

ITERATIVE METHODS FOR FINDING A TRUST-REGION STEP*

JENNIFER B. ERWAY[†], PHILIP E. GILL[‡], AND JOSHUA D. GRIFFIN[§]

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 inequality constraint. The Steihaug–Toint method uses the conjugate-gradient 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. The benefit of this approach is that an approximate solution may be obtained with minimal work and storage. However, the method does not allow the accuracy of a constrained solution to be specified. We propose an extension of the Steihaug–Toint method that allows a solution to be calculated to any prescribed accuracy. If the Steihaug–Toint point lies on the boundary, the constrained problem is solved on a sequence of evolving low-dimensional subspaces. Each subspace includes an accelerator direction obtained from a regularized Newton method applied to the constrained problem. A crucial property of this direction is that it can be computed by applying the conjugate-gradient method to a positive-definite system in both the primal and dual variables of the constrained problem. The method includes a parameter that allows the user to take advantage of the tradeoff between the overall number of function evaluations and matrix-vector products associated with the underlying trust-region method. At one extreme, a low-accuracy solution is obtained that is comparable to the Steihaug–Toint point. At the other extreme, a high-accuracy solution can be specified that minimizes the overall number of function evaluations at the expense of more matrix-vector products.

Key words. large-scale unconstrained optimization, trust-region methods, conjugate-gradient method, Lanczos tridiagonalization process

AMS subject classifications. 49J20, 49J15, 49M37, 49D37, 65F05, 65K05, 90C30

DOI. 10.1137/070708494

1. Introduction. The j th iteration of a two-norm trust-region method for unconstrained minimization involves finding an approximate solution of the trust-region subproblem

$$(1.1) \quad \underset{s \in \mathbb{R}^n}{\text{minimize}} \quad \mathcal{Q}_j(s) \equiv g_j^T s + \frac{1}{2} s^T H_j s \quad \text{subject to} \quad \|s\|_2 \leq \delta_j,$$

where δ_j is a given positive trust-region radius and $\mathcal{Q}_j(s)$ is a quadratic model of $f(x_j + s) - f(x_j)$, the change in the objective function. The quantities g_j and H_j are usually the gradient $\nabla f(x)$ and Hessian $\nabla^2 f(x)$ at x_j .

Many methods for solving (1.1) exploit the properties of direct matrix factorizations (see, e.g., [3, 8, 9, 10, 17, 19, 28]). In particular, the method of Moré and Sorensen [19] makes repeated use of the Cholesky factorization of a positive-semidefinite matrix. A feature of the Moré–Sorensen method is that it involves a scalar parameter κ_1 that specifies the accuracy of an approximate solution of (1.1). For a given problem, the optimal accuracy involves a tradeoff between the cost of the

*Received by the editors November 18, 2007; accepted for publication (in revised form) May 5, 2009; published electronically August 12, 2009. Research supported by National Science Foundation grants DMS-9973276 and DMS-0511766.

<http://www.siam.org/journals/siopt/20-2/70849.html>

[†]Department of Mathematics, 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).

[§]SAS Institute Inc., Cary, NC 27513 (Joshua.Griffin@sas.com).

linear algebra associated with solving (1.1) and the cost of evaluating f and its derivatives. Broadly speaking, increasing the accuracy of the trust-region solution decreases the overall number of trust-region subproblems, but increases the number of evaluations of f and its derivatives. As these costs are problem dependent, the parameter κ_1 allows the trust-region solver to be tailored to suite a particular problem.

Methods based on matrix factorization are designed to solve problems for which the cost of the factorization is not excessive—e.g., if n is sufficiently small or H_j is sufficiently sparse. However, many problems are sufficiently large that it becomes necessary to exploit structure in H_j in order to solve equations of the form $H_j u = v$ efficiently (this includes, but is not restricted to, the case where H_j is a large sparse matrix). In these cases it is necessary to use iterative methods for the solution of the constituent linear equations (see, e.g., [2, 12, 24, 25, 26, 29, 30, 31]). A crucial property of such methods is that H_j is used only as an operator for the definition of matrix-vector products of the form $H_j v$. This means that the linear algebra overhead associated with optimization methods based on iterative solvers is directly proportional to the average number of matrix-vector products.

Steihaug [30] and Toint [31] define iterative methods for (1.1) based on the properties of the conjugate-gradient (CG) method. 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 CG method to the Newton equations. This process is equivalent to minimizing $\mathcal{Q}_j$ over a sequence of expanding subspaces generated by the CG 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 point of intersection with the trust-region boundary. When H_j is not positive 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, 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. This difficulty was observed by Gould, Lucidi, Roma, and Toint [12], who proposed that the Steihaug–Toint procedure be supplemented by the generalized Lanczos trust-region (GLTR) method, which finds a constrained minimizer of (1.1) over the sequence of expanding subspaces associated with the Lanczos process for reducing H_j to tridiagonal form. This method is discussed further in section 2.

Finally, we note that Krylov-based iterative methods have also been proposed by Sorensen [29], Rojas and Sorensen [27], Rojas, Santos, and Sorensen [25, 26], and Rendl and Wolkowicz [24]. These methods are different from those considered here in the sense that they do not start by minimizing $\mathcal{Q}_j$ as an unconstrained function. Instead, they find an approximate solution of (1.1) by estimating the eigenvalues of a matrix defined by augmenting H_j by a row and column.

1.1. Overview of the proposed method. Here we propose the *phased sequential subspace minimization (phased-SSM) method*, which, like the GLTR method, has a constrained second phase if there is no minimizer of $\mathcal{Q}_j$ inside the trust region. The iterates of the second phase optimize on the trust-region boundary using a sequential subspace minimization (SSM) method. Methods of this type, first proposed by Hager [15] for minimizing $g^T x + \frac{1}{2} x^T H x$ subject to the equality constraint $x^T x = \delta^2$,

approximate a constrained minimizer over a sequence of evolving low-dimensional subspaces of constant dimension (see also, Griffin [14] and Erway [7]).

Phased-SSM has several features that distinguish it from existing methods. First, a simple inexpensive estimate of the smallest eigenvalue of H is computed in both the constrained and unconstrained phases. This estimate extends the Steihaug–Toint method to the case where $g = 0$ and provides a better point on the constraint boundary to start the second phase. In addition, the low-dimensional subspace used in the second phase includes an accelerator direction obtained from a regularized Newton method applied to the constrained problem. Finally, the method incorporates a parameter that allows the user to take advantage of the tradeoff between the overall number of function evaluations and matrix-vector products.

Our numerical results indicate that if a low-accuracy solution is requested, phased-SSM requires fewer function evaluations than Steihaug’s method with a negligible increase in matrix-vector products. Moreover, solving the subproblem more accurately results in a further decrease in the overall number of function evaluations at the cost of increasing the number of matrix-vector products. Based on the results, it appears that phased-SSM would be particularly effective in applications where the cost of a function evaluation is expensive relative to the cost of a matrix-vector product (e.g., in simulation-based applications).

The paper is organized in five sections. In section 2, we discuss the trust-region subproblem (1.1) and review related work, including the Steihaug–Toint method, the GLTR method [12], and Hager’s SSM method. The phased-SSM method is described in section 3. Section 4 includes numerical results that compare the phased-SSM method with the Steihaug–Toint method on large unconstrained problems from the CUTER test collection (see Bongartz et al. [1] and Gould, Orban, and Toint [13]). Finally, section 5 includes some concluding remarks and observations.

1.2. Notation and glossary. Unless explicitly indicated, $\|\cdot\|$ denotes the vector two-norm or its subordinate matrix norm. The symbol e_j denotes the j th column of the identity matrix I , where the dimensions of e_j and I depend on the context. The eigenvalues of a real symmetric matrix H are denoted by $\{\lambda_j\}$, where $\lambda_n \leq \lambda_{n-1} \leq \dots \leq \lambda_1$. The associated eigenvectors are denoted by $\{u_j\}$. An eigenvalue λ and a corresponding normalized eigenvector u such that $\lambda = \lambda_n$ are known as the *leftmost eigenpair* of H . The matrix $A^\dagger$ denotes the Moore–Penrose pseudoinverse of A . 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 labeled $[Ax]$. Similarly, if P is a matrix with columns $p_1, p_2, \dots, p_m$, then $[AP]$ denotes the matrix of computed columns $[Ap_1], [Ap_2], \dots, [Ap_m]$.

2. Background: The constrained trust-region subproblem. In this section we drop the suffix j and focus on a single trust-region subproblem of the form

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

The optimality conditions for this problem are summarized in the following result (see, e.g., Gay [8], Sorensen [28], Moré and Sorensen [20], or Conn, Gould and Toint [4]).

THEOREM 2.1. *Let δ be a given positive constant. A vector s^* is a global solution of the trust-region problem (2.1) if and only if $\|s^*\| \leq \delta$ and there exists a unique*

$\sigma^* \geq 0$ such that $H + \sigma^* I$ is positive semidefinite with

$$(2.2) \quad (H + \sigma^* I)s^* = -g \quad \text{and} \quad \sigma^*(\delta - \|s^*\|) = 0.$$

Moreover, if $H + \sigma^* I$ is positive definite, then the global minimizer is unique.

The trust-region problem is said to be *degenerate* if $\sigma^* = -\lambda_n$ and $\|s_L\| < \delta$, where s_L is the least-length solution of the (necessarily compatible) system $(H + \sigma^* I)s = -g$, (i.e., $s_L = -(H - \lambda_n I)^\dagger g$). In this case, the system $(H + \sigma I)s = -g$ cannot be used alone to determine s^* . However, the eigenvector u_n is a null vector of $H - \lambda_n I$, and there exists a scalar τ such that

$$(H - \lambda_n I)(s_L + \tau u_n) = -g \quad \text{and} \quad \|s_L + \tau u_n\| = \delta.$$

In this case, $\sigma = -\lambda_n$ and $s = s_L + \tau u_n$ satisfy the optimality conditions (2.2) and thereby constitute a global solution of (2.1).

The preferred method for solving (2.1) using direct linear solvers is the Moré–Sorensen method [19]. The accuracy of an approximate solution is specified by the tolerances $\kappa_1, \kappa_2 \in (0, 1)$. At each iteration, the Cholesky factorization of $H + \sigma I$ is used to compute the vector q such that

$$(2.3) \quad (H + \sigma I)q = -g.$$

If $\|q\| < (1 - \kappa_1)\delta$, then an approximate null vector z is also computed. A safeguarding scheme is used to ensure that σ remains within $(-\lambda_n, \infty)$. The Moré–Sorensen algorithm gives an approximate solution s that satisfies

$$(2.4) \quad \mathcal{Q}(s) - \mathcal{Q}^* \leq \kappa_1(2 - \kappa_1) \max(|\mathcal{Q}^*|, \kappa_2) \quad \text{and} \quad \|s\| \leq (1 + \kappa_1)\delta,$$

where $\mathcal{Q}^*$ denotes the global minimum of (2.1). In the context of a standard underlying trust-region method, the Moré–Sorensen algorithm gives convergence to points that satisfy both the first- and second-order necessary conditions for optimality.

All methods for solving (2.1) that use iterative linear solvers are based on the CG method. Toint [31] and Steihaug [30] independently proposed CG-based methods for solving the trust-region problem. The methods begin by applying the CG method to the Newton equations $HS = -g$ under the assumption that H is positive definite. The CG iterates $\{s_k\}$ have the form

$$s_0 = 0, \quad s_k = s_{k-1} + \alpha_{k-1} p_{k-1}, \quad k \geq 1,$$

where the CG directions $\{p_k\}$ satisfy the conjugacy conditions $p_\ell^T H p_m = 0$ for all $0 \leq \ell \leq k$ and $0 \leq m \leq k$ such that $\ell \neq m$. Each iterate s_k is such that

$$s_k = \operatorname{argmin}_{s \in \mathcal{S}_k} \{g^T s + \frac{1}{2} s^T H s\},$$

where $\mathcal{S}_k$ is a k -dimensional subspace spanned by the directions $p_0, p_1, \dots, p_{k-1}$. The set $\mathcal{S}_k$ is a member of a sequence of expanding subspaces $\{\mathcal{S}_k\}$ such that $\mathcal{S}_{k-1} \subset \mathcal{S}_k$.

If H is positive definite and the Newton step $-H^{-1}g$ lies inside the trust region, then the CG iterations are terminated with the iterate s_k such that $\|g + H s_k\| \leq \tau \|g\|$, where τ is a given positive tolerance. For small values of τ , this s_k approximates the unconstrained step $-H^{-1}g$.

Steihaug establishes the key property that if $p_\ell^T H p_\ell > 0$ for $0 \leq \ell \leq k-1$, then the norms of the CG iterates $\{s_k\}$ are strictly increasing in the two-norm. This

implies that there is no reason to continue computing CG iterates once they cross the trust-region boundary. In particular, if one of the conditions

$$(2.5) \quad p_{k-1}^T H p_{k-1} \leq 0 \quad \text{or} \quad \|s_{k-1} + \alpha_{k-1} p_{k-1}\| \geq \delta$$

holds, then the solution of (2.1) lies on the boundary of the trust region and the CG iterations are terminated. If one of the conditions (2.5) hold, Steihaug's method redefines the final iterate as $s_k = s_{k-1} + \gamma_{k-1} p_{k-1}$, where γ_{k-1} is a solution of the one-dimensional trust-region problem

$$\underset{\gamma}{\text{minimize}} \quad \mathcal{Q}(s_{k-1} + \gamma p_{k-1}) \quad \text{subject to} \quad \|s_{k-1} + \gamma p_{k-1}\| \leq \delta.$$

Toint uses a different choice of s_k if $p_{k-1}^T H p_{k-1} \leq 0$ and redefines s_k as the Cauchy step s_c , which solves the one-dimensional problem

$$(2.6) \quad \underset{s \in \mathbb{R}^n}{\text{minimize}} \quad \mathcal{Q}(s) \quad \text{subject to} \quad \|s\| \leq \delta, \quad s \in \text{span}\{g\}.$$

Either choice gives an approximate solution that is never worse than the Cauchy step. As a result, if the underlying trust-region method is endowed with an appropriate strategy for adjusting δ , then the iterates will exhibit first-order convergence on a bounded continuously differentiable function f (see Powell [22, 23]).

The Steihaug–Toint method terminates at the first boundary point, which implies that s_k may be a poor approximate solution of (2.1) in the constrained case. This lack of accuracy control was noted by Gould et al. [12], who proposed solving the constrained problem using the GLTR method. This method solves (2.1) on an expanding sequence of subspaces generated by the vectors $v_0, v_1, \dots, v_{k-1}$ associated with the Lanczos process for reducing H to tridiagonal form. The subspace minimization problem at the k th step is given by

$$(2.7) \quad \underset{y \in \mathbb{R}^k}{\text{minimize}} \quad g^T V_k y + \frac{1}{2} y^T T_k y \quad \text{subject to} \quad \|y\| \leq \delta,$$

where T_k is tridiagonal and V_k is the matrix of Lanczos vectors. The reduced problem (2.7) is solved using a variant of the Moré–Sorensen algorithm [19] that exploits the tridiagonal structure of the reduced Hessian T_k . Once an optimal y_k for the reduced problem has been found, the solution $s_k = V_k y_k$ is computed by repeating the Lanczos recurrence and regenerating the columns of V_k .

If the Lanczos process is always restarted when T_k is reducible, then, in theory, the GLTR method may be used to solve the trust-region problem to arbitrary accuracy. However, the need to regenerate V_k in the constrained case substantially increases the number of matrix-vector products. Another more serious difficulty is that rounding errors quickly lead to a loss of orthogonality of the Lanczos vectors. This loss of orthogonality implies that the solution of reduced problem (2.7) rapidly diverges from the required solution, which is based on the problem

$$\underset{y \in \mathbb{R}^k}{\text{minimize}} \quad g^T V_k y + \frac{1}{2} y^T V_k^T H V_k y \quad \text{subject to} \quad \|V_k y\| \leq \delta.$$

These considerations prompt Gould et al. to impose a modest limit on the number of Lanczos iterations in the constrained case. (When GLTR is used as part of a trust-region method for unconstrained optimization, the Steihaug point is accepted if it is within 90% of the best value found so far.)

In [15], Hager considers subspace minimization methods for finding an exact solution of the equality constrained problem:

$$(2.8) \quad \underset{s \in \mathbb{R}^n}{\text{minimize}} \quad \mathcal{Q}(s) = g^T s + \frac{1}{2} s^T H s \quad \text{subject to} \quad s^T s = \delta^2.$$

In contrast to the Steihaug–Toint and GLTR methods, which generate a sequence of *expanding* subspaces, Hager’s method relies on generating good quality low-dimensional subspaces. At the start of the k th iteration, values (s_{k-1}, σ_{k-1}) are known such that $s_{k-1}^T s_{k-1} = \delta^2$ and $\sigma_{k-1} \in (-\lambda_n, \infty)$ (cf. Theorem 2.2). The k th iterate (s_k, σ_k) is a solution of the subspace minimization problem

$$(2.9) \quad \underset{s \in \mathcal{S}_k}{\text{minimize}} \quad \mathcal{Q}(s) \quad \text{subject to} \quad s^T s = \delta^2, \quad s \in \mathcal{S}_k,$$

where $\mathcal{S}_k = \text{span}\{s_{k-1}, \nabla \mathcal{Q}(s_{k-1}), z_0, s_{\text{SQP}}\}$. The use of the previous iterate s_{k-1} in $\mathcal{S}_k$ guarantees that $\mathcal{Q}(s_k) < \mathcal{Q}(s_{k-1})$. The vector z_0 is the best estimate of the leftmost eigenvector computed as part of a startup phase that solves a reduced problem of dimension $\ell = \max\{10, n/100\}$. (The startup problem may be solved a number of times.) The vector s_{SQP} is computed from one step of Newton’s method applied to (2.8). As the Newton equations are not positive definite, Hager uses a projected method to ensure that the CG iterates are well defined. This method is equivalent to applying the CG method with constraint preconditioning (see, e.g., [11, 18]). These methods require that the initial iterate satisfies the constraint, which implies that the reduced problem (2.9) must be solved to high accuracy.

Hager and Park [16] show that any SSM method based on a subspace $\mathcal{S}_k$ containing the vectors s_{k-1} , $\nabla \mathcal{Q}(s_{k-1})$ and u_n is globally convergent to a solution of the trust-region problem. This result provides a justification of the composition of $\mathcal{S}_k$, but it does not constitute a convergence proof for the SSM method because u_n is unknown in general.

3. Phased SSM. The proposed method combines three basic components: (i) the Lanczos variant of the CG method for solving positive-definite linear equations [21]; (ii) a simple inexpensive method for estimating the leftmost eigenpair of H ; and (iii) a SSM method in which the low-dimensional subspace includes a regularized Newton accelerator direction. The method generates a sequence of subspace minimizers $\{s_k\}$ such that $\|s_k\| \leq \delta$ and $\mathcal{Q}(s_k) < \mathcal{Q}(s_{k-1})$. On termination, s_k satisfies $\mathcal{Q}(s_k) \leq \mathcal{Q}(s_c)$, where s_c is the Cauchy step defined in (2.6).

The phased-SSM method has two phases. Phase 1 is the Steihaug–Toint method with two enhancements. First, if $\|g\|$ is small (see section 4 for the precise condition), Phase 1 starts by approximating the leftmost eigenpair using a low-dimensional subspace minimization based on the Lanczos process. This extends the Steihaug–Toint method to the case where the starting point is close to a nonoptimal stationary point. Second, a low-dimensional subspace minimization is used to find a better exit point when the solution lies outside the trust region. This can give a substantially better estimate of the trust-region solution on the boundary.

Phase 2 is activated if Phase 1 gives a boundary exit point with insufficient accuracy. In Phase 2, a CG-based SSM method is used to solve the constrained problem over a sequence of evolving low-dimensional subspaces. Each subspace is spanned by three vectors: the current best approximate solution; an estimate of the leftmost eigenvector; and the regularized Newton accelerator direction.

The Lanczos process is the “driving mechanism” for both phases. The Lanczos vectors not only generate the conjugate directions for solving the positive-definite

equations of both phases, but also provide independent vectors for the definition of the evolving low-dimensional subspaces associated with the reduced versions of the trust-region and leftmost eigenvector problems. As these processes require a steady stream of Lanczos vectors within each phase, the Lanczos process is restarted with a random initial vector if the tridiagonal matrix is reducible (i.e., if an off-diagonal element β_k of T_k is zero).

To allow for the case $g = 0$ with H indefinite, a preassigned positive scalar tolerance τ_0 is used to define a “zero” vector g . If $\|g\| > \tau_0$, the Lanczos process is initialized with $v_0 = -g/\|g\|$. Otherwise, the vector g is assumed to be negligible, and v_0 is set to be a normalized random vector.

3.1. Phase 1 overview. In the first phase, standard Lanczos-CG iterates are generated until a sufficiently accurate solution of $Hz = -g$ is found inside the trust region or it becomes evident that the solution lies on the boundary. The principal cost of each CG iteration is the matrix-vector product associated with the Lanczos process. Embedded in the Lanczos-CG algorithm is the calculation of an estimate of a leftmost eigenpair. The estimate is computed by solving the reduced eigenproblem

$$(3.1) \quad \underset{z \in \mathbb{R}^n}{\text{minimize}} \quad z^T H z \quad \text{subject to} \quad \|z\| = 1, \quad z \in \mathcal{Z}_k,$$

where $\mathcal{Z}_k = \text{span}\{v_k, z_{k-1}\}$ with v_k the most recently computed Lanczos vector and z_{k-1} the leftmost eigenvector estimate from the previous CG iteration.

Given the matrix Z_k whose columns form a maximally linearly independent subset of $\{v_k, z_{k-1}\}$, the solution z_k of (3.1) may be written as $z_k = Z_k w_k$, where w_k minimizes $w^T Z_k^T H Z_k w$ subject to $w^T Z_k^T Z_k w = 1$. This problem has at most two variables and may be solved in closed form. Once z_k has been determined, the leftmost eigenvalue is estimated by the Rayleigh quotient $\zeta_k = z_k^T H z_k$. The inclusion of z_{k-1} in the reduced space $\mathcal{Z}_k$ ensures that the Rayleigh quotients decrease monotonically.

An estimate of the leftmost eigenpair is available at almost no additional cost. Apart from the calculation of $H z_0$, the estimate requires no additional matrix-vector products. To see this, note that the calculation of $Z_k^T H Z_k$ requires the vectors $H z_{k-1}$ and $H v_k$. The vector $H v_k$ is available as part of the two-term Lanczos recurrence, and $H z_{k-1}$ is available as part of the previous reduced eigenproblem. 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 $H v_k$ and $H z_{k-1}$. The calculation of the eigenpair is summarized in Algorithm 3.1 below.

ALGORITHM 3.1. $[z, \zeta, [Hz]] = \text{subspaceEig}(z, v, [Hz], [Hv])$

Define Z from a maximally linearly independent subset of v and z ;

Form $Z^T H Z$ and $Z^T Z$ from $z, v, [Hv]$, and $[Hz]$;

$w := \text{argmin} \{ z^T Z^T H Z z : z^T Z^T Z z = 1 \}$;

$z := Z w; \quad \zeta = z^T H z$;

$[Hz] := [H Z] w$;

In the context of the j th iteration of a method for unconstrained minimization, an initial vector z_0 is not used to start the first trust-region problem (i.e., for $j = 0$, the eigenproblem is only solved over the one-dimensional subspace $\text{span}\{v_0\}$). However, in subsequent iterations, z_0 is the final eigenvector estimate associated with the previous trust-region problem. Thus, the initial generalized eigenvalue problem is solved over the subspace $\mathcal{Z}_0 = \text{span}\{v_0, z_0\} = \text{span}\{-g, z_0\}$. As the unconstrained

solver converges, the sequence of Hessians $\{H_j\}$ converges, and z_0 should be a good estimate of the leftmost eigenvector for the current Hessian.

3.2. Phase 1 termination. In the first phase, Lanczos-CG iterates s_k and leftmost eigenpair $(z_k, \zeta_k) = (z_k, z_k^T H z_k)$ until one of several termination conditions are satisfied. Termination may occur at an interior or boundary point.

Termination inside the trust region. Let $\{\gamma_i\}_{i=0}^{k-1}$ and $\{\beta_i\}_{i=1}^{k-1}$ denote the diagonal and off-diagonal elements of the Lanczos-CG tridiagonal matrix T_k . Let β_k denote the coefficient of the newest Lanczos vector v_k . The Lanczos-CG iterates are terminated inside the trust region if the following conditions hold:

$$(3.2) \quad \begin{aligned} & (\|g\| > \tau_0 \quad \text{and} \quad (\|g + Hs_k\| \leq \tau_1 \|g\| \quad \text{or} \quad |\beta_k| \leq \max\{1, \gamma_{\max}\} \sqrt{\epsilon_M})) \\ & \text{or } (\|g\| \leq \tau_0 \quad \text{and} \quad \|\zeta_k z_k - H z_k\| \leq \tau_1 \|\zeta_0 z_0 - H z_0\|), \end{aligned}$$

where τ_0 and τ_1 are preassigned scalars and $\gamma_{\max}$ is the diagonal of T_k with largest magnitude, i.e., $\gamma_{\max} = \max_{0 \leq i \leq k-1} |\gamma_i|$. (See section 4 for more details concerning the definition of τ_0 and τ_1 in the context of a trust-region method for unconstrained minimization.) When $\|g\| > \tau_0$, the first condition of (3.2) ensures that the size of the final residual is sufficiently reduced relative to the initial residual g . The condition $|\beta_k| \leq \max\{1, \gamma_{\max}\} \sqrt{\epsilon_M}$ is used to detect the case where T_k is badly scaled and close to being reducible. If either of these tests is satisfied, the point s_k is considered to be an acceptable approximate solution of (2.1).

When $\|g\| \leq \tau_0$, the second condition of (3.2) is intended to provide an approximate leftmost eigenvector of H . If termination occurs with $\zeta_k > 0$, then it is likely that $s_k = 0$ is the solution of (2.1).

If the condition $|\beta_k| \leq \max\{1, \gamma_{\max}\} \sqrt{\epsilon_M}$ holds when $\|g\| \leq \tau_0$, the matrix T_k is assumed to be reducible, and the Lanczos process is restarted with a random vector.

Termination on the boundary. Phase 1 is terminated if any one of the following events occur:

- (i) Lanczos-CG generates an iterate s_k that lies outside of the trust region.
- (ii) Lanczos-CG computes a direction p_{k-1} such that $p_{k-1}^T H p_{k-1} \leq 0$.
- (iii) The Rayleigh quotient $\zeta_k = z_k^T H z_k$ is negative, where z_k is the estimate of the leftmost eigenvector.

The occurrence of any one of these events implies that the solution of (2.1) must lie on the constraint boundary. A final point on the boundary is defined by solving the trust-region subproblem over the subspace

$$\mathcal{S}_k = \text{span}\{s_{k-1}, p_{k-1}, z_k\},$$

where s_{k-1} is the last CG iterate inside the trust region, p_{k-1} is the last CG direction, and z_k is the approximation to the leftmost eigenvector. The reduced problem has the form

$$(3.3) \quad \underset{y}{\text{minimize}} \quad g^T P_k y + \frac{1}{2} y^T P_k^T H P_k y \quad \text{subject to} \quad \|P_k y\| \leq \delta,$$

where P_k is a matrix whose columns span $\mathcal{S}_k$. The QR decomposition with column interchanges may be used to determine a maximally linearly independent subset of the vectors $\{s_{k-1}, p_{k-1}, z_k\}$. The calculations associated with the solution of the reduced problem are given in Algorithm 3.2. As in Algorithm 3.1, the quantities $P_k^T H P_k$ and $P_k^T P_k$ may be formed with no additional matrix-vector products. On exit, the vectors s_k and Hs_k are defined in readiness for the start of Phase 2.

ALGORITHM 3.2. $[s, \sigma, [Hs]] := \text{subspaceSolve}(s, p, z, [Hs], [Hp], [Hz])$

Define P from a maximally linearly independent subset of s, p , and z ;

Form $P^T H P$, $P^T P$, and $P^T g$ from s, p , and $z, [Hs], [Hp]$, and $[Hz]$;

$y := \operatorname{argmin} \{ g^T P y + \frac{1}{2} y^T P^T H P y : y^T P^T P y = \delta \}$;

$s := P y; \quad [Hs] := [H P] y$;

The reduced problem has at most three dimensions and may be solved efficiently by a method that exploits direct matrix factorizations (the Moré–Sorensen algorithm was used to obtain the results of section 4).

3.3. Phase 2 overview. For Phase 2 to be initiated, the solution of the trust-region problem must lie on the boundary of the trust-region constraint, which implies that the Phase 2 iterations must minimize $\mathcal{Q}(s)$ subject to the equality constraint $\|s\| = \delta$. Without loss of generality, we consider the equivalent problem

$$(3.4) \quad \underset{s \in \mathbb{R}^n}{\text{minimize}} \quad \mathcal{Q}(s) = g^T s + \frac{1}{2} s^T H s \quad \text{subject to} \quad \frac{1}{2} \delta^2 - \frac{1}{2} s^T s = 0.$$

Phase 2 refines the solution on the boundary by solving

$$(3.5) \quad \underset{s \in \mathbb{R}^n}{\text{minimize}} \quad \mathcal{Q}(s) \quad \text{subject to} \quad \|s\| \leq \delta, \quad s \in \mathcal{S}_k = \operatorname{span}\{s_{k-1}, z_k, s_k^a\},$$

where s_{k-1} is the current best approximate solution, z_k is the current best estimate of the leftmost eigenvector of H , and s_k^a is a Newton “accelerator” direction defined below. The reduced problem has at most three dimensions and may be solved using Algorithm 3.2 with arguments $s = s_{k-1}$, $p = s_k^a$, and $z = z_k$.

3.4. Definition of the Newton accelerator direction. The accelerator direction s_k^a is defined as one step of a regularized Newton method applied to the equality constrained problem (3.4). Given a nonnegative scalar Lagrange multiplier σ , the Lagrangian function associated with (3.4) is

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

where $c(s)$ denotes the value of the constraint residual

$$(3.6) \quad c(s) = \frac{1}{2} s^T s - \frac{1}{2} \delta^2.$$

The gradient of the Lagrangian with respect to s and σ is given by

$$\nabla L(s, \sigma) = \begin{pmatrix} \nabla \mathcal{Q}(s) + \sigma s \\ \frac{1}{2} s^T s - \frac{1}{2} \delta^2 \end{pmatrix} = \begin{pmatrix} g + (H + \sigma I)s \\ c(s) \end{pmatrix}.$$

Similarly, the Hessian matrix of second derivatives is

$$\nabla^2 L(s, \sigma) = \begin{pmatrix} H + \sigma I & s \\ s^T & 0 \end{pmatrix}.$$

Optimal values of s and σ may be found by applying Newton’s method to find a zero of the function $F(s, \sigma) \triangleq \nabla L(s, \sigma)$. Given an estimate $w = (s, \sigma)$ of a zero, Newton’s method defines a new estimate $(s + p, \sigma + q)$, where $\Delta w = (p, q)$ is a solution of the

Newton equations $F'(w)\Delta w = -F(w)$. This system may be written in terms of σ and s as

$$(3.7) \quad \begin{pmatrix} H + \sigma I & s \\ s^T & 0 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = - \begin{pmatrix} g + (H + \sigma I)s \\ c(s) \end{pmatrix}.$$

The Newton equations are indefinite and cannot be solved directly using the Lanczos-CG method. Instead, we solve a related system that is positive semidefinite in the neighborhood of (s^*, σ^*) .

Given a positive scalar μ and a nonnegative scalar σ_e , consider the function of both s and σ given by

$$L_\mu(s, \sigma) = \mathcal{Q}(s) + \sigma_e c(s) + \frac{1}{2\mu} c(s)^2 + \frac{1}{2\mu} (\mu(\sigma - \sigma_e) - c(s))^2.$$

The gradient and Hessian of $L_\mu(s, \sigma)$ with respect to (s, σ) are

$$\nabla L_\mu(s, \sigma) = \begin{pmatrix} g + (H + \sigma I)s + 2(\hat{\sigma} - \sigma)s \\ \mu(\sigma - \sigma_e) - c(s) \end{pmatrix}$$

and

$$\nabla^2 L_\mu(s, \sigma) = \begin{pmatrix} H + \sigma I + 2(\hat{\sigma} - \sigma)I + \frac{2}{\mu} ss^T & -s \\ -s^T & \mu \end{pmatrix},$$

where $\hat{\sigma} = \hat{\sigma}(s) = \sigma_e + c(s)/\mu$.

THEOREM 3.1. *Let (s^*, σ^*) be a solution of (3.4); then there exists a $\bar{\mu}$ such that for all $\mu < \bar{\mu}$, the point (s^*, σ^*) minimizes the function*

$$\mathcal{Q}(s) + \sigma^* c(s) + \frac{1}{2\mu} c(s)^2 + \frac{1}{2\mu} (\mu(\sigma - \sigma^*) - c(s))^2.$$

This result suggests that, given a nonnegative σ_e such that $\sigma_e \approx \sigma^*$, we may obtain a better estimate of (s^*, σ^*) by minimizing

$$L_\mu(s, \sigma) = \mathcal{Q}(s) + \sigma_e c(s) + \frac{1}{2\mu} c(s)^2 + \frac{1}{2\mu} (\mu(\sigma - \sigma_e) - c(s))^2$$

with respect to both s and σ . Moreover, this result suggests that if $\sigma_e \approx \sigma^*$, μ does not need to be driven to zero as in conventional penalty methods. The Newton equations for minimizing $L_\mu(s, \sigma)$ are

$$\begin{pmatrix} H + (\sigma + 2(\hat{\sigma} - \sigma))I + \frac{2}{\mu} ss^T & -s \\ -s^T & \mu \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = - \begin{pmatrix} g + (H + (\sigma + 2(\hat{\sigma} - \sigma)I))s \\ \mu(\sigma - \sigma_e) - c(s) \end{pmatrix}$$

or, equivalently,

$$(3.8) \quad \begin{pmatrix} H + \bar{\sigma}I + \frac{2}{\mu} ss^T & -s \\ -s^T & \mu \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = - \begin{pmatrix} g + (H + \bar{\sigma}I)s \\ \mu(\sigma - \sigma_e) - c(s) \end{pmatrix},$$

where $\bar{\sigma} = \sigma + 2(\hat{\sigma} - \sigma) = 2\hat{\sigma} - \sigma$.

The Moré–Sorensen algorithm applied to the reduced problem provides estimates of both σ^* and s^* . This suggests that the optimal σ from the reduced problem (3.5) is a good choice for σ_e .

The linear system (3.8) has only one row and column more than the equations associated with the unconstrained case. The Lanczos-CG method may be used to compute an approximate Newton step. The accuracy of the accelerator step affects only the rate of convergence to the constrained solution and does not affect the convergence properties of the SSM method. It follows that it is not necessary to minimize $L_\mu(s, \sigma)$ to high accuracy. In practice we define s_k^a as an approximate solution of the first Newton system (3.8). In addition, for reasons of efficiency, the number of Lanczos-CG iterations used to find an approximate solution of (3.8) is limited. In the results of section 4 the iterations were limited to 10.

A benefit of using the Lanczos-CG method for solving (3.8) is that s need not satisfy $c(s) = 0$, i.e., s need not lie exactly on the boundary of the trust region.

The calculations associated with the definition of the Newton accelerator direction are given in Algorithm 3.3 below.

ALGORITHM 3.3. $[s, \sigma] := \textbf{NewtonAccelerator}(s, \sigma, \sigma_e, \tau_{\text{tol}})$
 Set $\hat{\sigma} = \sigma_e + c(s)/\mu$; $\bar{\sigma} = \sigma + 2(\hat{\sigma} - \sigma)$;
 Find the solution (p, q) of (3.8) using Lanczos-CG with tolerance τ_{tol} ;
 $\alpha_M = \textbf{if } q < 0 \textbf{ then } (\sigma + q - \sigma_e)/q \textbf{ else } +\infty$;
 $\alpha_M := \min\{1, \eta\alpha_M\}$;
 Compute α ($0 < \alpha \leq \alpha_M$) satisfying the Wolfe line-search conditions for $L_\mu(s, \sigma)$;
 $s := s + \alpha p$; $\sigma := \sigma + \alpha q$;

3.5. Phase 2 termination. Given a positive tolerance τ_2 , the Phase 2 iterations are terminated if $r_S \leq \tau_2 \|g\|$, where r_S is the residual associated with the current best estimate (s, σ_e) , i.e.,

$$(3.9) \quad r_S = \|g + (H + \sigma_e I)q\| + \sigma_e |c(s)|,$$

where $q = P\bar{q}$ with $\bar{q}$ the solution of the reduced system analogous to (2.3), i.e.,

$$(P^T H P + \sigma_e P^T P)\bar{q} = P^T g.$$

The condition (3.9) takes into account that the Moré–Sorensen algorithm gives only an *approximate* solution of the reduced problem. The reduced trust-region problem must be solved to an accuracy that is at least as good as that required for the full problem. Suitable values for the constants κ_1 and κ_2 of (2.4) are $\kappa_1 = \min\{10^{-1}\tau_2, 10^{-6}\}$ and $\kappa_2 = 0$.

Similarly, we define the error in the optimality conditions for the Newton accelerator (s_a, σ_a) (the approximate minimizer of L_μ). In this case, the residual is

$$(3.10) \quad r_A = \|g + (H + \sigma_a I)s_a\| + \sigma_a |c(s_a)|.$$

In practice, the residual associated with the accelerator is generally larger than the residual associated with the reduced-problem solution. To see this, consider the value of $|c(s)|$, the error in the “constraint” part of the optimality conditions. As the Newton system is not being solved accurately, the value of $|c(s)|$ at a typical Newton iterate may be large even when the solution lies on the boundary. By contrast, every solution of the reduced subproblem will have $|c(s)|$ of the order of the Moré–Sorensen tolerance κ_1 (cf. (2.4)).

3.6. Properties of the Newton accelerator. The method above may be regarded as a regularization of Newton's method. If both sides of the system (3.8) are multiplied by the nonsingular matrix $\begin{pmatrix} I & \frac{2}{\mu}s \\ 0 & 1 \end{pmatrix}$ and the last row is scaled by -1 , we obtain

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

If $c(s) \approx 0$ and $\sigma_e \approx \sigma$, these equations are a perturbation of the Newton equations (3.7). The following result shows that the perturbation μ serves as a *regularization* parameter in the degenerate case.

THEOREM 3.2 (regularization of the degenerate case). *Suppose that (s, σ) denotes a solution of the trust-region subproblem and 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 leftmost eigenvalue of H has algebraic multiplicity 1, then the augmented system matrix*

$$(3.11) \quad \begin{pmatrix} H + \sigma I + \frac{2}{\mu}ss^T & -s \\ -s^T & \mu \end{pmatrix}$$

is positive definite for any $\mu > 0$.

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

$$(3.12) \quad s = -(H - \lambda_n I)^\dagger g + \beta z,$$

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 (3.11):

$$(3.13) \quad \begin{pmatrix} H + \sigma I + \frac{2}{\mu}ss^T & -s \\ -s^T & \mu \end{pmatrix} = \begin{pmatrix} I & -\frac{1}{\mu}s \\ 0 & 1 \end{pmatrix} \begin{pmatrix} H - \lambda_n I + \frac{1}{\mu}ss^T & 0 \\ 0 & \mu \end{pmatrix} \begin{pmatrix} I & 0 \\ -\frac{1}{\mu}s^T & 1 \end{pmatrix}.$$

Assume that $H + \sigma I + \frac{2}{\mu}ss^T$ is not positive definite. Then there exists a nonzero p such that $p^T(H - \lambda_n I + \frac{1}{\mu}ss^T)p \leq 0$. 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 $H + \sigma I + \frac{2}{\mu}ss^T$ must be positive definite, and the result follows. $\square$

A safeguarded Newton method may be used to minimize $L_\mu(s, \sigma)$ with respect to both s and σ . Algorithm 3.4 is an approximate safeguarding scheme that attempts to ensure the system in (3.8) is positive definite based on the current values of σ_e , σ_ℓ , σ_a , $c(s_a) = \frac{1}{2}s_a^T s_a - \delta^2$, and the error in the optimality conditions. At all times, the safeguarding algorithm ensures that both σ_a and σ_e are greater than the current estimate σ_ℓ of $\max\{-\lambda_n, 0\}$. The algorithm adjusts σ_e so that the matrix $H + \bar{\sigma}I$ of (3.8) is positive definite. We emphasize that the subspace minimizer s_k is never overwritten, which guarantees monotonicity of the sequence $Q(s_k)$. Also, the Newton iterates (s_a, σ_a) are only overwritten with the subspace solve iterates if $\sigma_a < \sigma_\ell$ and $\sigma_e > \sigma_\ell$. In the event that both σ_a and σ_e are less than σ_ℓ , the leftmost eigenpair is used to update the iterates.

ALGORITHM 3.4. $[s_a, \sigma_e, \sigma_\ell] := \text{safeguard}(s_a, s, \sigma_e, \sigma_\ell)$
if $\sigma_a < \sigma_\ell$ **and** $\sigma_\ell < \sigma_e$ **then** $\sigma_a := \sigma_e$; $s_a := s$; **end**
 $\hat{\sigma} = \sigma_e + c(s_a)/\mu$; $\bar{\sigma} = \sigma_a + 2(\hat{\sigma} - \sigma_a)$;
if $\bar{\sigma} < \sigma_\ell$ **then**
 if $\sigma_a < \sigma_\ell$ **and** $\sigma_\ell < \sigma_e$ **then**
 $\sigma_a := \sigma_e$; $s_a := s$; $\hat{\sigma} = \sigma_e + c(s_a)/\mu$; $\bar{\sigma} = \sigma_a + 2(\hat{\sigma} - \sigma_a)$;
 if $\bar{\sigma} < \sigma_\ell$ **then** $\sigma_e := \sigma_a + |c(s_a)|/\mu$; **end**
 else if $\sigma_a > \sigma_\ell$ **and** $\sigma_e < \sigma_\ell$ **then**
 if $c(s_a) > 0$ **then** $\sigma_e := \sigma_a$; **else** $\sigma_e := \sigma_a + |c(s_a)|/\mu$; **end**
 else if $\sigma_a > \sigma_\ell$ **and** $\sigma_e > \sigma_\ell$ **then**
 $r_S = \|g + (H + \sigma_e I)q\| + \sigma_e |c(s)|$; $r_A = \|g + (H + \sigma_a I)s_a\| + \sigma_a |c(s_a)|$;
 if $r_S < r_A$ **then**
 $s_a := s$; $\sigma_a := \sigma_e$;
 $\hat{\sigma} = \sigma_e + c(s_a)/\mu$; $\bar{\sigma} = \sigma_a + 2(\hat{\sigma} - \sigma_a)$;
 if $\bar{\sigma} < \sigma_\ell$ **then** $\sigma_e := \sigma_a + |c(s_a)|/\mu$; **end**
 else
 $\sigma_e := \sigma_a$; $\hat{\sigma} = \sigma_e + c(s_a)/\mu$; $\bar{\sigma} = \sigma_a + 2(\hat{\sigma} - \sigma_a)$;
 if $\bar{\sigma} < \sigma_\ell$ **then** $\sigma_e := \sigma_a + |c(s_a)|/\mu$; **end**
 end
 else
 $\sigma_a := |\zeta|$; $\sigma_e := |\zeta|$; $s_a := \delta \times z$;
 $\hat{\sigma} = \sigma_e + c(s_a)/\mu$; $\bar{\sigma} = \sigma_a + 2(\hat{\sigma} - \sigma_a)$;
 if $\bar{\sigma} < \sigma_\ell$ **then** $\sigma_e := \sigma_a + |c(s_a)|/\mu$; **end**
 end
end
end

Provided $\sigma \in (-\lambda_n, \infty)$, then with safeguarding, the system (3.8) is positive definite and may be solved using CG. However, the next result shows that if a direction of negative curvature is generated by CG, then it may be used to provide a direction of negative curvature for $H + \sigma I$, which may be used in turn to safeguard σ and update an estimate of the leftmost eigenpair.

THEOREM 3.3. *Assume that p is a direction of negative curvature for the matrix*

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

where μ is a positive scalar. Then the vector of first n elements of p is a direction of negative curvature for $H + \sigma I$.

Proof. The result follows trivially from the identity

$$B = \begin{pmatrix} H + \sigma I & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} I & -\frac{1}{\mu}s \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{\mu}ss^T & 0 \\ 0 & \mu \end{pmatrix} \begin{pmatrix} I & 0 \\ -\frac{1}{\mu}s^T & 1 \end{pmatrix}$$

and the fact that the second term in the matrix sum is positive semidefinite. $\square$

In Phase 2, the regularization parameter μ is initially defined as $\mu = 10^{-2}$ and reduced each Phase 2 iteration (before safeguarding) by a factor of $1/3$ if a direction of negative curvature for (3.11) is encountered while computing the Newton accelerator direction.

4. Numerical results. The Steihaug–Toint and phased-SSM methods were implemented and run in MATLAB. Numerical results are given for unconstrained prob-

lems from the CUTer test collection (see Bongartz et al. [1] and Gould, Orban, and Toint [13]). 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 locate the twice continuously differentiable unconstrained problems for which the number of variables in the data file can be varied. The final test set consisted of 49 problems. For all problems, the dimension was $n = 1000$ unless otherwise recommended in the CUTer documentation. In all cases, $n \geq 1000$. A combination line-search trust-region method was used to define the update to the trust-region radius.

The trust-region method is considered to have solved a CUTer problem successfully when a trust-region iterate x_j satisfies

$$(4.1) \quad \|g(x_j)\| \leq \max\{\epsilon\|g(x_0)\|, \epsilon|f(x_0)|, \sqrt{\epsilon_M}\},$$

where $\epsilon = 10^{-6}$ and ϵ_M denotes machine precision. If x_0 is a nonoptimal stationary point, the presence of the term $f(x_0)$ prevents the trust-region algorithm from terminating at x_0 . If a solution is not found within $2n$ iterations, the iterations are terminated, and the algorithm is considered to have failed. Throughout this section we refer to s_j as the approximate solution of the j th trust-region problem.

4.1. Termination of Phase 1 and Lanczos-CG. The principal termination condition for the Steihaug–Toint method and Phase 1 of the phased-SSM method is based on the Dembo–Eisenstat–Steihaug criterion [5]. In particular, if s_j denotes the approximate solution of the j th trust-region problem, then the Lanczos-CG method terminates successfully with a point s_j inside the trust region if

$$(4.2) \quad \|g_j + H_j s_j\| \leq \tau_{1j} \|g_j\|, \quad \text{where} \quad \tau_{1j} = \min\{10^{-1}, \|g_j\|^{0.1}\}.$$

This condition forces a relative decrease in the residual comparable to that required by Gould et al. [12].

4.2. Termination of Phase 2. A test for Phase 2 convergence is made immediately after the subspace minimization. A user-specified parameter ϵ_s ($0 < \epsilon_s \leq 1$) is used to specify the accuracy of the trust-region solution in the constrained case. The parameter τ_2 for the j th step is

$$(4.3) \quad \tau_{2j} = \frac{1}{\epsilon_s} \min\{10^{-1}, \|g_j\|^{0.1}\}.$$

The value $\epsilon_s = 1$ corresponds to solving the constrained problem to the same accuracy as the unconstrained problem. The value $\epsilon_s \approx \epsilon_M$ corresponds to accepting the Steihaug point as the Phase 2 solution.

The iteration limit imposed in Phase 2 is smaller than that imposed on Phase 1. If Phase 2 is not converging well, this usually implies that the estimate of the leftmost eigenpair is poor. In this case, it is sensible to terminate the solution of the subproblem. In all the runs reported here, a limit of 10 iterations was enforced during the second phase. In all runs, a limit of 50 Lanczos vectors was imposed for the calculation of the Newton accelerator direction. If this iteration limit is reached, the Lanczos-CG iterate with the smallest residual is returned as the accelerator direction.

4.3. The trust-region algorithm. The approximate solution s_j of the j th 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 using a line search based on Gertz's "biased" Wolfe line search (see Gertz [9]). In Algorithm 4.1 below,

$$(4.4) \quad \mathcal{Q}_j^-(s) = g_j^T s + \frac{1}{2} [s^T H_j s]_-,$$

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

$$(4.5) \quad \frac{f(x_j + s_j(\alpha_j)) - f(x_j)}{\mathcal{Q}_j^-(s_j(\alpha_j))} > \eta_1,$$

where η_1 is a preassigned scalar such that $0 < \eta_1 < \frac{1}{2}$. The line-search parameters used for the experiments were $\eta_1 = 10^{-4}$, $\eta_2 = 0.25$, $\omega = 0.9$, and $\gamma_3 = 1.5$.

ALGORITHM 4.1. Combination line-search/trust-region method.

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 $\min \{ \mathcal{Q}_j(s) : \|s\| \leq \delta_j \}$;

Find α_j satisfying the Wolfe conditions:

$$f(x_j + \alpha_j s_j) \leq f(x_j) + \eta_1 \mathcal{Q}_j^-(\alpha_j s_j) \text{ and } |g(x_j + \alpha_j s_j)^T s_j| \leq -\omega \mathcal{Q}_j^{-'}(\alpha_j s_j);$$

$$x_{j+1} := x_j + \alpha_j s_j;$$

if $(f(x_{j+1}) - f(x_j)) / \mathcal{Q}_j^-(s_j) \geq \eta_2$ **then**

if $\|s_j\| = \delta_j$ **and** $\alpha_j = 1$ **then**

$$\delta_{j+1} := \gamma_3 \delta_j;$$

else if $\|s_j\| < \delta_j$ **and** $\alpha_j = 1$ **then**

$$\delta_{j+1} := \max\{\delta_j, \gamma_3 \|s_j\|\};$$

else

$$\delta_{j+1} := \alpha_j \|s_j\|;$$

end if

else

$$\delta_{j+1} := \min\{\alpha_j \|s_j\|, \alpha_j \delta_j\};$$

end if

$$j := j + 1;$$

end do

A key feature of the combination line-search trust-region method is that the trust-region radius is updated as a function of α_j . The term "biased" is used by Gertz to refer to a deliberate bias against reducing the trust-region radius when α_j is small. Algorithm 4.1 above differs from Gertz's line search in that it is possible for the trust-region radius to be reduced even when α_j is small. Nevertheless, Algorithm 4.1 still retains a natural bias against decreasing the trust-region radius; in particular, the trust-region radius is not decreased if $\|s_j\| < \delta_j$ and $\alpha_j = 1$.

Tables 1–2 give the results of applying the Steihaug–Toint method and the phased-SSM method with $\epsilon_s = 1$ on the 49 problems from the CUTer test set. For each solver, the columns give the total number of function evaluation ("Fe"), the total number of matrix-vector products ("Prods"), and the final values of f and $\|g\|$. The final values are listed to help identify local solutions and to identify cases where the converged

TABLE 1
Steihaug and phased-SSM on CUTEr problems a–e.

Problem	Steihaug				Phased-SSM ($\epsilon_s = 1$)			
	Fe	Prods	$f(x)$	$\ g(x)\ $	Fe	Prods	$f(x)$	$\ g(x)\ $
<i>arwhead</i>	6	6	1.69e-10	6.37e-05	6	10	1.69e-10	6.37e-05
<i>bdqrtic</i>	14	41	3.98e+03	7.95e-02	13	44	3.98e+03	1.66e-01
<i>broydn7d</i>	139	404	3.75e+02	3.70e-04	65	1536	3.45e+02	3.44e-04
<i>brybnd</i>	12	46	6.65e-07	5.76e-03	12	57	3.93e-07	4.56e-03
<i>chainwoo</i>	27	57	1.31e+01	2.08e+00	25	105	3.93e+03	6.96e-01
<i>cosine</i>	12	10	-9.99e+02	3.01e-04	12	17	-9.99e+02	3.32e-04
<i>cragglvy</i>	14	36	3.36e+02	3.60e-01	14	48	3.36e+02	3.17e-01
<i>dixmaana</i>	13	11	1.00e+00	1.43e-03	13	21	1.00e+00	2.27e-03
<i>dixmaanb</i>	13	11	1.00e+00	1.10e-03	13	21	1.00e+00	1.10e-03
<i>dixmaanc</i>	13	11	1.00e+00	1.74e-02	13	21	1.00e+00	1.78e-02
<i>dixmaand</i>	14	12	1.00e+00	2.63e-02	14	23	1.00e+00	2.63e-02
<i>dixmaane</i>	14	93	1.00e+00	2.72e-03	14	93	1.00e+00	4.48e-03
<i>dixmaanf</i>	15	30	1.00e+00	9.92e-03	15	42	1.00e+00	9.95e-03
<i>dixmaang</i>	15	24	1.01e+00	2.43e-02	15	36	1.01e+00	2.43e-02
<i>dixmaanb</i>	15	19	1.03e+00	7.15e-02	15	31	1.03e+00	7.17e-02
<i>dixmaanb</i>	16	40	1.00e+00	5.71e-03	16	53	1.00e+00	5.72e-03
<i>dixmaank</i>	16	30	1.00e+00	1.41e-02	16	43	1.00e+00	1.41e-02
<i>dixmaanl</i>	16	25	1.01e+00	3.24e-02	16	38	1.01e+00	3.25e-02
<i>dqdrtic</i>	14	11	1.90e-03	1.21e+00	14	20	3.46e-05	1.19e-01
<i>dqrtic</i>	27	20	1.63e+11	1.98e+08	28	39	5.52e+10	8.83e+07
<i>edensch</i>	15	25	2.19e+02	4.36e-02	15	37	2.19e+02	4.41e-02
<i>eg2</i>	4	3	-9.99e+02	5.96e-09	4	5	-9.99e+02	5.96e-09
<i>engval1</i>	14	17	1.11e+03	2.63e-02	14	27	1.11e+03	2.50e-02
<i>extrosh</i>	31	70	2.24e-02	2.46e-01	29	75	4.42e-02	1.50e-01

gradient does not correspond to a local minimizer. (For problems with large $\|g_0\|$, requiring g_j to satisfy a small absolute tolerance is unreasonable.)

Tables 1–2 show that Steihaug’s method and the phased-SSM method behave very similarly on many problems. These problems correspond to situations when the approximate solution of every subproblem is in the interior of the trust region. In these cases, the phased-SSM method never enters the second phase and is as efficient as Steihaug’s method. (Note that the extra matrix-vector products are associated with computing the initial estimate of the leftmost eigenvector for each Hessian.)

We would expect the Steihaug–Toint method not to perform well in terms of the number of function evaluations when solutions of the trust-region subproblem frequently occur on the boundary. In these cases, the number of function evaluations required by the methods are sometimes significantly different, (e.g., see *broydn7*, *genrose*, *fminsurf*, or *fminsrf2*). On a few problems, the performance of phased-SSM was slightly inferior to that of Steihaug’s method. And, in one case (problem *ncb20*), phased-SSM performed significantly worse. The superiority of Steihaug’s method in these cases appears to be the effect of good fortune rather than a consistently better subproblem solution.

As noted by Gould et al. [12], it is sometimes better not to solve the subproblem to high accuracy when the solution lies on the boundary. This may be especially true when the trust-region iterates are far from a minimizer of f . Tables 3–4 give results for

TABLE 2
Steihaug and phased-SSM on CUTEr problems f–z.

Problem	Steihaug				Phased-SSM ($\epsilon_s = 1$)			
	Fe	Prods	$f(x)$	$\ g(x)\ $	Fe	Prods	$f(x)$	$\ g(x)\ $
<i>fminsrf2</i>	363	1515	1.00e+00	2.47e-05	59	1974	1.00e+00	4.48e-06
<i>fminsurf</i>	334	750	1.00e+00	2.47e-06	60	1270	1.00e+00	3.31e-06
<i>freuroth</i>	16	19	1.21e+05	5.41e-02	16	29	1.21e+05	5.41e-02
<i>genrose</i>	1218	5940	1.00e+00	3.04e-03	822	19592	1.00e+00	5.69e-04
<i>liarwhd</i>	19	27	4.03e-07	1.28e-03	19	41	1.51e-07	7.83e-04
<i>ncb20</i>	61	452	9.10e+02	2.92e-04	94	2674	9.19e+02	1.07e-03
<i>ncb20b</i>	10	61	1.68e+03	9.99e-04	9	79	1.68e+03	1.96e-04
<i>noncvxu2</i>	36	26	1.15e+06	2.21e+03	46	242	1.49e+06	2.05e+03
<i>noncvxun</i>	37	28	6.60e+05	2.19e+03	49	322	6.53e+05	2.01e+03
<i>nondia</i>	4	3	6.27e-03	9.86e-02	4	5	6.27e-03	9.86e-02
<i>nondquar</i>	23	115	5.52e-04	3.84e-03	18	151	5.26e-04	3.82e-03
<i>penalty1</i>	28	17	3.01e+13	5.14e+10	28	33	3.01e+13	5.14e+10
<i>penalty2</i>	2	1	1.45e+83	4.94e+38	2	1	1.45e+83	4.94e+38
<i>powellsq</i>	16	40	4.63e-03	3.54e-02	16	55	1.78e-03	1.73e-02
<i>power</i>	15	33	3.56e+04	1.25e+05	15	46	3.63e+04	1.28e+05
<i>quartc</i>	27	20	1.63e+11	1.98e+08	28	39	5.52e+10	8.83e+07
<i>schmvett</i>	9	37	-2.99e+03	6.79e-04	8	36	-2.99e+03	1.01e-03
<i>sparsqur</i>	14	23	4.24e-03	7.20e-02	14	47	4.27e-03	7.71e-02
<i>spmsrtls</i>	18	126	2.83e-09	9.33e-05	29	564	4.98e-08	5.61e-04
<i>srosenbr</i>	9	10	1.61e-09	3.58e-05	9	17	2.25e-09	4.24e-05
<i>testquad</i>	15	168	1.87e+01	1.09e+02	13	122	7.48e+01	2.44e+02
<i>tointgss</i>	15	13	1.00e+01	4.68e-03	15	22	1.00e+01	2.22e-03
<i>vardim</i>	13	12	6.87e+08	2.78e+10	13	23	6.87e+08	2.78e+10
<i>vareigul</i>	14	26	3.54e-04	1.54e-02	14	37	3.54e-04	1.53e-02
<i>woods</i>	12	14	1.97e+03	2.49e+00	13	27	1.97e+03	2.66e-01

different values of the tolerance ϵ_s in Phase 2 (i.e., when the subproblem solution lies on the boundary). In particular, the table gives the number of function evaluations and matrix-vector products required by phased-SSM for several values of ϵ_s in (3.9). (The recommended value is $\epsilon_s = 1$.) The value $\epsilon_s = \epsilon_M$ has the effect of forcing phased-SSM to terminate before entering Phase 2. In this case, the results indicate that phased-SSM gives a significant reduction in the number of function evaluations for little or no sacrifice in computation time.

The results highlight the tradeoff between the accuracy of the subproblem solutions and the computational effort. Table 5 compares Steihaug's method and phased-SSM for various values of ϵ_s . Generally speaking, as $\epsilon_s \rightarrow 1$, the required number of function evaluations for the test set decreases and the number of matrix-vector products increases. Depending on the application and cost of a matrix-vector product relative to the cost of a function evaluation, a less stringent stopping criteria (e.g., $\epsilon_s \ll 1$) may result in a more efficient algorithm.

The unconstrained examples in the CUTEr set are dominated by functions that are relatively inexpensive to evaluate. Many involve taking a prototype low-dimensional problem and extending it to an arbitrary number of dimensions. For such problems, the overall solution time is directly proportional to the total number of matrix-vector products. Ironically, this implies that the CUTEr test set is a class of problems for

TABLE 3
Inexact phased-SSM on CUTEr problems a–e.

ϵ_s	ϵ_M		0.05		0.1		0.5		1.0	
Problem	Fe	Prods	Fe	Prods	Fe	Prods	Fe	Prods	Fe	Prods
<i>arwhead</i>	6	10	6	10	6	10	6	10	6	10
<i>bdqrtic</i>	13	44	13	44	13	44	13	44	13	44
<i>broydn7d</i>	132	482	74	2409	68	2026	68	1768	65	1536
<i>brybnd</i>	12	57	12	57	12	57	12	57	12	57
<i>chainwoo</i>	23	107	25	109	25	107	25	105	25	105
<i>cosine</i>	12	17	12	17	12	17	12	17	12	17
<i>cragglvy</i>	14	48	14	48	14	48	14	48	14	48
<i>dixmaana</i>	13	21	13	21	13	21	13	21	13	21
<i>dixmaanb</i>	13	21	13	21	13	21	13	21	13	21
<i>dixmaanc</i>	13	21	13	21	13	21	13	21	13	21
<i>dixmaand</i>	14	23	14	23	14	23	14	23	14	23
<i>dixmaane</i>	14	93	14	93	14	93	14	93	14	93
<i>dixmaanf</i>	15	42	15	42	15	42	15	42	15	42
<i>dixmaang</i>	15	36	15	36	15	36	15	36	15	36
<i>dixmaanh</i>	15	31	15	31	15	31	15	31	15	31
<i>dixmaanj</i>	16	53	16	53	16	53	16	53	16	53
<i>dixmaank</i>	16	43	16	43	16	43	16	43	16	43
<i>dixmaanl</i>	16	38	16	38	16	38	16	38	16	38
<i>dqdrtic</i>	14	20	14	20	14	20	14	20	14	20
<i>dqrtic</i>	28	36	28	48	28	39	28	39	28	39
<i>edensch</i>	15	37	15	37	15	37	15	37	15	37
<i>eg2</i>	4	5	4	5	4	5	4	5	4	5
<i>engval1</i>	14	27	14	27	14	27	14	27	14	27
<i>extrosnb</i>	29	73	29	78	29	75	29	75	29	75

which *low-accuracy* trust-region solutions give the best run times (and hence are best solved using low-accuracy approximate trust-region solutions). Nevertheless, the CUTEr problems still constitute a suitable test of whether or not the number of functions decreases as the accuracy of the trust-region solution is increased. Table 5 indicates that requesting more accurate solutions is of more benefit on problems for which the objective function is some complex composite function (e.g., problems derived from solving a constrained problem by a sequence of constrained problems) or an objective function arising in simulation-based applications. We believe that it is important to be able to tackle such problems when they arise.

The results of Tables 1–2 and 3–4 are summarized in Table 5. In general, the phased-SSM method required between 24% and 35% fewer function evaluations than Steihaug’s method. By comparison, Gould et al. [12] report that GLTR solved 16 of 17 problems, and, for those solved by both GLTR and Steihaug’s method, GLTR obtained 12.5% fewer function evaluations than Steihaug’s method.

Table 6 summarizes the results of using different trust-region algorithms. The column with heading “Steihaug-basic” gives the results obtained using Steihaug’s method in conjunction with a “standard” trust-region algorithm (see, e.g., Conn, Gould and Toint [4]). The other columns give results obtained using the recommended “biased” line search (Algorithm 4.1) for several values of ϵ_s . The improvement in function evaluations (evals) is calculated based on the improvement compared to those of “Steihaug-basic.”

TABLE 4
Inexact phased-SSM on CUTEr problems f–z.

ϵ_s	ϵ_M		0.05		0.1		0.5		1.0	
Problem	Fe	Prods	Fe	Prods	Fe	Prods	Fe	Prods	Fe	Prods
<i>fminsrf2</i>	134	1354	73	2463	51	1929	76	2706	59	1974
<i>fminsurf</i>	104	494	75	2167	62	2000	60	1514	60	1270
<i>freuroth</i>	16	29	16	29	16	29	16	29	16	29
<i>genrose</i>	996	5785	814	31869	819	28690	816	22050	822	19592
<i>liarwhd</i>	19	41	19	41	19	41	19	41	19	41
<i>ncb20</i>	83	633	100	3922	106	4181	104	3181	94	2674
<i>ncb20b</i>	11	64	10	83	8	65	8	53	9	79
<i>noncvxu2</i>	39	54	44	340	49	457	44	286	46	242
<i>noncvxun</i>	39	54	47	360	48	374	50	324	49	322
<i>nondia</i>	4	5	4	5	4	5	4	5	4	5
<i>nondquar</i>	25	121	21	480	18	285	18	205	18	151
<i>penalty1</i>	28	33	28	33	28	33	28	33	28	33
<i>penalty2</i>	2	1	2	1	2	1	2	1	2	1
<i>powellsg</i>	16	55	16	55	16	55	16	55	16	55
<i>power</i>	15	46	15	46	15	46	15	46	15	46
<i>quartc</i>	28	36	28	48	28	39	28	39	28	39
<i>schmvett</i>	8	36	8	36	8	36	8	36	8	36
<i>sparsqr</i>	14	47	14	47	14	47	14	47	14	47
<i>spmsrtls</i>	17	128	28	835	28	782	23	366	29	564
<i>srosenbr</i>	9	17	9	17	9	17	9	17	9	17
<i>testquad</i>	15	183	13	125	13	122	13	122	13	122
<i>tointgss</i>	15	22	15	22	15	22	15	22	15	22
<i>vardim</i>	13	23	13	23	13	23	13	23	13	23
<i>vareigvl</i>	14	37	14	37	14	37	14	37	14	37
<i>woods</i>	13	27	13	27	13	27	13	27	13	27

TABLE 5
Comparison of methods. $\delta_0 = 1$.

	Steihaug	$\epsilon_s = \epsilon_M$	$\epsilon_s = 0.05$	$\epsilon_s = 0.1$	$\epsilon_s = 0.5$	$\epsilon_s = 1$
Problems solved	49	49	49	49	49	49
Function evals (fe)	2817	2153	1859	1830	1840	1824
Matrix mults (prods)	10528	10710	46442	42277	33939	29890
Improvement in fe	—	24%	34%	35%	35%	35%

TABLE 6
Comparison of methods and line searches. $\delta_0 = 1$.

	Steihaug-basic	Steihaug-biased	$\epsilon_s = \epsilon_M$	$\epsilon_s = 1$
Problems solved	49	49	49	49
Function evals (fe)	3180	2817	2153	1824
Matrix mults (prods)	31060	10528	10710	29890
Improvement in fe	—	11%	32%	43%

We summarize results from Tables 1–2 and 3–4 in Figures 1 and 2, respectively, using performance profiles (in $\log_2$ scale) proposed by Dolan and Moré [6]. Figure 1

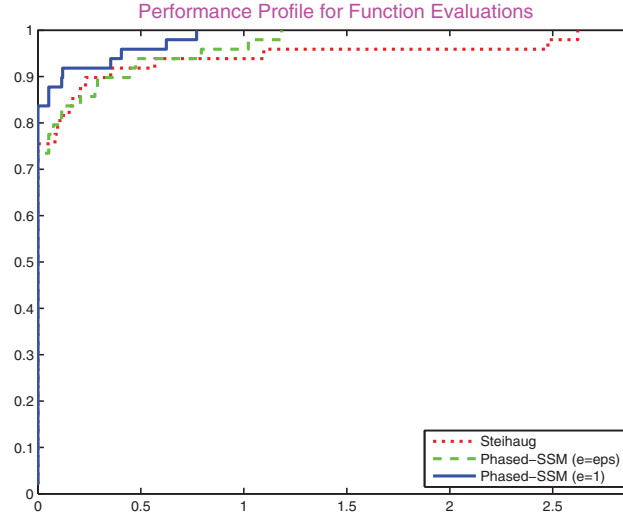


FIG. 1. $\log_2$ -scale performance profile comparing function evaluations on 49 CUTEr test problems.

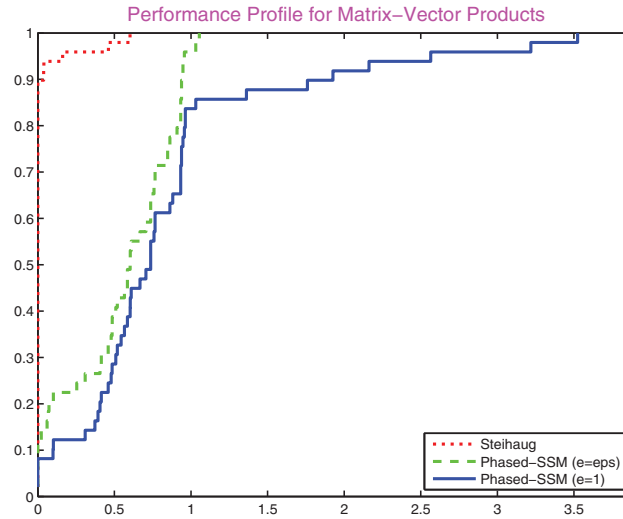


FIG. 2. $\log_2$ -scale performance profile comparing matrix-vector products on 49 CUTEr test problems.

plots the function $\pi_s : [0, r_M] \rightarrow \mathbb{R}^+$ defined by

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

where $\mathcal{P}$ denotes the set of test problems and $r_{p,s}$ denotes the ratio of the number of function evaluations needed to solve problem p with method s with the least number of function evaluations needed to solve problem p . Here r_M denotes the maximum value of $\log_2(r_{p,s})$. Figure 2 gives an equivalent plot in terms of matrix-vector products.

In order for phased-SSM to start the j th problem, it is necessary to form the product $H z_0$ for the current H and the best leftmost estimate (“ z_0 ”) from the previous

subproblem. This implies that every subproblem—even those whose solution lies in the interior of the trust region—costs at least one more matrix-vector product than that of Steihaug’s method. Nevertheless, the results of Table 5 indicate that Steihaug’s method and phased-SSM with $\epsilon_s \approx \epsilon_M$ require comparable numbers of matrix-vector products—a fact that is obscured by the performance profile.

5. Concluding remarks. Phased-SSM is a large-scale trust-region solver that allows the accuracy of the trust-region subproblem to be specified. This feature provides the ability to exploit the tradeoff between the overall number of function evaluations and matrix-vector products.

The numerical results of section 4 indicate that if a low-accuracy solution is requested, phased-SSM requires fewer function evaluations than Steihaug’s method with a negligible increase in matrix-vector products. Moreover, solving the subproblem more accurately results in a further decrease in the overall number of function evaluations at the cost of increasing the number of matrix-vector products. Based on the results, it appears that phased-SSM would be particularly effective in applications where the cost of a function evaluation is expensive relative to the cost of a matrix-vector product (e.g., in simulation-based applications).

An appropriate choice for the accuracy parameter is very problem dependent (see, e.g., Gould et al. [12]). Our results of section 4 indicate that it is possible to solve the trust-region problem to less accuracy in the constrained case without detracting from the efficiency of the method.

Acknowledgments. The authors are 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. Software, 21 (1995), pp. 123–160.
- [2] M. A. BRANCH, T. F. COLEMAN, AND Y. LI, *A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems*, SIAM J. Sci. Comput., 21 (1999), pp. 1–23.
- [3] R. H. BYRD, R. B. SCHNABEL, AND G. A. SHULTZ, *Approximate solution of the trust region problem by minimization over two-dimensional subspaces*, Math. Program., 40 (1988), pp. 247–263.
- [4] A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *Trust-Region Methods*, SIAM, Philadelphia, 2000.
- [5] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.
- [6] E. D. DOLAN AND J. J. MORÉ, *Benchmarking Optimization Software with COPS*, Technical memorandum ANL/MCS-TM-246, Argonne National Laboratory, Argonne, IL, 2000.
- [7] J. B. ERWAY, *Iterative Methods for Large-Scale Unconstrained Optimization*, Ph.D. thesis, Department of Mathematics, University of California, San Diego, 2006.
- [8] D. M. GAY, *Computing optimal locally constrained steps*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 186–197.
- [9] E. M. GERTZ, *Combination Trust-Region Line-Search Methods for Unconstrained Optimization*, Ph.D. thesis, Department of Mathematics, University of California, San Diego, 1999.
- [10] E. M. GERTZ, *A quasi-Newton trust-region method*, Math. Program., 100 (2004), pp. 447–470.
- [11] N. I. M. GOULD, M. E. HRIBAR, AND J. NOCEDAL, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395.
- [12] 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.
- [13] N. I. M. GOULD, D. ORBAN, AND P. L. TOINT, *CUTEr and SifDec: A constrained and unconstrained testing environment, revisited*, ACM Trans. Math. Software, 29 (2003), pp. 373–394.

- [14] J. D. GRIFFIN, *Interior-point Methods for Large-scale Nonconvex Optimization*, Ph.D. thesis, Department of Mathematics, University of California, San Diego, 2005.
- [15] W. W. HAGER, *Minimizing a quadratic over a sphere*, SIAM J. Optim., 12 (2001), pp. 188–208.
- [16] W. W. HAGER AND S. PARK, *Global convergence of SSM for minimizing a quadratic over a sphere*, Math. Comp., 74 (2004), pp. 1413–1423.
- [17] M. D. HEBDEN, *An Algorithm for Minimization using Exact Second Derivatives*, Technical report T.P. 515, Atomic Energy Research Establishment, Harwell, England, 1973.
- [18] C. KELLER, N. I. M. GOULD, AND A. J. WATHEN, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1300–1317.
- [19] J. J. MORÉ AND D. C. SORESENSEN, *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 553–572.
- [20] J. J. MORÉ AND D. C. SORESENSEN, *Newton's method*, in Studies in Numerical Analysis, G. Golub, ed., Math. Assoc. America, Washington, DC, 1984, pp. 29–82.
- [21] C. C. PAIGE AND M. A. SAUNDERS, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.
- [22] M. J. D. POWELL, *A new algorithm for unconstrained optimization*, in Nonlinear Programming (Proc. Sympos., Univ. of Wisconsin, Madison, Wis., 1970), J. B. Rosen, O. L. Mangasarian, and K. Ritter, eds., Academic Press, New York, 1970, pp. 31–65.
- [23] M. J. D. POWELL, *Convergence properties of a class of minimization algorithms*, in Nonlinear Programming, 2 (Proc. Sympos. Special Interest Group on Math. Programming, Univ. Wisconsin, Madison, Wis., 1974), O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, eds., Academic Press, New York, 1974, pp. 1–27.
- [24] F. RENDL AND H. WOLKOWICZ, *A semidefinite framework for trust region subproblems with applications to large scale minimization*, Math. Program., 77 (1997), pp. 273–299.
- [25] M. ROJAS, S. A. SANTOS, AND D. C. SORESENSEN, *A new matrix-free algorithm for the large-scale trust-region subproblem*, SIAM J. Optim., 11 (2001), pp. 611–646.
- [26] M. ROJAS, S. A. SANTOS, AND D. C. SORESENSEN, *Algorithm 873: LSTRS: MATLAB software for large-scale trust-region subproblems and regularization*, ACM Trans. Math. Software, 34 (2008), pp. 11:1–11:28.
- [27] M. ROJAS AND D. C. SORESENSEN, *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.
- [28] D. C. SORESENSEN, *Newton's method with a model trust region modification*, SIAM J. Numer. Anal., 19 (1982), pp. 409–426.
- [29] D. C. SORESENSEN, *Minimization of a large-scale quadratic function subject to a spherical constraint*, SIAM J. Optim., 7 (1997), pp. 141–161.
- [30] T. STEihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.
- [31] P. L. TOINT, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, I. S. Duff, ed., Academic Press, London and New York, 1981, pp. 57–88.