A SUBSPACE MINIMIZATION METHOD FOR THE TRUST-REGION STEP*

JENNIFER B. ERWAY† AND PHILIP E. GILL‡

Abstract. We consider methods for large-scale unconstrained minimization based on finding an approximate minimizer of a quadratic function subject to a two-norm trust-region constraint. The Steihaug-Toint method uses the conjugate-gradient (CG) algorithm to minimize the quadratic over a sequence of expanding subspaces until the iterates either converge to an interior point or cross the constraint boundary. However, if the CG method is used with a preconditioner, the Steihaug-Toint method requires that the trust-region norm be defined in terms of the preconditioning matrix. If a different preconditioner is used for each subproblem, the shape of the trust-region can change substantially from one subproblem to the next, which invalidates many of the assumptions on which standard methods for adjusting the trust-region radius are based. In this paper we propose a method that allows the trust-region norm to be defined independently of the preconditioner. The method solves the inequality constrained trust-region subproblem over a sequence of evolving low-dimensional subspaces. Each subspace includes an accelerator direction defined by a regularized Newton method for satisfying the optimality conditions of a primal-dual interior method. A crucial property of this direction is that it can be computed by applying the preconditioned CG method to a positive-definite system in both the primal and dual variables of the trust-region subproblem. Numerical experiments on problems from the CUTEr test collection indicate that the method can require significantly fewer function evaluations than other methods. In addition, experiments with general-purpose preconditioners show that it is possible to significantly reduce the number of matrix-vector products relative to those required without preconditioning.

Key words. Large-scale unconstrained optimization, trust-region methods, conjugate-gradient methods, Krylov methods, preconditioning.

AMS subject classifications. 49M37, 65F10, 65K05, 65K10, 90C06, 90C26, 90C30

1. Introduction.

The jth iteration of a trust-region method for unconstrained minimization involves finding an approximate solution of the trust-region subproblem:

$$\underset{s\in\mathbb{R}^n}{\text{minimize}} \;\; \mathcal{Q}_j(s) \equiv g_j^T s + \tfrac{1}{2} s^T H_j s \quad \text{subject to} \quad \|s\| \leq \delta_j, \tag{1.1}$$

where δ_j is a given positive trust-region radius and $\mathcal{Q}_j(s)$ is the quadratic model of a scalar-valued function with gradient g_j and Hessian H_j. The focus of this paper is on the solution of (1.1) when the matrix H_j is best accessed as an *operator* for the definition of matrix-vector products of the form $H_j v$.

In this context, Steihaug [22] and Toint [23] independently proposed methods for solving (1.1) when the trust-region is defined in terms of the two-norm, i.e., the constraint is $\|s\|_2 \leq \delta_j$. If H_j is positive definite, the Newton equations $H_j s = -g_j$ define the *unconstrained* minimizer of (1.1). The Steihaug-Toint method begins with the application of the conjugate-gradient (CG) method to the Newton equations. This process is equivalent to minimizing $\mathcal{Q}_j$ over a sequence of expanding subspaces generated by the conjugate-gradient directions. As long as the curvature of $\mathcal{Q}_j$ remains positive on each of these subspaces, the CG iterates steadily increase in norm and the CG iterates either converge inside the trust region or form a piecewise-linear path with a unique intersection-point on the trust-region boundary. When H_j is not positive

*Research supported by the National Science Foundation grant DMS-0511766.

†Department of Mathematics, PO Box 7388, Wake Forest University, Winston-Salem, NC 27109 (erwayjb@wfu.edu).

‡Department of Mathematics, University of California, San Diego, La Jolla, CA 92093-0112 (pgill@ucsd.edu).

definite, a solution of (1.1) must lie on the boundary of the trust region and the CG method may generate a direction p along which $\mathcal{Q}_j$ has zero or negative curvature. In this case, the algorithm is terminated at the point on p that intersects the boundary of the trust region.

If the Steihaug-Toint method is terminated on the boundary of the trust region, the step may bear little relation to an optimal solution of (1.1). This means that, in contrast to line-search methods, it is not possible to choose an approximate solution that balances the cost of computing the problem functions with the cost of computing the trust-region step (see, e.g., [4] for more discussion of this issue). Several extensions of the Steihaug-Toint method have been proposed that allow the accuracy of a constrained solution to be specified. Gould, Lucidi, Roma, and Toint [10] proposed the generalized Lanczos trust-region (GLTR) algorithm, which finds a constrained minimizer of (1.1) over a sequence of expanding subspaces associated with the Lanczos process for reducing H_j to tridiagonal form. Erway, Gill and Griffin [4] continue to optimize on the trust-region boundary using the sequential subspace minimization (SSM) method. This method approximates a constrained minimizer over a sequence of evolving low-dimensional subspaces that do not necessarily form a nested sequence. Erway, Gill and Griffin use a basis for each subspace that includes an accelerator vector defined by a primal-dual augmented Lagrangian method.

These recent extensions to the Steihaug-Toint method add the ability to increase the accuracy of the trust-region solution when needed. The result is a reliable and efficient method for applying the CG method to large-scale optimization. However, there are some situations where the Steihaug-Toint approach may not be efficient.

Preconditioning the conjugate-gradient method. In many applications the convergence rate of CG can be significantly improved by using a preconditioner, which is usually available in the form of a positive-definite operator M_j^{-1} that clusters the eigenvalues of $M_j^{-1}H_j$. If a preconditioned CG method is used, the increasing norm property of the iterates holds only in the *weighted* norm $\|x\|_{M_j} = (x^T M_j x)^{1/2}$, which mandates the use of a trust region of the form $\|s\|_{M_j} \leq \delta_j$. Unfortunately, if a different preconditioner is used for each trust-region subproblem, the shape of the trust-region may alter dramatically from one subproblem to the next. Since a fundamental tenet of trust-region methods is that the value of δ_j be used to determine the value of δ_{j+1}, the effectiveness of the trust-region strategy may be seriously compromised. We emphasize the distinction between the *constant* weighted trust region $\|Ns\|_2 = (s^T N^T N s)^{1/2} \leq \delta_j$ typically associated with a constant nonsingular scaling matrix N, and the *varying* trust region $\|s\|_{M_j} \leq \delta_j$ induced by the preconditioner.

Convergence to second-order points. The Steihaug-Toint method and its extensions are *first-order methods*, in the sense that they are guaranteed to converge to points that satisfy the first-order necessary conditions for optimality (i.e., $g = 0$). If direct matrix factorizations are used, it is possible to approximate a global minimizer of the trust-region subproblem and thereby guarantee convergence to points that satisfy the second-order conditions for optimality, i.e., points at which the gradient is zero and the Hessian is positive semidefinite (see, e.g., Moré and Sorensen [15]). We know of no method based on the conjugate-gradient method that is guaranteed to find a global solution of (1.1) in finite-precision. For example, the Steihaug-Toint method is not guaranteed to compute a solution on the boundary when $\mathcal{Q}_j$ is unbounded below. (Suppose that H_j is indefinite and $\mathcal{Q}_j(s)$ has a stationary point $\hat{s}$ such that $\|\hat{s}\| < \delta_j$. If H_j is positive definite on the Krylov subspace spanned by g_j, $H_j g_j$, $H_j^2 g_j$, ..., then CG will terminate at the interior point $\hat{s}$.) Notwithstanding these

theoretical difficulties, it seems worthwhile devising strategies that have the potential of providing convergence to a global solution in "most cases".

Efficiency for repeated constrained subproblems. When solving a difficult problem, it is often the case that a sequence of problems of the form (1.1) must be solved in which only the trust-region radius δ_j changes. However, the Steihaug-Toint method is unable to exploit this information during the generation of the expanding sequence of subspaces.

In this paper we consider an *interior-point sequential subspace minimization (IP-SSM) method* that is designed to mitigate these ill-effects. (i) The method allows the use of CG preconditioning in conjunction with a standard method for updating the trust-region radius. (ii) The likelihood of approximating the global minimizer of (1.1) is increased by the computation of an approximate left-most eigenpair of H_j that is not based on the CG Krylov subspace. In particular, it allows the computation of a nonzero step when $g_j = 0$ and H_j is indefinite. (iii) Information garnered during the solution of one subproblem may be used to expedite the solution of the next.

The IP-SSM method is a member of the class of sequential subspace minimization (SSM) methods first proposed for the equality-constraint case by Hager [12, 13]. These methods approximate a constrained minimizer over a sequence of evolving low-dimensional subspaces that include an "accelerator" direction designed to increase the rate of convergence. Broadly speaking, SSM methods differ in the composition of the basis for the subspace and in the definition of the accelerator direction. Hager employs a subspace based on the gradient vector and the left-most eigenvector. The accelerator direction is found by applying the CG method with a constraint preconditioner to the KKT optimality conditions. (Hager's method uses a very accurate approximation to the left-most eigenvector and is not designed to find the low-cost approximate solutions needed in the trust-region context.) An important property of the IP-SSM method is that the accelerator direction is computed by applying the preconditioned CG method to a regularized positive-definite system in both the primal and dual variables of the *inequality* constrained problem.

Finally, we mention several Krylov-based iterative methods that are intended to find a solution of the problem of minimizing $\mathcal{Q}_j(s) = g_j^T s + \frac{1}{2} s^T H_j s$ subject to the *equality constraint* $\|s\|_2 = \delta_j$. The methods of Sorensen [21], Rojas and Sorensen [19], Rojas, Santos and Sorensen [18], and Rendl and Wolkowicz [17] approximate the eigenvalues of a matrix obtained by augmenting H_j by a row and column.

The paper is organized in four sections. In Section 2 we formulate the proposed SSM method and consider some properties of the regularized Newton equations used to generate the SSM accelerator direction. Section 3 includes numerical comparisons with the Steihaug-Toint and GLTR methods on unconstrained problems from the CUTEr test collection (see Bongartz et al. [1] and Gould, Orban and Toint [11]). Finally, Section 4 includes some concluding remarks and observations.

1.1. Notation and Glossary. Unless explicitly indicated, $\|\cdot\|$ denotes the vector two-norm or its subordinate matrix norm. The symbol e_i denotes the ith column of the identity matrix I, where the dimensions of e_i and I depend on the context. The eigenvalues of a real symmetric matrix H are denoted by $\{\lambda_i\}$, where $\lambda_n \leq \lambda_{n-1} \leq \cdots \leq \lambda_1$. The associated eigenvectors are denoted by $\{u_i\}$. An eigenvalue λ and a corresponding normalized eigenvector u such that $\lambda = \lambda_n$ are known as a *left-most eigenpair* of H. The Moore-Penrose pseudoinverse of a matrix A is denoted by $A^\dagger$. Some sections include algorithms written in a MATLAB-style pseudocode. In these algorithms, brackets will be used to differentiate between computed and stored

quantities. For example, the expression $[Ax] := Ax$ signifies that the matrix-vector product of A with x is computed and assigned to the vector $[Ax]$. Similarly, if P is a matrix with columns p_1, p_2, ..., p_m, then $[AP]$ denotes the matrix with columns $[Ap_1]$, $[Ap_2]$, ..., $[Ap_m]$.

2. A SSM Method with Interior-Point Acceleration. In this section we omit the suffix j and focus on a typical trust-region subproblem of the form

$$\underset{s\in\mathbb{R}^n}{\text{minimize}} \;\; \mathcal{Q}(s) \equiv g^T s + \tfrac{1}{2} s^T H s \quad \text{subject to} \;\; \|s\| \leq \delta. \tag{2.1}$$

The Steihaug-Toint method and its extensions start with the unconstrained minimization of $\mathcal{Q}$ and consider the constraint only if the unconstrained solution lies outside the trust-region. In the proposed *interior-point sequential subspace minimization (IP-SSM) method*, the inequality constrained problem (2.1) is minimized directly over a sequence of low-dimensional subspaces, giving a sequence of reduced inequality constraint problems of the form

$$\underset{s\in\mathbb{R}^n}{\text{minimize}} \;\; \mathcal{Q}(s) \quad \text{subject to} \;\; \|s\| \leq \delta, \quad s \in \mathcal{S}_k = \text{span}\{s_{k-1}, z_k, s_k^a\}, \tag{2.2}$$

where s_{k-1} is the current best estimate of the subproblem solution, z_k is the current best estimate of u_n (the left-most eigenvector of H), and s_k^a is an interior-point accelerator direction. The Lanczos-CG algorithm is used to define the accelerator direction and to provide basis vectors for the low-dimensional subspaces associated with the reduced versions of the left-most eigenvalue problem.

2.1. Definition of the accelerator direction. The accelerator direction s_k^a is an approximate Newton direction for the perturbed optimality conditions associated with the problem

$$\underset{s\in\mathbb{R}^n}{\text{minimize}} \;\; \mathcal{Q}(s) = g^T s + \tfrac{1}{2} s^T H s \quad \text{subject to} \;\; \tfrac{1}{2}\delta^2 - \tfrac{1}{2} s^T s \geq 0. \tag{2.3}$$

This is an inequality constrained optimization problem with Lagrange multiplier σ and Lagrangian function

$$L(s,\sigma) = \mathcal{Q}(s) - \sigma(\tfrac{1}{2}\delta^2 - \tfrac{1}{2} s^T s) = \mathcal{Q}(s) - \sigma c(s),$$

where $c(s)$ denotes the constraint residual $c(s) = \frac{1}{2}\delta^2 - \frac{1}{2} s^T s$. The necessary and sufficient conditions for a global solution of (2.3) imply the existence of a vector s and scalar σ such that

$$\begin{aligned} (H + \sigma I)s &= -g, \quad \text{with } \; H + \sigma I \; \text{ positive semidefinite,} \\ c(s)\sigma &= \;\; 0, \quad \text{with } \; \sigma \geq 0 \; \text{ and } \; c(s) \geq 0. \end{aligned} \tag{2.4}$$

(For a proof, see, e.g., Gay [7], Sorensen [20], Moré and Sorensen [16], or Conn, Gould and Toint [2].) The conventional primal-dual interior-point approach to solving (2.3) is based on finding s and σ that satisfy the perturbed optimality conditions

$$\begin{aligned} (H + \sigma I)s &= -g, \qquad \sigma > 0, \\ c(s)\sigma &= \mu, \qquad c(s) > 0 \end{aligned} \tag{2.5}$$

for a sequence of decreasing values of the positive parameter μ. Let $F(s,\sigma)$ denote the vector-valued function whose components are the residuals $(H + \sigma I)s + g$ and

$c(s)\sigma - \mu$. Given an approximate zero (s, σ) of F such that $c(s) > 0$ and $\sigma > 0$, the Newton equations for the next iterate $(s + p, \sigma + q)$ are:

$$\begin{pmatrix} H + \sigma I & s \\ -\sigma s^T & c(s) \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = - \begin{pmatrix} g + (H + \sigma I)s \\ c(s)\sigma - \mu \end{pmatrix}.$$

The assumption that $\sigma > 0$ implies that it is safe to divide the last equation by $-\sigma$ to give the symmetrized equations:

$$\begin{pmatrix} H + \sigma I & s \\ s^T & -d \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = - \begin{pmatrix} g + (H + \sigma I)s \\ d(\hat{\sigma} - \sigma) \end{pmatrix},$$

where $d = c(s)/\sigma$ and $\hat{\sigma} = \mu/c(s)$. The presence of the nonzero $(2, 2)$ block implies that the conventional interior-point approach defines a *regularization* of Newton's method for a solution of the optimality conditions (2.4). The regularized solution lies on the central path of solutions $\big(s(\mu), \sigma(\mu)\big)$ of (2.5) (see, e.g., [6]). This implies that the regularized solution $\big(s(\mu), \sigma(\mu)\big)$ will be different from (s^*, σ^*) for a given nonzero μ. Moreover, the influence of the regularization on the Newton equations diminishes as $\mu \to 0$.

These considerations suggest that we seek an alternative "exact" regularization that allows the use of a fixed value of μ, but does not perturb the regularized solution. Consider the perturbed optimality conditions

$$\begin{aligned} (H + \sigma I)s &= -g, & \sigma &> 0, \\ c(s)\sigma &= \mu(\sigma_e - \sigma), & c(s) &> -\mu, \end{aligned} \tag{2.6}$$

where σ_e is a nonnegative estimate of σ^*. If $\sigma_e = \sigma^*$, these conditions are satisfied by (s^*, σ^*) for any positive μ. The symmetrized Newton equations associated with conditions (2.6) are

$$\begin{pmatrix} H + \sigma I & s \\ s^T & -d \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = - \begin{pmatrix} g + (H + \sigma I)s \\ d(\hat{\sigma} - \sigma) \end{pmatrix},$$

where now, $d = (c(s) + \mu)/\sigma$ and $\hat{\sigma} = \mu\sigma_e/(c(s) + \mu)$. Forsgren, Gill and Griffin [5] show that these equations are equivalent to the so-called *doubly-augmented system*:

$$\begin{pmatrix} H(\sigma) + (2/d)ss^T & -s \\ -s^T & d \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = - \begin{pmatrix} g + H(\sigma)s - 2(\sigma - \hat{\sigma})s \\ d(\sigma - \hat{\sigma}) \end{pmatrix},$$

where $H(\sigma) = H + \sigma I$. Finally, we multiply the last equation and last variable by $d^{-\frac{1}{2}}$ and $d^{\frac{1}{2}}$, respectively, to improve the scaling when $\sigma \to 0$. This gives

$$\begin{pmatrix} H(\sigma) + 2\bar{s}\bar{s}^T & -\bar{s} \\ -\bar{s}^T & 1 \end{pmatrix} \begin{pmatrix} p \\ \bar{q} \end{pmatrix} = - \begin{pmatrix} g + H(\sigma)s - 2d^{\frac{1}{2}}\big(\sigma - \hat{\sigma}\big)\bar{s} \\ d^{\frac{1}{2}}(\sigma - \hat{\sigma}) \end{pmatrix}, \tag{2.7}$$

where $\bar{s} = d^{-\frac{1}{2}}s$ and $q = d^{-\frac{1}{2}}\bar{q}$. These equations are positive definite in a neighborhood of a minimizer (s, σ) such that $\sigma \in (-\lambda_n, \infty)$, and they may be solved using the CG method. If a direction of negative or zero curvature is detected, the direction is used to update a lower bound on the best estimate of σ (see Section 2.2).

It is not necessary to solve the perturbed equations (2.6) to high accuracy because the quality of the accelerator step affects only the *rate* of convergence of the SSM method. In the runs described in Section 3 only one Newton iteration was performed.

The CG method may be used in conjunction with a preconditioner of the form

$$P = \begin{pmatrix} M(\sigma) + 2\bar{s}\bar{s}^T & -\bar{s} \\ -\bar{s}^T & 1 \end{pmatrix},$$

where $M(\sigma)$ is a positive-definite approximation to $H(\sigma)$. The equations $Pv = u$ used to apply the preconditioner are solved by exploiting the equivalence of the systems:

$$\begin{pmatrix} M(\sigma) + 2\bar{s}\bar{s}^T & -\bar{s} \\ -\bar{s}^T & 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \tag{2.8a}$$

and

$$\begin{pmatrix} M(\sigma) & \bar{s} \\ \bar{s}^T & -1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} u_1 + 2u_2\bar{s} \\ -u_2 \end{pmatrix} \tag{2.8b}$$

(see Forsgren, Gill and Griffin [5]). Equations (2.8b) are solved analytically if $M(\sigma)$ is diagonal. Alternatively, if $M(\sigma)$ is defined using an incomplete Cholesky factorization of $H(\sigma)$ we solve (2.8b) using the block factorization:

$$\begin{pmatrix} M(\sigma) & \bar{s} \\ \bar{s}^T & -1 \end{pmatrix} = \begin{pmatrix} I & \\ w^T & 1 \end{pmatrix} \begin{pmatrix} M(\sigma) & \\ & -\big(1 + w^T\bar{s}\big) \end{pmatrix} \begin{pmatrix} I & w \\ & 1 \end{pmatrix},$$

where w satisfies $M(\sigma)w = \bar{s}$. Thus, the preconditioned CG computations may be arranged so that only solves with $M(\sigma)$ are required.

The calculations associated with the calculation of the accelerator direction s_k^a are summarized in Algorithm ***ipAccelerator*** below.

Algorithm *ipAccelerator*.
$\big(s_a, [Hs_a], \sigma_a, \sigma_\ell, z, \zeta\big) =$ ***ipAccelerator***$\big(s_a, [Hs_a], \sigma_a, \sigma_\ell\big)$;
  $\hat{\sigma} := \mu\sigma_e/\big(c(s_a) + \mu\big)$;
  Find an approximate solution (p, q) of (2.7) with $(s, \sigma) = (s_a, \sigma_a)$;
  (During the Lanczos process, estimate (z, ζ), and update σ_ℓ if (2.7) is indefinite);
  $\alpha_\sigma :=$ **if** $q < 0$ **then** $(\sigma_a - \sigma_\ell)/q$ **else** $+\infty$;
  $\alpha_s :=$ the positive root of $c(\alpha s_a) + \mu = 0$;
  $\alpha_{\scriptscriptstyle M} := \min\{1, (1-\mu)\alpha_\sigma, (1-\mu)\alpha_s\}$; Compute α such that $0 < \alpha \leq \alpha_{\scriptscriptstyle M}$;
  $s_a := s_a + \alpha p$; $\sigma_a := \sigma_a + \alpha q$; $[Hs_a] := [Hs_a] + \alpha[Hp]$;

Now we consider the precise effect of the regularization parameter μ. The subproblem (2.3) is said to be *degenerate* if the equations $(H - \lambda_n I)s = -g$ are compatible with least-length solution $s_{\scriptscriptstyle L}$ satisfying $\|s_{\scriptscriptstyle L}\| < \delta$. If $\lambda_n \geq 0$, the quantities $\sigma = 0$ and $s = s_{\scriptscriptstyle L}$ satisfy the optimality conditions (2.4). (If $\lambda_n > 0$, $s_{\scriptscriptstyle L}$ is just the Newton direction $-H^{-1}g$.) If $\lambda_n < 0$, the equations $(H + \sigma I)s = -g$ cannot be used alone to determine the optimal s. However, the left-most eigenvector u_n is a null vector of $H - \lambda_n I$, and there exists a scalar τ such that

$$(H - \lambda_n I)(s_{\scriptscriptstyle L} + \tau u_n) = -g \;\text{ and }\; \|s_{\scriptscriptstyle L} + \tau u_n\| = \delta.$$

In this case, $\sigma = -\lambda_n$ and $s = s_{\scriptscriptstyle L} + \tau u_n$ satisfy the optimality conditions (2.4) and thereby constitute a global solution of (2.3).

THEOREM 2.1 (Regularization of the degenerate case). *Let (s, σ) be a solution of the trust-region subproblem such that:* (i) $\|s\| = \delta$; (ii) $H + \sigma I$ *is positive semidefinite*

and singular; (iii) $g \in \text{null}(H + \sigma I)^{\perp}$*; and* (iv) $\|(H + \sigma I)^{\dagger} g\| < \delta$. *If the left-most eigenvalue of* H *has algebraic multiplicity 1, then the augmented system matrix*

$$\begin{pmatrix} H + \sigma I + (2/d) s s^T & -s \\ -s^T & d \end{pmatrix} \tag{2.9}$$

is positive definite for any $d > 0$.

Proof. Assumptions (i)–(iv) imply that (s, σ) is a constrained degenerate solution. In particular, it holds that $\sigma = -\lambda_n$, where λ_n is the left-most eigenvalue of H. A solution s of the trust-region subproblem is given by

$$s = -(H - \lambda_n I)^{\dagger} g + \beta z, \tag{2.10}$$

where z is a unit vector such that $z \in \text{null}(H - \lambda_n I)$ and β is a nonzero scalar such that $\|s\| = \delta$. Consider the following decomposition of (2.9):

$$\begin{pmatrix} H + \sigma I + \frac{2}{d} s s^T & -s \\ -s^T & d \end{pmatrix} = \begin{pmatrix} I & -\frac{1}{d} s \\ 0 & 1 \end{pmatrix} \begin{pmatrix} H - \lambda_n I + \frac{1}{d} s s^T & 0 \\ 0 & d \end{pmatrix} \begin{pmatrix} I & 0 \\ -\frac{1}{d} s^T & 1 \end{pmatrix}.$$

Assume that $H + \sigma I + (2/d) s s^T$ is not positive definite. Then there exists a nonzero p such that $p^T (H - \lambda_n I + (2/d) s s^T) p \leq 0$. As $H - \lambda_n I$ is positive semidefinite, it must hold that $p \in \text{null}(H - \lambda_n I)$ and $s^T p = 0$. Moreover, since $(H - \lambda_n I)^{\dagger} g \in \text{range}(H - \lambda_n I)$, it must hold that $s^T p = \beta z^T p = 0$, which implies that $z^T p = 0$. But this is only possible if $\dim(\text{null}(H - \lambda_n I)) > 1$. Thus $H + \sigma I + (2/d) s s^T$ must be positive definite and the result follows. ∎

2.2. Calculation of the approximate left-most eigenpair. The approximate Newton equations (2.7) are solved using the Lanczos-CG variant of the preconditioned conjugate-gradient method. During the evaluation of the Lanczos process, the Lanczos vectors are used to generate the subspace associated with an SSM method for an estimate of the left-most eigenpair of H. The estimate is computed by solving the reduced generalized eigenproblem

$$\underset{z \in \mathbb{R}^n}{\text{minimize}} \;\; z^T H z \quad \text{subject to} \quad \|z\|_2 = 1, \;\; z \in \mathcal{Z}_k = \text{span}\{z_{k-1}, \bar{v}_k, \bar{v}_{k-1}\}, \tag{2.11}$$

where z_{k-1} is the left-most eigenvector estimate from the previous CG iteration, and $\bar{v}_k$ and $\bar{v}_{k-1}$ are the first n components of the two most recently computed Lanczos vectors. Given the matrix Z_k whose columns form a maximally linearly independent subset of $\{z_{k-1}, \bar{v}_k, \bar{v}_{k-1}\}$, the solution z_k of (2.11) is defined as $z_k = Z_k w_k$, where w_k solves the reduced problem

$$\underset{y}{\text{minimize}} \;\; y^T Z_k^T H Z_k y \quad \text{subject to} \quad \|Z_k y\|_2 = 1.$$

This problem has at most three dimensions, and is solved in closed form. Once z_k has been determined, the left-most eigenvalue is estimated by the Rayleigh quotient $\zeta_k = z_k^T H z_k$. The inclusion of z_{k-1} as a generator of $\mathcal{Z}_k$ ensures that the Rayleigh quotients decrease monotonically.

The calculation of $Z_k^T H Z_k$ requires the vectors $H z_{k-1}$, $H \bar{v}_k$ and $H \bar{v}_{k-1}$. The vector $H z_{k-1}$ is the solution of the previous reduced eigenproblem. The vectors $H \bar{v}_k$ and $H \bar{v}_{k-1}$ are available as part of the two-term Lanczos recurrence. For the next step, the vector $H z_k$ is defined in terms of the identity $H z_k = H Z_k w_k$, which involves

a simple linear combination of Hz_{k-1}, $H\bar{v}_k$ and $H\bar{v}_{k-1}$. It follows that once Hz_0 is calculated, no additional matrix-vector products are needed. The calculation of the eigenpair is summarized in Algorithm ***ssmEig*** below. A random z_0 is used for the first outer iteration (i.e., $j = 0$). In subsequent iterations, z_0 is defined as the eigenvector estimate from the previous trust-region subproblem. As the sequence $\{H_j\}$ converges, z_0 should be a good estimate of the left-most eigenvector for each subproblem.

Algorithm *ssmEig*. (Part of the Lanczos-CG process within ***ipAccelerator***)
$\big(z, \zeta, [Hz], \sigma_\ell\big) = \textbf{\textit{ssmEig}}(z, \bar{v}_k, \bar{v}_{k-1}, [Hz], [H\bar{v}_k], [H\bar{v}_{k-1}], \sigma_\ell)$;
Define Z from a maximally linearly independent subset of $\bar{v}_k$, $\bar{v}_{k-1}$, and z;
Form Z^THZ and Z^TZ from z, $\bar{v}_k$, $\bar{v}_{k-1}$, $[H\bar{v}_k]$, $[H\bar{v}_{k-1}]$ and $[Hz]$;
$w := \operatorname{argmin}_y \big\{\, y^TZ^THZy \;:\; \|Zy\|_2 = 1 \,\big\}$;
$z := Zw$; $\quad \zeta = z^THz$;
$[Hz] := [HZ]w$;
$\sigma_\ell = \max\{|\zeta|, \sigma_\ell\}$;

Another estimate of the left-most eigenvector is available if the CG method detects that the Newton system (2.7) is not positive definite. The next result shows that if CG computes a conjugate direction p of negative curvature for (2.7), then p provides an estimate of the left-most eigenvector.

THEOREM 2.2. *If p is a direction of negative curvature for the matrix*

$$B = \begin{pmatrix} H + \sigma I + (2/d)ss^T & -s \\ -s^T & d \end{pmatrix},$$

with d a positive scalar, then the vector of first n elements of p is a direction of negative curvature for $H + \sigma I$.

Proof. As p is a direction of negative curvature for B, we have

$$p^TBp = p^T \begin{pmatrix} H + \sigma I + (2/d)ss^T & -s \\ -s^T & d \end{pmatrix} p < 0.$$

Let $\hat{p}$ and ρ denote the first n elements and the last element of p respectively. A simple calculation yields

$$\begin{aligned} p^TBp &= \hat{p}^T(H + \sigma I)\hat{p} + \frac{2}{d}(s^T\hat{p})^2 - 2\rho s^T\hat{p} + \rho^2 d \\ &= \hat{p}^T(H + \sigma I)\hat{p} + \frac{1}{d}(s^T\hat{p})^2 + \frac{1}{d}\big((s^T\hat{p})^2 - 2(s^T\hat{p})(\rho d) + (\rho d)^2\big) \\ &= \hat{p}^T(H + \sigma I)\hat{p} + \frac{1}{d}\big((s^T\hat{p})^2 + (s^T\hat{p} - \rho d)^2\big) < 0. \end{aligned}$$

It follows that $\hat{p}^T(H + \sigma I)\hat{p} < -\big((s^T\hat{p})^2 + (s^T\hat{p} - \rho d)^2\big)/d < 0$, as required. ∎

The algorithm maintains two approximate solutions: (s_e, σ_e) and (s_a, σ_a). The pair (s_e, σ_e) is the solution of the subspace minimization problem (2.2). The accelerator pair (s_a, σ_a) is the most recent estimate of a solution of the perturbed optimality conditions (2.6).

At each iteration, a safeguarding algorithm ensures that both σ_a and σ_e are strictly positive and not less than σ_ℓ, a current greatest lower bound on $-\lambda_n$. The

algorithm also attempts to adjust σ_a so that matrix $H+\sigma_a I$ of (2.7) is positive definite. In order to maintain the monotonicity of $\mathcal{Q}$, the value of s_e is always the solution of the subspace minimization problem. However, any σ_e such that $\sigma_e < \sigma_\ell < \sigma_a$ is overwritten by σ_a. In addition, (s_a, σ_a) is replaced by (s_e, σ_e) if $\sigma_a < \sigma_\ell < \sigma_e$. In the event that both σ_a and σ_e are less than σ_ℓ, the left-most eigenpair is used to update σ_a and σ_e.

Finally, in order that the matrix of (2.7) is positive definite, the IP-SSM method also ensures $c(s_a) + \mu > 0$. If $c(s_a) \leq -\mu$, the accelerator direction is replaced by s_e, with a suitable rescaling (if necessary) to guarantee that s_a lies exactly on the boundary.

Algorithm *safeguard*.
$\big(s_a, \sigma_a, \sigma_e, [Hs_a]\big) = \textbf{\textit{safeguard}}\big(s_a, \sigma_a, s_e, \sigma_e, \zeta, z, \sigma_\ell, [Hs_a], [Hs_e], [Hz], \delta\big)$;
  Choose $\sigma_{\min} > 0$;
  if $\sigma_a < \sigma_\ell$ **and** $\sigma_\ell < \sigma_e$ **then**
    $\sigma_a := \max\{\sigma_e, \sigma_{\min}\}$; $s_a := s_e$; $[Hs_a] := [Hs_e]$
  else if $\sigma_e < \sigma_\ell$ **and** $\sigma_\ell < \sigma_a$ **then**
    $\sigma_e := \sigma_a$;
  else if $\sigma_a < \sigma_\ell$ **and** $\sigma_e < \sigma_\ell$ **then**
    $\sigma_e := -\zeta$; $\sigma_a := -\zeta$; $s_a := \delta \times z$; $[Hs_a] := [Hz]$;
  end

2.3. Solving the reduced subproblem. At the core of the algorithm is a reduced trust-region subproblem (2.2) with at most three variables. Given the matrix P_k whose columns form a maximally linearly independent subset of $\{s_{k-1}, z_k, s_k^a\}$, the solution s_k of (2.2) may be written as $s_k = P_k w_k$, where w_k solves the reduced problem

$$\underset{y}{\text{minimize}} \;\; \mathcal{Q}(P_k y) \equiv g^T P_k y + \tfrac{1}{2} y^T P_k^T H P_k y, \quad \text{subject to} \;\; \|P_k y\|_2 \leq \delta. \tag{2.12}$$

A maximally linearly independent subset of the vectors $\{s_{k-1}, z_k, s_k^a\}$ is found using a QR decomposition with column interchanges. As in algorithm ***ssmEig***, the matrices $P_k^T H P_k$ and $P_k^T P_k$ can be formed with no additional matrix-vector products. The vector Hs_k is defined in terms of the identity $Hs_k = HP_k w_k$, which involves a simple linear combination of Hs_{k-1}, Hz_k, and Hs_k^a. The matrices $P_k^T H P_k$ and $P_k^T P_k$ are symmetrized in each case.

The reduced problem is solved using a modified version of the Moré-Sorensen algorithm [15] that computes an exact left-most eigenpair of the 3×3 shifted Hessian. At each iteration, the Cholesky factorization of $P_k^T H P_k + \sigma P_k^T P_k$ is used to compute a vector w_R such that

$$(P_k^T H P_k + \sigma P_k^T P_k) w_R = -P_k^T g.$$

The accuracy of an approximate solution of (2.12) is determined by preassigned tolerances κ_1, $\kappa_2 \in (0, 1)$. On termination, the approximate solution of (2.2) is $s = s_R + s_N$, where $s_R = P_k w_R$ and $s_N = P_k w_N$, with w_N defined as the zero vector if $(1 - \kappa_1)\delta \leq \|s_R\| \leq (1 + \kappa_1)\delta$, or a left-most eigenvector of $P_k^T H P_k + \sigma P_k^T P_k$ if

$\|s_R\| < (1-\kappa_1)\delta$. The resulting value of s satisfies

$$\mathcal{Q}(s) - \mathcal{Q}^* \leq \kappa_1(2-\kappa_1)\max(|\mathcal{Q}^*|, \kappa_2), \quad \text{and} \quad \|s\| \leq (1+\kappa_1)\delta, \tag{2.13}$$

where $\mathcal{Q}^*$ denotes the global minimum of (2.12) (see Moré and Sorensen [15]).

The calculations associated with the solution of the reduced problem are given in algorithm ***ssmSolve***, with $s_e = s_{k-1}$, $z = z_k$, and $s_a = s_k^a$. The inclusion of the best approximation s_{k-1} in $\text{span}\{s_{k-1}, z_k, s_k^a\}$ guarantees that $\mathcal{Q}$ decreases at each step. Care must be taken to separate the nullspace components of the Moré-Sorensen solution to test the optimality conditions correctly, i.e., both s_e and s_R are stored. The Moré-Sorensen algorithm also returns the optimal σ for the reduced problem, which is denoted by σ_e in ***ssmSolve***.

Algorithm *ssmSolve*.
$\big(s_e, s_R, \sigma_e, [Hs_e], [Hs_R]\big) = \textbf{\textit{ssmSolve}}\big(s_e, s_a, z, [Hs_e], [Hz], [Hs_a]\big)$;
Define P from a maximally linearly independent subset of s_e, z and s_a;
Form P^THP, P^TP and P^Tg from s_e, z, s_a, $[Hs]$, $[Hz]$, and $[Hs_a]$;
Find y, an approximate solution of $\min\big\{\, g^TPy + \frac{1}{2}y^TP^THPy \;:\; \|Py\|_2 \leq \delta \,\big\}$;
(The Moré-Sorensen method finds s_R and σ_e such that $P^T\big((H+\sigma_e I)s_R + g\big) = 0$);
$s_e := Py$; $[Hs_e] := [HP]y$;

2.4. Solving the trust-region subproblem. At the start of each subspace minimization, the regularization parameter μ is re-initialized at a fixed value μ_0 and reduced by a factor of two if a direction of negative curvature for (2.9) is found while computing the accelerator direction. More details are given in Algorithm ***IP-SSM*** below. A crucial feature of the method is that each subproblem is started with the σ_e and approximate eigenpair from the previous subproblem. At the start of each subproblem, the initial value of the interior-point accelerator variable σ_a is σ_e, as long as σ_e is larger than $\sigma_{\min}$, a preassigned constant that specifies the smallest allowable value of σ_a. (In the final iterations of the trust-region method, the trust-region constraint will be inactive and $\sigma_e = 0$.)

Algorithm *IP-SSM*.
$\big(s_e, [Hs_e], \sigma_e, z\big) = \textbf{\textit{IP-SSM}}\big(g, \delta, \sigma_e, z\big)$;
Specify $\tau > 0$; $k_{\max} > 0$; $\sigma_{\min} > 0$; $\mu_0 > 0$;
$\sigma_\ell := 0$; $\sigma_a := \max\{\sigma_e, \sigma_{\min}\}$; $\mu := \mu_0$;
$s_e := -g$; $[Hs_e] := Hs_e$; $s_a := 0$; $[Hs_a] := 0$;
$r_e := \|g + (H+\sigma_e I)s_e\|_{M^{-1}} + \sigma_e|c(s_e)|$;
while $k < k_{\max}$ **and** $r_e > \tau$ **do**
  $\big(s_a, [Hs_a], \sigma_a, \sigma_\ell, z, \zeta\big) := \textbf{\textit{ipAccelerator}}\big(s_a, [Hs_a], \sigma_a, \sigma_\ell\big)$;
  $\big(\tilde{s}, s_R, \tilde{\sigma}, [H\tilde{s}], [Hs_R]\big) := \textbf{\textit{ssmSolve}}\big(s_e, s_a, z, [Hs_e], [Hz], [Hs_a]\big)$;
  $\tilde{r} := \|g + (H+\tilde{\sigma}I)s_R\|_{M^{-1}} + \tilde{\sigma}|c(\tilde{s})|$;
  if $\tilde{r} \leq r_e$ **and** $\tilde{\sigma} > \sigma_\ell$ **then**
    $s_e := \tilde{s}$; $\sigma_e := \tilde{\sigma}$; $[Hs_e] := [H\tilde{s}]$; $r_e := \tilde{r}$;
  end
  $\big(s_a, \sigma_a, \sigma_e, [Hs_a]\big) := \textbf{\textit{safeguard}}(s_a, \sigma_a, s_e, \sigma_e, \zeta, z, \sigma_\ell, [Hs_a], [Hs_e], [Hz], \delta)$;

```
    if r_a < r_e/10 then σ_e := σ_a;
    if ipAccelerator found a direction of negative curvature then
        μ := μ/2;  Compute c(s_a);
        if c(s_a) + μ ≤ 0 then
            s_a := s_e;  σ_a := σ_e;  [Hs_a] := [Hs_e];  Compute c(s_a);
            if c(s_a) + μ ≤ 0 then
                s_a := δ × s_a/‖s_a‖_2;  [Hs_a] := [Hs_a]/‖s_a‖_2;
            end
        end
    end
    if r_a < τ and r_e > τ then
        if Q(s_a) ≤ Q(s_e) then
            s_e := s_a;  σ_e := σ_a;  break;
        end
    end
    σ_a := max{σ_a, σ_min};  k := k + 1;
end
```

Algorithm ***IP-SSM*** is terminated with final iterate (s, σ) given by either (s_e, σ_e) or (s_a, σ_a), depending on the values of the residuals r_e and r_a such that

$$r_e = \|g + (H + \sigma_e I)s_R\|_{M^{-1}} + \sigma_e |c(s_e)|, \quad \text{and} \tag{2.14a}$$

$$r_a = \|g + (H + \sigma_a I)s_a\|_{M^{-1}} + \sigma_a |c(s_a)|. \tag{2.14b}$$

The idea is to choose the iterate with the least residual, subject to the requirement that s improves on the Cauchy step. Given a positive tolerance τ, the final iterate is $(s, \sigma) = (s_e, \sigma_e)$ if $r_e \leq \tau$, or $(s, \sigma) = (s_a, \sigma_a)$ if $r_a \leq \tau < r_e$ and $\mathcal{Q}(s_a) \leq \mathcal{Q}(s_e)$. The initial value $s_e = -g$ guarantees that every subspace minimizer improves on the Cauchy step. The condition $\mathcal{Q}(s_a) \leq \mathcal{Q}(s_e)$ ensures that this improvement is inherited by the final point.

3. Numerical Results. Numerical results were obtained using MATLAB implementations of the solvers: Steihaug-Toint, GLTR and IP-SSM. For comparison purposes each solver was used within the same trust-region method, which was based on the combination line search/trust-region method proposed by Gertz [8,9]. In this method, an approximate solution s_j of the jth trust-region subproblem is used to update the trust-region iterate as $x_{j+1} = x_j + \alpha_j s_j$, where α_j is obtained a variant of the Wolfe line search (see, Erway et al. [4]). In the combination line-search trust-region algorithm given below, $\mathcal{Q}_j^-(s) = g_j^T s + \frac{1}{2}\big[s^T H_j s\big]_-$, where $[\,c\,]_-$ denotes the negative part of c, i.e., $[\,c\,]_- = \min\{\,0, c\,\}$. With this choice of quadratic model, the sufficient decrease condition on α_j is

$$f(x_j + \alpha_j s_j) - f(x_j) \geq \eta_1\, \mathcal{Q}_j^-(\alpha_j s_j), \tag{3.1}$$

where η_1 is a preassigned scalar such that $0 < \eta_1 < \frac{1}{2}$. The parameters were set to $\eta_1 = 10^{-4}$, $\eta_2 = \frac{1}{4}$, $\omega = \frac{9}{10}$, and $\gamma_3 = \frac{3}{2}$.

The initial trust-region radius was chosen as $\delta_0 = 1$. Thereafter, δ_j was updated as a function of α_j. Updating the trust-region radius in this way allowed the trust-region to adapt rapidly to changes in f. This property mitigated the bad effects of linking the trust-region norm to the current preconditioner (see the discussion of

Section 1). In particular, the reliability and efficiency of the Steihaug-Toint and GLTR solvers considerably improved when a conventional method for updating δ_j (see, e.g., pp. 116–117 of Conn, Gould and Toint [2]) was replaced by the line-search. For the remainder of the discussion we refer to the "Steihaug-Toint method", the "GLTR method" and the "IP-SSM method" as being the combination trust-region/line-search method with the appropriate solver.

Combination Line-Search/Trust-Region Algorithm.
Specify constants $0 < \eta_1 < \eta_2 < 1$; $0 < \eta_1 < \frac{1}{2}$; $0 < \eta_1 < \omega < 1$; $1 < \gamma_3$;
Choose x_0; $\delta_0 := 1$; $j := 0$;
while not *converged* **do**
  Find an approximate solution s_j for $\min\{\mathcal{Q}_j(s) : \|s\|_2 \le \delta_j\}$;
  Find α_j satisfying the Wolfe conditions:
    $f(x_j + \alpha_j s_j) \le f(x_j) + \eta_1 \mathcal{Q}_j^-(\alpha_j s_j)$ and $|g(x_j + \alpha_j s_j)^T s_j| \le -\omega \mathcal{Q}_j^-{}'(\alpha_j s_j)$;
  $x_{j+1} := x_j + \alpha_j s_j$;
  if $\big(f(x_{j+1}) - f(x_j)\big)/\mathcal{Q}_j^-(s_j) \ge \eta_2$ **then**
    if $\|s_j\|_2 = \delta_j$ **and** $\alpha_j = 1$ **then**
      $\delta_{j+1} := \gamma_3 \delta_j$;
    else if $\|s_j\|_2 < \delta_j$ **and** $\alpha_j = 1$ **then**
      $\delta_{j+1} := \max\{\delta_j, \gamma_3 \|s_j\|_2\}$;
    else
      $\delta_{j+1} := \alpha_j \|s_j\|_2$;
    end if
  else
    $\delta_{j+1} := \min\{\alpha_j \|s_j\|_2, \alpha_j \delta_j\}$;
  end if
  $j := j + 1$;
end do

The subspace trust-region subproblem was solved to an accuracy that was at least as good as that required for the full problem. The constants κ_1 and κ_2 of (2.13) were $\kappa_1 = \min\{10^{-1}\tau, 10^{-6}\}$ and $\kappa_2 = 0$, where τ is the accuracy required in the full space (see condition (3.2) below). The remaining parameters of ***IP-SSM*** were specified as $k_{\max} = 10$, $\mu_0 = 10^{-1}$ and $\sigma_{\min} = 100\sqrt{\epsilon_M}$, where ϵ_M is the machine precision.

In our implementation of the Steihaug-Toint solver, the Lanczos-CG process was terminated with a point s_j inside the trust region if

$$\|g_j + H_j s_j\|_{M_j^{-1}} \le \tau, \quad \text{where} \quad \tau = \min\left\{10^{-1}, \|g_j\|_{M_j^{-1}}^{0.1}\right\} \|g_j\|_{M_j^{-1}}, \tag{3.2}$$

The same condition is used by GLTR (see Gould et al. [10]). This τ was also used in the IP-SSM termination conditions (2.14). A limit of 20 Lanczos vectors was imposed on all calculations involving the Lanczos-CG process. If this limit was reached during the accelerator calculation, the Lanczos-CG iterate with the smallest residual was returned.

3.1. The test environment. Numerical results are given for unconstrained problems from the CUTEr test collection (see Bongartz et al. [1] and Gould, Orban

and Toint [11]). The test set was constructed using the CUTEr interactive `select` tool, which allows the identification of groups of problems with certain characteristics. In our case, the `select` tool was used to identify the twice-continuously differentiable unconstrained problems for which the number of variables can be varied. This process selected 67 problems: `arwhead`, `bdqrtic`, `broydn7d`, `brybnd`, `chainwoo`, `cosine`, `cragglvy`, `curly10`, `curly20`, `curly30`, `dixmaana`, `dixmaanb`, `dixmaanc`, `dixmaand`, `dixmaane`, `dixmaanf`, `dixmaang`, `dixmaanh`, `dixmaani`, `dixmaanj`, `dixmaank`, `dixmaanl`, `dixon3dq`, `dqdrtic`, `dqrtic`, `edensch`, `eg2`, `engval1`, `extrosnb`, `fletchcr`, `fletcbv2`, `fminsrf2`, `fminsurf`, `freuroth`, `genhumps`, `genrose`, `liarwhd`, `morebv`, `ncb20`, `ncb20b`, `noncvxu2`, `noncvxun`, `nondia`, `nondquar`, `penalty1`, `penalty2`, `powellsg`, `power`, `quartc`, `sbrybnd`, `schmvett`, `scosine`, `scurly10`, `scurly20`, `scurly30`, `sinquad`, `sparsine`, `sparsqur`, `spmsrtls`, `srosenbr`, `testquad`, `tointgss`, `tquartic`, `tridia`, `vardim`, `vareigvl` and `woods`. The dimensions were selected so that $n \geq 1000$, with a default of $n = 1000$ unless otherwise recommended in the CUTEr documentation. The problems `sbrybnd`, `scosine`, `scurly10`, `scurly20` and `scurly30` are scaled versions of `brybnd`, `cosine`, `curly10`, `curly20` and `curly30`. This scaling leads to a severely ill-conditioned Hessian for which a matrix-vector product has little or no precision, often causing a compete breakdown of the CG iterations. As all the methods under consideration are CG-based, the unpredictability of the testing process on these problems forced us to remove them from the test set, resulting in a final set of 62 problems.

Each solver (embedded within the same trust-region method as described above) was tested in three contexts associated with the underlying CG method: (i) CG with no preconditioning; (ii) CG with diagonal preconditioning; and (ii) CG with incomplete Cholesky preconditioning. For a given preconditioner, the runs for all three solvers are included in a single table to facilitate comparison. A method was considered to have solved a problem successfully when the iterate x_j satisfied

$$\|g(x_j)\|_2 \leq \max\{\epsilon\|g(x_0)\|_2, \epsilon|f(x_0)|, 10^{-5}\}, \tag{3.3}$$

with $\epsilon = 10^{-6}$. For each test problem we list the number of function evaluations ("fe") and matrix-vector products ("prds") required for each of the three solvers. Runs for which one of the methods converged to an alternate local minimizer are marked with an "a". A run was considered to have failed if the method could not satisfy condition (3.3) in $2n$ iterations. These "failed" runs are marked with an "$*$".

The CUTEr collection includes problems with a wide range of difficulty. However, many of the test problems for large-scale unconstrained optimization involve functions with a few variables that are extended artificially to higher dimension. Such problems are not necessarily representative of problems that arise in practice. The evaluation of methods is complicated further by the fact that many of the problems are variants of one case (see, e.g., the problems `dixmaana`–`dixmaanl`). Typically, a method will behave in a similar way on all the problems of one type, which can distort the results of numerical tests (such as performance profiles, see Section 3.5).

3.2. Results obtained without preconditioning. Tables 2–3 give results on 57 of the 62 CUTEr problems. The tables do not include results for the 5 problems `curly10`, `curly20`, `curly30`, `dixon3dq`, and `genhumps`, which could not be solved by any method without preconditioning. In addition, the table gives the percentage improvement in function evaluations relative to the Steihaug-Toint method. Of the 57 problems listed, the Steihaug-Toint method solved 56, GLTR solved 54, and IPSSM solved 57. The methods converged to the same local minimizer in every case.

An "at-a-glance" comparison is afforded by Table 1. This table gives the cumulative totals on the 54 problems for which all methods converged.

TABLE 1
Summary of methods with no preconditioning.

	Steihaug	GLTR	IP-SSM
Function evals (fe)	4218	3540	1888
Matrix mults (prds)	10519	58918	54300
Improvement in fe	—	+16%	+55%

IP-SSM required 55% fewer function evaluations than Steihaug-Toint, compared to the 16% reduction provided by GLTR. The results for GLTR are comparable with those obtained by Gould et al. [10], who report that GLTR solved 16 of a set of 17 CUTEr problems and required 12.5% fewer function evaluations than Steihaug-Toint. Table 1 also indicates that for both IP-SSM and GLTR, the decrease in function evaluations is achieved at the expense of additional matrix-vector products. For some of the Steihaug-Toint runs (e.g., `genrose`, `dqdrtic` and `fminsurf`) the number of function evaluations exceeds the number of matrix-vector products. In these cases the line-search needed to work hard to improve the solution estimate when trust-region acceptance test failed. The use of a line search in this situation contributed significantly to the reliability of the solvers.

3.3. Diagonal preconditioning. The methods were tested with a diagonal preconditioner based on the matrix $D = \text{diag}(d_1, d_2, \ldots, d_n)$ of diagonals of the Hessian evaluated at x_j. For the Steihaug-Toint and GLTR solvers, the elements of the diagonal preconditioner M were:

$$M_{ii} = \max\{|d_i|, \|D\|_2/\texttt{condmax}\},$$

where `condmax` is a preassigned upper bound on the condition number of M. Similarly, the IP-SSM preconditioner was based on $D + \sigma_a I$, i.e.,

$$M_{ii} = \max\{|d_i + \sigma_a|,\ \|D + \sigma_a I\|_2/\texttt{condmax}\}.$$

The value of `condmax` was 10^8 in all runs. Tables 5–6 give the results for diagonal preconditioning. They do not include runs for the 4 problems: `curly10`, `curly20`, `curly30`, and `dixon3dq`, which could not be solved by any method. We also omitted the 5 problems `fminsurf`, `penalty1`, `penalty2`, `power` and `vareigvl` because of MATLAB memory limitations when extracting the diagonals from the CUTEr Hessian. Steihaug-Toint solved all the remaining 53 problems, GLTR solved 48 and IP-SSM solved 51. The methods converged to the same local minimizer in every case except `testquad`. Table 4 gives the cumulative totals on the 46 problems for which all three methods converged and converged to the same local minimizer. Compared to Steihaug-Toint, GLTR and IP-SSM required 1% and 49% fewer function evaluations respectively. Overall, the use of diagonal preconditioning appeared to marginally improve the robustness of all the methods.

Ideally, a preconditioner should reduce the number of matrix-vector products without increasing the number of function evaluations. This appeared to be the case for IP-SSM. Of the 46 problems included in the summary of Table 4, diagonally preconditioned IP-SSM required more function evaluations than unpreconditioned IP-SSM for

TABLE 2
CUTEr problems A–E. No preconditioning.

	Steihaug		GLTR		IP-SSM	
Problem	fe	prds	fe	prds	fe	prds
arwhead	6	6	6	6	6	21
bdqrtic	14	41	13	40	13	66
broydn7d	137	400	71	1279	48	797
brybnd	12	46	13	56	12	81
chainwoo	27	57	20	112	15	85
cosine	12	10	13	11	12	37
cragglvy	14	36	14	35	14	36
dixmaana	13	11	12	15	13	11
dixmaanb	13	11	13	12	13	11
dixmaanc	13	11	13	12	13	11
dixmaand	14	12	14	12	14	12
dixmaane	15	82	14	71	15	82
dixmaanf	15	30	15	44	15	30
dixmaang	15	24	15	24	15	24
dixmaanh	15	19	15	19	15	19
dixmaani	19	162	19	171	14	153
dixmaanj	16	40	16	40	16	40
dixmaank	16	30	16	30	16	30
dixmaanl	16	25	16	25	16	25
dqdrtic	14	11	13	14	13	28
dqrtic	27	20	28	24	28	60
edensch	15	25	15	33	15	59
eg2	4	3	4	3	11	217
engval1	14	17	14	18	14	45
extrosnb	31	70	31	82	25	95

9 problems only (20% of the cases). By contrast, diagonally preconditioned Steihaug-Toint required more function evaluations for 39 problems (85%). GLTR required more function evaluations for 41 problems (89%). In some cases, Steihaug-Toint and GLTR required substantially more function evaluations than the unpreconditioned case, see, e.g., ncb20 and noncvxu2. It could be argued that the overall statistics are unfairly influenced by the block of 12 problems dixmaana–dixmaanl, which tend to exhibit similar behavior when tested (and hence distort any conclusions based on the results of battery testing). If results from these problems are excluded from the totals, the overall increase in function evaluations for Steihaug-Toint, GLTR and IP-SSM decreases to 82%, 88% and 18% respectively.

As the main purpose of preconditioning is to reduce the number of CG iterations (and hence the number of matrix-vector products) it is useful to consider the number of products before and after preconditioning. As above, data was considered for the 46 problems summarized in Table 4. Compared to no preconditioning, there is a reduction in matrix-vector products for 23 problems (50% of the cases) for Steihaug-Toint, 20 problems (43%) for GLTR, and 27 problems (59%) for IP-SSM. If the problems dixmaana–dixmaanl are excluded, the percentage of improved cases is 44%, 41% and 71% respectively. These results indicate that the fixed trust-region norm used by IP-SSM leads to some improvements in function evaluations and matrix-vector products

TABLE 3
CUTEr problems F–Z. No preconditioning.

	Steihaug		GLTR		IP-SSM	
Problem	fe	prds	fe	prds	fe	prds
fletchcr	1998	16422	*	*	1751	21325
fletcbv2	1513	3024	1515	30318	90	17940
fminsrf2	*	*	*	*	71	1666
fminsurf	340	93	82	1332	56	1830
freuroth	16	1	21	49	16	49
genrose	1174	577	808	17916	830	23743
liarwhd	19	2	19	37	17	63
morebv	27	511	27	511	5	427
ncb20	153	2205	117	2305	74	2848
ncb20b	10	61	9	65	9	103
noncvxu2	36	26	44	240	43	298
noncvxun	37	28	42	162	47	298
nondia	4	3	4	3	5	15
nondquar	23	115	18	101	22	248
penalty1	28	17	28	17	28	52
penalty2	2	1	2	1	2	4
powellsg	16	40	15	52	15	102
power	15	33	15	33	15	61
quartc	27	20	28	24	28	61
schmvett	9	37	*	*	7	57
sinquad	19	27	19	36	17	92
sparsine	17	197	15	167	11	324
sparsqur	14	23	14	30	14	66
spmsrtls	18	117	20	213	29	684
srosenbr	9	10	9	10	9	36
testquad	72	1259	115	2138	16	937
tointgss	15	13	14	19	15	47
tquartic	18	23	17	25	15	53
tridia	50	842	50	869	19	1709
vardim	13	12	13	12	13	41
vareigvl	14	26	14	27	14	56
woods	12	14	13	18	13	38

from preconditioning.

TABLE 4
Summary of methods with diagonal preconditioning.

	Steihaug	GLTR	IP-SSM
Function evals (fe)	4830	4789	2455
Matrix mults (prds)	17072	21619	24054
Improvement in fe	—	+1%	+49%

3.4. Incomplete Cholesky preconditioning. The methods were also run with a preconditioner based on an incomplete Cholesky factorization. These tests used the software (`icfs`), which implements the factorization proposed by Lin and

TABLE 5
CUTEr problems A–E. Diagonal preconditioning.

	Steihaug		GLTR		IP-SSM	
Problem	fe	prds	fe	prds	fe	prds
`arwhead`	15	11	14	15	6	21
`bdqrtic`	22	19	21	19	13	53
`broydn7d`	192	330	169	1091	48	645
`brybnd`	20	19	19	22	12	45
`chainwoo`	27	33	24	36	14	59
`cosine`	23	20	19	23	12	30
`cragglvy`	29	40	27	39	14	59
`dixmaana`	17	13	16	15	12	38
`dixmaanb`	17	12	16	15	12	37
`dixmaanc`	19	14	18	14	13	40
`dixmaand`	21	16	20	15	14	44
`dixmaane`	24	26	23	48	13	41
`dixmaanf`	20	22	19	21	16	59
`dixmaang`	20	20	20	20	16	60
`dixmaanh`	21	19	21	20	16	59
`dixmaani`	23	27	35	151	13	51
`dixmaanj`	20	22	20	23	16	60
`dixmaank`	21	22	21	22	16	60
`dixmaanl`	22	21	22	21	16	59
`dqdrtic`	21	12	18	17	13	26
`dqrtic`	50	28	50	28	28	64
`edensch`	24	21	23	28	15	53
`eg2`	8	6	8	6	25	257
`engval1`	19	15	19	15	13	37
`extrosnb`	39	48	40	86	26	91

Moré [14]. The `icfs` software requires the specification of an integer p that limits the fill in the Cholesky factors to pn elements. The suggested default value of $p = 5$ was used in all cases.

For the Steihaug-Toint and GLTR solvers, the preconditioner was the Lin-Moré factorization of the Hessian H_j. If the Hessian is not positive definite, the Lin-Moré algorithm computes an incomplete factorization of a positive-definite matrix $H_j + \nu I$, where ν is a positive scalar. Several factorizations may be necessary before an appropriate value of ν is found. For IP-SSM, the preconditioner was the incomplete Cholesky factorization of the positive-definite matrix $H_j + \sigma_a I$, where σ_a is the initial value of the accelerator variable (usually σ_e, see Algorithm ***ipAccelerator***). Clearly, it is beneficial to apply the incomplete Cholesky factorization to a positive-definite matrix. However, the preconditioner becomes less effective as the subspace minimizations proceed because σ_a changes from its initial value.

A total of 57 of the 62 problems were run with incomplete Cholesky preconditioning. The five problems `fminsurf`, `penalty1`, `penalty2`, `power` and `vareigvl` were omitted because of memory limitations (our implementation of GLTR required a solve with a dense preconditioner). Tables 7–8 give details of the runs using incomplete Cholesky preconditioning. Steihaug-Toint and IP-SSM solved 56 of the 57 problems, GLTR solved 50. As with diagonal preconditioning, the methods converged

TABLE 6
CUTEr problems F–Z. Diagonal preconditioning.

	Steihaug		GLTR		IP-SSM	
Problem	fe	prds	fe	prds	fe	prds
fletchcr	1665	13557	1671	14827	1617	17724
fletcbv2	1513	30240	1515	30318	*	*
fminsrf2	199	293	*	*	94	3695
freuroth	23	22	23	22	14	39
genhumps	1488	1410	*	*	*	*
genrose	1863	3758	*	*	821	10560
liarwhd	26	27	26	28	17	75
morebv	25	474	25	474	7	895
ncb20	432	431	477	1392	84	1451
ncb20b	10	60	10	79	9	64
noncvxu2	745	505	702	932	46	302
noncvxun	656	447	627	815	46	281
nondia	83	63	59	85	5	18
nondquar	26	98	27	172	16	126
powellsg	23	40	22	52	16	101
quartc	50	28	50	28	28	64
schmvett	14	25	*	*	8	44
sinquad	35	31	30	61	16	75
sparsine	30	158	66	360	11	240
sparsqur	25	26	25	26	14	59
spmsrtls	70	125	87	279	24	241
srosenbr	20	17	*	*	9	34
testquad	28^a	16^a	24^a	23^a	10^a	29^a
tointgss	21	17	19	31	14	34
tquartic	16	24	12	28	11	46
tridia	24	27	22	35	12	54
vardim	56	34	56	34	13	44
woods	27	26	27	26	13	44

to the same local minimizer in every case except `testquad`. In terms of the number of problems solved, incomplete Cholesky preconditioning was the most reliable preconditioner. However, in terms the number of function evaluations and matrix-vector products, the performance of all three methods was somewhat erratic—which we interpret as evidence of the difficulty of formulating "general-purpose" preconditioners. Although a given problem was more likely to be solved using incomplete Cholesky preconditioning, the less challenging problems often required more function evaluations and matrix-vector products. Table 9 gives cumulative totals for the 49 problems on which all three methods converged (with the exception of `testquad` as discussed above). Compared to Steihaug-Toint, GLTR required 29% more function evaluations. IP-SSM required 40% fewer function evaluations.

Of the 49 problems included in the summary of Table 9, incomplete Cholesky preconditioned IP-SSM required more function evaluations than unpreconditioned IP-SSM for 12 problems (25% of the cases). The corresponding numbers for Steihaug-Toint and GLTR were 35 (71% of the cases) and 38 (78% of the cases) respectively. If the statistics for the 9 solved problems from `dixmaana`–`dixmaanl` are excluded, the

percentage of cases for which function evaluations increased becomes 68%, 75% and 20% for Steihaug-Toint, GLTR and IP-SSM respectively.

Compared to no preconditioning, a reduction in matrix-vector products was seen in 28 problems (57% of the cases) for Steihaug-Toint, 27 problems (55%) for GLTR and 35 problems (71%) for IP-SSM. Incomplete Cholesky preconditioning appeared to be less effective for problems `dixmaana`–`dixmaanl`. If the solved instances of these problems are excluded, the percentages increase to 58%, 62% and 80% respectively. Again, there is a marked improvement for IP-SSM compared to Steihaug-Toint and GLTR.

TABLE 7
CUTEr problems A–E. ICFS preconditioning.

	Steihaug		GLTR		IP-SSM	
Problem	fe	prds	fe	prds	fe	prds
`arwhead`	14	10	14	10	7	22
`bdqrtic`	22	15	22	15	11	41
`broydn7d`	124	73	150	187	38	143
`brybnd`	19	14	21	19	16	49
`chainwoo`	100	71	85	123	72	244
`cosine`	29	20	28	33	11	23
`cragglvy`	27	19	27	19	14	43
`curly10`	40	22	44	41	16	141
`curly20`	27	19	33	40	15	129
`curly30`	60	32	35	38	14	116
`dixmaana`	16	11	15	20	12	35
`dixmaanb`	18	12	17	16	12	32
`dixmaanc`	28	19	89	104	13	36
`dixmaand`	49	28	62	90	20	69
`dixmaane`	46	28	177	256	13	40
`dixmaanf`	29	20	*	*	20	63
`dixmaang`	37	22	*	*	27	78
`dixmaanh`	35	21	*	*	39	175
`dixmaani`	26	19	173	252	13	37
`dixmaanj`	29	20	31	40	21	77
`dixmaank`	39	26	32	46	35	163
`dixmaanl`	42	28	44	68	36	134
`dixon3dq`	4	3	4	3	12	58
`dqdrtic`	21	12	18	17	13	26
`dqrtic`	50	28	50	28	28	64
`edensch`	25	18	23	22	16	46
`eg2`	8	6	8	6	18	131
`engval1`	19	13	19	13	13	32
`extrosnb`	38	27	36	32	27	78

3.5. Performance profiles. The results of Tables 2–8 are summarized using performance profiles (see Dolan and Moré [3]). Let $\mathrm{card}(\mathcal{S})$ denote the number of elements in a finite set $\mathcal{S}$. Let $\mathcal{P}$ denote the set of problems used for a given numerical experiment. For each method s we define the function $\pi_s : [0, r_M] \mapsto \mathbb{R}^+$ such that

$$\pi_s(\tau) = \frac{1}{\mathrm{card}(\mathcal{P})} \mathrm{card}(\{p \in \mathcal{P} : \log_2(r_{p,s}) \leq \tau\}),$$

TABLE 8
CUTEr problems F–Z. ICFS preconditioning.

	Steihaug		GLTR		IP-SSM	
Problem	fe	prds	fe	prds	fe	prds
fletchcr	*	*	*	*	1708	5717
fletcbv2	2	1	2	1	5	160
fminsrf2	24	21	23	19	56	213
freuroth	22	14	22	14	13	27
genhumps	983	973	*	*	*	*
genrose	1375	866	1396	1680	823	2630
liarwhd	27	28	27	31	17	65
morebv	2	1	2	1	3	11
ncb20	123	79	211	342	94	416
ncb20b	18	11	*	*	14	49
noncvxu2	82	56	228	424	45	154
noncvxun	65	45	215	407	46	126
nondia	21	15	19	20	6	19
nondquar	20	17	20	17	16	109
powellsg	24	18	22	21	15	52
quartc	50	28	50	28	28	64
schmvett	40	23	8	6	7	19
sinquad	28	17	26	39	19	72
sparsine	45	203	50	245	11	149
sparsqur	26	19	26	19	14	44
spmsrtls	23	15	210	223	28	106
srosenbr	20	14	*	*	9	28
testquad	28^a	16^a	24^a	23^a	10^a	29^a
tointgss	13	8	12	11	16	37
tquartic	14	24	11	26	11	46
tridia	19	11	17	16	10	31
vardim	64	38	64	38	13	38
woods	27	18	27	18	13	38

TABLE 9
Summary of methods with ICFS preconditioning.

	Steihaug	GLTR	IP-SSM
Function evals (fe)	3084	3969	1835
Matrix mults (prds)	2174	5207	6664
Improvement in fe	—	−29%	+40%

where $r_{p,s}$ denotes the ratio of the number of function evaluations needed to solve problem p with method s and the least number of function evaluations needed to solve problem p. The number r_M is the maximum value of $\log_2(r_{p,s})$. Figures 1–3 depict the functions π_s for each of the methods tested. All performance profiles are plotted on a $\log_2$-scale.

4. Discussion and Summary. We have considered an interior-point sequential subspace minimization method (IP-SSM) that solves the inequality constrained trust-region subproblem over a sequence of evolving low-dimensional subspaces. Each subspace includes an accelerator direction defined by a regularized Newton method

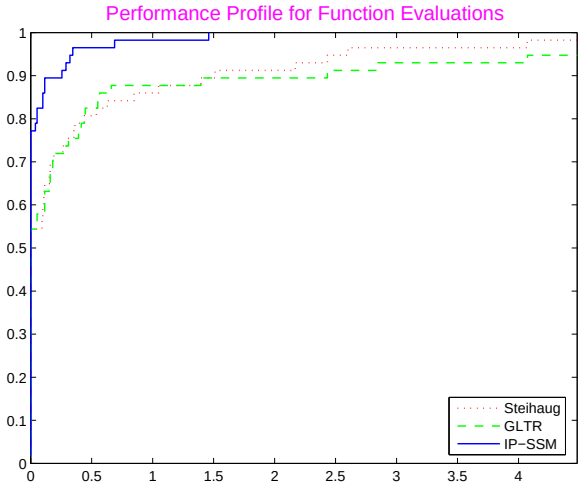

FIG. 1. *Function evaluations for methods without preconditioning.*

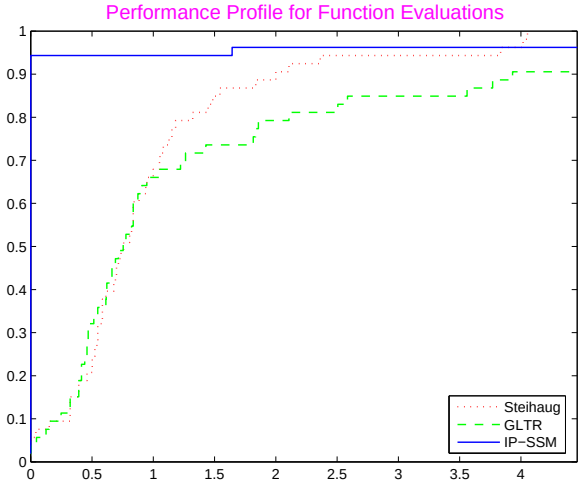

FIG. 2. *Function evaluations for methods with diagonal preconditioning.*

for the optimality conditions of a primal-dual interior method. A crucial property of this direction is that it can be computed by applying the preconditioned CG method to a positive-definite system in both the primal and dual variables of the trust-region subproblem. IP-SSM has several properties. (i) The method does not require the definition of the trust-region to depend on the CG preconditioner. This allows the application of preconditioning in conjunction with a conventional method for updating the trust-region radius. (ii) The likelihood of approximating the global minimizer of the trust region subproblem is increased by the computation of an approximate

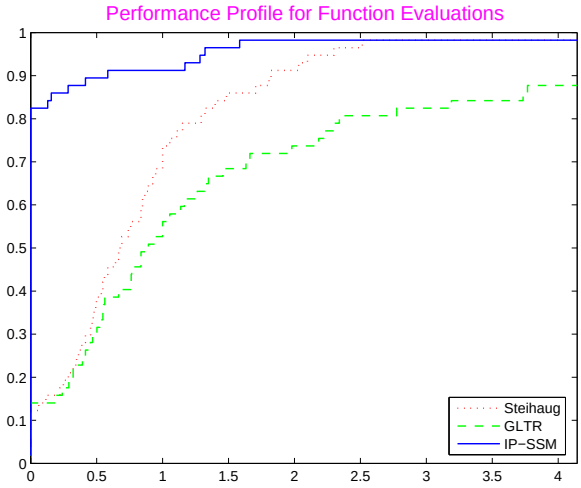

FIG. 3. *Function evaluations for methods with* `icfs` *preconditioning.*

left-most eigenpair of the Hessian. (iii) Information garnered during the solution of one trust-region subproblem may be used to expedite the solution of the next.

Numerical experiments on problems from the CUTEr test collection indicate that IP-SSM can require significantly fewer function evaluations than Steihaug-Toint and GLTR. This implies that IP-SSM may be more efficient when the cost of a function evaluation is expensive relative to the cost of a matrix-vector product. As is the case with other iterative trust-region methods, the decrease in function evaluations is achieved at the expense of additional matrix-vector products. However, in contrast to Steihaug-Toint and GLTR, our experiments with general-purpose CG preconditioners show that it is possible to significantly reduce the number of matrix-vector products relative to those required without preconditioning. Moreover, these savings may be achieved without increasing the number of function evaluations. This implies that the use of custom preconditioners (such as those available for PDE constrained optimization) have the potential of substantially reducing the cost of large-scale minimization, regardless of the relative cost of evaluating the objective function and forming a matrix-vector product.

Acknowledgments. We would like to thank Nick Gould for graciously providing a MATLAB implementation of GLTR. We are also grateful to the referees for constructive comments that significantly improved the presentation.

REFERENCES

[1] I. BONGARTZ, A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *CUTE: Constrained and unconstrained testing environment*, ACM Trans. Math. Softw., 21 (1995), pp. 123–160.

[2] A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *Trust-Region Methods*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[3] E. D. DOLAN AND J. J. MORÉ, *Benchmarking optimization software with COPS*, Technical Memorandum ANL/MCS-TM-246, Argonne National Laboratory, 2000.

[4] J. B. Erway, P. E. Gill, and J. D. Griffin, *Iterative methods for finding a trust-region step*, Numerical Analysis Report 07-2, Department of Mathematics, University of California San Diego, La Jolla, CA, 2007. To appear in SIAM J. Optim.

[5] A. Forsgren, P. E. Gill, and J. D. Griffin, *Iterative methods for systems arising in interior-point methods*, SIAM J. Optim., 18 (2007), pp. 666–690.

[6] A. Forsgren, P. E. Gill, and M. H. Wright, *Interior methods for nonlinear optimization*, SIAM Rev., 44 (2002), pp. 525–597.

[7] D. M. Gay, *Computing optimal locally constrained steps*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 186–197.

[8] E. M. Gertz, *Combination Trust-Region Line-Search Methods for Unconstrained Optimization*, PhD thesis, Department of Mathematics, University of California San Diego, 1999.

[9] E. M. Gertz, *A quasi-Newton trust-region method*, Math. Program., 100 (2004), pp. 447–470.

[10] N. I. M. Gould, S. Lucidi, M. Roma, and P. L. Toint, *Solving the trust-region subproblem using the Lanczos method*, SIAM J. Optim., 9 (1999), pp. 504–525.

[11] N. I. M. Gould, D. Orban, and P. L. Toint, *CUTEr and SifDec: A constrained and unconstrained testing environment, revisited*, ACM Trans. Math. Softw., 29 (2003), pp. 373–394.

[12] W. W. Hager, *Minimizing a quadratic over a sphere*, SIAM J. Optim., 12 (2001), pp. 188–208 (electronic).

[13] W. W. Hager and S. Park, *Global convergence of SSM for minimizing a quadratic over a sphere*, Math. Comp., 74 (2004), pp. 1413–1423.

[14] C.-J. Lin and J. J. Moré, *Incomplete Cholesky factorizations with limited memory*, SIAM J. Sci. Comput., 21 (1999), pp. 24–45 (electronic).

[15] J. J. Moré and D. C. Sorensen, *Computing a trust region step*, SIAM J. Sci. and Statist. Comput., 4 (1983), pp. 553–572.

[16] J. J. Moré and D. C. Sorensen, *Newton's method*, in Studies in Mathematics, Volume 24. Studies in Numerical Analysis, Math. Assoc. America, Washington, DC, 1984, pp. 29–82.

[17] F. Rendl and H. Wolkowicz, *A semidefinite framework for trust region subproblems with applications to large scale minimization*, Math. Programming, 77 (1997), pp. 273–299.

[18] M. Rojas, S. A. Santos, and D. C. Sorensen, *A new matrix-free algorithm for the large-scale trust-region subproblem*, SIAM J. Optim., 11 (2000/01), pp. 611–646 (electronic).

[19] M. Rojas and D. C. Sorensen, *A trust-region approach to the regularization of large-scale discrete forms of ill-posed problems*, SIAM J. Sci. Comput., 23 (2002), pp. 1842–1860 (electronic).

[20] D. C. Sorensen, *Newton's method with a model trust region modification*, SIAM J. Numer. Anal., 19 (1982), pp. 409–426.

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

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

[23] P. L. Toint, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, I. S. Duff, ed., London and New York, 1981, Academic Press, pp. 57–88.