Analysis of Inexact Trust-Region Interior-Point SQP Algorithms

Matthias Heinkenschloss

Luis N. Vicente

TR95-18
June 1995 (revised April 1996)

Department of Computational and Applied Mathematics
MS 134
Rice University
6100 Main Street
Houston, TX 77005–1892

ANALYSIS OF INEXACT TRUST–REGION INTERIOR–POINT SQP ALGORITHMS

MATTHIAS HEINKENSCHLOSS * AND LUíS N. VICENTE †

Abstract. In this paper we analyze inexact trust–region interior–point (TRIP) sequential quadratic programming (SQP) algorithms for the solution of optimization problems with nonlinear equality constraints and simple bound constraints on some of the variables. Such problems arise in many engineering applications, in particular in optimal control problems with bounds on the control. The nonlinear constraints often come from the discretization of partial differential equations. In such cases the calculation of derivative information and the solution of linearized equations is expensive. Often, the solution of linear systems and derivatives are computed inexactly yielding nonzero residuals.

This paper analyzes the effect of the inexactness onto the convergence of TRIP SQP and gives practical rules to control the size of the residuals of these inexact calculations. It is shown that if the size of the residuals is of the order of both the size of the constraints and the trust–region radius, then the TRIP SQP algorithms are globally convergent to a stationary point. Numerical experiments with two optimal control problems governed by nonlinear partial differential equations are reported.

Keywords. nonlinear programming, trust–region methods, interior–point algorithms, Coleman and Li affine scaling, simple bounds, inexact linear systems solvers, Krylov subspace methods, optimal control

AMS subject classification. 49M37, 90C06, 90C30

1. Introduction. In this paper we study a class of optimization algorithms that allow the use of inexact information for the solution of minimization problems with nonlinear equality constraints and simple bound constraints on some of the variables. More precisely, the problems we are interested in are of the form

$$\begin{aligned} &\text{minimize} f(y,u) \\ &\text{subject to} C(y,u) = 0, \\ &\qquad u \in \mathcal{B} = \{u : \ a \leq u \leq b\}, \end{aligned} \tag{1.1}$$

where $y \in \mathbb{R}^m$, $u \in \mathbb{R}^{n-m}$, $a \in (\mathbb{R} \cup \{-\infty\})^{n-m}$, $b \in (\mathbb{R} \cup \{+\infty\})^{n-m}$, $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $C : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, $m < n$, and f and C are assumed to be at least twice continuously differentiable functions. Applications include optimal control problems, parameter identification problems and inverse problems, and design optimization.

The algorithms investigated in this paper are extensions of the trust–region interior–point (TRIP) sequential quadratic programming (SQP) algorithms introduced and analyzed in [15]. The TRIP SQP algorithms are SQP methods that use trust regions as a strategy for globalization and for regularization of the subproblems and that apply an affine scaling interior–point approach to deal with the bounds on the variables u. However, the analysis in this paper will also be relevant for other SQP algorithms.

The minimization problem (1.1) often arises from the discretization of optimal control problems. Here y are the state variables, u are the control variables, and $C(y,u) = 0$ is the discretized state equation. TRIP SQP algorithms utilize the structure of the problems induced by the partitioning of the variables into states and controls. Subproblems in the TRIP SQP

*Department of Computational and Applied Mathematics Rice University, Houston, TX 77005–1892, USA. E-Mail: heinken@caam.rice.edu. This author was supported by the NSF under Grant DMS–9403699, by the DoE under Grant DE–FG03–95ER25257, and by the AFOSR under Grant F49620–93–1–0280.

†Departamento de Matemática, Universidade de Coimbra, 3000 Coimbra, Portugal. This work was developed while the author was a graduate student at the Department of Computational and Applied Mathematics of Rice University. E-Mail: lvicente@mat.uc.pt. Support of this author has been provided by INVOTAN (NATO scholarship), CCLA (Fulbright scholarship), FLAD, and NSF cooperative agreement CCR–9120008.

algorithms are solved iteratively. As a consequence, only directional derivatives are needed in the implementation of these algorithms. However, differentiability is required to guarantee convergence. In [15] it is assumed that derivative information is available exactly and that linearized equations can be solved exactly. In many applications these assumptions are unrealistic. Derivative information may be approximated, for example, by finite differences. Moreover, the linearized equations are often discretizations of partial differential equations and iterative solvers are used for their solution. The purpose of this paper is to extend the exact TRIP SQP algorithms to allow inexact calculations in tasks involving derivatives of $C(y,u)$. Inexactness in derivatives of the objective function f also can be allowed, but it is not done here to keep the presentation simpler. Since we treat states and controls as independent variables, and since the objective functions are often rather simple, e.g. least squares functionals, this does not present a severe restriction. One goal for our analysis is to derive measures of inexactness and controls of inexactness that are simple to implement.

To explain how we deal with inexactness and to present the main results of this paper, we need to introduce some of the structure of problem (1.1) (see also references [26], [30], [31], [32]). For convenience we write

$$x = \begin{pmatrix} y \\ u \end{pmatrix}.$$

Due to the partitioning of the variable x into y and u, the Jacobian matrix of $C(x)$ can be written as

$$J(x) = \begin{pmatrix} C_y(x) & C_u(x) \end{pmatrix},$$

where $C_y(x) \in \mathbb{R}^{m\times m}$ and $C_u(x) \in \mathbb{R}^{m\times(n-m)}$.

In the exact TRIP SQP algorithms, we have to compute quantities of the form $C_u(x)d_u$ and $C_u^T(x)d_y$, and we have to solve linear systems of the form

$$C_y(x_k)s = \bar{b}_k \quad \text{and} \quad C_y(x_k)^T s = \hat{b}_k. \tag{1.2}$$

Since these systems are solved inexactly, what is computed are $\bar{s}_k$ and $\hat{s}_k$ such that

$$C_y(x_k)\bar{s}_k = \bar{b}_k + \bar{e}_k \quad \text{and} \quad C_y(x_k)^T \hat{s}_k = \hat{b}_k + \hat{e}_k,$$

where $\bar{e}_k$ and $\hat{e}_k$ are residual vectors. In many iterative methods, like for instance Krylov subspace methods, the norms $\|\bar{e}_k\|$ and $\|\hat{e}_k\|$ can be computed efficiently with few extra operations. Such residuals are used to measure inexactness.

We give conditions on the amount of inexactness allowed in the inexact TRIP SQP algorithms that guarantee global convergence to a point satisfying the first–order necessary optimality conditions. In the case of the linear solvers, these conditions are the following:

$$\|\bar{e}_k\| = \mathcal{O}\Big(\min\{\delta_k, \|C(x_k)\|\}\Big) \quad \text{and} \quad \|\hat{e}_k\| = \mathcal{O}\Big(\|C(x_k)\|\Big), \tag{1.3}$$

where δ_k is the trust–region radius and $\|C(x_k)\|$ is the norm of the residual of the constraints. Thus as the iterates approach feasibility the accuracy with which the linear systems are solved has to increase. Moreover, the accuracy of the linear system solves has to increase if the region where the quadratic model is trusted becomes small. This also is reasonable since the trust region should not be reduced unnecessarily. Similar results are derived for the inexactness that arises in the computation of directional derivatives of $C(x)$. The details are presented in this paper.

The convergence results presented in this paper rely on the theory given in [14], [15]. A comprehensive convergence theory is presented in [52].

We have applied the inexact TRIP SQP algorithms to the solution of two optimal control problems, a boundary control problem for a nonlinear heat equation and a distributed control problem for a semi–linear elliptic equation. Preconditioned Krylov subspace methods were used to solve the linearized state and adjoint equations (1.2). The numerical results reported in Section 9 confirm our analysis.

It should be pointed out that by inexactness we mean inexact derivative information and inexact solution of linear systems. Trust–region methods allow another level of inexactness that is also treated here and in most other papers on trust–region methods: in trust–region methods the quadratic programming subproblems do not have to be solved exactly. It is sufficient to compute steps that predict the so–called fraction of Cauchy decrease condition. This allows the application of a variety of methods for the approximate solution of subproblems.

In the context of systems of nonlinear equations, inexact or truncated Newton methods have been proposed and analyzed by many authors. Some of the pioneer work in this area can be found in [11], [50]. More recent references are [3], [4], [18], [20], [19], [29]. Most of the recent papers investigate the use of Krylov subspace methods for the solution of linear systems, like GMRES [46], in inexact Newton methods. These Krylov subspace methods are attractive because they monitor the residual norm of the linear system in an efficient way and only require Jacobian times a vector, not the Jacobian in explicit form. The results for the solution of systems of nonlinear equations have been extended to analyze inexact Newton methods for the solution of unconstrained minimization problems, e.g. [12], [39], [41]. In a recent paper [55], the impact of inexactness in reduced gradient methods for design optimization has been analyzed.

In nonlinear programming, inexactness has been studied by [2], [10], [13], [22], [34], [40], [53] among others. The papers [13], [22], [34], [40] investigating SQP methods mostly study the influence of inexactness on the local convergence rate. In [40] conditions on the inexactness are given that guarantee descent in the merit function. In the papers mentioned previously, the inexactness is often measured using the residual of the linearized system of nonlinear equations arising from the first–order necessary optimality conditions, or some variation thereof. If globalizations are included in the investigations, then line–search strategies are used. To our knowledge, inexactness for SQP methods with trust–region globalizations has not been studied in the literature. Due to the computation of the step in two stages, the computation of the quasi–normal component and of the tangential component, the analysis of inexactness in SQP methods with trust–region globalizations requires techniques different from those that can be used for line search globalizations. Other papers which investigate SQP methods for large scale problems, but without treatment of inexact linear systems solves and inexact derivative information include [1], [33], [38].

This paper is organized as follows. In Section 2, we state the first–order necessary optimality conditions of problem (1.1). A review of the features of the exact TRIP SQP algorithms necessary for the inexact analysis is given in Section 3. The inexact TRIP SQP algorithms are presented in Section 4. Here we also list the assumptions under which convergence can be guaranteed. In Section 5, we prove global convergence. The remaining of the paper deals with practical issues concerning the step and multipliers calculations. Each step is decomposed in two components: a quasi–normal component and a tangential component. In Section 6, we present several techniques to compute quasi–normal components and show how they fit into the theoretical framework given in Section 4. In Section 7, we discuss conjugate–gradient methods to compute the tangential component and analyze the influence of the inexactness. The inexact calculation of the multipliers is discussed in Section 8. In

Section 9, we present our numerical experiments. Section 10 reports on our conclusions and directions of future work.

We use subscripted indices to represent the evaluation of a function at a particular point of the sequences $\{x_k\}$ and $\{\lambda_k\}$. For instance, f_k represents $f(x_k)$. The vector and matrix norms used are the ℓ_2 norms, and I_l represents the identity matrix of order l. Also $(z)_y$ and $(z)_u$ represent the subvectors of $z \in \mathbb{R}^n$ corresponding to the y and u components, respectively.

2. First–order necessary optimality conditions. We say that

$$s = \begin{pmatrix} s_y \\ s_u \end{pmatrix}$$

satisfies the linearized state equations at x if $J(x)s = -C(x)$ or equivalently if

$$C_y(x)s_y + C_u(x)s_u = -C(x).$$

From the previous equation we can see that the columns of

$$W(x) = \begin{pmatrix} -C_y(x)^{-1}C_u(x) \\ I_{n-m} \end{pmatrix} \tag{2.1}$$

form a basis of the null space of $J(x)$.

The structure of the Jacobian and the definition of its null space using the matrix $W(x)$ is important for the formulation of the first–order necessary optimality conditions. In the following, we give a brief derivation of the form of the these conditions used in this paper. Further details can be found in [15]. For box constrained problems see also [8], [16].

We introduce the Lagrangian function

$$\ell(x, \lambda) = f(x) + \lambda^T C(x)$$

and we note that, due to the form of the bound constraints, the invertibility of $C_y(x)$ implies the linear independence of the active constraints. If x_* is a local solution of problem (1.1), then it satisfies the first–order Karush–Kuhn–Tucker (KKT) conditions:

$$\begin{aligned}
&C(x_*) = 0, \;\; a \le u_* \le b, \\
&\lambda_* = -C_y(x_*)^{-T}\nabla_y f(x_*), \\
&a_i < (u_*)_i < b_i \quad \Longrightarrow \quad (\nabla_u \ell(x_*, \lambda_*))_i = 0, \\
&(u_*)_i = a_i \quad\;\; \Longrightarrow \quad (\nabla_u \ell(x_*, \lambda_*))_i \ge 0, \\
&(u_*)_i = b_i \quad\;\; \Longrightarrow \quad (\nabla_u \ell(x_*, \lambda_*))_i \le 0.
\end{aligned}$$

It is not difficult to show that $\nabla_u \ell(x_*, \lambda_*) = W(x_*)^T \nabla f(x_*)$. Thus, if we define the diagonal matrix $D(x)$ with diagonal elements given by

$$(D(x))_{ii} = \begin{cases}
(b-u)_i^{\frac{1}{2}} & \text{if} \quad \big(W(x)^T\nabla f(x)\big)_i < 0 \text{ and } b_i < +\infty, \\
1 & \text{if} \quad \big(W(x)^T\nabla f(x)\big)_i < 0 \text{ and } b_i = +\infty, \\
(u-a)_i^{\frac{1}{2}} & \text{if} \quad \big(W(x)^T\nabla f(x)\big)_i \ge 0 \text{ and } a_i > -\infty, \\
1 & \text{if} \quad \big(W(x)^T\nabla f(x)\big)_i \ge 0 \text{ and } a_i = -\infty,
\end{cases}$$

$i = 1, \ldots, n-m$, then we can write the first–order KKT conditions in the form

$$C(x_*) = 0, \;\; a \le u_* \le b, \tag{2.2}$$

$$D(x_*)\, W(x_*)^T \nabla f(x_*) = 0. \tag{2.3}$$

3. Exact TRIP SQP algorithms. The algorithms described in this section have been proposed and analyzed in [15]. They use exact first–order derivative information and require that the linear systems are solved exactly. The purpose of this section is to provide the framework for this class of algorithms. The TRIP SQP algorithms are iterative algorithms. At a given iteration, an approximation x_k to the solution is given, and a step

$$s_k = \begin{pmatrix} (s_k)_y \\ (s_k)_u \end{pmatrix},$$

of the form $s_k = s_k^{\mathsf{n}} + s_k^{\mathsf{t}}$ is computed. The components s_k^{n} and s_k^{t} of the step are called the quasi–normal component and the tangential component, respectively. If the step is accepted, the process continues by setting x_{k+1}, the new iterate, to $x_k + s_k$. Otherwise the step has to be recomputed and tested again for acceptance.

The role of the quasi–normal component s_k^{n} is to move towards feasibility of the equality constraints. This component is of the form

$$s_k^{\mathsf{n}} = \begin{pmatrix} (s_k^{\mathsf{n}})_y \\ 0 \end{pmatrix}, \tag{3.1}$$

and it is computed as an approximate solution of the trust–region subproblem

$$\begin{aligned} &\text{minimize} \frac{1}{2}\|J_k s^{\mathsf{n}} + C_k\|^2 = \frac{1}{2}\|C_y(x_k)\, s_y^{\mathsf{n}} + C_k\|^2 \\ &\text{subject to} \|s_y^{\mathsf{n}}\| \leq \delta_k, \end{aligned}$$

where δ_k is the trust radius. This quasi–normal component must satisfy

$$\|s_k^{\mathsf{n}}\| \leq \kappa_1 \|C_k\| \tag{3.2}$$

and

$$\|C_k\|^2 - \|C_y(x_k)\,(s_k^{\mathsf{n}})_y + C_k\|^2 \geq \kappa_2 \|C_k\| \min\{\kappa_3 \|C_k\|, \delta_k\}, \tag{3.3}$$

where κ_1, κ_2, and κ_3 are positive constants independent of k. In Section 6, we describe several practical ways to compute the quasi–normal component that satisfy conditions (3.1)–(3.3).

The tangential component minimizes a quadratic model of the Lagrangian function in the null space of the linearized constraints and subject to a trust–region constraint. The tangential component is of the form $s_k^{\mathsf{t}} = W_k\,(s_k)_u$, where $W_k = W(x_k)$ is the representation of the null space of J_k defined in (2.1). The component $(s_k)_u$ must satisfy the bound constraints

$$\sigma_k(a - u_k) \leq (s_k)_u \leq \sigma_k(b - u_k), \tag{3.4}$$

where $\sigma_k \in [\sigma, 1) \subset (0, 1)$. This ensures that $u_k + (s_k)_u$ remains strictly feasible with respect to the bound constraints $a \leq u \leq b$.

For further description on how $(s_k)_u$ is computed, consider the quadratic model:

$$\begin{aligned} \Psi_k(s_u) &= q_k(s_k^{\mathsf{n}} + W_k s_u) + \tfrac{1}{2} s_u^T \left(E_k \bar{D}_k^{-2}\right) s_u \\ &= q_k(s_k^{\mathsf{n}}) + (W_k^T \nabla q_k(s_k^{\mathsf{n}}))^T s_u + \tfrac{1}{2} s_u^T \left(W_k^T H_k W_k + E_k \bar{D}_k^{-2}\right) s_u, \end{aligned} \tag{3.5}$$

where $q_k(s)$ is the quadratic approximation of the Lagrangian function $\ell(x_k + s, \lambda_k)$ given by:

$$q_k(s) = \ell_k + \nabla_x \ell(x_k, \lambda_k)^T s + \frac{1}{2} s^T H_k s,$$

and $W_k^T \nabla q_k(s_k^{\mathsf{n}}) = W_k^T (H_k s_k^{\mathsf{n}} + \nabla f_k)$. The matrix H_k denotes an approximation to $\nabla^2_{xx} \ell(x_k, \lambda_k)$, and $\bar{D}_k$ and E_k are diagonal matrices whose i–th diagonal element is given by

$$(\bar{D}_k)_{ii} = \begin{cases} (b - u_k)_i^{\frac{1}{2}} & \text{if} \quad \left(W_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i < 0 \text{ and } b_i < +\infty, \\ 1 & \text{if} \quad \left(W_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i < 0 \text{ and } b_i = +\infty, \\ (u_k - a)_i^{\frac{1}{2}} & \text{if} \quad \left(W_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i \geq 0 \text{ and } a_i > -\infty, \\ 1 & \text{if} \quad \left(W_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i \geq 0 \text{ and } a_i = -\infty, \end{cases} \tag{3.6}$$

and

$$(E_k)_{ii} = \begin{cases} \left|\left(W_k^T \nabla f_k\right)_i\right| & \text{if } \left(W_k^T \nabla f_k\right)_i \neq 0, \\ 0 & \text{otherwise,} \end{cases}$$

respectively. The role of these matrices in the quadratic (3.5) is related to the application of Newton's method to the system of nonlinear equations arising from the first–order KKT conditions. See [15] for more details.

Exact TRIP SQP algorithms include two approaches to compute $(s_k)_u$: a coupled approach and a decoupled approach.

The decoupled approach [15] is associated with the trust–region subproblem

$$\text{minimize} \Psi_k(s_u) \tag{3.7}$$

$$\text{subject to} \|D_k^{-1} s_u\| \leq \delta_k. \tag{3.8}$$

In this approach the tangential component $(s_k)_u$ has to satisfy a fraction of Cauchy decrease condition associated with the trust–region subproblem (3.7)–(3.8). This condition requires $(s_k)_u$ to give as much decrease on the quadratic (3.5) as the decrease given by $-\bar{D}_k^2 W_k^T \nabla q_k(s_k^{\mathsf{n}})$. It can be proved (see [15, Lemma 6.2]) that such a condition implies

$$\begin{aligned} q_k(s_k^{\mathsf{n}}) \quad &- \quad q_k(s_k^{\mathsf{n}} + W_k (s_k)_u) \\ &\geq \quad \kappa_4 \|\bar{D}_k W_k^T \nabla q_k(s_k^{\mathsf{n}})\| \min\left\{\kappa_5 \|\bar{D}_k W_k^T \nabla q_k(s_k^{\mathsf{n}})\|,\ \kappa_6 \delta_k\right\}, \end{aligned} \tag{3.9}$$

where κ_4, κ_5, and κ_6 are positive constants independent of k.

The use of conjugate gradients to solve trust–region subproblems in unconstrained minimization has been suggested in [49], [51]. An adaptation of these algorithms to compute the tangential component that takes into account the problem structure and the bound constraints is presented in [15] and is given by:

ALGORITHM 3.1 (Exact computation of $s_k = s_k^{\mathsf{n}} + W_k (s_k)_u$ using the decoupled approach).

1 Set $s_u^0 = 0$, $r^0 = -W_k^T \nabla q_k(s_k^{\mathsf{n}})$, $q^0 = \bar{D}_k^2 r^0$, $d^0 = q^0$, and $\epsilon > 0$.

2 For $i = 0, 1, 2, \ldots$ do

2.1 Compute $\gamma^i = \frac{(r^i)^T (q^i)}{(d^i)^T \left(W_k^T H_k W_k + E_k \bar{D}_k^{-2}\right)(d^i)}$.

2.2 Compute

$$\tau^i = \max\Big\{\tau > 0 \quad : \quad \|\bar{D}_k^{-1}(s_u^i + \tau d^i)\| \leq \delta_k, \\ \sigma_k(a - u_k) \leq s_u^i + \tau d^i \leq \sigma_k(b - u_k)\Big\}.$$

2.3 If $\gamma^i \leq 0$, or if $\gamma^i > \tau^i$, then set $(s_k)_u = s_u^i + \tau^i d^i$, where τ^i is given as in 2.2 and go to 3; otherwise set $s_u^{i+1} = s_u^i + \gamma^i d^i$.

2.4 Update the residuals $r^{i+1} = r^i - \gamma^i \left(W_k^T H_k W_k + E_k \bar{D}_k^{-2}\right) d^i$ and $q^{i+1} = \bar{D}_k^2 r^{i+1}$.

2.5 Check truncation criteria: if $\sqrt{\frac{(r^{i+1})^T (q^{i+1})}{(r^0)^T (q^0)}} \leq \epsilon$, set $(s_k)_u = s_u^{i+1}$ and go to 3.

2.6 Compute $\alpha^i = \frac{(r^{i+1})^T (q^{i+1})}{(r^i)^T (q^i)}$ and set $d^{i+1} = q^{i+1} + \alpha^i d^i$.

3 Compute $s_k = s_k^{\mathsf{n}} + W_k (s_k)_u$ and stop.

Step 2 iterates entirely in the space of the u variables. After the u component $(s_k)_u$ of the step has been computed, its y component is calculated in Step 3.

The decoupled approach allows an efficient use of an approximation $\widehat{H}_k$ to the reduced Hessian $W_k^T H_k W_k$. In this case only two linear systems are required, one with $C_y(x_k)^T$ in Step 1 and the other with $C_y(x_k)$ in Step 3, cf. (2.1). If the full Hessian H_k is approximated, then the total number of linear system solves is $2I(k) + 2$, where $I(k)$ is the number of conjugate–gradient iterations. See Table 3.1.

In the decoupled approach only the u part of the tangential component is required to be in the trust region. The coupled approach requires the whole tangential component to be in the trust region. The trust–region subproblem is the following:

$$\text{minimize} \quad \Psi_k(s_u) \tag{3.10}$$

$$\text{subject to} \quad \left\| \begin{pmatrix} (s_k^{\mathsf{n}})_y - C_y(x_k)^{-1} C_u(x_k) s_u \\ \bar{D}_k^{-1} s_u \end{pmatrix} \right\| \leq \delta_k . \tag{3.11}$$

The tangential component $(s_k)_u$ has to satisfy a fraction of Cauchy decrease condition associated with the trust–region subproblem (3.10)–(3.11). This condition requires $(s_k)_u$ to give as much decrease on the quadratic (3.5) as the decrease given by $-\bar{D}_k^2 W_k^T \nabla q_k(s_k^{\mathsf{n}})$. It is shown in [15, Lemma 6.2] that this condition also implies (3.9). Again one can use a conjugate–gradient method to compute the tangential component.

ALGORITHM 3.2 (Exact computation of $s_k = s_k^{\mathsf{n}} + W_k (s_k)_u$ using the coupled approach).

1 Set $s^0 = s_k^{\mathsf{n}}$, $r^0 = -W_k^T \nabla q_k(s_k^{\mathsf{n}})$, $q^0 = \bar{D}_k^2 r^0$, $d^0 = W_k q^0$, and $\epsilon > 0$.

2 For $i = 0, 1, 2, \ldots$ do

2.1 Compute $\gamma^i = \frac{(r^i)^T (q^i)}{(d^i)^T H_k (d^i) + (d^i)_u^T E_k \bar{D}_k^{-2} (d^i)_u}$.

2.2 Compute

$$\tau^i = \max \Big\{ \tau > 0 \quad : \quad \left\| \begin{pmatrix} (s_k^{\mathsf{n}})_y - \tau C_y(x_k)^{-1} C_u(x_k) (d^i)_u \\ \tau \bar{D}_k^{-1} (d^i)_u \end{pmatrix} \right\| \leq \delta_k , \\ \sigma_k (a - u_k) \leq s_u^i + \tau (d^i)_u \leq \sigma_k (b - u_k) \Big\}.$$

2.3 If $\gamma^i \leq 0$, or if $\gamma^i > \tau^i$, then stop and set $s_k = s^i + \tau^i d^i$, where τ^i is given as in 2.2; otherwise set $s^{i+1} = s^i + \gamma^i d^i$.

2.4 Update the residuals $r^{i+1} = r^i - \gamma^i \left(W_k^T H_k d^i + E_k \bar{D}_k^{-2} (d^i)_u\right)$ and $q^{i+1} = \bar{D}_k^2 r^{i+1}$.

2.5 Check truncation criteria: if $\sqrt{\frac{(r^{i+1})^T (q^{i+1})}{(r^0)^T (q^0)}} \leq \epsilon$, set $s_k = s^{i+1}$ and stop.

2.6 Compute $\alpha^i = \frac{(r^{i+1})^T (q^{i+1})}{(r^i)^T (q^i)}$ and set $d^{i+1} = W_k (q^{i+1} + \alpha^i d^i)$.

Note that in Step 2 both the y and the u components of the step are computed. The coupled approach is particularly suitable when an approximation to the full Hessian H_k is used. However, the coupled approach can also be used with an approximation $\widehat{H}_k$ to the

reduced Hessian $W_k^T H_k W_k$. In this case we set

$$H_k = \begin{pmatrix} 0 & 0 \\ 0 & \widehat{H}_k \end{pmatrix}. \tag{3.12}$$

If H_k is given by (3.12), then the definition of W_k implies the equalities

$$H_k d = \begin{pmatrix} 0 \\ \widehat{H}_k d_u \end{pmatrix}, \quad d^T H_k d = d_u^T \widehat{H}_k d_u, \quad \text{and} \quad W_k^T H_k d = \widehat{H}_k d_u, \tag{3.13}$$

and this shows that the reduced Hessian approximation $\widehat{H}_k$ can be used in Algorithm 3.2. The number of linear systems needed is given in Table 3.1.

TABLE 3.1
Number of linear solvers to compute the tangential component. Here $I(k)$ denotes the number of conjugate gradient iterations.

Linear solver	Decoupled		Coupled	
	Reduced $\widehat{H}_k$	Full H_k	Reduced $\widehat{H}_k$	Full H_k
$C_y(x_k)$	1	$I(k)+1$	$I(k)+1$	$I(k)+1$
$C_y(x_k)^T$	1	$I(k)+1$	1	$I(k)+1$

While Table 3.1 indicates that the decoupled approach is more efficient in terms of linear system solves, in applications with ill–conditioned $C_y(x)$ the coupled approach may be favorable. The reason is that in this case the decoupled approach may underestimate the size of $W_k\ (s_k)_u$ vastly and, as a consequence, may require more unsuccessful iterations. See also [15, §5.2.2].

4. Inexact TRIP SQP algorithms. In this section, we assume that the terms involving $C_y(x_k)$ and $C_u(x_k)$ are computed inexactly. This includes the solution of linear systems with $C_y(x_k)$ and $C_y(x_k)^T$ and the matrix–vector products with $C_u(x_k)$ and $C_u(x_k)^T$. The inexact analysis for the quasi–normal component is presented in Section 6 and does not interfere with the analysis developed in this section. We assume that the quasi–normal component s_k^{n}, no matter how it is computed, satisfies conditions (3.1), (3.2), and (3.3). We show in Section 6 that this can be accomplished by a variety of techniques to compute quasi–normal components.

4.1. Representation of the inexactness. The computation of the tangential component requires the calculation of matrix–vector products of the form $W_k d_u$ and $W_k^T d$. Thus we need to compute quantities like

$$-C_y(x_k)^{-1} C_u(x_k) d_u \quad \text{and} \quad -C_u(x_k)^T C_y(x_k)^{-T} d_y.$$

As we have pointed out earlier, often these computations cannot be done exactly. Therefore we have to incorporate errors originating perhaps from finite difference approximations of $C_u(x_k)d_u$ or from the iterative solution of the systems $C_y(x_k)d_y = -C_u(x_k)d_u$.

In practice, the computation of the y component z_y of $z = W_k d_u$ is done as follows:

$$\begin{array}{lrcl} \text{Compute} & v_y & = & -C_u(x_k)d_u + e_u. \\ \text{Solve} & C_y(x_k) z_y & = & v_y + e_y. \end{array} \tag{4.1}$$

The u component of $W_k d_u$ is equal to d_u. In (4.1), e_u and e_y are the error terms accounting for the inexactness in the computation of $C_u(x_k)\,d_u$ and the inexactness in the solution of the linear system $C_y(x_k)z_y = v_y$. Since the u component of W_k is the identity, we only have an error in the y component z_y of $W_k d_u$ computed via (4.1). It holds that

$$z_y = -C_y(x_k)^{-1}C_u(x_k)d_u + C_y(x_k)^{-1}\,(e_u + e_y)\,. \tag{4.2}$$

Of course, the errors e_u and e_y depend in general on d_u.

Similarly, for given d the matrix–vector product $z = W_k^T\,d$ is computed successively by the following procedure:

$$\begin{array}{lrcl} \text{Solve} & C_y(x_k)^T\,v_y & = & -d_y + e_y\,. \\ \text{Compute} & v_u & = & C_u(x_k)^T v_y + e_u\,. \\ \text{Compute} & z & = & v_u + d_u\,. \end{array} \tag{4.3}$$

Again, e_u and e_y are error terms accounting for the inexactness in the computation of $C_u\,(x_k)^T\,v_y$ and the inexactness in the solution of the linear system $C_y(x_k)^T\,v_y = -d_y$. For simplicity we use the same notation, but the error terms in (4.3) are different from those in (4.1). The errors e_u and e_y depend in general on d_y. The computed result can be related to the exact result via the equation

$$z = -C_u(x_k)^T C_y(x_k)^{-T}\,d_y + d_u + C_u(x_k)^T C_y(x_k)^{-T} e_y + e_u\,. \tag{4.4}$$

These two sources of inexactness influence the computation of the following important quantities:

$$W_k^T \nabla q_k(s_k^{\mathsf{n}}) = -C_u(x_k)^T C_y(x_k)^{-T} \nabla_y q_k(s_k^{\mathsf{n}}) + \nabla_u q_k(s_k^{\mathsf{n}})\,, \tag{4.5}$$

and

$$s_k = s_k^{\mathsf{n}} + W_k(s_k)_u = s_k^{\mathsf{n}} + \begin{pmatrix} -C_y(x_k)^{-1}C_u(x_k)(s_k)_u \\ (s_k)_u \end{pmatrix}. \tag{4.6}$$

As we have seen in Section 3, these two calculations are the only ones that appear in the decoupled approach involving derivatives of $C(y,u)$ if an approximation $\widehat{H}_k$ to the reduced Hessian $W_k^T H_k W_k$ is used. This is not the case in all the other situations (see Table 3.1). If an approximation H_k to the full Hessian is used, then we have to account for the inexactness in the calculation of $W_k^T H_k W_k$. Thus, there is no guarantee of monotonicity in the quadratic $\Psi_k(s_u)$ in the conjugate–gradient method, and therefore there is no guarantee that a fraction of Cauchy decrease condition of the form (3.9) would be satisfied. This raises some interesting problems related to the computation of the tangential component that are addressed in Section 7. There we also show that, instead of (4.5) and (4.6), the inexact operations with derivatives of $C(y,u)$ lead to quantities in the form

$$P_k^T \nabla q_k(s_k^{\mathsf{n}}) = -A_k \nabla_y q_k(s_k^{\mathsf{n}}) + \nabla_u q_k(s_k^{\mathsf{n}})\,, \tag{4.7}$$

and

$$s_k = s_k^{\mathsf{n}} + Q_k(s_k)_u = s_k^{\mathsf{n}} + \begin{pmatrix} -B_k(s_k)_u \\ (s_k)_u \end{pmatrix}, \tag{4.8}$$

where A_k and B_k are linear operators representing the inexactness,

$$P_k = \begin{pmatrix} -A_k^T \\ I_{n-m} \end{pmatrix}, \quad \text{and} \quad Q_k = \begin{pmatrix} -B_k \\ I_{n-m} \end{pmatrix}. \tag{4.9}$$

A detailed derivation and analysis of the linear operators A_k and B_k is given in Section 7 together with an extension of Algorithms 3.1 and 3.2 for the computation of the tangential component.

As a consequence of assuming this inexactness, we no longer have condition (3.9). Instead, we have the following condition:

$$\begin{aligned} & q_k(s_k^{\mathsf{n}}) - q_k(s_k^{\mathsf{n}} + Q_k(s_k)_u) \\ & \geq \kappa_4 \|\bar{D}_k^{\mathsf{p}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\| \min\left\{\kappa_5 \|\bar{D}_k^{\mathsf{p}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\|,\ \kappa_6 \delta_k\right\} - \kappa_7 \|C_k\|, \end{aligned} \tag{4.10}$$

where once again κ_4, κ_5, κ_6, and κ_7 are positive and independent of k. The matrix $\bar{D}_k^{\mathsf{p}}$ is a diagonal matrix of order $n-m$ with diagonal elements given by:

$$(\bar{D}_k^{\mathsf{p}})_{ii} = \begin{cases} (b-u_k)_i^{\frac{1}{2}} & \text{if} \quad \left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i < 0 \text{ and } b_i < +\infty, \\ 1 & \text{if} \quad \left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i < 0 \text{ and } b_i = +\infty, \\ (u_k-a)_i^{\frac{1}{2}} & \text{if} \quad \left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i \geq 0 \text{ and } a_i > -\infty, \\ 1 & \text{if} \quad \left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)_i \geq 0 \text{ and } a_i = -\infty, \end{cases}$$

This matrix is the inexact version of $\bar{D}_k$ defined in (3.6). We show in Section 7 how (4.10) can be satisfied. Of course, we still require the tangential component to be feasible with respect to the trust region constraint (3.8) or (3.11) and to the bound constraints (3.4).

In the computation of the actual and predicted decreases, we need to evaluate $J_k s_k$ after the step s_k has been computed. Since we allow the derivatives of $C(y,u)$ to be approximated, we do not have $J_k s_k$ but rather

$$J_k s_k + e_k, \tag{4.11}$$

where e_k is an error term.

4.2. Inexact TRIP SQP algorithms and general assumptions. To decide whether to accept or reject a step s_k, we evaluate the ratio $ared(s_k;\rho_k)/pred(s_k;\rho_k)$, where the actual decrease $ared(s_k;\rho_k)$ is given by

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

and the predicted decrease $pred(s_k;\rho_k)$ by

$$\begin{aligned} pred(s_k;\rho_k) \quad = \quad & L(x_k,\lambda_k;\rho_k) \\ & - \left(q_k(s_k;e_k) + \Delta\lambda_k^T(J_k s_k + e_k + C_k) + \rho_k \|J_k s_k + e_k + C_k\|^2\right). \end{aligned}$$

Here $\Delta\lambda_k = \lambda_{k+1} - \lambda_k$, $L(x,\lambda;\rho)$ is the augmented Lagrangian function

$$L(x,\lambda;\rho) = f(x) + \lambda^T C(x) + \rho C(x)^T C(x),$$

and the quadratic term $q_k(s_k;e_k)$ is given by

$$\begin{aligned} q_k(s_k;e_k) \quad &= \quad \ell_k + \nabla f_k^T s_k + \lambda_k^T(J_k s_k + e_k) + \tfrac{1}{2} s_k^T H_k s_k \\ &= \quad q_k(s_k) + \lambda_k^T e_k. \end{aligned} \tag{4.12}$$

The update of the penalty parameter ρ_k follows El–Alem [21].

ALGORITHM 4.1 (Inexact TRIP SQP algorithms).

1 Choose x_0 such that $a < u_0 < b$, pick $\delta_0 > 0$, and calculate λ_0. Set $\rho_{-1} \geq 1$ and $\epsilon_{tol} > 0$. Choose α_1, η_1, σ, δ_{min}, δ_{max}, and $\bar{\rho}$ such that $0 < \alpha_1, \eta_1, \sigma < 1$, $0 < \delta_{min} \leq \delta_{max}$, and $\bar{\rho} > 0$.

2 For $k = 0, 1, 2, \ldots$ do

2.1 If $\|C_k\| + \|\bar{D}_k^{\mathsf{P}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\| \leq \epsilon_{tol}$, stop and return x_k as an approximate solution for problem (1.1).

2.2 Compute s_k^{n} satisfying (3.1), (3.2), and (3.3). Then, compute $s_k = s_k^{\mathsf{n}} + s_k^{\mathsf{t}} = s_k^{\mathsf{n}} + Q_k (s_k)_u$, where $(s_k)_u$ satisfies (3.4), (3.8) or (3.11), and (4.10).

2.3 Compute λ_{k+1} and set $\Delta\lambda_k = \lambda_{k+1} - \lambda_k$.

2.4 Compute $pred(s_k\,; \rho_{k-1})$.

If $pred(s_k\,; \rho_{k-1}) \geq \frac{\rho_{k-1}}{2}\left(\|C_k\|^2 - \|J_k s_k + e_k + C_k\|^2\right)$ then set $\rho_k = \rho_{k-1}$. Otherwise set

$$\rho_k = \frac{2\left(q_k(s_k\,; e_k) - q_k(0) + \Delta\lambda_k^T (J_k s_k + e_k + C_k)\right)}{\|C_k\|^2 - \|J_k s_k + e_k + C_k\|^2} + \bar{\rho}.$$

2.5 If $\frac{ared(s_k;\rho_k)}{pred(s_k;\rho_k)} < \eta_1$, set

$$\delta_{k+1} = \alpha_1 \max\left\{\|s_k^{\mathsf{n}}\|, \|(\bar{D}_k^{\mathsf{P}})^{-1}(s_k)_u\|\right\} \text{ in the decoupled case or}$$

$$\delta_{k+1} = \alpha_1 \max\left\{\|s_k^{\mathsf{n}}\|, \left\|\begin{pmatrix} -C_y(x_k)^{-1} C_u(x_k)(s_k)_u \\ (\bar{D}_k^{\mathsf{P}})^{-1}(s_k)_u \end{pmatrix}\right\|\right\} \text{ in the}$$

coupled case, and reject s_k.

Otherwise accept s_k and choose δ_{k+1} such that

$$\max\{\delta_{min}, \delta_k\} \leq \delta_{k+1} \leq \delta_{max}.$$

2.6 If s_k was rejected set $x_{k+1} = x_k$ and $\lambda_{k+1} = \lambda_k$. Otherwise set $x_{k+1} = x_k + s_k$ and $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$.

Of course the rules to update the trust radius in the previous algorithm can be much more involved but the above suffices to prove convergence results and to understand the trust–region mechanism. From these rules we have the following lemma (see [15, Lemma 6.1]).

LEMMA 4.1. *Every step satisfies*

$$\|s_k\| \leq \kappa_8 \delta_k \quad \textit{and} \quad \delta_{k+1} \geq \kappa_8 \|s_k\|, \tag{4.13}$$

where κ_8 is a positive constant independent of k.

For the convergence theory we need the following set of assumptions (see [15]). For all iterations k, we assume that $x_k,\ x_k + s_k \in \Omega$, where Ω is an open subset of $\mathbb{R}^n$.

A.1 The functions f, c_i, $i = 1, \ldots, m$ are twice continuously differentiable functions in Ω. Here $c_i(x)$ represents the i–th component of $C(x)$.

A.2 The partial Jacobian $C_y(x)$ is nonsingular for all $x \in \Omega$.

A.3 The functions f, ∇f, $\nabla^2 f$, C, J, $\nabla^2 c_i$, $i = 1, \ldots, m$, are bounded in Ω. The matrix $C_y(x)^{-1}$ is uniformly bounded in Ω.

A.4 The sequences $\{H_k\}$, $\{W_k\}$, and $\{\lambda_k\}$ are bounded.

A.5 The sequence $\{u_k\}$ is bounded.

Assumptions A.1–A.4 reduce to the weakest assumptions required to prove global convergence for equality–constrained optimization (see [14] and the references therein). Assumption A.5 is used in [8], [16] for box–constrained optimization and is trivially satisfied if

$a, b \in \mathbb{R}^{n-m}$. We comment on possible relaxations of some of these conditions in Section 10.

Now we introduce the conditions on the inexact calculations. An important point here is that Algorithms 4.1 can be particularized to satisfy these conditions. See Sections 6, 7, and 8.

IA.1 The sequences $\{A_k\}$ and $\{B_k\}$ are bounded.

IA.2 $\| \left(-C_y(x_k)B_k + C_u(x_k)\right) (s_k)_u\| \leq \theta \min\{\kappa_3\|C_k\|, \delta_k\}$, where $\theta = \min\left\{\frac{1}{\kappa_3}, \frac{\kappa_2}{4}\right\}$.

IA.3 The error term e_k given in (4.11) obeys $\|e_k\| \leq \theta \min\left\{\kappa_3\|C_k\|, \frac{1}{\delta_{max}\kappa_8^2}\|s_k\|^2\right\}$, where θ is given in IA.2.

IA.4 $\lim_j \|(-A_{k_j}^T + C_u(x_{k_j})^T C_y(x_{k_j})^{-T})\nabla_y q_{k_j}(s_{k_j}^{\mathsf{n}})\| = 0$ for all index subsequences $\{k_j\}$ such that $\lim_j \|C_{k_j}\| = 0$.

The Assumption IA.2 imposes a bound on the distance of $Q_k(s_k)_u$ to the null space of the linearized constraints. It is obvious that IA.2 is satisfied when $B_k = C_y(x_k)^{-1}C_u(x_k)$. The Assumption IA.4 is only needed to derive Theorem 5.1 and restricts the accuracy of the reduced gradient calculation. We will be more precise later. This assumption is satisfied if $A_k = C_u(x_k)^T C_y(x_k)^{-T}$.

For the rest of this paper we suppose that Assumptions A.1–A.5 and Conditions IA.1–IA.4 on the inexactness are always satisfied.

5. Global convergence. For the global convergence of the inexact TRIP SQP Algorithms 4.1 we need conditions (3.1), (3.2), and (3.3) on the quasi–normal component s_k^{n} and condition (4.10) on the tangential component $(s_k)_u$. The following lemma states a lower bound on the decrease given by s_k on the linearized constraints. The need for this lemma is the fact that, due to the inexactness assumption, s_k^{t} might not lie in the null space of J_k.

LEMMA 5.1. *The step s_k satisfies*

$$\|C_k\|^2 - \|J_k s_k + e_k + C_k\|^2 \geq \frac{\kappa_2}{2}\|C_k\| \min\{\kappa_3\|C_k\|, \delta_k\}. \tag{5.1}$$

Proof. From IA.2 we get

$$\| \left(-C_y(x_k)B_k + C_u(x_k)\right) (s_k)_u\|^2 \leq \frac{1}{\kappa_3}\kappa_3\|C_k\|\frac{\kappa_2}{4} \min\{\kappa_3\|C_k\|, \delta_k\}.$$

From IA.3, (4.13), and $\delta_k \leq \delta_{max}$ we obtain

$$\|e_k\|^2 \leq \frac{1}{\kappa_3}\kappa_3\|C_k\|\frac{\kappa_2}{4} \min\{\kappa_3\|C_k\|, \delta_k\}.$$

Using these inequalities, (3.3), $s_k = s_k^{\mathsf{n}} + Q_k(s_k)_u$, and the form (4.9) of Q_k, we have

$$\begin{aligned}
\|C_k\|^2 - \|J_k s_k + e_k + C_k\|^2 &\geq \|C_k\|^2 - \|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\|^2 \\
&\quad - \| \left(-C_y(x_k)B_k + C_u(x_k)\right) (s_k)_u\|^2 - \|e_k\|^2 \\
&\geq \tfrac{\kappa_2}{2}\|C_k\| \min\{\kappa_3\|C_k\|, \delta_k\}.
\end{aligned}$$

□

We also need the following three inequalities.

LEMMA 5.2. *There exist positive constants κ_9, κ_{10}, and κ_{11} independent of k such that*

$$q_k(0) - q_k(s_k^{\mathsf{n}}) - \Delta\lambda_k^T\left(J_k s_k + e_k + C_k\right) \geq -\kappa_9\|C_k\|, \tag{5.2}$$

$$|ared(s_k;\rho_k) - pred(s_k;\rho_k)| \leq \kappa_{10}\Big(\|s_k\|^2 + \rho_k\|s_k\|^3 + \rho_k\|C_k\|\,\|s_k\|^2\Big) \tag{5.3}$$

and

$$|ared(s_k;\rho_k) - pred(s_k;\rho_k)| \leq \kappa_{11}\rho_k\|s_k\|^2. \tag{5.4}$$

Proof. For the proof of the first inequality, we have

$$q_k(0) - q_k(s_k^{\mathsf{n}}) \;=\; -(\nabla_x \ell_k)^T s_k^{\mathsf{n}} - \frac{1}{2}{s_k^{\mathsf{n}}}^T H_k s_k^{\mathsf{n}} \;\geq\; -\left(\|\nabla_x \ell_k\| + \frac{1}{2}\|H_k\|\,\|s_k^{\mathsf{n}}\|\right)\|s_k^{\mathsf{n}}\|.$$

Also, since $\|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\| \leq \|C_k\|$, we use IA.2 and IA.3, $s_k = s_k^{\mathsf{n}} + Q_k(s_k)_u$ and the form (4.9) of Q_k and get

$$\begin{aligned} -\Delta\lambda_k^T(J_k s_k + e_k + C_k) &\geq -\|\Delta\lambda_k\|\Big(\|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\| \\ &\quad + \|\left(-C_y(x_k)B_k + C_u(x_k)\right)(s_k)_u\| + \|e_k\|\Big) \\ &\geq -3\,\|\Delta\lambda_k\|\,\|C_k\|. \end{aligned}$$

Using (3.2), the fact that $\|s_k^{\mathsf{n}}\| \leq \delta_{max}$, and Assumptions A.3 and A.4, we obtain the desired inequality (5.2).

Now we prove the other two inequalities. If we add and subtract $\ell(x_{k+1},\lambda_k)$ to $ared(s_k;\rho_k) - pred(s_k;\rho_k)$ and expand $\ell(\cdot,\lambda_k)$ around x_k, we get

$$\begin{aligned} ared(s_k;\rho_k) - pred(s_k;\rho_k) &= \tfrac{1}{2}s_k^T\left(H_k - \nabla_x^2\ell(x_k + \pi_k^1 s_k,\lambda_k)\right)s_k \\ &\quad + \Delta\lambda_k^T(-C_{k+1} + C_k + J_k s_k) + \mathcal{O}(\|s_k\|^2) \\ &\quad - \rho_k\Big(\|C_{k+1}\|^2 - \|J_k s_k + C_k\|^2 - \mathcal{O}(\|s_k\|^3)\Big) \end{aligned}$$

for some $\pi_k^1 \in (0,1)$. The terms $\mathcal{O}(\|s_k\|^2)$ and $\mathcal{O}(\|s_k\|^3)$ come from IA.3. The rest of the proof follows from [15, Lemma 6.5]. ∎

The following four lemmas bound the predicted decrease.

LEMMA 5.3. *If $(s_k)_u$ satisfies (4.10), then the predicted decrease in the merit function satisfies*

$$\begin{aligned} pred(s_k;\rho) &\geq \kappa_4\|\bar{D}_k^{\mathsf{p}}P_k^T\nabla q_k(s_k^{\mathsf{n}})\|\min\left\{\kappa_5\|\bar{D}_k^{\mathsf{p}}P_k^T\nabla q_k(s_k^{\mathsf{n}})\|, \kappa_6\delta_k\right\} \\ &\quad -(\kappa_7 + \kappa_9 + \nu)\|C_k\| + \rho\Big(\|C_k\|^2 - \|J_k s_k + e_k + C_k\|^2\Big), \end{aligned} \tag{5.5}$$

for every $\rho > 0$, where from Assumption A.4, ν is a uniform bound for $\|\lambda_k\|$.

Proof. The inequality (5.5) follows from a direct application of the form (4.12) of $q_k(s_k;e_k)$ followed by (4.10), (5.2), A.4, and IA.3. ∎

LEMMA 5.4. *Assume that $(s_k)_u$ satisfies (4.10) and that $\|\bar{D}_k^{\mathsf{p}}P_k^T\nabla q_k(s_k^{\mathsf{n}})\| + \|C_k\| > \epsilon_{tol}$. There exists a positive constant α independent of k such that, if $\|C_k\| \leq \alpha\delta_k$ then*

$$\begin{aligned} pred(s_k;\rho) &\geq \tfrac{\kappa_4}{2}\|\bar{D}_k^{\mathsf{p}}P_k^T\nabla q_k(s_k^{\mathsf{n}})\|\min\left\{\kappa_5\|\bar{D}_k^{\mathsf{p}}P_k^T\nabla q_k(s_k^{\mathsf{n}})\|, \kappa_6\delta_k\right\} \\ &\quad + \rho\Big(\|C_k\|^2 - \|J_k s_k + e_k + C_k\|^2\Big), \end{aligned} \tag{5.6}$$

for every $\rho > 0$.

Proof. The proof is the same as the proof of Lemma 7.2 in [15]. ∎

We can use Lemma 5.4 with $\rho = \rho_{k-1}$, and conclude that if $\|\bar{D}_k^{\mathsf{P}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\| + \|C_k\| > \epsilon_{tol}$ and $\|C_k\| \leq \alpha\delta_k$, then the penalty parameter at the current iterate does not need to be increased. This is equivalent to Lemma 7.7 in [14]. The next lemma states an equivalent result to Lemma 7.8 in [14].

LEMMA 5.5. *Let $(s_k)_u$ satisfy (4.10) and $\|\bar{D}_k^{\mathsf{P}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\| + \|C_k\| > \epsilon_{tol}$. There exists a positive constant α such that, if $\|C_k\| \leq \alpha\delta_k$ then*

$$pred(s_k\,;\rho_k) \geq \kappa_{12}\delta_k\,, \tag{5.7}$$

where κ_{12} is positive and does not depend on k.

Proof. See [15, Lemma 7.3]. □

LEMMA 5.6. *The predicted decrease satisfies*

$$pred(s_k\,;\rho_k) \geq \frac{\rho_k}{2}\left(\|C_k\|^2 - \|J_k s_k + e_k + C_k\|^2\right), \tag{5.8}$$

for all k.

Proof. It follows directly from the Scheme 2.4 that updates ρ_k. □

Now we use the theory given in [14], [15] to state the following result. This result shows that for a subsequence of the iterates, the first–order KKT conditions (2.2)–(2.3) of problem (1.1) are satisfied in the limit.

THEOREM 5.1. *The sequences of iterates generated by the inexact TRIP SQP Algorithms 4.1 satisfy*

$$\liminf_{k\to\infty} \left(\|D_k W_k^T \nabla f_k\| + \|C_k\|\right) = 0. \tag{5.9}$$

Proof. First, we use the theory given in [14] to show that:

$$\liminf_{k\to\infty} \left(\|\bar{D}_k^{\mathsf{P}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\| + \|C_k\|\right) = 0.$$

In fact, Lemmas 7.9–7.13 and 8.2 as well as Theorems 8.1, 8.3, and 8.4 in [14] can be applied based on (3.2), (4.10), (4.13), (5.1), (5.2), (5.3), (5.4), (5.7), (5.8) and on the fact that if $\|\bar{D}_k^{\mathsf{P}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\| + \|C_k\| > \epsilon_{tol}$ and $\|C_k\| \leq \alpha\delta_k$, then the penalty parameter at the current iterate does not need to be increased. Thus this result is just a restating of Theorem 8.4 of [14]. So, there exists an index subsequence $\{k_i\}$ such that

$$\lim_{i\to\infty} \left(\|\bar{D}_{k_i}^{\mathsf{P}} P_{k_i}^T \nabla q_{k_i}(s_{k_i}^{\mathsf{n}})\| + \|C_{k_i}\|\right) = 0.$$

Now we apply Assumption IA.4 and the forms (2.1) and (4.9) of $W_k = W(x_k)$ and P_k, to obtain

$$\lim_{i\to\infty} \left(P_{k_i} - W_{k_i}\right)^T \nabla q_{k_i}(s_{k_i}^{\mathsf{n}}) = 0.$$

Using this and the continuity of $D(x)W(x)^T\nabla f(x)$ we get

$$\lim_{i\to\infty} \left(\|\bar{D}_{k_i} W_{k_i}^T \nabla q_{k_i}(s_{k_i}^{\mathsf{n}})\| + \|C_{k_i}\|\right) = 0.$$

The rest of the proof is given in the last paragraph of the proof of Theorem 7.1 in [15]. □

The condition imposed in IA.4 is related to the computation of the reduced gradient. If the adjoint multipliers are used, then this condition can be interpreted as a restriction on how accurate these multipliers have to be computed. We comment on this again in Sections 7 and 8.

6. Computation of the quasi–normal component. The quasi–normal component s_k^{n} is an approximate solution of the trust–region subproblem

$$\begin{aligned} &\text{minimize} && \tfrac{1}{2}\|C_y(x_k)(s^{\mathsf{n}})_y + C_k\|^2 \\ &\text{subject to} && \|(s^{\mathsf{n}})_y\| \le \delta_k\,, \end{aligned} \tag{6.1}$$

and it is required to satisfy the conditions (3.1), (3.2), and (3.3). The property (3.2) is a consequence of (3.3). In fact, using $\|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\| \le \|C_k\|$ and the boundedness of $\{C_y(x_k)^{-1}\}$ we find that

$$\|s_k^{\mathsf{n}}\| \le \|C_y(x_k)^{-1}\| \left(\|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\| + \|C_k\|\right) \le 2\|C_y(x_k)^{-1}\|\,\|C_k\|\,.$$

Whether the property (3.3) holds depends on the way in which the quasi–normal component is computed. We show below that (3.3) is satisfied for a variety of techniques to compute s_k^{n}. We concentrate on methods that are suitable for the large scale case and do not require the matrix $C_y(x_k)$ in explicit form. The first two groups of methods tackle the trust–region subproblem (6.1) directly. The first group of methods are Krylov subspace methods that require the computation of matrix–vector products $C_y(x_k)s_y$ and $C_y(x_k)^T s_y$, while the second group of methods only require $C_y(x_k)s_y$. The third group of methods compute steps by solving the linear system $C_y(x_k)(s^{\mathsf{n}})_y = -C_k$ approximately. The trust–region constraint is enforced by scaling the solution.

6.1. Subspace methods. There are various ways to compute the quasi–normal component s_k^{n} for large scale problems based on Krylov subspace methods. For example, one can use the conjugate–gradient method applied to the normal equation as suggested for the general quadratic case in [49], [51], or one can use the Lanczos bidiagonalization as described in [25]. Both methods compute an approximate solution of (6.1) from a subspace that contains the negative gradient $-C_y(x_k)^T C_k$ of the least squares functional. Thus, the steps s_k^{n} generated by these methods satisfy $\|s_k^{\mathsf{n}}\| \le \delta_k$ and

$$\begin{aligned} &\frac{1}{2}\|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\|^2 \\ &\le\ \min\left\{\frac{1}{2}\|C_y(x_k)s + C_k\|^2\ :\ s = -tC_y(x_k)^T C_k\ ,\ \|s\| \le \delta_k\right\}\,. \end{aligned} \tag{6.2}$$

We can appeal to a classical result due to Powell, see [44, Theorem 4], [37, Lemma 4.8], to prove the following result:

LEMMA 6.1. *If $(s_k^{\mathsf{n}})_y$ satisfies (6.2), then there exist positive constants κ_2 and κ_3, independent of k, such that*

$$\|C_k\|^2 - \|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\|^2 \ge \kappa_2\|C_k\|\min\{\kappa_3\|C_k\|, \delta_k\}.$$

Another method to solve large scale trust–region subproblems is analyzed in [47]. In this approach the trust–region subproblem (6.1) is reformulated as an eigenvalue problem which is then solved by the implicitly restarted Lanczos method. This method and the method in [25] compute a step that satisfies the fraction of optimal decrease condition, i.e. these methods compute steps $(s_k^{\mathsf{n}})_y$ satisfying

$$\|C_k\|^2 - \|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\|^2 \ge \beta\Big(\|C_k\|^2 - \|C_y(x_k)(s_*^{\mathsf{n}})_y + C_k\|^2\Big),$$

where $(s_*^{\mathsf{n}})_y$ is the solution of (6.1). Thus, the method in [47] yields a step satisfying (3.3).

The previous approaches require the evaluation of $C_y(x_k)v$ and $C_y(x_k)^T u$ for given v and u. For some applications, the evaluation of $C_y(x_k)^T u$ is more expensive than the application of $C_y(x_k)v$, and therefore it may be more efficient to use methods that avoid the use of $C_y(x_k)^T u$. In this case one can apply nonsymmetric Krylov subspace methods based on minimum residual approximations, such as GMRES [46]. In the context of nonlinear system solving the use of such methods is described e.g. in [4].

If GMRES is used and if

$$\frac{1}{2}C_k^T\Big(C_y(x_k)^T + C_y(x_k)\Big)C_k \geq \beta\|C_k\|^2 \tag{6.3}$$

holds with $\beta > 0$, then

$$\|C_k\|^2 - \|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\|^2 \geq \kappa_2\|C_k\|\min\{\kappa_3\|C_k\|, \delta_k\},$$

where κ_2 and κ_3 are positive constants that do not depend on k. The condition (6.3) is implied by the positive definiteness of the symmetric part of $C_y(x_k)$, a condition also important for the convergence of nonsymmetric Krylov subspace methods. A proof of this result and more details concerning the use of these methods can be found in [52].

6.2. Scaled approximate solutions. An alternative to the previous procedures is to compute an approximate solution $\hat{s}_k^{\mathsf{n}}$ of the linear system $C_y(x_k)s = -C_k$ and to scale this step back into the trust region, i.e. to set

$$s_k^{\mathsf{n}} = \begin{pmatrix} \xi_k\hat{s}_k^{\mathsf{n}} \\ 0 \end{pmatrix}, \quad \text{where } \xi_k = \begin{cases} 1 & \text{if } \|\hat{s}_k^{\mathsf{n}}\| \leq \delta_k, \\ \frac{\delta_k}{\|\hat{s}_k^{\mathsf{n}}\|} & \text{otherwise.} \end{cases} \tag{6.4}$$

We assume that the linear system $C_y(x_k)s = -C_k$ is solved inexactly and that the residual vector satisfies $\|C_y(x_k)\hat{s}_k^{\mathsf{n}} + C_k\| \leq \epsilon\|C_k\|$ with $\epsilon < 1$. Then we have the following result (the proof can be found in [52]).

LEMMA 6.2. *If $\|C_y(x_k)\hat{s}_k^{\mathsf{n}} + C_k\| \leq \epsilon\|C_k\|$ with $\epsilon < 1$ for all k, then the quasi–normal component (3.1) satisfies*

$$\|C_k\|^2 - \|C_y(x_k)(s_k^{\mathsf{n}})_y + C_k\|^2 \geq \kappa_2\|C_k\|\min\{\kappa_3\|C_k\|, \delta_k\},$$

where κ_2 and κ_3 are positive constants independent of k.

7. Computation of the tangential component. Ideally, the tangential component minimizes the quadratic model $\Psi_k(s_u)$ in the null space of the linearized constraints subject to the trust region and the bound constraints. Since the null space of the linearized constraints is characterized by W_k, the exact tangential component has the form $s_k^{\mathsf{t}} = W_k(s_k)_u$. The u part of the tangential component is computed by a conjugate–gradient method and its computation requires the calculation of matrix–vector products $W_k d_u$ and $W_k^T d$. We assume that these calculations are inexact.

7.1. Reduced gradient calculation. For the computation of the tangential component we first have to compute the reduced gradient $W_k^T\nabla q_k(s_k^{\mathsf{n}})$ of the quadratic model $\Psi_k(s_u)$. If this is done using (4.3), then we have an approximation to the reduced gradient $W_k^T\nabla q_k(s_k^{\mathsf{n}})$ of the form:

$$W_k^T\nabla q_k(s_k^{\mathsf{n}}) + e_A, \tag{7.1}$$

where the error term e_A depends on $W_k^T \nabla q_k(s_k^{\mathsf{n}})$. By using the error term in (4.4), we find that

$$\|e_A\| \leq \|C_u(x_k)^T C_y(x_k)^{-T}\| \, \|(e_A)_y\| + \|(e_A)_u\|. \tag{7.2}$$

We can interpret the inexact computation of $W_k^T \nabla q_k(s_k^{\mathsf{n}})$ as the exact solution of a perturbed equation. If we set

$$E_A = \frac{1}{\|\nabla_y q_k(s_k^{\mathsf{n}})\|^2} \, e_A \, (\nabla_y q_k(s_k^{\mathsf{n}}))^T,$$

then

$$\Big(- C_u(x_k)^T C_y(x_k)^{-T} + E_A \Big) \nabla_y q_k(s_k^{\mathsf{n}}) = -C_u(x_k)^T C_y(x_k)^{-T} \nabla_y q_k(s_k^{\mathsf{n}}) + e_A.$$

Thus we can define $A_k = C_y(x_k)^{-1} C_u(x_k) - E_A^T$ and

$$P_k = \begin{pmatrix} -A_k^T \\ I_{n-m} \end{pmatrix} = \begin{pmatrix} -C_y(x_k)^{-1} C_u(x_k) + E_A^T \\ I_{n-m} \end{pmatrix}. \tag{7.3}$$

With this definition we can write $W_k^T \nabla q_k(s_k^{\mathsf{n}}) + e_A = P_k^T \nabla q_k(s_k^{\mathsf{n}})$. The linear operator A_k satisfies

$$\begin{aligned} \| - A_k + C_u(x_k)^T C_y(x_k)^{-T}\| \;&=\; \|E_A^T\| \;\leq\; \|e_A\| / \|\nabla_y q_k(s_k^{\mathsf{n}})\| \\ &\leq \Big(\|C_u(x_k)^T C_y(x_k)^{-T}\| \, \|(e_A)_y\| + \|(e_A)_u\| \Big) / \|\nabla_y q_k(s_k^{\mathsf{n}})\| \end{aligned} \tag{7.4}$$

and

$$\begin{aligned} \big\| \big(- A_k + C_u(x_k)^T C_y(x_k)^{-T} \big) \nabla_y q_k(s_k^{\mathsf{n}}) \big\| \;&=\; \|E_A^T \nabla_y q_k(s_k^{\mathsf{n}})\| \;=\; \|e_A\| \\ &\leq \Big(\|C_u(x_k)^T C_y(x_k)^{-T}\| \, \|(e_A)_y\| + \|(e_A)_u\| \Big). \end{aligned} \tag{7.5}$$

If for given $\nabla_y q_k(s_k^{\mathsf{n}})$, the error terms in the computation of the reduced gradient via (4.3) obey

$$\max \Big\{ \|(e_A)_y\|, \|(e_A)_u\| \Big\} \;\leq\; \eta \, \|C_k\|, \tag{7.6}$$

then (7.5) and Assumptions A.3–A.4 imply the Condition IA.4. Moreover, if

$$\max \Big\{ \|(e_A)_y\|, \|(e_A)_u\| \Big\} \leq \eta \, \|\nabla_y q_k(s_k^{\mathsf{n}})\|, \tag{7.7}$$

then (7.4) and Assumptions A.3–A.4 imply the boundedness of $\{A_k\}$. This gives the first part of Condition IA.1.

7.2. Conjugate–gradient algorithms. In the following, we formulate extensions of the conjugate–gradient Algorithms 3.1 and 3.2 for the computation of the tangential component. To keep the presentation simple, we continue to use the notation W_k and W_k^T. However, whenever matrix–vector products with W_k or W_k^T are computed, we assume that this is done using (4.1) or (4.3). The degree of inexactness, i.e. the size of the error terms e_y and e_u, is specified later. The reduced gradient $W_k^T \nabla q_k(s_k^{\mathsf{n}})$ of the quadratic model $\Psi_k(s_u)$ is assumed to be computed by (7.1) with errors $(e_A)_y$, $(e_A)_u$ satisfying (7.6) and (7.7).

In the case where an approximation $\widehat{H}_k$ to the reduced Hessian $W_k^T H_k W_k$ is used, the quadratic

$$-\left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T s_u - \frac{1}{2} s_u^T \left(\widehat{H}_k + E_k(\bar{D}_k^{\mathsf{P}})^{-2}\right) s_u$$

is reduced at every iteration of the conjugate–gradient algorithm. If we use an approximation H_k to the full Hessian we have to compute matrix–vector multiplications with $W_k^T H_k W_k$. One of the consequences of the inexactness is that the quadratic evaluated at the iterates of the conjugate–gradient algorithms is not guaranteed to decrease. For instance, the inexact application of W_k and W_k^T may cause $W_k^T H_k W_k$ to be nonsymmetric. Hence we need to measure the Cauchy decrease at Step 3 of the algorithm. The extension of the conjugate–gradient Algorithm 3.1 is given below.

ALGORITHM 7.1 (Inexact computation of $s_k = s_k^{\mathsf{n}} + W_k (s_k)_u$ (decoupled case)).

1 Set $s_u^0 = 0$, $r^0 = -P_k^T \nabla q_k(s_k^{\mathsf{n}})$, $q^0 = (\bar{D}_k^{\mathsf{P}})^2 r^0$, $d^0 = q^0$, and $\epsilon > 0$.

2 For $i = 0, 1, 2, \ldots$ do

2.1 Compute

$$\gamma^i = \begin{cases} \dfrac{(r^i)^T (q^i)}{(d^i)^T \left(\widehat{H}_k + E_k(\bar{D}_k^{\mathsf{P}})^{-2}\right)(d^i)} & \text{(reduced Hessian)}, \\ \dfrac{(r^i)^T (q^i)}{(d^i)^T \left(W_k^T H_k W_k + E_k(\bar{D}_k^{\mathsf{P}})^{-2}\right)(d^i)} & \text{(full Hessian)}. \end{cases}$$

2.2 Compute

$$\tau^i = \max\Big\{\tau > 0 \quad : \quad \|(\bar{D}_k^{\mathsf{P}})^{-1}(s_u^i + \tau d^i)\| \le \delta_k\,, \\ \sigma_k(a - u_k) \le s_u^i + \tau d^i \le \sigma_k(b - u_k)\Big\}.$$

2.3 If $\gamma^i \le 0$, or if $\gamma^i > \tau^i$, then set $s_u^* = s_u^i + \tau^i d^i$, where τ^i is given as in 2.2 and go to 3; otherwise set $s_u^{i+1} = s_u^i + \gamma^i d^i$.

2.4 Update the residuals:

$$r^{i+1} = \begin{cases} r^i - \gamma^i \left(\widehat{H}_k + E_k(\bar{D}_k^{\mathsf{P}})^{-2}\right) d^i & \text{(reduced Hessian)}, \\ r^i - \gamma^i \left(W_k^T H_k W_k + E_k(\bar{D}_k^{\mathsf{P}})^{-2}\right) d^i & \text{(full Hessian)}, \end{cases}$$

and $q^{i+1} = (\bar{D}_k^{\mathsf{P}})^2 r^{i+1}$.

2.5 Check truncation criteria: if $\sqrt{\frac{(r^{i+1})^T (q^{i+1})}{(r^0)^T (q^0)}} \le \epsilon$, set $s_u^* = s_u^{i+1}$ and go to 3.

2.6 Compute $\alpha^i = \frac{(r^{i+1})^T (q^{i+1})}{(r^i)^T (q^i)}$ and set $d^{i+1} = q^{i+1} + \alpha^i d^i$.

3 Compute $W_k s_u^*$.

If a reduced Hessian approximation is used, set $(s_k)_u = s_u^*$ and $s_k = s_k^{\mathsf{n}} + W_k s_u^*$.

If a full Hessian approximation is used and if

$$\begin{aligned} -\left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T s_u^* \quad &- \quad \tfrac{1}{2}(W_k s_u^*)^T H_k (W_k s_u^*) \\ &< \quad -\left(W_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T s_u^1 - \tfrac{1}{2} {s_u^1}^T W_k^T H_k W_k s_u^1\,, \end{aligned}$$

then set $(s_k)_u = s_u^1$ and $s_k = s_k^{\mathsf{n}} + W_k s_u^1$. Otherwise $(s_k)_u = s_u^*$ and $s_k = s_k^{\mathsf{n}} + W_k s_u^*$.

The extension for the coupled approach is analogous and is omitted.

7.3. Distance to the null space of the linearized constraints. Let $(s_k^{\mathbf{t}})_y$ and $(s_k^{\mathbf{t}})_u = (s_k)_u$ be the quantities computed by Algorithm 7.1. Since $W_k(s_k)_u$ is not computed exactly in Step 3, it holds that

$$\begin{aligned}(s_k^{\mathbf{t}})_y &= -C_y(x_k)^{-1}C_u(x_k)(s_k)_u + C_y(x_k)^{-1}\left((e_B)_u + (e_B)_y\right)\\ &= -C_y(x_k)^{-1}C_u(x_k)(s_k)_u + e_B,\end{aligned}$$

where the error term e_B depends on $(s_k)_u$ and satisfies

$$\|e_B\| \le \|C_y(x_k)^{-1}\|\Big(\|(e_B)_u\| + \|(e_B)_y\|\Big), \tag{7.8}$$

cf. (4.2). As before, we can interpret the inexact computation $(s_k^{\mathbf{t}})_y$ of $s_k^{\mathbf{t}} = W_k(s_k)_u$ as the exact solution of a perturbed equation. If

$$E_B = \frac{1}{\|(s_k)_u\|^2}\, e_B\, (s_k)_u^T,$$

then

$$\Big(-C_y(x_k)^{-1}C_u(x_k) + E_B\Big)(s_k)_u = -C_y(x_k)^{-1}C_u(x_k)(s_k)_u + e_B = (s_k^{\mathbf{t}})_y.$$

We define $B_k = C_y(x_k)^{-1}C_u(x_k) - E_B$ and

$$Q_k = \begin{pmatrix} -B_k \\ I_{n-m}\end{pmatrix} = \begin{pmatrix} -C_y(x_k)^{-1}C_u(x_k) + E_B \\ I_{n-m}\end{pmatrix}. \tag{7.9}$$

With this definition, we can write

$$s_k^{\mathbf{t}} = Q_k(s_k)_u.$$

The linear operator B_k satisfies

$$\begin{aligned}\|-B_k + C_y(x_k)^{-1}C_u(x_k)\| &= \|E_B\| \le \|e_B\|/\|(s_k)_u\|\\ &\le \Big(\|C_y(x_k)^{-1}\|\left(\|(e_B)_u\| + \|(e_B)_y\|\right)\Big)/\|(s_k)_u\|\end{aligned}$$

and

$$\begin{aligned}\Big\|\big(-C_y(x_k)B_k + C_u(x_k)\big)(s_k)_u\Big\| = \|C_y(x_k)E_B(s_k)_u\| &= \|C_y(x_k)e_B\|\\ &\le \|(e_B)_u\| + \|(e_B)_y\|.\end{aligned} \tag{7.10}$$

If the error terms in the computation of $(s_k^{\mathbf{t}})_y$ using (4.1) obey

$$\max\Big\{\|(e_B)_y\|, \|(e_B)_u\|\Big\} \le \frac{\theta}{2}\min\{\kappa_3\|C_k\|, \delta_k\}, \tag{7.11}$$

where θ and κ_3 are defined as in IA.2, then one can see from (7.10) that B_k satisfies Condition IA.2. Moreover, since $\{C_y(x_k)^{-1}\}$ is bounded, if

$$\max\Big\{\|e_y\|, \|e_u\|\Big\} \le \eta\,\|(s_k)_u\|,$$

then (7.3) implies the boundedness of $\{B_k\}$, cf. IA.1.

7.4. Cauchy decrease condition. Now we establish the decrease condition (4.10). We analyze reduced and full Hessians approximations separately.

In the reduced Hessian approximation case, an approximation $\widehat{H}_k$ for $W_k^T H_k W_k$ is used and all the calculations of Step 2 of Algorithm 7.1 are performed exactly. In this case $(s_k)_u$ satisfies the following condition

$$\begin{aligned} & -\left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T (s_k)_u - \frac{1}{2}(s_k)_u^T \widehat{H}_k (s_k)_u \\ & \geq \kappa_4 \|\bar{D}_k^{\mathsf{p}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\| \min\left\{\kappa_5 \|\bar{D}_k^{\mathsf{p}} P_k^T \nabla q_k(s_k^{\mathsf{n}})\|,\ \kappa_6 \delta_k\right\}. \end{aligned} \tag{7.12}$$

This is just a consequence of Powell's classical result ([44, Theorem 4], [37, Lemma 4.8]) adapted to the current context [15, Lemma 6.2].

Now recall that we need to establish (4.10), where the left hand side is given by

$$-\left(Q_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T (s_k)_u - \frac{1}{2}(s_k)_u^T Q_k^T H_k Q_k (s_k)_u .$$

However, in (7.12) the left hand side is

$$-\left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T (s_k)_u - \frac{1}{2}(s_k)_u^T \widehat{H}_k (s_k)_u .$$

First we use (3.12) and (7.9) to write $\frac{1}{2}(s_k)_u^T \widehat{H}_k (s_k)_u = \frac{1}{2}(s_k)_u^T Q_k^T H_k Q_k (s_k)_u$. Then we relate the inexactness represented by P_k and Q_k with the constraint residual $\|C_k\|$. In fact, by using (7.3) and (7.9), we write

$$\begin{aligned} & \left(P_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T (s_k)_u \\ & = -\nabla q_k(s_k^{\mathsf{n}})^T Q_k (s_k)_u - \nabla q_k(s_k^{\mathsf{n}})^T \begin{pmatrix} E_A^T \\ 0 \end{pmatrix} (s_k)_u + \nabla q_k(s_k^{\mathsf{n}})^T \begin{pmatrix} E_B \\ 0 \end{pmatrix} (s_k)_u \\ & = -\left(Q_k^T \nabla q_k(s_k^{\mathsf{n}})\right)^T (s_k)_u - e_A^T (s_k)_u + e_B^T \nabla_y q_k(s_k^{\mathsf{n}}) . \end{aligned}$$

The error bounds (7.2), (7.6), (7.8), (7.11), and Assumptions A.3–A.4 give

$$e_A^T (s_k)_u - e_B^T \nabla_y q_k(s_k^{\mathsf{n}}) \ \leq\ \|e_A\|\,\|(s_k)_u\| + \|e_B\|\,\|\nabla_y q_k(s_k^{\mathsf{n}})\| \ \leq\ \kappa_7 \|C_k\|,$$

where κ_7 is a positive constant independent of k. Hence we have proved (4.10). The analysis for the full Hessian is given in [52].

8. Computation of the Lagrange multiplier estimates. Note that the only assumption on λ_k required to prove the global convergence result (5.9) is the boundedness of the sequence $\{\lambda_k\}$ (see Assumption A.4).

A choice of λ_k that is available from the reduced gradient calculation of $q_k(s)$ is $\lambda_k = -C_y(x_k)^{-T} \nabla_y q_k(s_k^{\mathsf{n}})$. Due to inexactness λ_k actually satisfies

$$-C_y(x_k)^T \lambda_k = \nabla_y q_k(s_k^{\mathsf{n}}) + e_k^{\lambda},$$

where e_k^{λ} is the corresponding residual vector. From Assumptions A.3–A.4, if $\{e_k^{\lambda}\}$ is bounded, then $\{\lambda_k\}$ is also bounded.

Another choice for λ_k is $\lambda_k = -C_y(x_k)^{-T} \nabla_y f_k$. We refer the reader to Section 10.3 of [15] for a discussion on these choices of λ_k.

9. Numerical experiments. We implemented the inexact TRIP SQP Algorithms 4.1 and compared them with the exact TRIP SQP Algorithms proposed in [15]. The numerical test computations were done on a Sun Sparcstation 10 in double precision `Fortran 77`. We tested our algorithms on examples that have the structure described in this paper. The numerical results are satisfactory and revealed interesting properties of the algorithms. Two examples are described in this section. Numerical results with two other examples are are documented in [7], [27].

We use the formula (6.4) to compute the quasi–normal component, and conjugate-gradients to calculate the tangential component. The scheme used to update the trust radius and the inexact form of diagonal scaling matrices D_k and $\bar{D}_k$ are the same as in [15]. We have used $\sigma_k = \sigma = 0.99995$ for all k; $\delta_0 = 1$ as initial trust radius; $\rho_{-1} = 1$ and $\bar{\rho} = 10^{-2}$ in the penalty scheme. The tolerances used were $\epsilon_{tol} = 10^{-8}$ for the main iteration, Algorithm 4.1, and $\epsilon = 10^{-4}$ for the conjugate–gradient Algorithm 7.1 and the corresponding coupled version.

The tolerance for inexact solvers with $C_y(x_k)$ was set to

$$\min\left\{10^{-2},\ 10^{-2}\min\{\|C_k\|,\ \delta_k\}\right\}, \tag{9.1}$$

and for inexact solvers with $C_y(x_k)^T$ to

$$\min\left\{10^{-2},\ 10^{-2}\|C_k\|\right\}. \tag{9.2}$$

This scheme for setting the tolerances satisfies (1.3).

For both, the decoupled and the coupled approaches, we used approximations to reduced and to full Hessians. We approximate these matrices using the limited memory BFGS representations given in [6] with a memory size of 5 pairs of vectors. For the reduced Hessian we use a null–space secant update (see [43], [54]). The initial Hessian approximation is γI_{n-m} for the reduced Hessian and γI_n for the full Hessian, where γ is the regularization parameter in the objective function, set in both examples to 10^{-3}.

In both examples the starting vector is $x_0 = 0$.

Since our algorithms are tailored for problems originally governed by infinite dimensional equations, our implementation allows the use of weighted scalar products. In particular, we use scalar products $\langle u^1, u^2\rangle_U$ and $\langle y^1, y^2\rangle_Y$ instead of $(u^1)^T u^2$ and $(y^1)^T y^2$. The scalar product for the unknown x is given by $\langle x^1, x^2\rangle_X = \langle u^1, u^2\rangle_U + \langle y^1, y^2\rangle_Y$. For most applications these scalar products are defined by appropriate discretizations of the inner products in the infinite dimensional control space and the state space, respectively. This feature is important for the correct computation of the adjoint and the appropriate scaling of the problem. Moreover, in many cases, we could observe a mesh independent behavior of our algorithms. These scalar products are used in the conjugate–gradient algorithms, see e.g. Algorithm 7.1, and in the quasi–Newton updates.

It is not the purpose of this paper to explore and analyze this feature of our algorithms. We refer to [7] for the detailed study of one application and the exposition of the importance of the scalar product. For a detailed description of how the scalar products are used and for a comprehensive description of the implementation we refer to [28].

9.1. Boundary control of a nonlinear heat equation. The first example is the boundary control of a nonlinear heat equation. This and similar control problems are discussed e.g. in [5], [32], [35], [42].

The goal is to control the heating process in such a way that the temperature at the boundary $x = 1$ follows a certain desired temperature profile $y_d(t)$. The control $u(t)$ acts on

the boundary $x = 0$. The problem can be formulated as follows:

$$\text{minimize } \frac{1}{2}\int_0^T [(y(1,t) - y_d(t))^2 + \gamma u^2(t)]dt$$

subject to

$$\begin{aligned}
\tau(y(x,t))\tfrac{\partial y}{\partial t}(x,t) - \partial_x(\kappa(y(x,t))\partial_x y(x,t)) &= q(x,t), \quad (x,t) \in (0,1)\times(0,T),\\
\kappa(y(0,t))\partial_x y(0,t) &= g[y(0,t) - u(t)], \quad t \in (0,T),\\
\kappa(y(1,t))\partial_x y(1,t) &= 0, \quad t \in (0,T),\\
y(x,0) &= y_0(x), \quad x \in (0,1),\\
u_{low} \le \quad u \quad &\le u_{upp},
\end{aligned}$$

where $y \in L^2(0,T;H^1(0,1))$, and $u \in L^2(0,T)$. The functions $\tau, \kappa \in C^1(\mathbb{R})$ denote the specific heat capacity and the heat conduction, respectively. $y_0 \in H^1(0,1)$ is the initial temperature distribution, $q \in L^2(0,T;H^1(0,1))$ is the source term, g is a given scalar, and γ is a positive regularization parameter. Here $u_{low}, u_{upp} \in L^\infty(0,T)$ are given functions.

If the partial differential equation and the integral are discretized we obtain an optimization problem of the form (1.1). The discretization uses finite elements and is discussed in [5] (see also [26], [32]). The spatial discretization is done using piecewise linear finite elements with N_x subintervals of equidistant length in $(0,1)$. The time discretization is performed by partitioning the interval $[0,T]$ into N_t equidistant subintervals. Then the backward Euler method is used to approximate the state space in time, and piecewise constant functions are used to approximate the control space.

With this discretization scheme, $C_y(x)$ is a block bidiagonal matrix with nonsymmetric tridiagonal blocks. In the exact implementation we use the `LINPACK` subroutine `DGTSL` to solve the tridiagonal systems. These calculations are reported in [15]. We introduce inexactness into this problem by solving these tridiagonal systems inexactly. For this purpose we tested several iterative methods like GMRES, QMR, and BiCGSTAB. The results were quite similar and we report here those obtained with GMRES(10). Since we have to solve a nonsymmetric tridiagonal system at each time step, we require the residual norms for these systems to be smaller than the tolerances given in (9.1) and (9.2) divided by N_t.

For this example, the inner products $\langle u^1, u^2\rangle_U$ and $\langle y^1, y^2\rangle_Y$ are chosen to be discretizations of the $L^2(0,T)$ and $L^2(0,T;H^1(0,1))$ scalar products of the control and the state spaces respectively.

If spatial and time discretization are chosen properly, the partial Jacobian $C_y(x)$ is invertible with uniformly bounded inverse. This follows from the ellipticity of the problems that have to be solved in every time step. See [5], [32]. Due to the simple structure of the objective function, derivatives of f are bounded. Since we use the adjoint multiplier, the previous results imply the boundedness of the Lagrange multiplier estimates.

The functions in this example are those used in [32, Example 4.1], [15]. The size of the problem tested is $n = 2100$, $m = 2000$ corresponding to the values $N_t = 100$, $N_x = 20$. The upper and lower bounds are $b_i = 0.01$, $a_i = -1000$, $i = 1, \ldots, n-m$.

We ran the exact and inexact TRIP SQP algorithms using decoupled and coupled approaches and reduced and full Hessians. The total number of iterations for each case is given in Table 9.1. The quantities $f(x)$, $\|C(x)\|$, and $\|D(x)W(x)^T\nabla f(x)\|$ are plotted in Figure 9.1. There were no rejected steps. In all the cases the algorithms took less than fifty iterations to attain the convergence criteria. The coupled approach did not perform as well as the decoupled approach. This is explained by the accumulation of errors due to inexactness. In fact,

if the decoupled approach is used, the y component of the tangential s_k^{t} is computed only in Step 3 of Algorithm 7.1, and although this computation is inexact, there is no accumulation of errors. In the coupled approach, the y part of the tangential component s_k^{t} of the step is updated at every conjugate–gradient iteration through an inexact linearized state solver. This destroys the symmetry of the subproblem and the conjugate–gradient method requires more iterations. As the number of conjugate–gradient iterations increases, this error propagates, and the steps that are computed are farther away from the null space of the linearized constraints.

We illustrate this situation in Figure 9.2, where we show how far $\|J_k s_k^{\mathsf{n}}\|$ and $\|J_k(s_k^{\mathsf{n}} + s_k^{\mathsf{t}})\|$ are from each other. The dotted line shows the size of the residual of the linearized state equation after the computation of the quasi–normal component. If the tangential component is in the null–space of the Jacobian, then this would be the size of the residual of the linearized state equation for the whole step. In other words, we would have $\|J_k s_k^{\mathsf{n}}\| = \|J_k(s_k^{\mathsf{n}} + s_k^{\mathsf{t}})\|$. However, due to the inexactness in the application of W_k and W_k^T, the size of the residual of the linearized state equation for the whole step is larger and is given by the solid line. It can be seen that the difference grows as W_k or W_k^T are applied more often in the computation of the tangential component. In particular, the difference is larger if the coupled approach is used.

TABLE 9.1
Number of iterations to solve the optimal control problems.

Optimal control problem governed by	Decoupled		Coupled	
	Reduced $\widehat{H}_k$	Full H_k	Reduced $\widehat{H}_k$	Full H_k
heat equation (exact solvers)	16	18	17	19
heat equation (inexact solvers)	16	18	29	48
semi-linear elliptic equation	18	20	27	36(39)

9.2. Distributed control of a semi–linear elliptic equation. The second example is the distributed control of a semi–linear elliptic equation. The control problem is given by

$$\text{minimize } \frac{1}{2}\int_\Omega [(y - y_d)^2 + \gamma u^2]dx \tag{9.3}$$

over all y and u satisfying the state equation

$$\begin{aligned} -\Delta y + g(y) &= u, && \text{in } \Omega, \\ y &= d, && \text{on } \partial\Omega\,, \end{aligned} \tag{9.4}$$

and the control constraints

$$u_{low} \le u \le u_{upp}\,, \tag{9.5}$$

where $y \in H^1(\Omega)$, $u \in L^2(\Omega)$, $u_{low}\,, u_{upp} \in L^\infty(\Omega)$ are given functions, and Ω is a bounded domain of $\mathbb{R}^2$, with boundary $\partial\Omega$. In our examples we choose $\Omega = (0,1)^2$, $d = 0$, $g(y) = e^y$, and $y_d = \sin(2\pi x_1)\sin(2\pi x_2)$. In this case the state equation (9.4) is a particular Bratu problem. Solvability and applications of the state equation are discussed e.g. in [23, Section IV.2], [24]. For the discretization of the problem, we use piecewise linear finite elements with a uniform triangulation obtained by first subdividing the x and the y subinterval into a

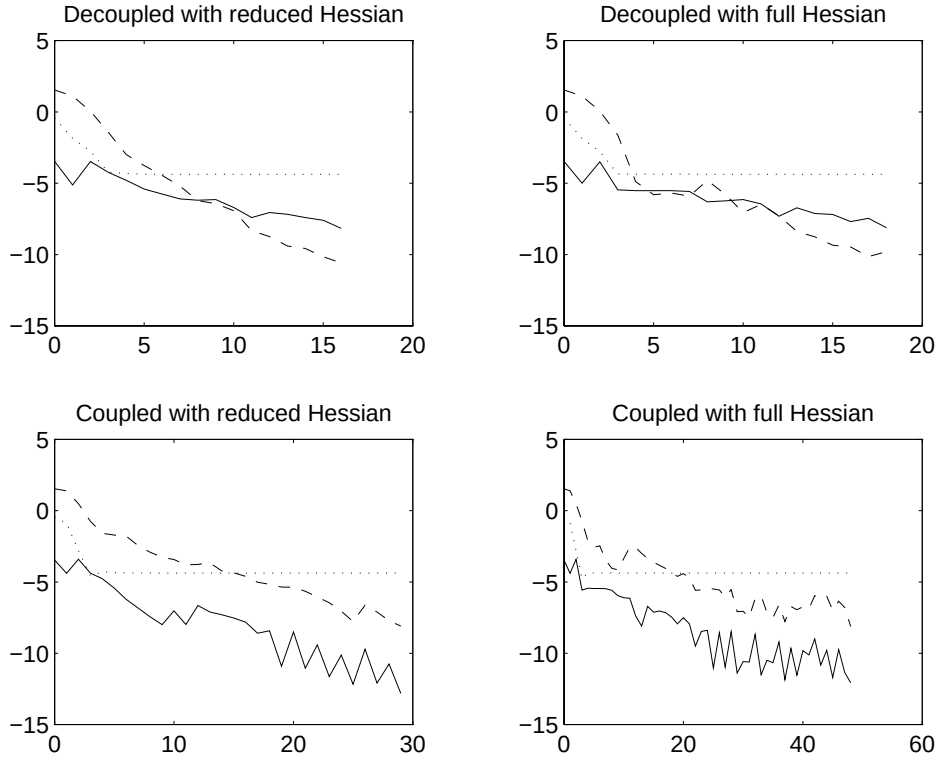

FIG. 9.1. *Performance of the inexact TRIP SQP algorithms applied to a boundary control problem of a nonlinear heat equation. Here* $log_{10} f(x_k)$ *(dotted line),* $log_{10}||C(x_k)||$ *(dashed line), and* $log_{10}||D(x_k)W(x_k)^T \nabla f(x_k)||$ *(solid line) are plotted as a function of* k.

sample of subintervals and then cutting each resulting subsquare into two triangles. The same discretization was used for the states and the controls.

Since the linearizations of the infinite dimensional state equation is elliptic and is discretized by conforming finite elements, the matrices $C_y(x)^{-1}$ are uniformly bounded. As in the previous example, the simple structure of the objective function implies that the derivatives of f are bounded. This also implies the boundedness of the adjoint multiplier estimates.

The norms used for the states and controls are the discretizations of the $H^1(\Omega)$ and $L^2(\Omega)$ norms. The linearized state equation and the adjoint equation are solved using GMRES(20) preconditioned from the left with the inverse Laplacian. To apply this preconditioner, one has to compute the solution of the discrete Laplace equation with different right hand sides. This was done using multilevel preconditioned conjugate gradients. Note that for $g(y) = e^y$, the problem is self–adjoint. Therefore a conjugate–gradient algorithm could have been used instead of GMRES. However, the implementation was done for the more general problem with state equation $-\Delta y + g(y, \nabla y) = u$, which in general is not self–adjoint.

In this example, the number of controls is equal to the number of states. In the computations reported below we use $m = n = 289$ which corresponds to a uniform triangulation with 512 triangles. The upper and lower bounds are $b_i = 5$, $a_i = -1000$, $i = 1, \ldots, n - m$.

The total number of iterations needed by the inexact TRIP SQP algorithms to solve this problem are presented in Table 9.1. In all situations but one, all the steps were accepted. (The situation we refer to is the coupled approach with full Hessian approximation where

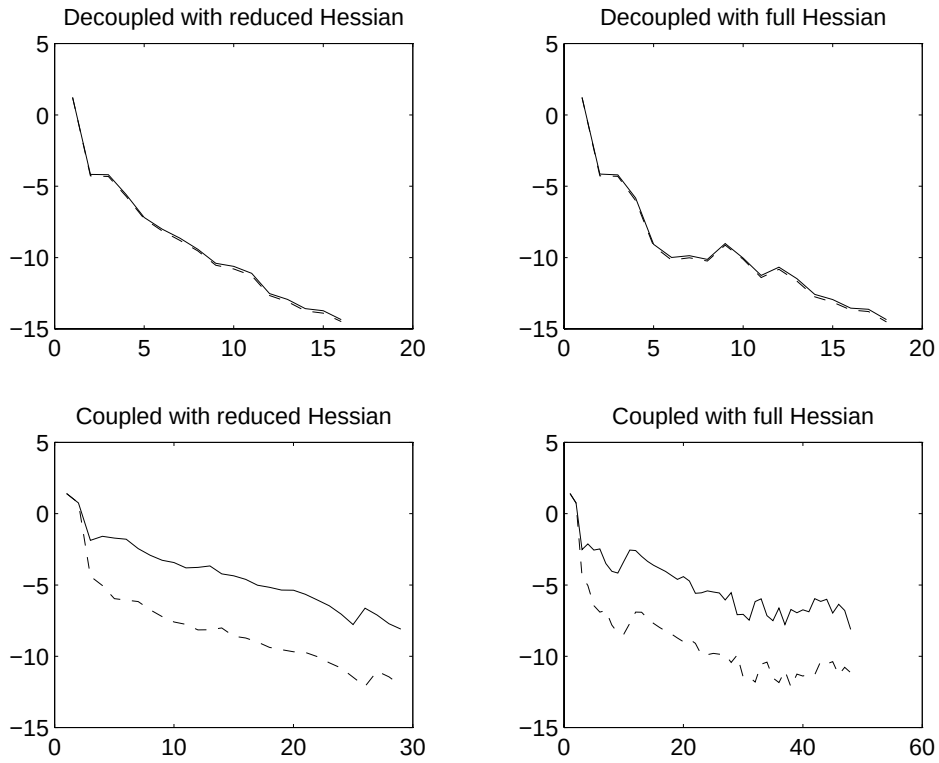

FIG. 9.2. *Illustration of the performance of the inexact TRIP SQP algorithms applied to a boundary control problem of a nonlinear heat equation. These plots show the residuals of the linearized state equations* $log\ _{10}||J_k s_k^{\mathrm{n}}||$ *in dashed line and* $log\ _{10}||J_k(s_k^{\mathrm{n}} + s_k^{\mathrm{t}})||$ *in solid line.*

there were 36 accepted steps among the 39 computed.) The quantities $f(x)$, $\|C(x)\|$, and $\|D(x)W(x)^T\nabla f(x)\|$ are plotted in Figure 9.3. The convergence behavior of the inexact TRIP SQP algorithms is similar to the convergence behavior for the other example. Again the decoupled approach performs better than the coupled one due to the fact that less errors are accumulated. See Figure 9.4.

The last experiment that we report consisted of applying the inexact TRIP SQP Algorithms 4.1 to solve large instances of the distributed semi–linear control problem. In this experiment, we used the decoupled approach with a limited memory BFGS update to approximate the reduced Hessian matrix as described above. The number of iterations corresponding to four instances of this control problem are given in Table 9.2. These instances were generated by decreasing the mesh size, i.e. by increasing the number of triangles in the discretization. In this table we include the number of linearized state and adjoint equations of the form (1.2) solved by the algorithms.

We point out that in this example the control is distributed in Ω and the number of components in u is $\frac{n}{2}$. For the values $b_i = 5$, $a_i = -1000$, $i = 1, \ldots, n-m$ of the upper and lower bounds that we chose, the number of control variables u active at the solution is roughly equal to $\frac{n}{10}$. These observations are important for the conclusions we draw in the next paragraph.

It is well known that in many interior–point algorithms for linear and convex programming problems the number of iterations is a polynomial function of the size of the problem.

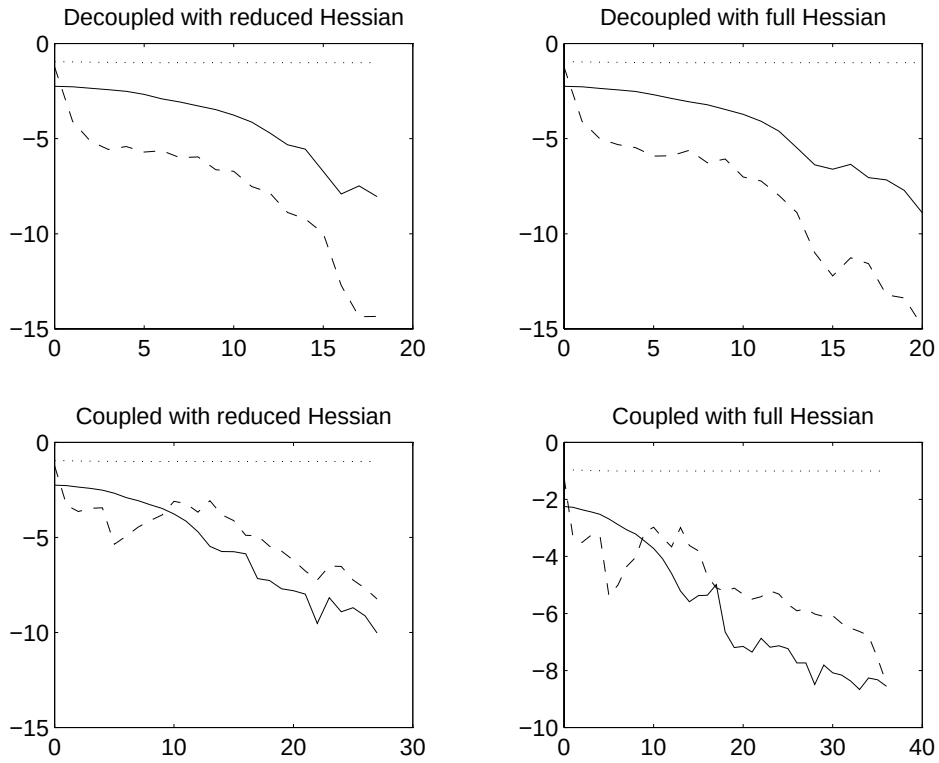

FIG. 9.3. *Performance of the inexact TRIP SQP algorithms applied to a distributed control problem of a semi–linear elliptic equation. Here* $log_{10} f(x_k)$ *(dotted line),* $log_{10}||C(x_k)||$ *(dashed line), and* $log_{10}||D(x_k)W(x_k)^T \nabla f(x_k)||$ *(solid line) are plotted as a function of* k.

On the other hand, most active set methods have a exponential worst–case complexity. In interior–point algorithms, as we increase the dimension of the problem we should observe at most a polynomial increase in the number of the iterations. We can see from Table 9.2 that this is clearly the case for the TRIP SQP algorithms. These results once more show the effectiveness of these algorithms for optimal control problems with bound constraints on the controls. (If there are rejected steps, then the number of iterations in brackets corresponds to all the accepted and rejected iterations.)

TABLE 9.2
Number of iterations to solve large distributed semi–linear control problems.

variables (n)	constraints (m)	iterations	$C_y(x_k)$ solvers	$C_y(x_k)^T$ solvers
578	289	18	54	37
2178	1089	22	66	45
8450	4225	26 (31)	83	58
33282	16641	49	147	99

10. Conclusions and future work. In this paper we have investigated the theoretical and numerical behavior of a class of trust–region interior–point SQP algorithms under the

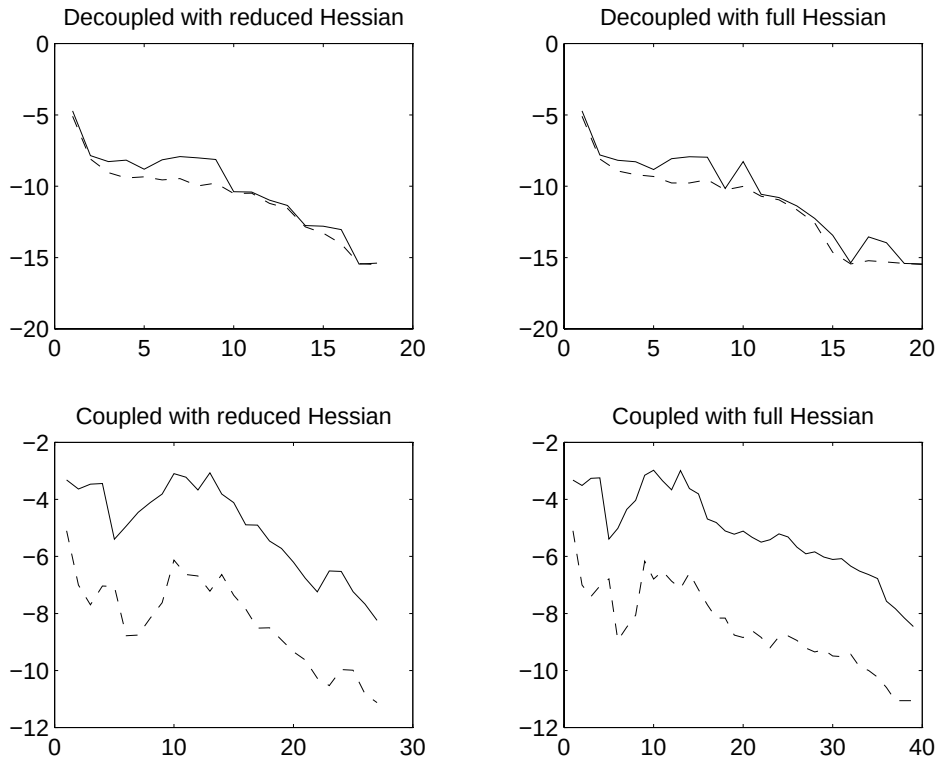

FIG. 9.4. *Illustration of the performance of the inexact TRIP SQP algorithms applied to a distributed control problem of a semi–linear elliptic equation. These plots show the residuals of the linearized state equations* $log_{10}||J_k s_k^n||$ *in dashed line and* $log_{10}||J_k(s_k^n + s_k^t)||$ *in solid line.*

presence of inexactness. These algorithms have been proposed in [15] for problems of the type (1.1), where the equality constraints often come from the discretization of partial differential equations. We have generalized the global convergence result given in [15] to the case where linear solvers and directional derivatives associated with these constraints are inexact. We proved that global convergence to a point that satisfies the first–order KKT conditions (2.2)–(2.3) can be guaranteed if the absolute error in the solution of linear systems with $C_y(x_k)$ and $C_y(x_k)^T$ and in the calculation of directional derivatives of C at x_k is $\mathcal{O}\left(\min\{\delta_k, \|C_k\|\}\right)$. Numerical experiments with two optimal control problems has confirmed our analysis and showed how the inexact calculation of the quantities $W_k d_u$ and $W_k^T d$ can affect the use of conjugate gradients to compute the tangential component of the step and the overall performance of the algorithms.

The conditions on the inexactness described in this paper, summarized in (1.3), are sufficient to guarantee global convergence to a stationary point. However, as it is the case for systems of nonlinear equations, the practical implementation of conditions greatly influences the performance of the algorithm. Issues like oversolving and forcing faster rates of local convergence are of importance and will be the subject of future investigations. Since the quasi–normal component can be viewed as one step of Newton's method (with a trust–region globalization) towards feasibility for given u, there is a close relationship with the studies of inexact Newton methods for systems of nonlinear equations [20], [19].

The computation of the tangential component using the coupled approach is another

issue that will be investigated further. In particular the loss of symmetry due the inexactness deserves attention and the use of nonsymmetric methods for the solution of these subproblems will be investigated. See also [36].

In our applications the uniform boundedness of $C_y(x)^{-1}$ can be shown. However, the uniform boundedness of $C_y(x)^{-1}$, or even the global invertibility is not guaranteed in other important applications. Possible relaxations of this condition will be investigated along with relaxations of other assumptions. For example, the requirement of the boundedness of $\{H_k\}$ might be relaxed using the ideas in [45] for trust–region methods for unconstrained optimization.

Acknowledgements. Most of this paper was written during stays of LNV at the Interdisciplinary Center for Applied Mathematics (ICAM) at Virginia Tech, and of MH at the Center for Research on Parallel Computation (CRPC) at Rice University. The authors would like to thank the personnel at those centers for their hospitality and financial support. During the work on this paper we profited from many stimulating discussions with John Dennis. We are indebted to him for his insights and support.

REFERENCES

[1] L. T. BIEGLER, J. NOCEDAL, AND C. SCHMID, *A reduced Hessian method for large–scale constrained optimization*, SIAM J. on Optimization, 5 (1995), pp. 314–347.

[2] P. T. BOGGS, J. W. TOLLE, AND A. J. KEARSLEY, *A truncated SQP algorithm for large scale nonlinear programming problems*, in Proceedings of the Sixth Conference on Numerical Analysis and Optimization, S. Gomez and J.-P. Hennart, eds., Boston, 1994, Kluwer Academic Publishers.

[3] P. N. BROWN, *A local convergence theory for combined inexact–Newton/ finite–difference projection methods*, SIAM J. Numer. Anal., 24 (1987), pp. 407–434.

[4] P. N. BROWN AND Y. SAAD, *Convergence theory of nonlinear Netwon–Krylov algorithms*, SIAM J. Optim., 4 (1994), pp. 297–330.

[5] J. BURGER AND M. POGU, *Functional and numerical solution of a control problem originating from heat transfer*, J. Optim. Theory Appl., 68 (1991), pp. 49–73.

[6] R. H. BYRD, J. NOCEDAL, AND R. B. SCHNABEL, *Representations of quasi-Newton matrices and their use in limited memory methods*, Math. Programming, 63 (1994), pp. 129–156.

[7] E. M. CLIFF, M. HEINKENSCHLOSS, AND A. SHENOY, *An optimal control problem for flows with discontinuities*, Journal of Optimization Theory and Applications, 94 (1997), pp. 273–309.

[8] T. F. COLEMAN AND Y. LI, *An interior trust region approach for nonlinear minimization subject to bounds*, CTC93TR131, Center for Theory and Simulation in Science and Engineering, Cornell University, Ithaca, NY 14853–3801, 1993. appeared as [9].

[9] ———, *An interior trust region approach for nonlinear minimization subject to bounds*, SIAM J. Optimization, 6 (1996), pp. 418–445.

[10] A. R. CONN, N. I. M. GOULD, A. SARTENAER, AND P. L. TOINT, *Global convergence of a class of trust region algorithms for optimization using inexact projections onto convex constraints*, SIAM J. Optimization, 3 (1993), pp. 164–221.

[11] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[12] R. S. DEMBO AND T. STEIHAUG, *Truncated–Newton algorithms for large–scale unconstrained optimization*, Math. Prog., 19 (1983), pp. 190–212.

[13] R. S. DEMBO AND U. TULOWITZKI, *Sequential truncated quadratic programming*, in Numerical Optimization 1984, P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., SIAM, Philadelphia, 1985, pp. 83–101.

[14] J. E. DENNIS, 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.

[15] J. E. DENNIS, M. HEINKENSCHLOSS, AND L. N. VICENTE, *Trust–region interior–point algorithms for a class of nonlinear programming problems*, SIAM J. Control and Optimization, 36 (1998), pp. 1750–1794.

[16] J. E. DENNIS AND L. N. VICENTE, *Trust–region interior–point algorithms for minimization problems with simple bounds*, Tech. Report TR94–42, Department of Computational and Applied Mathematics, Rice University, 1994. appeared as [17].

[17] ———, *Trust-region interior-point algorithms for minimization methods with simple bounds*, in Applied Mathematics and Parallel Computing, Festschrift for Klaus Ritter, H. Fis-

cher, B. Riedmüller, and S. Schäffler, eds., Heidelberg, 1996, Physica-Verlag, pp. 97–107. http://www.mat.uc.pt/∼lvicente/papers/papers.html.

[18] P. DEUFLHARD, *Global inexact Newton methods for very large scale nonlinear problems*, Impact of Computing in Science and Engineering, 4 (1991), pp. 366–393.

[19] S. C. EISENSTAT AND H. F. WALKER, *Choosing the forcing terms in an inexact Newton method*, no. 6–94–75, Dept. of Mathematics and Statistics, Utah State University, Logan, Utah 84322–3900, 1994.

[20] ———, *Globally convergent inexact Newton methods*, SIAM J. Optimization, 4 (1994), pp. 393–422.

[21] M. EL-ALEM, *A global convergence theory for the Celis–Dennis–Tapia trust region algorithm for constrained optimization*, SIAM J. Numer. Anal., 28 (1991), pp. 266–290.

[22] R. FONTECILLA, *On inexact quasi–Newton methods for constrained optimization*, in Numerical Optimization 1984, P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., SIAM, Philadelphia, 1985, pp. 102–118.

[23] R. GLOWINSKI, *Numerical Methods for Nonlinear Variational Problems*, Springer-Verlag, Berlin, Heidelberg, New-York, Tokyo, 1984.

[24] R. GLOWINSKI, H. B. KELLER, AND L. RHEINHART, *Continuation-conjugate gradient methods for the least-squares solution of nonlinear boundary value problems*, SIAM J. Sci. Stat. Comp., 6 (1985), pp. 793–832.

[25] G. H. GOLUB AND U. VON MATT, *Quadratically constrained least squares and quadratic problems*, Numerische Mathematik, 59 (1991), pp. 561–580.

[26] M. HEINKENSCHLOSS, *On the solution of a two ball trust–region subproblem*, Mathematical Programming, 64 (1994), pp. 249–276.

[27] ———, *Projected sequential quadratic programming methods*, SIAM J. Optimization, 6 (1996), pp. 373–417.

[28] M. HEINKENSCHLOSS AND L. N. VICENTE, *TRICE: A package of trust–region interior–point algorithms for the solution of optimal control and engineering design problems. User's guide*, tech. report, Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005–1892, 1996. `http://www.caam.rice.edu/∼trice`.

[29] H. M. KLÍE, M. RAMÉ, AND M. F. WHEELER, *Krylov–secant methods for solving systems of nonlinear equations*, Tech. Report TR95–27, Department of Computational and Applied Mathematics, Rice University, 1995.

[30] F.-S. KUPFER, *An infinite dimensional convergence theory for reduced SQP methods in Hilbert space*, SIAM J. Optimization, 6 (1996), pp. 126–163.

[31] F.-S. KUPFER AND E. W. SACHS, *A prospective look at SQP methods for semilinear parabolic control problems*, in Optimal Control of Partial Differential Equations, Irsee 1990, K.-H. Hoffmann and W. Krabs, eds., Lect. Notes in Control and Information Sciences, Vol. 149, Springer Verlag, 1991, pp. 143–157.

[32] ———, *Numerical solution of a nonlinear parabolic control problem by a reduced SQP method*, Computational Optimization and Applications, 1 (1992), pp. 113–135.

[33] M. LALEE, J. NOCEDAL, AND T. PLANTENGA, *On the implementation of an algorithm for large–scale equality constrained optimization*. 1994.

[34] F. LEIBFRITZ AND E. W. SACHS, *Numerical solution of parabolic state constrained control problems using SQP– and interior–point–methods*, in Large Scale Optimization: State of the Art, W. W. Hager, D. Hearn, and P. Pardalos, eds., Kluwer, 1994, pp. 251–264.

[35] J. C. MEZA AND T. D. PLANTENGA, *Optimal control of a CVD reactor for prescibed temperature behavior*, Tech. Report SAND95–8224, Sandia National Laboratories, 1995.

[36] J. C. MEZA AND W. W. SYMES, *Conjugate residual methods for almost symmetric linear systems*, J. Optim. Theory Appl., 72 (1992), pp. 415–440.

[37] J. J. MORÉ, *Recent developments in algorithms and software for trust region methods*, in Mathematical Programming, The State of The Art, A. Bachem, M. Grötschel, and B. Korte, eds., Springer Verlag, Berlin, Heidelberg, New-York, 1983, pp. 258–287.

[38] W. MURRAY AND F. J. PRIETO, *A sequential quadratic programming algorithm using an incomplete solution of the subproblem*, SIAM J. Optimization., 5 (1995), pp. 590–640.

[39] S. G. NASH, *Newton–like minimization via the Lanczos method*, SIAM J. on Numer. Anal., 21 (1984), pp. 770–788.

[40] ———, *Solving nonlinear programming problems using truncated-Newton techniques*, in Numerical Optimization 1984, P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., SIAM, Philadelphia, 1985, pp. 119–136.

[41] S. G. NASH AND J. NOCEDAL, *A numerical study of the limited memory BFGS method and the truncated–Newton method for large scale optimization*, SIAM J. Optimization, 1 (1991), pp. 358–372.

[42] P. NEITTAANMÄKI AND D. TIBA, *Optimal Control of Nonlinear Parabolic Systems. Theory, Algorithms, and Applications*, Marcel Dekker, New York, Basel, Hong Kong, 1994.

[43] J. NOCEDAL AND M. L. OVERTON, *Projected Hessian updating algorithms for nonlinearly constrained optimization*, SIAM J. Numer. Anal., 22 (1985), pp. 821–850.

[44] M. J. D. POWELL, *Convergence properties of a class of minimization algorithms*, in Nonlinear Programming 2, O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, eds., Boston, New-York, London,..., 1975,

Academic Press, pp. 1–27.

[45] ———, *On the global convergence of trust region algorithms for unconstrained minimization*, Math. Programming, 29 (1984), pp. 297–303.

[46] Y. SAAD AND M. H. SCHULTZ, *GMRES a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comp., 7 (1986), pp. 856–869.

[47] D. C. SORENSEN, *Minimization of a large scale quadratic function subject to an ellipsoidal constraint*, Tech. Report TR94–27, Department of Computational and Applied Mathematics, Rice University, 1994. appeared as [48].

[48] ———, *Minimization of a large scale quadratic function subject to an spherical constraint*, SIAM J. Optimization, 7 (1997), pp. 141–161.

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

[50] ———, *Local and superlinear convergence for truncated iterated projections methods*, Math. Programming, 27 (1983), pp. 176–190.

[51] P. 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.

[52] L. N. VICENTE, *Trust–Region Interior–Point Algorithms for a Class of Nonlinear Programming Problems*, PhD thesis, Rice University, Department of Computational and Applied Mathematics, 1996.

[53] S. J. WRIGHT, *Interior point methods for optimal control of discrete-time systems*, J. Optim. Theory Appl., 77 (1993), pp. 161–187.

[54] Y. XIE, *Reduced Hessian Algorithms for Solving Large–Scale Equality Constrained Optimization Problems*, PhD thesis, Dept. of Computer Science, University of Colorado, 1991.

[55] D. P. YOUNG, W. P. HUFFMAN, R. G. MELVIN, M. B. BIETERMAN, C. L. HILMES, AND F. T. JOHNSON, *Inexactness and Global Convergence in Design Optimization*. AIAA Paper 94-4386.