Les Cahiers du GERAD ISSN: 0711–2440

The conjugate residual method in linesearch and trust-region methods

M.-A. Dahito,
D. Orban

G–2018–50

July 2018

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

The conjugate residual method in linesearch and trust-region methods

Marie-Ange Dahito
Dominique Orban

GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7

marie-ange.dahito@polymtl.ca
dominique.orban@gerad.ca

July 2018
Les Cahiers du GERAD
G–2018–50

Abstract: The minimum residual method (MINRES) of Paige and Saunders (1975), which is often the method of choice for symmetric linear systems, is a generalization of the method of conjugate residuals (CR), proposed by Hestenes and Stiefel (1952). Like the conjugate gradient method (CG), CR possesses properties that are desirable for unconstrained optimization, but is only defined for symmetric positive-definite operators. CR's main property, that it minimizes the residual, is particularly appealing in inexact Newton methods, typically used in a linesearch context. CR is also relevant in a trust-region context as it causes monotonic decrease of convex quadratic models (Fong and Saunders, 2012). We investigate modifications that make CR suitable, even in the presence of negative curvature, and perform comparisons on convex and nonconvex problems with the conjugate gradient method. We complete our investigation with an extension suitable for nonlinear least-squares problems. Our experiments reveal that CR performs as well as or better than CG, and mainly yields savings in operator-vector products.

Résumé : La méthode des résidus minimaux (MINRES) de Paige et Saunders (1975), qui est souvent la méthode privilégiée pour les systèmes linéaires symétriques, est une généralisation de la méthode des résidus conjugués (CR) proposée par Hestenes et Stiefel (1952). Tout comme la méthode du gradient conjugué (CG), CR possède des propriétés souhaitables en optimisation sans contraintes mais n'est définie que pour des opérateurs symétriques définis positifs. La principale propriété de cette méthode est qu'elle minimise le résidu, ce qui la rend particulièrement intéressante pour les méthodes de type Newton inexact, généralement utilisées en recherche linéaire. L'utilisation de CR est également pertinente en régions de confiance puisqu'elle fait décroître les modèles quadratiques convexes de manière monotone (Fong et Saunders, 2012). Nous étudions des modifications rendant CR applicable, même en présence de courbure négative, et les comparons avec CG sur des problèmes convexes et non convexes. Notre étude se termine par une extension aux problèmes aux moindres carrés non linéaires. Nos tests révèlent que CR se comporte aussi bien ou mieux que CG, et effectue en général moins de produits opérateur-vecteur.

Acknowledgments: Research partially supported by an NSERC Discovery Grant. The authors express their gratitude to Michael Saunders for comments on an earlier draft of the present research, and to Åke Björck for providing a copy of (Björck, 1979).

1 Introduction

We consider the large-scale unconstrained problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; f(x), \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is twice-continuously differentiable and possibly nonconvex. We wish to identify a first-order critical point of (1), i.e., $x_* \in \mathbb{R}^n$ such that $\nabla f(x_*) = 0$, using either a linesearch or a trust-region method. In a linesearch method, we compute a step as an approximate solution of

$$Hs = -g, \tag{2}$$

with $H = H^T \approx \nabla^2 f(x)$ and $g = \nabla f(x)$, while in a trust-region method, the step is an approximate solution of

$$\underset{s\in\mathbb{R}^n}{\text{minimize}} \;\; q(s) \quad \text{subject to } \|s\| \leq \Delta, \qquad q(s) := g^T s + \tfrac{1}{2} s^T H s, \tag{3}$$

where $\Delta > 0$ is the trust-region radius. We consider the inexact solution of (2) and (3) via an iterative method.

CG (Hestenes and Stiefel, 1952) has long been a workhorse in optimization because of its desirable properties. In particular, it is well suited for (2) because H is often forced to be positive definite so as to obtain a descent direction. Dembo and Steihaug (1983) develop modifications of CG to cope with directions of negative curvature in the context of inexact Newton methods that ensure global and fast local convergence. CG is also well suited for (3) because the value of q is monotonically decreasing along the CG iterations and the norm of the iterates is monotonically increasing. Thus if the iterates were to leave the trust region, they would never return. The k-th CG solves

$$\underset{s\in\mathbb{R}^n}{\text{minimize}} \;\; q(s) \quad \text{subject to } s \in K_k,$$

where $K_k = \text{Span}\{g, Hg, H^2 g, \ldots, H^{k-1} g\}$ is the k-th Krylov subspace. Steihaug (1983) describes truncated CG, in which directions of negative curvature are followed to the boundary of the trust region. Yuan (2000) shows that the approximate solution s^C identified by the truncated CG is such that $q(s^C) \leq \frac{1}{2} q(s^*)$, where s^* is a global solution of (3).

CR was introduced by Hestenes and Stiefel (1952) and Stiefel (1955), although the name is not mentioned in the first paper. Like CG, CR is a Krylov subspace method to solve (2) with positive definite H. At iteration k CR minimizes the residual norm $\|Hs + g\|_2$ in K_k. The search occurs along directions p_i that are conjugate with respect to H^2, while the residuals $r_i := -g - Hs_i$ are conjugate with respect to H, which explains the name of the method. In inexact Newton methods, we seek a step s such that $\|Hs + g\| \leq \tau \|g\|$ for a certain tolerance $\tau > 0$. Monotonicity of the residual norm seems like an appealing property to attain this stopping condition. Fong and Saunders (2012) establish that when H is positive definite, the values of q decrease monotonically along the CR iterations and the norm of the iterates is also monotonically increasing. Those observations lead us to believe that CR could also be well suited to computing steps in both linesearch and trust-region methods.

We describe modifications of CR analogous to those of CG to cope with directions of negative curvature in both linesearch and trust-region methods, and establish that global and local convergence properties are preserved.

Our numerical experiments on convex problems indicate that CR performs comparably to CG. CR exhibits a slight advantage over CG in terms of function, gradient and Hessian-vector evaluations on nonconvex problems, which makes it a viable general-purpose subproblem solver.

Paige and Saunders (1975) describe MINRES, which generates the same iterates as CR in the definite case but is more general in that it also solves indefinite systems. Because the implementation of MINRES is

substantially more complicated than that of CR, we do not consider it in this paper. We note however that it should be possible to implement our modified versions of CR as part of MINRES.

The rest of this paper is organized as follows. In Section 2, we introduce the basic CR and its main properties. Section 3 gives a variant of CR in a linesearch inexact-Newton context, corresponding global and local convergence results, and numerical comparisons against CG. In Section 4, we study CR in a trust-region context, provide theoretical results ensuring global convergence, and report on numerical experience. In Section 5, we provide a variant of CR suitable for the solution of nonlinear least-squares problems in a trust-region context, and present numerical comparisons with the corresponding variant of CG. Concluding remarks appear in Section 6. Complete details of our numerical experiments are provided in the appendix.

Notation

The norm used throughout is the Euclidean norm. Uppercase Latin letters denote matrices, lowercase Latin letters denote vectors, and Greek letters denote scalars.

2 Derivation of CR

Luenberger (1970) and Fong (2011) establish properties of CR when H is symmetric positive definite. We base our description of CR on the pseudocode of Fong and Saunders (2012) with some modifications explained below.

The main idea behind CR is to solve (2) with H symmetric and positive definite by solving the equivalent problem

$$\underset{s\in\mathbb{R}^n}{\text{minimize}}\ \bar{q}(s), \qquad \bar{q}(s) := \tfrac{1}{2}\|\nabla q(s)\|^2 = \tfrac{1}{2}\|g + Hs\|^2. \tag{4}$$

Note that $\nabla\bar{q}(s) = H(g + Hs)$ and $\nabla^2\bar{q}(s) = H^2$. For any s, let $r := -g - Hs$ denote the residual of (2) so that $\nabla\bar{q}(s) = -Hr$. The algorithm starts from any s_0 and initializes $r_0 = -g - Hs_0$. It is customary to set $s_0 := 0$ and $r_0 := -g$. Given an iterate s_k and a descent direction p_k, CR computes a steplength $\alpha_{k+1} > 0$ such that $s_{k+1} := s_k + \alpha_{k+1}p_k$ minimizes $\bar{q}$ in the direction p_k, i.e., such that $\nabla\bar{q}(s_{k+1})^T p_k = 0$. Thus by construction, CR produces monotonic $\|r_k\|$.

Because $\bar{q}$ is quadratic,

$$\nabla\bar{q}(s_{k+1}) = \nabla\bar{q}(s_k) + \alpha_{k+1}\nabla^2\bar{q}(s_k)p_k, \tag{5}$$

and

$$\nabla\bar{q}(s_{k+1})^T p_k = -r_k^T Hp_k + \alpha_{k+1}p_k^T H^2 p_k,$$

so that

$$\alpha_{k+1} = r_k^T Hp_k / p_k^T H^2 p_k.$$

As we show below in (15), $r_k^T Hp_k = r_k^T Hr_k$. Thanks to (5), the residual at s_{k+1} can be updated as $r_{k+1} = r_k - \alpha_{k+1}Hp_k$. The next search direction is defined as $p_{k+1} := r_{k+1} + \beta_{k+1}p_k$, where β_{k+1} is chosen so that $p_{k+1}^T H^2 p_k = 0$, i.e., $\beta_{k+1} = -r_{k+1}^T H^2 p_k / p_k^T H^2 p_k$. Luenberger (1970, Theorem 1) shows that a consequence of the choice of p_{k+1} is that the search directions are H^2-conjugate and that $\beta_{k+1} = r_{k+1}^T Hr_{k+1} / r_k^T Hr_k$.

Algorithm 1 describes the classical CR. It differs from the one presented by Fong and Saunders (2012) by the addition of $\rho_k = \|r_k\|^2$, $\nu_k = \|r_k\|$ and a stopping criterion with absolute and relative tolerances.

Theorem 1 summarizes properties of CR that we use throughout the rest of this paper.

Theorem 1 *Assume H is positive definite in Algorithm 1 and $r_i \neq 0$ for $i = 0, \ldots, k$ in Algorithm 1. The following properties hold:*

$$p_i^T H^2 p_j = 0, \qquad i \neq j, \qquad i, j = 0, \ldots, k, \tag{6}$$

$$r_i^T Hr_j = 0, \qquad i \neq j, \qquad i, j = 0, \ldots, k, \tag{7}$$

Algorithm 1 CR for (2)

Require: H, g, $\tau_a > 0$, $\tau_r > 0$
Initialize: $k = 0$, $s_0 = 0$, $r_0 = -g$, $u_0 = Hr_0$, $\zeta_0 = r_0^T u_0$, $p_0 = r_0$, $q_0 = u_0$, $\rho_0 = r_0^T r_0$, $\nu_0 = \sqrt{\rho_0}$
while $\nu_k > \tau_a + \tau_r \|g\|$ **do**
  $k \leftarrow k + 1$
  $\alpha_k = \zeta_{k-1} / \|q_{k-1}\|^2$
  $s_k = s_{k-1} + \alpha_k p_{k-1}$
  $r_k = r_{k-1} - \alpha_k q_{k-1}$
  $\rho_k = \rho_{k-1} - \alpha_k \zeta_{k-1}$ $\rho_k = \|r_k\|^2$
  $\nu_k = \sqrt{\rho_k}$
  $u_k = Hr_k$
  $\zeta_k = r_k^T u_k$ $\zeta_k = r_k^T H r_k$
  $\beta_k = \zeta_k / \zeta_{k-1}$
  $p_k = r_k + \beta_k p_{k-1}$
  $q_k = u_k + \beta_k q_{k-1}$ $q_k = Hp_k$
return s_k

$$p_i^T H r_j = 0, \qquad 0 \le i < j \le k, \tag{8}$$
$$r_i^T H r_i > 0, \qquad i = 0, \ldots, k, \tag{9}$$
$$\text{Span}\{r_0, r_1, \ldots, r_k\} = \text{Span}\{g, Hg, \ldots, H^k g\}, \tag{10}$$
$$\|s_i\| < \|s_j\|, \qquad i < j, \qquad i, j = 0, \ldots, k, \tag{11}$$
$$\alpha_i > 0, \qquad i = 1, \ldots, k, \tag{12}$$
$$p_i^T r_j > 0, \qquad 0 \le i, j \le k. \tag{13}$$

Proof. Properties (6)–(8) come respectively from (Luenberger, 1970, Theorem 1 (a), (d) and (b)), (9) follows from the positive definiteness of H, (11) is proven by Fong (2011, Theorem 2.1.6), and (12)–(13) come from (Fong, 2011, Theorem 2.1.5).

By construction, s_i minimizes the norm of the residual in K_i. We show by induction that $r_i \in K_{i+1}$ and $p_i \in K_{i+1}$. Initially, $r_0 = p_0 = g \in K_1$. Let the assumption be satisfied for index i. We have $r_{i+1} = r_i - \alpha_{i+1} H p_i$. Since r_i and p_i are in K_{i+1}, we know $Hp_i \in K_{i+2}$ and so $r_{i+1} \in K_{i+2}$. Also, $p_{i+1} = r_{i+1} + \beta_{i+1} p_i$ so $p_{i+1} \in K_{i+2}$, which establishes (10). □

Theorem 2 *Assume H is positive definite in Algorithm 1. For all $k \ge 0$,*

$$q_k = Hp_k, \tag{14}$$
$$\zeta_k = r_k^T q_k, \tag{15}$$
$$\rho_k = \|r_k\|^2, \tag{16}$$
$$\nu_k = \|r_k\|. \tag{17}$$

Proof. Equality (14) can be established by induction. Initially, $q_0 = u_0 = Hr_0 = Hp_0$. Now assume that the property holds for index k. Then, by recurrence,

$$q_{k+1} = u_{k+1} + \beta_{k+1} q_k = Hr_{k+1} + \beta_{k+1} Hp_k = H(r_{k+1} + \beta_{k+1} p_k) = Hp_{k+1}.$$

We have $r_k^T q_k = r_k^T (u_k + \beta_k q_{k-1})$. But (8) and (14) yield $r_k^T q_{k-1} = 0$, so $r_k^T q_k = r_k^T H r_k = \zeta_k$, which establishes (15).

Because $\rho_0 = r_0^T r_0 = \|r_0\|^2$, (16) holds for $k = 0$. Then, (8), (14), (15) and a recursion assumption yield

$$\|r_{k+1}\|^2 = r_{k+1}^T (r_k - \alpha_{k+1} q_k) = (r_k - \alpha_{k+1} q_k)^T r_k = \rho_k - \alpha_{k+1} \zeta_k = \rho_{k+1}.$$

Property (17) follows because $\nu_k = \sqrt{\rho_k}$. □

The following result guides the detection of nonpositive curvature in the next sections, and parallels a similar result for CG. Its proof is as in Dembo and Steihaug (1983) with directions d_k replaced by residuals r_k.

Theorem 3 (Dembo and Steihaug, 1983, Theorem A.5) *Consider Algorithm 1. Let m be the number of distinct eigenvalues of H. If g has a nonzero projection on each eigenspace and if*

$$r_i^T H r_j = 0, \quad i, j = 0, \ldots, m-1, \ (i \neq j), \tag{18}$$

$$r_i^T H r_i > 0, \quad i = 0, \ldots, m-1, \tag{19}$$

$$\text{Span}\{r_0, \ldots, r_{m-1}\} = \text{Span}\{g, Hg, \ldots, H^{m-1}g\}, \tag{20}$$

then H is positive definite.

3 CR in a linesearch context

In this section, we are interested in solving (1) by way of a linesearch method. The main motivation is that because CR produces monotonic $\|r_k\|$, it appears suitable for computing steps in an inexact Newton scheme. The linesearch subproblem is described by (2). As f may be nonconvex, H may not be positive definite.

Motivated by saddle-point systems and constrained problems, Luenberger (1970) modifies CR to solve indefinite systems. His modification is active when Algorithm 1 computes $\alpha_k = 0$, which may happen if H is not positive definite, and coincides with the original CR in the positive definite case. A disadvantage of his approach is that one must decide numerically when α_k should be treated as zero. Our strategy differs in that we check the curvature and not the step length, in the spirit of the modified CG developed by Dembo and Steihaug (1983). We maintain all the properties of CR because our modification stops when negative curvature is encountered.

3.1 Linesearch CR algorithm

Let $\pi_i = p_i^T p_i$, $\rho_i = r_i^T r_i$ for $i = 0, 1, \ldots, k$ and $\epsilon > 0$ be a constant value. In Algorithm 2 we modify CR in the context of a linesearch method to solve (2) when H is not necessarily positive definite.

Algorithm 2 Modified CR (linesearch version)

Require: H, g, $\tau_a > 0$, $\tau_r > 0$, $\epsilon > 0$
Initialize: $k = 0$, $s_0 = 0$, $r_0 = -g$, $u_0 = Hr_0$, $\zeta_0 = r_0^T u_0$, $p_0 = r_0$, $q_0 = u_0$, $\delta_0 = \zeta_0$, $\rho_0 = r_0^T r_0$, $\mu_0 = \rho_0$, $\pi_0 = \rho_0$, $\nu_0 = \sqrt{\rho_0}$
while $\nu_k > \tau_a + \tau_r \|g\|$ **do**
  $k \leftarrow k + 1$
  if $\delta_{k-1} \leq \epsilon \pi_{k-1}$ or $\zeta_{k-1} \leq \epsilon \rho_{k-1}$ **then** *(near) negative curvature detected*
    if $k = 1$ **then**
      return $-g$
    else
      return s_{k-1}
  $\alpha_k = \zeta_{k-1} / \|q_{k-1}\|^2$
  $s_k = s_{k-1} + \alpha_k p_{k-1}$
  $r_k = r_{k-1} - \alpha_k q_{k-1}$
  $\rho_k = \rho_{k-1} - \alpha_k \zeta_{k-1}$ $\quad \rho_k = \|r_k\|^2$
  $\nu_k = \sqrt{\rho_k}$ $\quad \nu_k = \|r_k\|$
  $u_k = Hr_k$
  $\zeta_k = r_k^T u_k$ $\quad \zeta_k = r_k^T H r_k$
  $\beta_k = \zeta_k / \zeta_{k-1}$
  $p_k = r_k + \beta_k p_{k-1}$
  $\pi_k = \rho_k + 2\beta_k(\mu_{k-1} - \alpha_k \delta_{k-1}) + \beta_k^2 \pi_{k-1}$ $\quad \pi_k = \|p_k\|^2$
  $\mu_k = \rho_k + \beta_k(\mu_{k-1} - \alpha_k \delta_{k-1})$ $\quad \mu_k = p_k^T r_k$
  $q_k = u_k + \beta_k q_{k-1}$ $\quad q_k = Hp_k$
  $\delta_k = \zeta_k + \beta_k^2 \delta_{k-1}$ $\quad \delta_k = p_k^T H p_k$
return s_k

The main difference with Algorithm 1 resides in the condition on line 4 and recursion for π_k. We first note that Theorem 1 continues to hold for Algorithm 2 for as long as the search directions and residuals are directions of positive curvature.

Corollary 1 *Let $\epsilon > 0$. If $p_i^T H p_i > \epsilon \pi_i$ and $r_i^T H r_i > \epsilon \rho_i$ for $i = 0, \ldots, k$. The properties of Theorem 1 hold for Algorithm 2 at iterations $i = 0, \ldots, k$.*

Because of (7), there will be an iteration k such that $r_k^T H r_k \leq 0$ if H is not positive definite and Algorithm 2 does not terminate earlier. Thus, the residual will eventually signal the presence of nonpositive curvature. However, because of (6), we must check the sign of $p_k^T H p_k$ explicitly at each iteration to discover whether p_k is a direction of positive curvature or not. If nearly negative curvature is detected, either the previous iterate s_{k-1} is returned if $k \geq 2$ or $-g$ is returned if $k = 1$ as it is known to be a descent direction.

Theorem 4 *Let H be symmetric but not necessarily positive definite. Identities (14)–(17) continue to hold for Algorithm 2. The following equalities also hold:*

$$\delta_k = p_k^T H p_k, \tag{21}$$

$$\mu_k = p_k^T r_k, \tag{22}$$

$$\pi_k = \|p_k\|^2. \tag{23}$$

Proof. The proof of (14)–(17) is the same as in Theorem 2. We have $\delta_0 = \zeta_0$ and $\forall k \geq 1, \delta_k = \zeta_k + \beta_k^2 \delta_{k-1}$. By induction we can show that $\delta_k = p_k^T H p_k$. Indeed, as $p_0 = r_0$ then $\delta_0 = p_0^T H p_0$. Now suppose the property satisfied for $k \geq 0$. We then have

$$p_{k+1}^T H p_{k+1} = p_{k+1}^T q_{k+1} = (r_{k+1} + \beta_{k+1} p_k)^T (u_{k+1} + \beta_{k+1} q_k).$$

Since H is symmetric, $p_k^T u_{k+1} = r_{k+1}^T q_k = 0$ according to (8). This leads to

$$p_{k+1}^T H p_{k+1} = r_{k+1}^T u_{k+1} + \beta_{k+1}^2 p_k^T q_k = \zeta_{k+1} + \beta_{k+1}^2 \delta_k = \delta_{k+1}$$

Thus, (21) is established.

We have $\mu_0 = \rho_0 = \|r_0\|^2$. But $r_0 = p_0$ so (22) is satisfied at iteration 0. Suppose by induction that it is verified at iteration $k \geq 0$. Then

$$\begin{aligned} p_{k+1}^T r_{k+1} &= (r_{k+1} + \beta_{k+1} p_k)^T r_{k+1} \\ &= \rho_{k+1} + \beta_{k+1} p_k^T r_{k+1} \\ &= \rho_{k+1} + \beta_{k+1} p_k^T (r_k - \alpha_{k+1} q_k) \\ &= \rho_{k+1} + \beta_{k+1} \mu_k - \alpha_{k+1} \beta_{k+1} \delta_k \\ &= \mu_{k+1} \end{aligned}$$

which establishes (22).

Similarly, (23) can be shown by induction. First, $\pi_0 = \rho_0 = \|p_0\|^2$. If $k \geq 0$ satisfies the property then

$$p_{k+1}^T p_{k+1} = (r_{k+1} + \beta_{k+1} p_k)^T (r_{k+1} + \beta_{k+1} p_k) = \rho_{k+1} + 2\beta_{k+1} r_{k+1}^T p_k + \beta_{k+1}^2 \pi_k.$$

The update for r_{k+1} yields

$$r_{k+1}^T p_k = (r_k - \alpha_{k+1} q_{k-1})^T p_k = \mu_k - \alpha_{k+1} \delta_k.$$

Finally,

$$\pi_{k+1} = \rho_{k+1} + 2\beta_{k+1}(\mu_k - \alpha_{k+1} \delta_k) + \beta_{k+1}^2 \pi_k.$$

□

The last result of this section follows from Theorem 3 and justifies that if g is not orthogonal to the eigenspaces of H associated with nonpositive eigenvalues, Algorithm 2 eventually detects nonpositive curvature.

Theorem 5 (Dembo and Steihaug, 1983, Theorem 2.4) *Let the stopping criterion in Algorithm 2 be $\|r_k\| = 0$ and let $\epsilon = 0$. If H is not positive definite, either*

1. *$r_k^T H r_k \leq 0$ for a certain iteration k, or*
2. *g is orthogonal to the eigenspaces associated with the nonpositive eigenvalues of H.*

3.2 Global convergence

In this subsection we show that the truncated-Newton method paired with Algorithm 2 is globally convergent. Our analysis follows that of Dembo and Steihaug (1983), and so does Algorithm 3.

Algorithm 3 Truncated Newton Method for (1)

Require: x_0, $\epsilon_a > 0$, $\epsilon_r > 0$
1: **Compute:** $f_0 = f(x_0)$, $g_0 = g(x_0)$, set $k = 0$
2: **while** $\|g_k\| > \epsilon_a + \epsilon_r \|g_0\|$ **do**
3: Choose $H_k = H_k^T \approx \nabla^2 f(x_k)$, $\tau_a > 0$, $\tau_r > 0$
4: Compute s_k such that $\|H_k s + g_k\| \leq \tau_a + \tau_r \|g_k\|$ *use Algorithm 2*
5: Compute $t_k > 0$ that satisfies the Wolfe conditions

$$f(x_k + t_k s_k) \leq f_k + \alpha t_k g_k^T s_k \qquad \alpha \in (0, \tfrac{1}{2}) \tag{24}$$

$$\nabla f(x_k + t_k s_k)^T s_k \geq \beta g_k^T s_k \qquad \beta \in (\alpha, 1) \tag{25}$$

6: $x_{k+1} = x_k + t_k s_k$
7: $g_{k+1} = \nabla f(x_{k+1})$
8: $k \leftarrow k + 1$

Lemma 1 (Dembo and Steihaug, 1983, Lemma A.2) *Let $\epsilon > 0$ be as in Algorithm 2. Assume that there exists $M > 0$ such that $\|H_k\| \leq M$ for all k in Algorithm 3. If $g_k \neq 0$, there exist constants $\gamma_0 > 0$ and $\gamma_1 > 0$ that only depend on ϵ and M such that*

$$g_k^T s_k \leq -\gamma_0 \|g_k\|^2, \quad \textit{and} \tag{26}$$

$$\|s_k\| \leq \gamma_1 \|g_k\|. \tag{27}$$

Proof. In Step 4 of Algorithm 3, Algorithm 2 is initialized with $r_0 = -g_k$. If $r_0^T H_k r_0 \leq \epsilon \rho_0$ then $s_k = -g_k$ and we may choose $\gamma_0 = 1$ in (26). Otherwise, the i-th iteration of Algorithm 2 sets $s_i = \sum_{j=1}^{i} \alpha_j p_{j-1}$ where $\alpha_j = r_{j-1}^T H_k r_{j-1} / \|H_k p_{j-1}\|^2$. Thus, $g_k^T s_i = -\sum_{j=1}^{i} \alpha_j r_0^T p_{j-1}$. For $j = 1, \ldots, i$, we have from Corollary 1 and (9) that $r_{j-1}^T H_k r_{j-1} > 0$, while (13) yields $r_0^T p_{j-1} > 0$. Hence, for $j = 1, \ldots, i$,

$$g_k^T s_i \leq -r_0^T p_0 \frac{r_0^T H_k r_0}{\|H_k p_0\|^2} \leq -\|g_k\|^2 \frac{g_k^T H_k g_k}{\|H_k\|^2 \|g_k\|^2} \leq -\frac{g_k^T H_k g_k}{\|H_k\|^2}.$$

But $g_k^T H_k g_k > \epsilon g_k^T g_k$, so $g_k^T s_i \leq -\epsilon \|g_k\|^2 / \|H_k\|^2 \leq -\epsilon \|g_k\|^2 / M^2$. Thus, (26) holds with $\gamma_0 = \min(1, \epsilon / M^2)$.

Similarly, if $s_i = -g_k$, we may choose $\gamma_1 = 1$ in (27). Otherwise,

$$\|s_i\| = \left\| \sum_{j=1}^{i} \frac{r_{j-1}^T H_k r_{j-1}}{\|H_k p_{j-1}\|^2} p_{j-1} \right\| \leq \sum_{j=1}^{i} \frac{r_{j-1}^T H_k r_{j-1}}{\|H_k p_{j-1}\|^2} \|p_{j-1}\| \leq \frac{r_0^T H_k r_0}{\|H_k p_0\|^2} \|p_0\|.$$

But $\epsilon \|p_0\|^2 < p_0^T H_k p_0 \leq \|p_0\| \|H_k p_0\|$, so $\|H_k p_0\| > \epsilon \|p_0\|$. Moreover $r_0^T H_k r_0 \leq \|H_k\| \|r_0\|^2$. Thus, $\|s_i\| \leq \|H_k\| / \epsilon^2 \, \|g_k\| \leq M / \epsilon^2 \, \|g_k\|$. Finally, (27) holds with $\gamma_1 = \max(1, M/\epsilon^2)$. □

Global convergence of Algorithm 3 follows from the next two results, whose proofs are identical to those of (Dembo and Steihaug, 1983, Theorems 2.1 and A.3).

Theorem 6 (Dembo and Steihaug, 1983, Theorem A.3) *Consider Algorithm 3 in which steps are computed using Algorithm 2. The sequence of iterates $\{x_k\}$ is well defined and $\lim_{k\to\infty} \|g_k\| = 0$.*

Theorem 7 (Dembo and Steihaug, 1983, Theorem 2.1) *If the sequence $\{x_k\}$ generated by Algorithm 3 has a limit point x^* where $H(x^*)$ is positive definite then the whole sequence $\{x_k\}$ converges to x^*.*

3.3 Local convergence

We now show that Algorithm 3 has fast local convergence properties provided the tolerances τ_a and τ_r are chosen appropriately.

Lemma 2 *Consider Algorithm 3 in which steps are computed using Algorithm 2 and assume that $g_k \neq 0$ and that there exists $M > 0$ such that $\|H_k\| \leq M$ and H_k is positive definite for all k. If $\tau_a = 0$ and $\tau_r = o(1)$, then $\|g_k + H_k s_k\| = o(\|s_k\|)$.*

Proof. The termination condition of CR is $\|g_k + H_k s_k\| \leq \tau_k$ with $\tau_k = \tau_a + \tau_r \|g_k\| = o(\|g_k\|)$ by assumption. Moreover, CR does at least one iteration. Indeed, if it were not the case and $s_k = 0$, that would mean that $\|g_k\| \leq \tau_r \|g_k\|$, and therefore that $g_k = 0$, so that Algorithm 3 would have stopped earlier. Because H_k is positive definite,

$$g_k^T H_k g_k = p_{k0}^T H_k p_{k0} > \epsilon \|p_{k0}\|^2 = \epsilon \|g_k\|^2.$$

By (11), $\|s_k\| \geq \|s_{k1}\| = \alpha_{k1} g_k$, the second index being the CR iteration. So,

$$\frac{\|g_k + H_k s_k\|}{\|s_k\|} \leq \frac{\tau_k}{\alpha_{k1}\|g_k\|} = \frac{\|H_k g_k\|^2}{g_k^T H_k g_k} \frac{\tau_k}{\|g_k\|} \leq \frac{\|H_k\|^2 \|g_k\|^2}{g_k^T H_k g_k} \frac{\tau_k}{\|g_k\|} \leq \frac{M^2}{\epsilon} \frac{\tau_k}{\|g_k\|}.$$

Because $\tau_k = o(\|g_k\|)$, we have $\|g_k + H_k s_k\| = o(\|s_k\|)$. □

Lemma 2 allows us to establish the following result, which parallels (Dembo and Steihaug, 1983, Theorem 2.2) and is an application of (Dennis and Moré, 1977, Theorem 6.4).

Theorem 8 *Let the sequence of iterates $\{x_k\}$ generated by Algorithm 3 converge to x^* such that $H(x^*)$ is symmetric positive definite. If the Goldstein-Armijo conditions (24) and (25) are satisfied and $\|g_k + H_k s_k\| = o(\|s_k\|)$, then for ϵ sufficiently small in Algorithm 2, there exists an iteration k_0 such that the stopping criterion on the residual norm in Algorithm 2 is satisfied and such that for all $k \geq k_0$, $t_k = 1$ is an acceptable step for the linesearch.*

The following result states the local convergence rate. The proof is identical to that of (Dembo and Steihaug, 1983, Theorem 2.3).

Theorem 9 (Dembo and Steihaug, 1983, Theorem 2.3) *Let the sequence $\{x_k\}$ generated by Algorithm 3 converge to x^* where $H(x^*)$ is positive definite. Suppose that H is Lipschitz continuous at x^*. If the k-th iteration of Algorithm 3 sets $\tau_a = 0$ and $\tau_r = \min(1/k, \|g(x_k)\|^t)$ for some $0 < t \leq 1$, then $\{x_k\}$ converges at a rate $1 + t$.*

3.4 Numerical results

In this section, we compare the performance of Algorithm 3 where either CG or CR is used to compute steps. We use the Julia[1] programming language, version 0.6 to implement all three of Algorithm 2, the truncated CG algorithm of Dembo and Steihaug (1983), and Algorithm 3.

[1] https://julialang.org

We use convex and nonconvex unconstrained problems from the CUTEst collection of Gould, Orban, and Toint (2015), accessed via the CUTEst.jl[2] Julia interface, and the modified CUTE problems of Lukšan, Matonoha, and Vlček (2010) as implemented in the Julia library OptimizationProblems.jl.[3] We temporarily exclude nonlinear least-squares problems, which are evaluated in Section 5. We further eliminate problems with fewer than 10 variables and we remove duplicates. Five additional problems are eliminated because they could not be solved within 1.5 hours: *indefm*, *nonmsqrt*, *sbrybnd*, *scosine* and *sscosine*.

Among our problems, 17 from OptimizationProblems.jl are known to be convex (Fourer, Maheshwari, Neumaier, Orban, and Schichl, 2010), : *arglina*, *arglinb*, *arglinc*, *arwhead*, *bdqrtic*, *clplatea*, *clplateb*, *clplatec*, *dixon3dq*, *dqdrtic*, *dqrtic*, *engval1*, *nondquar*, *power*, *quartc*, *tridia*, *vardim*, and their dimension varies from 10 to 10,000. The other 50 from OptimizationProblems.jl are nonconvex and their dimension ranges from 10 to 100. In total, we have 107 problems of dimension 10 to 100,000.

In Algorithm 2 and truncated CG, we set the maximum number of iterations to the number of variables. We impose a maximum of 10,000 iterations in Algorithm 3 and set $\tau_a = 0$, $\tau_r = \min(0.1, \sqrt{\|g_k\|})$, and $\epsilon_a = \epsilon_r = 10^{-6}$. Finally, we use a simple Armijo backtracking linesearch with $\alpha = 10^{-4}$, which only ensures (24). This is the way Algorithm 3 is often implemented in practice.

Our results are presented in the form of $\log_2$-scaled performance profiles of the number of objective evaluations, gradient evaluations, Hessian-vector products, and the sum of the three measures. We report results on convex and nonconvex problems separately, as convex problems do not trigger the modifications to CR pertaining to negative curvature. We also report results on the entire problem set.

Figure 1 contains the profiles for convex problems and shows that CG and CR perform equivalently, though CR yields slight savings in terms of all measures.

Figure 2 illustrates that both methods are essentially equivalent on nonconvex problems in terms of gradient evaluations. CR shows more substantial savings in terms of function evaluations and Hessian-vector products than in the convex case, indicating that fewer iterations of Algorithm 2 than of CG are necessary to attain the stopping criterion, and the search direction induces fewer backtracking steps.

Profiles for all 107 problems together, including those from CUTEst.jl, appear in Figure 3. CG performs slightly better in terms of gradient evaluations while CR shows a slight advantage in terms of Hessian-vector products. CR fails on 10 problems overall while CG fails on 6 problems. On those, the maximum number of iterations of Algorithm 3 is reached, except for problem parkch, where the magnitude of the negative curvature directions identified by Algorithm 2 becomes so large that we eventually evaluate the log likelihood objective with arguments that result in NaNs.

In conclusion, the CR variant performs equivalently or slightly better than CG in a linesearch context. The difference is most visible in terms of Hessian-vector products.

Overall, we observe that CR has a slight advantage over CG in an inexact Newton context, where a method that produces monotonic residuals is desirable.

4 CR in a trust-region context

Most large-scale implementations of trust-region methods compute an approximate solution s of (3) using the truncated CG method of Steihaug (1983). The basic mechanism is to temporarily ignore the trust-region constraint on the step and apply CG as if q were convex. The fundamental properties of CG, reviewed in Section 4.1, make it particularly well suited to this task. Steihaug (1983) describes simple modifications to CG to account for situations where the next CG iterate lies outside the trust region, or where the current CG search direction p is a direction of nonpositive curvature for q, i.e., $p^T H p \leq 0$. Yuan (2000) establishes that truncated CG yields an approximate minimizer s^C such that $q(s^C) \leq \frac{1}{2} q_k(s^*)$, where s^* is a global minimizer

[2] https://github.com/JuliaSmoothOptimizers/CUTEst.jl
[3] https://github.com/JuliaSmoothOptimizers/OptimizationProblems.jl

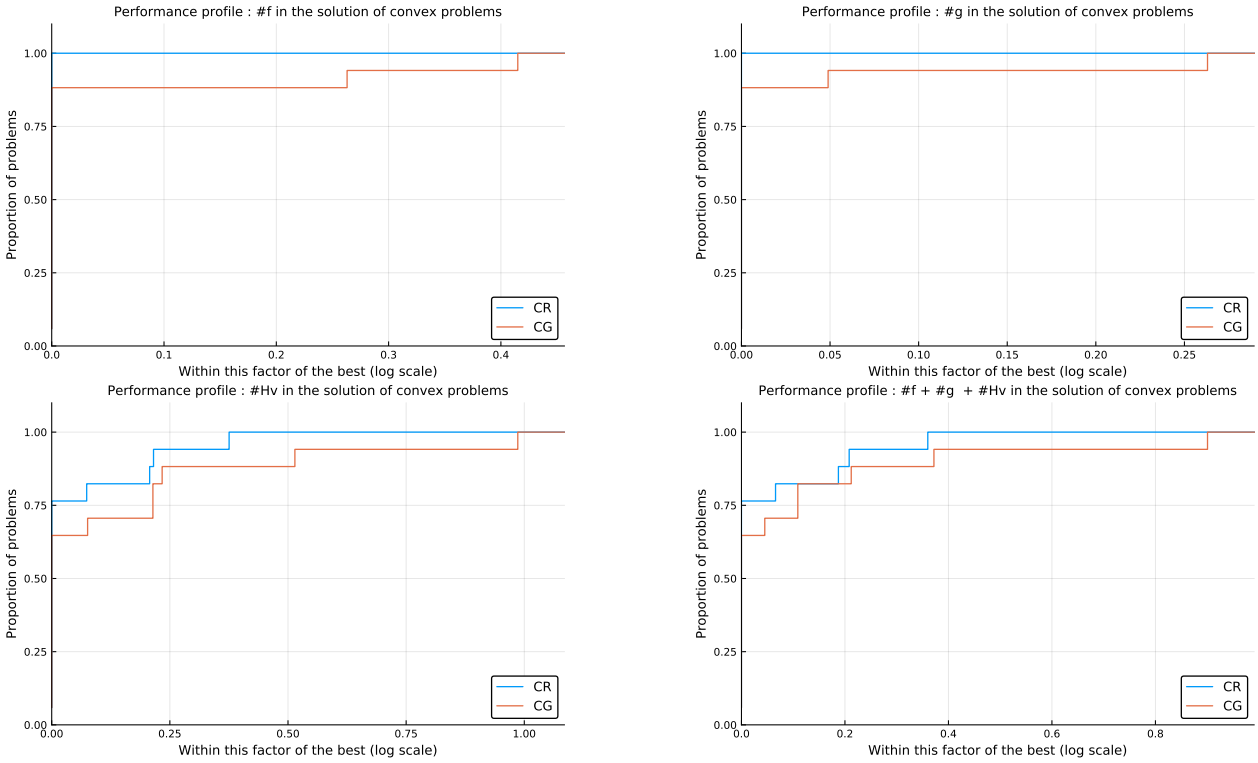

Figure 1: Performance of linesearch CR and CG to solve 17 convex problems in terms of evaluations of f, g and products with H.

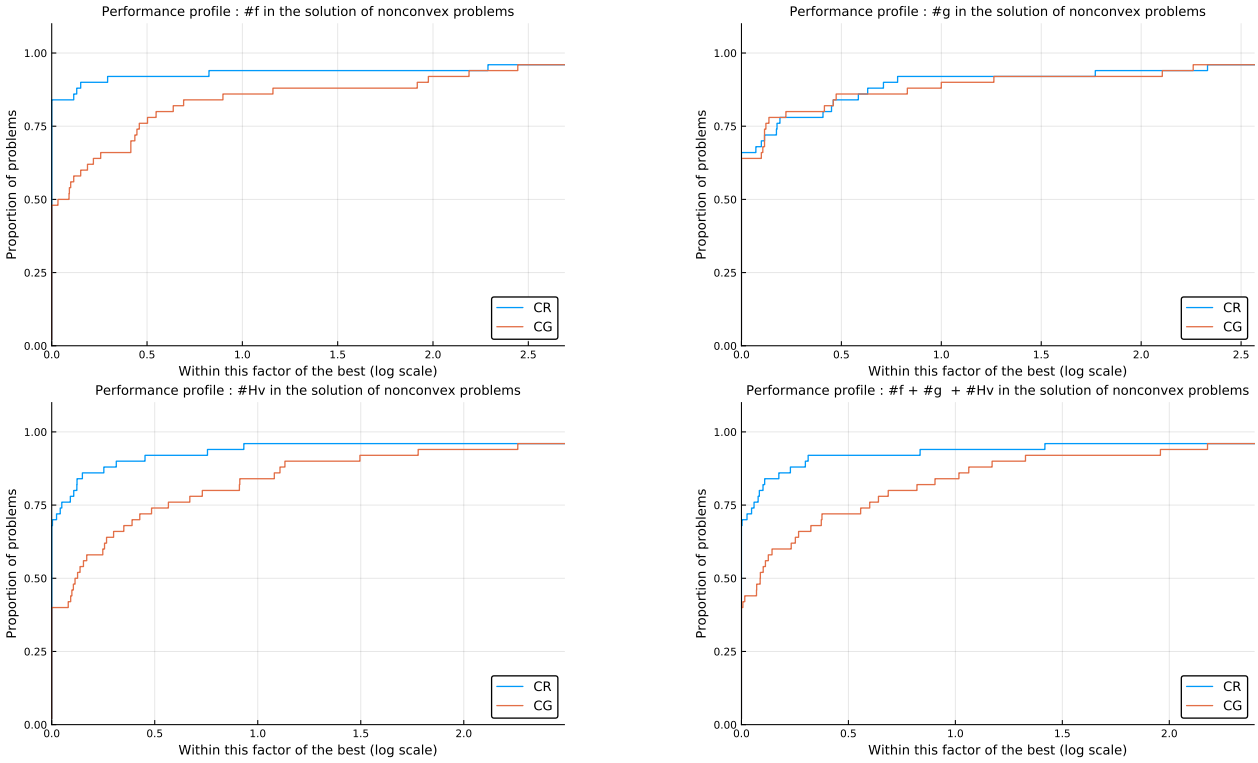

Figure 2: Performance of linesearch CR and CG on 50 nonconvex problems in terms of evaluations of f, g and products with H.

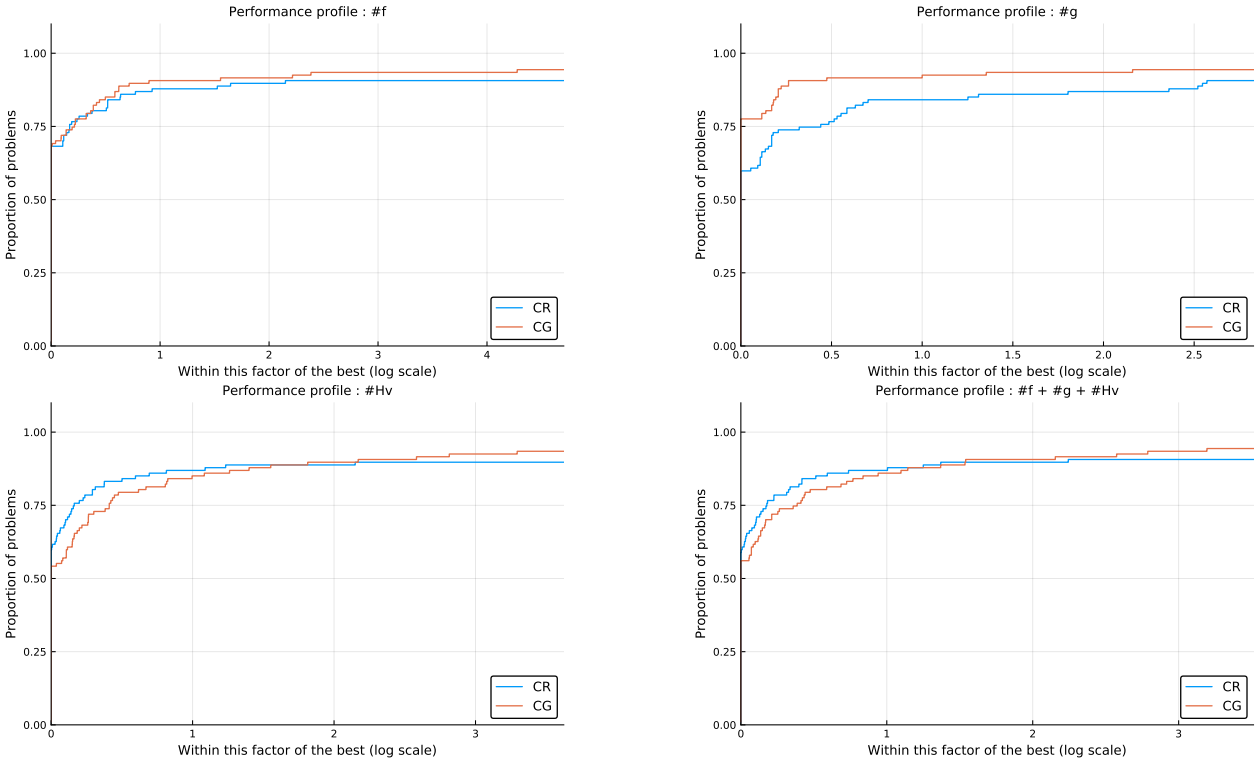

Figure 3: Performance of linesearch CR and CG on 107 problems in terms of evaluations of f, g and products with H.

of (3). Algorithm 4 gives the trust-region process. In the remainder of this section, we introduce a version of CR to be used at line 5. We refer the reader to (Conn, Gould, and Toint, 2000) for a comprehensive account of trust-region methods.

Algorithm 4 Trust-Region Method for (1)

Require: x_0, $\Delta_0 > 0$, $\epsilon_a > 0$, $\epsilon_r > 0$, $0 < \eta_1 \leq \eta_2 < 1$, $0 < \gamma_1 \leq \gamma_2 < 1$
1: **evaluate** $f_0 = f(x_0)$, $g_0 = \nabla f(x_0)$
2: **initialize** $k = 0$
3: **while** $\|g_k\| > \epsilon_a + \epsilon_r \|g_0\|$ **do**
4: **choose** $H_k = H_k^T \approx \nabla^2 f(x_k)$
5: **compute** s_k by approximately solving (3);
6: stop if $\|H_k s_k + g_k\| \leq \tau_a + \tau_r \|g_k\|$ or $\|s_k\| = \Delta_k$ *use Algorithm 5*
7: **compute** $\sigma_k = (f_k - f(x_k + s_k))/(q_k(x_k) - q_k(x_k + s_k))$
8: **if** $\sigma_k < \eta_1$ **then** *unsuccessful iteration*
9: $x_{k+1} = x_k$, $f_{k+1} = f_k$, $g_{k+1} = g_k$
10: **set** $\Delta_{k+1} \in [\gamma_1 \Delta_k, \gamma_2 \Delta_k]$
11: **else** *successful iteration*
12: $x_{k+1} = x_k + s_k$
13: **evaluate** $f_{k+1} = f(x_{k+1})$, $g_{k+1} = \nabla f(x_{k+1})$
14: **if** $\sigma_k \geq \eta_2$ **then** *very successful iteration*
15: **set** $\Delta_{k+1} \in [\Delta_k, \infty)$
16: **else**
17: **set** $\Delta_{k+1} \in [\gamma_2 \Delta_k, \Delta_k]$
18: $k \leftarrow k + 1$
19: **return** x_k

4.1 Background

Convergence of trust-region methods to a stationary point hinges around the concept of decrease obtained at the Cauchy point, which is determined by the solution of

$$\underset{\alpha}{\text{minimize}}\ q_k(-\alpha g_k) \quad \text{subject to } 0 \leq \alpha \leq \Delta/\|g_k\|,$$

i.e., the minimizer of the model along the steepest-descent direction and inside the trust region. The Cauchy point may or may not lie on the boundary. If q_k is convex along the steepest-descent direction but the unconstrained minimizer lies outside the trust region, or if q is concave along the steepest-descent direction, the Cauchy point lies on the boundary. However we compute an approximate solution s_k of (3), convergence will be guaranteed provided we satisfy the sufficient decrease condition, which requires that our step achieve at least a fixed fraction of Cauchy decrease at each trust-region subproblem. The sufficient-decrease condition demands that there exist a constant $\kappa \in (0, 1)$, independent of k, such that

$$q_k(s_k) \leq -\kappa \|g_k\| \min\left(\frac{\|g_k\|}{1 + \|H_k\|}, \Delta_k\right). \tag{28}$$

We refer to (Conn et al., 2000, Section 6.3.4) for more information.

If we neglect the trust-region constraint temporarily, the first-order optimality condition of (3) may be stated as the symmetric linear system $H_k s = -g_k$.

Both CG and CR can be constructed from the Lanczos (1950) process, which theoretically generates an orthonormal basis $\{v_1, v_2, v_3, \dots\}$ of the increasing Krylov subspaces $\text{Span}\{-g, -Hg, -H^2g, \dots\}$.

CG has several desirable properties that make it a natural candidate for (3), even when H_k is indefinite. Those properties follow from the definition of the method and (Steihaug, 1983, Theorem 2.1) and are summarized in the following result.

Theorem 10 *Assume H is positive definite and s_1, s_2, ... are the iterates generated by CG on $Hs = -g$. Then,*

1. $\|s_{j+1}\| > \|s_j\|$ *for* $j = 1, 2, \dots,$
2. $s_j \in \arg\min\{q(s) \mid s \in \text{Span}\{v_1, \dots, v_j\}\}$,
3. *the function* $\alpha \mapsto q(s_j + \alpha(s_{j+1} - s_j))$ *is monotonically decreasing over* $[0, 1]$ *for all* $j = 1, 2, \dots$.

The properties of Theorem 10 are important in the solution of (3) for the following reasons. First, property 2 is relevant because minimizing $q(s)$ is the end goal. In case the solution of (3) lies interior to the trust region, standard CG will identify it in at most n iterations in exact arithmetic. If the problem is convex and the solution lies outside the trust-region, property 1 of the iterates implies that there will be an iteration j such that $\|s_j\| \leq \Delta$ but $\|s_{j+1}\| > \Delta$. By property 3, we may compute $\alpha \in [0, 1]$ such that $\|s_j + \alpha(s_{j+1} - s_j)\| = \Delta$ and possibly further improve q. If H is indefinite, CG is guaranteed to observe negative curvature along one of its search directions provided Δ is sufficiently large because those search directions, denoted p_j, are *conjugate* with respect to H, i.e, they satisfy $p_i^T H p_j = 0$ if $i \neq j$. Note that this property is not guaranteed for the Lanczos vectors v_i. Thus if CG encounters a situation where $\|s_j\| < \Delta$ and p_j is a direction of negative curvature, $(p_j^T H p_j < 0)$, then p_j may simply be followed to the boundary of the trust region because it is guaranteed to be a descent direction. Further information on the procedure just outlined, traditionally referred to as the *truncated conjugated gradient* method, or the *Steihaug-Toint* strategy, may be found in (Steihaug, 1983) and (Conn et al., 2000, Section 7.5.1).

In contrast, CR aims to reduce the residual norm $\|g + Hs\|$ at each iteration. The following result summarizes properties of CR on a positive definite system that are relevant in a trust-region context.

Theorem 11 (Fong and Saunders, 2012, Theorems 2.3, 2.5) *Assume H is positive definite and s_1, s_2, ... are the iterates generated by CR on $Hs = -g$. Then, for $j = 1, 2, \dots,$*

1. $\|s_{j+1}\| \geq \|s_j\|$,
2. $s_j \in \arg\min \{\|g + Hs\| \mid s \in \text{Span}\{v_1, \ldots, v_j\}\}$,
3. $q(s_j + t(s_{j+1} - s_j)) \leq q(s_j)$ *for* $t \in [0, 1]$,
4. $q(s_{j+1}) < q(s_j)$.

Theorem 11 shows that CR possesses properties closely related to those of CG and suggests that CR may be viable as a trust-region subproblem solver. In the next section, we examine how standard CR may be modified to solve (3).

4.2 Truncated CR

If H is symmetric but not positive definite, q is nonconvex and Algorithm 1 no longer ensures that the search directions p_k are descent directions. Fong (2011, Theorem 2.1.5) shows that $-p_k^T \nabla q(s_k) = p_k^T r_k > 0$ as long as $p_k^T H p_k > 0$, and $\alpha_{k+1} > 0$ as long as $r_k^T H r_k > 0$. Theorem 1 shows that in CR, the search directions p_k are conjugate with respect to H^2, while the residuals r_k are conjugate with respect to H for as long as positive curvature holds. Thus, $r_k^T H r_k$ is a reliable indicator of convexity, but it is possible to observe $p_k^T H p_k \leq 0$ before $r_k^T H r_k \leq 0$, so that we must monitor the sign of both quantities at each iteration.

In Algorithm 1, we see that if $r_k^T H r_k = 0$ then $\beta_k = \zeta_k = 0$, so $p_k = r_k$. This is a special case of $p_k^T H p_k = 0$. But $\zeta_k = 0$ implies $\alpha_{k+1} = 0$, so if nothing is changed in Algorithm 1, an error will occur because $\zeta_{k+1} = r_{k+1}^T H r_{k+1} = 0$ and β_{k+1} will be undefined.

Our strategy is such that as soon as nonpositive curvature is detected, s_k is updated one last time and the algorithm stops.

Algorithm 5 describes CR for trust-region. A separate procedure, described later in Algorithm 6, returns a boolean `terminate`, a search direction p_{k-1} and a steplength α_k. `terminate` is `false` as long as Algorithm 1 continues to apply, and `true` when either nonpositive curvature is discovered or s_k is on the boundary of the trust region. If q is convex, or if a solution of (2) lies inside the trust region and is found before any nonpositive curvature is discovered, Algorithm 5 is equivalent to Algorithm 1.

Algorithm 5 CR for (3) (trust-region version)

Require: H, g, $\Delta > 0$, $\tau_a > 0$, $\tau_r > 0$
Initialize: $k = 0$, $s_0 = 0$, $r_0 = -g$, $u_0 = Hr_0$, $\zeta_0 = r_0^T u_0$, $p_0 = r_0$, $q_0 = u_0$, $\delta_0 = \zeta_0$, $\rho_0 = r_0^T r_0$, $\nu_0 = \sqrt{\rho_0}$, $\mu_0 = \rho_0$
while $\nu_k > \tau_a + \tau_r \|g\|$ **do**
  $k \leftarrow k + 1$
  compute `terminate`, α_k and p_{k-1} using Algorithm 6
  $s_k = s_{k-1} + \alpha_k p_{k-1}$
  if `terminate` **then return** s_k — *q is nonconvex or $\|s_k\| = \Delta$*
  $r_k = r_{k-1} - \alpha_k q_{k-1}$
  $\rho_k = \rho_{k-1} - \alpha_k \zeta_{k-1}$ — *$\rho_k = \|r_k\|^2$*
  $\nu_k = \sqrt{\rho_k}$ — *$\nu_k = \|r_k\|$*
  $u_k = Hr_k$
  $\zeta_k = r_k^T u_k$
  $\beta_k = \zeta_k / \zeta_{k-1}$
  $p_k = r_k + \beta_k p_{k-1}$
  $q_k = u_k + \beta_k q_{k-1}$ — *$q_k = Hp_k$*
  $\mu_k = \rho_k + \beta_k(\mu_{k-1} - \alpha_k \delta_{k-1})$ — *$\mu_k = p_k^T r_k$*
  $\delta_k = \zeta_k + \beta_k^2 \delta_{k-1}$
return s_k

Whether or not we discover nonpositive curvature along p_{k-1} or r_{k-1}, we must monitor the sign of α_k and select a final iterate inside the trust region. Because r_{k-1} is always a descent direction, the direction of best decrease among p_{k-1} and r_{k-1} is followed to either the minimum of the quadratic along that direction, or the boundary of the trust region. To that end, we define $\alpha_{p+} > 0$ and $\alpha_{p-} < 0$ as the step lengths to the boundary in the direction p_{k-1}, and $\alpha_r > 0$ the step length in the direction r_{k-1}. Note that α_{p+}, α_{p-} and α_r

can be computed at a moderate cost because ρ_{k-1} contains the value of $\|r_{k-1}\|^2$ and $\|p_{k-1}\|^2$ can be recurred as in Algorithm 2.

In Algorithm 6, we use the notation "$\alpha \leftarrow$ *value1* **if** *condition* **else** *value2*" to mean that if *condition* evaluates to true, α receives *value1*, and receives *value2* otherwise. Below, line numbers refer to Algorithm 6. We distinguish several cases.

Algorithm 6 Step computation for Algorithm 5

Require: $\Delta > 0$, ζ_{k-1}, δ_{k-1}, ρ_{k-1}, μ_{k-1}, ν_{k-1}, s_{k-1}, r_{k-1}, p_{k-1}, q_{k-1}, $\epsilon > 0$

1: `terminate = false`
2: **Compute** $\alpha_{p+} > 0$, $\alpha_{p-} < 0$ such that $\|s + \alpha p\| = \Delta$, $\alpha \in \{\alpha_{p+}, \alpha_{p-}\}$
3: **if** $|\delta_{k-1}| \leq \epsilon \|p_{k-1}\| \|q_{k-1}\|$ **then** — $p_{k-1}^T H p_{k-1} \approx 0$
4: `terminate = true` — q *is nonconvex*
5: **if** $|\mu_{k-1}| \leq \epsilon \|p_{k-1}\| \nu_{k-1}$ **then** — $p_{k-1}^T r_{k-1} \approx 0$
6: $p_{k-1} \leftarrow r_{k-1}$
7: **Compute** $\alpha_r > 0$ such that $\|s_{k-1} + \alpha_r r_{k-1}\| = \Delta$
8: $\alpha_k \leftarrow \min(\alpha_r, \rho_{k-1}/\zeta_{k-1})$ **if** $\zeta_{k-1} > 0$ **else** α_r
9: **else**
10: **Compute** $\alpha_r > 0$ such that $\|s_{k-1} + \alpha_r r_{k-1}\| = \Delta$
11: **if** $\zeta_{k-1} > 0$ **then**
12: $\alpha_r \leftarrow \min(\alpha_r, \rho_{k-1}/\zeta_{k-1})$
13: $\alpha_k \leftarrow \alpha_{p+}$ **if** $\mu_{k-1} > 0$ **else** α_{p-}
14: $\xi_k = -\alpha_k \mu_{k-1} + \alpha_r \rho_{k-1} - \frac{1}{2}\alpha_r^2 \zeta_{k-1}$
15: **if** $\xi_k > 0$ **then** — $q(s_{k-1} + \alpha_r r_{k-1}) < q(s_{k-1} + \alpha_k p_{k-1})$
16: $p_{k-1} \leftarrow r_{k-1}$, $\alpha_k \leftarrow \alpha_r$
17: **else if** $\delta_{k-1} > 0$ and $\zeta_{k-1} > 0$ **then**
18: $\alpha_k = \zeta_{k-1}/\|q_{k-1}\|^2$
19: **if** $\alpha_k \geq \alpha_{p+}$ **then**
20: `terminate = true` — s_k *is on the boundary of the trust region*
21: $\alpha_k \leftarrow \alpha_{p+}$
22: **else if** $\delta_{k-1} > 0$ and $\zeta_{k-1} < 0$ **then**
23: `terminate = true`
24: $\alpha_k \leftarrow \min(\alpha_{p+}, \mu_{k-1}/\delta_{k-1})$ **if** $\mu_{k-1} > 0$ **else** $\max(\alpha_{p-}, \mu_{k-1}/\delta_{k-1})$
25: **Compute** $\alpha_r > 0$ such that $\|s_{k-1} + \alpha_r r_{k-1}\| = \Delta$
26: $\xi_k = -\alpha_k \mu_{k-1} + \alpha_r \rho_{k-1} + \frac{1}{2}(\alpha_k^2 \delta_{k-1} - \alpha_r^2 \zeta_{k-1})$
27: **if** $\xi_k > 0$ **then** $p_{k-1} \leftarrow r_{k-1}$, $\alpha_k \leftarrow \alpha_r$
28: **else if** $\delta_{k-1} < 0$ and $\zeta_{k-1} > 0$ **then**
29: `terminate = true`
30: $\alpha_k \leftarrow \alpha_{p+}$ **if** $\mu_{k-1} > 0$ **else** α_{p-}
31: **Compute** $\alpha_r > 0$ such that $\|s_{k-1} + \alpha_r r_{k-1}\| = \Delta$
32: $\alpha_r \leftarrow \min(\alpha_r, \rho_{k-1}/\zeta_{k-1})$
33: $\xi_k = -\alpha_k \mu_{k-1} + \alpha_r \rho_{k-1} + \frac{1}{2}(\alpha_k^2 \delta_{k-1} - \alpha_r^2 \zeta_{k-1})$
34: **if** $\xi_k > 0$ **then** $p_{k-1} \leftarrow r_{k-1}$, $\alpha_k \leftarrow \alpha_r$
35: **else if** $\delta_{k-1} < 0$ and $\zeta_{k-1} < 0$ **then**
36: `terminate = true`
37: **Compute** $\alpha_r > 0$ such that $\|s_{k-1} + \alpha_r p_{k-1}\| = \Delta$
38: $\alpha_k \leftarrow \alpha_{p+}$ **if** $\mu_{k-1} > 0$ **else** α_{p-}
39: $\xi_k = -\alpha_k \mu_{k-1} + \alpha_r \rho_{k-1} + \frac{1}{2}(\alpha_k^2 \delta_{k-1} - \alpha_r^2 \zeta_{k-1})$
40: **if** $\xi_k > 0$ **then** $p_{k-1} \leftarrow r_{k-1}$, $\alpha_k \leftarrow \alpha_r$
41: **return** `terminate`, α_k, p_{k-1}

Case I $p_{k-1}^T H p_{k-1} = 0$ (line 3), which covers the case $r_{k-1}^T H r_{k-1} = 0$ by (30): $q(s_{k-1} + \alpha_k p_{k-1}) = q(s_{k-1}) - \alpha_k r_{k-1}^T p_{k-1}$ is linear along p_{k-1}. If H is not positive definite, (13) may not hold. We consider two cases:

1. $p_{k-1}^T r_{k-1} = 0$ (line 5): q is constant along p_{k-1}. We reset $p_{k-1} \leftarrow r_{k-1}$, which is a descent direction. For any steplength α, $q(s_{k-1} + \alpha r_{k-1}) = q(s_{k-1}) - \alpha \|r_{k-1}\|^2 + \frac{1}{2}\alpha^2 r_{k-1}^T H r_{k-1} = q(s_{k-1}) - \alpha \rho_{k-1} + \frac{1}{2}\alpha^2 \zeta_{k-1}$ is stationary when $\alpha \zeta_{k-1} = \rho_{k-1}$. Thus, we set α_k on line 8:
   (a) $\zeta_{k-1} > 0$: q decreases from $\alpha = 0$ to $\alpha_b = \rho_{k-1}/\zeta_{k-1}$. To remain inside the trust-region, we choose $\alpha_k = \min(\alpha_b, \alpha_r)$.
   (b) $\zeta_{k-1} \leq 0$: q is unbounded below along r_{k-1}. We set $\alpha_k = \alpha_r$.

2. $p_{k-1}^T r_{k-1} \neq 0$ (line 10): q is linear but not constant along p_{k-1} and α_k must be such that $\alpha_k p_{k-1}$ is a descent direction. As r_{k-1} is a descent direction, we consider the decrease in q along both directions and choose the best. Call α_{p*} the optimal steplength along p_{k-1}, which is α_{p+} if $\mu_{k-1} = p_{k-1}^T r_{k-1} > 0$, and α_{p-} otherwise because q is linear along p_{k-1}. Call α_{r*} the optimal steplength along r_{k-1}, i.e., $\min(\alpha_b, \alpha_r)$ if $\zeta_{k-1} > 0$ and α_r otherwise. According to (29),

$$\begin{aligned}\xi_k &:= q(s_{k-1} + \alpha_{p*} p_{k-1}) - q(s_{k-1} + \alpha_{r*} r_{k-1}) \\ &= -\alpha_{p*} p_{k-1}^T r_{k-1} + \alpha_{r*} \|r_{k-1}\|^2 + \tfrac{1}{2}(\alpha_{p*}^2 p_{k-1}^T H p_{k-1} - \alpha_{r*}^2 r_{k-1}^T H r_{k-1}) \\ &= -\alpha_{p*} \mu_{k-1} + \alpha_{r*} \rho_{k-1} - \tfrac{1}{2}\alpha_{r*}^2 \zeta_{k-1},\end{aligned}$$

which is computed at line 14. If $\xi_k > 0$, the best decrease occurs along r_{k-1}, which is then chosen as search direction with $\alpha_k = \alpha_{r*}$. Otherwise, we select p_{k-1} with $\alpha_k = \alpha_{p*}$.

Case II $p_k^T H p_k > 0$ and $r_k^T H r_k > 0$ (line 17): q is convex along p_k and r_k, so Corollary 2 applies and standard CR is applied within the trust-region.

Case III in all other situations, $p_{k-1}^T H p_{k-1} > 0$ and $r_{k-1}^T H r_{k-1} < 0$ (line 22), $p_{k-1}^T H p_{k-1} < 0$ and $r_{k-1}^T H r_{k-1} > 0$ (line 28), or $p_{k-1}^T H p_{k-1} < 0$ and $r_{k-1}^T H r_{k-1} < 0$ (line 35): negative curvature is detected. We compute ξ_k to select the direction of best decrease between r_{k-1} and p_{k-1}, and follow it to the minimum of q or the trust-region boundary.

4.3 Main properties

Theorem 12 *In Algorithm 5, the properties of Theorem 1 continue to hold for symmetric positive definite H.*

Proof. As long as $\delta_{i-1} > 0$ and $\mu_{i-1} > 0$, $i = 0, 1, \ldots, k$, Algorithm 5 coincides with Algorithm 1. For H symmetric positive definite, this is ensured for all iterations. □

Numerically, we relax the condition $\delta_{i-1} = 0$ to $|\delta_{i-1}| \leq \epsilon \|p_{i-1}\| \|q_{i-1}\|$, and $\mu_{i-1} = 0$ to $|\mu_{i-1}| \leq \epsilon \|p_{i-1}\| \nu_{i-1}$ for a user-defined tolerance $\epsilon > 0$.

Corollary 2 *If H is symmetric and $\delta_{i-1} > 0$ and $\mu_{i-1} > 0$ for $i = 0, 1, \ldots, k$, then the properties of Theorem 1 continue to hold at those iterations in Algorithm 5.*

Theorem 13 *Let H be symmetric but not necessarily positive definite. Algorithm 5 continues to satisfy (14)–(17) and (21)–(22) for as long as* `terminate` *is* `false`*. In addition, for $k \geq 0$,*

$$\xi_k = q(s_{k-1} + \alpha_k p_{k-1}) - q(s_{k-1} + \alpha_r r_{k-1}), \tag{29}$$

$$\zeta_k = 0 \implies \delta_k = 0. \tag{30}$$

Proof. The proof of (14)–(17) is as in Theorem 2. That of (21)–(22) is as in Theorem 4.

Because q is quadratic, $\nabla q(s_k) = -r_k$, and $\nabla^2 q(s_k) = H$, we have the expansion

$$\begin{aligned}q(s_{k-1} + \alpha_k p_{k-1}) &= q(s_{k-1}) - \alpha_k p_{k-1}^T r_{k-1} + \tfrac{1}{2}\alpha_k^2 p_{k-1}^T H p_{k-1} \\ &= q(s_{k-1}) - \alpha_k \mu_{k-1} + \tfrac{1}{2}\alpha_k^2 \delta_{k-1}.\end{aligned}$$

Likewise,

$$\begin{aligned}q(s_{k-1} + \alpha_r r_{k-1}) &= q(s_{k-1}) - \alpha_r \|r_{k-1}\|^2 + \tfrac{1}{2}\alpha_r^2 r_{k-1}^T H r_{k-1} \\ &= q(s_{k-1}) - \alpha_r \rho_{k-1} + \tfrac{1}{2}\alpha_k^2 \zeta_{k-1},\end{aligned}$$

and (29) follows.

If $\zeta_k = 0$, lines 12 and 13 of Algorithm 5 yield $\beta_k = 0$ and $p_k = r_k$, and $\delta_k = 0$. □

Note that (30) also holds for all previous variants of CR.

4.4 Convergence analysis

In this section, we establish that a step computed by Algorithm 5 satisfies the sufficient-decrease condition (28), and therefore that convergence of Algorithm 4 to a stationary point is guaranteed under standard assumptions on (1).

Both CG and CR begin by performing a search along the steepest-descent direction $-g$. The CG steplength $\alpha_C > 0$ is determined by minimizing

$$q(-\alpha g) = -\alpha \|g\|^2 + \tfrac{1}{2}\alpha^2 g^T H g.$$

The CR steplength $\alpha_M > 0$ is determined by minimizing

$$R(-\alpha g) = \tfrac{1}{2}\|g - \alpha H g\|^2.$$

Both of the above minimizations must take the trust-region bound into account. By definition, the first CG iterate is precisely the Cauchy point. If $g^T H g \leq 0$, both methods step to the boundary and therefore achieve the same decrease. If $g^T H g > 0$, the unconstrained minimizers are

$$\alpha_C = \frac{\|g\|^2}{g^T H g}, \qquad \text{and} \qquad \alpha_M = \frac{g^T H g}{g^T H^2 g}.$$

In addition,

$$q_C := q(-\alpha_C g) = -\tfrac{1}{2}\frac{\|g\|^4}{g^T H g}, \tag{31}$$

$$q_M := q(-\alpha_M g) = \frac{g^T H g}{g^T H^2 g}\left(\tfrac{1}{2}\frac{(g^T H g)^2}{g^T H^2 g} - \|g\|^2\right). \tag{32}$$

Theorem 11 implies that $q_M \leq 0$, which yields $\frac{1}{2}\alpha_M \leq \alpha_C$. Lemma 3 is more precise.

Lemma 3 *Let H be symmetric and g such that $g^T H g > 0$. Then $\alpha_M \leq \alpha_C$.*

Proof. Let $y = Hg$. The result follows from the Cauchy-Schwartz inequality $(g^T y)^2 \leq \|g\|^2 \|y\|^2$. □

Assume first that the CR minimizer along $-g$ lies inside the trust region. Lemma 3 implies that

$$\frac{(g^T H g)^2}{g^T H^2 g} \leq \|g\|^2,$$

so that (32) yields $q_M \leq -\frac{1}{2}\alpha_M \|g\|^2$. But

$$\alpha_M \geq \frac{g^T H g}{(1 + \|H\|)\, g^T H g} = \frac{1}{1 + \|H\|},$$

and therefore

$$q_M \leq -\tfrac{1}{2}\frac{1}{1 + \|H\|}\|g\|^2.$$

Suppose now that the CG minimizer lies on the boundary or outside the trust region. In this case, we reset $\alpha_C = \Delta/\|g\|$ and obtain

$$q_C := q(-\alpha_C g) = \Delta\left(\tfrac{1}{2}\Delta\frac{g^T H g}{\|g\|^2} - \|g\|\right).$$

If the CR minimizer also lies on the boundary or outside the trust region, $q_M = q_C$. If on the other hand $\alpha_M < \Delta/\|g\|$, (32) yields

$$q_M < \frac{\Delta}{\|g\|}\left(\tfrac{1}{2}\frac{\Delta}{\|g\|}\, g^T H g - \|g\|^2\right) = q_C.$$

By Theorem 11, in the event that $g^T H g > 0$ and $\alpha_M < \Delta/\|g\|$, subsequent CR iterations further reduce the value of $q(s)$. Should CR encounter a direction of negative curvature, Algorithm 5 guarantees that no increase in the value of $q(s)$ can result, but that further decrease may occur. Thus in all cases, CR improves upon its first iterate and therefore yields an approximate solution of (3) that satisfies the sufficient-decrease condition (28).

4.5 Numerical results

We use the same benchmark problems as in Section 3.4. We eliminate problems *jimack*, *sbrybnd* and *scosine* because they did not solve within 1.5 hours. In total, we have 109 problems.

Algorithms 4 to 6 are implemented in Julia v0.6. Truncated CG is implemented as described by Steihaug (1983).

Algorithm 4 uses $\epsilon_a = \epsilon_r = 10^{-6}$, $\Delta_0 = 10$, $\eta_1 = 10^{-4}$, $\eta_2 = 0.99$, $\gamma_1 = \gamma_2 = \frac{1}{3}$. Line 15 is changed to $\Delta_{k+1} = 3\Delta$, and line 17 becomes $\Delta_{k+1} = \Delta_k$. We impose a maximum of $10,000$ iterations.

Truncated CG and CR have a maximum number of iterations equal to the number of variables in the problem. Algorithms 5 and 6 set ϵ to the machine precision for the detection of negative curvature. Finally, at iteration k of the trust-region algorithm, the tolerance for the inner iterations is $\tau_a + \tau_r\|g_k\|$ with $\tau_a = 0$ and $\tau_r = \min(0.1, \sqrt{\|g_k\|})$.

Both variants perform equivalently on convex problems in terms of function and gradient evaluations, as illustrated by the profiles of Figure 4. In particular, both variants solve all problems. However, CG turns out to be slightly better in terms of Hessian-vector products.

On nonconvex problems, Figure 5 shows that CR performs better for all types of evaluations, which is a surprise given the CG optimality property of minimizing the quadratic objective as long as it is convex. However, the gap between the performance curves remains small. Both variants solve all problems.

Figures 6 shows that the trend persists on the set of 109 problems, where CR has a more substantial advantage in terms of Hessian-vector products. The CG variant fails on 6 problems, while CR fails on 4 problems.

Overall, although CG is conceptually the best suited iterative method for (3) because of its optimality property, CR performs fewer Hessian-vector products per trust-region subproblem on average, which in turn results in savings in terms of objective and gradient evaluations.

5 Extension to nonlinear least squares

Suppose the objective of (1) is $f(x) = \frac{1}{2}\|F(x)\|^2$, where $F(x) = (f_1(x), \ldots, f_m(x))$. The classical approach of Levenberg (1944) and Marquardt (1963) may be implemented by replacing $q(s)$ in (3) with the Gauss-Newton model $q^{\text{GN}}(s) := \frac{1}{2}\|J(x)s + F(x)\|^2$, where $J(x)$ is the Jacobian of F at x, and solving (1) using Algorithm 4. Thus, at each iteration the subproblem to solve is

$$\underset{s\in\mathbb{R}^n}{\text{minimize}}\ q^{\text{GN}}(s) \quad \text{subject to } \|s\| \le \Delta. \tag{33}$$

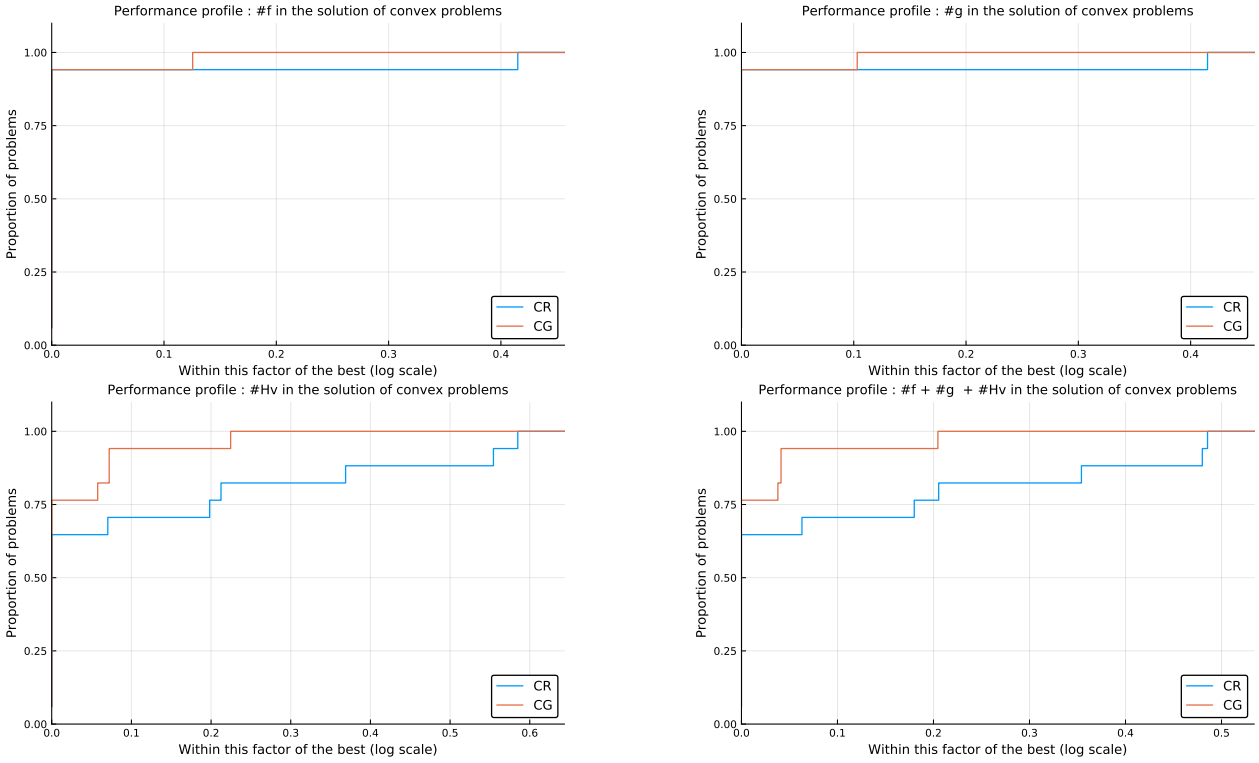

Figure 4: Performance of trust-region CR and CG on 17 convex problems in terms of evaluations of f, g and products with H.

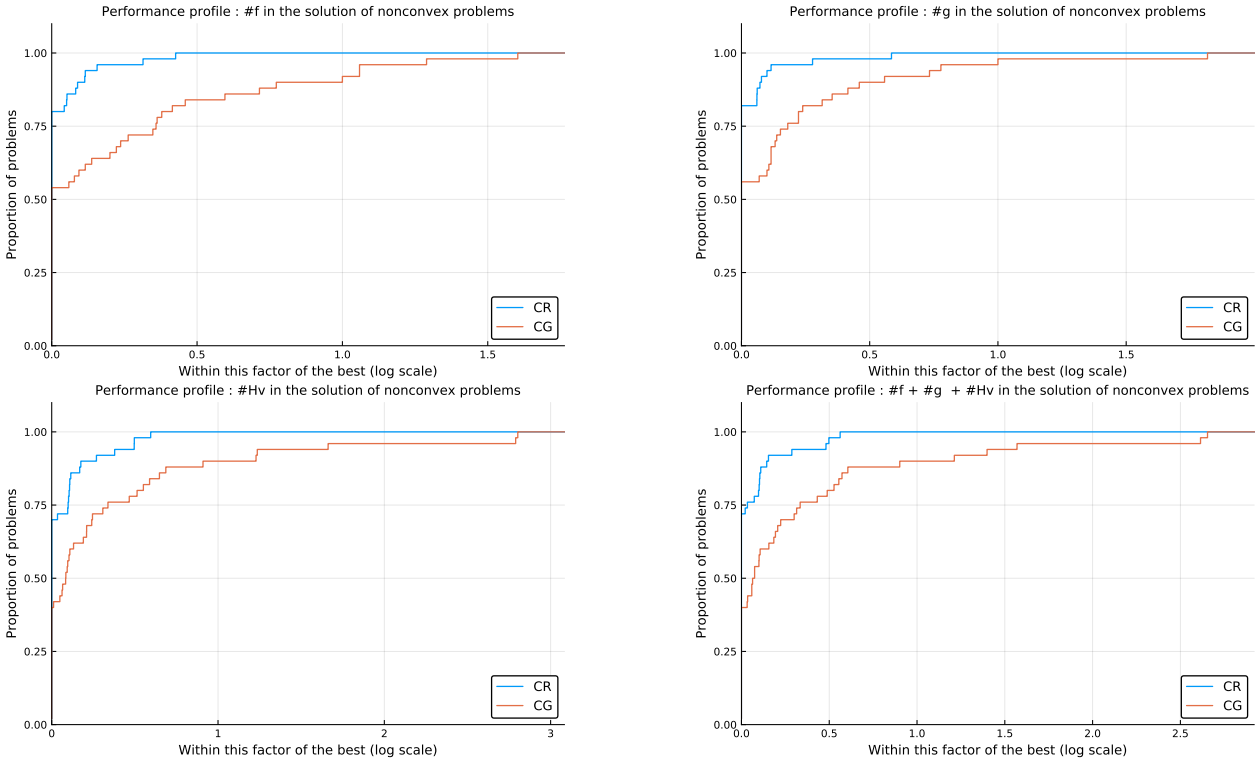

Figure 5: Performance of trust-region CR and CG on 50 nonconvex problems in terms of evaluations of f, g and products with H.

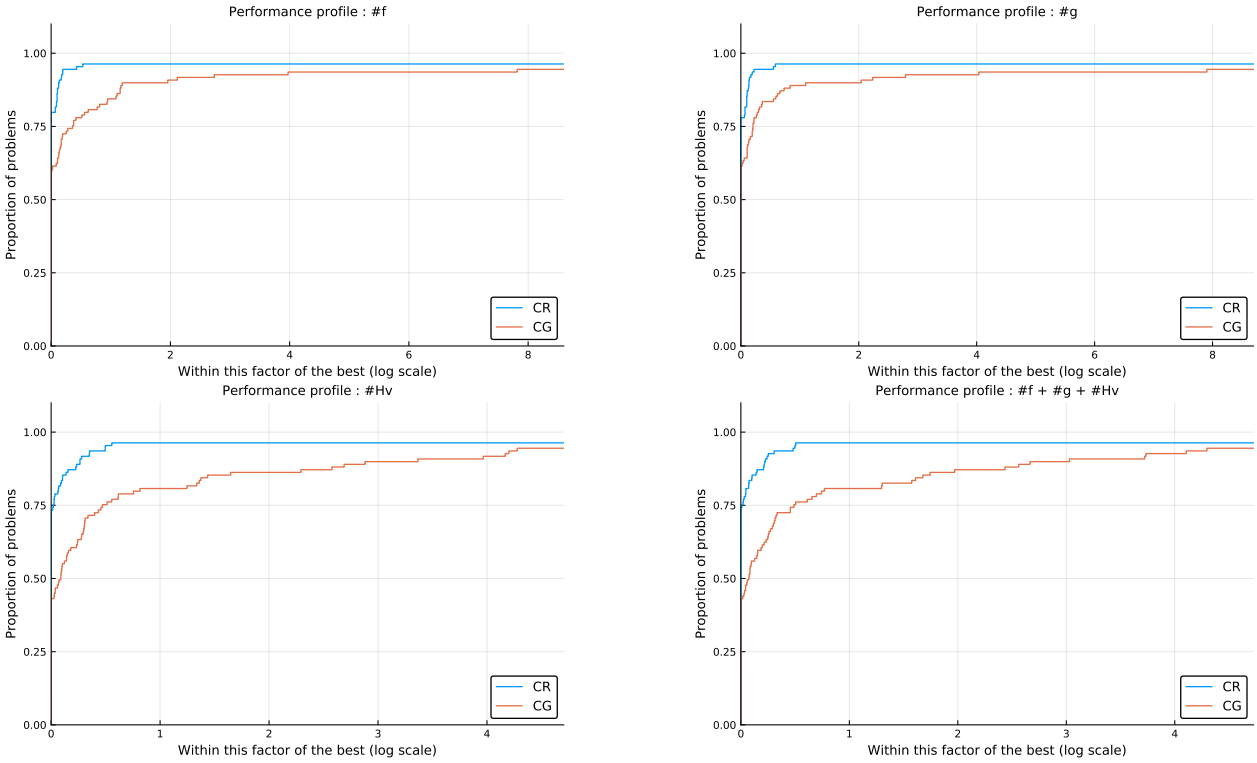

Figure 6: Performance of trust-region CR and CG on 109 problems in terms of evaluations of f, g and products with H.

Both CG and CR can be used to solve (33). Products with $J(x)^T J(x)$ may be dissociated, and this gives rise to variants named CGLS[4] (Hestenes and Stiefel, 1952, Section 10) and CRLS (Fong, 2011) specific to linear least squares.[5]

5.1 CRLS for trust region

For simplicity we use J and F to refer respectively to $J(x)$ and $F(x)$ in the description of the algorithm.

Algorithm 7 is a modification of CRLS adapted to a trust-region context, with radius Δ as input parameter. Note that $\nabla f(x) = J^T F = -\tilde{r}_0$.

At any iteration k, let $\alpha_p > 0$ be the step length to the trust-region boundary in the direction p_{k-1}. Because $J^T J$ is positive semi-definite, the curvature of q^{GN} can only be zero or positive. If $p_{k-1}^T J^T J p_{k-1} = 0$ then $Jp_{k-1} = q_{k-1} = 0$, and q^{GN} is constant in the direction p_{k-1}. In that case, we select $-\nabla q^{\text{GN}}(s_{k-1}) = \tilde{r}_{k-1}$ as the new search direction. We compute the steplength to the minimizer of q^{GN} in the direction $\tilde{r}_{k-1}$, and either step to the minimizer or stop at the trust-region boundary if the minimizer lies outside.

[4]The name CGLS appears to have been coined by Paige and Saunders (1982).

[5]The earliest reference mentioning CRLS, though not under that name, that we are aware of is Björck (1979).

Algorithm 7 CRLS for (33)

Require: J, F, $\Delta > 0$, $\tau_a > 0$, $\tau_r > 0$, $\epsilon > 0$
Initialize: $k = 0$, $s_0 = 0$, $r_0 = -F$, $\tilde{r}_0 = J^T r_0$, $w_0 = J\tilde{r}_0$, $\zeta_0 = w^T w$, $p_0 = \tilde{r}_0$, $q_0 = w_0$
while $\|\tilde{r}_{k-1}\| > \tau_a + \tau_r\|F\|$ **do**
  $k \leftarrow k + 1$
  $v_k = J^T q_{k-1}$
  $\alpha_k = \zeta_{k-1}/\|v\|^2$
  Compute $\alpha_p > 0$ such that $\|s_{k-1} + \alpha_p p_{k-1}\| = \Delta$
  if $\|q_{k-1}\|^2 \leq \epsilon\|p_{k-1}\|\|v_k\|$ **then** *(near) zero curvature detected*
    $p_{k-1} \leftarrow \tilde{r}_{k-1}$
    $\alpha_k \leftarrow \min\left(\alpha_p, \|\tilde{r}_{k-1}\|^2/\zeta_{k-1}\right)$
    $s_k = s_{k-1} + \alpha_k p_{k-1}$
    return s_k
  else
    if $\alpha_k \geq \alpha_p$ **then**
      $\alpha_k \leftarrow \alpha_p$
      $s_k = s_{k-1} + \alpha_k p_{k-1}$
      return s_k
  $s_k = s_{k-1} + \alpha_k p_{k-1}$
  $r_k = r_{k-1} - \alpha_k q_{k-1}$
  $\tilde{r}_k = J^T r_k$
  $w_k = J\tilde{r}_k$
  $\zeta_k = w_k^T w_k$
  $\beta_k = \zeta_k/\zeta_{k-1}$
  $p_k = \tilde{r}_k + \beta_k p_{k-1}$
  $q_k = w_k + \beta_k q_{k-1}$
return s_k

5.2 Numerical results

In exact arithmetic, CGLS and CRLS are equivalent to LSQR (Paige and Saunders, 1982) and LSMR (Fong and Saunders, 2011), respectively, which are based on the Golub and Kahan (1965) process and numerically preferable. LSQR and LSMR require the same number of operator-vector products per iteration as CGLS and CRLS. Björck, Elfving, and Strakoš (1998) analyze several versions of CGLS, notably using recurred residuals, and compare their numerical stability. They conclude that the standard version of CGLS, in which the optimality residual is not recurred, is more stable than LSQR and achieves similar accuracy. Björck and Saunders (2017) perform similar comparisons between CRLS and LSMR and conclude that in its default version, CRLS is inferior to LSMR and is unable to achieve comparable accuracy. They devise a version of CRLS that requires an extra operator-vector product per iteration that is competitive with LSMR. Kloek (2012) performs similar experiments and makes similar observations. For the above reasons, in our experiments, we use LSQR and LSMR as implemented in the package Krylov.jl,[6] with appropriate changes to accomodate a trust-region constraint.

We use nonlinear least-squares problems implemented in Julia from the NLSProblems.jl[7] collection, together with those from CUTEst.jl. We eliminate problems with fewer than 10 variables. We exclude *ba-l16, ba-l21, ba-l49* and *ba-l52* as they require more than 1.5 hour to be solved. We further eliminate *mgh11* because its objective is not $\mathcal{C}^2$. In total, we run 47 problems of size 10 to 123,200.

The trust-region parameters are as in Section 4.5. The maximum number of LSQR and LSMR iterations is $n + m$, $\tau_a = 0$ and $\tau_r = \min(0.1, \sqrt{\|g_k\|})$.

Figure 7 gives the profiles using $\#F$, $\#Ju + \#J^T v$, and $\#F + \#Ju + \#J^T v$. The profiles show that LSQR and LSMR perform equivalently, with a slight advantage for LSMR in terms of residual evaluations. Both methods fail on a single problem: *ba-l73*.

[6] https://github.com/JuliaSmoothOptimizers/Krylov.jl
[7] https://github.com/JuliaSmoothOptimizers/NLSProblems.jl

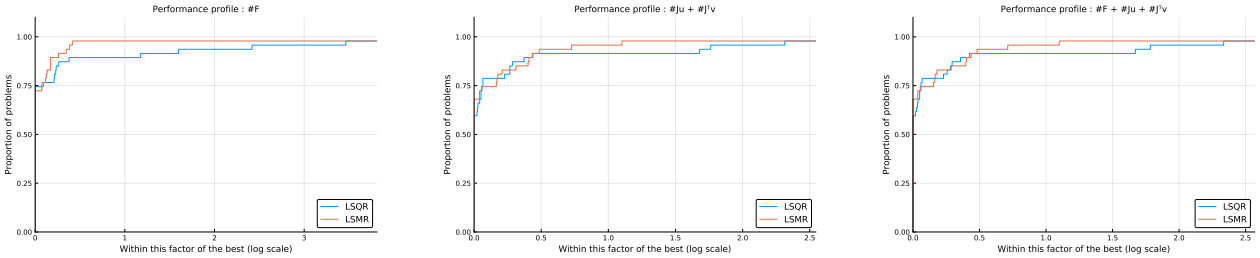

Figure 7: Performance of trust-region LSQR and LSMR on 101 nonlinear least-squares problems in terms of $\#F$, $\#Ju + \#J^T v$, and the sum of both.

6 Discussion

Most implementations of linesearch inexact Newton and trust-region methods with iterative step computation employ the conjugate gradient method. While CG is conceptually the correct method in a trust-region context due to it minimization property of the quadratic model, the CG residual is typically erratic and it is difficult to justify why it should be the method of choice in a linesearch inexact Newton context. When applied to a convex quadratic model, CR shares similarities with CG. Although CR does not minimize the quadratic model at each iteration, the value of the latter decreases monotonically. By construction, CR also produces a monotonic residual, which is in fact minimized at each iteration. Our adaptations of CR to handle nonpositive curvature in both linesearch and trust-region contexts show that CR performs as well as or slightly better than CG, particularly in terms of Hessian-vector products. Our extension to nonlinear least-squares problems via the LSMR implementation of CRLS shows that it behaves comparably to the LSQR implementation of CGLS in a trust-region context.

Our implementations try to save computations and recur a number of quantities such as $p_k^T H p_k$ and $r_k^T H r_k$. It is conceivable that such recurrence formulae are subject to accumulation of rounding errors, especially on ill-conditioned problems, though we have not observed damaging results in our experiments. A finite-precision arithmetic analysis such as that of Björck et al. (1998) is warranted to shed light on the matter.

MINRES (Paige and Saunders, 1975) should be the preferred implementation of CR, and it generalizes CR to indefinite systems. We have not used MINRES in the present research because its implementation is substantially more involved and certain quantities of interest, such as $p_k^T H p_k$, are not readily available. However, corresponding variants of MINRES adapted to linesearch and trust-region contexts would be highly relevant.

We have deliberately left questions of preconditioning aside as they are typically application dependent. A study of the behavior of CR with generic—e.g., diagonal or incomplete Cholesky—preconditioners is left for future work.

The satisfactory performance of CR illustrated in this research raises the question of whether it would also be a worthwhile subproblem solver in constrained optimization, e.g., in projected direction methods for bound-constrained problems such as that of Lin and Moré (1998).

Finally, in trust-region methods, Yuan (2000, Theorem 2) establishes that the decrease in the quadratic model achieved by truncated CG is at least half of that obtained at a global solution of the trust-region subproblem. This is an important result and it is relevant to determine whether a similar result holds for CR.

A Appendix

A.1 Detailed results for the linesearch method

Detailed results for each problem are given in Table A.1 and Table A.2.

Table A.1: Solution of 107 nonlinear problems with linesearch CR.

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
arglina	200	2.000e+02	1.000e+03	5.8e−13	5.7e+01	2	2	2	1
arglinb	200	9.963e+01	8.651e+15	1.2e−01	1.4e+15	2	2	2	1
arglinc	200	1.011e+02	8.353e+15	7.8e−02	1.4e+15	2	2	2	1
arwhead	5000	1.110e−12	1.500e+04	5.9e−06	4.0e+04	6	6	12	5
bdqrtic	5000	2.001e+04	1.129e+06	9.2e−01	1.5e+06	10	10	40	9
box	10000	−1.865e+03	0.000e+00	2.7e−06	5.0e+01	12	5	14	4
boxpower	20000	4.716e−02	1.764e+05	2.1e−02	1.7e+05	8	7	18	6
brownal	200	2.037e−08	2.010e+06	2.8e−02	5.7e+05	2	2	2	1
broydn7d	5000	1.854e+03	1.760e+04	7.6e−04	1.1e+03	1977	1782	7610	1781
brybnd	5000	2.050e−06	1.249e+05	6.5e−03	7.8e+03	9	9	92	8
chainwoo	4000	2.388e+03	1.445e+07	1.7e−01	4.2e+05	5291	3454	85956	3453
chnrosnb	50	2.206e−06	7.636e+03	3.5e−03	3.6e+03	70	48	984	47
chnrosnb_mod	100	1.115e−05	1.764e+04	6.1e−03	6.4e+03	200	112	1195	111
chnrsnbm	50	2.275e−09	8.633e+03	2.7e−04	4.4e+03	126	70	1311	69
clplatea	5041	−1.259e−02	0.000e+00	1.1e−06	1.0e−01	8	7	646	6
clplateb	5041	−5.095e−03	0.000e+00	3.6e−07	1.2e−02	4	4	411	3
clplatec	5041	−5.021e−03	0.000e+00	7.3e−07	9.9e−02	5	5	10961	4
cosine	10000	−9.999e+03	8.775e+03	8.7e−07	7.2e+01	8	7	19	6
cragglvy	5000	1.688e+03	2.749e+06	1.1e−01	2.8e+05	13	13	92	12
curly10	10000	−1.003e+06	−6.306e−01	1.3e−04	1.3e+02	21	19	86127	18
curly20	10000	−1.003e+06	−1.344e+00	2.4e−05	3.0e+02	25	21	99683	20
curly30	10000	−1.003e+06	−2.190e+00	7.6e−05	5.1e+02	28	22	106049	21
deconvu	63	1.825e−07	1.104e+02	2.5e−05	1.1e+02	14	13	191	12
dixmaana	3000	1.000e+00	2.850e+04	2.6e−04	1.2e+03	10	8	20	7
dixmaanb	3000	1.000e+00	4.724e+04	2.1e−04	2.0e+03	8	8	14	7
dixmaanc	3000	1.000e+00	8.248e+04	2.8e−04	3.7e+03	9	9	16	8
dixmaand	3000	1.000e+00	1.586e+05	2.2e−03	7.6e+03	10	10	18	9
dixmaane	3000	1.000e+00	2.850e+04	2.6e−04	1.2e+03	10	8	20	7
dixmaanf	3000	1.000e+00	4.104e+04	2.7e−04	1.9e+03	12	12	252	11
dixmaang	3000	1.000e+00	7.607e+04	1.3e−03	3.6e+03	12	12	152	11
dixmaanh	3000	1.001e+00	1.517e+05	3.6e−03	7.4e+03	12	12	64	11
dixmaani	3000	1.000e+00	2.002e+04	3.3e−05	1.0e+03	11	11	1260	10
dixmaanj	3000	1.000e+00	3.900e+04	4.7e−04	1.8e+03	13	13	166	12
dixmaank	3000	1.000e+00	7.400e+04	9.7e−04	3.6e+03	13	13	122	12
dixmaanl	3000	1.000e+00	1.496e+05	2.4e−03	7.4e+03	13	13	86	12
dixmaanm	3000	1.001e+00	9.358e+03	3.1e−04	4.4e+02	9	9	524	8
dixmaann	3000	1.000e+00	2.018e+04	4.9e−04	1.0e+03	13	13	290	12
dixmaano	3000	1.001e+00	3.635e+04	1.1e−03	2.0e+03	13	13	198	12
dixmaanp	3000	1.002e+00	7.128e+04	2.5e−03	4.0e+03	13	13	136	12
dixon3dq	10000	9.661e−04	8.000e+00	1.8e−05	5.7e+00	10001	10001	42496	10000
dqdrtic	5000	3.143e−05	9.041e+06	1.1e−02	8.5e+04	5	5	16	4
dqrtic	5000	4.137e+09	6.241e+17	8.1e+06	1.3e+13	13	13	52	12
edensch	2000	1.200e+04	7.358e+06	2.8e−02	1.0e+05	14	14	48	13
eg2	5000	5.549e+03	2.949e+05	1.7e−03	8.8e+03	9	9	34	8
engval1	5000	5.549e+03	2.949e+05	1.7e−03	8.8e+03	9	9	34	8
errinros	50	4.022e+01	1.102e+05	4.1e−02	1.2e+05	32	27	341	26
errinros_mod	100	7.862e+01	3.140e+05	1.9e−01	1.9e+05	22	18	93	17
errinrsm	50	3.873e+01	1.529e+05	1.1e−01	1.3e+05	25	20	209	19
extrosnb	1000	8.403e−03	3.996e+05	3.8e−02	3.8e+04	48	30	216	29
fletbv3m	5000	−2.326e+05	1.982e+02	3.2e−05	4.4e+01	293	292	528	291
fletcbv2	5000	−5.003e−01	−5.003e−01	3.7e−06	4.4e−06	10001	10001	32266	10000
fletcbv3	5000	−2.286e+08	1.982e+02	5.0e+01	4.4e+01	10001	10001	13158	10000
fletcbv3_mod	100	−8.684e−02	−1.879e−02	2.8e−03	1.6e−03	10001	10001	10001	10000
fletchbv	5000	−1.916e+19	−2.302e+11	4.5e+09	2.8e+09	15817	10001	29997	10000
fletchcr	1000	7.199e−11	9.990e+02	2.2e−05	6.3e+01	2726	1634	37090	1633
fminsrf2	5625	1.000e+00	2.846e+01	2.5e−06	3.3e−01	10133	10001	27849	10000

Continued on next page

Table A.1 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
fminsurf	5625	1.000e+00	2.859e+01	2.8e−07	3.3e−01	227	38	16029	37
freuroth	5000	6.082e+05	5.049e+06	4.4e−03	5.5e+04	28	10	43	9
genhumps	5000	1.735e−07	1.281e+08	3.0e−04	6.0e+03	7947	7530	28065	7529
genrose	500	1.000e+00	1.870e+03	2.2e−06	3.0e+02	915	457	7380	456
genrose_nash	100	1.000e+00	4.041e+02	8.7e−05	1.3e+02	196	110	790	109
hilbertb	10	2.216e−14	5.102e+02	6.8e−07	1.1e+02	6	6	12	5
indef	5000	−2.499e+07	4.603e+03	7.1e+01	8.0e+01	10001	10001	20062	10000
liarwhd	5000	1.671e−04	2.925e+06	2.6e−02	4.8e+05	13	13	36	12
mancino	100	1.059e−01	1.103e+12	9.1e+02	2.9e+09	6	6	12	5
modbeale	20000	1.695e−02	1.264e+07	2.4e−01	3.1e+05	14	13	150	12
morebv	5000	3.637e−11	1.597e−07	9.5e−08	8.0e−04	3	3	898	2
ncb20	5010	−1.463e+03	1.000e+04	2.6e−04	2.8e+02	238	218	1561	217
ncb20b	5000	7.351e+03	1.000e+04	7.2e−05	2.8e+02	14	10	203	9
noncvxu2	5000	1.182e+04	3.235e+11	2.6e+00	3.3e+06	3291	3287	13292	3286
noncvxun	5000	1.217e+04	3.335e+11	2.7e+00	3.6e+06	2343	2337	9987	2336
nondia	5000	2.570e−04	2.000e+06	1.3e−02	2.0e+06	3	3	4	2
nondquar	5000	4.830e−03	5.006e+03	1.9e−02	2.0e+04	17	15	112	14
osborneb	11	4.014e−02	3.166e+00	7.2e−06	6.5e+00	118	113	689	112
oscigrad	100000	1.062e−03	6.121e+08	2.7e+02	2.2e+09	10	10	66	9
oscipath	10	1.000e+00	1.000e+00	3.7e−06	1.0e+00	10001	10001	75635	10000
parkch	15	NaN	2.150e+03	NaN	2.5e+04	24	14	97	13
penalty1	1000	3.933e+08	1.114e+17	1.1e+07	2.4e+13	13	13	24	12
penalty2	200	4.712e+13	4.712e+13	3.6e+00	1.6e+07	12	12	232	11
penalty3	200	1.016e−03	1.584e+09	7.9e−01	2.2e+06	49	20	115	19
powellsg	5000	5.539e−04	2.688e+05	4.8e−03	1.6e+04	14	14	90	13
power	10000	9.819e+07	2.501e+15	8.7e+07	1.2e+14	13	13	76	12
quartc	5000	4.137e+09	6.241e+17	8.1e+06	1.3e+13	13	13	52	12
schmvett	5000	−1.499e+04	−1.429e+04	3.2e−05	7.5e+01	7	7	80	6
scosine	100	−4.591e+01	8.688e+01	9.1e+04	8.1e+02	10225	10001	25764	10000
scurly10	10000	3.636e+24	7.006e+31	8.3e+23	1.3e+30	13	13	116	12
scurly20	10000	4.752e+25	9.031e+32	1.1e+25	1.6e+31	13	13	108	12
scurly30	10000	1.914e+26	4.163e+33	4.7e+25	7.5e+31	13	13	104	12
sensors	100	−2.109e+03	−5.648e+01	6.2e−09	7.1e+01	33	18	49	17
sinquad	5000	−6.749e+06	6.561e−01	4.6e−03	5.1e+03	53	18	62	17
sparsine	5000	1.022e−01	5.173e+07	2.3e+00	3.0e+06	9	9	429	8
sparsqur	10000	9.566e−02	1.406e+07	7.3e−01	1.2e+06	13	13	70	12
spmsrtls	4999	1.341e−11	4.141e+03	1.0e−06	7.7e+01	16	15	554	14
srosenbr	5000	1.831e−08	4.850e+04	1.2e−04	1.1e+04	8	8	18	7
ssbrybnd	5000	6.737e−04	1.249e+05	3.0e−01	9.0e+05	18	17	16568	16
stratec	50	0.000e+00	0.000e+00	0.0e+00	0.0e+00	1	1	0	0
testquad	5000	4.741e+01	1.250e+09	1.5e+01	4.4e+07	7	7	748	6
tointgor	50	1.374e+03	5.074e+03	1.9e−05	6.0e+02	9	9	234	8
tointgss	5000	1.000e+01	4.499e+04	2.1e−04	4.2e+02	4	4	6	3
tointpsp	50	2.256e+02	1.828e+03	5.6e−05	1.1e+02	53	20	226	19
tointqor	50	1.175e+03	2.335e+03	1.9e−04	2.1e+02	6	6	66	5
tquartic	5000	2.749e−15	8.100e−01	1.5e−09	1.8e+00	43	33	71	32
tridia	5000	2.749e−15	8.100e−01	1.5e−09	1.8e+00	43	33	71	32
vardim	200	1.149e+08	3.257e+16	7.3e+09	1.6e+16	13	13	24	12
vareigvl	50	1.052e−13	1.126e+02	1.1e−06	4.2e+01	7	7	59	6
watson	12	1.602e−07	3.000e+01	4.8e−06	2.1e+02	14	14	112	13
woods	4000	7.877e+03	1.919e+07	1.1e−02	5.2e+05	10	10	38	9

Table A.2: Solution of 107 nonlinear problems with linesearch CG.

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
arglina	200	2.000e+02	1.000e+03	5.8e−13	5.7e+01	2	2	2	1
arglinb	200	9.963e+01	8.651e+15	1.2e−01	1.4e+15	2	2	2	1
arglinc	200	1.011e+02	8.353e+15	4.1e−01	1.4e+15	2	2	2	1
arwhead	5000	1.110e−12	1.500e+04	5.9e−06	4.0e+04	6	6	12	5
bdqrtic	5000	2.001e+04	1.129e+06	8.4e−01	1.5e+06	10	10	40	9
box	10000	−1.865e+03	0.000e+00	1.5e−06	5.0e+01	12	5	14	4
boxpower	20000	4.716e−02	1.764e+05	2.0e−02	1.7e+05	8	7	20	6

Continued on next page

Table A.2 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
brownal	200	2.037e−08	2.010e+06	2.8e−02	5.7e+05	2	2	2	1
broydn7d	5000	1.875e+03	1.760e+04	4.2e−05	1.1e+03	1040	300	4326	299
brybnd	5000	1.510e−06	1.249e+05	7.2e−03	7.8e+03	8	8	78	7
chainwoo	4000	1.535e+02	1.445e+07	1.0e−01	4.2e+05	5324	1450	79310	1449
chnrosnb	50	1.877e−08	7.636e+03	1.9e−03	3.6e+03	72	45	978	44
chnrosnb_mod	100	6.582e−08	1.764e+04	2.3e−03	6.4e+03	213	99	1192	98
chnrsnbm	50	4.301e−09	8.633e+03	2.1e−03	4.4e+03	100	56	1176	55
clplatea	5041	−1.259e−02	0.000e+00	3.0e−07	1.0e−01	8	7	773	6
clplateb	5041	−5.095e−03	0.000e+00	2.2e−07	1.2e−02	4	4	456	3
clplatec	5041	−5.021e−03	0.000e+00	2.7e−08	9.9e−02	5	5	9370	4
cosine	10000	−9.999e+03	8.775e+03	9.1e−07	7.2e+01	8	7	19	6
cragglvy	5000	1.688e+03	2.749e+06	1.0e−01	2.8e+05	13	13	102	12
curly10	10000	−1.003e+06	−6.306e−01	6.9e−05	1.3e+02	27	14	116337	13
curly20	10000	−1.003e+06	−1.344e+00	1.2e−04	3.0e+02	29	15	138593	14
curly30	10000	−1.003e+06	−2.190e+00	3.2e−04	5.1e+02	35	15	127354	14
deconvu	63	1.773e−07	1.104e+02	7.4e−06	1.1e+02	23	15	254	14
dixmaana	3000	1.000e+00	2.850e+04	2.6e−04	1.2e+03	11	9	24	8
dixmaanb	3000	1.000e+00	4.724e+04	1.9e−04	2.0e+03	8	8	14	7
dixmaanc	3000	1.000e+00	8.248e+04	2.8e−04	3.7e+03	9	9	16	8
dixmaand	3000	1.000e+00	1.586e+05	2.1e−03	7.6e+03	10	10	18	9
dixmaane	3000	1.000e+00	2.850e+04	2.6e−04	1.2e+03	11	9	24	8
dixmaanf	3000	1.000e+00	4.104e+04	1.7e−03	1.9e+03	14	12	444	11
dixmaang	3000	1.000e+00	7.607e+04	1.8e−03	3.6e+03	15	13	202	12
dixmaanh	3000	1.000e+00	1.517e+05	4.1e−03	7.4e+03	18	13	225	12
dixmaani	3000	1.000e+00	2.002e+04	4.5e−05	1.0e+03	11	11	3696	10
dixmaanj	3000	1.000e+00	3.900e+04	1.4e−03	1.8e+03	12	12	128	11
dixmaank	3000	1.001e+00	7.400e+04	3.3e−03	3.6e+03	12	12	94	11
dixmaanl	3000	1.000e+00	1.496e+05	1.9e−03	7.4e+03	13	13	100	12
dixmaanm	3000	1.000e+00	9.358e+03	3.0e−04	4.4e+02	9	9	1382	8
dixmaann	3000	1.001e+00	2.018e+04	9.8e−04	1.0e+03	17	13	445	12
dixmaano	3000	1.000e+00	3.635e+04	1.9e−03	2.0e+03	20	15	395	14
dixmaanp	3000	1.000e+00	7.128e+04	1.9e−03	4.0e+03	17	15	326	14
dixon3dq	10000	5.207e−10	8.000e+00	6.4e−06	5.7e+00	6	6	21578	5
dqdrtic	5000	4.340e−09	9.041e+06	6.6e−04	8.5e+04	5	5	16	4
dqrtic	5000	2.888e+09	6.241e+17	6.8e+06	1.3e+13	13	13	56	12
edensch	2000	1.200e+04	7.358e+06	8.0e−03	1.0e+05	14	13	46	12
eg2	5000	5.549e+03	2.949e+05	1.5e−03	8.8e+03	9	9	34	8
engval1	5000	5.549e+03	2.949e+05	1.5e−03	8.8e+03	9	9	34	8
errinros	50	4.033e+01	1.102e+05	5.5e−02	1.2e+05	48	26	361	25
errinros_mod	100	7.838e+01	3.140e+05	5.8e−02	1.9e+05	41	25	148	24
errinrsm	50	3.855e+01	1.529e+05	5.6e−02	1.3e+05	34	24	280	23
extrosnb	1000	8.625e−03	3.996e+05	3.3e−02	3.8e+04	37	26	196	25
fletbv3m	5000	−2.492e+05	1.982e+02	2.7e−05	4.4e+01	66	50	119	49
fletcbv2	5000	−5.003e−01	−5.003e−01	1.7e−08	4.4e−06	2	2	9884	1
fletcbv3	5000	−1.302e+09	1.982e+02	3.3e+01	4.4e+01	10001	10001	15043	10000
fletcbv3_mod	100	−8.701e−02	−1.879e−02	2.8e−03	1.6e−03	10001	10001	10001	10000
fletchbv	5000	−3.044e+22	−2.302e+11	1.3e+10	2.8e+09	14970	10001	20057	10000
fletchcr	1000	4.824e−13	9.990e+02	2.3e−05	6.3e+01	1758	1488	33547	1487
fminsrf2	5625	1.000e+00	2.846e+01	9.7e−09	3.3e−01	465	47	101670	46
fminsurf	5625	1.000e+00	2.859e+01	5.3e−08	3.3e−01	349	43	72166	42
freuroth	5000	6.082e+05	5.049e+06	4.2e−03	5.5e+04	25	11	53	10
genhumps	5000	2.405e−06	1.281e+08	1.2e−03	6.0e+03	5593	4720	18566	4719
genrose	500	1.000e+00	1.870e+03	2.3e−04	3.0e+02	829	281	6893	280
genrose_nash	100	1.000e+00	4.041e+02	4.2e−06	1.3e+02	209	71	742	70
hilbertb	10	3.437e−14	5.102e+02	8.4e−07	1.1e+02	6	6	12	5
indef	5000	−2.322e+09	4.603e+03	3.2e+02	8.0e+01	10348	10001	29124	10000
liarwhd	5000	1.626e−04	2.925e+06	2.6e−02	4.8e+05	13	13	36	12
mancino	100	1.101e−01	1.103e+12	9.2e+02	2.9e+09	6	6	13	5
modbeale	20000	1.299e−03	1.264e+07	7.3e−02	3.1e+05	9	9	106	8
morebv	5000	1.053e−11	1.597e−07	1.0e−07	8.0e−04	3	3	5380	2
ncb20	5010	−1.458e+03	1.000e+04	9.4e−06	2.8e+02	76	38	734	37
ncb20b	5000	7.351e+03	1.000e+04	2.6e−04	2.8e+02	65	20	1992	19
noncvxu2	5000	1.161e+04	3.235e+11	1.9e+00	3.3e+06	2755	941	8218	940
noncvxun	5000	1.163e+04	3.335e+11	7.1e−01	3.6e+06	3127	942	11348	941

Continued on next page

Table A.2 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
nondia	5000	2.570e−04	2.000e+06	1.3e−02	2.0e+06	3	3	4	2
nondquar	5000	1.301e−03	5.006e+03	1.3e−02	2.0e+04	24	18	196	17
osborneb	11	4.014e−02	3.166e+00	1.5e−08	6.5e+00	41	22	293	21
oscigrad	100000	1.146e−03	6.121e+08	3.9e+02	2.2e+09	10	10	64	9
oscipath	10	9.999e−01	1.000e+00	1.5e−03	1.0e+00	17132	10001	162076	10000
parkch	15	1.624e+03	2.150e+03	3.6e−03	2.5e+04	27	22	280	21
penalty1	1000	3.933e+08	1.114e+17	1.1e+07	2.4e+13	13	13	24	12
penalty2	200	4.712e+13	4.712e+13	1.3e+00	1.6e+07	12	12	250	11
penalty3	200	1.011e−03	1.584e+09	7.3e−01	2.2e+06	43	18	118	17
powellsg	5000	2.065e−03	2.688e+05	1.3e−02	1.6e+04	13	13	86	12
power	10000	4.738e+07	2.501e+15	8.0e+07	1.2e+14	13	13	80	12
quartc	5000	2.888e+09	6.241e+17	6.8e+06	1.3e+13	13	13	56	12
schmvett	5000	−1.499e+04	−1.429e+04	2.8e−05	7.5e+01	7	7	78	6
scosine	100	−8.359e+01	8.688e+01	2.7e+03	8.1e+02	10186	10001	1232544	10000
scurly10	10000	1.409e+24	7.006e+31	7.2e+23	1.3e+30	13	13	130	12
scurly20	10000	1.847e+25	9.031e+32	9.1e+24	1.6e+31	13	13	124	12
scurly30	10000	1.006e+26	4.163e+33	4.3e+25	7.5e+31	13	13	102	12
sensors	100	−2.109e+03	−5.648e+01	1.8e−11	7.1e+01	28	16	49	15
sinquad	5000	−6.749e+06	6.561e−01	4.3e−03	5.1e+03	37	16	54	15
sparsine	5000	5.750e−03	5.173e+07	1.1e+00	3.0e+06	47	23	3018	22
sparsqur	10000	6.964e−02	1.406e+07	6.5e−01	1.2e+06	13	13	64	12
spmsrtls	4999	2.706e−10	4.141e+03	1.8e−05	7.7e+01	47	17	981	16
srosenbr	5000	1.916e−08	4.850e+04	1.2e−04	1.1e+04	8	8	18	7
ssbrybnd	5000	3.273e−06	1.249e+05	1.7e−01	9.0e+05	349	76	35100	75
stratec	50	0.000e+00	0.000e+00	0.0e+00	0.0e+00	1	1	0	0
testquad	5000	1.768e+00	1.250e+09	1.5e+01	4.4e+07	7	7	1020	6
tointgor	50	1.374e+03	5.074e+03	7.5e−05	6.0e+02	8	8	218	7
tointgss	5000	1.000e+01	4.499e+04	3.8e−05	4.2e+02	4	4	6	3
tointpsp	50	2.256e+02	1.828e+03	1.3e−05	1.1e+02	31	14	182	13
tointqor	50	1.175e+03	2.335e+03	1.2e−05	2.1e+02	7	7	86	6
tquartic	5000	1.474e−15	8.100e−01	1.1e−09	1.8e+00	30	22	58	21
tridia	5000	1.474e−15	8.100e−01	1.1e−09	1.8e+00	30	22	58	21
vardim	200	1.149e+08	3.257e+16	7.3e+09	1.6e+16	13	13	24	12
vareigvl	50	9.344e−13	1.126e+02	4.7e−06	4.2e+01	8	8	66	7
watson	12	1.597e−07	3.000e+01	1.2e−05	2.1e+02	13	13	100	12
woods	4000	7.877e+03	1.919e+07	1.8e−01	5.2e+05	10	10	37	9

A.2 Detailed results for the trust-region method

Detailed results on individual problems are given in Table A.3 and Table A.4.

Table A.3: Solution of 109 nonlinear problems with trust-region CR.

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
arglina	200	$2.000e+02$	$1.000e+03$	$1.8e-13$	$5.7e+01$	3	3	4	2
arglinb	200	$1.011e+02$	$8.353e+15$	$3.1e-03$	$1.4e+15$	3	3	4	2
arglinc	200	$1.011e+02$	$8.353e+15$	$3.1e-03$	$1.4e+15$	3	3	4	2
arwhead	5000	$1.110e-12$	$1.500e+04$	$5.9e-06$	$4.0e+04$	6	6	11	5
bdqrtic	5000	$2.001e+04$	$1.129e+06$	$9.2e-01$	$1.5e+06$	10	10	29	9
box	10000	$-1.865e+03$	$0.000e+00$	$3.5e-07$	$5.0e+01$	7	7	17	6
boxpower	20000	$4.736e-02$	$1.764e+05$	$6.9e-02$	$1.7e+05$	11	6	22	10
brownal	200	$2.037e-08$	$2.010e+06$	$2.8e-02$	$5.7e+05$	2	2	2	1
broydn7d	5000	$1.846e+03$	$1.760e+04$	$9.8e-04$	$1.1e+03$	428	422	2239	427
brybnd	5000	$6.074e-08$	$1.249e+05$	$1.1e-03$	$7.8e+03$	11	11	67	10
chainwoo	4000	$2.799e+03$	$1.445e+07$	$3.7e-01$	$4.2e+05$	5656	3545	68977	5655
chnrosnb	50	$3.192e-06$	$7.636e+03$	$2.6e-03$	$3.6e+03$	95	53	973	94
chnrosnb_mod	100	$2.487e-08$	$1.764e+04$	$6.7e-04$	$6.4e+03$	184	122	1901	183
chnrsnbm	50	$4.924e-08$	$8.633e+03$	$9.7e-04$	$4.4e+03$	101	68	965	100
clplatea	5041	$-1.259e-02$	$0.000e+00$	$5.2e-07$	$1.0e-01$	21	15	681	20
clplateb	5041	$-5.095e-03$	$0.000e+00$	$3.6e-07$	$1.2e-02$	4	4	414	3
clplatec	5041	$-5.021e-03$	$0.000e+00$	$7.3e-07$	$9.9e-02$	5	5	10965	4

Continued on next page

Table A.3 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
cosine	10000	$-9.999e+03$	$8.775e+03$	$3.0e-05$	$7.2e+01$	12	12	26	11
agglvy	5000	$1.688e+03$	$2.749e+06$	$1.1e-01$	$2.8e+05$	13	13	57	12
curly10	10000	$-1.003e+06$	$-6.306e-01$	$3.9e-05$	$1.3e+02$	24	21	48883	23
curly20	10000	$-1.003e+06$	$-1.344e+00$	$2.6e-04$	$3.0e+02$	24	21	47039	23
curly30	10000	$-1.003e+06$	$-2.190e+00$	$7.5e-05$	$5.1e+02$	26	22	48385	25
deconvu	63	$5.824e-08$	$1.104e+02$	$1.6e-06$	$1.1e+02$	20	13	257	19
dixmaana	3000	$1.000e+00$	$2.850e+04$	$5.5e-04$	$1.2e+03$	9	9	19	8
dixmaanb	3000	$1.000e+00$	$4.724e+04$	$1.8e-03$	$2.0e+03$	8	8	14	7
dixmaanc	3000	$1.000e+00$	$8.248e+04$	$1.6e-04$	$3.7e+03$	10	10	18	9
dixmaand	3000	$1.000e+00$	$1.586e+05$	$5.0e-03$	$7.6e+03$	10	10	18	9
dixmaane	3000	$1.000e+00$	$2.209e+04$	$3.8e-05$	$1.1e+03$	11	11	204	10
dixmaanf	3000	$1.000e+00$	$4.104e+04$	$1.1e-03$	$1.9e+03$	12	12	91	11
dixmaang	3000	$1.001e+00$	$7.607e+04$	$3.2e-03$	$3.6e+03$	12	12	46	11
dixmaanh	3000	$1.001e+00$	$1.517e+05$	$2.0e-03$	$7.4e+03$	13	13	60	12
dixmaani	3000	$1.000e+00$	$2.002e+04$	$2.4e-04$	$1.0e+03$	11	11	297	10
dixmaanj	3000	$1.000e+00$	$3.900e+04$	$9.4e-04$	$1.8e+03$	13	13	79	12
dixmaank	3000	$1.001e+00$	$7.400e+04$	$2.5e-03$	$3.6e+03$	13	13	56	12
dixmaanl	3000	$1.001e+00$	$1.496e+05$	$5.7e-03$	$7.4e+03$	13	13	46	12
dixmaanm	3000	$1.000e+00$	$9.358e+03$	$6.1e-05$	$4.4e+02$	10	10	507	9
dixmaann	3000	$1.001e+00$	$2.018e+04$	$9.3e-04$	$1.0e+03$	13	13	123	12
dixmaano	3000	$1.000e+00$	$3.635e+04$	$6.3e-04$	$2.0e+03$	14	14	134	13
dixmaanp	3000	$1.001e+00$	$7.128e+04$	$1.7e-03$	$4.0e+03$	14	14	89	13
dixon3dq	10000	$1.195e-04$	$8.000e+00$	$4.9e-06$	$5.7e+00$	6	6	10799	5
dqdrtic	5000	$6.563e-06$	$9.041e+06$	$5.1e-03$	$8.5e+04$	8	8	18	7
dqrtic	5000	$1.741e+09$	$6.241e+17$	$4.2e+06$	$1.3e+13$	21	21	54	20
edensch	2000	$1.200e+04$	$7.358e+06$	$2.4e-02$	$1.0e+05$	13	13	30	12
eg2	1000	$-9.989e+02$	$-8.406e+02$	$6.0e-09$	$5.4e+02$	4	4	6	3
engval1	5000	$5.549e+03$	$2.949e+05$	$7.9e-04$	$8.8e+03$	11	11	29	10
errinros	50	$4.034e+01$	$1.102e+05$	$1.1e-01$	$1.2e+05$	42	26	382	41
errinros_mod	100	$7.858e+01$	$3.140e+05$	$1.4e-01$	$1.9e+05$	30	18	220	29
errinrsm	50	$3.866e+01$	$1.529e+05$	$8.4e-02$	$1.3e+05$	36	20	300	35
extrosnb	1000	$4.800e-03$	$3.996e+05$	$1.5e-02$	$3.8e+04$	61	34	343	60
fletbv3m	5000	$-2.394e+05$	$1.982e+02$	$5.0e-06$	$4.4e+01$	16	12	30	15
fletcbv2	5000	$-5.003e-01$	$-5.003e-01$	$1.7e-08$	$4.4e-06$	2	2	4843	1
fletcbv3	5000	$-1.941e+09$	$1.982e+02$	$3.6e+01$	$4.4e+01$	10001	9728	25199	10000
fletcbv3_mod	100	$-2.047e+00$	$-1.879e-02$	$1.6e-08$	$1.6e-03$	43	39	103	42
fletchbv	5000	$-2.372e+17$	$-2.302e+11$	$3.7e+09$	$2.8e+09$	10001	10000	25863	10000
fletch	1000	$8.100e-11$	$9.990e+02$	$5.0e-05$	$6.3e+01$	2347	1564	28041	2346
fminsrf2	5625	$1.000e+00$	$2.846e+01$	$2.2e-09$	$3.3e-01$	221	212	1522	220
fminsurf	5625	$1.000e+00$	$2.859e+01$	$5.0e-07$	$3.3e-01$	710	703	2689	709
freuroth	5000	$6.082e+05$	$5.049e+06$	$4.2e-03$	$5.5e+04$	12	12	34	11
genhumps	5000	$5.468e-06$	$1.281e+08$	$1.4e-03$	$6.0e+03$	6453	6222	25419	6452
genrose	500	$1.000e+00$	$1.870e+03$	$4.1e-05$	$3.0e+02$	378	308	4362	377
genrose_nash	100	$1.000e+00$	$4.041e+02$	$4.1e-05$	$1.3e+02$	96	77	931	95
hilbertb	10	$2.216e-14$	$5.102e+02$	$6.8e-07$	$1.1e+02$	6	6	11	5
indef	5000	$-2.024e+13$	$4.603e+03$	$7.1e+01$	$8.0e+01$	10001	7246	30578	10000
indefm	100000	$-1.005e+07$	$9.207e+04$	$1.3e-06$	$3.6e+02$	24	20	69	23
liarwhd	5000	$1.825e-02$	$2.925e+06$	$2.7e-01$	$4.8e+05$	13	13	31	12
mancino	100	$5.784e-02$	$1.103e+12$	$6.6e+02$	$2.9e+09$	12	11	24	11
modbeale	20000	$1.041e-02$	$1.264e+07$	$2.1e-01$	$3.1e+05$	10	10	53	9
morebv	5000	$3.637e-11$	$1.597e-07$	$9.5e-08$	$8.0e-04$	3	3	451	2
ncb20	5010	$-1.459e+03$	$1.000e+04$	$2.6e-05$	$2.8e+02$	64	52	526	63
ncb20b	5000	$7.351e+03$	$1.000e+04$	$8.7e-05$	$2.8e+02$	33	21	1214	32
noncvxu2	5000	$1.326e+04$	$3.235e+11$	$3.2e+00$	$3.3e+06$	935	864	4710	934
noncvxun	5000	$1.335e+04$	$3.335e+11$	$3.5e+00$	$3.6e+06$	906	824	4605	905
nondia	5000	$2.570e-04$	$2.000e+06$	$1.3e-02$	$2.0e+06$	3	3	4	2
nondquar	5000	$1.519e-03$	$5.006e+03$	$5.7e-03$	$2.0e+04$	31	19	229	30
osborneb	11	$4.014e-02$	$3.166e+00$	$5.8e-07$	$6.5e+00$	26	21	186	25
oscigrad	100000	$1.062e-03$	$6.121e+08$	$2.7e+02$	$2.2e+09$	10	10	42	9
oscipath	10	$9.997e-01$	$1.000e+00$	$1.5e-02$	$1.0e+00$	10001	6302	86167	10000
parkch	15	$1.624e+03$	$2.150e+03$	$1.8e-02$	$2.5e+04$	28	20	234	27
penalty1	1000	$8.186e+08$	$1.114e+17$	$1.9e+07$	$2.4e+13$	18	18	34	17
penalty2	200	$4.712e+13$	$4.712e+13$	$3.6e+00$	$1.6e+07$	12	12	127	11
penalty3	200	$1.002e-03$	$1.584e+09$	$1.9e-01$	$2.2e+06$	32	25	117	31

Continued on next page

Table A.3 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
powellsg	5000	$2.085e-03$	$2.688e+05$	$1.3e-02$	$1.6e+04$	14	14	57	13
power	10000	$6.432e+07$	$2.501e+15$	$6.0e+07$	$1.2e+14$	14	14	54	13
quartc	5000	$1.741e+09$	$6.241e+17$	$4.2e+06$	$1.3e+13$	21	21	54	20
schmvett	5000	$-1.499e+04$	$-1.429e+04$	$2.5e-06$	$7.5e+01$	8	8	55	7
scosine	100	$-9.900e+01$	$8.688e+01$	$7.7e-04$	$8.1e+02$	262	242	38423	261
scurly10	10000	$1.602e+24$	$7.006e+31$	$4.1e+23$	$1.3e+30$	15	15	74	14
scurly20	10000	$2.013e+25$	$9.031e+32$	$5.3e+24$	$1.6e+31$	15	15	72	14
scurly30	10000	$8.866e+25$	$4.163e+33$	$2.4e+25$	$7.5e+31$	15	15	71	14
sensors	100	$-2.055e+03$	$-5.648e+01$	$1.9e-10$	$7.1e+01$	17	15	51	16
sinquad	5000	$-6.757e+06$	$6.561e-01$	$6.8e-05$	$5.1e+03$	14	14	38	13
sparsine	5000	$3.184e-02$	$5.173e+07$	$8.0e-01$	$3.0e+06$	10	10	262	9
sparsqur	10000	$1.693e-01$	$1.406e+07$	$1.1e+00$	$1.2e+06$	13	13	39	12
spmsrtls	4999	$1.647e-08$	$4.141e+03$	$5.0e-05$	$7.7e+01$	14	14	207	13
srosenbr	5000	$1.831e-08$	$4.850e+04$	$1.2e-04$	$1.1e+04$	8	8	16	7
ssbrybnd	5000	$2.227e-03$	$1.249e+05$	$4.9e-01$	$9.0e+05$	27	19	7783	26
sscosine	5000	$-4.997e+03$	$4.387e+03$	$5.4e-03$	$5.9e+03$	441	316	1022191	440
stratec	10	$2.212e+03$	$2.818e+03$	$1.4e-02$	$4.7e+04$	49	39	302	48
testquad	5000	$7.180e+00$	$1.250e+09$	$5.6e+00$	$4.4e+07$	9	9	456	8
tointgor	50	$1.374e+03$	$5.074e+03$	$2.3e-05$	$6.0e+02$	9	9	125	8
tointgss	5000	$1.000e+01$	$4.499e+04$	$1.0e-05$	$4.2e+02$	9	9	24	8
tointpsp	50	$2.256e+02$	$1.828e+03$	$6.7e-05$	$1.1e+02$	29	24	126	28
tointqor	50	$1.175e+03$	$2.335e+03$	$1.8e-05$	$2.1e+02$	7	7	47	6
tquartic	5000	$2.424e-21$	$8.100e-01$	$1.4e-08$	$1.8e+00$	11	11	24	10
tridia	5000	$4.377e-04$	$1.250e+07$	$1.0e-01$	$4.1e+05$	9	9	580	8
vardim	200	$1.149e+08$	$3.257e+16$	$7.3e+09$	$1.6e+16$	13	13	24	12
vareigvl	50	$6.242e-11$	$1.126e+02$	$2.7e-05$	$4.2e+01$	8	7	33	7
watson	12	$1.602e-07$	$3.000e+01$	$4.8e-06$	$2.1e+02$	14	14	69	13
woods	4000	$7.877e+03$	$1.919e+07$	$4.4e-02$	$5.2e+05$	9	9	25	8

Table A.4: Solution of 109 nonlinear problems with trust-region CG.

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
arglina	200	2.000e+02	1.000e+03	1.8e−13	5.7e+01	3	3	4	2
arglinb	200	1.011e+02	8.353e+15	4.6e−02	1.4e+15	3	3	4	2
arglinc	200	1.011e+02	8.353e+15	4.6e−02	1.4e+15	3	3	4	2
arwhead	5000	1.110e−12	1.500e+04	5.9e−06	4.0e+04	6	6	11	5
bdqrtic	5000	2.001e+04	1.129e+06	8.4e−01	1.5e+06	10	10	29	9
box	10000	−1.865e+03	0.000e+00	1.9e−07	5.0e+01	10	9	25	9
boxpower	20000	4.736e−02	1.764e+05	6.9e−02	1.7e+05	12	7	29	11
brownal	200	2.037e−08	2.010e+06	2.8e−02	5.7e+05	2	2	2	1
broydn7d	5000	1.825e+03	1.760e+04	4.2e−05	1.1e+03	480	472	2345	479
brybnd	5000	6.154e−07	1.249e+05	4.3e−03	7.8e+03	10	10	57	9
chainwoo	4000	2.856e+03	1.445e+07	9.9e−02	4.2e+05	4939	3223	46889	4938
chnrosnb	50	2.153e−07	7.636e+03	2.7e−03	3.6e+03	95	58	810	94
chnrosnb_mod	100	4.134e−07	1.764e+04	3.3e−03	6.4e+03	173	117	1768	172
chnrsnbm	50	4.507e−10	8.633e+03	4.7e−04	4.4e+03	102	69	955	101
clplatea	5041	−1.259e−02	0.000e+00	3.2e−07	1.0e−01	28	22	1150	27
clplateb	5041	−5.095e−03	0.000e+00	2.2e−07	1.2e−02	4	4	459	3
clplatec	5041	−5.021e−03	0.000e+00	2.7e−08	9.9e−02	5	5	9374	4
cosine	10000	−9.999e+03	8.775e+03	2.5e−05	7.2e+01	12	12	26	11
cragglvy	5000	1.688e+03	2.749e+06	1.1e−01	2.8e+05	13	13	62	12
curly10	10000	−1.003e+06	−6.306e−01	5.1e−05	1.3e+02	24	20	57435	23
curly20	10000	−1.003e+06	−1.344e+00	1.7e−04	3.0e+02	22	18	64559	21
curly30	10000	−1.003e+06	−2.190e+00	4.2e−04	5.1e+02	18	15	59948	17
deconvu	63	4.700e−08	1.104e+02	4.1e−06	1.1e+02	26	16	319	25
dixmaana	3000	1.000e+00	2.850e+04	4.7e−04	1.2e+03	9	9	19	8
dixmaanb	3000	1.000e+00	4.724e+04	1.9e−03	2.0e+03	8	8	14	7
dixmaanc	3000	1.000e+00	8.248e+04	1.6e−04	3.7e+03	10	10	18	9
dixmaand	3000	1.000e+00	1.586e+05	4.7e−03	7.6e+03	10	10	18	9
dixmaane	3000	1.000e+00	2.209e+04	5.7e−05	1.1e+03	11	11	251	10
dixmaanf	3000	1.000e+00	4.104e+04	9.8e−04	1.9e+03	23	12	587	22
dixmaang	3000	1.000e+00	7.607e+04	2.3e−03	3.6e+03	27	13	892	26

Continued on next page

Table A.4 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
dixmaanh	3000	1.000e+00	1.517e+05	4.8e−03	7.4e+03	29	14	294	28
dixmaani	3000	1.000e+00	2.002e+04	2.7e−04	1.0e+03	11	11	762	10
dixmaanj	3000	1.000e+00	3.900e+04	4.5e−04	1.8e+03	28	14	815	27
dixmaank	3000	1.000e+00	7.400e+04	1.8e−03	3.6e+03	13	13	68	12
dixmaanl	3000	1.001e+00	1.496e+05	5.0e−03	7.4e+03	13	13	49	12
dixmaanm	3000	1.000e+00	9.358e+03	5.7e−05	4.4e+02	10	10	1589	9
dixmaann	3000	1.000e+00	2.018e+04	5.9e−05	1.0e+03	29	15	1922	28
dixmaano	3000	1.000e+00	3.635e+04	1.4e−03	2.0e+03	27	14	987	26
dixmaanp	3000	1.002e+00	7.128e+04	3.2e−03	4.0e+03	13	13	85	12
dixon3dq	10000	5.207e−10	8.000e+00	6.4e−06	5.7e+00	6	6	10794	5
dqdrtic	5000	1.111e−09	9.041e+06	4.3e−04	8.5e+04	8	8	18	7
dqrtic	5000	6.837e+09	6.241e+17	1.2e+07	1.3e+13	20	20	53	19
edensch	2000	1.200e+04	7.358e+06	7.5e−02	1.0e+05	14	14	37	13
eg2	1000	−9.989e+02	−8.406e+02	6.0e−09	5.4e+02	4	4	6	3
engval1	5000	5.549e+03	2.949e+05	5.8e−04	8.8e+03	11	11	31	10
errinros	50	4.009e+01	1.102e+05	1.0e−01	1.2e+05	62	40	586	61
errinros_mod	100	7.846e+01	3.140e+05	1.2e−01	1.9e+05	39	23	314	38
errinrsm	50	3.856e+01	1.529e+05	1.1e−01	1.3e+05	41	24	331	40
extrosnb	1000	5.872e−03	3.996e+05	3.0e−02	3.8e+04	54	31	269	53
fletbv3m	5000	−2.394e+05	1.982e+02	1.3e−06	4.4e+01	18	14	34	17
fletcbv2	5000	−5.003e−01	−5.003e−01	1.7e−08	4.4e−06	2	2	4943	1
fletcbv3	5000	−7.478e+12	1.982e+02	3.7e+01	4.4e+01	10001	9993	24984	10000
fletcbv3_mod	100	−2.034e+00	−1.879e−02	5.0e−07	1.6e−03	32	26	73	31
fletchbv	5000	−1.074e+21	−2.302e+11	3.8e+09	2.8e+09	10001	9999	24970	10000
fletchcr	1000	1.308e−12	9.990e+02	4.4e−05	6.3e+01	2390	1415	29831	2389
fminsrf2	5625	1.000e+00	2.846e+01	3.1e−07	3.3e−01	1473	1469	3612	1472
fminsurf	5625	1.000e+00	2.859e+01	4.0e−09	3.3e−01	1510	1506	3631	1509
freuroth	5000	6.082e+05	5.049e+06	5.4e−02	5.5e+04	11	11	28	10
genhumps	5000	2.154e+01	1.281e+08	3.1e+00	6.0e+03	10001	9726	32001	10000
genrose	500	1.000e+00	1.870e+03	7.0e−05	3.0e+02	582	465	4275	581
genrose_nash	100	1.000e+00	4.041e+02	6.4e−05	1.3e+02	164	128	1100	163
hilbertb	10	3.437e−14	5.102e+02	8.4e−07	1.1e+02	6	6	11	5
indef	5000	−4.959e+07	4.603e+03	9.8e+02	8.0e+01	10001	10001	29258	10000
indefm	100000	−9.865e+06	9.207e+04	1.3e−05	3.6e+02	377	328	1240	376
liarwhd	5000	1.535e−02	2.925e+06	2.5e−01	4.8e+05	14	14	33	13
mancino	100	2.340e−02	1.103e+12	4.2e+02	2.9e+09	12	11	23	11
modbeale	20000	3.158e−04	1.264e+07	2.8e−02	3.1e+05	11	11	65	10
morebv	5000	1.053e−11	1.597e−07	1.0e−07	8.0e−04	3	3	2692	2
ncb20	5010	−1.460e+03	1.000e+04	6.0e−05	2.8e+02	71	60	412	70
ncb20b	5000	7.351e+03	1.000e+04	7.9e−05	2.8e+02	31	21	1535	30
noncvxu2	5000	1.159e+04	3.235e+11	9.5e−01	3.3e+06	3628	3558	12209	3627
noncvxun	5000	1.227e+04	3.335e+11	2.8e+00	3.6e+06	3930	3882	12459	3929
nondia	5000	2.570e−04	2.000e+06	1.3e−02	2.0e+06	3	3	4	2
nondquar	5000	1.581e−03	5.006e+03	1.0e−02	2.0e+04	31	19	246	30
osborneb	11	4.014e−02	3.166e+00	5.8e−08	6.5e+00	31	24	230	30
oscigrad	100000	1.146e−03	6.121e+08	3.9e+02	2.2e+09	10	10	41	9
oscipath	10	9.997e−01	1.000e+00	1.7e−02	1.0e+00	10001	6194	83375	10000
parkch	15	1.624e+03	2.150e+03	4.6e−03	2.5e+04	32	23	223	31
penalty1	1000	8.186e+08	1.114e+17	1.9e+07	2.4e+13	18	18	34	17
penalty2	200	4.712e+13	4.712e+13	1.3e+00	1.6e+07	12	12	136	11
penalty3	200	1.002e−03	1.584e+09	3.5e−01	2.2e+06	35	26	129	34
powellsg	5000	2.277e−03	2.688e+05	1.4e−02	1.6e+04	14	14	57	13
power	10000	3.252e+07	2.501e+15	5.6e+07	1.2e+14	14	14	55	13
quartc	5000	6.837e+09	6.241e+17	1.2e+07	1.3e+13	20	20	53	19
schmvett	5000	−1.499e+04	−1.429e+04	7.3e−06	7.5e+01	7	7	50	6
scosine	100	−9.900e+01	8.688e+01	5.8e−04	8.1e+02	336	301	27244	335
scurly10	10000	2.554e+24	7.006e+31	1.2e+24	1.3e+30	14	14	76	13
scurly20	10000	3.493e+25	9.031e+32	1.5e+25	1.6e+31	14	14	71	13
scurly30	10000	1.703e+26	4.163e+33	7.1e+25	7.5e+31	14	14	66	13
sensors	100	−2.041e+03	−5.648e+01	2.1e−05	7.1e+01	17	14	48	16
sinquad	5000	−6.757e+06	6.561e−01	6.5e−05	5.1e+03	17	17	45	16
sparsine	5000	1.499e−03	5.173e+07	7.4e−01	3.0e+06	10	10	461	9
sparsqur	10000	1.322e−01	1.406e+07	9.9e−01	1.2e+06	13	13	41	12
spmsrtls	4999	2.434e−10	4.141e+03	4.2e−05	7.7e+01	32	25	523	31

Continued on next page

Table A.4 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
srosenbr	5000	1.916e−08	4.850e+04	1.2e−04	1.1e+04	8	8	16	7
ssbrybnd	5000	2.774e−04	1.249e+05	5.6e−01	9.0e+05	6086	4554	143194	6085
sscosine	5000	−3.992e+03	4.387e+03	6.0e+03	5.9e+03	10001	9847	1955112	10000
stratec	10	2.212e+03	2.818e+03	2.7e−02	4.7e+04	52	40	310	51
testquad	5000	4.323e−01	1.250e+09	7.4e+00	4.4e+07	9	9	632	8
tointgor	50	1.374e+03	5.074e+03	3.8e−04	6.0e+02	8	8	104	7
tointgss	5000	1.000e+01	4.499e+04	4.5e−07	4.2e+02	9	9	25	8
tointpsp	50	2.256e+02	1.828e+03	1.0e−06	1.1e+02	51	38	193	50
tointqor	50	1.175e+03	2.335e+03	4.9e−05	2.1e+02	7	7	44	6
tquartic	5000	2.439e−21	8.100e−01	1.4e−08	1.8e+00	11	11	24	10
tridia	5000	1.432e−05	1.250e+07	9.7e−02	4.1e+05	9	9	648	8
vardim	200	1.149e+08	3.257e+16	7.3e+09	1.6e+16	13	13	24	12
vareigvl	50	6.049e−14	1.126e+02	9.6e−07	4.2e+01	9	8	40	8
watson	12	1.597e−07	3.000e+01	1.2e−05	2.1e+02	13	13	62	12
woods	4000	7.877e+03	1.919e+07	5.9e−02	5.2e+05	9	9	25	8

A.3 Detailed results for nonlinear least squares trust-region method

Detailed results for individual problems are given in Table A.5 and Table A.6.

Table A.5: Solution of 47 nonlinear least-squares problems with LSMR.

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
argtrig	200	8.9e−09	3.3e+01	1.6e−04	1.3e+03	9	631	631	8
ba-l1	57	2.3e−10	6.4e+04	2.2e−02	3.1e+05	7	34	34	6
ba-l16	200	8.9e−09	3.3e+01	1.6e−04	1.3e+03	9	631	631	8
ba-l73	33753	2.2e+07	1.2e+08	7.7e+06	3.1e+08	10001	32592	32592	10000
broydn3d	5000	1.5e−11	2.5e+03	2.0e−05	2.8e+02	7	36	36	6
broydnbd	5000	1.5e−08	6.2e+04	3.7e−04	3.9e+03	12	77	77	11
dmn15103	99	2.4e+02	2.0e+06	3.5e+01	5.3e+07	77	1725	1725	76
dmn15332	66	1.1e+02	2.5e+05	4.6e+00	5.8e+06	179	3298	3298	178
dmn15333	99	7.5e+01	2.4e+05	3.8e+00	5.5e+06	171	6922	6922	170
dmn37142	66	1.1e+02	1.3e+05	3.1e+00	3.8e+06	145	2190	2190	144
dmn37143	99	1.7e+02	7.5e+04	3.3e+00	3.8e+06	32	137	137	31
eigena	2550	1.8e−07	2.0e+04	2.9e−04	4.5e+02	100	4095	4095	99
eigenb	2550	6.3e−06	5.0e+01	3.7e−06	1.9e+01	1192	74133	74133	1191
eigenc	2652	2.4e−06	5.6e+03	8.2e−05	2.4e+02	430	53610	53610	429
inteqne	12	2.4e−17	3.2e−02	7.1e−09	3.1e−01	4	14	14	3
lukšan-vlček5.1	20	8.1e−08	2.3e+03	2.4e−04	1.5e+03	61	529	529	60
lukšan-vlček5.11	20	2.7e−08	4.5e+00	3.3e−06	6.2e+00	8	43	43	7
lukšan-vlček5.12	21	1.9e−08	4.2e+01	4.3e−06	2.7e+01	9	89	89	8
lukšan-vlček5.13	20	1.8e−07	2.5e+02	2.8e−05	6.9e+01	8	37	37	7
lukšan-vlček5.14	20	1.1e−03	1.6e+05	1.0e−02	5.7e+04	9	37	37	8
lukšan-vlček5.15	21	4.2e−01	6.4e+06	3.3e−01	4.1e+05	17	68	68	16
lukšan-vlček5.16	21	2.0e−07	5.6e+01	3.1e−05	5.2e+01	8	43	43	7
lukšan-vlček5.17	21	2.0e−07	1.4e+02	3.0e−05	7.0e+01	8	44	44	7
lukšan-vlček5.18	21	1.9e−09	1.5e+01	1.3e−06	7.1e+00	9	33	33	8
lukšan-vlček5.2	20	3.5e+01	7.8e+03	6.2e−03	6.9e+03	12	92	92	11
lukšan-vlček5.3	20	1.5e−05	2.2e+03	1.5e−03	1.5e+03	8	48	48	7
lukšan-vlček5.4	20	1.7e+00	2.4e+03	5.5e−03	5.7e+03	25	214	214	24
mgh19	11	4.4e−02	1.6e+01	7.1e−06	2.2e+01	57	507	507	56
mgh21	20	2.2e−11	1.2e+02	3.0e−06	3.7e+02	31	110	110	30
mgh22	20	5.5e−06	5.4e+02	4.0e−04	5.1e+02	9	44	44	8
mgh25	10	3.7e−06	1.1e+06	5.3e−02	2.2e+06	9	25	25	8
mgh26	10	1.4e−05	3.5e−03	6.4e−07	5.0e−02	29	245	245	28
mgh27	10	3.1e−07	1.4e+02	7.8e−05	1.7e+02	5	14	14	4
mgh28	10	2.0e−16	3.9e−04	4.0e−09	2.0e−02	4	35	35	3
mgh29	10	2.4e−17	3.2e−02	7.1e−09	3.1e−01	4	14	14	3
mgh30	10	1.2e−11	1.1e+01	2.1e−05	2.5e+01	6	33	33	5
mgh31	10	1.4e−12	1.8e+02	9.9e−06	4.1e+02	7	27	27	6
mgh32	10	5.0e+00	2.5e+01	7.0e−16	6.3e+00	2	4	4	1

Continued on next page

Table A.5 — continued from previous page

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
mgh33	10	2.3e+00	4.3e+06	1.4e−09	3.1e+06	2	4	4	1
mgh34	10	3.1e+00	2.0e+06	4.3e−10	1.6e+06	2	4	4	1
mgh35	10	3.3e−03	1.7e−02	1.1e−06	6.7e−01	30	228	228	29
msqrta	1024	5.5e−07	4.0e+03	7.4e−05	1.7e+02	47	16071	16071	46
msqrtb	1024	2.2e−08	4.0e+03	1.6e−05	1.7e+02	34	6992	6992	33
nzf1	13	3.9e−15	5.0e+03	3.4e−07	9.3e+02	7	36	36	6
spmsqrt	4999	1.7e−10	2.1e+03	2.6e−05	3.9e+01	13	242	242	12
yatp1sq	123200	1.6e−01	1.3e+09	2.3e+00	8.1e+06	20	60	60	19
yatp2sq	123200	3.0e−08	3.8e+09	1.3e−02	3.8e+05	35	102	102	34

Table A.6: Solution of 47 nonlinear least-squares problems with LSQR.

model	nvar	$f(x)$	$f(x_0)$	$\|g(x)\|$	$\|g(x_0)\|$	$\#f$	$\#g$	$\#hv$	$\#it$
argtrig	200	1.1e−11	3.3e+01	7.6e−05	1.3e+03	8	654	654	7
ba-l1	57	1.7e−09	6.4e+04	7.1e−02	3.1e+05	7	35	35	6
ba-l16	200	1.1e−11	3.3e+01	7.6e−05	1.3e+03	8	654	654	7
ba-l73	33753	7.4e+07	1.2e+08	5.3e+07	3.1e+08	10001	37754	37754	10000
broydn3d	5000	1.1e−10	2.5e+03	6.6e−05	2.8e+02	7	35	35	6
broydnbd	5000	2.4e−07	6.2e+04	1.8e−03	3.9e+03	12	62	62	11
dmn15103	99	1.7e+02	2.0e+06	3.8e+01	5.3e+07	174	2075	2075	173
dmn15332	66	5.5e+02	2.5e+05	5.5e+00	5.8e+06	957	11189	11189	956
dmn15333	99	4.7e+01	2.4e+05	5.4e+00	5.5e+06	1888	34421	34421	1887
dmn37142	66	1.1e+02	1.3e+05	3.7e+00	3.8e+06	114	1324	1324	113
dmn37143	99	1.4e+02	7.5e+04	5.1e−01	3.8e+06	97	439	439	96
eigena	2550	4.7e−08	2.0e+04	2.3e−05	4.5e+02	95	3943	3943	94
eigenb	2550	9.6e−07	5.0e+01	4.3e−06	1.9e+01	995	75269	75269	994
eigenc	2652	1.5e−08	5.6e+03	1.5e−04	2.4e+02	408	39554	39554	407
inteqne	12	2.3e−17	3.2e−02	7.0e−09	3.1e−01	4	14	14	3
lukšan-vlček5.1	20	8.2e−11	2.3e+03	1.7e−04	1.5e+03	61	468	468	60
lukšan-vlček5.11	20	2.1e−08	4.5e+00	2.7e−06	6.2e+00	8	45	45	7
lukšan-vlček5.12	21	2.2e−07	4.2e+01	2.5e−05	2.7e+01	8	67	67	7
lukšan-vlček5.13	20	1.3e−07	2.5e+02	2.1e−05	6.9e+01	8	37	37	7
lukšan-vlček5.14	20	8.0e−04	1.6e+05	7.5e−03	5.7e+04	9	36	36	8
lukšan-vlček5.15	21	2.1e−02	6.4e+06	7.8e−02	4.1e+05	20	88	88	19
lukšan-vlček5.16	21	1.4e−07	5.6e+01	2.3e−05	5.2e+01	8	45	45	7
lukšan-vlček5.17	21	2.0e−07	1.4e+02	3.0e−05	7.0e+01	8	44	44	7
lukšan-vlček5.18	21	1.9e−09	1.5e+01	1.3e−06	7.1e+00	9	33	33	8
lukšan-vlček5.2	20	3.5e+01	7.8e+03	6.1e−03	6.9e+03	11	82	82	10
lukšan-vlček5.3	20	5.6e−06	2.2e+03	6.4e−04	1.5e+03	8	49	49	7
lukšan-vlček5.4	20	1.7e+00	2.4e+03	4.4e−03	5.7e+03	29	223	223	28
mgh19	11	4.4e−02	1.6e+01	1.0e−05	2.2e+01	66	618	618	65
mgh21	20	1.8e−11	1.2e+02	2.7e−06	3.7e+02	33	112	112	32
mgh22	20	4.0e−06	5.4e+02	3.5e−04	5.1e+02	9	44	44	8
mgh25	10	3.7e−06	1.1e+06	5.3e−02	2.2e+06	9	25	25	8
mgh26	10	1.4e−05	3.5e−03	3.4e−07	5.0e−02	34	212	212	33
mgh27	10	3.1e−07	1.4e+02	7.8e−05	1.7e+02	5	14	14	4
mgh28	10	4.8e−16	3.9e−04	6.2e−09	2.0e−02	3	25	25	2
mgh29	10	2.3e−17	3.2e−02	7.0e−09	3.1e−01	4	14	14	3
mgh30	10	7.1e−12	1.1e+01	1.8e−05	2.5e+01	6	33	33	5
mgh31	10	1.3e−12	1.8e+02	1.0e−05	4.1e+02	7	27	27	6
mgh32	10	5.0e+00	2.5e+01	2.5e−16	6.3e+00	2	4	4	1
mgh33	10	2.3e+00	4.3e+06	1.1e−09	3.1e+06	2	4	4	1
mgh34	10	3.1e+00	2.0e+06	6.0e−10	1.6e+06	2	4	4	1
mgh35	10	3.3e−03	1.7e−02	1.1e−06	6.7e−01	36	267	267	35
msqrta	1024	2.3e−09	4.0e+03	1.7e−05	1.7e+02	36	7492	7492	35
msqrtb	1024	8.0e−11	4.0e+03	4.3e−06	1.7e+02	31	5290	5290	30
nzf1	13	1.8e−09	5.0e+03	2.3e−04	9.3e+02	7	32	32	6
spmsqrt	4999	1.6e−13	2.1e+03	3.3e−07	3.9e+01	12	291	291	11
yatp1sq	123200	1.6e−01	1.3e+09	2.3e+00	8.1e+06	26	81	81	25
yatp2sq	123200	2.3e−08	3.8e+09	1.1e−02	3.8e+05	35	102	102	34

References

Å. Björck. Use of conjugate gradients for solving linear least squares problems. In I. S. Duff, editor, *Conjugate Gradient Methods and Similar Techniques*, pages 49–71, Harwell, Didcot, UK, 1979. AERE Harwell.

Å. Björck and M. A. Saunders. Krylov subspace algorithms for overdetermined and underdetermined linear systems. Technical report, Stanford University, Stanford, CA, USA, 2017.

Å. Björck, T. Elfving, and Z. Strakoš. Stability of conjugate gradient and Lanczos methods for linear least squares problems. *SIAM Journal on Matrix Analysis and Applications*, 19(3):720–736, 1998. DOI: 10.1137/S089547989631202X.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*, volume 1 of *MPS/SIAM Series on Optimization*. SIAM, Philadelphia, 2000. DOI: 10.1137/1.9780898719857.

R. S. Dembo and T. Steihaug. Truncated-Newton algorithms for large-scale unconstrained optimization. *Mathematical Programming*, 26(2):190–212, 1983.

J. E. Dennis, Jr and J. J. Moré. Quasi-Newton methods, motivation and theory. *SIAM Review*, 19(1):46–89, 1977.

D. C.-L. Fong. *Minimum-Residual Methods for Sparse Least-Squares Using Golub-Kahan Bidiagonalization*. PhD thesis, Stanford University, 2011.

D. C.-L. Fong and M. A. Saunders. LSMR: An iterative algorithm for sparse least-squares problems. *SIAM Journal on Scientific Computing*, 33(5):2950–2971, 2011. DOI: 10.1137/10079687X.

D. C.-L. Fong and M. A. Saunders. CG versus MINRES: An empirical comparison. *SQU Journal for Science*, 17(1): 44–62, 2012.

R. Fourer, C. Maheshwari, A. Neumaier, D. Orban, and H. Schichl. Convexity and concavity detection in computational graphs. *INFORMS Journal on Computing*, 22:26–43, 2010. DOI: 10.1287/ijoc.1090.0321.

G. H. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *SIAM Journal on Numerical Analysis*, 2(2):205–224, 1965. DOI: 10.1137/0702016.

N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEst: a Constrained and Unconstrained Testing Environment with safe threads for Mathematical Optimization. *Computational Optimization and Applications*, 60:545–557, 2015. DOI: 10.1007/s10589-014-9687-3.

M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436, 1952.

T. Kloek. Conjugate gradients and conjugate residuals type methods for solving least squares problems from tomography. Bachelor's thesis, TU Delft, Delft, Netherlands, 2012.

C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of research of the National Bureau of Standards B*, 45:225–280, 1950.

K. Levenberg. A method for the solution of certain problems in least squares. *Quarterly Journal on Applied Mathematics*, 2:164–168, 1944.

C.-J. Lin and J. J. Moré. Newton's method for large bound-constrained optimization problems. *SIAM Journal on Optimization*, 9:1100–1127, 1998. DOI: 10.1137/S1052623498345075.

D. G. Luenberger. The conjugate residual method for constrained minimization problems. *SIAM Journal on Numerical Analysis*, 7(3):390–398, 1970.

L. Lukšan, C. Matonoha, and J. Vlček. Modified CUTE problems for sparse unconstrained optimization. Technical Report 1081, Institute of Computer Science, Academy of Science of the Czech Republic, Pod Vodàrenskou vèžì 2, 182 07 Prague 8, 2010.

D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11:431–441, 1963. DOI: 10.1137/0111030.

C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975. DOI: 10.1137/0712047.

C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, 1982.

T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, 20(3):626–637, 1983. DOI: 10.1137/0720042.

E. Stiefel. Relaxationsmethoden bester Strategie zur Lösung linearer Gleichungssysteme. *Commentarii Mathematici Helvetici*, 29(1):157–179, 1955.

Y. Yuan. On the truncated conjugate gradient method. *Mathematical Programming*, 87(3):561–573, 2000. DOI: 10.1007/s101070050012.