SIAM J. OPTIM.
Vol. 24, No. 3, pp. 1507–1541

A MATRIX-FREE TRUST-REGION SQP METHOD FOR EQUALITY CONSTRAINED OPTIMIZATION*

MATTHIAS HEINKENSCHLOSS† AND DENIS RIDZAL‡

Abstract. We develop and analyze a trust-region sequential quadratic programming (SQP) method for the solution of smooth equality constrained optimization problems, which allows the inexact and hence iterative solution of linear systems. Iterative solution of linear systems is important in large-scale applications, such as optimization problems with partial differential equation constraints, where direct solves are either too expensive or not applicable. Our trust-region SQP algorithm is based on a composite-step approach that decouples the step into a quasi-normal and a tangential step. The algorithm includes critical modifications of substep computations needed to cope with the inexact solution of linear systems. The global convergence of our algorithm is guaranteed under rather general conditions on the substeps. We propose algorithms to compute the substeps and prove that these algorithms satisfy global convergence conditions. All components of the resulting algorithm are specified in such a way that they can be directly implemented. Numerical results indicate that our algorithm converges even for very coarse linear system solves.

Key words. sequential quadratic programming, trust-region, large-scale optimization, matrix free, inexact linear system solvers, PDE-constrained optimization, Krylov subspace methods

AMS subject classifications. 90C55, 49M25, 49K20, 65K05, 65J10

DOI. 10.1137/130921738

1. Introduction.

Sequential quadratic programming (SQP) methods are used successfully for the solution of smooth nonlinear programming problems. Each iteration of an SQP method requires the solution of linear systems that involve the constraint Jacobian or its transpose. Most convergence theories for SQP methods and most SQP implementations require that these linear systems be solved exactly. For many large-scale problems, especially problems with partial differential equation (PDE) constraints, this is not possible. In many such applications, constraint Jacobians are not formed explicitly and only their action and the action of their transpose on a vector are available. Even if these matrices are formed explicitly the solution of the linear systems using direct linear algebra is prohibitively expensive. In these cases, iterative linear system solvers must be applied. As a consequence, all linear systems are solved inexactly and it is crucial to account for this inexactness in the design and in the convergence analysis of SQP methods.

In this paper we introduce a general trust-region SQP algorithm for nonconvex equality constrained optimization that incorporates inexact linear system solves, we prove its global convergence, we propose subalgorithms to fully define its implementation, and we construct easily implementable stopping criteria for iterative

*Received by the editors May 21, 2013; accepted for publication (in revised form) July 7, 2014; published electronically September 11, 2014.
http://www.siam.org/journals/siopt/24-3/92173.html

†Department of Computational and Applied Mathematics, MS-134, Rice University, Houston, TX 77005-1892 (heinken@rice.edu). This author's research was supported in part by NSF grants ACI-0121360 and DMS-0511624, DMS-0915238.

‡Optimization and Uncertainty Quantification, MS-1320, Sandia National Laboratories, Albuquerque, NM 87185-1320 (dridzal@sandia.gov). Sandia National Laboratories is a multiprogram laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. This author's research was supported by the DOE-SC ASCR Office, through the John von Neumann Fellowship, and by the DOE-NNSA ASC program.

linear solvers. Our algorithm is based on the composite-step trust-region SQP framework presented by Dennis, El-Alem, and Maciel [8], but contains several algorithmic modifications necessary to ensure its convergence in the presence of inexact linear system solves. The algorithmic modifications are derived from Heinkenschloss and Vicente [16]. Like [16] we only tighten the linear system stopping tolerances as necessary, allowing coarse and relatively inexpensive linear system solves whenever possible. Unlike [16], where the optimization variables are decomposed into basic and nonbasic (state and control) variables, this paper uses a full-space approach, which allows one to incorporate recent advances in iterative solvers and preconditioners for so-called KKT or augmented systems. Moreover, we specify and analyze detailed algorithms for substep computations that are necessary for the implementation of the SQP method. One major algorithmic contribution is our extension of the classic Steihaug–Toint conjugate gradient method [25, 26] to include inexact projections onto the constraint null space. The numerical examples documented here and in our report, with Aguiló, [23] show that our SQP algorithm is remarkably robust to the parameter choices in the proposed linear solver stopping conditions.

Reduced-space SQP methods have been investigated by, e.g., Jäger and Sachs [18] and Biros and Ghattas [3]. In both cases, the implementation of the algorithm requires the knowledge of Lipschitz constants and similar quantities that are difficult to estimate in practice. In contrast, our algorithm does not depend on such parameters, and its implementation is therefore fully defined. Other papers like [21] describe SQP methods with globalization strategies and with inexact linear systems solves. However, they only provide heuristics for linear solver tolerances, with no theoretical results.

Inexact line-search SQP methods for nonconvex equality constrained problems are introduced by Byrd, Curtis, and Nocedal in [5], where the optimization steps are computed by an inexact solution of the KKT system. To deal with nonconvexity a perturbation of the Hessian of the Lagrangian is used. This perturbation is determined iteratively and may require repeated KKT solves per step computation. In this paper we use a trust-region approach. In contrast to line-search methods, trust-region methods determine the step by approximately minimizing a quadratic model of the Lagrangian subject to the linearized equality constraints and a trust-region constraint. The resulting subproblems are well posed even if the Hessian is not positive definite on the null space of the linearized constraints and, consequently, no Hessian perturbation is needed to enforce convexity of the subproblems. Moreover, the trust-region constraint acts as a regularizing term for the step, which can be beneficial for ill-conditioned problems. However, compared to line-search methods, the step computation can be more involved, especially for convex well-posed problems.

Recently, Ziems and Ulbrich [27] presented an inexact trust-region SQP method for PDE-constrained optimization. The major contribution of [27] is the rigorous treatment of adaptive PDE discretizations. We focus on the rigorous treatment of iterative linear system solves. Our results are independent of the problem type, i.e., large-scale equality constraints other than PDE constraints can be considered. As in [16], Ziems and Ulbrich use the splitting of the variables x into states y and controls u and the null-space representation given in (2.7) below. Our full-space approach does not require this splitting. It is based on (2.8) below and enables a robust use of iterative saddle-point and KKT solvers in large-scale optimization. Overall, the algorithms for substep computations developed in this paper complement [27].

The paper is organized as follows. In section 2 we review composite-step trust-region SQP with exact linear system solves. In section 3 we introduce our trust-region

SQP algorithm with inexact linear system solves and state the global convergence result. The statement of the algorithm is based on general convergence conditions on its substeps. We present concrete algorithms to compute the substeps in section 4, along with the analyses to prove that the substeps meet the global convergence conditions. Section 5 illustrates the performance of our algorithm on a discretized optimal control problem governed by the Navier–Stokes equations.

2. The composite-step trust-region SQP algorithm and approximate representations of the constraint null space. We begin with a summary of the classical composite-step trust-region SQP framework, based on the work presented in [8]; see also [7, section 15.4.2].

Let $\mathcal{X}$ and $\mathcal{C}$ be Hilbert spaces and let $f : \mathcal{X} \to \mathbb{R}$ and $c : \mathcal{X} \to \mathcal{C}$ be sufficiently smooth functions. We consider the equality constrained NLP

$$\min \quad f(x) \tag{2.1a}$$

$$\text{s.t.} \quad c(x) = 0. \tag{2.1b}$$

The basic formulation of our algorithm requires that f and c are continuously Fréchet differentiable and that the Fréchet derivative $c_x(x)$ of the constraint is surjective. To prove convergence we will need additional assumptions, stated in section 3.5.

Throughout the paper we identify the duals of the Hilbert spaces $\mathcal{X}$ and $\mathcal{C}$ with themselves, i.e., $\mathcal{X}^* = \mathcal{X}$ and $\mathcal{C} = \mathcal{C}^*$. This greatly simplifies the presentation of the optimization algorithm. Of course, one can take $\mathcal{X}^* = \mathcal{X} = \mathbb{R}^n$ and $\mathcal{C} = \mathcal{C}^* = \mathbb{R}^m$.

Let $\mathcal{L} : \mathcal{X} \times \mathcal{C} \to \mathbb{R}$,

$$\mathcal{L}(x, \lambda) = f(x) + \langle \lambda, c(x) \rangle_{\mathcal{C}},$$

be the Lagrangian for (2.1). Let x_k be the kth SQP iterate and λ_k the Lagrange multiplier estimate at x_k. Let $H_k = H(x_k, \lambda_k)$ be the Hessian $\nabla_{xx}\mathcal{L}(x_k, \lambda_k)$ of the Lagrangian or a self-adjoint approximation thereof. Trust-region SQP methods compute an approximate solution of (2.1) by approximately solving a sequence of subproblems derived from

$$\min \quad \frac{1}{2} \langle H_k s, s \rangle_{\mathcal{X}} + \langle \nabla_x \mathcal{L}(x_k, \lambda_k), s \rangle_{\mathcal{X}} + \mathcal{L}(x_k, \lambda_k) \tag{2.2a}$$

$$\text{s.t.} \quad c_x(x_k)s + c(x_k) = 0, \tag{2.2b}$$

$$\|s\|_{\mathcal{X}} \le \Delta_k, \tag{2.2c}$$

where Δ_k is the trust-region radius. To deal with the possible incompatibility of the constraints (2.2b), (2.2c) we apply a Byrd–Omojokun-like composite step approach. See [7, section 15.4.2] for an overview.

The trial step s_k is computed as the sum of a quasi-normal step n_k and a tangential step t_k; see Figure 1 (left pane). The role of the quasi-normal step n_k is to reduce linear infeasibility. It is computed as an approximate solution of

$$\min \quad \|c_x(x_k)n + c(x_k)\|_{\mathcal{C}}^2 \tag{2.3a}$$

$$\text{s.t.} \quad \|n\|_{\mathcal{X}} \le \zeta \Delta_k, \tag{2.3b}$$

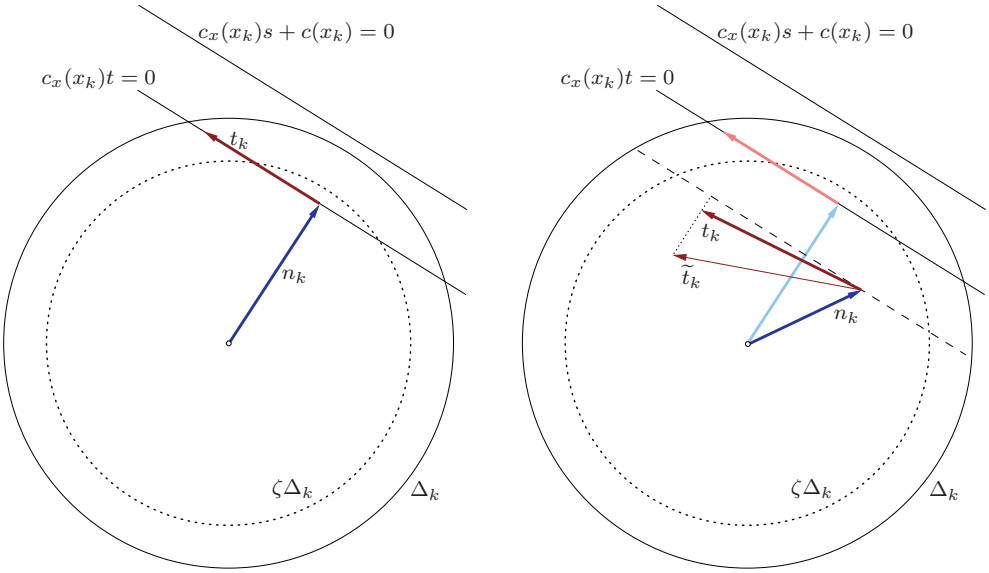

FIG. 1. *Left: a sketch of the quasi-normal step n_k and the tangential step t_k computed using exact linear system solves. Right: a sketch of the quasi-normal step n_k, the solution $\widetilde{t}_k$ of the tangential subproblem, and the tangential step t_k computed using inexact linear system solves.*

where $\zeta \in (0,1)$ is a fixed constant. Once the quasi-normal step n_k is computed, the tangential step t_k is computed as an approximate solution of the subproblem

$$\begin{aligned} &\min \quad \frac{1}{2}\langle H_k(t+n_k), t+n_k\rangle_{\mathcal{X}} + \langle \nabla_x \mathcal{L}(x_k,\lambda_k), t+n_k\rangle_{\mathcal{X}} + \mathcal{L}(x_k,\lambda_k) && \text{(2.4a)}\\ &\text{s.t.} \quad c_x(x_k)t = 0, && \text{(2.4b)}\\ &\qquad \|t+n_k\|_{\mathcal{X}} \le \Delta_k. && \text{(2.4c)} \end{aligned}$$

For the computation of the tangential step one typically eliminates the constraints (2.4b) using a representation of the null space of $c_x(x_k)$. Let $\mathcal{Z}$ be a Hilbert space and let $W_k : \mathcal{Z} \to \mathcal{X}$ be a bounded linear operator such that

$$\text{Range}(W_k) = \text{Null}(c_x(x_k)).$$

We can set $t = W_k w$ and replace (2.4) by

$$\begin{aligned} &\min \quad \frac{1}{2}\langle W_k^* H_k W_k w, w\rangle_{\mathcal{Z}} + \langle W_k^* g_k, w\rangle_{\mathcal{Z}} && \text{(2.5a)}\\ &\text{s.t.} \quad \|n_k + W_k w\|_{\mathcal{X}} \le \Delta_k, && \text{(2.5b)} \end{aligned}$$

where

$$g_k = \nabla_x \mathcal{L}(x_k,\lambda_k) + H_k n_k. \qquad \text{(2.6)}$$

Problems (2.4) and (2.5) are equivalent up to the constant $\mathcal{L}(x_k,\lambda_k) + \langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k\rangle_{\mathcal{X}} + \frac{1}{2}\langle H_k n_k, n_k\rangle_{\mathcal{X}}$ in (2.4a). An approximate solution can be computed, e.g., by the Steihaug–Toint conjugate gradient method [7, section 7.5.1].

There are several ways to define a null-space representation W_k. For small to medium size problems in $\mathcal{X} = \mathbb{R}^n$ one can use a QR decomposition. If $x = (y, u) \in \mathcal{X} = \mathcal{Y} \times \mathcal{U}$ and if the partial Jacobian $c_y(y_k, u_k)$ has a continuous inverse, one can use $\mathcal{Z} = \mathcal{U}$ and

$$W_k = \begin{pmatrix} -c_y(y_k, u_k)^{-1} c_u(y_k, u_k) \\ I \end{pmatrix}. \tag{2.7}$$

This choice is often possible in PDE-constrained optimization, where the application of $c_y(y_k, u_k)^{-1}$ to a vector, which is needed for the application of (2.7), requires the solution of a linearized PDE. However, the splitting $x = (y, u)$ of the optimization variables into basic and nonbasic variables is not natural for all optimization problems. A more general alternative is to choose $\mathcal{Z} = \mathcal{X}$ and W_k to be the projection onto the null space of $c_x(x_k)$. In this case $W_k = W_k^* = W_k^2$ and $t = W_k w$ can be computed by solving the so-called *augmented system*

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} t \\ z \end{pmatrix} = \begin{pmatrix} w \\ 0 \end{pmatrix}. \tag{2.8}$$

First, we note that for (2.8) to have a solution it is sufficient that $c_x(x_k)$ be surjective, i.e., a splitting $x = (y, u)$ of optimization variables, such that the partial Jacobian $c_y(y_k, u_k)$ is invertible, need not be known a priori. Another potential advantage is that the system (2.8) can be solved using a variety of iterative methods for saddle-point problems, including recent advances in KKT solvers; see, e.g., [2, 15, 11] and references therein. Third, to obtain fast convergence, optimization algorithms based on the null-space representation (2.7) typically require an efficient preconditioner for the reduced Hessian operator, $W_k^* H_k W_k$. Such preconditioners are often difficult to construct. Algorithms based on the solution of (2.8) can make use of solvers for the full KKT system, i.e., the system (2.8), where the operator I is replaced by H_k, and bypass the challenge of reduced-Hessian preconditioning [14, p. 1381].

Once the trial step s_k is computed, we must decide whether to accept the step and how to update the trust-region radius Δ_k. We use the augmented Lagrangian merit function

$$\phi(x, \lambda; \rho) = f(x) + \langle \lambda, c(x) \rangle_{\mathcal{C}} + \rho \|c(x)\|_{\mathcal{C}}^2 = \mathcal{L}(x, \lambda) + \rho \|c(x)\|_{\mathcal{C}}^2 \tag{2.9}$$

to perform these tasks. The step is accepted or rejected and the trust-region radius is updated based on the ratio between the actual reduction

$$ared(s_k; \rho_k) = \phi(x_k, \lambda_k; \rho_k) - \phi(x_k + s_k, \lambda_{k+1}; \rho_k) \tag{2.10}$$

and the predicted reduction

$$\begin{aligned} pred(s_k; \rho_k) = \phi(x_k, \lambda_k; \rho_k) - \Big[& \mathcal{L}(x_k, \lambda_k) + \langle \nabla_x \mathcal{L}(x_k, \lambda_k), s_k \rangle_{\mathcal{X}} + \frac{1}{2} \langle H_k s_k, s_k \rangle_{\mathcal{X}} \\ & + \langle \lambda_{k+1} - \lambda_k, c_x(x_k) s_k + c(x_k) \rangle_{\mathcal{C}} + \rho_k \|c_x(x_k) s_k + c(x_k)\|_{\mathcal{C}}^2 \Big]. \end{aligned} \tag{2.11}$$

Here λ_{k+1} is a Lagrange multiplier estimate corresponding to the trial iterate $x_k + s_k$.

In summary, one step of the composite-step trust-region SQP algorithm involves the following tasks.

ALGORITHM 2.1. ONE STEP OF A COMPOSITE-STEP TRUST-REGION SQP ALGORITHM.

1. Compute quasi-normal step n_k.
2. Compute tangential step t_k.
3. Compute new Lagrange multiplier estimate λ_{k+1}.
4. Update penalty parameter ρ_k.
5. Compute $ared_k$, $pred_k$.
6. Decide whether to accept the new iterate $x_{k+1} = x_k + n_k + t_k$, and update Δ_{k+1} from Δ_k, based on $ared_k/pred_k$.

Convergence results for this and related algorithms can be found, e.g., in [8, 9, 10]; see also the overview in [7, section 15.4]. A key assumption in these analyses is that the tangential step t_k lies exactly in the range of the null-space operator W_k. If the linear systems arising from null-space representations (2.7) and (2.8) are solved approximately, this assumption is no longer true. This and additional complications are analyzed in the following section.

3. The composite-step trust-region SQP algorithm with inexact linear system solves. Methods for the computation of the quasi-normal step n_k and the computation of the Lagrange multiplier λ_{k+1} require the solution of one linear system involving the Jacobian of c or its adjoint. The computation of the tangential step t_k requires the solution of several linear systems involving the Jacobian of c or its adjoint. If these linear systems are solved iteratively, then n_k, λ_{k+1}, and t_k cannot be computed exactly and several properties of the steps, such as the property that the tangential step lies in the null space of the linearized constraints, are violated; see Figure 1. The design of stopping criteria for iterative linear systems solves (the accuracy requirements for these steps) is delicate, since one quantity depends on others. For example, the tangential-step subproblem (2.4) depends on the quasi-normal step n_k. If the quasi-normal step n_k is not computed accurately enough, then no matter how accurately the linear systems in the tangential-step computation are solved, the trial step $s_k = n_k + t_k$ will not be accepted. On the other hand, the expense of a very accurate computation of the quasi-normal step will be wasted if the following tangential-step computation uses linear system solves that are too coarse. Thus we want to design stopping criteria for the linear system solves arising in Algorithm 2.1 so that we obtain a globally convergent algorithm, but avoid the oversolving of linear systems. In particular, we must adjust the accuracy of the linear system solves based on the progress of the trust-region SQP algorithm.

Adjusting the accuracy of the linear system solves based on the progress of the trust-region SQP algorithm can be achieved using the techniques developed in [16], which result in subtle but important algorithmic changes in the trust-region SQP algorithm. In this section we extend the inexactness framework of [16] by removing the requirement that $\mathcal{X} = \mathcal{Y} \times \mathcal{U}$ and that the optimization variable $x = (y, u)$ be decomposed into basic and nonbasic variables so that the partial Jacobian $c_y(y_k, u_k)$ has a continuous inverse. We focus on the scenario where $\mathcal{Z} = \mathcal{X}$ and W_k is the projection onto the null space of $c_x(x_k)$ computed by solving the augmented system (2.8).

3.1. Inexactness in the quasi-normal step. The quasi-normal step n_k is computed as an approximate solution of (2.3). The approximate solution must satisfy the boundedness condition

$$\|n_k\|_{\mathcal{X}} \le \kappa_1 \|c(x_k)\|_{\mathcal{C}}, \tag{3.1}$$

where $\kappa_1 > 0$ is independent of k, and the fraction of Cauchy decrease condition

$$\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2 \geq \kappa_2 \|c(x_k)\|_{\mathcal{C}} \min\{\kappa_3 \|c(x_k)\|_{\mathcal{C}}, \Delta_k\}, \tag{3.2}$$

where $\kappa_2, \kappa_3 > 0$ are independent of k. These conditions on the quasi-normal step are derived in [8]. In section 4.1 we specify a concrete algorithm for the computation of n_k, based on the inexact solution of an augmented system, and demonstrate that conditions (3.1) and (3.2) can be easily satisfied.

3.2. Inexactness in the tangential step. The tangential step t_k is an approximate solution of (2.4) or, equivalently, $t_k = W_k w_k$, where w_k is an approximate solution of (2.5). In practice, one first computes w_k as an approximate solution of (2.5), and then one applies W_k to get $t_k = W_k w_k$. In the inexactness framework it is important to separate these two steps.

With inexact linear system solves in the application of W_k, the reduced gradient $W_k g_k$ and the reduced Hessian $W_k H_k W_k$ are no longer available. We will show in section 4.2 that the inexact solution of the linear systems (2.8) leads to an approximation $\widetilde{W}_k$ of W_k. This approximation is in general not self-adjoint. Furthermore, we will show that if inexact linear system solvers are used, an algorithm that computes the solution w_k of (2.5) effectively solves

$$\min \quad \frac{1}{2}\langle \widetilde{W}_k^* H_k \widetilde{W}_k w, w\rangle_{\mathcal{X}} + \langle \widetilde{W}_k^* \widetilde{W}_k g_k, w\rangle_{\mathcal{X}} \tag{3.3a}$$

$$\text{s.t.} \quad \|n_k + \widetilde{W}_k w\|_{\mathcal{X}} \leq \Delta_k \tag{3.3b}$$

instead of (2.5). As before, $g_k = \nabla_x \mathcal{L}(x_k, \lambda_k) + H_k n_k$. Note that (3.3) is equivalent to

$$\min \quad \frac{1}{2}\langle H_k \widetilde{t}, \widetilde{t}\rangle_{\mathcal{X}} + \langle \widetilde{W}_k g_k, \widetilde{t}\rangle_{\mathcal{X}} \tag{3.4a}$$

$$\text{s.t.} \quad \widetilde{t} \in \text{Range}(\widetilde{W}_k), \tag{3.4b}$$

$$\|n_k + \widetilde{t}\|_{\mathcal{X}} \leq \Delta_k. \tag{3.4c}$$

The approximate solutions of (3.3) and (3.4) are denoted by w_k and $\widetilde{t}_k$, respectively, and they obey $\widetilde{t}_k = \widetilde{W}_k w_k$. We describe in the next section how the tangential step t_k is computed once an approximate solution w_k of (3.3) or $\widetilde{t}_k$ of (3.4) is found.

Remark 3.1. Comparing (3.3) with (2.5) one may expect an objective function in (3.3a) with $\widetilde{W}_k g_k$ instead of $\widetilde{W}_k^* \widetilde{W}_k g_k$. Of course, if $\widetilde{W}_k = W_k$, then $W_k = W_k^* = W_k^2$ implies that (3.3) and (2.5) are identical. In the inexact case, using (3.3a) will allow us to prove the important decrease condition (3.7) stated below. We will discuss these issues further in section 4.2.

We define

$$\widetilde{q}_k(t) \stackrel{\text{def}}{=} \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}} + \langle \widetilde{W}_k g_k, t\rangle_{\mathcal{X}}. \tag{3.5}$$

To establish convergence of our trust-region algorithm with inexact linear system solves, we need to impose requirements on the size of the perturbation $\widetilde{W}_k - W_k$. First, the inexact reduced gradient $\widetilde{W}_k g_k$ needs to satisfy

$$\|\widetilde{W}_k g_k - W_k g_k\|_{\mathcal{X}} \leq \xi_1 \min\{\|\widetilde{W}_k g_k\|_{\mathcal{X}}, \Delta_k\} \tag{3.6}$$

for some $\xi_1 > 0$ independent of k. Second, we impose the fraction of Cauchy decrease condition on $\widetilde{t}_k$ with respect to the inexact quadratic model $\widetilde{q}_k$,

$$\widetilde{q}_k(0) - \widetilde{q}_k(\widetilde{t}_k) \geq \kappa_4 \|\widetilde{W}_k g_k\|_{\mathcal{X}} \min\left\{\kappa_5 \|\widetilde{W}_k g_k\|_{\mathcal{X}}, \kappa_6 \Delta_k\right\} \tag{3.7}$$

for κ_4, κ_5, $\kappa_6 > 0$, independent of k. Establishing the existence of $\widetilde{W}_k$ and ensuring that it satisfies (3.6) and (3.7) is not trivial. We will present and analyze an algorithm for the computation of $\widetilde{t}_k$ in section 4.2.

3.3. Balancing progress in the computation of the tangential and the quasi-normal steps. After we have computed an approximate solution $\widetilde{t}_k = \widetilde{W}_k w_k$ of (3.4) we have to compute the step s_k. In the exact case, where $\widetilde{W}_k = W_k$, we set $s_k = n_k + \widetilde{t}_k$. With inexactness, however, $\widetilde{t}_k = \widetilde{W}_k w_k$ will, in general, no longer be in the null space of $c_x(x_k)$ and may destroy some of the linear feasibility gained by the quasi-normal step n_k. To compensate for this, we will compute t_k from $\widetilde{t}_k$ to restore linear feasibility as needed. See Figure 1 (right pane) for an illustration. This computation of the tangential step t_k from $\widetilde{t}_k$ is already used in [16], and we adapt it to the full-space case. To motivate our approach, we revisit the computation of the predicted reduction (2.11) if the null-space projection can be performed exactly.

If W_k is the exact projection onto $\mathrm{Null}(c_x(x_k))$, then $s_k = n_k + W_k w_k$ and

$$c_x(x_k)s_k = c_x(x_k)(n_k + W_k w_k) = c_x(x_k)n_k.$$

In this case, using $W_k = W_k^* = W_k^2$, the predicted reduction can be written as

$$\begin{aligned} pred(s_k;\rho_k) = &- \langle W_k^* W_k(\nabla_x \mathcal{L}(x_k,\lambda_k) + H_k n_k), w_k\rangle_{\mathcal{X}} - \frac{1}{2}\langle W_k^* H_k W_k w_k, w_k\rangle_{\mathcal{X}} \\ &- \langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k\rangle_{\mathcal{X}} - \frac{1}{2}\langle H_k n_k, n_k\rangle_{\mathcal{X}} \\ &- \langle \lambda_{k+1} - \lambda_k, c_x(x_k)n_k + c(x_k)\rangle_{\mathcal{C}} \\ &+ \rho_k\left(\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2\right). \end{aligned}$$

Our goal is to define a new expression for the predicted reduction that is flexible enough to account for inexactness. We make two important changes. First, above, we replace W_k by $\widetilde{W}_k$, in other words, $W_k w_k$ by $\widetilde{W}_k w_k = \widetilde{t}_k$. Hence the term

$$- \langle W_k^* W_k(\nabla_x \mathcal{L}(x_k,\lambda_k) + H_k n_k), w_k\rangle_{\mathcal{X}} - \frac{1}{2}\langle W_k^* H_k W_k w_k, w_k\rangle_{\mathcal{X}}$$

in the predicted reduction becomes

$$\begin{aligned} &- \langle \widetilde{W}_k^* \widetilde{W}_k(\nabla_x \mathcal{L}(x_k,\lambda_k) + H_k n_k), w_k\rangle_{\mathcal{X}} - \frac{1}{2}\langle \widetilde{W}_k^* H_k \widetilde{W}_k w_k, w_k\rangle_{\mathcal{X}} \\ &= - \langle \widetilde{W}_k g_k, \widetilde{t}_k\rangle_{\mathcal{X}} - \frac{1}{2}\langle H_k \widetilde{t}_k, \widetilde{t}_k\rangle_{\mathcal{X}} \\ &= - \widetilde{q}_k(\widetilde{t}_k). \end{aligned}$$

Second, we use $s_k = n_k + t_k$ as the trial step, with

$$c_x(x_k)s_k = c_x(x_k)(n_k + t_k) = c_x(x_k)n_k + r_k^t \neq c_x(x_k)n_k,$$

where we have defined

$$r_k^t \stackrel{\text{def}}{=} c_x(x_k)t_k,$$

and replace each occurrence of $c_x(x_k)n_k$ by $c_x(x_k)n_k + r_k^t$. Hence the term

$$-\langle \lambda_{k+1} - \lambda_k, c_x(x_k)n_k + c(x_k)\rangle_{\mathcal{C}} + \rho_k \left(\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2\right)$$

in the predicted reduction is appended with the term

$$-\left\langle \lambda_{k+1} - \lambda_k, r_k^t\right\rangle_{\mathcal{C}} - \rho_k \|r_k^t\|_{\mathcal{C}}^2 - 2\rho_k \left\langle r_k^t, c_x(x_k)n_k + c(x_k)\right\rangle_{\mathcal{C}}.$$

This motivates the following definition of the *predicted reduction with inexactness*,

$$\widehat{pred}(s_k;\rho_k) \stackrel{\text{def}}{=} pred(n_k, \widetilde{t}_k;\rho_k) + rpred(r_k^t;\rho_k),$$

where we have used the definitions

$$\begin{aligned} pred(n_k, \widetilde{t}_k;\rho_k) \stackrel{\text{def}}{=} & -\left\langle \widetilde{W}_k g_k, \widetilde{t}_k\right\rangle_{\mathcal{X}} - \frac{1}{2}\left\langle H_k \widetilde{t}_k, \widetilde{t}_k\right\rangle_{\mathcal{X}} \\ & - \langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k\rangle_{\mathcal{X}} - \frac{1}{2}\langle H_k n_k, n_k\rangle_{\mathcal{X}} \\ & - \langle \lambda_{k+1} - \lambda_k, c_x(x_k)n_k + c(x_k)\rangle_{\mathcal{C}} \\ & + \rho_k \left(\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2\right) \end{aligned} \tag{3.8}$$

and

$$rpred(r_k^t;\rho_k) \stackrel{\text{def}}{=} -\left\langle \lambda_{k+1} - \lambda_k, r_k^t\right\rangle_{\mathcal{C}} - \rho_k \|r_k^t\|_{\mathcal{C}}^2 - 2\rho_k \left\langle r_k^t, c_x(x_k)n_k + c(x_k)\right\rangle_{\mathcal{C}}. \tag{3.9}$$

In our algorithm, we first compute a penalty parameter ρ_k satisfying

$$pred(n_k, \widetilde{t}_k;\rho_k) \geq \frac{\rho_k}{2}\left(\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2\right),$$

and then, if necessary, we apply an approximate projection to $\widetilde{t}_k$ to compute the tangential step t_k such that it satisfies the requirement

$$|rpred(r_k^t;\rho_k)| \leq \eta_0\, pred(n_k, \widetilde{t}_k;\rho_k), \tag{3.10}$$

where $\eta_0 \in (0, 1-\eta_1)$, and $\eta_1 \in (0,1)$ is the smallest acceptable ratio of the actual and predicted reduction. Since

$$|rpred(r_k^t;\rho_k)| \leq (\|\lambda_{k+1} - \lambda_k\| + 2\rho_k \|c_x(x_k)n_k + c(x_k)\|)\|r_k^t\| + \rho_k \|r_k^t\|^2,$$

(3.10) is satisfied if

$$\|c_x(x_k)t_k\| = \|r_k^t\| \leq -\sigma_k + \sqrt{\sigma_k^2 + \eta_0\, pred(n_k, \widetilde{t}_k;\rho_k)/\rho_k}, \tag{3.11}$$

where $\sigma_k \stackrel{\text{def}}{=} (\|\lambda_{k+1} - \lambda_k\| + 2\rho_k\|c_x(x_k)n_k + c(x_k)\|)/(2\rho_k)$.

There are two additional conditions on the tangential step. The first condition is related to how much the tangential step t_k can deviate from the projection $W_k\widetilde{t}_k$ of $\widetilde{t}_k$ and reads

$$\|t_k - W_k\widetilde{t}_k\|_{\mathcal{X}} \leq \xi_3 \Delta_k \min\{\Delta_k, \|s_k\|_{\mathcal{X}}\}, \tag{3.12}$$

for some $\xi_3 > 0$ independent of k. The second condition is

$$\|\widetilde{t}_k\|_{\mathcal{X}} \leq \xi_4 \|s_k\|_{\mathcal{X}}, \tag{3.13}$$

for $\xi_4 > 0$ independent of k. We note that (3.12), (3.13), and $\|W_k\| = 1$ imply that

$$\|t_k\|_{\mathcal{X}} \leq \|W_k\widetilde{t}_k\|_{\mathcal{X}} + \|t - W_k\widetilde{t}_k\|_{\mathcal{X}} \leq (\xi_4 + \xi_3\Delta_k)\|s_k\|_{\mathcal{X}} \leq (\xi_4 + \xi_3\Delta_{\max})\|s_k\|_{\mathcal{X}}. \tag{3.14}$$

3.4. Lagrange multipliers. The Lagrange multiplier estimate λ_k is computed as an approximate solution of $\min \|\nabla f(x_k) + c_x(x_k)^* \lambda\|_{\mathcal{X}^*}$. The global convergence theory for our inexact trust-region SQP algorithm only requires that the sequence of Lagrange multiplier estimates $\{\lambda_k\}_{k\in\mathbb{N}}$ be bounded. It is relatively easy to compute a Lagrange multiplier estimate so that this assumption is satisfied. A concrete algorithm based on the approximate solution of an augmented system is specified in section 4.4. Of course, poor Lagrange multiplier estimates often lead to slowly converging SQP algorithms, and thus additional assumptions are often desired; see section 4.4.

Remark 3.2. After n_k and $\widetilde{t}_k$ are computed, the tangential step t_k is computed to satisfy (3.10). However, the right-hand side in (3.10), defined in (3.8), contains a reference to λ_{k+1}, the Lagrange multiplier estimate at the trial step $x_k + n_k + t_k$. This apparent circular dependence arises because we want to relax our accuracy requirements on the tangential step as much as possible. In the exact trust-region SQP algorithm this issue does not arise, since $r_k^t = c_x(x_k)t_k = 0$. We will describe our approach to break the circular dependence of λ_{k+1} and t_k in section 4.3.

3.5. The trust-region SQP algorithm with inexactness control. We can now formulate the SQP algorithm in the inexact setting.

ALGORITHM 3.3. TRUST-REGION SQP ALGORITHM WITH INEXACTNESS CONTROL.

1. Initialization. Choose initial point x_0, initial trust-region radius Δ_0, constants $0 < \alpha_1$, $\eta_1 < 1$, $0 < \eta_0 < 1 - \eta_1$, $\rho_{-1} \geq 1$, $\bar{\rho} > 0$, and $tol^{SQP} > 0$. Set Δ_{min}, Δ_{max} so that $0 < \Delta_{min} < \Delta_{max}$. Compute an initial Lagrange multiplier estimate λ_0.
2. For $k = 0, 1, 2, \ldots$
   (a) Convergence check. If $\|\nabla_x \mathcal{L}(x_k, \lambda_k)\|_{\mathcal{X}} < tol^{SQP}$ and $\|c(x_k)\|_{\mathcal{C}} < tol^{SQP}$, then terminate.
   (b) Step computation.
      i. Compute a quasi-normal step n_k satisfying (3.1) and (3.2).
      ii. Compute $\widetilde{t}_k$ satisfying (3.6)–(3.7).
   (c) Acceptance test.
      i. Compute a new Lagrange multiplier estimate λ_{k+1}.
      ii. Update the penalty parameter. If

$$pred(n_k, \widetilde{t}_k; \rho_{k-1}) \geq \frac{\rho_{k-1}}{2} \left(\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2 \right)$$

      then set $\rho_k = \rho_{k-1}$. Otherwise set

$$\rho_k = \frac{-2\, pred(n_k, \widetilde{t}_k; \rho_{k-1})}{\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2} + 2\rho_{k-1} + \bar{\rho}.$$

      iii. Compute the tangential step t_k satisfying the conditions (3.10), (3.12), and (3.13); see Remark 3.4.
      iv. Compute the trial step $s_k = n_k + t_k$.
      v. Compute $\theta_k = ared(s_k; \rho_k)/pred(n_k, \widetilde{t}_k; \rho_k)$.
      vi. If $\theta_k \geq \eta_1$, set $x_{k+1} = x_k + s_k$, and choose Δ_{k+1} such that

$$\max\{\Delta_{min}, \Delta_k\} \leq \Delta_{k+1} \leq \Delta_{max}.$$

      Otherwise set $x_{k+1} = x_k$, $\lambda_{k+1} = \lambda_k$, and $\Delta_{k+1} = \alpha_1 \|s_k\|_{\mathcal{X}}$.

Remark 3.4. Algorithm 3.3 provides a general inexact trust-region SQP framework, which is used in our analysis. However, its implementation is not fully defined at this point. This is done in section 4. For instance, to compute a tangential step t_k satisfying (3.10), (3.12), and (3.13) the steps 2(bi) and 2(bii) may need to be repeated; a fully implementable procedure for the steps 2(bi)–2(civ) is presented in Algorithm 4.18.

To prove global convergence of our algorithm we make the following assumptions, comparable to those made in, e.g., [8, 9, 10, 16].

- (A1) There exists a convex open set Ω in $\mathcal{X}$ such that for all iterations k, $x_k \in \Omega$ and $x_k + s_k \in \Omega$.
- (A2) The function(al)s f and c are twice continuously Fréchet differentiable with Lipschitz continuous second derivatives.
- (A3) The operator $c_x(x)$ is surjective for all $x \in \Omega$.
- (A4) The following function(al)s and operators are uniformly bounded over all $x \in \Omega$: $f(x)$, $\nabla_x f(x)$, $\nabla_{xx} f(x)$, $c(x)$, $c_x(x)$, $(c_x(x)c_x(x)^*)^{-1}$, and $c_{xx}(x)$.
- (A5) The sequence of operators $\{H_k\}_{k\in\mathbb{N}}$ is bounded.
- (A6) The sequence of Lagrange multipliers $\{\lambda_k\}_{k\in\mathbb{N}}$ is bounded.
- (A7) Let $W_k : \mathcal{X} \to \mathcal{X}$, $k \in \mathbb{N}$, be the projection onto $\mathrm{Null}(c_x(x_k))$. In particular $\|W_k\|_{L(\mathcal{X})} = 1$.

The global convergence property of Algorithm 3.3 is stated in Theorem 3.5.

THEOREM 3.5. *Let assumptions* (A1)–(A7) *be satisfied. The sequences of iterates generated by Algorithm* 3.3 *satisfy*

$$\liminf_{k\to\infty} (\|\widetilde{W}_k g_k\|_{\mathcal{X}} + \|c(x_k)\|_{\mathcal{C}}) = 0. \tag{3.15}$$

Additionally, we have

$$\liminf_{k\to\infty} (\|W_k \nabla_x f(x_k)\|_{\mathcal{X}} + \|c(x_k)\|_{\mathcal{C}}) = 0. \tag{3.16}$$

The proof of Theorem 3.5 is based on the global convergence analysis given in [8] and follows the modifications given in [16]. It is stated in Appendix A.

4. Subproblem algorithms with stopping conditions for linear solvers. In section 3 we described our inexact trust-region SQP method and established a first-order global convergence result. We now specify concrete algorithms

- for computing a quasi-normal step n_k that satisfies conditions (3.1) and (3.2);
- for the approximate solution of the tangential subproblem (3.4), so that the existence of $\widetilde{W}_k$ can be guaranteed and so that the solution $\widetilde{t}_k$ of (3.4) satisfies conditions (3.6) and (3.7);
- for the computation of a tangential step t_k that satisfies conditions (3.10), (3.12), and (3.13); and
- for the computation of a Lagrange multiplier estimate λ_k.

4.1. Computation of the quasi-normal step. The quasi-normal step n_k is computed as an approximate solution of (2.3). It must satisfy (3.1) and (3.2). It follows from the standard theory of trust-region methods (see, e.g., [7, section 6.3.2]) that the fraction of Cauchy decrease condition (3.2) is satisfied if

$$\begin{aligned} &\|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}}^2 \\ &\quad \le \min\left\{ \|c_x(x_k)n + c(x_k)\|_{\mathcal{C}}^2 : n = -\alpha c_x(x_k)^* c(x_k), \|n\|_{\mathcal{X}} \le \zeta\Delta_k, \alpha \ge 0 \right\}. \end{aligned} \tag{4.1}$$

We use a dogleg method to compute an approximate solution of (2.3); see, e.g., [7, section 7.5.3]. Let n_k^{cp} be the Cauchy point, i.e., the solution of $\min\left\{\|c_x(x_k)n + c(x_k)\|_{\mathcal{C}}^2 : n = -\alpha c_x(x_k)^* c(x_k), \alpha \geq 0\right\}$. It is easy to verify that

$$n_k^{cp} = -\frac{\|c_x(x_k)^* c(x_k)\|_{\mathcal{X}}^2}{\|c_x(x_k)c_x(x_k)^* c(x_k)\|_{\mathcal{C}}^2} c_x(x_k)^* c(x_k). \tag{4.2}$$

If $\|n_k^{cp}\|_{\mathcal{X}} \geq \zeta\Delta_k$, then we set the quasi-normal step to $n_k = \zeta\Delta_k n_k^{cp}/\|n_k^{cp}\|_{\mathcal{X}}$.

If $\|n_k^{cp}\|_{\mathcal{X}} < \zeta\Delta_k$, then we compute an approximate minimum norm solution n_k^N of $\min \|c_x(x_k)n + c(x_k)\|_{\mathcal{C}}^2$ and compute the quasi-normal step by moving from n_k^{cp} as far as possible toward n_k^N while staying within the trust region with radius $\zeta\Delta_k$. The minimum norm solution n_k^N can be computed by solving an augmented system; see [4, p. 7]. However, when the augmented system is solved iteratively, we find it more efficient in practice to solve for the step increment $\delta n_k = n_k^N - n_k^{cp}$ rather than for n_k^N. That is we solve

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} \delta n_k \\ y \end{pmatrix} = \begin{pmatrix} -n_k^{cp} + e_1 \\ -c_x(x_k)n_k^{cp} - c(x_k) + e_2 \end{pmatrix}. \tag{4.3}$$

We allow a nonzero residual $(e_1 \;\; e_2) \in \mathcal{X} \times \mathcal{C}$ whose size is restricted in Lemma 4.2.

Our algorithm for computing the quasi-normal step is given as follows.

ALGORITHM 4.1. DOGLEG METHOD FOR THE QUASI-NORMAL SUBPROBLEM.

1. Compute n_k^{cp} as defined in (4.2).
2. If $\|n_k^{cp}\|_{\mathcal{X}} \geq \zeta\Delta_k$, then set $n_k = \zeta\Delta_k n_k^{cp}/\|n_k^{cp}\|_{\mathcal{X}}$.
3. Else compute δn_k via (4.3), such that e_1 and e_2 satisfy (4.4).
   If $\|n_k^{cp} + \delta n_k\|_{\mathcal{X}} \leq \zeta\Delta_k$, then set $n_k = n_k^N$.
   Else compute $\theta_k \in (0,1)$ such that $\|n_k^{cp} + \theta_k \delta n_k\|_{\mathcal{X}} = \zeta\Delta_k$, and set $n_k = n_k^{cp} + \theta_k \delta n_k$.

LEMMA 4.2. *Let Assumption* (A4) *be satisfied. If*

$$\|e_1\|_{\mathcal{X}}^2 + \|e_2\|_{\mathcal{C}}^2 \leq \|c_x(x_k)n_k^{cp} + c(x_k)\|_{\mathcal{C}}^2, \tag{4.4}$$

then the inexact quasi-normal step n_k computed using Algorithm 4.1 *satisfies* (3.1) *and* (3.2).

Proof. i. We show that (3.1) is satisfied. Assumption (A4) guarantees the existence of ν_1, $\nu_2 > 0$ such that $\|\left(c_x(x_k)c_x(x_k)^*\right)^{-1}\|_{L(\mathcal{C})} \leq \nu_1$ and $\|c_x(x_k)\|_{L(\mathcal{X},\mathcal{C})} = \|c_x(x_k)^*\|_{L(\mathcal{C},\mathcal{X})} \leq \nu_2$ for all k. The Cauchy point n_k^{cp} defined in (4.2) satisfies

$$\begin{aligned}
\|n_k^{cp}\|_{\mathcal{X}} &= \frac{\|c_x(x_k)^* c(x_k)\|_{\mathcal{X}}^2}{\|c_x(x_k)c_x(x_k)^* c(x_k)\|_{\mathcal{C}}^2} \|c_x(x_k)^* c(x_k)\|_{\mathcal{C}} \\
&= \frac{\|c_x(x_k)^*(c_x(x_k)c_x(x_k)^*)^{-1} c_x(x_k)c_x(x_k)^* c(x_k)\|_{\mathcal{X}}^2}{\|c_x(x_k)c_x(x_k)^* c(x_k)\|_{\mathcal{C}}^2} \|c_x(x_k)^* c(x_k)\|_{\mathcal{C}} \\
&\leq \frac{\|c_x(x_k)^*\|_{L(\mathcal{C},\mathcal{X})}^2 \, \|(c_x(x_k)c_x(x_k)^*)^{-1}\|_{L(\mathcal{C})}^2 \, \|c_x(x_k)c_x(x_k)^* c(x_k)\|_{\mathcal{C}}^2}{\|c_x(x_k)c_x(x_k)^* c(x_k)\|_{\mathcal{C}}^2} \\
&\quad \times \|c_x(x_k)^*\|_{L(\mathcal{C},\mathcal{X})} \, \|c(x_k)\|_{\mathcal{C}} \\
&\leq \nu_1^2 \nu_2^3 \|c(x_k)\|_{\mathcal{C}}.
\end{aligned}$$

If n_k is computed in step 2 of Algorithm 4.1, then $n_k = \zeta\Delta_k n_k^{cp}/\|n_k^{cp}\|_{\mathcal{X}}$ and $\zeta\Delta_k/\|n_k^{cp}\|_{\mathcal{X}} \in (0,1]$. Hence

$$\|n_k\|_{\mathcal{X}} \leq \|n_k^{cp}\|_{\mathcal{X}} \leq \nu_1^2 \nu_2^3 \|c(x_k)\|_{\mathcal{C}}.$$

If n_k is computed in step 3 of Algorithm 4.1, then $n_k = n_k^{cp} + \theta_k \delta n_k$. It is easily shown from (4.3) that

$$\delta n_k = -n_k^{cp} + e_1 + c_x(x_k)^* \left(c_x(x_k)c_x(x_k)^*\right)^{-1} \left(e_2 - c_x(x_k)e_1 - c(x_k)\right).$$

Consequently,

$$\begin{aligned}\|\delta n_k\|_{\mathcal{X}} &\le \|n_k^{cp}\|_{\mathcal{X}} + (1+\nu_1\nu_2^2)\|e_1\|_{\mathcal{X}} + \nu_1\nu_2\|e_2\|_{\mathcal{C}} + \nu_1\nu_2\|c(x_k)\|_{\mathcal{C}} \\ &\le (2+\nu_1\nu_2^2+\nu_1\nu_2^3)\|n_k^{cp}\|_{\mathcal{X}} + (2\nu_1\nu_2+\nu_1\nu_2^2)\|c(x_k)\|_{\mathcal{C}},\end{aligned}$$

where we have used that $\|e_1\|_{\mathcal{X}}, \|e_2\|_{\mathcal{C}} \le \|c_x(x_k)n_k^{cp} + c(x_k)\|_{\mathcal{C}} \le \nu_2\|n_k^{cp}\|_{\mathcal{X}} + \|c(x_k)\|_{\mathcal{C}}$. Since $\theta_k \in (0,1]$, the bound for $\|n_k^{cp}\|_{\mathcal{X}}$ implies that $n_k = n_k^{cp} + \theta_k \delta n_k$ satisfies (3.1).

ii. We show that (3.2) is satisfied. If n_k is computed in step 2 of Algorithm 4.1, then n_k solves (4.1) and, hence, satisfies the Cauchy decrease condition (3.2).

If n_k is computed in step 3 of Algorithm 4.1, then $n_k = n_k^{cp} + \theta_k \delta n_k$ and, due to the second equation in (4.3), $c_x(x_k)\delta n_k = -c_x(x_k)n_k^{cp} - c(x_k) + e_2$. Consequently,

$$\begin{aligned}\|c_x(x_k)n_k + c(x_k)\|_{\mathcal{C}} &= \|c_x(x_k)n_k^{cp} + c(x_k) + \theta_k c_x(x_k)\delta n_k\|_{\mathcal{C}} \\ &= \|(1-\theta_k)(c_x(x_k)n_k^{cp} + c(x_k)) + \theta_k e_2\|_{\mathcal{C}} \\ &\le (1-\theta_k)\|c_x(x_k)n_k^{cp} + c(x_k)\|_{\mathcal{C}} + \theta_k\|e_2\|_{\mathcal{C}} \\ &\le \|c_x(x_k)n_k^{cp} + c(x_k)\|_{\mathcal{C}}.\end{aligned}$$

Since n_k^{cp} solves (4.1) and, hence, satisfies the Cauchy decrease condition (3.2), so does n_k. ∎

Criteria other than (4.4) also guarantee that the inexact quasi-normal step n_k computed using Algorithm 4.1 satisfies (3.1) and (3.2). In the second part of the proof of the previous lemma it was used that $\|e_2\|_{\mathcal{C}} \le \|c_x(x_k)n_k^{cp} + c(x_k)\|_{\mathcal{C}}$. In the first part of the proof it is needed that $\|e_i\|_{(\cdot)} \le \eta_i\|c(x_k)\|_{\mathcal{C}}$, $i = 1, 2$, for some η_i independent of k. The criterion (4.4) is computationally convenient since it specifies a bound on the residual. To additionally control the rate of convergence of the SQP algorithm, we enforce the following *linear solver stopping condition (LSSC)*.

LSSC 4.3. *Let $0 < \xi^{qn} \le 1$. In our implementation, we replace* (4.4) *with the stronger condition*

$$\|e_1\|_{\mathcal{X}}^2 + \|e_2\|_{\mathcal{C}}^2 \le \left(\xi^{qn}\right)^2 \|c_x(x_k)n_k^{cp} + c(x_k)\|_{\mathcal{C}}^2. \tag{4.5}$$

If an iterative solver used for the solution of (4.3) *is initialized with the zero vector, the condition* (4.5) *implies that the initial residual* $(\|c_x(x_k)n_k^{cp} + c(x_k)\|_{\mathcal{C}}^2 + \|n_k^{cp}\|_{\mathcal{X}}^2)^{1/2}$ *is reduced by at least ξ^{qn}.*

4.2. Solution of the tangential subproblem. The second task is the computation of an approximate solution of the tangential subproblem (3.4). In section 4.2.1 we extend the Steihaug–Toint conjugate gradient (STCG) method [25, 26] to handle inexact projections based on the iterative solution of (2.8). Our extension justifies the use of the operator $\widetilde{W}_k$ in the subproblem (3.4). The solution $\widetilde{t}_k$ of (3.4) obtained by our STCG algorithm automatically satisfies condition (3.7). This is a key result, proved in Theorem 4.11. In section 4.2.2 we give conditions on $\widetilde{W}_k$ that are sufficient to guarantee (3.6) and promote fast convergence of the SQP algorithm.

4.2.1. An inexact conjugate gradient method for the solution of the tangential subproblem. In the case where projections W_k can be computed exactly, the solution of the tangential subproblem (2.5) can be accomplished by the STCG method; see [25, 26], [7, section 7.5.1], [19, p. 75]. The application of STCG to (2.5) gives an algorithm which iterates in the w variables. Rather than iterating in the w variables, one can reorganize the computations to iterate in the $t = W_k w$ variables; see [14, 22]. In the case where W_k is the projection onto the null space of $c_x(x_k)$, i.e., has the properties $W_k = W_k^2 = W_k^*$, this leads to the following algorithm in which we use $t_{k,i}$ to denote the ith STCG iterate for the computation of the tangential step t_k.

ALGORITHM 4.4. STCG METHOD FOR THE SOLUTION OF (2.5), IN t_k VARIABLES.

0. Given $tol^{CG} \in (0,1)$. Let $t_{k,_0} = 0$. Let $r_0 = g_k$, $z_0 = W_k g_k$, $p_0 = -z_0$.
1. For $i = 0, 1, 2, \ldots, i_{\max}$
   (a) If $\|z_i\|_{\mathcal{X}} \leq tol^{CG} \|z_0\|_{\mathcal{X}}$, return $t_{k,i}$.
   (b) If $\langle p_i, H_k p_i \rangle_{\mathcal{X}} \leq 0$, compute $\theta > 0$ such that $\|n_k + t_{k,i} + \theta p_i\|_{\mathcal{X}} = \Delta_k$ and return $t_{k,i+1} = t_{k,i} + \theta p_i$.
   (c) $\alpha_i = -\langle r_i, p_i \rangle_{\mathcal{X}} / \langle p_i, H_k p_i \rangle_{\mathcal{X}}$
   (d) $t_{k,i+1} = t_{k,i} + \alpha_i p_i$
   (e) If $\|n_k + t_{k,i+1}\|_{\mathcal{X}} \geq \Delta_k$, compute $\theta > 0$ such that $\|n_k + t_{k,i} + \theta p_i\|_{\mathcal{X}} = \Delta_k$ and return $t_{k,i+1} = t_{k,i} + \theta p_i$.
   (f) $r_{i+1} = H_k t_{k,i+1} + g_k = r_i + \alpha_i H_k p_i$
   (g) $z_{i+1} = W_k r_{i+1}$
   (h) $\beta_i = \langle z_{i+1}, z_{i+1} \rangle_{\mathcal{X}} / \langle z_i, z_i \rangle_{\mathcal{X}}$
   (i) $p_{i+1} = -z_{i+1} + \beta_i p_i$

The application of W_k requires the solution of the augmented system (2.8). If (2.8) is solved iteratively, this leads to inexactness in W_k. Moreover, if the chosen iterative solver depends nonlinearly on initial guesses (this is true, e.g., for Krylov subspace methods such as GMRES, MINRES, or SymmLQ), then an application of W_k to a given vector r is replaced by $\mathcal{W}_k(r)$, where

$$\mathcal{W}_k : \mathcal{X} \to \mathcal{X}$$

is in general a nonlinear operator. To extend the STCG method to handle such inexactness the formulation of the STCG method in the form of Algorithm 4.4 is more suitable than the direct application of STCG to (2.5). In Algorithm 4.4, the application of W_k appears as the application of a preconditioner (although in this form it does not accelerate the convergence). This issue of "nonlinear preconditioners" has been addressed in [24], and specifically for preconditioned conjugate gradient methods in [1, 13, 20]. Our modifications of Algorithm 4.4 are in part motivated by [1, 13, 20]. However, since the conjugate gradient algorithm now serves as a subproblem solver in the SQP method, rather than a linear system solver, the computed solution $\widetilde{t}_k$ must meet the requirements outlined in section 3.2 and must be compatible with the Steihaug-Toint termination criteria. Therefore, a new analysis is needed.

In the inexact regime, one obstacle in applying the STCG method to the tangential subproblem is the loss of symmetry of the inexact reduced Hessian. This is one reason to move nonlinearities (due to the inexact application of W_k) away from the Hessian operator by using Algorithm 4.4 instead of directly applying STCG to (2.5). In addition, due to the inexact application of W_k we lose the three-term recurrence enjoyed by the conjugate gradient method and we must apply full orthogonalization. Our extension of the STCG Algorithm 4.4 to handle inexact applications of W_k is given next. We use $\widetilde{t}_{k,i}$ to denote the ith STCG iterate in the computation of $\widetilde{t}_k$.

ALGORITHM 4.5. STCG METHOD WITH INEXACT NULL-SPACE PROJECTIONS.

0. Given $tol^{CG} \in (0,1)$. Let $\widetilde{t}_{k,0} = 0 \in \mathcal{X}$. Let $\widetilde{r}_0 = \mathcal{W}_k(g_k)$ and $\widetilde{z}_0 = \widetilde{r}_0$.
1. For $i = 0, 1, 2, \ldots, i_{\max}$
   (a) If $\|\widetilde{z}_i\|_{\mathcal{X}} \leq tol^{CG}\|\widetilde{r}_0\|_{\mathcal{X}}$
   then: if $i = 0$ return $\widetilde{t}_k = \widetilde{t}_k^{cp} = 0$; if $i > 0$ return $\widetilde{t}_k = \widetilde{t}_{k,i}$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$.
   (b) $\widetilde{p}_i = -\widetilde{z}_i + \sum_{j=0}^{i-1} \frac{\langle \widetilde{z}_i, H_k\widetilde{p}_j\rangle_{\mathcal{X}}}{\langle \widetilde{p}_j, H_k\widetilde{p}_j\rangle_{\mathcal{X}}}\widetilde{p}_j$
   (c) If $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} \neq 0$ and $\langle \widetilde{p}_i, H\widetilde{p}_i\rangle_{\mathcal{X}} \leq 0$, compute θ such that $\operatorname{sign}(\theta) = \operatorname{sign}(-\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}})$ and $\|n_k + \widetilde{t}_{k,i} + \theta\widetilde{p}_i\|_{\mathcal{X}} = \Delta_k$, and return $\widetilde{t}_k = \widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \theta\widetilde{p}_i$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$.
   If $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} = 0$ and $\langle \widetilde{p}_i, H\widetilde{p}_i\rangle_{\mathcal{X}} < 0$, compute θ such that $\|n_k + \widetilde{t}_{k,i} + \theta\widetilde{p}_i\| = \Delta_k$, and return $\widetilde{t}_k = \widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \theta\widetilde{p}_i$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$.
   (d) If $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} = 0$, return $\widetilde{t}_k = \widetilde{t}_{k,i}$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$.
   (e) $\widetilde{\alpha}_i = -\frac{\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}}}{\langle \widetilde{p}_i, H_k\widetilde{p}_i\rangle_{\mathcal{X}}}$
   (f) $\widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \widetilde{\alpha}_i\widetilde{p}_i$
   (g) If $\|n_k + \widetilde{t}_{k,i+1}\|_{\mathcal{X}} \geq \Delta_k$, compute θ such that $\operatorname{sign}(\theta) = \operatorname{sign}(\widetilde{\alpha}_i)$ and $\|n_k + \widetilde{t}_{k,i} + \theta\widetilde{p}_i\|_{\mathcal{X}} = \Delta_k$, and return $\widetilde{t}_k = \widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \theta\widetilde{p}_i$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$.
   (h) $\widetilde{r}_{i+1} = H_k\widetilde{t}_{k,i+1} + \mathcal{W}_k(g_k) = \widetilde{r}_i + \widetilde{\alpha}_i H_k\widetilde{p}_i$
   (i) $\widetilde{z}_{i+1} = \mathcal{W}_k(\widetilde{r}_{i+1})$

Remark 4.6. i. A straightforward induction argument can be used to show that if $\mathcal{W}_k = W_k$, the quantities generated by Algorithms 4.4 and 4.5 obey

$$t_{k,i} = \widetilde{t}_{k,i}, \quad p_i = \widetilde{p}_i, \quad z_i = \widetilde{z}_i, \quad \alpha_i = \widetilde{\alpha}_i \quad \text{for all } i.$$

However, Algorithm 4.5 includes a subtle modification that will be important to ensure the decrease condition (3.7). In Algorithm 4.4 we have

$$r_0 = g_k, \quad r_{i+1} = H_k t_{k,i+1} + g_k = r_i + \alpha_i H_k p_i,$$

whereas in Algorithm 4.5 we have

$$\widetilde{r}_0 = \mathcal{W}_k(g_k), \quad \widetilde{r}_{i+1} = H_k\widetilde{t}_{k,i+1} + \mathcal{W}_k(g_k) = \widetilde{r}_i + \widetilde{\alpha}_i H_k\widetilde{p}_i.$$

If $\mathcal{W}_k = W_k$ and $W_k = W_k^* = W_k^2$, then $z_{i+1} = W_k r_{i+1} = W_k\widetilde{r}_{i+1} = \widetilde{z}_{i+1}$, which implies then $t_{k,i} = \widetilde{t}_{k,i}$, but

$$r_{i+1} = H_k t_{k,i+1} + g_k \neq \widetilde{r}_{i+1} = H_k t_{k,i+1} + W_k g_k.$$

Another interpretation of this modification is the following. Algorithm 4.4 generates iterates $t_{k,i}$ that are approximate solutions of

$$\min \quad \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}} + \langle g_k, t\rangle_{\mathcal{X}} \tag{4.6a}$$

$$\text{s.t.} \quad t \in \operatorname{Range}(W_k), \tag{4.6b}$$

$$\|n_k + t\|_{\mathcal{X}} \leq \Delta_k. \tag{4.6c}$$

If we replace r_i, $i = 0, 1, \ldots$, in Algorithm 4.4 by

$$r_0 = W_k g_k, \quad r_{i+1} = H_k t_{k,i+1} + W_k g_k = r_i + \alpha_i H_k r_i,$$

then the iterates $t_{k,i}$ are approximate solutions of

$$\min \quad \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}} + \langle W_k g_k, t\rangle_{\mathcal{X}} \tag{4.7a}$$

$$\text{s.t.} \quad t \in \text{Range}(W_k), \tag{4.7b}$$

$$\|n_k + t\|_{\mathcal{X}} \le \Delta_k. \tag{4.7c}$$

Of course, due to $W_k = W_k^* = W_k^2$ the problems (4.6) and (4.7) are identical, since for $t \in \text{Range}(W_k)$ we have $t = W_k t$ and $\langle g_k, t\rangle_{\mathcal{X}} = \langle g_k, W_k t\rangle_{\mathcal{X}} = \langle W_k g_k, t\rangle_{\mathcal{X}}$. If W_k is applied inexactly, then these identities no longer hold and therefore our modification in the definition of $\widetilde{r}_i$ will be important for the proof of the fraction of Cauchy decrease condition (3.7).

ii. If $\mathcal{W}_k = W_k$, then $W_k = W_k^* = W_k^2$ implies $\langle g_k, \widetilde{r}_0\rangle_{\mathcal{X}} = \|W_k g_k\|_{\mathcal{X}}^2$. Hence the stopping criterion in step 1(a) of Algorithm 4.5 reduces to the one in step 1(a) of Algorithm 4.4.

iii. If $\mathcal{W}_k = W_k$, then $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} < 0$ for all i. Since Algorithm 4.5 uses $\mathcal{W}_k$ instead of W_k, it is possible that $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} \ge 0$. This makes the check for negative curvature in step 1(c) of Algorithm 4.5 slightly more involved. Furthermore, if $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} = 0$ in step 1(d), then $\langle \widetilde{p}_i, H\widetilde{p}_i\rangle_{\mathcal{X}} \ge 0$ and Algorithm 4.5 stagnates. Therefore, the stopping condition 1(d) is included in Algorithm 4.5. Due to inexact application of W_k we may have $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} > 0$. Let

$$\widetilde{q}_k(t) \stackrel{\text{def}}{=} \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}} + \langle \mathcal{W}_k(g_k), t\rangle_{\mathcal{X}} \tag{4.8}$$

(we will see at the end of this section why it is justified to use the same notation $\widetilde{q}_k$ in (3.5) and (4.8)). Since $\widetilde{r}_i = \nabla \widetilde{q}_k(\widetilde{t}_{k,i})$, $\widetilde{p}_i$ is not a descent direction of $\widetilde{q}_k$ at $\widetilde{t}_{k,i}$, but $-\widetilde{p}_i$ is. If $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} > 0$, then $\widetilde{\alpha}_i < 0$, hence $\widetilde{q}_k$ will be reduced when moving from $\widetilde{t}_{k,i}$ to $\widetilde{t}_{k,i} + \widetilde{\alpha}_i\widetilde{p}_i$. If $\widetilde{t}_{k,i} + \widetilde{\alpha}_i\widetilde{p}_i$ is outside the trust region, we can only move from $\widetilde{t}_{k,i}$ to $\widetilde{t}_{k,i} + \widetilde{\alpha}_i\widetilde{p}_i$ until the trust-region boundary is reached. This is accomplished in Algorithm 4.5 in step 1(g) by choosing the sign of θ equal to the sign of $\widetilde{\alpha}_i$.

iv. In addition to the solution $\widetilde{t}_k$ of the tangential subproblem, Algorithm 4.5 always returns the tangential Cauchy point $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$, which may be required by Algorithm 4.18 (see section 4.3) to ensure global convergence. Clearly, it is possible that $\widetilde{t}_k = \widetilde{t}_k^{cp}$, in which case only one vector is returned.

Next we state a few useful properties of Algorithm 4.5, labeled conjugacy and orthogonality, respectively. These properties are known to hold for the STCG Algorithm 4.4 with exact linear system solves. Then we state the equivalence of Algorithms 4.5 and 4.4 in the case when W_k can be applied exactly. This result is already mentioned in Remark 4.6(i). The equivalence property is important, as we would like to recover the convergence behavior of the exact STCG method if linear systems can be solved accurately. We then use a standard technique to prove that every iterate of the algorithm minimizes $\widetilde{q}_k(t)$ over the set of previously computed search directions. Finally, we show that, from the point of view of Algorithm 4.5, the nonlinear operator $\mathcal{W}_k$ can be replaced with a linear representation $\widetilde{W}_k$. The existence of $\widetilde{W}_k$ enables us to formulate an inexact quadratic functional which is minimized by Algorithm 4.5.

LEMMA 4.7 (conjugacy property). *Let $\{\widetilde{p}_i\}$ be the sequence of search directions generated by Algorithm 4.5. Then*

$$\langle \widetilde{p}_i, H_k\widetilde{p}_\ell\rangle_{\mathcal{X}} = \langle \widetilde{p}_\ell, H_k\widetilde{p}_i\rangle_{\mathcal{X}} = 0 \quad \textit{for } 0 \le \ell \le i-1.$$

Proof. The result can be proved by a straightforward induction argument. ∎

LEMMA 4.8 (orthogonality property). *Let $\{\widetilde{r}_i\}$ be the sequence of residuals generated by Algorithm* 4.5. *Then*

$$\langle \widetilde{r}_i, \widetilde{p}_\ell \rangle = 0 \text{ and } \langle \widetilde{r}_i, \widetilde{z}_\ell \rangle = 0 \text{ for } 0 \le \ell \le i-1.$$

Proof. The result can be proved by a straightforward induction argument. ∎

The following minimization property is a generalization of the minimization property of the STCG method, proved in [25, Thm. 2.1].

LEMMA 4.9 (minimization property). *Suppose Algorithm* 4.5 *terminates with $\widetilde{t}_{k,i+1}$. Then the iterates $\widetilde{t}_{k,j}$, $j = 1, 2, \ldots, i$, minimize the quadratic function $\widetilde{q}_k$ defined in* (4.8) *over span$\{\widetilde{p}_0, \widetilde{p}_1, \ldots, \widetilde{p}_{j-1}\}$. Moreover,*

$$\widetilde{q}_k(\widetilde{t}_{k,i+1}) < \widetilde{q}_k(\widetilde{t}_{k,i}) < \cdots < \widetilde{q}_k(\widetilde{t}_{k,0}) = 0.$$

Proof. Using the conjugacy property, we find that for each $j \in \{1, 2, \ldots, i\}$

$$\widetilde{q}_k\left(\sum_{\ell=0}^{j-1} \gamma_\ell \widetilde{p}_\ell\right) = \sum_{\ell=0}^{j-1} \left(\frac{\gamma_\ell^2}{2} \langle \widetilde{p}_\ell, H_k \widetilde{p}_\ell \rangle_{\mathcal{X}} + \gamma_\ell \langle \mathcal{W}_k(g_k), \widetilde{p}_\ell \rangle_{\mathcal{X}} \right).$$

Since Algorithm 4.5 does not terminate in iteration $i-1$, and $j \le i$, we have $\langle \widetilde{p}_\ell, H_k \widetilde{p}_\ell \rangle_{\mathcal{X}} > 0$ for $\ell = 0, 1, \ldots, j-1$. Therefore the unique minimizer $(\gamma_0^*, \gamma_1^*, \ldots, \gamma_{j-1}^*)$ of $\widetilde{q}_k$ is given by

$$\gamma_\ell^* = -\frac{\langle \mathcal{W}_k(g_k), \widetilde{p}_\ell \rangle_{\mathcal{X}}}{\langle \widetilde{p}_\ell, H_k \widetilde{p}_\ell \rangle_{\mathcal{X}}} = -\frac{\langle \widetilde{r}_0, \widetilde{p}_\ell \rangle_{\mathcal{X}}}{\langle \widetilde{p}_\ell, H_k \widetilde{p}_\ell \rangle_{\mathcal{X}}},$$

in other words $\sum_{\ell=0}^{j-1} \frac{-\langle \widetilde{r}_0, \widetilde{p}_\ell \rangle_{\mathcal{X}}}{\langle \widetilde{p}_\ell, H_k \widetilde{p}_\ell \rangle_{\mathcal{X}}} \widetilde{p}_\ell$ minimizes $\widetilde{q}_k$ over span$\{\widetilde{p}_0, \widetilde{p}_1, \ldots, \widetilde{p}_{j-1}\}$. Step 1(h) of Algorithm 4.5 gives

$$\widetilde{r}_\ell = \widetilde{r}_0 - \sum_{s=0}^{\ell-1} \frac{\langle \widetilde{r}_s, \widetilde{p}_s \rangle_{\mathcal{X}}}{\langle \widetilde{p}_s, H_k \widetilde{p}_s \rangle \mathcal{X}} H_k \widetilde{p}_s,$$

thus

$$\langle \widetilde{r}_\ell, \widetilde{p}_\ell \rangle = \left\langle \widetilde{r}_0 - \sum_{s=0}^{\ell-1} \frac{\langle \widetilde{r}_s, \widetilde{p}_s \rangle_{\mathcal{X}}}{\langle \widetilde{p}_s, H_k \widetilde{p}_s \rangle_{\mathcal{X}}} H_k \widetilde{p}_s, \widetilde{p}_\ell \right\rangle_{\mathcal{X}} = \langle \widetilde{r}_0, \widetilde{p}_\ell \rangle_{\mathcal{X}},$$

where we have used the conjugacy property. This implies that $\widetilde{t}_{k,j} = \sum_{\ell=0}^{j-1} \widetilde{\alpha}_\ell \widetilde{p}_\ell = \sum_{\ell=0}^{j-1} \frac{-\langle \widetilde{r}_\ell, \widetilde{p}_\ell \rangle_{\mathcal{X}}}{\langle \widetilde{p}_\ell, H_k \widetilde{p}_\ell \rangle_{\mathcal{X}}} \widetilde{p}_\ell$ minimizes $\widetilde{q}_k$ over span$\{\widetilde{p}_0, \widetilde{p}_1, \ldots, \widetilde{p}_{j-1}\}$. Additionally, since $\gamma_j \neq 0$, $j = 0, \ldots, i-1$ (otherwise, Algorithm 4.5 would have terminated earlier),

$$\widetilde{q}_k(\widetilde{t}_{k,i}) < \cdots < \widetilde{q}_k(\widetilde{t}_{k,0}) = 0. \tag{4.9}$$

To complete the proof, we need to consider the last iterate $\widetilde{t}_{k,i+1}$ and show that $\widetilde{q}_k(\widetilde{t}_{k,i+1}) < \widetilde{q}_k(\widetilde{t}_{k,i})$.

If $\widetilde{t}_{k,i+1}$ is returned in step 1(a) or in step 1(d) (of iteration $i+1$), then using the same arguments as before we can show that $\widetilde{t}_{k,i+1}$ minimizes $\widetilde{q}_k$ over span$\{\widetilde{p}_0, \widetilde{p}_1, \ldots, \widetilde{p}_i\}$, hence $\widetilde{q}_k(\widetilde{t}_{k,i+1}) < \widetilde{q}_k(\widetilde{t}_{k,i})$.

If $\widetilde{t}_{k,i+1}$ is generated by step 1(c) or by step 1(g) (of iteration i), we have

$$\widetilde{q}_k(\widetilde{t}_{k,i+1}) = q(\widetilde{t}_{k,i} + \theta\widetilde{p}_i) = q(\widetilde{t}_{k,i}) + \theta\left\langle \mathcal{W}_k(g_k) + H_k\widetilde{t}_{k,i}, \widetilde{p}_i\right\rangle_{\mathcal{X}} + \frac{\theta^2}{2}\left\langle \widetilde{p}_i, H_k\widetilde{p}_i\right\rangle_{\mathcal{X}}$$
$$= \widetilde{q}_k(\widetilde{t}_{k,i}) + \theta\left\langle \widetilde{r}_i, \widetilde{p}_i\right\rangle_{\mathcal{X}} + \frac{\theta^2}{2}\left\langle \widetilde{p}_i, H_k\widetilde{p}_i\right\rangle_{\mathcal{X}}.$$

If $\widetilde{t}_{k,i+1}$ is generated by step 1(c), then $\langle \widetilde{p}_i, H_k\widetilde{p}_i\rangle_{\mathcal{X}} \le 0$ and $\theta\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} < 0$ or, alternatively, $\langle \widetilde{p}_i, H_k\widetilde{p}_i\rangle_{\mathcal{X}} < 0$ and $\theta\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} = 0$. In both cases we obtain $\widetilde{q}_k(\widetilde{t}_{k,i+1}) < \widetilde{q}_k(\widetilde{t}_{k,i})$. Note that since the algorithm did not terminate in the previous iteration in step 1(g), we have $\|n_k + \widetilde{t}_{k,i}\|_{\mathcal{X}} < \Delta_k$ and, consequently, $\theta \neq 0$.

If $\widetilde{t}_{k,i+1}$ is generated by step 1(g), then $\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} < 0$ and $\langle \widetilde{p}_i, H_k\widetilde{p}_i\rangle_{\mathcal{X}} > 0$. The quadratic $\varphi \mapsto \varphi\langle \widetilde{r}_i, \widetilde{p}_i\rangle_{\mathcal{X}} + \frac{\varphi^2}{2}\langle \widetilde{p}_i, H_k\widetilde{p}_i\rangle_{\mathcal{X}}$ strictly decreases as φ goes from 0 to α_i computed in step 1(e). Since $0 < |\theta| \le |\alpha_i|$ and $\operatorname{sign}(\theta) = \operatorname{sign}(\alpha_i)$, this implies $\widetilde{q}_k(\widetilde{t}_{k,i+1}) < \widetilde{q}_k(\widetilde{t}_{k,i})$. Again, note that since the algorithm did not terminate in the previous iteration in step 1(g), we have $\|n_k + \widetilde{t}_{k,i}\| < \Delta_k$, thus $|\theta| > 0$. ∎

In the remainder of this section, we turn to proving the existence of a fixed linear operator $\widetilde{W}_k$ that can replace its nonlinear counterpart $\mathcal{W}_k$ in Algorithm 4.5. As noted in [20, p. 1450], the existence of $\widetilde{W}_k$ is easy if $g_k, \widetilde{r}_1, \ldots, \widetilde{r}_i$ are linearly independent. In [20] the existence of $\widetilde{W}_k$ is tied to conditions on $\mathcal{W}_k$, which is not necessary in our case.

LEMMA 4.10. *Let $\widetilde{r}_j$, $j = 0, \ldots, i$, be the residuals generated by Algorithm 4.5. If $\widetilde{r}_i \neq 0$, then $\widetilde{r}_0, \widetilde{r}_1, \ldots, \widetilde{r}_i$ are linearly independent and g_k, $\widetilde{r}_1, \ldots, \widetilde{r}_i$ are linearly independent.*

Proof. i. Suppose that the vectors $\widetilde{r}_0, \ldots, \widetilde{r}_i$ are linearly dependent. Then there exist coefficients $\{\varrho_j\}_{j=0}^i$ with $\sum_{j=0}^i |\varrho_j| > 0$, i.e., not all ϱ_j can be zero, such that

$$\sum_{j=0}^{i} \varrho_j \widetilde{r}_j = 0.$$

Let ϱ_s be the first nonzero coefficient in $\{\varrho_j\}_{j=0}^i$, i.e., the sum starts at index s. Thus

$$0 = \left\langle \sum_{j=s}^{i} \varrho_j \widetilde{r}_j, \widetilde{p}_s \right\rangle_{\mathcal{X}} = \varrho_s \langle \widetilde{r}_s, \widetilde{p}_s\rangle_{\mathcal{X}},$$

due to Lemma 4.8, i.e., $\langle \widetilde{r}_s, \widetilde{p}_s\rangle_{\mathcal{X}} = 0$. This means that Algorithm 4.5 would have terminated at iteration s in step 1(c) with $\langle \widetilde{r}_s, \widetilde{p}_s\rangle_{\mathcal{X}} = 0$, and the linear combination from above reduces to $\varrho_s\widetilde{r}_s = 0$, which is a contradiction to the assumptions $\varrho_s \neq 0$ and $\widetilde{r}_s \neq 0$.

ii. Suppose that the vectors $g_k, \widetilde{r}_1, \ldots, \widetilde{r}_i$ are linearly dependent. Then there exists a sequence of coefficients $\{\varrho_j\}_{j=0}^i$ with $\sum_{j=0}^i |\varrho_j| > 0$, i.e., not all ϱ_j can be zero, such that

$$\varrho_0 g_k + \sum_{j=1}^{i} \varrho_j \widetilde{r}_j = 0.$$

Again, let ϱ_s be the first nonzero coefficient in $\{\varrho_j\}_{j=0}^i$. If $s > 0$, then we can proceed as in part i to derive a contradiction. If $s = 0$, then due to Lemma 4.8

$$0 = \left\langle \varrho_0 g_k + \sum_{j=1}^{i} \varrho_j \widetilde{r}_j, \widetilde{p}_0 \right\rangle_{\mathcal{X}} = \varrho_0 \langle g_k, \widetilde{p}_0\rangle_{\mathcal{X}} = -\varrho_0 \langle g_k, \mathcal{W}_k(g_k)\rangle_{\mathcal{X}}.$$

Hence $\langle g_k, \mathcal{W}_k(g_k)\rangle_{\mathcal{X}} = 0$ and Algorithm 4.5 would have terminated in step 1(a), which contradicts $\widetilde{r}_0 = \mathcal{W}_k(g_k) \neq 0$. ∎

Since the vectors $\widetilde{r}_j$, $j = 0, 1, \ldots, i$, are linearly independent, the operator $R_i : \mathbb{R}^{i+1} \to \mathcal{X}$, given by

$$R_i = [g_k, \widetilde{r}_1, \ldots, \widetilde{r}_i] \tag{4.10}$$

has full column rank. Furthermore, we introduce matrices $Y_i : \mathbb{R}^{i+1} \to \mathcal{X}$ and $\widetilde{Y}_i : \mathbb{R}^{i+1} \to \mathcal{X}$, given by

$$Y_i = [W_k g_k, W_k \widetilde{r}_1, \ldots, W_k \widetilde{r}_i], \qquad \widetilde{Y}_i = [\mathcal{W}_k(g_k), \mathcal{W}_k(\widetilde{r}_1), \ldots, \mathcal{W}_k(\widetilde{r}_i)]. \tag{4.11}$$

It is easy to verify that the operator

$$\widetilde{W}_k = W_k + (\widetilde{Y}_i - Y_i)(R_i^* R_i)^{-1} R_i^* \tag{4.12}$$

has the property

$$\mathcal{W}_k(g_k) = \widetilde{W}_k g_k, \text{ and } \mathcal{W}_k(\widetilde{r}_j) = \widetilde{W}_k \widetilde{r}_j, \quad j = 1, \ldots, i. \tag{4.13}$$

The linear operator $\widetilde{W}_k$ with the property (4.13) is not unique. In fact if $\widetilde{W}_k$ satisfies (4.13) and if E_k is an operator with $E_k g_k = 0$, $E_k \widetilde{r}_j = 0$, $j = 1, \ldots, i.$, then $\widetilde{W}_k + E_k$ also satisfies (4.13). For example, if the inverse of $\widetilde{Y}_i^{\,*} R_i$ exists, then

$$\widetilde{W}_k = W_k + (\widetilde{Y}_i - Y_i)(\widetilde{Y}_i^{\,*} R_i)^{-1} \widetilde{Y}_i^{\,*} \tag{4.14}$$

also satisfies (4.13). By Algorithm 4.5, $\widetilde{z}_0 = \mathcal{W}_k(g_k)$ and $\widetilde{z}_j = \mathcal{W}_k(\widetilde{r}_j)$, $j \geq 1$. Therefore,

$$(\widetilde{Y}_i^{\,*} R_i)_{\ell,j} = \langle \widetilde{z}_\ell, r_j \rangle \quad \text{ for } 0 \leq \ell \leq j-1,$$

and by Lemma 4.8, the matrix $\widetilde{Y}_i^{\,*} R_i$ is lower triangular. If the orthogonal projections are computed exactly, $\mathcal{W}_k = W_k = W_k^2 = W_k^*$, then $\widetilde{Y}_i^{\,*} R_i$ is a nonsingular diagonal matrix. This property of operator (4.14) will be useful later for the construction of heuristics that improve the performance of our inexact STCG algorithm. For our convergence result, we only need the existence of $\widetilde{W}_k$ with the property (4.13). The operator (4.12), which is always well-defined, has this property.

Having shown the existence of a fixed linear representation of the null–space operator, we can reinterpret Algorithm 4.5 as the standard STCG algorithm with W_k replaced by $\widetilde{W}_k$. Note that even though $W_k = W_k^* = W_k^2$ the linear operator $\widetilde{W}_k$, in general, does not have these properties.

Step 1(b) of Algorithm 4.5 yields $\widetilde{p}_i \in \text{Range}(\widetilde{W}_k)$. Furthermore, $\widetilde{t}_{k,0} = 0 \in \text{Range}(\widetilde{W}_k)$. From step 1(f) we directly obtain

$$\widetilde{t}_{k,i} \in \text{Range}(\widetilde{W}_k)$$

for all i. This implies that Algorithm 4.5 effectively sees problem (4.7) as

$$\min \quad \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}} + \langle \widetilde{W}_k g_k, t\rangle_{\mathcal{X}} \tag{4.15a}$$

$$\text{s.t.} \quad t \in \text{Range}(\widetilde{W}_k), \tag{4.15b}$$

$$\|n_k + t\|_{\mathcal{X}} \leq \Delta_k, \tag{4.15c}$$

i.e., that Algorithm 4.5 is effectively an STCG algorithm applied to (4.15). Problem (4.15) is exactly the problem (3.4) introduced earlier in section 3.2. With this observation we are able to prove the condition (3.7) independent of which linear operator $\widetilde{W}_k$ with the property (4.13) is chosen.

THEOREM 4.11. *Let the assumption* (A5) *be satisfied and let* $\widetilde{W}_k$ *satisfy* (4.13). *If Algorithm* 4.5 *returns* $\widetilde{t}_{k,i}$, $i \geq 0$, *there exist* κ_4, κ_5, $\kappa_6 > 0$, *independent of* k, *such that the fraction of Cauchy decrease condition* (3.7) *holds for* $\widetilde{t}_k = \widetilde{t}_{k,i}$.

Proof. Recall the definition (3.5), $\widetilde{q}_k(t) = \langle \widetilde{W}_k g_k, t\rangle_{\mathcal{X}} + \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}}$, and note $\widetilde{q}_k(0) = 0$.

If Algorithm 4.5 returns $\widetilde{t}_k = \widetilde{t}_{k,0} = 0$, then $\widetilde{r}_0 = \widetilde{W}_k g_k = 0$ and (3.7) is satisfied for any choice of $\kappa_4, \kappa_5, \kappa_6$ (as both sides of the inequality (3.7) are zero).

Otherwise, since $\widetilde{p}_0 = -\widetilde{W}_k g_k$, the first iterate $\widetilde{t}_{k,1}$ generated by Algorithm 4.5 is the solution of

$$\min_{\alpha\in(0,\infty)} \quad \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}} + \langle \widetilde{W}_k g_k, t\rangle_{\mathcal{X}} \tag{4.16a}$$

$$\text{s.t.} \quad t = -\alpha \widetilde{W}_k g_k,\ \|n_k + t\|_{\mathcal{X}} \leq \Delta_k. \tag{4.16b}$$

Furthermore, since

$$\{t : \|t\|_{\mathcal{X}} \leq (1-\zeta)\Delta_k \leq \Delta_k - \|n_k\|_{\mathcal{X}}\} \subset \{t : \|n_k + t\|_{\mathcal{X}} \leq \Delta_k\},$$

the solution $\widetilde{t}$ of

$$\min_{\alpha\in(0,\infty)} \quad \frac{1}{2}\langle H_k t, t\rangle_{\mathcal{X}} + \langle \widetilde{W}_k g_k, t\rangle_{\mathcal{X}} \tag{4.17a}$$

$$\text{s.t.} \quad t = -\alpha \widetilde{W}_k g_k,\ \|t\|_{\mathcal{X}} \leq (1-\zeta)\Delta_k \tag{4.17b}$$

and the solution $\widetilde{t}_{k,1}$ of (4.16) obey

$$\widetilde{q}_k(\widetilde{t}_{k,1}) \leq \widetilde{q}_k(\widetilde{t}).$$

Problem (4.17) is a standard trust-region subproblem (see, e.g., [7, section 6.3.2] or [19, p. 69]). Well-known arguments (see, e.g., [7, Thm. 6.3.1] or [19, Lemma 4.5]) can be used to show that

$$-\widetilde{q}_k(\widetilde{t}) \geq \frac{1}{2}\|\widetilde{W}_k g_k\|_{\mathcal{X}} \min\left\{\frac{\|\widetilde{W}_k g_k\|_{\mathcal{X}}}{\|H_k\|}, (1-\zeta)\Delta_k\right\}.$$

This proves the claim for the case $\widetilde{t}_k = \widetilde{t}_{k,1}$. If Algorithm 4.5 terminates with $\widetilde{t}_{k,i}$, $i \geq 2$, Lemma 4.9 implies

$$-\widetilde{q}_k(\widetilde{t}_{k,i}) > -\widetilde{q}_k(\widetilde{t}_{k,1}) \geq -\widetilde{q}_k(\widetilde{t}) \geq \frac{1}{2}\|\widetilde{W}_k g_k\|_{\mathcal{X}} \min\left\{\frac{\|\widetilde{W}_k g_k\|_{\mathcal{X}}}{\|H_k\|}, (1-\zeta)\Delta_k\right\},$$

which gives (3.7). ∎

4.2.2. Controlling inexactness in the application of $\widetilde{W}_k$. In section 4.2.1 we have shown that if $\mathcal{W}_k$ is an inexact orthogonal projection onto the null space,

applied by iteratively solving augmented linear systems (2.8), then Algorithm 4.5 solves problem (4.15). This is true no matter how accurately the systems (2.8) are solved. In Theorem 4.11 we have also shown that STCG iterates $\widetilde{t}_k$ generated by Algorithm 4.5 satisfy (3.7) for any $\widetilde{W}_k$ with property (4.13). Again, this result does not depend on how accurately the systems (2.8) are solved. However, conditions (3.6) and (3.13) can only be satisfied if the systems (2.8) are solved sufficiently accurately. We will address the accuracy requirements next.

Recall that steps 0 and 1(i) in Algorithm 4.5 each require the solution of a system of the type (2.8). In step 0 of Algorithm 4.5 the inexact projected gradient $\widetilde{r}_0 = \mathcal{W}_k(g_k) = \widetilde{W}_k g_k$ is computed. The iterative linear system solver returns $\widetilde{r}_0$ satisfying

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} \widetilde{r}_0 \\ y \end{pmatrix} = \begin{pmatrix} g_k \\ 0 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}, \tag{4.18}$$

where we allow a nonzero residual $e = (e_1 \;\; e_2) \in \mathcal{X} \times \mathcal{C}$.

THEOREM 4.12. *Let the assumption* (A4) *be satisfied and let* $\widetilde{r}_0 = \widetilde{W}_k g_k$ *be computed as in* (4.18). *If*

$$\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}} \leq \min\left\{\|\widetilde{r}_0\|_{\mathcal{X}}, \Delta_k\right\}, \tag{4.19}$$

then $\widetilde{r}_0$ *satisfies* (3.6).

Proof. Let

$$G_k = \begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} : \mathcal{X} \times \mathcal{C} \to \mathcal{X} \times \mathcal{C}$$

and let $\Phi : \mathcal{X} \to \mathcal{X} \times \mathcal{C}$, $\Psi : \mathcal{X} \times \mathcal{C} \to \mathcal{X}$ be prolongation / restriction operators given by

$$\Phi g_k = \begin{pmatrix} g_k \\ 0 \end{pmatrix}, \qquad \Psi \begin{pmatrix} \widetilde{r}_0 \\ y \end{pmatrix} = \widetilde{r}_0.$$

We obtain

$$\widetilde{r}_0 - W_k g_k = \left(\Psi G_k^{-1} \Phi g_k + \Psi G_k^{-1} e\right) - \Psi G_k^{-1} \Phi g_k = \Psi G_k^{-1} e.$$

Assumption (A4) guarantees the existence of $\nu > 0$ such that $\|G_k^{-1}\|_{L(\mathcal{X}\times\mathcal{C})} \leq \nu$ for all k, a consequence of the boundedness of $c_x(x)$ and $(c_x(x)c_x(x)^*)^{-1}$ and [19, pp. 449–450]. In addition, $\|\Psi\|_{L(\mathcal{X}\times\mathcal{C},\mathcal{X})} = 1$, thus

$$\begin{aligned} \|\widetilde{r}_0 - W_k g_k\|_{\mathcal{X}} &\leq \|\Psi\|_{L(\mathcal{X}\times\mathcal{C},\mathcal{X})} \|G_k^{-1}\|_{L(\mathcal{X}\times\mathcal{C})} \|e\|_{\mathcal{X}\times\mathcal{C}} \leq \nu \left(\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}}\right) \\ &\leq \nu \min\left\{\|\widetilde{r}_0\|_{\mathcal{X}}, \Delta_k\right\}, \end{aligned}$$

which proves the claim. ∎

Remark 4.13. To enforce (4.19) as the stopping condition for a linear solver we compare the size of the residual vector $\left(e_1^\ell \;\; e_2^\ell\right)$ to the size of $\widetilde{r}_0^\ell$, the $\mathcal{X}$-component of the corresponding iterate, at every linear solver iteration ℓ.

LSSC 4.14. *Let $0 < \xi^{pg} \leq 1$ be a selectable parameter. In our implementation, we replace condition* (4.19) *with the stronger stopping condition*

$$\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}} \leq \xi^{pg} \min\left\{\|\widetilde{r}_0\|_{\mathcal{X}}, \Delta_k, \|g_k\|_{\mathcal{X}}\right\}. \tag{4.20}$$

Provided that the iterative solver used for the solution of (4.18) *is initialized with the zero vector, this stopping condition ensures that the iterative solver reduces the residual by at least ξ^{pg}.*

The linear system (4.18) and the residual tolerances described in Theorem 4.12 and LSSC 4.14 specify the computation of $\mathcal{W}_k$ in step 0 of Algorithm 4.5. For iterations $i = 0, 1, \ldots$, we also have to specify residual tolerances for the computation of $\widetilde{z}_i = \mathcal{W}_k(\widetilde{r}_i) = \widetilde{W}_k\widetilde{r}_i$ in step 1(i) of Algorithm 4.5. The iterative linear system solver returns $\widetilde{z}_i$ satisfying

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} \widetilde{z}_i \\ y \end{pmatrix} = \begin{pmatrix} \widetilde{r}_i \\ 0 \end{pmatrix} + \begin{pmatrix} e_{1,i} \\ e_{2,i} \end{pmatrix}, \tag{4.21}$$

where we allow a nonzero residual $e = (e_{1,i} \;\; e_{2,i}) \in \mathcal{X} \times \mathcal{C}$.

LSSC 4.15. *Let $0 < \xi^{proj} \leq 1$. In our implementation, we compute $\widetilde{z}_i = \mathcal{W}_k(\widetilde{r}_i)$, $i = 1, \ldots, i_k$, from* (4.21) *with*

$$\|e_{1,i}\|_{\mathcal{X}} + \|e_{2,i}\|_{\mathcal{C}} \leq \xi^{proj} \min\left\{\|\widetilde{z}_i\|_{\mathcal{X}}, \|\widetilde{r}_i\|_{\mathcal{X}}\right\}, \quad i = 1, \ldots, i_k. \tag{4.22}$$

Provided that the iterative solver used for the solution of (4.21) *is initialized with the zero vector,* (4.22) *ensures that the residual is reduced by at least ξ^{proj}.*

Clearly, the accuracy with which we compute $\widetilde{z}_i = \mathcal{W}_k(\widetilde{r}_i) = \widetilde{W}_k\widetilde{r}_i$ in step 1(i) of Algorithm 4.5, i.e, the choice (4.22) for the stopping tolerance of the system (4.21), impacts the practical performance of the algorithm, but our first-order convergence result is determined by the accuracy with which we solve (4.18) in step 0 of Algorithm 4.5. This was described completely in Theorem 4.12 and LSSC 4.14. In Appendix B we will show, under suitable conditions, that if Algorithm 4.5 terminates after $i_k + 1$ iterations there exists a constant $C > 0$ independent of k such that for $\widetilde{W}_k$ defined in (4.14)

$$\|\widetilde{W}_k - W_k\|_{\mathcal{X}} \leq C(i_k + 1) \max\left\{\xi^{pg}, \xi^{proj}\right\}. \tag{4.23}$$

This bound is related to an optional stopping criterion that improves the performance of our inexact STCG algorithm, but requires the invertibility of $\widetilde{Y}_i^* R_i$. The latter can be checked numerically assuming that the maximum number of STCG iterations is not too large, e.g., $\mathcal{O}(100)$. The precise statement is given in Theorem B.1. We emphasize here that this result is not needed for the proof of first-order global convergence.

To summarize, the iterative linear system solves in steps 0 and 1(i) of Algorithm 4.5, which are required to compute $\widetilde{r}_0 = \mathcal{W}_k(g_k)$ and and $\widetilde{z}_i = \mathcal{W}_k(\widetilde{r}_i)$, respectively, are performed using an iterative solver so that the residuals in (4.18) and (4.21) satisfy (4.20) and (4.22), respectively. As a result, condition (3.6) is satisfied.

4.3. Computation of the tangential step. Once the approximate solution $\widetilde{t}_k$ of the tangential subproblem (3.4) has been obtained, the tangential step t_k is computed subject to conditions (3.10), (3.12), and (3.13).

Condition (3.12) is very similar to condition (3.6) and entails another inexact null space projection,

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} t_k \\ y \end{pmatrix} = \begin{pmatrix} \widetilde{t}_k \\ 0 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}, \tag{4.24}$$

where we allow a nonzero residual $e = (e_1 \;\; e_2) \in \mathcal{X} \times \mathcal{C}$.

LEMMA 4.16. *Let the assumption* (A4) *be satisfied and let t_k be computed as in* (4.24). *If*

$$\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}} \leq \Delta_k \min\{\Delta_k, \|n_k + t_k\|_{\mathcal{X}}\}, \tag{4.25}$$

then t_k satisfies (3.12).

The proof of Lemma 4.16 is analogous to the proof of Theorem 4.12.

Instead of (4.25) we implement the following stronger condition.

LSSC 4.17. *Let $0 < \xi^{tang} \leq 1$. In our implementation we replace condition* (4.25) *with the more stringent stopping condition*

$$\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}} \leq \Delta_k \min\left\{\Delta_k, \|n_k + t_k\|_{\mathcal{X}}, \xi^{tang}\|\widetilde{t}_k\|_{\mathcal{X}}/\Delta_k\right\}. \tag{4.26}$$

Provided that the iterative solver used for the solution of (4.24) *is initialized with the zero vector, this stopping condition ensures that the residual is reduced by at least ξ^{tang}.*

Next we discuss the enforcement of condition (3.10). If the new Lagrange multiplier λ_{k+1} is known, then $pred(n_k, \widetilde{t}_k; \rho_k)$ given by (3.8) is a fixed constant, and $rpred(r_k^t; \rho_k)$ defined in (3.9) depends on the size of $r_k^t \stackrel{\text{def}}{=} c_x(x_k)t_k$. Since t_k satisfies (4.24), i.e., is an inexact projection of $\widetilde{t}_k$ into the null space of $c_x(x_k)$, the size of r_k^t is determined by the size of $\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}}$. In particular the condition (3.10) can be satisfied by making ξ^{tang} in (4.26) sufficiently small. This is what is done in our implementation. We fix ξ^{tang} and solve (4.24) with the stopping criterion (4.26). If the condition (3.10) is not satisfied, we reduce ξ^{tang} and re-solve (4.24) with the refined stopping criteria. In practice, we restart the iterative solver so as not to waste the work performed in the previous solve of (4.24). The previous considerations apply when the new Lagrange multiplier λ_{k+1} is known. The basic convergence result Theorem 3.5 only requires that the sequence $\{\lambda_k\}_{k\in\mathbb{N}}$ be bounded and therefore allows great flexibility in the choice of Lagrange multiplier estimates. However, for the actual performance of the algorithm, λ_{k+1} is computed at the trial iterate $x_k + n_k + t_k$ as discussed in section 4.4 below. Therefore, whenever we refine ξ^{tang} in the stopping tolerance (4.26), i.e., change t_k, we recompute λ_{k+1}. The loop in step 1(a) of Algorithm 4.18 enforces the condition (3.10).

It remains to discuss how we enforce condition (3.13). If n_k is orthogonal to the subspace and if $t_k = \widetilde{t}_k$ is in the null space of the constraints, then $s_k = n_k + t_k = n_k + \widetilde{t}_k$, so $\|\widetilde{t}_k\|_{\mathcal{X}} \leq \|s_k\|_{\mathcal{X}}$. Therefore, in the exact case, (3.13) is satisfied. In the inexact case, it could happen that with coarse solutions of the augmented systems the substep $\widetilde{t}_k$ is relatively far away from the null space, and condition (3.13) is violated. Therefore, we fix ξ_4 and monitor condition (3.13). If it is violated, then we further tighten the accuracy conditions used to compute $\widetilde{t}_k$ and t_k. This is done in the loop in step 1 of Algorithm 4.18 below. This can be expensive, but in our experiments such recomputations are rarely needed. We will comment more on this below.

In Algorithm 4.18, $\eta_0 \in (0, 1)$ and $\xi_4 > \sqrt{2}$ are the parameters in (3.10) and (3.13), respectively. The parameters $\xi^{qn}, \xi^{pg}, \xi^{proj}, \xi^{tang} \in (0, 1)$ are the forcing parameters

used in (4.5), (4.20), (4.22), (4.26) to control the reduction of the relative residuals. The vector n_k is the quasi-normal step as returned by Algorithm 4.1. The vectors $\widetilde{t}_k$ and $\widetilde{t}_k^{cp}$ are the solution of the tangential subproblem and the tangential Cauchy point, respectively, as returned by Algorithm 4.5.

ALGORITHM 4.18. COMPUTATION OF THE STEP s_k.

0. Given $\eta_0 \in (0,1)$, penalty parameter $\rho_{k-1} > 0$, forcing parameters ξ^{qn}, ξ^{pg}, ξ^{proj}, $\xi^{tang} \in (0,1)$ and $\xi_4 > \sqrt{2}$.
   Compute n_k using Algorithm 4.1 with linear solver tolerance (4.5), and compute $\widetilde{t}_k$, $\widetilde{t}_k^{cp}$ using Algorithm 4.5 with linear solver tolerances (4.20), (4.22).
1. For $i = 0, 1, \ldots$
   (a) For $j = 0, 1, \ldots$
      i. Compute t_k by solving (4.24) with the stopping condition (4.26).
      ii. Compute Lagrange multiplier estimate λ_{k+1} at $x_k + n_k + t_k$.
      iii. Compute penalty parameter ρ_k as in Algorithm 3.3.
      iv. If $|rpred(c_x(x_k)t_k; \rho_k)| > \eta_0\, pred(n_k, \widetilde{t}_k; \rho_k)$, set $\xi^{tang} \to 10^{-1}\, \xi^{tang}$.
   End For
   Reset ξ^{tang} to its value at outer iteration i prior to step 1(a).
   (b) If $\|\widetilde{t}_k\|_{\mathcal{X}} > \xi_4 \|n_k + t_k\|_{\mathcal{X}}$ and $\widetilde{t}_k = \widetilde{t}_k^{cp}$
      Set $\xi^{qn} \to 10^{-1}\, \xi^{qn}$, $\xi^{pg} \to 10^{-1} \xi^{pg}$, $\xi^{proj} \to 10^{-1} \xi^{proj}$, $\xi^{tang} \to 10^{-1}\, \xi^{tang}$. Recompute n_k using Algorithm 4.1 with linear solver tolerance (4.5), and recompute $\widetilde{t}_k$, $\widetilde{t}_k^{cp}$ using Algorithm 4.5 with linear solver tolerances (4.20), (4.22). Recompute n_k, $\widetilde{t}_k$, and $\widetilde{t}_k^{cp}$.
   Else If $\|\widetilde{t}_k\|_{\mathcal{X}} > \xi_4 \|n_k + t_k\|_{\mathcal{X}}$ and $\widetilde{t}_k \neq \widetilde{t}_k^{cp}$
      Set $\widetilde{t}_k \to \widetilde{t}_k^{cp}$.
   Else
      Go to step 2.
   End For
2. Optional: Reset ξ^{qn}, ξ^{pg}, ξ^{proj}, and ξ^{tang} to their values from step 0.
   Return t_k.

Remark 4.19. i. All our numerical examples indicate that setting $\xi_4 = 2$ is sufficient for Algorithm 4.18 to terminate at iteration $i = 0$ or, less frequently, iteration $i = 1$. Therefore, the outer loop in step 1 of Algorithm 4.18 should be interpreted as a safeguard that is rarely activated in practice.

ii. The constant reduction factor 10^{-1} in Algorithm 4.18 can be replaced by more elaborate choices as long as these reduction factors are bounded away from one.

THEOREM 4.20. *Let the assumptions* (A4) *and* (A7) *be satisfied. The tangential step t_k generated by Algorithm* 4.18 *satisfies conditions* (3.10), (3.12), *and* (3.13).

Proof. Condition (3.12) is trivially satisfied due to step 1(ai) and Lemma 4.16. For a given outer iteration i, after a finite number of reductions of ξ^{tang} we have $\xi^{tang}\|\widetilde{t}_k\|_{\mathcal{X}} \leq -\sigma_k + \sqrt{\sigma_k^2 + \eta_0\, pred(n_k, \widetilde{t}_k; \rho_k)/\rho_k}$ which implies

$$\|c_x(x_k)t_k\| \leq \|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}} \leq -\sigma_k + \sqrt{\sigma_k^2 + \eta_0\, pred(n_k, \widetilde{t}_k; \rho_k)/\rho_k}$$

(see (4.26)) and, by (3.11), condition (3.10) is also satisfied.

It remains to show that after a finite number of iterations i, Algorithm 4.18 generates n_k and t_k such that (3.13) is satisfied. Note that if (3.13) is not satisfied step 1(b) enforces $\widetilde{t}_k = \widetilde{t}_k^{cp}$ and in the subsequent iterate i reduces ξ^{qn}, ξ^{pg}, ξ^{proj}, and ξ^{tang} by a constant factor. Therefore, it is sufficient to show that (3.13) is satisfied if

$\widetilde{t}_k = \widetilde{t}_k^{cp}$ and ξ^{qn}, ξ^{pg}, ξ^{proj}, and ξ^{tang} are sufficiently small. For the rest of the proof we therefore assume $\widetilde{t}_k = \widetilde{t}_k^{cp}$.

First we will show that

$$\frac{\xi_4}{\sqrt{2}} \|t_k\|_{\mathcal{X}} \geq \|\widetilde{t}_k\|_{\mathcal{X}} \tag{4.27}$$

for sufficiently small ξ^{pg} and ξ^{tang}. Note that the Cauchy point $\widetilde{t}_k^{cp}$ returned by Algorithm 4.5 satisfies $\widetilde{t}_k^{cp} = \gamma \widetilde{r}_0$, where $\gamma \neq 0$ depends on the termination scenario encountered by Algorithm 4.5 and where $\widetilde{r}_0 = \widetilde{W}_k g_k$ satisfies

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} \widetilde{r}_0 \\ \bar{y} \end{pmatrix} = \begin{pmatrix} g_k \\ 0 \end{pmatrix} + \begin{pmatrix} \bar{e}_1 \\ \bar{e}_2 \end{pmatrix}$$

with $\|\bar{e}_1\|_{\mathcal{X}} + \|\bar{e}_2\|_{\mathcal{C}} \leq \xi^{pg} \|\widetilde{r}_0\|_{\mathcal{X}}$; see (4.20). In particular, we have $\|c_x(x_k)\widetilde{r}_0\|_{\mathcal{X}} \leq \xi^{pg}\|\widetilde{r}_0\|_{\mathcal{X}}$, which due to $\widetilde{t}_k^{cp} = \gamma \widetilde{r}_0$ implies

$$\|c_x(x_k)\widetilde{t}_k^{cp}\|_{\mathcal{X}} \leq \xi^{pg} \|\widetilde{t}_k^{cp}\|_{\mathcal{X}}. \tag{4.28}$$

We note that this result is independent of the value of γ. From

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} t_k \\ y \end{pmatrix} = \begin{pmatrix} \widetilde{t}_k^{cp} \\ 0 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

with $\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}} \leq \xi^{tang} \|\widetilde{t}_k^{cp}\|_{\mathcal{X}}$ (see (4.26)), we obtain

$$t_k - \widetilde{t}_k^{cp} = -c_x(x_k)^* (c_x(x_k) c_x(x_k)^*)^{-1} \left(-e_2 + c_x(x_k)\widetilde{t}_k^{cp} + c_x(x_k) e_1 \right) + e_1.$$

Combining this result with (4.28) and the assumption (A4) implies that there exists a constant C independent of k such that

$$\|t_k - \widetilde{t}_k^{cp}\|_{\mathcal{X}} \leq C(\xi^{pg} + \xi^{tang}) \|\widetilde{t}_k^{cp}\|_{\mathcal{X}}.$$

Using the inequality $\|\widetilde{t}_k^{cp}\|_{\mathcal{X}} \leq \|\widetilde{t}_k^{cp} - t_k\|_{\mathcal{X}} + \|t_k\|_{\mathcal{X}}$ this implies

$$\left(1 - C(\xi^{pg} + \xi^{tang})\right) \|\widetilde{t}_k^{cp}\|_{\mathcal{X}} \leq \|t_k\|_{\mathcal{X}}.$$

Hence, if ξ^{pg} and ξ^{tang} are small enough such that $C(\xi^{pg} + \xi^{tang}) < 1$ and

$$\frac{\xi_4}{\sqrt{2}} > \frac{1}{\left(1 - C(\xi^{pg} + \xi^{tang})\right)^2},$$

then (4.27) holds.

Let n_k^* be the quasi-normal step computed with the exact null-space projector W_k and let t_k^* solve (4.24) with $e_1 = 0$ and $e_2 = 0$. In this case $n_k^* = n_k^{cp} + \theta_k \delta n_k^*$, where $n_k^{cp} \in \text{Range}(c_x(x_k)^*)$ and, since δn_k^* solves (4.3) with $r_k^1 = 0, r_k^2 = 0$, $\delta n_k^* \in \text{Range}(c_x(x_k)^*)$. Consequently $n_k^* = n_k^{cp} + \theta_k \delta n_k^* \in \text{Range}(c_x(x_k)^*)$ and $W_k n_k^* = 0$, Furthermore, $t_k^* = W_k \widetilde{t}_k$. This implies $\langle n_k^*, t_k^* \rangle_{\mathcal{X}} = 0$.

Since $n_k \to n_k^*$ and $t_k \to t_k^*$ as $\xi^{qn}, \xi^{tang} \to 0$, it holds that $2|\langle n_k, t_k \rangle_{\mathcal{X}}| \leq \|n_k\|_{\mathcal{X}} \|t_k\|_{\mathcal{X}}$ for sufficiently small ξ^{qn}, ξ^{tang}. Hence,

$$\begin{aligned} \xi_4^2 \|n_k + t_k\|_{\mathcal{X}}^2 &= \xi_4^2 (\|n_k\|_{\mathcal{X}}^2 + \|t_k\|_{\mathcal{X}}^2 + 2\langle n_k, t_k \rangle_{\mathcal{X}}) \\ &\geq \xi_4^2 (\|n_k\|_{\mathcal{X}}^2 + \|t_k\|_{\mathcal{X}}^2 - \|n_k\|_{\mathcal{X}} \|t_k\|_{\mathcal{X}}) \\ &= \xi_4^2 \left(\frac{1}{2}\|n_k\|_{\mathcal{X}}^2 + \frac{1}{2}\|t_k\|_{\mathcal{X}}^2 + \frac{1}{2}(\|n_k\|_{\mathcal{X}} - \|t_k\|_{\mathcal{X}})^2 \right) \geq \frac{\xi_4^2}{2} \|t_k\|_{\mathcal{X}}^2. \end{aligned}$$

Combining this with (4.27) gives the desired inequality (3.13). ∎

4.4. Lagrange multiplier computation. If linear systems are solved exactly, the Lagrange multiplier estimate λ_k can be computed by solving the linear system

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} z \\ \lambda_k \end{pmatrix} = \begin{pmatrix} -\nabla_x f(x_k) \\ 0 \end{pmatrix}. \tag{4.29}$$

If linear systems are solved inexactly, i.e., involve a nonzero system residual, it is advantageous to relate the norm of the residual to the norm of the gradient of the Lagrangian. Therefore, given a previous Lagrange multiplier estimate λ_{k-1}, the new multipliers λ_k can be computed via $\lambda_k = \lambda_{k-1} + \Delta\lambda$, where $\Delta\lambda$ satisfies

$$\begin{pmatrix} I & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} z \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} -\nabla_x f(x_k) - c_x(x_k)^*\lambda_{k-1} + e_1 \\ e_2 \end{pmatrix}. \tag{4.30}$$

Note that (4.30) and (4.29) are equivalent if $(e_1 \;\; e_2) = 0$, where $(e_1 \;\; e_2) \in \mathcal{X} \times \mathcal{C}$.

Global convergence proofs for SQP methods require boundedness of the Lagrange multipliers, as stated in the assumption (A6). This requirement is easily verified.

LEMMA 4.21. *Let the assumption* (A4) *be satisfied. Let the sequence* $\{\lambda_k\}$ *of Lagrange multipliers be generated by* $\lambda_k = \lambda_{k-1} + \Delta\lambda$*, where* $\Delta\lambda$ *satisfies* (4.30) *with*

$$\|e_1\|_{\mathcal{X}} + \|e_2\|_{\mathcal{C}} \leq \xi^{lmg} \tag{4.31}$$

for a fixed $\xi^{lmg} > 0$ *independent of* k*. Then the sequence* $\{\lambda_k\}$ *is bounded.*

Proof. From (4.30) we directly obtain

$$\begin{aligned} \lambda_k &= \left(c_x(x_k)c_x(x_k)^*\right)^{-1}\left(-c_x(x_k)\nabla_x f(x_k) + c_x(x_k)e_1 - e_2\right) \\ &= -\left(c_x(x_k)c_x(x_k)^*\right)^{-1} c_x(x_k)\nabla_x f(x_k) + \left(c_x(x_k)c_x(x_k)^*\right)^{-1}\left(c_x(x_k)e_1 - e_2\right). \end{aligned}$$

General assumptions (A4) give uniform boundedness of the quantities $\|c_x(x)\|_{L(\mathcal{X},\mathcal{C})}$, $\|\left(c_x(x)c_x(x)^*\right)^{-1}\|_{L(\mathcal{C})}$, and $\|\nabla_x f(x)\|_{\mathcal{X}}$ for all x. The claim follows from elementary norm inequalities and the boundedness of the residual norms $\|e_1\|_{\mathcal{X}}$ and $\|e_2\|_{\mathcal{C}}$. ∎

LSSC 4.22. *Let* $0 < \xi^{lmh} \leq 1$ *be a selectable parameter. In our implementation we replace condition* (4.31) *with the stopping condition*

$$\|r_1\|_{\mathcal{X}} + \|r_2\|_{\mathcal{C}} \leq \min\left\{\xi^{lmg}, \xi^{lmh}\|\nabla_x f(x_k) + c_x(x_k)^*\lambda_{k-1}\|_{\mathcal{X}}\right\}. \tag{4.32}$$

Provided that the iterative solver used for the solution of (4.30) *is initialized with the zero vector,* (4.32) *ensures that the residual is reduced by at least* ξ^{lmh}*.*

5. Numerical results. In this section we examine the performance of our trust-region SQP algorithm with inexact linear system solves, Algorithm 3.3, on a PDE-constrained optimization problem. Additional numerical results are given in [15, 23]. In particular, the report [23] contains a compact statement of Algorithm 3.3 and numerical studies of its application to various PDE-constrained optimization problems. The numerical results presented in this section are taken from [23], where further details of the problem discretization, linear systems solvers, and results for that particular example, and others, can be found.

As discussed earlier, our trust-region SQP algorithm requires the solution of augmented systems of the form, e.g., (4.3) or (4.24), which are necessary and sufficient optimality conditions for convex quadratic programs (QPs) with QP Hessians given by identities. Hence, all augmented system solvers can be used. In the numerical

example of this section the augmented systems are solved using a left-preconditioned GMRES algorithm with an incomplete-LU preconditioner. Since for a given SQP iteration the matrix is the same in all augmented systems, the cost of setting up the preconditioner can be amortized over several linear system solves.

To examine the benefits of inexactness control in Algorithm 3.3, we compare it to a closely related conventional trust-region SQP algorithm (sketched in section 2), in which the direct solves of all augmented systems are replaced by preconditioned GMRES solves with a fixed relative tolerance. This is what one may do when forced to retrofit an existing NLP code with iterative linear system solves. We note that in addition to the static treatment of linear solver tolerances, the conventional SQP algorithm contains none of the global convergence safeguards of Algorithm 3.3 to deal with inexactness.

For brevity, we use the label I-SQP for our algorithm with inexactness control, Algorithm 3.3, and the label C-SQP for the conventional algorithm. The algorithm parameters shared between I-SQP and C-SQP are

tol^{SQP}	tol^{CG}	ζ	Δ_0	Δ_{min}	Δ_{max}	α_1	η_1	η_0	ρ_{-1}	$\bar{\rho}$
10^{-6}	10^{-2}	0.8	10^2	10^{-10}	10^8	0.5	10^{-8}	0.5	1	10^{-8}

We introduce a *nominal* linear solver tolerance, tol^{LS}, which is varied in our experiments. Given tol^{LS}, the remaining I-SQP parameters are set to $\xi^{qn} = \xi^{pg} = \xi^{proj} = \xi^{tang} = \xi^{lmh} = tol^{LS}$, $\xi_4 = 2$, and $\xi^{lmg} = 10^4$. To improve the efficiency of inexact STCG, Algorithm 4.5, we use an additional termination criterion, developed in Appendix B. The fixed relative solver tolerance used in C-SQP is set to tol^{LS}.

Our example problem is the optimal boundary control of the Navier–Stokes equations to minimize vorticity behind a backward facing step, and is modeled after [17].

Let $\Omega \in \mathbb{R}^2$ be the channel with a backward facing step shown in Figure 2. We consider an optimal control problem governed by the Navier–Stokes equations formally stated as follows:

$$\underset{\mathbf{u},\mathbf{g}}{\text{minimize}} \quad \frac{1}{2}\int_D (\partial_{x_1}\mathbf{u}_2 - \partial_{x_2}\mathbf{u}_1)^2 d\mathbf{x} + \frac{\alpha}{2}\int_{\Gamma_c} |\mathbf{g}|^2 d\mathbf{s},$$

subject to the Navier–Stokes equations

$$\begin{aligned} -\nu\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p = \mathbf{0} \quad &\text{in} \quad \Omega, \\ \nabla\cdot\mathbf{u} = 0 \quad &\text{in} \quad \Omega, \end{aligned}$$

with boundary conditions

$$(\nu\nabla\mathbf{u} - pI)\mathbf{n} + \frac{1}{\delta}\mathbf{u} = \frac{1}{\delta}\mathbf{g} \quad \text{on } \Gamma_c, \tag{5.1}$$

$(\nu\nabla\mathbf{u} - pI)\mathbf{n} = \mathbf{0}$ on Γ_{out}, $\mathbf{u} = \mathbf{b}$ on Γ_{in}, and $\mathbf{u} = \mathbf{0}$ on $\partial\Omega\backslash(\Gamma_{in}\cup\Gamma_c\cup\Gamma_{out})$. Here $\mathbf{u}$ is the flow velocity, p is the pressure, $\mathbf{g}$ is the control. The viscosity is $\nu = 5\times 10^{-3}$

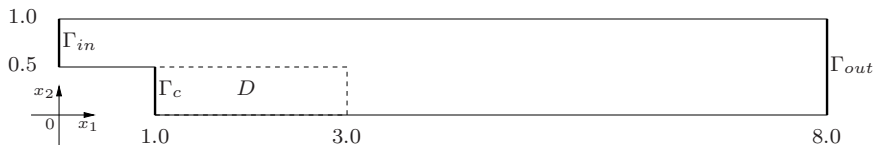

FIG. 2. *Geometry of the backward facing step channel.*

TABLE 1

Inexact SQP algorithm (I-SQP). Note: all numbers are totals across the entire SQP run; prec builds *denotes the number of times the incomplete-LU preconditioner for augmented systems is computed;* a posteriori refine *denotes the number of times the safeguard* $\|\widetilde{t}_k\|_{\mathcal{X}} > \xi_4\|n_k + t_k\|_{\mathcal{X}}$ *is activated in Algorithm* 4.18*;* nonconvex QP *denotes the number of times zero or negative curvature is encountered in STCG, Algorithm* 4.5*. I-SQP handles iterative linear system solves very robustly.*

Nominal Tol tol^{LS}		0.5	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
(S)uccess/(F)ailure		S	S	S	S	S	S	S	S	S
GMRES	iters	4703	3186	2493	2352	2250	2600	2924	3304	3603
GMRES	calls	690	474	287	193	176	176	176	176	176
GMRES	iters/call	6.8	6.7	8.7	12.2	12.8	14.8	16.6	18.8	20.5
GMRES	prec builds	47	34	21	11	9	9	9	9	9
STCG iters		570	389	231	156	143	143	143	143	143
SQP iters		38	28	16	9	8	8	8	8	8
a posteriori refine		1	0	0	0	0	0	0	0	0
nonconvex QP		0	0	2	0	0	0	0	0	0

and the penalty parameters $\alpha = 10^{-1}$ and $\delta = 10^{-5}$ are given. By $\mathbf{n}$ we denote the unit outward normal; $\mathbf{b}$ is a given function. We only apply suction and blowing in the wall-normal direction; that is, $\mathbf{g} = g\mathbf{n}$. The boundary condition (5.1) is a penalized form of the Dirichlet condition $\mathbf{u} = \mathbf{g}$ on Γ_c.

At the inflow boundary $\Gamma_{in} = \{0\} \times [0.5, 1]$ we assume that the flow is a parabolic function with $\mathbf{b}(x_1, x_2) = 8(x_2 - 0.5)(1 - x_2)$. The control is applied on the boundary $\Gamma_c = \{1\} \times [0, 0.5]$. The computational domain is divided into 352 triangles with a finer mesh across the area in which recirculation occurs. We discretize the Navier–Stokes equations using Taylor–Hood finite elements.

Table 1 documents the performance of the I-SQP algorithm as the nominal linear solver tolerance tol^{LS} is increased from 10^{-8} to 0.5. The I-SQP algorithm is extremely robust with respect to variation of the nominal tolerance tol^{LS}; it converges in all cases. As reported in Figure 3, the algorithm dynamically chooses linear solver tolerances as it makes progress toward a solution. The nominal tolerance tol^{LS} can

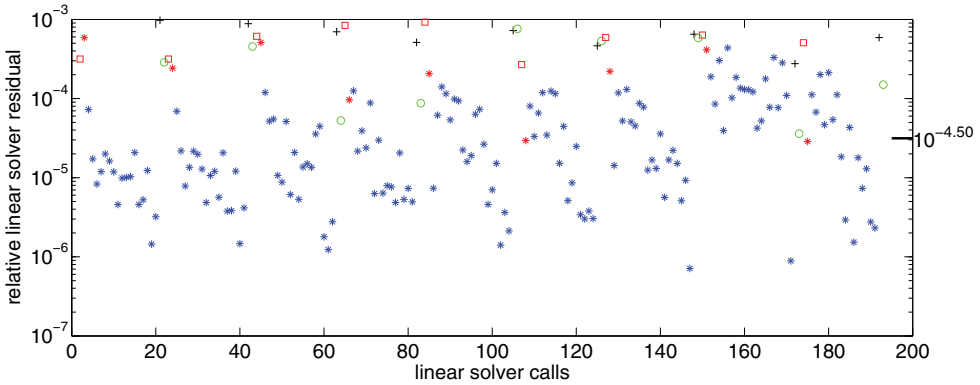

FIG. 3. *Relative residuals for all augmented systems solved by the I-SQP algorithm with* $tol^{LS} = 10^{-3}$*. The red squares denote quasi-normal step computations (LSSC* 4.3*); the red asterisks correspond to the first STCG iterations (LSSC* 4.14*); the blue asterisks denote the subsequent STCG iterations (LSSC* 4.15*); the black crosses correspond to tangential step computations (LSSC* 4.17*); the green circles denote Lagrange multiplier computations (LSSC* 4.22*). The median (log scale) residual is marked by the bold line on the right vertical axis. I-SQP allows us to trade the accuracy of linear system solves for optimization iterations and vice versa.*

TABLE 2
Conventional SQP algorithm (C-SQP). Failure is reported if the number of SQP iterations exceeds 100. *Lack of robustness is evident.*

Nominal Tol tol^{LS}	0.5	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
(S)uccess/(F)ailure	F	F	F	F	F	F	F	S	S
GMRES iters	—	—	—	—	—	—	—	2906	3180
SQP iters	—	—	—	—	—	—	—	8	8

be interpreted as an upper bound on the relative augmented system residuals computed by I-SQP. At the same time, the median residual is not significantly smaller than tol^{LS}.

In contrast, as shown in Table 2, C-SQP only converges if the nominal linear solver tolerance is small. We terminate the optimization after 100 iterations. It is not clear how to choose the nominal linear solver tolerance *a priori*. In other examples, documented in [23], even a fine nominal linear solver tolerance of $tol^{LS} = 10^{-8}$ does not lead to convergence of C-SQP.

For our numerical tests, the total number of GMRES iterations is a good measure of the overall efficiency of the algorithm. In this example, for the nominal tolerances of $tol^{LS} = 10^{-7}$ and $tol^{LS} = 10^{-8}$, C-SQP required fewer total GMRES iterations than I-SQP. This is due, e.g., to the computation of t_k, which is not necessary for C-SQP. However, the fine tolerances $tol^{LS} = 10^{-7}$ and $tol^{LS} = 10^{-8}$ are not needed in I-SQP, and nominal tolerances of $tol^{LS} = 10^{-2}$ to $tol^{LS} = 10^{-5}$ result in fewer GMRES iterations compared to C-SQP with $tol^{LS} = 10^{-7}$. Moreover, as mentioned earlier, it is impossible to determine *a priori* whether $tol^{LS} = 10^{-7}$ leads to convergence of C-SQP.

Table 1 shows that the number of I-SQP iterations decreases as the nominal tolerance tol^{LS} is decreased. However, smaller tol^{LS} values imply a larger computational cost per iteration, shown in the row GMRES iters/call, the average number of GMRES iterations per augmented system solve. Therefore, I-SQP enables tradeoffs between the accuracy of linear system solves and the number of optimization iterations. In this example a nominal tolerance of $tol^{LS} = 10^{-2}$ or $tol^{LS} = 10^{-3}$ is an excellent choice. Additional numerical experiments in [23] show that a nominal tolerance between $tol^{LS} = 10^{-2}$ and $tol^{LS} = 10^{-4}$ is a good choice, in general.

Table 1 shows that only for the very coarse nominal tolerance of $tol^{LS} = 0.5$ is the condition $\|\widetilde{t}_k\|_{\mathcal{X}} > \xi_4 \|n_k + t_k\|_{\mathcal{X}}$ violated, in a single I-SQP iteration. In this one case step 1 of Algorithm 4.18 terminates with $i = 1$; in all other cases, $i = 0$. Hence, there are virtually no recomputations.

Finally, we note that the inexactness in linear system solves can cause nonconvexity in quadratic subproblems, as evidenced by the last row in Table 1, $tol^{LS} = 10^{-2}$. Our algorithm remains robust and efficient without the need for Hessian modifications.

Our report [23] contains detailed descriptions of applications of Algorithm 3.3 to optimal control, optimal design, and inverse problems with various governing equations, including Burgers, scalar Ginzburg–Landau, Poisson–Boltzmann, and linear elastodynamics equations. The observations made for the example presented in this paper hold for all numerical examples documented in [23].

6. Conclusion. We have developed and analyzed a full-space composite-step trust-region SQP algorithm for the solution of large-scale equality constrained optimization problems that allows the inexact and therefore iterative solution of the linear systems arising in the step and Lagrange multiplier computations. If exact

linear system solves are performed, the algorithm fits into the framework presented in [8]. To handle inexact linear systems solves, several algorithmic modifications are needed. These are motivated by [16], but in this paper we use a full-space approach, which provides more flexibility in the choice of linear system solvers and preconditioners, and in particular allows the use of recently proposed, efficient KKT solvers. A significant contribution of this paper is the detailed specification and analysis of algorithms for substep computations that are necessary for an efficient implementation of our SQP algorithm. The resulting algorithm accommodates user-specified parameters to tune its efficiency; however, the algorithm does not depend on problem-specific parameters, such as Lipschitz constants, that cannot be computed and are even expensive to estimate. We have proved first-order global convergence of the algorithm for a wide range of user-supplied linear solver parameters. In practice the algorithm is remarkably robust to large amounts of inexactness in linear system solves.

We are currently investigating extensions of our approach to large-scale finite-dimensional problems with inequality constraints, by using the ideas from [6].

Appendix A. Proof of global convergence of the inexact trust-region SQP algorithm.

Our proof of Theorem 3.5 is based on the global convergence analysis given in [8] and follows the modifications given in [16].

Proof. There are three modifications to the convergence analysis in [8].

i. Relation between the size of the trial step s_k to the trust-region radius Δ_k. First, we need to relate the size of the trial step s_k to the trust-region radius Δ_k. In particular we need the existence[1] of $\kappa_7, \kappa_8 > 0$ independent of k such that

$$\|s_k\|_{\mathcal{X}} \le \kappa_7 \Delta_k, \tag{A.1}$$

and, if s_k is rejected,

$$\|s_k\|_{\mathcal{X}} \ge \kappa_8 \Delta_{k+1}.$$

The first inequality follows from $s_k = n_k + t_k$, (2.3b), (3.12), $\|W_k\| = 1$, and

$$\begin{aligned}\|t_k\|_{\mathcal{X}} &\le \|W_k \widetilde{t}_k\|_{\mathcal{X}} + \|t_k - W_k \widetilde{t}_k\|_{\mathcal{X}} \le \|\widetilde{t}_k + n_k\|_{\mathcal{X}} + \|n_k\|_{\mathcal{X}} + \|t_k - W_k \widetilde{t}_k\|_{\mathcal{X}} \\ &\le (1 + \zeta + \xi_3 \Delta_{\max}) \Delta_k.\end{aligned}$$

The second inequality follows directly from the trust region update in step 2(cvi) of Algorithm 3.3.

ii. Error between the actual and the predicted reduction. The second modification concerns the estimates of the actual and the predicted reduction. Analogously to [16] we will be able to show

$$\begin{aligned}&|ared(s_k; \rho_k) - pred(n_k, \widetilde{t}_k; \rho_k) - rpred(r_k^t; \rho_k)| \\ &\quad \le \kappa_9 \Delta_k \|s_k\|_{\mathcal{X}} + \kappa_{10} \rho_k \|s_k\|_{\mathcal{X}}^3 + \kappa_{11} \rho_k \|s_k\|_{\mathcal{X}}^2 \|c(x_k)\|\end{aligned} \tag{A.2}$$

instead of [8, Lemma 7.4] and

$$|ared(s_k; \rho_k) - pred(n_k, \widetilde{t}_k; \rho_k) - rpred(r_k^t; \rho_k)| \le \kappa_{12} \rho_k \Delta_k \|s_k\|_{\mathcal{X}} \tag{A.3}$$

[1] The naming of the constants in this proof is chosen such that it corresponds to the convergence proof in [16].

instead of [8, Lemma 7.5]. In fact, (A.3) follows immediately from (A.2) since $\|s_k\|_{\mathcal{X}} \leq \kappa_7 \Delta_k \leq \kappa_7 \Delta_{\max}$, $\rho_k \geq 1$, and $\{\|c(x_k)\|\}$ is bounded by Assumption (A4).

To prove inequality (A.2) we recall the definition of the actual reduction

$$ared(s_k;\rho_k) = \mathcal{L}(x_k,\lambda_k) + \rho_k \|c(x_k)\|_{\mathcal{C}}^2 - \mathcal{L}(x_k+s_k,\lambda_{k+1}) - \rho_k \|c(x_k+s_k)\|_{\mathcal{C}}^2$$

and we note that definitions (3.8) and (3.9) give

$$\begin{aligned} &pred(n_k,\widetilde{t}_k;\rho_k) + rpred(r_k^t;\rho_k) \\ &\quad = -\left\langle \widetilde{W}_k g_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} - \frac{1}{2}\left\langle H_k \widetilde{t}_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} - \langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k \rangle_{\mathcal{X}} - \frac{1}{2}\langle H_k n_k, n_k \rangle_{\mathcal{X}} \\ &\qquad - \langle \lambda_{k+1} - \lambda_k, c_x(x_k)s_k + c(x_k) \rangle_{\mathcal{C}} + \rho_k(\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)s_k + c(x_k)\|_{\mathcal{C}}^2). \end{aligned}$$

Using $g_k = H_k n_k + \nabla_x \mathcal{L}(x_k,\lambda_k)$ and $W_k = W_k^* = W_k^2$, yield

$$\begin{aligned} (\text{A.4}) \quad &ared(s_k;\rho_k) - pred(n_k,\widetilde{t}_k;\rho_k) - rpred(r_k^t;\rho_k) \\ &\quad = -\mathcal{L}(x_k+s_k,\lambda_k) + \mathcal{L}(x_k,\lambda_k) \\ &\qquad + \left\langle W_k g_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} + \langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k \rangle_{\mathcal{X}} + \frac{1}{2}\left\langle H_k \widetilde{t}_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} + \frac{1}{2}\langle H_k n_k, n_k \rangle_{\mathcal{X}} \\ &\qquad + \left\langle \widetilde{W}_k g_k - W_k g_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} \\ &\qquad + \langle \lambda_{k+1} - \lambda_k, -c(x_k+s_k) + c_x(x_k)s_k + c(x_k) \rangle_{\mathcal{C}} \\ &\qquad - \rho_k \Big(\|c(x_k+s_k)\|_{\mathcal{C}}^2 - \|c_x(x_k)s_k + c(x_k)\|_{\mathcal{C}}^2 \Big). \end{aligned}$$

We write (recall that $W_k = W_k^*$ and $g_k = \nabla_x \mathcal{L}(x_k,\lambda_k) + H_k n_k$)

$$\begin{aligned} &-\mathcal{L}(x_k+s_k,\lambda_k) + \mathcal{L}(x_k,\lambda_k) + \left\langle W_k g_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} \\ &\qquad + \langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k \rangle_{\mathcal{X}} + \frac{1}{2}\left\langle H_k \widetilde{t}_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} + \frac{1}{2}\langle H_k n_k, n_k \rangle_{\mathcal{X}} \\ &\quad = -\mathcal{L}(x_k+s_k,\lambda_k) + \mathcal{L}(x_k,\lambda_k) + \langle \nabla_x \mathcal{L}(x_k,\lambda_k), s_k \rangle_{\mathcal{X}} + \frac{1}{2}\langle H_k s_k, s_k \rangle_{\mathcal{X}} \\ &\qquad - \langle \nabla_x \mathcal{L}(x_k,\lambda_k), s_k \rangle_{\mathcal{X}} - \frac{1}{2}\langle H_k s_k, s_k \rangle_{\mathcal{X}} + \left\langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k + W_k \widetilde{t}_k \right\rangle_{\mathcal{X}} \\ &\qquad + \left\langle H_k n_k, W_k \widetilde{t}_k \right\rangle_{\mathcal{X}} + \frac{1}{2}\left\langle H_k \widetilde{t}_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} + \frac{1}{2}\langle H_k n_k, n_k \rangle_{\mathcal{X}}. \end{aligned}$$

Using a Taylor expansion of $\mathcal{L}(x_{k+1},\lambda_k)$ and $s_k = n_k + t_k$ we find that

$$\begin{aligned} &\Big| -\mathcal{L}(x_k+s_k,\lambda_k) + \mathcal{L}(x_k,\lambda_k) + \left\langle W_k g_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} + \langle \nabla_x \mathcal{L}(x_k,\lambda_k), n_k \rangle_{\mathcal{X}} \\ &\qquad + \frac{1}{2}\left\langle H_k \widetilde{t}_k, \widetilde{t}_k \right\rangle_{\mathcal{X}} + \frac{1}{2}\langle H_k n_k, n_k \rangle_{\mathcal{X}} \Big| \\ &\leq \frac{1}{2}\|H_k - \nabla_{xx}\mathcal{L}(x_k + \tau_k^1 s_k, \lambda_k)\| \|s_k\|_{\mathcal{X}}^2 \\ &\qquad + \|H_k n_k + \nabla_x \mathcal{L}(x_k,\lambda_k)\|_{\mathcal{X}} \, \|t_k - W_k \widetilde{t}_k\|_{\mathcal{X}} + \frac{1}{2}\|H_k\| \, \|\widetilde{t}_k\|_{\mathcal{X}}^2 + \frac{1}{2}\|H_k\| \, \|t_k\|_{\mathcal{X}}^2, \end{aligned}$$

where $0 < \tau_k^1 < 1$. The inequalities (3.12), (3.13), (3.14), (A.1), and Assumptions (A4)–(A6) imply the existence of $\tilde{\kappa}_9 > 0$ independent of k such that

$$\begin{aligned}(\text{A.5}) \qquad &\Big| - \mathcal{L}(x_k + s_k, \lambda_k) + \mathcal{L}(x_k, \lambda_k) + \langle W_k g_k, \widetilde{t}_k \rangle_{\mathcal{X}} + \langle \nabla_x \mathcal{L}(x_k, \lambda_k), n_k \rangle_{\mathcal{X}} \\ &\quad + \frac{1}{2} \langle H_k \widetilde{t}_k, \widetilde{t}_k \rangle_{\mathcal{X}} + \frac{1}{2} \langle H_k n_k, n_k \rangle_{\mathcal{X}} \Big| \\ &\quad \leq \tilde{\kappa}_9 \Delta_k \|s_k\|_{\mathcal{X}}.\end{aligned}$$

Inequalities (3.6) and (3.13) imply

$$(\text{A.6}) \qquad \langle \widetilde{W}_k g_k - W_k g_k, \widetilde{t}_k \rangle_{\mathcal{X}} \leq \xi_1 \Delta_k \|\widetilde{t}_k\|_{\mathcal{X}} \leq \xi_1 \xi_4 \Delta_k \|s_k\|_{\mathcal{X}}.$$

The final term in (A.4) can be estimated exactly as in [16, p. 297]. After a series of Taylor expansions, we obtain

$$\begin{aligned}(\text{A.7}) \qquad &\langle \lambda_{k+1} - \lambda_k, -c(x_k + s_k) + c_x(x_k) s_k + c(x_k) \rangle_{\mathcal{C}} \\ &\quad - \rho_k (\|c(x_k + s_k)\|_{\mathcal{C}}^2 - \|c_x(x_k) s_k + c(x_k)\|_{\mathcal{C}}^2) \\ &\quad \leq \kappa_{10} \rho_k \|s_k\|_{\mathcal{X}}^3 + \kappa_{11} \rho_k \|s_k\|_{\mathcal{X}}^2 \|c(x_k)\|_{\mathcal{C}}.\end{aligned}$$

Substitution of (A.5)–(A.7) into (A.4) along with the assumptions (A4)–(A6) gives the estimate (A.2).

iii. Predicted reduction in the model. Using (3.7) one can use the same arguments as those applied in [8, Lemma 7.3, 7.6] to show the existence of a constant K_5 independent of k such that for any positive ρ

$$\begin{aligned}pred(n_k, \widetilde{t}_k; \rho) \geq\ & \kappa_4 \|\widetilde{W}_k g_k\|_{\mathcal{X}} \min \left\{ \kappa_5 \|\widetilde{W}_k g_k\|_{\mathcal{X}}, \kappa_6 \Delta_k \right\} \\ & - K_5 \|c(x_k)\|_{\mathcal{C}} + \rho \left(\|c(x_k)\|_{\mathcal{C}}^2 - \|c_x(x_k) n_k + c(x_k)\|_{\mathcal{C}}^2 \right).\end{aligned}$$

iv. We can now complete the proof as in [16]. The estimates (A.2) and (A.3) are used in the analysis only when rejection occurs in step 2(cvi) of Algorithm 3.3. If s_k is rejected, we know that

$$0 < 1 - \eta_1 \leq \left| \frac{ared(s_k; \rho_k)}{pred(n_k, \widetilde{t}_k; \rho_k)} - 1 \right|,$$

which in our inexact context implies

$$1 - \eta_1 \leq \left| \frac{ared(s_k; \rho_k) - pred(n_k, \widetilde{t}_k; \rho_k) - rpred(r_k^t; \rho_k)}{pred(n_k, \widetilde{t}_k; \rho_k)} \right| + \eta_0.$$

Thus, when the estimate (A.3) is required, we obtain

$$0 < 1 - \eta_0 - \eta_1 \leq \frac{\kappa_{12} \rho_k \Delta_k \|s_k\|}{pred(n_k, \widetilde{t}_k; \rho_k)},$$

and the analysis in [8] remains unchanged except for the fact that $pred(n_k, \widetilde{t}_k; \rho_k)$ is used instead of $pred(s_k; \rho_k)$ and the lower bound $1 - \eta_0 - \eta_1 \in (0, 1)$ is used instead of $1 - \eta_1$. A similar bound is obtained when the estimate is given by (A.3).

The limit (3.16) is obtained by combining (3.15), (3.6), and (3.1). ∎

Appendix B. Error Estimates for inexact null-space projections.

In this section we provide a bound for $\|\widetilde{W}_k - W_k\|_{L(\mathcal{X})}$ by managing the size of the residuals $e_i = (e_{1,i}\ e_{2,i}) \in \mathcal{X} \times \mathcal{C}$ in (4.21). Instead of using the representation (4.12) we use (4.14). Let i_k be the number of iterations performed in Algorithm 4.5. Furthermore, let R_{i_k} and $\widetilde{Y_{i_k}}$ be defined as in (4.10) and (4.11), respectively.

The advantage of representation (4.14) is that if exact linear system solves are used in Algorithm 4.5, then the projected residual vectors $W_k g_k, W_k \widetilde{r}_1, \ldots, W_k \widetilde{r}_{i_k}$ are orthogonal, and since $W_k^* = W_k = W_k W_k$, the matrix $\widetilde{Y_{i_k}}^* R_{i_k}$ simplifies to $\widetilde{Y_{i_k}}^* R_{i_k} = D_{i_k}^2$, where

$$D_{i_k} = \operatorname{diag}(\|\mathcal{W}_k(g_k)\|_{\mathcal{X}}, \|\mathcal{W}_k(\widetilde{r}_1)\|_{\mathcal{X}}, \ldots, \|\mathcal{W}_k(\widetilde{r}_{i_k})\|_{\mathcal{X}}).$$

We define

$$\widehat{S}_{i_k} = D_{i_k}^{-1}(\widetilde{Y_{i_k}}^T R_{i_k} - D_{i_k}^2) D_{i_k}^{-1}, \quad \text{i.e.} \quad \widetilde{Y_{i_k}}^T R_{i_k} = D_{i_k}(I_{i_k} + \widehat{S}_{i_k}) D_{i_k}.$$

If $\|\widehat{S}_{i_k}\|_2 < 1$, then $(I_{i_k} + \widehat{S}_{i_k})$ is invertible and $(I_{i_k} + \widehat{S}_{i_k})^{-1} = I_{i_k} + S_{i_k}$ with $S_{i_k} = -\widehat{S}_{i_k}(I_{i_k} + \widehat{S}_{i_k})^{-1}$ and $\|S_{i_k}\|_2 \le \|\widehat{S}_{i_k}\|_2/(1 - \|\widehat{S}_{i_k}\|_2)$; see, e.g., [12, p. 58]. In this case we can write

$$\begin{aligned} \widetilde{W}_k &= W_k + (\widetilde{Y_{i_k}} - Y_{i_k})(\widetilde{Y_{i_k}}^* R_{i_k})^{-1} \widetilde{Y_{i_k}}^* \\ &= W_k + (\widetilde{Y_{i_k}} - Y_{i_k}) D_{i_k}^{-1}(I_{i_k} + S_{i_k}) D_{i_k}^{-1} \widetilde{Y}_i^*. \end{aligned} \tag{B.1}$$

THEOREM B.1. *Let the assumption* (A4) *be satisfied. Suppose that Algorithm* 4.5 *terminates after $i_k + 1$ applications of $\mathcal{W}_k$. Suppose $\mathcal{W}_k(\widetilde{r}_{i_k}) \neq 0$. Let $\widetilde{r}_0$ be computed as in LSSC* 4.14 *and let $\widetilde{z}_i = \mathcal{W}_k(\widetilde{r}_i)$, $j = 1, \ldots, i$, be computed as in* (4.21). *If $\|\widehat{S}_{i_k}\|_2 \le \xi^{ortho} < 1$ and if*

$$\|e_{1,i}\|_{\mathcal{X}} + \|e_{2,i}\|_{\mathcal{C}} \le \xi^{proj} \min\left\{\|\widetilde{z}_i\|_{\mathcal{X}}, \|\widetilde{r}_i\|_{\mathcal{X}}\right\}, \tag{B.2}$$

then there exists a constant $C_1 > 0$ independent of k such that for $\widetilde{W}_k$ defined in (4.14)

$$\|\widetilde{W}_k - W_k\|_{\mathcal{X}} \le C_1 \left(1 + \frac{\xi^{ortho}}{1 - \xi^{ortho}}\right)(i_k + 1) \max\left\{\xi^{pg}, \xi^{proj}\right\}. \tag{B.3}$$

Proof. If $\|\widehat{S}_{i_k}\|_2 \le \xi^{ortho} < 1$, then we can write (B.1), which implies

(B.4)

$$\begin{aligned} \|\widetilde{W}_k - W_k\|_{L(\mathcal{X})} &\le (1 + \|S_{i_k}\|_2)\|(\widetilde{Y_{i_k}} - Y_{i_k}) D_{i_k}^{-1}\|_{L(\mathbb{R}^{i_k+1}, \mathcal{X})} \|\widetilde{Y_{i_k}} D_{i_k}^{-1}\|_{L(\mathcal{X}, \mathbb{R}^{i_k+1})}, \\ &\le \left(1 + \frac{\xi^{ortho}}{1 - \xi^{ortho}}\right) \|(\widetilde{Y_{i_k}} - Y_{i_k}) D_{i_k}^{-1}\|_{L(\mathbb{R}^{i_k+1}, \mathcal{X})} \|\widetilde{Y_{i_k}} D_{i_k}^{-1}\|_{L(\mathcal{X}, \mathbb{R}^{i_k+1})}. \end{aligned}$$

If $a_0, \ldots, a_{i_k} \in \mathcal{X}$ and $A = [a_0, \ldots, a_{i_k}] \in L(\mathbb{R}^{i_k+1}, \mathcal{X})$, then

$$\begin{aligned} \|A\|_{L(\mathbb{R}^{i_k+1}, \mathcal{X})} &= \sup_{x \neq 0} \frac{\|\sum_{i=0}^{i_k} x_i a_i\|_{\mathcal{X}}}{\|x\|_2} \le \sup_{x \neq 0} \frac{\sum_{i=0}^{i_k} |x_{i_k}| \|a_i\|_{\mathcal{X}}}{\|x\|_2} \\ &\le \max_{i=0,\ldots,i_k} \|a_i\|_{\mathcal{X}} \sup_{x \neq 0} \frac{\sum_{i=0}^{i_k} |x_i|}{\|x\|_2} \le \sqrt{i_k + 1} \max_{i=0,\ldots,i_k} \|a_i\|_{\mathcal{X}}. \end{aligned} \tag{B.5}$$

Consequently, the norm of

$$\widetilde{Y_{i_k}} D_{i_k}^{-1} = \left[\frac{\mathcal{W}_k(g_k)}{\|\mathcal{W}_k(g_k)\|_{\mathcal{X}}}, \frac{\mathcal{W}_k(\widetilde{r}_1)}{\|\mathcal{W}_k(\widetilde{r}_1)\|_{\mathcal{X}}}, \ldots, \frac{\mathcal{W}_k(\widetilde{r}_{i_k})}{\|\mathcal{W}_k(\widetilde{r}_{i_k})\|_{\mathcal{X}}} \right]$$

is given by

$$\|\widetilde{Y_{i_k}} D_{i_k}^{-1}\|_{L(\mathcal{X}, \mathbb{R}^{i_k+1})} = \sqrt{i_k + 1}. \tag{B.6}$$

We recall $\widetilde{r}_0 = \mathcal{W}_k(g_k)$. Following the proof of Theorem 4.12 and LSSC 4.14 there exists $\nu > \|G_k^{-1}\|_{L(\mathcal{X}\times\mathcal{C})}$ independent of k such that

$$\|\widetilde{r}_0 - W_k g_k\|_{\mathcal{X}} \leq \nu \xi^{pg} \min\{\|\widetilde{r}_0\|_{\mathcal{X}}, \|g_k\|_{\mathcal{X}}\}$$

and

$$\|\widetilde{z}_j - W_k \widetilde{r}_j\|_{\mathcal{X}} \leq \nu \xi^{proj} \min\{\|\widetilde{z}_j\|_{\mathcal{X}}, \|\widetilde{r}_i\|_{\mathcal{X}}\}, \quad j = 1, \ldots, i_k.$$

This directly yields $\|\widetilde{r}_0 - W_k g_k\|_{\mathcal{X}} / \|\widetilde{r}_0\|_{\mathcal{X}} \leq \nu \xi^{pg}$ and $\|\widetilde{z}_j - W_k \widetilde{r}_j\|_{\mathcal{X}} / \|\widetilde{z}_j\|_{\mathcal{X}} \leq \nu \xi^{proj}$, $j = 1, \ldots, i_k$. From

$$(\widetilde{Y_{i_k}} - Y_{i_k}) D_{i_k}^{-1} = \left[\frac{\widetilde{r}_0 - W_k g_k}{\|\widetilde{r}_0\|_{\mathcal{X}}}, \frac{\mathcal{W}_k(\widetilde{r}_1) - W_k \widetilde{r}_1}{\|\mathcal{W}_k(\widetilde{r}_1)\|_{\mathcal{X}}}, \ldots, \frac{\mathcal{W}_k(\widetilde{r}_{i_k}) - W_k \widetilde{r}_{i_k}}{\|\mathcal{W}_k(\widetilde{r}_{i_k})\|_{\mathcal{X}}} \right]$$

and (B.5) we obtain

$$\|(\widetilde{Y_{i_k}} - Y_{i_k}) D_{i_k}^{-1}\|_{L(\mathbb{R}^{i_k+1}, \mathcal{X})} \leq \nu \sqrt{i_k + 1} \max\left\{\xi^{pg}, \xi^{proj}\right\}. \tag{B.7}$$

Combining (B.4), (B.6), and (B.7) gives (B.3). ∎

Remark B.2. In our implementation we monitor the norm of $\widehat{S}_{i_k} = D_{i_k}^{-1}(\widetilde{Y_{i_k}}^T R_{i_k} - D_{i_k}^2) D_{i_k}^{-1}$ and terminate Algorithm 4.5 if $\|\widehat{S}_{i_k}\| > 1/2$.

Acknowledgments. We thank the referees for their careful reading and comments that led to clarifications in our presentation.

REFERENCES

[1] O. Axelsson and P. S. Vassilevski, *A black box generalized conjugate gradient solver with inner iterations and variable-step preconditioning*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 625–644.

[2] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[3] G. Biros and O. Ghattas, *Inexactness issues in the Lagrange-Newton-Krylov-Schur method for PDE-constrained optimization*, in Large-Scale Constrained Optimization, L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, eds., Lecture Notes in Comput. Sci. 30, Springer, Berlin, 2003, pp. 93–114.

[4] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

[5] R. H. Byrd, F. E. Curtis, and J. Nocedal, *An inexact Newton method for nonconvex equality constrained optimization*, Math. Program., 122 (2010), pp. 273–299.

[6] R. H. Byrd, M. E. Hribar, and J. Nocedal, *An interior point algorithm for large scale nonlinear programming*, SIAM J. Optimization, 9 (1999), pp. 877–900.

[7] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *Trust–Region Methods*, MPS/SIAM Ser. Optim. 1, SIAM, Philadelphia, 2000.

[8] J. E. Dennis, Jr., M. El-Alem, and M. C. Maciel, *A global convergence theory for general trust–region–based algorithms for equality constrained optimization*, SIAM J. Optimization, 7 (1997), pp. 177–207.

[9] 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 and Optimization, 36 (1998), pp. 1750–1794.
[10] J. E. Dennis and L. N. Vicente, *On the convergence theory of trust–region–based algorithms for equality–constrained optimization*, SIAM J. Optimization, 7 (1997), pp. 927–950.
[11] H. S. Dollar, N. I. M. Gould, M. Stoll, and A. J. Wathen, *Preconditioning saddle-point systems with applications in optimization*, SIAM J. Sci. Comput., 32 (2010), pp. 249–270.
[12] G. H. Golub and C. F. Van Loan, *Matrix computations*, 3rd ed., Johns Hopkins Stud. Math. Sci., Johns Hopkins University Press, Baltimore, MD, 1996.
[13] G. H. Golub and Q. Ye, *Inexact preconditioned conjugate gradient method with inner-outer iteration*, SIAM J. Sci. Comput., 21 (1999), pp. 1305–1320.
[14] N. I. M. Gould, M. E. Hribar, and J. Nocedal, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395.
[15] M. Heinkenschloss and D. Ridzal, *Integration of sequential quadratic programming and domain decomposition methods for nonlinear optimal control problems*, in Domain Decomposition Methods in Science and Engineering XVII, U. Langer, M. Discacciati, D. Keyes, O. Widlund, and W. Zulehner, eds., Lect. Notes Comput. Sci. Engrg. 60, Springer Verlag, Berlin, 2008, pp. 69–80.
[16] M. Heinkenschloss and L. N. Vicente, *Analysis of inexact trust–region SQP algorithms*, SIAM J. Optimization, 12 (2001), pp. 283–302.
[17] L. S. Hou and S. S. Ravindran, *Numerical approximation of optimal flow control problems by a penalty method: Error estimates and numerical results*, SIAM Journal of Scientific Computing, 20 (1999), pp. 1753–1777.
[18] H. Jäger and E. W. Sachs, *Global convergence of inexact reduced SQP-methods*, Optim. Methods and Software, 7 (1997), pp. 83–110.
[19] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Verlag, Berlin, 1999.
[20] Y. Notay, *Flexible conjugate gradients*, SIAM J. Sci. Comput., 22 (2000), pp. 1444–1460.
[21] E. E. Prudencio, R. Byrd, and X.-C. Cai, *Parallel full space SQP Lagrange–Newton–Krylov–Schwarz algorithms for PDE-constrained optimization problems*, SIAM J. Sci. Comput., 27 (2006), pp. 1305–1328.
[22] D. Ridzal, *Trust Region SQP Methods With Inexact Linear System Solves For Large-Scale Optimization*, PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 2006.
[23] D. Ridzal, M. Aguiló, and M. Heinkenschloss, *Numerical Study of a Matrix-Free Trust-Region SQP Method for Equality Constrained Optimization*, Technical Report SAND 2011-9346, Sandia National Laboratories, Albuquerque, NM, 2011.
[24] V. Simoncini and D. B. Szyld, *Theory of inexact Krylov subspace methods and applications to scientific computing*, SIAM J. Sci. Comput., 25 (2003), pp. 454–477.
[25] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.
[26] Ph. L. Toint, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, I. S. Duff, ed., Academic Press, New York, 1981, pp. 57–87.
[27] J. C. Ziems and S. Ulbrich, *Adaptive multilevel inexact SQP methods for PDE-constrained optimization*, SIAM Journal on Optimization, 21 (2011), pp. 1–40.