By Theorem 2.3, we reached a stationary point of the merit function when the algorithm stops, and otherwise Theorem 2.5 shows that only stationary points of the merit function occur as accumulation points.

3. An Application to Shape Optimization

Now, we present the application of the above algorithm to solve shape optimization problems with linear elasticity, followed by the obtained results. These optimization problems arise in the context of product development. We are concerned with steel profiles manufactured from sheet metal using roll forming and linear flow splitting, and our task is to find an optimal shape of the profile given a starting geometry and a load case while holding the topology fixed. The recently developed linear flow splitting process is presented in a paper [14] by Groche, Vucic, and Jöckel. An introductory book to shape optimization is the one by Haslinger and Mäkinen [16], or Hinze et al. [19].

In the following, Section 3.1 gives an overview of the optimization model, including the weak formulation of the elasticity equations and the objective function which we use. We want to tackle the resulting optimization problems by using a

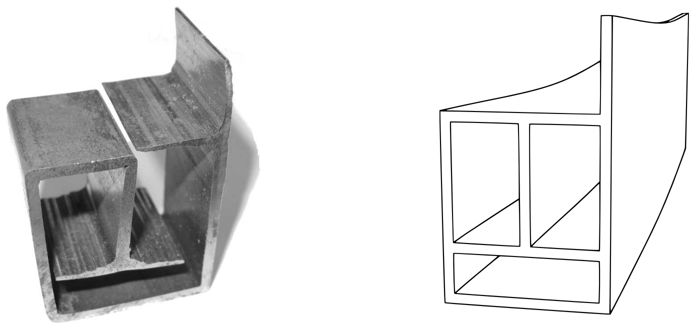

Figure 1. Photograph and model geometry of a steel profile made of branched sheet metal

sequence of successively refined approximate subproblems, which we obtain by employing the finite element method. This is presented in Section 3.2 together with our approach to solve these problems with iterative state and adjoint state solutions by Algorithm 1. Particularly, we develop estimates for stopping the iterative solver which can be easily computed. In Section 3.3, we describe three different stopping criteria that can be used to decide when to move from one approximate subproblem to the next. The results of the computations for two examples are given in Section 4.

3.1. *Optimization Model*

A good introduction to the finite element method is the book [6] by Brenner and Scott. The paper [1] by Alberty, Carstensen, Funken, and Klose describes a finite element code for linear elasticity using linear elements.

We look at three-dimensional shapes. The linear elasticity equations model the behavior of a body under load for small deformations, and their solution $y : \Omega \to \mathbb{R}^3$ is the displacement, i. e., the mapping $\mathbf{x} \mapsto \mathbf{x} + y(\mathbf{x})$ tells us where a point $\mathbf{x} \in \Omega$ of the reference configuration moves under load. For the weak formulation, we use the pure displacement approach and have to find a weak solution $y \in H^1_D(\Omega)^3 + y_D$ which satisfies

$$\int_\Omega \varepsilon(v) : \mathsf{C}\varepsilon(y)\,\mathrm{d}x = \int_\Omega f \cdot v\,\mathrm{d}x + \int_{\Gamma_N} g \cdot v\,\mathrm{d}s \qquad \forall v \in H^1_D(\Omega)^3 \tag{30}$$

where Ω is a Lipschitz domain, $H^1_D(\Omega) = \{v \in H^1(\Omega) : v|_{\Gamma_D} = 0\}$ with the Dirichlet boundary $\Gamma_D \subseteq \partial\Omega$ being a closed subset with positive measure. We denote the Dirichlet boundary values by $y_D \in H^1(\Omega)^3$, the Neumann boundary values by $g \in L^2(\Gamma_N)^3, \Gamma_N = \partial\Omega \setminus \Gamma_D$, and the volume forces by $f \in L^2(\Omega)^3$. We use the notation $\varepsilon(v) = (\nabla v + (\nabla v)^T)/2$ for the symmetric part of the gradient, and $A : B = \sum_{i,j} a_{ij} b_{ij}$.

The material tensor C describes the material law and is the connection between stresses and (linear) strains. In our case it is the material law of a homogeneous, isotropic, linear material, i. e., $\mathsf{C}\varepsilon = \lambda\,\mathrm{tr}(\varepsilon) I + 2\mu\varepsilon$, where only the two Lamé parameters λ and μ remain to be fixed. This parameters can be computed from Young's modulus E and Poisson's ratio ν, and we use $\nu = 0.3$ as we are interested in the material behavior of steel.

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

As we want to optimize the profile geometry, Ω, Γ_D, Γ_N, y_D, f and g depend on the design parameters $\mathbf{u}$. For every $\mathbf{u}$, we get a unique solution $y(\mathbf{u})$ of the weak formulation (30), and hence we can write down the reduced problem, which does not introduce the state y explicitly into the optimization problem, as

$$\begin{aligned} &\min j(\mathbf{u}) = J(y(\mathbf{u}), \mathbf{u}) \\ &\text{s.t. } \mathbf{w}(\mathbf{u}) \le 0 \\ &\quad\ \ \mathbf{u} \in \mathbb{R}^n \end{aligned} \tag{RP}$$

where $J : H^1(\Omega(\mathbf{u}))^3 \times \mathbb{R}^n \to \mathbb{R}$ is the objective function, and $j : \mathbb{R}^n \to \mathbb{R}$ is the corresponding reduced objective function. A frequently used objective function is the compliance

$$J_{\text{compliance}}(y, \mathbf{u}) = \int_{\Omega(\mathbf{u})} f(\mathbf{u}) \cdot y \, \mathrm{d}x + \int_{\Gamma_N(\mathbf{u})} g(\mathbf{u}) \cdot y \, \mathrm{d}s$$

which is linear in y. The presented approach works well in this case, but we want to use the somewhat more challenging mean displacement for our numerical results which is defined by

$$J(y, \mathbf{u}) = J_{\text{displacement}}(y, \mathbf{u}) = \frac{\|y\|^2_{L^2(\Omega(\mathbf{u}))^3}}{\operatorname{vol}(\Omega(\mathbf{u}))}. \tag{31}$$

The inequality constraints $\mathbf{w}$ can be efficiently computed from the design parameters $\mathbf{u}$ alone. They comprise bounds on lengths and thicknesses, available space, mass, the position of the center of mass, and the cross-sectional area of channels in the profile.

3.2. *Finite Element Subproblems*

To compute solutions, we begin at a coarse mesh and solve approximate problems on successively refined meshes until a stopping criterion is satisfied. Our finite element discretization uses quadratic tetrahedral elements which are adaptively refined from one mesh level to the next using an averaging error estimator as in the paper [2] by Bartels and Carstensen and a red-green mesh refinement as described in [4] by Bornemann, Erdmann, and Kornhuber. The error estimates used are reliable and efficient in the sense that with positive constants they constitute an upper and lower bound on the error up to higher order terms.

We want to find solutions to the reduced problem (RP) where $y(\mathbf{u})$ denotes the weak solution corresponding to the design with parameters $\mathbf{u}$. We look at a sequence of approximate problems of the following form where $y_l(\mathbf{u})$ denotes the finite element solution.

$$\begin{aligned} &\min j_l(\mathbf{u}) = J(y_l(\mathbf{u}), \mathbf{u}) \\ &\text{s.t. } \mathbf{w}(\mathbf{u}) \le 0 \\ &\quad\ \ \mathbf{u} \in \mathbb{R}^n. \end{aligned} \tag{ARP$_l$}$$

On each mesh, we solve the approximate problem (ARP$_l$) using Algorithm 1. Depending on the design parameters $\mathbf{u}$, we compute a transformation of a reference mesh which moves the mesh vertices but leaves the mesh topology unchanged. The reference mesh is fixed for each (ARP$_l$) independent of $\mathbf{u}$. Approximate solutions of the state and adjoint equations are computed using the preconditioned conjugate gradient method with a geometric multigrid V-cycle with symmetric Gauß-Seidel smoothing iterations as a preconditioner, see, e. g., the paper [5] by Braess. The preconditioned conjugate gradient method is stopped when an estimate suggests that (6) or (3) is satisfied for the state or adjoint state solutions, respectively. For the conjugate gradient method itself, see [18] where it was proposed by Hestenes and Stiefel.

3.2.1. Objective Derivatives

Let $\mathbf{A}\bar{\mathbf{y}} = \mathbf{b}$ denote the system of linear equations to compute the finite element solution $y_l(\mathbf{u})$, and let $\Psi_l(\mathbf{u})$ denote the corresponding basis and $y_D(\mathbf{u})$ the corresponding Dirichlet boundary condition, i. e., $y_l(\mathbf{u}) = \Psi_l\bar{\mathbf{y}} + y_D$. We write

$$J_l(\bar{\mathbf{y}}, \mathbf{u}) = J(\Psi_l\bar{\mathbf{y}} + y_D, \mathbf{u}). \tag{32}$$

For the objective function derivatives, an adjoint approach is used. For the exact solution $\bar{\mathbf{y}}(\mathbf{u})$, it holds that

$$j_l(\mathbf{u}) = J_l(\bar{\mathbf{y}}(\mathbf{u}), \mathbf{u}) + \bar{\mathbf{p}}^T\mathbf{C}(\bar{\mathbf{y}}(\mathbf{u}), \mathbf{u})$$

where $\mathbf{C}(\bar{\mathbf{y}}, \mathbf{u}) = \mathbf{A}(\mathbf{u})\bar{\mathbf{y}} - \mathbf{b}(\mathbf{u})$. We define the adjoint state $\bar{\mathbf{p}}(\mathbf{u})$ to satisfy

$$\nabla_{\bar{\mathbf{y}}} J_l(\bar{\mathbf{y}}, \mathbf{u}) + \nabla_{\bar{\mathbf{y}}}\mathbf{C}(\bar{\mathbf{y}}, \mathbf{u})\bar{\mathbf{p}} = 0.$$

Observing that $\mathbf{A}$ is symmetric, this leads to $\mathbf{A}\bar{\mathbf{p}} = -\nabla_{\bar{\mathbf{y}}} J_l(\bar{\mathbf{y}}, \mathbf{u})$, i. e., we have to solve the same system with a different right-hand side. The objective function derivative can now be computed as

$$\nabla j_l(\mathbf{u}) = \nabla_{\mathbf{u}} J_l(\bar{\mathbf{y}}, \mathbf{u}) + \nabla_{\mathbf{u}}\mathbf{C}(\bar{\mathbf{y}}, \mathbf{u})\bar{\mathbf{p}}.$$

3.2.2. Inexactness Estimates

We now assume that $\bar{\mathbf{y}}$ solves the system only up to some residual $\bar{\mathbf{r}}$, i. e., $\bar{\mathbf{r}} = \mathbf{b} - \mathbf{A}\bar{\mathbf{y}}$. Using (32), we linearize to get the estimate

$$|J_l(\bar{\mathbf{y}}, \mathbf{u}) - J(y_l(\mathbf{u}), \mathbf{u})| = |\big(\nabla_{\bar{\mathbf{y}}} J_l(\bar{\mathbf{y}}, \mathbf{u})\big)^T\mathbf{A}^{-1}\bar{\mathbf{r}}| + \text{h. o. t.}$$

for the error in the objective function. One possibility would be to estimate $\mathbf{A}^{-1}\bar{\mathbf{r}}$ directly. Such an approximation could, e. g., be computed by applying a multigrid V- or W-cycle. Here we take a different approach and use

$$|\big(\nabla_{\bar{\mathbf{y}}} J_l(\bar{\mathbf{y}}, \mathbf{u})\big)^T\mathbf{A}^{-1}\bar{\mathbf{r}}| \le \|\big(\nabla_{\bar{\mathbf{y}}} J_l(\bar{\mathbf{y}}, \mathbf{u})\big)^T\mathbf{M}^{-1/2}\|_2\|\mathbf{M}^{1/2}\mathbf{A}^{-1}\mathbf{M}^{1/2}\|_2\|\mathbf{M}^{-1/2}\bar{\mathbf{r}}\|_2$$

to get a suitable estimate of the residual norm. While the weak formulation of the PDE suggests using the matrix given by the H^1 inner products of basis functions as $\mathbf{M}$, we estimate the residual in the L^2 norm, since the former is again expensive. Moreover using a lumped mass matrix $\mathbf{M}_L \approx \mathbf{M}$ which is a diagonal matrix, we compute

$$c_1 = \|\big(\nabla_{\bar{\mathbf{y}}} J_l(\bar{\mathbf{y}}, \mathbf{u})\big)^T\mathbf{M}_L^{-1/2}\|_2 \approx \|\big(\nabla_{\bar{\mathbf{y}}} J_l(\bar{\mathbf{y}}, \mathbf{u})\big)^T\mathbf{M}^{-1/2}\|_2$$

and

$$c_2 = \lambda_{\min}^{-1}(\mathbf{M}_L^{-1/2}\mathbf{A}\mathbf{M}_L^{-1/2}) \approx \|\mathbf{M}^{1/2}\mathbf{A}^{-1}\mathbf{M}^{1/2}\|_2.$$

The choice of using the L^2 norm instead of the H^1 norm is made for efficiency of computation. We stop the preconditioned conjugate gradient iterations for the state equation when we satisfy

$$c_1 c_2 \|\mathbf{M}_L^{-1/2}\bar{\mathbf{r}}\|_2 \le c_y \varepsilon_k^\alpha. \tag{33}$$

For the example problems, we choose $c_y = 10^6$ to make sure that the inexact evaluations are not unnecessarily accurate. One possibility to automatically choose a suitable value for c_y is to fix it after the first few preconditioned conjugate gradient iterations at the first step of the second mesh such that (33) is satisfied as an equality. We proceed for the adjoint equation similar to the above. First, we observe that the error in the solution of the state equation is of higher order as we use exponents $\alpha > 1$. This leads to

$$\|\widetilde{\nabla j_l}^k - \nabla j_l(\mathbf{u})\| = \|\nabla_{\mathbf{u}}\mathbf{C}(\bar{\mathbf{y}},\mathbf{u})\mathbf{A}^{-1}\bar{\mathbf{r}}'\| + \text{h.o.t.}$$

where $\bar{\mathbf{r}}'$ is the residual in the adjoint state equation. We compute

$$c_3 = \|\nabla_{\mathbf{u}}\mathbf{C}(\bar{\mathbf{y}},\mathbf{u})\mathbf{M}_L^{-1/2}\|_2 \approx \|\nabla_{\mathbf{u}}\mathbf{C}(\bar{\mathbf{y}},\mathbf{u})\mathbf{M}^{-1/2}\|_2$$

and stop the iterative adjoint state solution when we satisfy

$$c_3 c_2 \|\mathbf{M}_L^{-1/2}\bar{\mathbf{r}}'\|_2 \le c_p \varepsilon_k.$$

For the numerical examples, $c_p = 10^6$ is used. It could be automatically chosen as described above for the parameter c_y.

3.2.3. Quasi-Newton Update

The matrix $\mathbf{H}_k$ in Algorithm 1 is chosen by first computing

$$\mathbf{H}_k' = \mathbf{F}_k + \sum_{i=1}^m \lambda_i^k \nabla^2 w_i(\mathbf{u}^k)$$

where $\mathbf{F}_k$ is a quasi-Newton approximation of the Hessian of the reduced objective function using the symmetric rank-1 update, see, e. g., [9] by Conn, Gould, and Toint for reference. Since we only compute inexact gradients, these are used in the quasi-Newton update, i. e., we use

$$\mathbf{y}^k = \widetilde{\nabla j_l}^{k+1} - \widetilde{\nabla j_l}^k.$$

The symmetric positive matrix $\mathbf{H}_k$ is now computed from $\mathbf{H}_k'$ by changing its eigenvalues such that all of them are 10^{-10} at minimum, i. e., for $\mathbf{H}_k' = \mathbf{Q}\mathbf{D}'\mathbf{Q}^T$ with diagonal matrix $\mathbf{D}'$, we get a new diagonal matrix $\mathbf{D}$ with entries $D_{ii} = \max\{D_{ii}', 10^{-10}\}$ and use $\mathbf{H}_k = \mathbf{Q}\mathbf{D}\mathbf{Q}^T$.

3.2.4. Penalty Parameter Update

Since we are interested in feasible solutions and the penalty parameter ρ might be chosen too small initially, the implementation uses a simple update of the penalty parameter as follows.

$$\rho_{k+1} = \begin{cases} 2\max_i \check{\lambda}_i^k & \text{if } \rho_k \leq 1.1\max_i \check{\lambda}_i^k \\ \rho_k & \text{otherwise.} \end{cases}$$

Observe, that if the penalty parameter is increased only finitely many times, it will eventually stay fixed and otherwise it grows to infinity. Adequate performance of this penalty update requires that the algorithm neither starts at points that exhibit a high degree of infeasibility nor that it reaches such a point due to ρ being too small. For a more elaborate update strategy see Byrd, Curtis, and Nocedal [7] and references therein.

3.3. *Nested Iterations*

We iterate in Algorithm 1 for the approximate reduced problems (ARP$_l$) until some stopping criterion is satisfied. From this we get a solution $\mathbf{u}^l$ for each problem (ARP$_l$) which is the iterate that satisfies the criterion, and we want to choose a stopping criterion which is eventually attained and for which we can show a convergence result for the generated sequence $\{\mathbf{u}^l\}$. For the coarsest mesh problem (ARP$_1$), we use a feasible starting design $\mathbf{u}^0$, $\lambda^0 = \mathbf{0}$, and $\mathbf{H}_0 = \mathbf{I}$. On the other meshes, the final iterates are carried over from the last mesh. The computations on refined meshes are significantly slower, hence it is important to remain on a mesh level as long as it is profitable. Here, a good choice of a stopping criterion and its parameters is crucial. In the following, we look at three different criteria.

3.3.1. ℓ_1 Stopping Criterion A

We begin by looking at two criteria which depend directly on the merit function. Our first choice for the stopping criterion is motivated as follows. We look at a model of the ℓ_1 penalty function that is given by a linearized objective function and linearized constraints as

$$\mathbf{u} \mapsto (\widetilde{\nabla j_l}^k)^T(\mathbf{u} - \mathbf{u}^k) + \rho \sum_{i=1}^{m} \max\{0, \nabla w_i(\mathbf{u}^k)^T(\mathbf{u} - \mathbf{u}^k) + w_i(\mathbf{u}^k)\} + \text{const}.$$

Now, the slope of the secant line of a step from $\mathbf{u}^k$ to $\mathbf{u}^k + \hat{\mathbf{d}}$ is given by

$$\frac{(\widetilde{\nabla j_l}^k)^T\hat{\mathbf{d}} + \rho\sum_{i=1}^{m}\Big(\max\{0, \nabla w_i(\mathbf{u}^k)^T\hat{\mathbf{d}} + w_i(\mathbf{u}^k)\} - \max\{0, w_i(\mathbf{u}^k)\}\Big)}{\|\hat{\mathbf{d}}\|}. \tag{34}$$

Let $\hat{\mathbf{d}} = \varepsilon_l'\mathbf{d}$ with $\varepsilon_l' = \|\hat{\mathbf{d}}\|$. Then $\|\mathbf{d}\| = 1$ and moreover (34) equals

$$(\widetilde{\nabla j_l}^k)^T\mathbf{d} + \rho\sum_{i=1}^{m}\Big(\max\{0, \nabla w_i(\mathbf{u}^k)^T\mathbf{d} + \varepsilon_l'^{-1}w_i(\mathbf{u}^k)\} - \max\{0, \varepsilon_l'^{-1}w_i(\mathbf{u}^k)\}\Big).$$

In the criterion, the exact directional derivative of the ℓ_1 penalty function gets replaced by the above approximation, which uses a secant with length ε_l' in a linear model. The inexactness is limited by an upper bound ε_l''' to the inexactness tolerance. Once the descent that is observable in this model vanishes up to ε_l'', the stopping criterion is satisfied. We use the ℓ_∞ norm of $\mathbf{d}$ to get to a linear program, and state the ℓ_1 Stopping Criterion A as follows. After Step 2 in Algorithm 1, we check whether

$$
\begin{aligned}
&\varepsilon_k \le \varepsilon_l''', \\
&(\widetilde{\nabla j_l}^k)^T \mathbf{d} + \rho \sum_{i=1}^{m} \Big(\max\{0, \nabla w_i(\mathbf{u}^k)^T \mathbf{d} + \varepsilon_l'^{-1} w_i(\mathbf{u}^k)\} - \max\{0, \varepsilon_l'^{-1} w_i(\mathbf{u}^k)\} \Big) \ge -\varepsilon_l'' \\
&\forall \mathbf{d} \in U \text{ with } \|\mathbf{d}\|_\infty \le 1
\end{aligned}
\tag{35}
$$

holds and stop if it does.

To evaluate the criterion, the $\mathbf{d} \in U$ with $\|\mathbf{d}\|_\infty \le 1$ that minimizes the left-hand side of the second inequality of (35) can be computed solving a linear program in $\mathbf{d} \in \mathbb{R}^n$ and auxiliary variables $\zeta \in \mathbb{R}^m$ which is stated as

$$
\begin{array}{lll}
\min & (\widetilde{\nabla j_l}^k)^T \mathbf{d} + \rho \sum_{i=1}^m \left(\zeta_i - \max\{0, \varepsilon_l'^{-1} w_i(\mathbf{u}^k)\} \right) & \\
\text{s.t.} & \zeta_i \ge 0 & \text{for } i \in \{1, \dots, m\} \\
& \zeta_i \ge \nabla w_i(\mathbf{u}^k)^T \mathbf{d} + \varepsilon_l'^{-1} w_i(\mathbf{u}^k) & \text{for } i \in \{1, \dots, m\} \\
& -1 \le d_i \le 1 & \text{for } i \in \{1, \dots, n\}.
\end{array}
$$

3.3.2. ℓ_1 Stopping Criterion B

As an alternative to (35), we use the directional derivative (7) of the ℓ_1 penalty function based on ε-active and -inactive sets. This leads to the ℓ_1 Stopping Criterion B

$$
\begin{aligned}
&\varepsilon_k \le \varepsilon_l''', \\
&(\widetilde{\nabla j_l}^k)^T \mathbf{d} + \rho \sum_{i \in \mathcal{G}^\star_{\varepsilon_l'}(\mathbf{u})} \nabla w_i(\mathbf{u})^T \mathbf{d} + \rho \sum_{i \in \mathcal{E}^\star_{\varepsilon_l'}(\mathbf{u})} \max\{0, \nabla w_i(\mathbf{u})^T \mathbf{d}\} \ge -\varepsilon_l'' \\
&\forall \mathbf{d} \in U \text{ with } \|\mathbf{d}\|_\infty \le 1
\end{aligned}
\tag{36}
$$

where

$$
\begin{aligned}
\mathcal{G}^\star_{\varepsilon_l'}(\mathbf{u}) &= \{ i \in \{1, \dots, m\} : w_i(\mathbf{u}) > \varepsilon_l' \}, \\
\mathcal{E}^\star_{\varepsilon_l'}(\mathbf{u}) &= \{ i \in \{1, \dots, m\} : |w_i(\mathbf{u})| \le \varepsilon_l' \}.
\end{aligned}
$$

The parameter ε_l' determines the tolerance we admit in the constraint values for the ε-active set, whereas the other parameters $\varepsilon_l'', \varepsilon_l'''$ have the same role as before. It is again possible to compute the minimizer $\mathbf{d} \in U$ with $\|\mathbf{d}\|_\infty \le 1$ of the left-hand side of the second inequality of (36) by solving a linear program in $\mathbf{d} \in \mathbb{R}^n$ and auxiliary variables $\zeta \in \mathbb{R}^{m'}$ with $m' = |\mathcal{E}^\star_{\varepsilon_l'}(\mathbf{u})|$

$$
\begin{array}{lll}
\min & (\widetilde{\nabla j_l}^k)^T \mathbf{d} + \rho \sum_{i \in \mathcal{G}^\star_{\varepsilon_l'}(\mathbf{u})} \nabla w_i(\mathbf{u})^T \mathbf{d} + \rho \sum_{i \in \mathcal{E}^\star_{\varepsilon_l'}(\mathbf{u})} \zeta_i & \\
\text{s.t.} & \zeta_i \ge 0 & \text{for } i \in \mathcal{E}^\star_{\varepsilon_l'}(\mathbf{u}) \\
& \zeta_i \ge \nabla w_i(\mathbf{u})^T \mathbf{d} & \text{for } i \in \mathcal{E}^\star_{\varepsilon_l'}(\mathbf{u}) \\
& -1 \le d_i \le 1 & \text{for } i \in \{1, \dots, n\}.
\end{array}
$$

Remark 1 If Algorithm 1 generates infinitely many steps that have an accumulation point, and the assumptions of Theorem 2.5 hold, then the stopping criteria of Sections 3.3.1 and 3.3.2 will eventually be satisfied for every choice of $\varepsilon_l', \varepsilon_l'', \varepsilon_l''' > 0$. Further assuming

$$\{j_l\} \to j, \qquad \{\nabla j_l\} \to \nabla j, \qquad \{\varepsilon_l'\} \to 0, \qquad \{\varepsilon_l''\} \to 0, \qquad \{\varepsilon_l'''\} \to 0,$$

we can show that every accumulation point $\mathbf{u}^\star$ of the sequence of outer iterates $\{\mathbf{u}^l\}$ is a stationary point of the ℓ_1 penalty function of (RP), i. e.,

$$P_\rho'(\mathbf{u}^\star; \mathbf{d}) \geq 0 \qquad \forall \mathbf{d} \in \mathbb{R}^n.$$

3.3.3. KKT Residual Criterion

We also look at nested iterations using the residual of the KKT system of the approximate problems (ARP$_l$) as a stopping criterion. We check after Step 3 in Algorithm 1 using the Lagrange multipliers $\check{\lambda}^k$ of the QP subproblem whether

$$\varepsilon_k \leq \varepsilon_l''', \qquad \left\| \begin{pmatrix} \widetilde{\nabla j_l}^k + \nabla \mathbf{w}(\mathbf{u}^k)\check{\lambda}^k \\ \left(\max\{0, w_i(\mathbf{u}^k)\}\right)_i \\ \left(\min\{\check{\lambda}_i^k, -w_i(\mathbf{u}^k)\}\right)_i \end{pmatrix} \right\| \leq \varepsilon_l'' \tag{37}$$

holds.

Remark 2 In addition to the assumptions of Section 3.3.2, we have to assume that Algorithm 1 stops at a feasible point or generates a feasible accumulation point. We address this by employing the simple penalty parameter update which we described above. Then the stopping criterion will eventually be satisfied, and every accumulation point $\mathbf{u}^\star$ of the sequence of outer iterates $\{\mathbf{u}^l\}$ is a first-order critical point of (RP).

4. Numerical Results

Most of the computations were carried out using computers with 8 Dual-Core AMD Opterons 8220 with 2.8 GHz and 132 GB memory. The computations used a single processor core each.

As described in Section 3.2, we use quadratic finite elements on tetrahedral meshes and refine the meshes adaptively based on a posteriori error estimates. The code is written in Matlab with C and C++ functions where necessary for performance. We use a conjugate gradient solver with a multigrid preconditioner and UMFPACK to obtain direct solutions.

Using direct solutions of the finite element approximations, it is possible to achieve a hierarchy of 6 mesh levels with the starting mesh of our example problem and the chosen refinement strategy before running out of memory. In the case of iterative solutions that we use later, this can be extended to up to 9 mesh levels.

4.1. *Multi-Chamber Profile*

The first example that we will use is the following one. We consider a metal piece with a fixed length of 5 cm. Its width and height are at most 5.2 cm. The geometry of the initial design is depicted in Figure 2 on the left. We have as design parameters $\mathbf{u}$ the lengths and thicknesses which appear in the cross-section of the

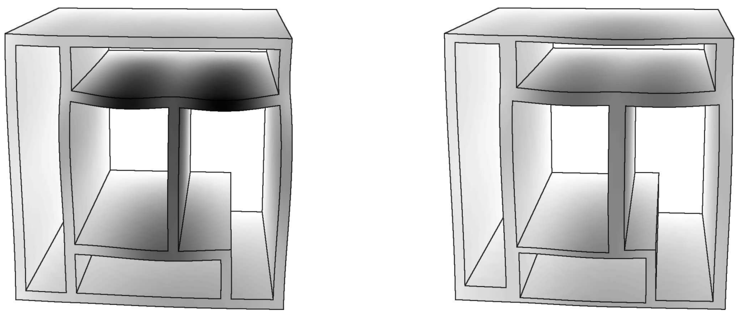

Figure 2. Multi-chamber profile before (left) and after (right) optimization under load (magnified)

metal piece, and we have as constraints **w** multiple channels of a prescribed size, a fixed center of mass, bounds on lengths and thicknesses and an upper bound on the total mass. The piece is fixed at the back, i. e., there we have a homogeneous Dirichlet boundary. On the bottom of the upper right channel, we have a uniform load pushing downwards, i. e., a constant Neumann boundary condition. On the remaining boundary, we prescribe a homogeneous Neumann boundary condition. We also exert a constant volume force as the self load due to gravity. As the objective function, we use the mean displacement objective (31). The state corresponding to the initial design is shown on the right of Figure 2. The different shades of gray show the norm of the displacement.

We use $\varepsilon_0 = 10^{-3}$, $\beta = 0.5$, $\gamma = 0.5$, $\sigma = 0.1$, $\alpha = 1.5$, $\nu = -1.5$. We decrease the tolerances $\varepsilon_l', \varepsilon_l'', \varepsilon_l'''$ of the stopping criteria by a constant factor $1/\varepsilon_{\text{el}}$ after each

Table 2. Numbers of iterations with Criterion (35) for different choices of ε_{el}; the numbers of failed steps are given in parentheses

$l \diagdown \varepsilon_{\text{el}}$	1.0		1.1		1.2		1.4	
1	130	(11)	94	(10)	91	(9)	90	(9)
2	24	(2)	54	(4)	52	(5)	10	(2)
3	23		27		22		39	(1)
4	17		17		14		9	
5	12		12		12		5	
6	10		11		8		9	(1)
7	11	(1)	5		9		3	
8	8		5		9		2	

Table 3. Numbers of iterations with Criterion (36) for different choices of ε_{el}; the numbers of failed steps are given in parentheses

$l \diagdown \varepsilon_{\text{el}}$	1.0		1.1		1.2		1.4	
1	115	(11)	92	(9)	89	(9)	88	(9)
2	18	(2)	39	(5)	41	(5)	6	(1)
3	16		27		27		31	(2)
4	9		6		7		11	
5	4		3		3		2	
6	3		3		3		2	
7	1		1		1		1	
8	1		1		1		1	

Table 4. Numbers of iterations with Criterion (37) for different choices of ε_{el}; the numbers of failed steps are given in parentheses

$l \setminus \varepsilon_{\text{el}}$	1.0		1.1		1.2		1.4	
1	91	(9)	91	(9)	90	(9)	90	(9)
2	52	(5)	46	(5)	13	(2)	9	(2)
3	22		25		31	(1)	48	(4)
4	14		10		8		8	
5	11		7		6		2	
6	6		3		3		2	
7	3		1		1		1	
8	0		1		1		1	

Table 5. Progression of the adaptively refined finite element approximation used for the algorithm with Criterion (36) for $\varepsilon_{\text{el}} = 1.1$

l	elements	vertices	edges	$\bar{\eta}_Z^y$	$\bar{\eta}_Z^p$
1	7 696	2 587	12 854	$3.638 \cdot 10^{-3}$	$1.404 \cdot 10^{-7}$
2	12 666	4 098	20 532	$2.787 \cdot 10^{-3}$	$1.058 \cdot 10^{-7}$
3	24 048	7 753	38 548	$2.142 \cdot 10^{-3}$	$8.020 \cdot 10^{-8}$
4	56 563	17 169	87 388	$1.589 \cdot 10^{-3}$	$5.967 \cdot 10^{-8}$
5	132 450	39 388	201 973	$1.163 \cdot 10^{-3}$	$4.267 \cdot 10^{-8}$
6	311 454	89 653	466 744	$8.359 \cdot 10^{-4}$	$2.984 \cdot 10^{-8}$
7	693 619	196 297	1 030 283	$6.201 \cdot 10^{-4}$	$2.087 \cdot 10^{-8}$
8	1 552 239	429 240	2 280 279	$4.472 \cdot 10^{-4}$	$1.471 \cdot 10^{-8}$

mesh refinement in such a way that we use the same tolerances on the final mesh independent of ε_{el}. The choice $\varepsilon_l''' = 10^{-2}\varepsilon_{\text{el}}^{8-l}$ has little effect as ε_k is always significantly smaller. For the ℓ_1 Stopping Criteria A and B, we use $\varepsilon_l' = 10^{-3}\varepsilon_{\text{el}}^{8-l}$ and $\varepsilon_l' = 10^{-5}\varepsilon_{\text{el}}^{8-l}$, respectively. For all criteria, we use $\varepsilon_l'' = 10^{-6}\varepsilon_{\text{el}}^{8-l}$. The magnitudes of ε_l' and ε_l'' were chosen on the basis of preliminary computations on the coarsest mesh alone. The speed ε_{el} with which the criteria are made stricter remains to be chosen. Results for the ℓ_1 Stopping Criteria A (35) and B (36) for different values of ε_{el} are given in Tables 2 and 3, respectively. We also solved the problem using the KKT Residual Criterion (37), the results of which are given in Table 4.

We show some details of the test run using the ℓ_1 Stopping Criterion (36) with the parameter $\varepsilon_{\text{el}} = 1.1$ in Tables 5 and 6. In Table 5, we can see the progress made by the adaptive mesh refinement. The numbers of tetrahedral mesh elements, vertices and edges are given, as are the error estimates $\bar{\eta}_Z^y$ for the discretization error in the state equation and $\bar{\eta}_Z^p$ for the discretization error in the adjoint state equation. In Table 6, the number of SQP iterations, the final objective value and constraint violation, and the average number of iterations in the preconditioned conjugate gradient method for each inexact solution of the state or adjoint state equation are given for each mesh level. Compared to the starting design, the objective value dropped by 40 %. The mesh after several levels of adaptive refinement is shown in Figure 3.

The wall clock time spent for the optimization is, e. g., 23.7 h, 25.0 h, 24.4 h, or 20.9 h for the ℓ_1 Stopping Criterion (36) with the parameter $\varepsilon_{\text{el}} = 1.0$, $\varepsilon_{\text{el}} = 1.1$, $\varepsilon_{\text{el}} = 1.2$, or $\varepsilon_{\text{el}} = 1.4$, respectively. The optimization using the ℓ_1 Stopping

Table 6. Progression of the algorithm with Criterion (36) for $\varepsilon_{\text{el}} = 1.1$

l	iterations	objective value	constraint violation	∅ pcg iterations
1	92	$4.568 \cdot 10^{-4}$	$1.124 \cdot 10^{-8}$	
2	39	$4.680 \cdot 10^{-4}$	$3.201 \cdot 10^{-7}$	1.929
3	27	$4.761 \cdot 10^{-4}$	$1.176 \cdot 10^{-6}$	4.171
4	6	$4.814 \cdot 10^{-4}$	$8.380 \cdot 10^{-7}$	5.2
5	3	$4.845 \cdot 10^{-4}$	$4.466 \cdot 10^{-7}$	5.636
6	3	$4.862 \cdot 10^{-4}$	$2.364 \cdot 10^{-7}$	5.455
7	1	$4.870 \cdot 10^{-4}$	$2.293 \cdot 10^{-7}$	7.25
8	1	$4.874 \cdot 10^{-4}$	$5.807 \cdot 10^{-8}$	7

Table 7. Running times of the inexact SQP algorithm compared to standard SQP with a direct or a fixed precision iterative solver; Criterion (36) and $\varepsilon_{\text{el}} = 1.1$

l	direct solver			fixed precision			Algorithm 1		
1	1.0 h		(84)	1.1 h		(84)	1.1 h		(93)
2	2.6 h	(+1.6 h)	(38)	3.1 h	(+2.1 h)	(39)	3.5 h	(+2.4 h)	(46)
3	4.8 h	(+2.2 h)	(22)	5.6 h	(+2.4 h)	(23)	6.9 h	(+3.4 h)	(28)
4	6.5 h	(+1.6 h)	(6)	7.1 h	(+1.5 h)	(6)	8.4 h	(+1.5 h)	(8)
5	8.7 h	(+2.3 h)	(3)	9.0 h	(+1.9 h)	(3)	10.3 h	(+1.8 h)	(4)
6	13.7 h	(+4.9 h)	(2)	12.1 h	(+3.1 h)	(2)	13.5 h	(+3.2 h)	(3)
7	23.7 h	(+10.1 h)	(1)	16.0 h	(+3.9 h)	(1)	17.1 h	(+3.6 h)	(1)
8	58.7 h	(+35.0 h)	(1)	25.9 h	(+9.8 h)	(1)	27.1 h	(+10.0 h)	(1)

Criterion (35) took up to 100.3 h for the choice $\varepsilon_{\text{el}} = 1.2$. Of most importance here are the iterations on the finest mesh. Small values for ε_{el}, e. g., $\varepsilon_{\text{el}} = 1.1$, seem to be a reasonable choice. For our example problem criterion (35) performed worse than criteria (36) and (37) both of which worked equally well.

When we look at the running times of different parts of the optimization code for one example, we get the following picture. The total running time (wall clock) was 24.6 h of which 30.4 % were spent computing inexact state and adjoint solutions using pcg, 16.6 % computing the derivatives of the stiffness matrix w. r. t. $\mathbf{u}$, 10.9 % computing the error estimates, 8.5 % assembling the stiffness matrix. The remaining 33.6 % were spent for the rest, i. e., assembling the mass matrix and the right-hand

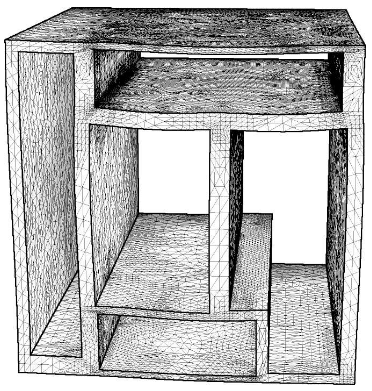

Figure 3. Adaptively refined mesh after several iterations

side; computing the mesh transformation, the objective function, the constraints, and their derivatives; mesh refinement, prolongation matrix; the optimization code, etc. To the time spent for the inexact solutions, the adjoint solutions contributed 56 % and the state solutions 44 %. The direct solutions on the coarsest mesh account for 2.1 % of the total time.

Some result about the efficiency of the inexact SQP algorithm is shown in Table 7. There, we compare the running times of an exact SQP approach using a direct solver for the state and adjoint state equations to the running times of the inexact algorithm. The fixed precision result is that of using Matlab's built-in preconditioned conjugate gradient method with a fixed precision of $\mathrm{tol} = 10^{-4}$ which barely avoids getting stuck when used in an SQP algorithm without regard for inexactness. The choice $\mathrm{tol} = 10^{-3}$ was examined, but in this case after several steps the algorithm failed to generate a suitable descent direction. This is meant for comparison only and we can see that Algorithm 1 is able to achieve competitive performance while determining the required amount of accuracy. These computations had to be done on a different machine with 331 GB memory of which more than 200 GB were used in the computations using direct solutions. The table shows for each refinement level l the cumulative running time followed by the time and number of iterations spent in the SQP algorithm at the specific mesh level.

4.2. *Overhead Conveyor*

Finally, we present a second example and its solution using the inexact algorithm. The length of the profile is 20 cm, the width and height are 6.2 cm. It is fixed on both ends and in addition to the self load, the lower flanges constitute an overhead conveyor track, i. e., a carrier can move in the open channel and the load, which is to be moved, is attached below. Thus, uniform forces are exerted on the upper side of these flanges and the center of mass in x-direction is fixed. The initial and optimized geometries are shown in Figure 4.

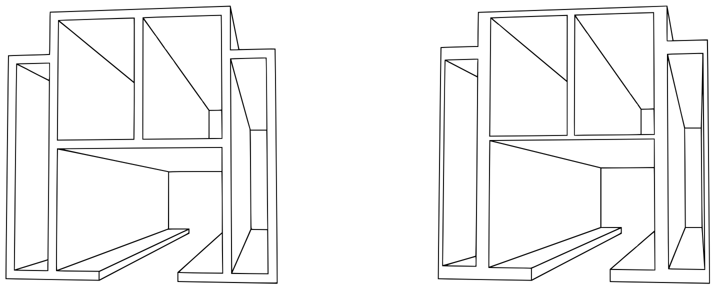
Figure 4. Geometry of the second example before (left) and after (right) the optimization

We choose for the stopping criteria $\varepsilon_l' = 10^{-5}\varepsilon_{\mathrm{el}}^{8-l}$ and $\varepsilon_l'' = 10^{-5}\varepsilon_{\mathrm{el}}^{8-l}$. Hence on the finest mesh, the stopping criteria use the same parameters independent of $\varepsilon_{\mathrm{el}}$. The iterations on the different mesh levels using ℓ_1 Stopping Criterion B (36) and the KKT Residual Criterion (37) are given in Tables 8 and 9, respectively. Running times were between 8.7 h and 15.7 h in wall clock time. Again, we can see that small values for $\varepsilon_{\mathrm{el}}$, e. g., $\varepsilon_{\mathrm{el}} = 1.1$, work well for all stopping criteria. As for the first example, we show the progression of a run of the algorithm in Table 10 by giving for each mesh level, the number of SQP iterations, the final objective value and constraint violation, and the average number of iterations in the preconditioned conjugate gradient method for each inexact solution of the state or adjoint state equation.

Table 8. Numbers of iterations with Criterion (36) for the second example

$l \diagdown \varepsilon_{\mathrm{el}}$	1.0		1.1		1.2		1.4	
1	72	(10)	68	(10)	64	(10)	51	(10)
2	3		3		6		22	
3	2		1		1		0	
4	2		2		1		1	
5	2		1		1		1	
6	0		1		0		0	
7	1		0		1		1	
8	0		0		0		0	

Table 9. Numbers of iterations with Criterion (37) for the second example

$l \diagdown \varepsilon_{\mathrm{el}}$	1.0		1.1		1.2		1.4	
1	68	(10)	65	(10)	65	(10)	48	(10)
2	3		4		4		6	
3	2		1		1		1	
4	2		1		1		26	(2)
5	2		2		1		0	
6	0		0		1		1	
7	0		0		1		0	
8	1		1		0		1	

Table 10. Progression of the algorithm with Criterion (36) for the second example with $\varepsilon_{\mathrm{el}} = 1.1$

l	iterations	objective value	constraint violation	$\varnothing$ pcg iterations
1	68	$4.757 \cdot 10^{-4}$	$1.949 \cdot 10^{-6}$	
2	3	$4.951 \cdot 10^{-4}$	$2.149 \cdot 10^{-6}$	5.667
3	1	$5.101 \cdot 10^{-4}$	$2.466 \cdot 10^{-6}$	12.6
4	2	$5.188 \cdot 10^{-4}$	$3.172 \cdot 10^{-6}$	9.571
5	1	$5.274 \cdot 10^{-4}$	$3.743 \cdot 10^{-6}$	16.4
6	1	$5.327 \cdot 10^{-4}$	$2.636 \cdot 10^{-6}$	20
7	0	$5.357 \cdot 10^{-4}$	$2.636 \cdot 10^{-6}$	37.5
8	0	$5.372 \cdot 10^{-4}$	$2.636 \cdot 10^{-6}$	39

5. Conclusions and Outlook

We developed and analyzed an inexact ℓ_1 penalty SQP algorithm that allows for the use of inexact evaluations of the objective function and its gradient. Numerical results were given for the application of this algorithm to PDE constrained optimization. Here, we applied nested iterations to reduced problems using discretizations of the linear elasticity equations to obtain optimal shapes of steel profiles. For our example problems, a low number of preconditioned conjugate gradient iterations for each inexact evaluation was sufficient for the inexact optimization algorithm. Most SQP iterations were spent on the coarse meshes and only a few on the finer meshes which dominate the computation time.

One possible extension of the current work would be the application of the approach to other PDEs, e. g., heat conduction and radiation, or different geometries, e. g. shapes described by B-spline surfaces.

An improvement of the described method might be obtained by employing other stopping criteria for the nested iterations. As computation times grow quite fast for refined meshes, it seems profitable to stay on a coarse mesh even if only a fraction of the observed decrease in the approximation is present in the exact objective function. The three different stopping criteria that were given here do not take this into account directly. One might devise a criterion which computes the reduction on finer meshes.

Acknowledgements

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) Collaborative Research Centre 666 “Integral Sheet Metal Design with Higher Order Bifurcations – Development, Production, Evaluation”. Moreover, the second author acknowledges the support by the Graduate School “Computational Engineering” at TU Darmstadt funded by the Deutsche Forschungsgemeinschaft (DFG).

References

[1] J. Alberty, C. Carstensen, S.A. Funken, and R. Klose, *Matlab implementation of the finite element method in elasticity*, Computing 69 (2002), pp. 239–263.

[2] S. Bartels and C. Carstensen, *Each averaging technique yields reliable a posteriori error control in FEM on unstructured grids. Part II: Higher order FEM*, Math. Comp. 71 (2002), pp. 971–994.

[3] P.T. Boggs and J.W. Tolle, *Sequential quadratic programming*, Acta Numerica 4 (1995), pp. 1–51.

[4] F. Bornemann, B. Erdmann, and R. Kornhuber, *Adaptive multilevel methods in three space dimensions*, Int. J. Numer. Methods Eng. 36 (1993), pp. 3187–3203.

[5] D. Braess, *On the combination of the multigrid method and conjugate gradients*, in *Multigrid Methods II*, W. Hackbusch and U. Trottenberg, eds., Lecture Notes in Mathematics, Vol. 1228, Springer, 1986, pp. 52–64.

[6] S.C. Brenner and L.R. Scott, *The Mathematical Theory of Finite Element Methods*, 2nd ed., Springer, 2002.

[7] R.H. Byrd, F.E. Curtis, and J. Nocedal, *Infeasibility detection and SQP methods for nonlinear optimization*, SIAM J. Optim. 20 (2010), pp. 2281–2299.

[8] C. Cartis, N.I.M. Gould, and P.L. Toint, *Adaptive cubic regularisation methods for unconstrained optimization. Part I: motivation, convergence and numerical results*, Math. Program. (2009).

[9] A.R. Conn, N.I.M. Gould, and P.L. Toint, *Convergence of quasi-newton matrices generated by the symmetric rank one update*, Math. Program. 50 (1991), pp. 177–195.

[10] A.R. Conn, N.I.M. Gould, and P.L. Toint, *Trust-Region Methods*, MPS-SIAM Series on Optimization, Vol. 1, SIAM, 2000.

[11] J.E. Dennis, M. Heinkenschloss, and L.N. Vicente, *Trust-region interior-point SQP algorithms for a class of nonlinear programming problems*, SIAM J. Control Optim. 36 (1998), pp. 1750–1794.

[12] R. Fletcher, N.I.M. Gould, S. Leyffer, P.L. Toint, and A. Wächter, *Global convergence of a trust-region SQP-filter algorithm for general nonlinear programming*, SIAM J. Optim. 13 (2002), pp. 635–659.

[13] C. Geiger and C. Kanzow, *Theorie und Numerik restringierter Optimierungsaufgaben*, Springer, 2002.

[14] P. Groche, D. Vucic, and M. Jöckel, *Basics of linear flow splitting*, J. of Materials Process. Technol. 183 (2007), pp. 249–255.

[15] S.P. Han, *A globally convergent method for nonlinear programming*, J. Optim. Theory Appl. 22 (1977), pp. 297–309.

[16] J. Haslinger and R.A.E. Mäkinen, *Introduction to shape optimization: Theory, Approximation, and Computation*, SIAM, 2003.

[17] M. Heinkenschloss and L.N. Vicente, *Analysis of inexact trust-region SQP algorithms*, SIAM J. Optim. 12 (2001), pp. 283–302.

[18] M.R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Stand. 49 (1952), pp. 409–436.

[19] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, *Optimization with PDE Constraints*, Mathematical Modelling: Theory and Applications, Vol. 23, Springer, 2009.

[20] H. Jäger and E.W. Sachs, *Global convergence of inexact reduced SQP methods*, Optim. Methods and Software 7 (1997), pp. 83–110.

[21] F. Leibfritz and E.W. Sachs, *Inexact SQP interior point methods and large scale optimal control problems*, SIAM J. Control Optim. 38 (1999), pp. 272–293.