TRUST REGION METHODS WITH HIERARCHICAL FINITE ELEMENT MODELS FOR PDE-CONSTRAINED OPTIMIZATION

ALANA KIRCHNER[†], DOMINIK MEIDNER[‡], AND BORIS VEXLER[§]

Abstract. In this paper, a Hierarchical Trust Region Algorithm for solving PDE-constrained optimization problems is developed. A hierarchy of finite element meshes is used to define a hierarchy of quadratic models for the approximation of the discrete reduced cost functional on the finest mesh. The proposed algorithm simultaneously controls the choice of the model and the size of the trust region radius. Application of the trust region convergence theory allows for proving that every accumulation point of the sequence produced by the algorithm is a stationary point of the discretized problem. Numerical examples illustrate the behavior of the method and show a considerable reduction of computation time compared to the standard Newton trust region scheme.

Key words. trust region, model hierarchy, finite elements, PDE-constrained optimization

AMS subject classifications.

1. Introduction. In this paper we propose a *Hierarchical Trust Region Algorithm* for solving optimization problems governed by (nonlinear) elliptic partial differential equations. The optimization problem under consideration is formally given by

$$\text{Minimize } J(q,u), \quad q \in Q,\ u \in V, \tag{1.1}$$

subject to

$$A(q,u) = f. \tag{1.2}$$

Here, u denotes the state variable from the state space V, q is the control variable from the control space Q, $J\colon Q \times V \to \mathbb{R}$ is the cost functional, and (1.2) is the state equation coupling u and q, see Section 2 for the precise formulation. This setting includes both optimal control and parameter identification problems. For optimal control problems, the operator A is typically given by $A(q,u) = \bar{A}(u) - B(q)$, with a (nonlinear) operator $\bar{A}$ and a (usually linear) control operator B. In parameter identification problems, the variable q denotes the unknown parameters to be determined and may enter the operator A in a nonlinear way.

The use of trust region methods is well accepted in the context of nonlinear optimization problems, see, e.g., the textbooks [3, 10]. In the past several years, these techniques were also applied to optimization problems governed by nonlinear PDEs, see, e.g., [6, 7, 8, 13].

In a usual trust region method for a nonlinear optimization problem

$$\text{Minimize } \ f(x), \quad x \in X$$

one builds up a quadratic model

$$m_k(x_k, \delta x) = f(x_k) + f'(x_k)(\delta x) + \frac{1}{2} g_k(\delta x, \delta x),$$

[†]Lehrstuhl für Mathematische Optimierung, Technische Universität München, Fakultät für Mathematik, Boltzmannstraße 3, 85748 Garching b. München, Germany (kirchner@ma.tum.de)

[‡]Lehrstuhl für Mathematische Optimierung, Technische Universität München, Fakultät für Mathematik, Boltzmannstraße 3, 85748 Garching b. München, Germany (meidner@ma.tum.de)

[§]Lehrstuhl für Mathematische Optimierung, Technische Universität München, Fakultät für Mathematik, Boltzmannstraße 3, 85748 Garching b. München, Germany (vexler@ma.tum.de)

where $x_k \in X$ is a current iterate and $g_k \colon X \times X \to \mathbb{R}$ is a bilinear form. This quadratic function is then (approximately) minimized under the additional constraint $\|\delta x\|_X \leq \Delta_k$ with the current value of the trust region radius Δ_k. Often, a Newton trust region method is applied with the choice $g_k(\delta x, \delta x) = f''(x_k)(\delta x, \delta x)$, which leads (under suitable assumptions) to second order convergence.

In the context of PDE-constrained optimization, the evaluation of second order information is usually rather expensive. There are in general two possibilities to obtain the required second derivatives: For small dimension of the control space Q one may build up the Hessian by solving several linearized PDEs. Another possibility is to compute a matrix-vector product of the Hessian and a given vector by solving one linearized and one adjoint equation in each step of an iterative solver (e.g., conjugate gradient), see, e.g., [1] or Section 2 for details. Applying a usual Newton trust region method for a finite element discretization of a PDE-constrained optimization problem, one has to solve all these additional equations on the current mesh, which may be quite expensive.

The idea of the Hierarchical Trust Region Algorithm is to use a hierarchy of finite element meshes and to define the corresponding hierarchy of quadratic models in the following way: The first part of each trust region model, which contains the first order information (first derivatives) is computed on the current (fine) mesh. For the second part of the model, the equations required for obtaining second order information are solved on a probably coarser mesh. In this way, for each choice of this coarser mesh, a different model is defined. In this paper we provide an algorithm for simultaneous control of the trust region radius Δ_k and the mesh level l_k for obtaining an approximation of the second derivatives in the trust region model.

By numerical experiments, it can be shown that the proposed algorithm leads to a considerable reduction of computation time for all tested configurations with finite- and infinite-dimensional control space. This reduction in terms of computation time is achieved although the reduced accuracy of the model (due to the approximation of the second derivatives on coarser meshes) leads to more steps of the outer trust region loop than in a classical Newton trust region algorithm.

The outline of the paper is as follows: In the next section, we recall the function analytic setting and optimality conditions for the optimal control problem under consideration. In Section 3, the finite element discretization for the state variable and the possibly necessary discretization of the control variable are presented. In Section 4, we sketch a standard general trust region algorithm and present the proposed choice of the approximative Hessian defining the Hierarchical Trust Region Algorithm. Then, Section 5 is devoted to the analysis of the proposed Hierarchical Trust Region Algorithm. We prove that every accumulation point of the sequence produced by the algorithm is a stationary point of the discrete version of the optimization problem under consideration. Finally, in the last section, we present some numerical examples illustrating the behavior of the proposed method.

2. Continuous problem. The optimization problems considered in this paper are formulated in the following abstract setting: Let Q be a Hilbert space for the controls (parameters) with inner product $(\cdot, \cdot)_Q$. Moreover, let V be a Hilbert space for the state. The duality pairing between the Hilbert spaces V and its dual V^* is denoted by $\langle \cdot, \cdot \rangle_{V^* \times V}$. A typical choice for these spaces could be

$$V = \left\{ v \in H^1(\Omega) \;\middle|\; v\big|_{\Gamma_D} = 0 \right\} \quad \text{and} \quad Q = L^2(\Omega),$$

where $\Gamma_D \subset \partial\Omega$ denotes the part of the boundary of Ω with prescribed Dirichlet boundary conditions. We introduce the semi-linear form $a\colon Q \times V \times V \to \mathbb{R}$ which is defined for a differential operator $A\colon Q \times V \to V^*$ by

$$a(q,u)(\varphi) := \langle A(q,u), \varphi\rangle_{V^*\times V}.$$

After these preliminaries, we pose the state equation (1.2) in a weak form: Find for given control $q \in Q$ the state variable $u \in V$ such that

$$a(q,u)(\varphi) = (f,\varphi) \quad \forall \varphi \in V, \tag{2.1}$$

where $f \in L^2(\Omega)$ represents the right hand side of the state equation and $(\cdot,\cdot)$ denotes the inner product of $L^2(\Omega)$.

REMARK 2.1. *There are several sets of assumptions on the nonlinearity in $a(\cdot,\cdot)(\cdot)$ and its dependence on the control variable q allowing the state equation* (2.1) *to be well-posed. Typical examples are different semilinear equations, where the form $a(\cdot,\cdot)(\cdot)$ consists of a linear elliptic part and a nonlinear term depending on u and ∇u. Due to the fact that the development of the proposed trust region algorithm does not depend on the particular structure of the nonlinearity in a, we do not specify a set of assumptions on it, but assume that the state equation* (2.1) *possesses a unique solution $u = u(q) \in V$ for each $q \in Q$.*

Using a functional $\hat{J}\colon V \to \mathbb{R}$ which is assumed to be bounded from below, the cost functional $J\colon Q \times V \to \mathbb{R}$ is defined by

$$J(q,u) = \hat{J}(u) + \frac{\alpha}{2}\|q - \hat{q}\|_Q^2, \tag{2.2}$$

where the regularization (or cost) term is added which involves $\alpha \geq 0$ and a reference control (parameter) $\hat{q} \in Q$.

Then, the corresponding optimization problem is formulated as follows:

$$\text{Minimize } J(q,u) \text{ subject to (2.1) and } (q,u) \in Q \times V. \tag{2.3}$$

The question of existence and uniqueness of solutions to such optimization problems is discussed, e.g., in [9, 4, 12]. Throughout the paper, we assume problem (2.3) to admit a (locally) unique solution $(\bar{q},\bar{u}) \in Q \times V$. Moreover, we assume the existence of a neighborhood $W \subset Q \times V$ of the optimal solution, such that the linearized form $a'_u(q,u)(\cdot,\cdot)$ considered as a linear operator

$$a'_u(q,u)\colon V \to V^*$$

is an isomorphism for all $(q,u) \in W$. This assumption will allow all considered linearized and adjoint problems to be well posed.

Provided the existence of a control-to-state mapping $q \mapsto u(q)$ (see Remark 2.1), we can define the reduced cost functional $j\colon Q \to \mathbb{R}$ by

$$j(q) = J(q,u(q)).$$

This definition allows to reformulate problem (2.3) as an unconstrained optimization problem:

$$\text{Minimize } j(q), \quad q \in Q. \tag{2.4}$$

Throughout this paper, we assume that j is twice continuously differentiable on Q. Then, the first order necessary optimality condition for (2.4) reads as

$$j'(\bar{q})(\delta q) = 0 \quad \forall \delta q \in Q.$$

To construct suitable expressions for the first and second derivatives of the reduced cost functional j, we introduce the Lagrangian $\mathcal{L}\colon Q \times V \times V \to \mathbb{R}$, defined as

$$\mathcal{L}(q,u,z) = J(q,u) + (f,z) - a(q,u)(z). \tag{2.5}$$

With its aid, we obtain the following standard representation of the first derivative $j'(q)(\tau q)$:

PROPOSITION 2.2. *If for given $q \in Q$ the state $u \in V$ fulfills the state equation*

$$\mathcal{L}'_z(q,u,z)(\varphi) = 0, \quad \forall \varphi \in V,$$

and if additionally $z \in V$ is chosen as solution of the adjoint state equation

$$\mathcal{L}'_u(q,u,z)(\varphi) = 0, \quad \forall \varphi \in V,$$

then the following expression of the first derivative of the reduced cost functional holds for given $\tau q \in Q$:

$$j'(q)(\tau q) = \mathcal{L}'_q(q,u,z)(\tau q).$$

In the same manner, one can gain representations of the second derivatives of j in terms of the Lagrangian. It is possible to derive two different kinds of expressions: one is preferable for building up the whole Hessian, the other one for computing matrix-vector products of the Hessian times a given vector. Details on this can be found, e.g., in [1].

PROPOSITION 2.3. *Let the prerequisites of Proposition 2.2 be fulfilled and let $\delta q, \tau q \in Q$ be given directions.*

(a) If $\delta u \in V$ fulfills the linearized state equation

$$\mathcal{L}''_{qz}(q,u,z)(\delta q,\varphi) + \mathcal{L}''_{uz}(q,u,z)(\delta u,\varphi) = 0 \quad \forall \varphi \in V, \tag{2.6}$$

and if $\delta z \in V$ is chosen as the solution of the additionally adjoint equation

$$\mathcal{L}''_{qu}(q,u,z)(\delta q,\varphi) + \mathcal{L}''_{uu}(q,u,z)(\delta u,\varphi) + \mathcal{L}''_{zu}(q,u,z)(\delta z,\varphi) = 0 \quad \forall \varphi \in V, \tag{2.7}$$

then the following expression of the second derivative of the reduced cost functional holds:

$$j''(q)(\delta q,\tau q) = \mathcal{L}''_{qq}(q,u,z)(\delta q,\tau q) + \mathcal{L}''_{uq}(q,u,z)(\delta u,\tau q) + \mathcal{L}''_{zq}(q,u,z)(\delta z,\tau q).$$

(b) If $\delta u, \tau u \in V$ fulfill the linearized state equations

$$\begin{aligned} \mathcal{L}''_{qz}(q,u,z)(\delta q,\varphi) + \mathcal{L}''_{uz}(q,u,z)(\delta u,\varphi) = 0 \quad \forall \varphi \in V, \\ \mathcal{L}''_{qz}(q,u,z)(\tau q,\varphi) + \mathcal{L}''_{uz}(q,u,z)(\tau u,\varphi) = 0 \quad \forall \varphi \in V, \end{aligned} \tag{2.8}$$

then the following expression of the second derivative of the reduced cost functional holds:

$$\begin{aligned} j''(q)(\delta q,\tau q) = \mathcal{L}''_{qq}(q,u,z)(\delta q,\tau q) + \mathcal{L}''_{qu}(q,u,z)(\delta q,\tau u) \\ + \mathcal{L}''_{uq}(q,u,z)(\delta u,\tau q) + \mathcal{L}''_{uu}(q,u,z)(\delta u,\tau u). \end{aligned}$$

3. Discrete problem. In this section, we briefly discuss the discretization of the optimization problem (2.3). To this end, we use the Galerkin finite element method to discretize the state equation. This allows us to give a natural computable representation of the discrete gradient and Hessian as for the continuous problem. The discretization of the control space Q is kept rather abstract by choosing a finite dimensional subspace $Q_d \subset Q$. A possible concretion of this choice is given below.

To describe the finite element discretization of the state space, we consider two- or three-dimensional shape-regular meshes, see, e.g., [2]. A mesh consists of quadrilateral or hexahedral cells K, which constitute a non-overlapping cover of the computational domain $\Omega \subset \mathbb{R}^n$, $n \in \{2, 3\}$. The corresponding mesh is denoted by $\mathcal{T}_h = \{K\}$, where we define the discretization parameter h as a cellwise constant function by setting $h\big|_K = h_K$ with the diameter h_K of the cell K.

On the mesh $\mathcal{T}_h$ we construct a conforming finite element space $V_h \subset V$ in a standard way:

$$V_h = \left\{ v \in V \;\middle|\; v\big|_K \in \mathcal{Q}^1(K) \text{ for } K \in \mathcal{T}_h \right\}.$$

Here, $\mathcal{Q}^1(K)$ consists of shape functions obtained via bi- or tri-linear transformations of bi- or tri-linear polynomials defined on the reference cell $\widehat{K} = (0,1)^n$.

Then, the space discretization of the state equation (2.1) can be stated as: Find for given control $q \in Q$ a state $u_h = u_h(q) \in V_h$ such that

$$a(q, u_h)(\varphi) = (f, \varphi) \quad \forall \varphi \in V_h. \tag{3.1}$$

This discrete state equation is assumed to posses a unique solution for each $q \in Q$.

To discretize the control, we distinguish two cases: the control space is finite-dimensional, i.e., $Q = \mathbb{R}^D$ for some $D \in \mathbb{N}$, or the control space is infinite-dimensional, i.e., $Q = L^2(\Omega)$. In the first case, the control space must not be discretized. Hence, we may choose here $Q_d = Q$. In the latter case, we consider a finite element discretization of the control as for the state variable. Thus, the discrete control space is chosen as

$$Q_d = \left\{ q \in C(\bar{\Omega}) \;\middle|\; q\big|_K \in \mathcal{Q}^1(K) \text{ for } K \in \mathcal{T}_h \right\} \supset V_h.$$

The optimization problem with discrete state and control variables is then given as follows:

$$\text{Minimize } J(q_\sigma, u_\sigma) \text{ subject to (3.1) and } (q_\sigma, u_\sigma) \in Q_d \times V_h. \tag{3.2}$$

We assume the existence of a locally unique optimal solution $(\bar{q}_\sigma, \bar{u}_\sigma) \in Q_d \times V_h$ of (3.2) (see the corresponding discussion in Secton 2), where the subscript σ collects the discretization parameters h and d.

Similar to the continuous case, we introduce the discrete reduced cost functional $j_h \colon Q \to \mathbb{R}$ by

$$j_h(q) \coloneqq J(q, u_h(q))$$

and reformulate the discrete optimal control problem (3.2) as

$$\text{Minimize } j_h(q_\sigma) \text{ subject to } q_\sigma \in Q_d.$$

Similar to the previous section, the first order necessary optimality condition for (3.2) is given by

$$j_h'(\bar{q}_\sigma)(\delta q) = 0 \quad \forall \delta q \in Q_d,$$

where we assume that also j_h is twice continuously differentiable on Q.

Since the state discretization as well as the control discretization are Galerkin methods, the representation formulas for the first and second derivatives of j from the Propositions 2.2 and 2.3 can directly be transferred to the discrete level for representing the derivatives of j_h, cf. [1].

4. Hierarchical trust region algorithm. To find a local minimum of the discrete reduced cost functional j_h, the classical trust region method uses a sequence of quadratic model functions $m\colon Q_d \times Q_d \to \mathbb{R}$ given by

$$m(q, \delta q) \coloneqq j_h(q) + j_h'(q)(\delta q) + \frac{1}{2}(\delta q, H(q)\delta q)_Q$$

where $H(q)$ is a symmetric approximation to the second derivative $\nabla^2 j_h(q)$. In one step of the trust region method, one has to minimize $m(q, \delta q)$ for given $q \in Q_d$ under the constraint $\|\delta q\|_Q \leq \Delta$. Hence, one trust region step consists of the solution of the subproblem

$$\text{Minimize } m(q, \delta q) \text{ for } \delta q \in Q_d \text{ such that } \|\delta q\|_Q \leq \Delta.$$

Besides a reasonable strategy for updating the trust region radius Δ, the efficiency of this algorithm relies on the choice of the approximative Hessian $H(q)$. In this article, we propose to compute this approximation on a finite element mesh which is coarser than the current mesh $\mathcal{T}_h$ or coincides with it. The choice of this mesh is directly controlled within the proposed algorithm.

For this, we suppose that we are given a hierarchy of refined meshes $\mathcal{T}_l \coloneqq \mathcal{T}_{h_l}$, $l = 0, 1, \ldots, L$ with corresponding finite element spaces $V_l \coloneqq V_{h_l}$ and discretization parameters h_l. We assume the meshes $\mathcal{T}_l$ to become finer with increasing l and the spaces V_l to be nested, i.e., $V_l \subset V_{l+1}$.

For the mesh transfer operations $p_l^{l+1}\colon V_l \to V_{l+1}$ (prolongation) and $r_l^{l-1}\colon V_l \to V_{l-1}$ (restriction) we choose as usual in the finite element context $p_l^{l+1} = \mathrm{id}$ and r_l^{l-1} as the $L^2(\Omega)$ projection. By composition of these operators, one can define the operators

$$p_l^L \coloneqq p_{L-1}^L \circ \cdots \circ p_l^{l+1}\colon V_l \to V_L \quad \text{and} \quad r_L^l \coloneqq r_{l+1}^l \circ \cdots \circ r_L^{L-1}\colon V_L \to V_l$$

for any $l \in \{0, 1, \ldots, L-1\}$.

Using this notation, the discrete reduced functional $j_h = j_{h_L}$ will be denoted by $j_L(q) \coloneqq J(q, u_L(q))$ where $u_L(q) \coloneqq u_{h_L}(q) = u_h(q)$ is the discrete solution of the state equation in V_L. Similarly to Proposition 2.2, we write

$$j_L'(q)(\tau q) = \mathcal{L}_q'(q, u_L, z_L)(\tau q), \tag{4.1}$$

where z_L is the discrete solution of the adjoint state equation in V_L.

To define the model for the subproblem of the Hierarchical Trust Region Algorithm presented in the sequel, we define an approximation $g_l^L(q)\colon Q_d \times Q_d \to \mathbb{R}$ of the second derivative of $j_L(q)$ computed on V_l as

$$g_l^L(q)(\delta q, \tau q) \coloneqq \tilde{g}_l^L(q)(\delta q, \tau q) + \alpha(\delta q, \tau q)_Q, \tag{4.2}$$

where $\tilde{g}_l^L(q)$ will contain those parts of $g_l^L(q)$ that contribute to the approximative second derivative of $\hat{J}(u(q))$.

For the computation of $\tilde{g}_l^L(q)$, we distinguish between the two variants presented in Proposition 2.3:

(a) Let $\delta u_l \in V_l$ and $\delta z_l \in V_l$ be the solutions of the discrete versions of the linearized state (2.6) and additional adjoint (2.7) equations in V_l with some $l \in \{0, 1, \ldots, L\}$. Then, $\tilde{g}_l^L(q)(\delta q, \tau q)$ is defined as

$$\begin{aligned}\tilde{g}_l^L(q)(\delta q, \tau q) \coloneqq & -a''_{qq}(q, r_L^l u_L)(\delta q, \tau q, r_L^l z_L) \\ & + \mathcal{L}''_{uq}(q, r_L^l u_L, r_L^l z_L)(\delta u_l, \tau q) + \mathcal{L}''_{zq}(q, r_L^l u_L, r_L^l z_L)(\delta z_l, \tau q) \quad (4.3)\end{aligned}$$

(b) Let $\delta u_l, \tau u_l \in V_l$ be the solutions of the discrete version of the linearized state equation (2.8). Then, $\tilde{g}_l^L(q)(\delta q, \tau q)$ is defined as

$$\begin{aligned}\tilde{g}_l^L(q)(\delta q, \tau q) \coloneqq & -a''_{qq}(q, r_L^l u_L)(\delta q, \tau q, r_L^l z_L) + \mathcal{L}''_{qu}(q, r_L^l u_L, r_L^l z_L)(\delta q, \tau u_l) \\ & + \mathcal{L}''_{uq}(q, r_L^l u_L, r_L^l z_L)(\delta u_l, \tau q) + \mathcal{L}''_{uu}(q, r_L^l u_L, r_L^l z_L)(\delta u_l, \tau u_l). \quad (4.4)\end{aligned}$$

REMARK 4.1. *Note that in the computations of $\tilde{g}_l^L$ and consequently of g_l^L only the solutions δu_l, τu_l, and δz_l are computed on the coarser level l. The state and adjoint solutions u_L and z_L are always computed on the finest level L and are then restricted to V_l. This approach avoids the recomputation of the state and adjoint solutions on the coarser levels. Therefore the approximation $\tilde{g}_l^L(q)(\delta q, \tau q)$ of $j_L''(q)(\delta q, \tau q)$ is not equal to $j_l''(q)(\delta q, \tau q)$.*

Using this notation, we define the level-dependent model functions m_l for $l \in \{0, 1, \ldots, L\}$ by

$$m_l(q, \delta q) \coloneqq j_L(q) + j_L'(q)(\delta q) + \frac{1}{2}(\delta q, H_l(q)\delta q)_Q,$$

where the approximative Hessian H_l relies on the approximative second derivatives g_l^L defined in (4.2). For the choice of H_l we have to distinguish the cases where Q is finite- or infinite-dimensional. Because of this, we first present the general *Hierarchical Trust Region Algorithm* in Algorithm 4.1. The concrete form of H_l will be discussed in the following Subsections 4.1 and 4.2.

ALGORITHM 4.1 (Hierarchical Trust Region Algorithm).

Set $k = 0$, $l_0 = 0$, and choose $\Delta_0 > 0$, $\Delta_{max} \geq \Delta_0$, and $q_\sigma^0 \in Q_d$.
repeat
  Compute the solution $u_L \in V_L$ of the discrete state equation (3.1).
  Compute the solution $z_L \in V_L$ of the discrete adjoint equation.
  Obtain $\nabla j_L(q_\sigma^k)$ by evaluating (4.1) and the relation

$$(\nabla j_L(q_\sigma^k), \tau q^i)_Q = j_L'(q_\sigma^k)(\tau q^i)$$

  for all elements τq^i, $i = 1, 2, \ldots, \dim Q_d$ from a basis of Q_d.
  Use $r_L^{l_k}$ to obtain the restricted solutions $r_L^{l_k} u_L$ and $r_L^{l_k} z_L$ in V_{l_k}.
  Compute the solution $\delta q \in Q_d$ of the problem

$$\text{Minimize } m_{l_k}(q_\sigma^k, \delta q) \text{ for } \delta q \in Q_d \text{ such that } \|\delta q\|_Q \leq \Delta_k.$$

  Choose Δ_{k+1} such that $0 < \Delta_{k+1} \leq \Delta_{max}$, $l_{k+1} \in \{0, 1, \ldots, L\}$, and $\gamma \in \{0, 1\}$ depending on the the current iteration accordingly to Algorithm 4.2
  Set $q_\sigma^{k+1} = q_\sigma^k + \gamma \delta q$.
until $\|\nabla j_L(q_\sigma^k)\|_Q < TOL$

The update of the trust region radius Δ_k and the mesh level used to calculate the approximative Hessian H_{l_k}, as well as the choice of the parameter γ describing whether a step will be accepted or not, is done by means of Algorithm 4.2 below. It makes use of the contraction rate ρ_l which is given for $l \in \{ 0, 1, \dots, L \}$ by

$$\rho_l := \frac{j_L(q_\sigma^k + \delta q) - j_L(q_\sigma^k)}{m_l(q_\sigma^k, \delta q) - m_l(q_\sigma^k, 0)}.$$

If this rate is close to 1, the approximative model used in the subproblem works well and it is reasonable to try if the approximation performs also well on a coarser level, where the second order information is cheaper to evaluate. To test this, the rate ρ_l is compared to the rate ρ_{l-1} computed on the coarser mesh level $l-1$. Based on the result of this test, the mesh level for the next step of the trust region algorithm will be set to $l-1$ or kept at l. In the latter case, the trust region radius Δ will be enlarged for the next step.

If, in contrast, the contraction rate is close to 0 or even negative, the subproblem does not constitute a good approximation of the continuous problem and the trust region radius will be shrunken. Furthermore, depending on the comparison of the actual rate ρ_l with the contraction rate ρ_{l+1} computed on the finer mesh level $l+1$, the mesh level for the next trust region step will be set to $l+1$ or kept at l.

ALGORITHM 4.2 (Adaptation of Δ_k and l_k and choice of γ).

1: *Choose constants* $1 > \eta_1 > \eta_2 > \eta_3 > \eta_4 \geq 0$, $0 < \sigma_1 < 1$, $\sigma_2, \sigma_3 > 1$, $\delta_1, \delta_2 > 1$, *and* $0 < \delta_3, \delta_4, \delta_5 < 1$.
2: *Compute the contraction rate* ρ_{l_k}.
3: ***if*** $\rho_{l_k} > \eta_1$ ***then*** {*step is very successful*}
4: *Compute the contraction rate* $\rho_{l_k - 1}$
5: ***if*** $\rho_{l_k - 1} > \sigma_1 \rho_{l_k}$ ***then***
6: *Set* $\Delta_{k+1} = \Delta_k$, $l_{k+1} = l_k - 1$, *and* $\gamma = 1$.
7: ***else***
8: *Set* $\Delta_{k+1} = \min \{ \delta_1 \Delta_k, \Delta_{max} \}$, $l_{k+1} = l_k$, *and* $\gamma = 1$.
9: ***else if*** $\rho_{l_k} > \eta_2$ ***then*** {*step is successfully*}
10: *Set* $\Delta_{k+1} = \min \{ \delta_2 \Delta_k, \Delta_{max} \}$, $l_{k+1} = l_k$, *and* $\gamma = 1$.
11: ***else if*** $\rho_{l_k} > \eta_3$ ***then*** {*step is acceptable*}
12: *Set* $\Delta_{k+1} = \Delta_k$, $l_{k+1} = l_k$, *and* $\gamma = 1$.
13: ***else if*** $\rho_{l_k} > \eta_4$ ***then*** {*step is bad*}
14: *Compute the contraction rate* $\rho_{l_k + 1}$.
15: ***if*** $\rho_{l_k + 1} > \sigma_2 \rho_{l_k}$ ***then***
16: *Set* $\Delta_{k+1} = \Delta_k$, $l_{k+1} = l_k + 1$, *and* $\gamma = 1$.
17: ***else***
18: *Set* $\Delta_{k+1} = \delta_3 \Delta_k$, $l_{k+1} = l_k$, *and* $\gamma = 1$.
19: ***else*** {*step is very bad*}
20: *Compute the contraction rate* $\rho_{l_k + 1}$.
21: ***if*** $\rho_{l_k + 1} > \sigma_3 \rho_{l_k}$ ***then***
22: *Set* $\Delta_{k+1} = \delta_4 \Delta_k$, $l_{k+1} = l_k + 1$, *and* $\gamma = 0$.
23: ***else***
24: *Set* $\Delta_{k+1} = \delta_5 \Delta_k$, $l_{k+1} = l_k$, *and* $\gamma = 0$

The minimization of $m_{l_k}(q_\sigma^k, \delta q)$ in line 7 of Algorithm 4.1 is done by means of the Steihaug conjugate gradient method (CG method), cf. [11]. There, only the action of $H_{l_k}(q_\sigma^k)$ on a given element δq is needed. For the computation of $H_{l_k}(q_\sigma^k)\delta q$, we have to distinguish the two considered cases for the control space, i.e., finite- or

infinite-dimensional controls.

4.1. Finite-dimensional control space. As discussed in Section 3, in this case, we have $Q_d = Q = \mathbb{R}^D$. Consequently, the (discrete) control variable is independent of the current mesh. Referring to the representation formula of the second derivatives j_l'' from (4.2) in combination with (4.3) and (4.4), one has to solve the linearized state and possibly the additional adjoint equations in V_l to compute the desired approximation $H_l(q)\delta q$.

Depending on the dimension D of the control space, it may be favorable to assemble the whole Hessian matrix instead of computing just matrix-vector products of it. In this case, the representation (4.4) avoiding the computation of δz_l is more efficient, see [1] for details on this. The concrete procedure for assembling the whole approximative Hessian $H_l(q)$ is given in the following algorithm:

ALGORITHM 4.3 (Computation of $H_l(q)$).

1: *Calculate $\delta u_l^i \in V_l$ by solving the linearized state equation for all elements δq^i, $i = 1, 2, \ldots, D$ from a basis of $Q_d = Q$.*
2: *Obtain $\tilde{g}_l^L(q)(\delta q^i, \delta q^j)$ by evaluating the expression from (4.4) for all elements δq^i, $i = 1, 2, \ldots, D$ from a basis of $Q_d = Q$.*
3: *Obtain $\tilde{G}_l^L(q) \in Q_d$ by the relation*

$$(\delta q^j, \tilde{G}_l^L(q)\delta q^i)_Q = \tilde{g}_l^L(q)(\delta q^i, \delta q^j), \quad i, j = 1, 2, \ldots, D.$$

4: *Set $H_l(q) \coloneqq \tilde{G}_l^L(q) + \alpha I$.*

If, in contrast to the discussion above, the dimension D of the control space is rather large, it is more efficient to only compute matrix-vector products of the (approximative) Hessian and a given vector δq. The procedure in this case is given in the following algorithm:

ALGORITHM 4.4 (Computation of $H_l(q)\delta q$).

1: *Calculate $\delta u_l \in V_l$ by solving the linearized state equation.*
2: *Calculate $\delta z_l \in V_l$ by solving the additional adjoint equation.*
3: *Obtain $\tilde{g}_l^L(q)(\delta q, \tau q^i)$ by evaluating the expression from (4.3) for all elements τq^i, $i = 1, 2, \ldots, D$ from a basis of $Q_d = Q$.*
4: *Obtain $\tilde{G}_l^L(q)\delta q \in Q_d$ by the relation*

$$(\tau q^i, \tilde{G}_l^L(q)\delta q)_Q = \tilde{g}_l^L(q)(\delta q, \tau q^i), \quad i = 1, 2, \ldots, D.$$

5: *Set $H_l(q)\delta q \coloneqq \tilde{G}_l^L(q)\delta q + \alpha\delta q$.*

4.2. Infinite-dimensional control space. If the control space is infinite-dimensional, we assume that it is discretized on the same mesh as the state, see Section 3. Hence, when computing the approximation H_l on mesh $\mathcal{T}_l$, the control has to be transferred between $Q_L \coloneqq Q_{d_L} = Q_d$ on $\mathcal{T}_L$ and the corresponding space Q_l defined on $\mathcal{T}_l$. It is clear, that by construction the transfer operators p_l^L and r_L^l can also be applied to elements of the hierarchy Q_l with $l \in \{\, 0, 1, \ldots, L \,\}$.

Since for infinite-dimensional continuous controls, the discrete control space is rather high-dimensional, the assembling of the whole Hessian is prohibitive. Hence, we only present the computation of the approximative matrix-vector products of the Hessian with a given vector δq in the following algorithm:

ALGORITHM 4.5 (Computation of $H_l(q)\delta q$).

1: *Restrict the control q and the direction δq from Q_L to Q_l also by application of r_L^l and obtain $r_L^l q$ and $r_L^l \delta q$.*

2: *Calculate* $\delta u_l \in V_l$ *by solving the linearized state equation.*
3: *Calculate* $\delta z_l \in V_l$ *by solving the additional adjoint equation.*
4: *Obtain* $\tilde{g}_l^L(r_L^l q)(r_L^l \delta q, \tau q^i)$ *by evaluating the expression from* (4.3) *for all elements* τq^i, $i = 1, 2, \ldots, \dim Q_l$ *from a basis of* Q_l.
5: *Obtain* $\tilde{G}_l^L(r_L^l q) r_L^l \delta q \in Q_l$ *by the relation*

$$(\tau q^i, \tilde{G}_l^L(r_L^l q) r_L^l \delta q)_Q = \tilde{g}_l^L(r_L^l q)(r_L^l \delta q, \tau q^i), \quad i = 1, 2, \ldots, \dim Q_l$$

6: *Define the corresponding element* $H_l(q)\delta q \in Q_L$ *by application of* p_l^L, *i.e., set*

$$H_l(q)\delta q \coloneqq p_l^L \tilde{G}_l^L(r_L^l q) r_L^l \delta q + \alpha \delta q.$$

5. Convergence. After presenting the Hierarchical Trust Region Algorithm in Section 4, we now show that it converges to an optimal solution. This is done by adapting the convergence proof for the usual trust region algorithm from [3, Section 6.4]. The essential requirement for doing so consists in showing the boundedness of the approximative Hessian $H_l(q)$ entering the model function $m_l(q, \delta q)$.

In order to do this, we assume that the error between the continuous Hessian and the one on the finite element space Q_L is bounded.

ASSUMPTION 1. *Let the inequality*

$$|(\delta q, \nabla^2 j_L(q)\tau q)_Q - (\delta q, \nabla^2 j(q)\tau q)_Q| \leq K_1 \|\delta q\|_Q \|\tau q\|_Q$$

hold for all q, $\delta q, \tau q \in Q_L$, *where* K_1 *only depends on the level* L *and continuously on* q.

This assumption is usually fulfilled, as it signifies in other words, that the discretization error caused by the finite element approximation is small, so that the discrete Hessian, obtained by solving discrete versions of (2.6) and (2.7) or (2.8), is still close to the continuous Hessian.

Additionally we assume that also the error between the Hessian on Q_L and the Hessian on Q_l is bounded.

ASSUMPTION 2. *Let the inequality*

$$|(\delta q, p_l^L \tilde{G}_l^L(q) r_L^l \tau q)_Q - (\delta q, \tilde{G}_L^L(q)\tau q)_Q| \leq K_2 \|\delta q\|_Q \|\tau q\|_Q$$

hold for all $q \in Q_l$, $\delta q, \tau q \in Q_L$ *and all levels* $l \in \{0, 1, \ldots, L\}$, *where* K_2 *depends on* L *and continuously on* q, *but not on* l.

REMARK 5.1. *Note that* $\tilde{G}_L^L(q)$ *is the part of the Hessian of* $j_L(q)$ *that does not contains the derivatives of the regularization term, i.e.,* $\nabla^2 j_L(q) = \tilde{G}_L^L(q) + \alpha I$.

Also this assumption is realistic, since it means that for a given control q on the coarser level l, the error between computing an approximative Hessian according to the Algorithms 4.4 or 4.3 (for finite-dimensional control) or Algorithm 4.5 (for infinite-dimensional control) and computing its equivalent on the finer level L is bounded. This can usually be assured by the nested finite element spaces and the prolongation and restriction operators. Following this argumentation, K_2 should still depend on l, more precisely on the mesh size of Q_l, but this value can be estimated from above by $\max_{0 \leq l \leq L} h_{\max}^l$, where $h_{\max}^l$ denotes the maximal cell diameter of Q_l. Hence, K_2 is indeed a constant, which does not depend on l.

THEOREM 5.2. *Let the Assumptions 1 and 2 hold and let* j *and* j_L *be twice continuously differentiable. Moreover, let the sequence* $\{q_k\}$ *be produced by Algorithm 4.1 and let* q^* *be an accumulation point of* $\{q_k\}$. *Let furthermore* $m_{l_k}(q, \cdot)$ *be*

twice continuously differentiable for all $q \in Q_L$ and all iterations k. Let $\{ q_m \} \subset \{ q_k \}$ be a subsequence with $q_m \to q^* \in Q_L$ in Q_L for $m \to \infty$ and let $B_L^m := \{ q \in Q_L \mid \|q - q_m\|_Q \leq \Delta_m \}$ be the ball with center q_m and trust region radius Δ_m.

Then, the estimates

$$|(\delta q, \nabla^2 j_L(q)\delta q)_Q| \leq C_1 \|\delta q\|_Q \|\tau q\|_Q$$

and

$$|(\delta q, H_l(q)\delta q)_Q| \leq C_2 \|\delta q\|_Q \|\tau q\|_Q$$

hold for all $q \in B_L^m$, $\delta q, \tau q \in Q_L$, where C_1 and C_2 only depend on the finest level L, but neither on the coarser level l, on which $H_l(q)$ is calculated nor on the iteration index k.

Proof. Let $q \in B_L^m \subseteq Q_L$. Obviously there exists a constant M such that $\|q_m\|_Q \leq M$ for all $m \in \mathbb{N}$ and therefore

$$q \in \mathcal{B}_L := \{ q \in Q_L \mid \|q\|_Q \leq M + \Delta_{\max} \}. \tag{5.1}$$

$\mathcal{B}_L$ is a bounded, closed subset of the finite-dimensional space Q_L and therewith compact. The reduced functional j is twice continuously differentiable, so $\nabla^2 j$ considered on $\mathcal{B}_L$ is a continuous function on a compact set. With (5.1), we can conclude that there exists a constant $\beta_1 > 0$ such that

$$|(\delta q, \nabla^2 j(q)\tau q)_Q| \leq \beta_1 \|\delta q\|_Q \|\tau q\|_Q \quad \forall \delta q, \tau q \in Q_L, \tag{5.2}$$

where β_1 does not depend on k. By the same argumentation we can deduce from Assumption 1, that there exists a constant $N_1 > 0$, such that

$$|(\delta q, \nabla^2 j_L(q)\tau q)_Q - (\delta q, \nabla^2 j(q)\tau q)_Q| \leq N_1 \|\delta q\|_Q \|\tau q\|_Q$$

for all $\delta q, \tau q \in Q_L$, where N_1 only depends on the level L, but not on k. With (5.2), we get

$$\begin{aligned} |(\delta q, \nabla^2 j_L(q)\tau q)_Q| &\leq |(\delta q, \nabla^2 j_L(q)\tau q)_Q - (\delta q, \nabla^2 j(q)\tau q)_Q| + |(\delta q, \nabla^2 j(q)\tau q)_Q| \\ &\leq (N_1 + \beta_1) \|\delta q\|_Q \|\tau q\|_Q, \end{aligned}$$

which shows the boundedness of the discrete Hessian.

For showing the second inequality, we distinguish again the following two cases: finite-dimensional control space and infinite-dimensional control space. We begin with the latter case. There, we have for $l \in \{ 0, 1, \ldots, L \}$ by Algorithm 4.5 that

$$H_l(q)\delta q = p_l^L \tilde{G}_l^L(r_L^l q) r_L^l \delta q + \alpha \delta q.$$

Because of the boundedness of the restriction map (the L^2 projection) r_L^l, i.e.,

$$\|r_L^l \delta q\|_Q \leq \gamma \|\delta q\|_Q \quad \forall \delta q \in Q_L$$

with a constant $\gamma > 0$, there exist constants $\beta_2 > 0$, $D_1 > 0$ and $D_2 > 0$ independent from k such that

$$|(\delta q, \nabla^2 j(r_L^l q)\tau q)_Q| \leq \beta_2 \|\delta q\|_Q \|\tau q\|_Q \quad \forall \delta q, \tau q \in Q_L, \tag{5.3}$$

$$|(\delta q, \nabla^2 j_L(r_L^l q)\tau q)_Q - (\delta q, \nabla^2 j(r_L^l q)\tau q)_Q| \leq D_1 \|\delta q\|_Q \|\tau q\|_Q \quad \forall \delta q, \tau q \in Q_L, \tag{5.4}$$

$$|(\delta q, p_l^L \tilde{G}_l^L(r_L^l q) r_L^l \tau q)_Q - (\delta q, \tilde{G}_L^L(r_L^l q)\tau q)_Q| \leq D_2 \|\delta q\|_Q \|\tau q\|_Q \quad \forall \delta q, \tau q \in Q_L, \tag{5.5}$$

where we have used the Assumptions 1 and 2, the fact that j and m_{l_k} are twice continuously differentiable, and (5.1) with the same reasoning as above.

Furthermore, we have by (4.2) that for all $q, \delta q, \tau q \in Q_L$

$$(\delta q, \nabla^2 j_L(q)\tau q)_Q = (\delta q, \tilde{G}_L^L(q)\tau q)_Q + \alpha(\delta q, \tau q)_Q.$$

Using this relation, (5.3), (5.4), and (5.5), the boundedness of $H_l(q)$ can then be shown as follows:

$$\begin{aligned}
|(\delta q, H_l(q)\tau q)_Q| &= |(\delta q, p_l^L \tilde{G}_l^L(r_L^l q) r_L^l \tau q)_Q + \alpha(\delta q, \tau q)_Q| \\
&\leq |(\delta q, p_l^L \tilde{G}_l^L(r_L^l q) r_L^l \tau q)_Q - (\delta q, \tilde{G}_L^L(r_L^l q)\tau q)_Q| \\
&\quad + |(\delta q, \nabla^2 j_L(r_L^l q)\tau q)_Q| \\
&\leq D_2\gamma \|\delta q\|_Q \|\tau q\|_Q + |(\delta q, \nabla^2 j_L(r_L^l q)\tau q)_Q - (\delta q, \nabla^2 j(r_L^l q)\tau q)_Q| \\
&\quad + |(\delta q, \nabla^2 j(r_L^l q)\tau q)_Q| \\
&\leq (D_2\gamma + D_1 + \beta_2)\|\delta q\|_Q \|\tau q\|_Q.
\end{aligned}$$

Considering a finite-dimensional control space (cf. the Algorithms 4.3 and 4.4), showing the boundedness of $H_l(q)$ can be done by simplifying the proof above. That means, we set $Q_l = Q_L$ and $r_L^l = p_l^L = \mathrm{id}$, but note that still $V_l \neq V_L$ and consequently $\tilde{G}_L^L \neq \tilde{G}_l^L$. □

With the shown boundedness of the approximative discrete Hessian we prove the following convergence result:

THEOREM 5.3. *Let the requirements of Theorem 5.2 be fulfilled. Then Algorithm 4.1 terminates with a stationary point or generates a sequence whose accumulation points are stationary points of j_L.*

Proof. The convergence result can be shown by means of the convergence proofs for the usual trust region algorithm (cf. [3, Theorem 6.4.6, p.137]), since almost all properties of the problem, the model, and the algorithm are also given in the context of this paper:

1. j_L is twice continuously differentiable on Q_L.
2. j_L is bounded below.
3. $\nabla^2 j_L(q)$ is bounded for all $q \in B_L^m$ and all m (see Theorem 5.2).
4. For all k the model $m_{l_k}(q, \cdot)$ is twice differentiable on Q_L for $q \in Q_L$ fixed.
5. $m_{l_k}(q_k, 0) = j_L(q_k)$ for all k.
6. $\nabla m_{l_k}(q_k, 0) = \nabla j_L(q_k)$ for all k.
7. $H_l(q)$ is bounded for all $q \in B_L^m$ and all m (see Theorem 5.2).
8. $m_{l_k}(q_k, 0) - m_{l_k}(q_k, \delta q_k) \geq c\|\nabla j_L(q_k)\|_Q \min\left\{\frac{\|\nabla j_L(q_k)\|_Q}{\|H_l(q_k)\|_Q}, \Delta_k\right\}$ for a constant $c > 0$.

Bullet point 3. is in [3] required to hold on the whole space Q_L, but as mentioned in assumption AF.3 in [3, p. 121], it suffices to require the boundedness of the continuous Hessian on B_L^m, as one can see also in [3, proof of Theorem 6.4.1].

Bullet point 8. follows from the fact that the Steihaug conjugate gradient method for generating an approximate solution of the trust region subproblem ensures a certain descent, due to the Cauchy point analysis (cf. [3, remark on p.205 or section 6.7]).

In contrast to [3], we restrict our consideration to a subsequence instead of the whole sequence of iterates. This, however, does not constitute a restriction to the convergence results, since we can proceed as in [5, proof of Lemma 14.24, p.290].

There, the convergence proof consists in assuming that there is an accumulation point, which is no stationary point. In order to raise a contradiction, they first show $\lim_{k\to\infty} j_L(q_k) = j_L(q_*)$. Indeed, this convergence also holds in the context of this paper, as we will show in the following.

If $\|\nabla j_L(q_k)\|_Q = 0$, Algorithm 4.1 terminates (see line 10.) and q_k is a stationary point of j_L. So in the case that the Algorithm does not terminate, there holds $\|\nabla j_L(q_k)\|_Q \neq 0$ for all $k \in \mathbb{N}$. As we assumed $\|\nabla j_L(q^*)\|_Q \neq 0$ and as $\|\nabla j_L(.)\|_Q$ is a continuous function, there exists a constant $\varepsilon > 0$, such that $\|\nabla j_L(q_k)\|_Q \geq \varepsilon$ for all $k \in \mathbb{N}$. Besides, there are infinitely many successfull iterations (see [3, Theorem 6.4.4, p.136]). Then, according to Algorithm 4.2 the inequality

$$\begin{aligned} j_L(q_k) - j_L(q_{k+1}) &> \eta_4(m_{l_k}(q_k, 0) - m_{l_k}(q_k, \delta q_k)) \\ &\geq c\|\nabla j_L(q_k)\|_Q \min\left\{ \frac{\|\nabla j_L(q_k)\|_Q}{\|H_l(q_k)\|_Q}, \Delta_k \right\} \\ &\geq c\varepsilon \min\left\{ \frac{\varepsilon}{\|H_l(q_k)\|_Q}, \Delta_k \right\} \\ &> 0, \end{aligned}$$

holds for all successfull iterations, which yields to the convergence of the whole sequence $\lim_{k\to\infty} j_L(q_k) = j_L(q_*)$.

With this convergence result and the boundedness of the Hessian on the subsequence $\{ q_m \}$, 8. leads to $\lim_{m\to\infty} \Delta_m = 0$ (see [5]), which is a contradiction to [5, Lemma 14.22]. □

6. Numerical results. For illustrating the performance of the presented method, we apply it so several examples, which, as in the Sections 4.1 and 4.2, we divide into two categories: finite-dimensional and infinite-dimensional control space. We consider only examples with nonlinear state equation as in the linear case the Newton trust region method in general only needs one step for finding the optimal solution (if the trust region radius is large enough) so that the proposed Hierarchical Trust Region Algorithm does barely have a chance to lead to a considerable time reduction.

For all subsequent examples, we consider the unit square $\Omega = (0, 1)^2$ as domain for the partial differential equation and choose the coarsest mesh $\mathcal{T}_0$ to consist of four quadratic cells of the same size. The constants in Algorithm 4.2 are chosen as $\eta_1 = 0.9625$, $\eta_2 = 0.7$, $\eta_3 = 0.5$, $\eta_4 = 0$, $\sigma_1 = 0.8$, $\sigma_2 = \sigma_3 = 1.2$, $\delta_1 = 12.25$, $\delta_2 = 3.5$, $\delta_3 = \delta_4 = 0.75$, $\delta_5 = 0.5$.

6.1. Finite-dimensional control space. In the following two examples, we consider optimal control problems with finite-dimensional control space, in particular $Q = \mathbb{R}^2$, and a nonlinear state equation. We will see in both examples that the presented Hierarchical Trust Region Algorithm leads to a considerable reduction of computation time.

6.1.1. Example 1. In this example, we aim at the minimization of the cost functional

$$J(q, u) = \|u - 10\|^2_{L^2(\Omega)}$$

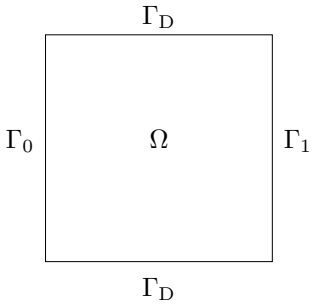

FIG. 6.1. *Example 1: Computational domain* Ω

TABLE 6.1
Example 1: Reduction of the computation time without assembling the Hessian matrix

Level L						
4	5	6	7	8	9	10
46%	47%	54%	53%	48%	57%	60%

subject to the state equation with Neumann boundary control

$$\begin{aligned} -\Delta u + u^2 &= 1 && \text{in } \Omega, \\ \partial_n u &= q_0 && \text{on } \Gamma_0, \\ \partial_n u &= q_1 && \text{on } \Gamma_1, \\ u &= 0 && \text{on } \Gamma_D \end{aligned}$$

and

$$(q,u) \in Q \times V = \mathbb{R}^2 \times \left\{ v \in H^1(\Omega) \;\middle|\; v\big|_{\Gamma_D} = 0 \right\},$$

where the domain Ω with boundary $\partial\Omega = \Gamma_0 \cup \Gamma_1 \cup \Gamma_D$ is given as in Figure 6.1.

The Tables 6.1 and 6.2 show the relative reduction of computation time in dependence of the finest level L achieved by the Hierarchical Trust Region Algorithm (Algorithm 4.1) compared to the Newton trust region algorithm without level adaptation.

For Table 6.2, the Hessian has been assembled explicitly following Algorithm 4.3. For Table 6.1 the Hessian was not assembled but only matrix-vector products were computed according to Algorithm 4.4. In the first case we get a time reduction up to 49% and in the second case even up to 60%.

This remarkable reduction is achieved, although the Hierarchical Trust Region Algorithm needs more trust region and CG steps, as it is shown in Table 6.3. This can be explained by considering the loss of accuracy of the approximative Hessian which leads to the need of more trust region and implicitly more CG steps on the one hand, but to cheaper CG steps on the other hand.

6.1.2. Example 2. In this section, we consider a parameter estimation problem with Tikhonov regularization which consists in the minimization of the functional

$$J(q,u) = \frac{1}{2}\sum_{k=0}^{4} (u(p_i) - \hat{u}_i)^2 + \frac{\alpha}{2}\|q\|_Q^2$$

TABLE 6.2
Example 1: Reduction of the computation time with assembling the Hessian matrix

Level L						
4	5	6	7	8	9	10
37%	36%	36%	35%	21%	49%	37%

TABLE 6.3
Example 1: Number of trust region and CG steps for Newton trust region (NTR) and Hierarchical Trust Region (HTR)

# trust region steps		# CG steps	
HTR	NTR	HTR	NTR
7	5	11	9

with artificial measurements

$$\hat{u} = 10^{-2}(8, 9, 9, 5, 5)^T$$

subject to the state equation with homogeneous Dirichlet boundary conditions

$$\begin{aligned} -\Delta u + q_0 \frac{\partial u}{\partial x_1} + q_1 \frac{\partial u}{\partial x_2} &= 2 \qquad \text{in } \Omega, \\ u &= 0 \qquad \text{on } \partial\Omega, \end{aligned}$$

and

$$(q, u) \in Q \times V = \mathbb{R}^2 \times H_0^1(\Omega),$$

where p_1, p_2, p_3, p_4 are points inside the computational domain, which are located as shown in Figure 6.2.

In Table 6.4 and Table 6.5, we can see an improvement concerning the needed computation time between 14% and 66%, although (as in Example 1) the number of iteration steps is higher than without level adaptation, as we can read off from Table 6.6.

The intensity of time reduction mainly depends on the size of the regularization parameter α: the stronger the regularization, the greater the time reduction. This is due to the fact that for a large α the second derivative of the regularization term $\frac{\alpha}{2}\|q\|_Q^2$ is weighted more. Since this part is always calculated exactly the error made by calculating the other part of the Hessian on a coarser level does barely take an effect. But even with a small regularization parameter $\alpha = 10^{-3}$, the algorithm still provides a time reduction of up to 20%.

Apart from that, a look at Table 6.4 and Table 6.5 reveals that the time reduction is larger for the case when not assembling the entire Hessian (cf. Algorithm 4.4). The reason for this issue probably lies in the fact that for a low-dimensional control space, e.g., as here $Q = \mathbb{R}^2$, the Newton trust region algorithm is much cheaper in the case of assembling the entire Hessian compared to the case of not assembling the Hessian. With level adaptation the difference between these two cases is not that large, since the computation of the Hessian or the Hessian-vector product respectively, is performed on a coarser level, which means it is cheap anyway.

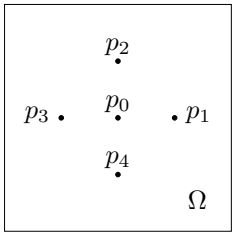

FIG. 6.2. *Example 2: Computational domain Ω and measurement points p_i*

TABLE 6.4
Example 2: Reduction of the computation time without assembling the Hessian matrix

α	Level L			
	6	7	8	9
10^0	59%	66%	60%	58%
10^{-1}	56%	53%	54%	52%
10^{-2}	50%	53%	50%	51%
10^{-3}	43%	45%	45%	42%

6.2. Infinite-dimensional control space. We also tested the Hierarchical Trust Region Method on examples with an infinite-dimensional control space. In more detail, we will restrict ourselves to the choice $Q = L^2(\Omega)$ with distributed control. In contrast to the case of finite-dimensional control, we need additional calculations for the transfer between the levels (cf. Section 4.2). However, because of the cheap calculation steps for the Hessian, we still get a respectable reduction of computation time in the following examples.

6.2.1. Example 3. In this example, we focus on the tracking-type cost functional

$$J(q,u) = \frac{1}{2}\|u - 10\|^2_{L^2(\Omega)} + \frac{\alpha}{2}\|q\|^2_{L^2(\Omega)}$$

subject to the nonlinear state equation with homogeneous Dirichlet boundary conditions

$$\begin{aligned} -\Delta u + u^3 &= q \qquad \text{in } \Omega, \\ u &= 0 \qquad \text{on } \partial\Omega, \end{aligned}$$

and

$$(q,u) \in Q \times V = L^2(\Omega) \times H^1_0(\Omega).$$

In the Tables 6.7 and 6.8 one can see a behavior which is similar to the one in the finite-dimensional case. A small difference consists in the higher number of trust region as well as CG Steps that are needed in comparison to the Newton trust region algorithm and the associated slightly smaller computation time reduction. But even for the double number of trust region and CG steps than in the Newton trust region algorithm (cf. Table 6.8), we still achieve a time reduction of at least 22%.

TABLE 6.5
Example 2: Reduction of the computation time with assembling the Hessian matrix

α	Level L			
	6	7	8	9
10^0	57%	58%	57%	52%
10^{-1}	51%	45%	46%	43%
10^{-2}	40%	40%	38%	40%
10^{-3}	20%	14%	16%	17%

TABLE 6.6
Example 2: Number of trust region and CG steps for Newton trust region (NTR) and Hierarchical Trust Region (HTR)

α	# trust region steps		# CG steps	
	HTR	NTR	HTR	NTR
10^0	1	2	1	2
10^{-1}	2	2	2	2
10^{-2}	2	2	2	2
10^{-3}	5	3	5	3

6.2.2. Example 4. In this example, we focus on the tracking-type cost functional

$$J(q,u) = \frac{1}{2}\|u - 10\|^2_{L^2(\Omega)} + \frac{\alpha}{2}\|q\|^2_{L^2(\Omega)}$$

subject to the bilinear state equation also with homogeneous Dirichlet boundary conditions

$$\begin{aligned} -\Delta u + qu &= 1 \qquad \text{in } \Omega, \\ u &= 0 \qquad \text{on } \partial\Omega, \\ q &\geq \varepsilon \end{aligned}$$

and

$$(q,u) \in Q \times V = L^2(\Omega) \times H^1_0(\Omega)$$

for a given $\varepsilon > 0$.

In the following tests we chose ε small enough and the starting value for q appropriately so that the control constraint is not active anywhere in Ω.

Concerning the relation between the regularization parameter α and the achieved reduction of computation time, Table 6.9 (as well as Table 6.7) confirms the results from the finite-dimensional examples also for an infinite-dimensional control space. Here it also holds: the larger alpha, the more time reduction. This can be explained by the same argumentation as in Section 6.2.1. In this example, the reduction of computation time is even larger than in Example 3: Also for a relatively small regularization parameter, e.g., $\alpha = 10^{-3}$ the improvement is still about 20%, despite of the almost triple number of trust region and CG steps, which can be seen in Table 6.10.

TABLE 6.7
Example 3: Reduction of the computation time

α	Level L					
	4	5	6	7	8	9
10^0	51%	52%	53%	52%	51%	51%
10^{-1}	22%	23%	22%	23%	25%	23%

TABLE 6.8
Example 3: Number of trust region and CG steps for Newton trust region (NTR) and Hierarchical Trust Region (HTR)

α	# trust region steps		# CG steps	
	HTR	NTR	HTR	NTR
10^0	2	2	4	4
10^{-1}	4	2	11	4

Summarizing the results of the subsections 6.1 and 6.2, it can be recorded that Algorithm 4.1 indeed leads to a considerable reduction of computation time for the tested finite-dimensional as well as infinite-dimensional examples, although in both cases, the number of iterations (trust region and CG steps) is increased in comparison to the Newton trust region method (due to loss of accuracy).

If there is a Tikhonov regularization, the amount of reduction mainly depends on the regularization parameter: the reduction is larger, the stronger the regularization is.

For finite-dimensional examples, it also depends on whether one computes the entire Hessian or matrix-vector products of the Hessian and a given vector, insofar as the faster the Newton trust region algorithm is, the less relative reduction of computation time by the Hierarchical Trust Region Algorithm is possible.

For infinite-dimensional examples, the performance is similar. One should keep in mind that in this setting, there are additional calculations required for the transfer between the levels. However, these calculations together with the cheap calculations for the approximate Hessian on a coarser level seem to be not as expensive as saving the additional level transfers but computing the exact Hessian on the finest level, such that, overall, a remarkable computation time reduction is achieved.

Acknowledgments. We would like to thank Ekkehard Sachs for fruitful discussions on trust region methods for PDE-constrained optimization. The first author gratefully acknowledges the support by the DFG project "Adaptive Discretization Methods for the Regularization of Inverse Problems" and by the TUM Graduate School's Thematic Graduate Center / Faculty Graduate Center at Technische Universität München, Germany.

REFERENCES

[1] R. BECKER, D. MEIDNER, AND B. VEXLER, *Efficient numerical solution of parabolic optimization problems by finite element methods*, Optim. Methods Softw., 22 (2007), pp. 813–833.
[2] P. G. CIARLET, *The Finite Element Method for Elliptic Problems*, vol. 40 of Classics Appl. Math., SIAM, Philadelphia, 2002.

TABLE 6.9
Example 4: Reduction of the computation time

α	Level L					
	4	5	6	7	8	9
10^0	55%	50%	52%	50%	55%	50%
10^{-1}	51%	51%	49%	48%	54%	55%
10^{-2}	33%	35%	35%	31%	40%	39%
10^{-3}	16%	19%	19%	16%	26%	20%

TABLE 6.10
Example 4: Number of trust region and CG steps for Newton trust region (NTR) and Hierarchical Trust Region (HTR)

α	# trust region steps		# CG steps	
	HTR	NTR	HTR	NTR
10^0	2	2	4	4
10^{-1}	2	2	4	4
10^{-2}	3	2	7	4
10^{-3}	6	3	18	7

[3] A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *Trust-Region Methods*, vol. 1 of MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[4] A. V. FURSIKOV, *Optimal Control of Distributed Systems: Theory and Applications*, vol. 187 of Transl. Math. Monogr., AMS, Providence, 1999.

[5] C. GEIGER AND C. KANZOW, *Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben*, Springer, 1999.

[6] M. HEINKENSCHLOSS AND D. RIDZAL, *An inexact trust-region SQP method with applications to pde-constrained optimization*, in Numerical Mathematics and Advanced Applications, K. Kunisch, G. Of, and O. Steinbach, eds., Berlin, 2008, Springer, pp. 613–620. Proceeding of ENUMATH 2007.

[7] M. HERTY AND G. THMMES, *A two-level trust-region method for optimal control problems with radiative transfer*, Adv. Model. Optim., 8 (2006), pp. 187–207.

[8] C. T. KELLEY AND E. W. SACHS, *A trust region method for parabolic boundary control problems*, SIAM J. Optim., 9 (1999), pp. 1064–1081.

[9] J.-L. LIONS, *Optimal Control of Systems Governed by Partial Differential Equations*, vol. 170 of Grundlehren Math. Wiss., Springer-Verlag, Berlin, 1971.

[10] J. NOCEDAL AND S. WRIGHT, *Numerical Optimization*, Springer Series in Operations Research, Springer, New York, 1999.

[11] T. STEIHAUG, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.

[12] F. TRÖLTZSCH, *Optimale Steuerung partieller Differentialgleichungen*, Friedr. Vieweg & Sohn Verlag, Wiesbaden, 2 ed., 2009.

[13] M. ULBRICH, S. ULBRICH, AND M. HEINKENSCHLOSS, *Global convergence of trust-region interior-point algorithms for infinite-dimensional nonconvex minimization subject to pointwise bounds*, SIAM J. Control Optim., 37 (1999), pp. 731–764.