# A Retrospective Trust-Region Method for Unconstrained Optimization

by F. Bastin[1], V. Malmedy[2,3], M. Mouffe[4],
Ph. L. Toint[2] and D. Tomanos[2,5]

Report 07/08 31 October 2007

[1] Computing Science and Operational Research Department
Université de Montréal
CP 6128, Succ. Centre-Ville, Montreal, Canada
Email: bastin@iro.umontreal.ca

[2] Department of Mathematics
FUNDP — University of Namur
61, Rue de Bruxelles, B-5000 Namur, Belgium
Email: vincent.malmedy@fundp.ac.be, philippe.toint@fundp.ac.be,
dimitri.tomanos@fundp.ac.be

[3] Fonds de la Recherche Scientifique – FNRS
5, Rue d'Egmont, B-1000 Bruxelles, Belgium
Research Fellow

[4] C.E.R.F.A.C.S.
42, Avenue Gustave Coriolis, F-31 057 Toulouse Cedex, France
Email: melodie.mouffe@cerfacs.fr

[5] Fonds pour la formation à la Recherche dans l'Industrie et l'Agriculture
5, Rue d'Egmont, B-1000 Bruxelles, Belgium
Research Fellow

# A Retrospective Trust-Region Method for Unconstrained Optimization

F. Bastin, V. Malmedy, M. Mouffe, Ph. Toint, D. Tomanos

31 October 2007

**Abstract**

We introduce a new trust-region method for unconstrained optimization where the radius update is computed using the model information at the current iterate rather than at the preceding one. The update is then performed according to how well the current model retrospectively predicts the value of the objective function at last iterate. Global convergence to first- and second-order critical points is proved under classical assumptions and preliminary numerical experiments on CUTEr problems indicate that the new method is very competitive.

**Keywords:** unconstrained minimization, trust-region methods, convergence theory, numerical experiments.

## 1 Introduction

Trust-region methods are well-known techniques in nonlinear nonconvex programming, whose concept has matured over more than thirty years (for an extensive coverage, see Conn, Gould and Toint, 2000). In such methods, one considers a model $m_k$ of the objective function which is assumed to be adequate in a "trust region", which is a neighbourhood of the current iterate $x_k$. This neighbourhood is often represented by a ball in some norm, whose radius $\Delta_k$ is then updated from iteration $k$ to iteration $k+1$ by considering how well $m_k$ predicts the objective function value at iterate $x_{k+1}$. In retrospect, this might seem unnatural since the new radius $\Delta_{k+1}$ will determine the region in which a possibly updated model $m_{k+1}$ is expected to predict the value of the objective function around $x_{k+1}$. Our aim in this paper is to propose a variant of the trust-region algorithm that determines $\Delta_{k+1}$ according to how well $m_{k+1}$ predicts the value of the objective function at $x_k$, thereby synchronizing the radius update with the change in models.

This paper explores the theoretical properties and practical numerical potential of the new trust-region algorithm. We introduce the new method in Section 2, and study its convergence in the next section. Section 4 presents preliminary numerical experience on standard nonlinear problems. We conclude and examine perspectives for future research in Section 5.

## 2 A retrospective trust-region algorithm

We consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{2.1}$$

where $f$ is a twice-continuously differentiable objective function which maps $\mathbb{R}^n$ into $\mathbb{R}$ and is bounded below. Trust-region methods are iterative processes, which, given a starting point $x_0$, construct a sequence $(x_k)_{k\geqslant 0}$ of iterates hopefully converging to a solution of (2.1). At each iteration $k$, a twice-continuously differentiable model $m_k$ is defined which we trust inside a (typically Euclidean) ball $\mathcal{B}_k$ of radius $\Delta_k > 0$ centred at the current iterate $x_k$, called the *trust region.* A step $s_k$ is then computed by (approximately) minimizing the model $m_k$ inside the trust region $\mathcal{B}_k$. The trial point $x_k + s_k$ is then accepted as the next iterate $x_{k+1}$ if and only if $\rho_k$, the ratio

$$\rho_k \stackrel{\rm def}{=} \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$$

of achieved reduction (in the objective function) to predicted reduction (in its local model $m_k$), is larger than a small positive constant $\eta_1$ (iteration $k$ is then called *successful*). In the classical framework, the trust-region radius is updated at the end of each iteration: it is decreased if the trial point is rejected (that is if $\rho_k < \eta_1$) and left unchanged or increased otherwise.

Our new algorithm differs in that the trust-region radius is updated after each successful iteration $k$ (that is at the beginning of iteration $k+1$) on the basis of the *retrospective* ratio

$$\tilde{\rho}_{k+1} \stackrel{\rm def}{=} \frac{f(x_{k+1}) - f(x_{k+1} - s_k)}{m_{k+1}(x_{k+1}) - m_{k+1}(x_{k+1} - s_k)} = \frac{f(x_k) - f(x_k + s_k)}{m_{k+1}(x_k) - m_{k+1}(x_k + s_k)}$$

of achieved to predicted changes, while continuing to use $\rho_k$ to decide whether the trial iterate may be accepted. Our method therefore distinguishes the two roles played by $\rho_k$ in the classical algorithm: that of deciding acceptance of the trial iterate and that of determining the radius update. It also explicitely takes into account that $m_{m+1}$, not $m_k$, is used within the trust region of radius $\Delta_{k+1}$.

This leads to the retrospective trust-region method described as Algorithm 2.1, in which we leave the precise definitions of the model (at Step 1) and of "sufficient reduction" (at Step 3) for the next section.

## 3 Convergence theory

We now investigate the convergence properties of our algorithm. Since it can be considered as a variant of the basic trust-region method of Conn et al. (2000), we expect similar results and significant similarities in their proofs. In what follows, we have attempted to be explicit on the assumptions and properties, but to refer to Chapter 6 of this reference whenever possible.

Our assumptions are identical to those used for the basic trust-region method.

**A.1** The Hessian of the objective function $\nabla_{xx} f$ is uniformly bounded, i.e. there exists a positive constant $\kappa_{\rm ufh}$ such that, for all $x \in \mathbb{R}^n$,

$$\|\nabla_{xx} f(x)\| \leqslant \kappa_{\rm ufh}.$$

**A.2** The model $m_k$ is first-order coherent with the function $f$ at each iteration $x_k$, i.e. their values and gradients are equal at $x_k$ for all $k$:

$$m_k(x_k) = f(x_k) \quad \text{and} \quad g_k \stackrel{\rm def}{=} \nabla_x m_k(x_k) = \nabla_x f(x_k).$$

**A.3** The Hessian of the model $\nabla_{xx} m_k$ is uniformly bounded, i.e. there exists a constant $\kappa_{\rm umh} \geqslant 1$ such that, for all $x \in \mathbb{R}^n$ and for all $k$,

$$\|\nabla_{xx} m_k(x)\| \leqslant \kappa_{\rm umh} - 1.$$

**Algorithm 2.1: Retrospective trust-region algorithm (RTR)**

**Step 0: Initialisation.** An initial point $x_0$ and initial trust-region radius $\Delta_0 > 0$ are given. The constants $\eta_1$, $\tilde{\eta}_1$, $\tilde{\eta}_2$, $\gamma_1$ and $\gamma_2$ are also given and satisfy $0 < \eta_1 < 1$, $0 < \tilde{\eta}_1 \leqslant \tilde{\eta}_2 < 1$ and $0 < \gamma_1 \leqslant \gamma_2 < 1$. Compute $f(x_0)$ and set $k = 0$.

**Step 1: Model definition.** Select a twice-continuously differentiable model $m_k$ defined in $\mathcal{B}_k$.

**Step 2: Retrospective trust-region radius update.** If $k = 0$, go to Step 3. If $x_k = x_{k-1}$, then choose $\Delta_k$ in $[\gamma_1\Delta_{k-1}, \gamma_2\Delta_{k-1})$. Else, define

$$\tilde{\rho}_k = \frac{f(x_{k-1}) - f(x_k)}{m_k(x_{k-1}) - m_k(x_k)} \tag{2.2}$$

and choose

$$\Delta_k \in \begin{cases} [\Delta_{k-1}, \infty) & \text{if } \tilde{\rho}_k \geqslant \tilde{\eta}_2, \\ [\gamma_2\Delta_{k-1}, \Delta_{k-1}) & \text{if } \tilde{\rho}_k \in [\tilde{\eta}_1, \tilde{\eta}_2), \\ [\gamma_1\Delta_{k-1}, \gamma_2\Delta_{k-1}) & \text{if } \tilde{\rho}_k < \tilde{\eta}_1. \end{cases} \tag{2.3}$$

**Step 3: Step calculation.** Compute a step $s_k$ that "sufficiently reduces the model" $m_k$ and such that $x_k + s_k \in \mathcal{B}_k$.

**Step 4: Acceptance of the trial point.** Compute $f(x_k + s_k)$ and define

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}. \tag{2.4}$$

If $\rho_k \geqslant \eta_1$, then define $x_{k+1} = x_k + s_k$; otherwise define $x_{k+1} = x_k$.
Increment $k$ by 1 and go to Step 1.

**A.4** The decrease on the model $m_k$ is at least as much as a fraction of that obtained at the Cauchy point; i.e. there exists a constant $\kappa_{\text{mdc}} \in (0, 1)$ such that, for all $k$,

$$m_k(x_k) - m_k(x_k + s_k) \geqslant \kappa_{\text{mdc}} \|g_k\| \min\left[\frac{\|g_k\|}{\beta_k}, \Delta_k\right]$$

with $\beta_k \stackrel{\text{def}}{=} 1 + \max_{x \in \mathcal{B}_k} \|\nabla_{xx} m_k(x)\|$.

Note that A.4 specifies the notion of "sufficient reduction" used in Step 3 of our algorithm, while the choice of $m_k$ in Step 1 is limited by A.2 and A.3. We also note that $s_k \neq 0$ whenever $g_k \neq 0$ because of A.4.

## 3.1 Convergence to First-Order Critical Points

In this section, we prove that the retrospective trust-region algorithm is globally convergent to first-order critical points, in the sense that every limit point $x_*$ of the sequence of iterates $(x_k)$ produced by the algorithm 2.1 satisfies

$$\nabla_x f(x_*) = 0$$

irrespective of the choice of the starting point $x_0$ and initial trust-region radius $\Delta_0$.

We first give a bound on the error between the true objective function $f$ and its current model $m_k$ at the previous iterate $x_{k-1}$.

**Theorem 3.1** Suppose that A.1–A.3 hold. Then we have that

$$|f(x_k) - m_{k-1}(x_k)| \leqslant \kappa_{\rm ubh}\Delta_{k-1}^2 \tag{3.5}$$

and, if iteration $k-1$ is successful, that

$$|f(x_{k-1}) - m_k(x_{k-1})| \leqslant \kappa_{\rm ubh}\Delta_{k-1}^2 \tag{3.6}$$

where

$$\kappa_{\rm ubh} \stackrel{\rm def}{=} \max[\kappa_{\rm ufh}, \kappa_{\rm umh}]. \tag{3.7}$$

**Proof.** The bound (3.5) directly results from Theorem 6.4.1 in Conn et al. (2000). We thus only prove (3.6). Because the objective function and the model are $C^2$ functions, we may apply the mean value theorem on the objective function $f$ and on the model $m_k$, and obtain from $x_{k-1} = x_k - s_{k-1}$ that

$$f(x_{k-1}) = f(x_k) - \langle s_{k-1}, \nabla_x f(x_k)\rangle + \tfrac{1}{2}\langle s_{k-1}, \nabla_{xx} f(\xi_k) s_{k-1}\rangle \tag{3.8}$$

$$m_k(x_{k-1}) = m_k(x_k) - \langle s_{k-1}, \nabla_x m_k(x_k)\rangle + \tfrac{1}{2}\langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1}\rangle \tag{3.9}$$

for some $\xi_k$, $\zeta_k$ in the segment $[x_{k-1}, x_k]$.

Because of A.2, the objective function $f$ and the model $m_k$ have the same value and gradient at $x_k$. Thus, subtracting (3.9) from (3.8) and taking absolute values yields that

$$\begin{aligned} |f(x_{k-1}) - m_k(x_{k-1})| &= \tfrac{1}{2}\,|\langle s_{k-1}, \nabla_{xx} f(\xi_k) s_{k-1}\rangle - \langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1}\rangle| \\ &\leqslant \tfrac{1}{2}\,[|\langle s_{k-1}, \nabla_{xx} f(\xi_k) s_{k-1}\rangle| + |\langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1}\rangle|] \\ &\leqslant \tfrac{1}{2}(\kappa_{\rm ufh} + \kappa_{\rm umh} - 1)\|s_{k-1}\|^2 \\ &\leqslant \tfrac{1}{2}(\kappa_{\rm ufh} + \kappa_{\rm umh} - 1)\Delta_{k-1}^2, \end{aligned} \tag{3.10}$$

where we successively used the triangle inequality, the Cauchy-Schwarz inequality, the induced matrix norm properties, A.1, A.3, and the fact that $x_k \in \mathcal{B}_{k-1}$ implies that $\|s_{k-1}\| \leqslant \Delta_{k-1}$. So (3.6) clearly holds. □

Thus the analog of Theorem 6.4.1 of Conn et al. (2000) holds in our case, where we replace the forward difference $f(x_{k+1}) - m_k(x_{k+1})$ by its retrospective variant $f(x_{k-1}) - m_k(x_{k-1})$.

As our new ratio $\tilde{\rho}_k$ uses the reduction in $m_k$ instead of the reduction in $m_{k-1}$, we are interested in a bound on their difference, which is provided by this next result.

**Lemma 3.2** Suppose that A.1–A.3 hold. Then we have that, for every successful iteration $k-1$,

$$|\,[m_{k-1}(x_{k-1}) - m_{k-1}(x_k)] - [m_k(x_{k-1}) - m_k(x_k)]\,| \leqslant 2\kappa_{\rm ubh}\Delta_{k-1}^2. \tag{3.11}$$

**Proof.** Using the model differentiability, we apply the mean value theorem on the model $m_{k-1}$, and we obtain that

$$m_{k-1}(x_k) = m_{k-1}(x_{k-1}) + \langle s_{k-1}, g_{k-1}\rangle + \tfrac{1}{2}\langle s_{k-1}, \nabla_{xx} m_{k-1}(\psi_{k-1}) s_{k-1}\rangle \tag{3.12}$$

for some $\psi_{k-1}$ in the segment $[x_{k-1}, x_k]$. Remember that (3.9) in the previous proof gives that

$$m_k(x_{k-1}) \quad = \quad m_k(x_k) - \langle s_{k-1}, g_k \rangle + \tfrac{1}{2} \langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1} \rangle \tag{3.13}$$

for some $\zeta_k$ in the segment $[x_{k-1}, x_k]$. Substituting (3.12) and (3.13) inside the left-hand size of (3.11), and using A.3, the triangle inequality, the Cauchy-Schwarz inequality, and the induced matrix norm properties yield that

$$\begin{aligned} & |\, [m_{k-1}(x_{k-1}) - m_{k-1}(x_k)] - [m_k(x_{k-1}) - m_k(x_k)]\, | \\ = \quad & \left| - \langle s_{k-1}, g_{k-1} - g_k \rangle - \tfrac{1}{2} \left( \langle s_{k-1}, \nabla_{xx} m_{k-1}(\psi_{k-1}) s_{k-1} \rangle + \langle s_{k-1}, \nabla_{xx} m_k(\zeta_k) s_{k-1} \rangle \right) \right| \\ \leqslant \quad & \|s_{k-1}\| \cdot \|g_{k-1} - g_k\| + \kappa_{\rm umh} \|s_{k-1}\|^2. \end{aligned} \tag{3.14}$$

Now observe that, because of A.2, $\|g_{k-1} - g_k\| = \|\nabla_x f(x_{k-1}) - \nabla_x f(x_k)\|$. We then apply the mean value theorem on $\nabla_x f$ and obtain that

$$\nabla_x f(x_k) = \nabla_x f(x_{k-1}) + \int_0^1 \nabla_{xx} f(x_{k-1} + \alpha s_{k-1}) s_{k-1} \, d\alpha. \tag{3.15}$$

Thus the Cauchy-Schwarz inequality, and A.1 give that

$$\|g_{k-1} - g_k\| \leqslant \int_0^1 \|\nabla_{xx} f(x_{k-1} + \alpha s_{k-1})\| \cdot \|s_{k-1}\| \, d\alpha \leqslant \int_0^1 \kappa_{\rm ufh} \|s_{k-1}\| \, d\alpha = \kappa_{\rm ufh} \|s_{k-1}\|. \tag{3.16}$$

Substituting this bound in (3.14), we obtain that

$$|\, [m_{k-1}(x_{k-1}) - m_{k-1}(x_k)] - [m_k(x_{k-1}) - m_k(x_k)]\, | \leqslant (\kappa_{\rm ufh} + \kappa_{\rm umh}) \|s_{k-1}\|^2 = 2\kappa_{\rm ubh} \Delta_{k-1}^2$$

where we finally use (3.7), and the fact that $x_k \in \mathcal{B}_{k-1}$. □

We conclude from this result that the denominators in the expression of $\tilde{\rho}_k$ and $\rho_{k-1}$ differ by a quantity which is of the same order as the error between the model and the objective function. Using this observation, we are now capable of showing that the iteration must be successful if the radius is sufficiently small compared to the gradient, and also that the trust-region radius has to increase in this case.

**Theorem 3.3** Suppose that A.1–A.4 hold. Suppose furthermore that $g_k \neq 0$ and that

$$\Delta_{k-1} \leqslant \min \left[ 1 - \eta_1, \frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)} \right] \frac{\kappa_{\rm mdc}}{\kappa_{\rm ubh}} \|g_{k-1}\|. \tag{3.17}$$

Then iteration $k-1$ is successful and

$$\Delta_k \geqslant \Delta_{k-1}. \tag{3.18}$$

**Proof.** We first apply Theorem 6.4.2 of Conn et al. (2000) to deduce that iteration $k-1$ is successful and thus that $x_k = x_{k-1} + s_{k-1} \neq x_{k-1}$. Observe now that the constants $\tilde{\eta}_2$ and $\kappa_{\rm mdc}$ lie in the interval $(0, 1)$, which implies that

$$\frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)} < \frac{1}{2} < 1 \quad \text{and} \quad \kappa_{\rm mdc} \frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)} < 1. \tag{3.19}$$

The conditions (3.17), (3.19), and (3.7), combined with the definition of $\beta_{k-1}$ in A.4 imply that

$$\Delta_{k-1} < \frac{1}{2}\frac{\kappa_{\rm mdc}}{\kappa_{\rm ubh}}\|g_{k-1}\| \quad \text{and} \quad \Delta_{k-1} < \frac{\|g_{k-1}\|}{\beta_{k-1}}. \tag{3.20}$$

As a consequence, A.4 immediately gives that

$$m_{k-1}(x_{k-1}) - m_{k-1}(x_k) \geqslant \kappa_{\rm mdc}\|g_{k-1}\| \min\left[\frac{\|g_{k-1}\|}{\beta_{k-1}}, \Delta_{k-1}\right] = \kappa_{\rm mdc}\|g_{k-1}\|\Delta_{k-1}. \tag{3.21}$$

On the other hand, we may apply Lemma 3.2 and use the triangle inequality to obtain that

$$\begin{array}{ll} & \big|m_{k-1}(x_{k-1}) - m_{k-1}(x_k)\big| - \big|m_k(x_{k-1}) - m_k(x_k)\big| \\ \leqslant & \big|\big[m_{k-1}(x_{k-1}) - m_{k-1}(x_k)\big] - \big[m_k(x_{k-1}) - m_k(x_k)\big]\big| \\ \leqslant & 2\kappa_{\rm ubh}\Delta_{k-1}^2 \end{array}$$

and therefore, with (3.21), that

$$\begin{array}{rcl} |m_k(x_{k-1}) - m_k(x_k)| & \geqslant & |m_{k-1}(x_{k-1}) - m_{k-1}(x_k)| - 2\kappa_{\rm ubh}\Delta_{k-1}^2 \\ & \geqslant & \kappa_{\rm mdc}\|g_{k-1}\|\Delta_{k-1} - 2\kappa_{\rm ubh}\Delta_{k-1}^2. \end{array} \tag{3.22}$$

We finally may apply Theorem 3.1 and deduce from A.2, (3.6) and (3.22) that

$$|\tilde{\rho}_k - 1| = \left|\frac{f(x_{k-1}) - m_k(x_{k-1})}{m_k(x_{k-1}) - m_k(x_k)}\right| \leqslant \frac{\kappa_{\rm ubh}\Delta_{k-1}}{\kappa_{\rm mdc}\|g_{k-1}\| - 2\kappa_{\rm ubh}\Delta_{k-1}} \leqslant 1 - \tilde{\eta}_2 \tag{3.23}$$

because (3.17) implies that $(3 - 2\tilde{\eta}_2)\kappa_{\rm ubh}\Delta_{k-1} \leqslant (1 - \tilde{\eta}_2)\kappa_{\rm mdc}\|g_{k-1}\|$ and thus that $\kappa_{\rm ubh}\Delta_{k-1} \leqslant (1 - \tilde{\eta}_2)(\kappa_{\rm mdc}\|g_{k-1}\| - 2\kappa_{\rm ubh}\Delta_{k-1})$ with $\kappa_{\rm mdc}\|g_{k-1}\| - 2\kappa_{\rm ubh}\Delta_{k-1} > 0$ by (3.20). Therefore, $\tilde{\rho}_k \geqslant \tilde{\eta}_2$ and (2.3) then ensures that (3.18) holds. □

It is therefore guaranteed that the trust-region radius can not be decreased indefinitely if the current iterate is not near critically. This is ensured by the next theorem.

**Theorem 3.4** Suppose that A.1–A.4 hold. Suppose furthermore that there exists a constant $\kappa_{\rm lbg}$ such that $\|g_k\| \geqslant \kappa_{\rm lbg}$ for all $k$. Then there is a constant $\kappa_{\rm lbd}$ such that

$$\Delta_k \geqslant \kappa_{\rm lbd} \tag{3.24}$$

for all $k$.

**Proof.** The proof is the same as for Theorem 6.4.3 in Conn et al. (2000) except that

$$\kappa_{\rm lbd} = \min\left[1 - \eta_1, \frac{(1 - \tilde{\eta}_2)}{(3 - 2\tilde{\eta}_2)}\right]\frac{\gamma_1\kappa_{\rm mdc}\kappa_{\rm lbg}}{\kappa_{\rm ubh}}.$$

□

>From here on, the proof for the basic trust region applies without change. We first deduce the global convergence of the algorithm to first-order critical points when it generates only finitely many successful iterations.

**Theorem 3.5** Suppose that A.1–A.4 hold. Suppose furthermore that there are only finitely many successful iterations. Then $x_k = x_*$ for all sufficiently large $k$ and $x_*$ is first-order critical.

**Proof.** The same argument as in Theorem 6.4.4 in Conn et al. (2000) may be applied since the radius update is identical to that of the basic trust region method for unsuccessful iterations. □

Finally, the next two results ensure the global convergence of the algorithm to first-order critical points, by showing in a first step that at least one accumulation point of the iterates sequence is first-order critical.

**Theorem 3.6** Suppose that A.1–A.4 hold. Then one has that

$$\liminf_{k\to\infty} \|\nabla_x f(x_k)\| = 0. \tag{3.25}$$

**Proof.** See Theorem 6.4.5 in Conn et al. (2000). □

As for the basic trust-region method, this can be extended to show that all limit points are first-order critical.

**Theorem 3.7** Suppose that A.1–A.4 hold. Then one has that

$$\lim_{k\to\infty} \|\nabla_x f(x_k)\| = 0. \tag{3.26}$$

**Proof.** See Theorem 6.4.6 in Conn et al. (2000). □

## 3.2 Convergence to Second-Order Critical Points

We now investigate the possibility to exploit second-order information on the objective function, with the aim of ensuring convergence to second-order critical points, i.e. points $x_*$ such that

$$\nabla_x f(x_*) = 0 \quad \text{and} \quad \nabla_{xx} f(x_*) \text{ is positive semidefinite.}$$

Of course, we need to clarify what we precisely mean by "second-order information". We therefore introduce the following additional assumptions:

**A.5** The model is asymptotically second-order coherent with the objective function near first-order critical points, i.e.

$$\lim_{k\to\infty} \|\nabla_{xx} f(x_k) - \nabla_{xx} m_k(x_k)\| = 0 \text{ whenever } \lim_{k\to\infty} \|g_k\| = 0.$$

**A.6** The Hessian of every model $m_k$ is Lipschitz continuous, that is, there exists a constant $\kappa_{\rm lch}$ such that, for all $k$,

$$\|\nabla_{xx} m_k(x) - \nabla_{xx} m_k(y)\| \leqslant \kappa_{\rm lch} \|x - y\|$$

for all $x, y \in \mathcal{B}_k$.

**A.7** If the smallest eigenvalue $\tau_k$ of the Hessian of the model $m_k$ at $x_k$ is negative, then

$$m_k(x_k) - m_k(x_k + s_k) \geqslant \kappa_{\rm sod}|\tau_k| \min(\tau_k^2, \Delta_k^2)$$

for some constant $\kappa_{\rm sod} \in (0, \frac{1}{2})$.

These assumptions are identical to those used in Sections 6.5 and 6.6 of Conn et al. (2000) for the basic trust-region method. In fact, the second-order convergence properties of the retrospective trust-region method also turn out to be exactly the same as those of the basic trust-region method, and their proofs can essentially been borrowed from this case, with the exception of Lemma 6.5.3. We therefore need to present a proof of that particular result for the new method. As we indicate below, all other results generalize without change and we only mention them for the sake of clarity.

In our analog of Lemma 6.5.3, we assume that the model reduction is eventually significant in the sense that it is at least of the same order as the error between the model and the objective function. We then show that the trust-region radius becomes asymptotically irrelevant if the steps tend to zero.

**Lemma 3.8** Suppose that A.1–A.3, and A.5 hold. Suppose also that there exists a sequence $(k_i)$ and a constant $\kappa_{\rm mqd} > 0$ such that

$$m_{k_i}(x_{k_i}) - m_{k_i}(x_{k_i} + s_{k_i}) \geqslant \kappa_{\rm mqd}\|s_{k_i}\|^2 > 0 \tag{3.27}$$

for all $i$ sufficiently large. Finally, suppose that

$$\lim_{i\to\infty} \|s_{k_i}\| = 0.$$

Then iteration $k_i$ is successful and

$$\tilde{\rho}_{k_i+1} \geqslant \tilde{\eta}_2 \;\text{ and }\; \Delta_{k_i+1} \geqslant \Delta_{k_i} \tag{3.28}$$

for $i$ sufficiently large.

**Proof.** We first apply Lemma 6.5.3 of Conn et al. (2000) to deduce that every iteration $k_i$ is successful for $i$ sufficiently large. Now, consider $k_i$ one such iteration. The equations (3.10) and (3.9) imply that for some $\xi_{k_i+1}$ and $\zeta_{k_i+1}$ in the segment $[x_{k_i}, x_{k_i+1}]$,

$$\begin{aligned} |\tilde{\rho}_{k_i+1} - 1| &= \left|\frac{f(x_{k_i}) - m_{k_i+1}(x_{k_i})}{m_{k_i+1}(x_{k_i}) - m_{k_i+1}(x_{k_i+1})}\right| \\ &= \left|\frac{\langle s_{k_i}, \nabla_{xx} f(\xi_{k_i+1}) s_{k_i}\rangle - \langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle}{-\langle s_{k_i}, g_{k_i+1}\rangle + \frac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle}\right| \\ &\leqslant \frac{\|s_{k_i}\|^2 \cdot \|\nabla_{xx} f(\xi_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\|}{|-\langle s_{k_i}, g_{k_i+1}\rangle + \frac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle|} \end{aligned} \tag{3.29}$$

where we also used the Cauchy-Schwarz inequality. By substituting $g_{k_i+1} = \nabla_x f(x_{k_i+1})$ (because of A.2) with its expression in (3.15), the denominator $D$ of the latter fraction can be rewritten as

$$D = \left| - \left\langle s_{k_i}, g_{k_i} + \int_0^1 \nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) s_{k_i}\, d\alpha \right\rangle + \tfrac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle \right|.$$

Then, replacing $-\langle s_{k_i}, g_{k_i}\rangle$ by its expression in (3.12), we obtain

$$\begin{aligned} D &= \Big| m_{k_i}(x_{k_i}) - m_{k_i}(x_{k_i+1}) + \tfrac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i}(\psi_{k_i}) s_{k_i}\rangle \\ &\quad + \tfrac{1}{2}\langle s_{k_i}, \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1}) s_{k_i}\rangle - \left\langle s_{k_i}, \int_0^1 \nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) s_{k_i}\, d\alpha \right\rangle \Big| \end{aligned}$$

for some $\psi_{k_i}$ in the segment $[x_{k_i}, x_{k_i+1}]$. The triangle inequality, properties of the integral, (3.27), and Cauchy-Schwarz inequality give therefore the following lower bound on $D$:

$$\begin{aligned} D &\geqslant |m_{k_i}(x_{k_i}) - m_{k_i}(x_{k_i+1})| \\ &\quad -\tfrac{1}{2}\Big| \left\langle s_{k_i}, \int_0^1 [\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})] s_{k_i}\, d\alpha \right\rangle \\ &\qquad + \left\langle s_{k_i}, \int_0^1 [\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})] s_{k_i}\, d\alpha \right\rangle \Big| \\ &\geqslant \kappa_{\rm mqd} \|s_{k_i}\|^2 - \tfrac{1}{2}\|s_{k_i}\| \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| \cdot \|s_{k_i}\|\, d\alpha \\ &\qquad - \tfrac{1}{2}\|s_{k_i}\| \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| \cdot \|s_{k_i}\|\, d\alpha \\ &\geqslant \|s_{k_i}\|^2 (\kappa_{\rm mqd} - \tfrac{1}{2}\epsilon_i) \end{aligned} \tag{3.30}$$

where

$$\epsilon_i \stackrel{\rm def}{=} \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\|\, d\alpha + \int_0^1 \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\|\, d\alpha.$$

The triangle inequality now implies that

$$\begin{aligned} \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| &\leqslant \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} f(x_{k_i})\| \\ &\quad + \|\nabla_{xx} f(x_{k_i}) - \nabla_{xx} m_{k_i}(x_{k_i})\| + \|\nabla_{xx} m_{k_i}(x_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| \end{aligned} \tag{3.31}$$

and, similarly, that

$$\begin{aligned} \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| &\leqslant \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} f(x_{k_i+1})\| \\ + \|\nabla_{xx} f(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(x_{k_i+1})\| &+ \|\nabla_{xx} m_{k_i+1}(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\|. \end{aligned} \tag{3.32}$$

Since we now observe that

$$\begin{array}{cc} \|(x_{k_i} + \alpha s_{k_i}) - x_{k_i}\| \leqslant \|s_{k_i}\|, & \|\psi_{k_i} - x_{k_i}\| \leqslant \|s_{k_i}\|, \\ \|(x_{k_i} + \alpha s_{k_i}) - x_{k_i+1}\| \leqslant \|s_{k_i}\|, & \|\zeta_{k_i+1} - x_{k_i+1}\| \leqslant \|s_{k_i}\|, \end{array}$$

we may deduce that both

$$\|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i}(\psi_{k_i})\| \text{ and } \|\nabla_{xx} f(x_{k_i} + \alpha s_{k_i}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\|$$

converge to zero with $\|s_{k_i}\|$ because the first and third terms of the right-hand side of (3.31) and (3.32) tend to zero by continuity of the the objective function's and model's Hessians, and because the middle term in the right-hand side of these inequalities also converges to zero because of A.5 and Theorem 3.7. As a consequence, $\epsilon_i \leqslant \kappa_{\rm mqd}$ when $i$ is sufficiently large, and therefore, combining (3.29) and (3.30), and using the triangle inequality, we obtain

$$\begin{aligned}
|\tilde{\rho}_{k_i+1} - 1| &\leqslant \frac{2}{\kappa_{\rm mqd}} \|\nabla_{xx} f(\xi_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| \\
&\leqslant \frac{2}{\kappa_{\rm mqd}} \Big[ \|\nabla_{xx} f(\xi_{k_i+1}) - \nabla_{xx} f(x_{k_i+1})\| \\
&\qquad + \|\nabla_{xx} f(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(x_{k_i+1})\| \\
&\qquad + \|\nabla_{xx} m_{k_i+1}(x_{k_i+1}) - \nabla_{xx} m_{k_i+1}(\zeta_{k_i+1})\| \Big]
\end{aligned} \tag{3.33}$$

By the same reasoning as for (3.31)–(3.32), the right-hand side of (3.33) tends to zero when $i$ goes to infinity, and $\tilde{\rho}_{k_i+1}$ therefore tends to 1. It is thus larger than $\tilde{\eta}_2 < 1$ for $i$ sufficiently large and (3.28) follows. □

As in Lemma 6.5.4 of Conn et al. (2000), we may apply this result to the entire sequence of iterates and deduce that all iterations are eventually successful and the trust-region radius bounded away from zero.

>From here on, the theory in Conn et al. (2000) generalizes without significant change, yielding the following results.

**Theorem 3.9** Suppose that A.1–A.5 hold and that $x_{k_i}$ is a subsequence of the iterates generated by Algorithm RTR converging to a first-order critical point $x_*$ where the Hessian of the objective function $\nabla_{xx} f(x_*)$ is positive definite. Suppose furthermore that $s_k \neq 0$ for all $k$ sufficiently large. Then the complete sequence of iterates converges to $x_*$, all iterations are eventually very successful, and the trust-region radius $\Delta_k$ is bounded away from zero.

**Proof.** See Theorem 6.5.5 in Conn et al. (2000). □

We now proof that if the sequence of iterates remains in a compact set, then the existence of at least one second-order critical accumulation point is guaranteed.

**Theorem 3.10** Suppose that A.1–A.7 hold and that all iterates remain in some compact set. Then there exists at least one limit point $x_*$ of the sequence of iterates $x_k$ produced by Algorithm RTR, which is second-order critical.

**Proof.** See Theorem 6.6.5 in Conn et al. (2000). □

By just strengthening the radius update rule by requiring that

$$\text{if } \tilde{\rho}_k \geqslant \tilde{\eta}_2 \text{ and } \Delta_k \leqslant \Delta_{\max}, \text{ then } \Delta_{k+1} \in [\gamma_3 \Delta_k, \gamma_4 \Delta_k] \tag{3.34}$$

for some $\gamma_4 \geqslant \gamma_3 > 1$ and some $\Delta_{\max} > 0$, we moreover obtain the second-order criticality of any limit point of the sequence of iterates generated by Algorithm RTR.

**Theorem 3.11** Suppose that A.1–A.7, and (3.34) hold and let $x_*$ be any limit point of the sequence of iterates. Then $x_*$ is a second-order critical point.

**Proof.** See Theorem 6.6.8 in Conn et al. (2000). □

Thus the retrospective trust-region algorithm shares all the (interesting) convergence properties of the basic trust-region method under the same assumptions. We conclude this theory section by noting that the above convergence results are still valid if one replaces the Euclidean norm by any (possibly iteration dependent) uniformly equivalent norm, thereby allowing problem scaling and preconditioning.

## 4 Preliminary numerical experience

We now consider the numerical behaviour of the new algorithm, in comparison with the basic trust-region algorithm BTR (see page 116 of Conn et al. (2000)). We test both algorithms on all of the 146 unconstrained problems of the CUTEr collection (see Gould, Orban and Toint, 2003). For the problems whose dimension may be changed, we chose a reasonably small value in order not to overload the CUTEr interface with MATLAB. The starting points are the standard ones provided by the CUTEr library.

For the basic algorithm, the trust-region radius update was implemented by using the simple rule (and the corresponding parameters) proposed in Conn et al. (2000), p. 782:

$$\Delta_{k+1} = \begin{cases} \max(\alpha_1\|s_k\|, \Delta_k) & \text{if } \rho_k \geqslant \eta_2, \\ \Delta_k & \text{if } \rho_k \in [\eta_1, \eta_2), \\ \alpha_2\|s_k\| & \text{if } \rho_k < \eta_1, \end{cases}$$

where $\alpha_1$ is fixed at 2.5, $\alpha_2$ at 0.25, $\eta_1$ at 0.05 and $\eta_2$ at 0.9. To avoid biasing the comparison, we fave decided to make as few adaptations as possible to that rule in our retrospective variant (i.e. Step 2 in Algorithm 2.1). Thus, if iteration $k$ is unsuccessful, i.e. $\rho_k < \eta_1$ and consequently $x_k = x_{k+1}$, we also decrease the trust-region to $\Delta_{k+1} = \alpha_2\|s_k\|$. If, on the contrary, iteration $k$ is successful, i.e. $\rho_k \geqslant \eta_1$, the trust-region is updated as follows:

$$\Delta_{k+1} = \begin{cases} \max(\alpha_1\|s_k\|, \Delta_k) & \text{if } \tilde{\rho}_{k+1} \geqslant \tilde{\eta}_2, \\ \Delta_k & \text{if } \tilde{\rho}_{k+1} \in [\tilde{\eta}_1, \tilde{\eta}_2), \\ \alpha_2\|s_k\| & \text{if } \tilde{\rho}_{k+1} < \tilde{\eta}_1. \end{cases}$$

where we choose the same values as above for $\alpha_1$ and $\alpha_2$, and take $\tilde{\eta}_1 = \eta_1 = 0.05$ and $\tilde{\eta}_2 = \eta_2 = 0.9$. The model was chosen, in both cases, to be the exact Taylor's series truncated to second-order, and the exact minimizer of this model inside the trust-region, was computed using the Moré-Sorensen algorithm (see Moré and Sorensen, 1983).

We considered that the iterative process converged when the Euclidean norm of the gradient became smaller than $10^{-5}$. Failure was declared if the algorithm did not converge within the maximum number of 100 000 iterations.

We chose to compare the number of iterations to achieve convergence instead of the CPU time or number of function evaluations. Indeed, the cost per iteration is the same for both algorithms and they both evaluate the objective function once per iteration and compute one gradient at every successful iteration. Moreover, timings in MATLAB are often difficult to interpret.

All runs were performed in Matlab v. 7.1.0.183 (R14) Service Pack 3 on a 3.2 Ghz Intel single-core processor computer with 2 GB of RAM. The Figure 4.1 represents the comparison by a performance profile (see Dolan and Moré, 2002) of the number of iterations of the two algorithms. In this figure, we have only kept the problems for which both algorithms converged to the same local solution (we excluded `BROYDN7D`, `FLETCHBV`, `NONCVXU2` and `NONCVXUN`).

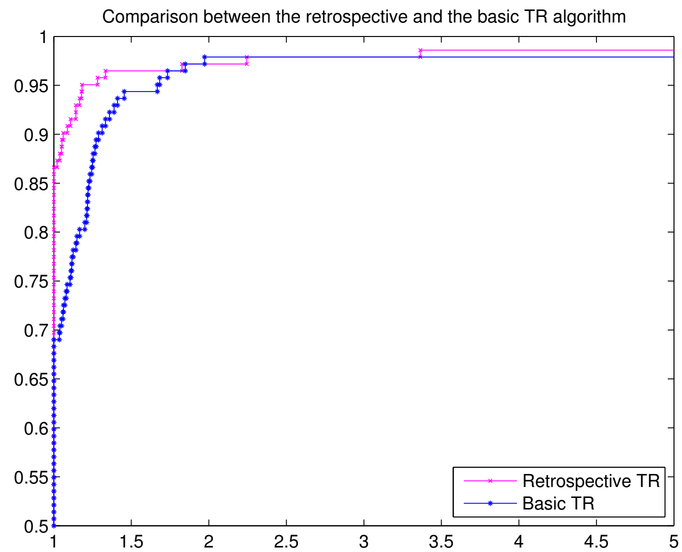


Figure 4.1: Performance profile comparing the number of iterations of the RTR and BTR algorithms

Our results show that the retrospective algorithm is overall more efficient than the classical one, and just as reliable. Both algorithms failed on `MEYER3`, a problem well-known for its extreme conditioning, and on `FLETCBV3`. On the other hand, BTR failed on `SCOSINE`, which was solved by RTR.

## 5 Conclusion and perspectives

We have introduced a natural variant of the basic trust-region algorithm, where the most recent model information is exploited to update the trust-region radius. We have also shown that limit points of sequences of iterates produced by the new algorithm are second-order critical points for the minimization problem. Our preliminary numerical experiments indicate that the method is very competitive and deserves further study.

This new method is especially interesting for adaptive techniques which exploit the information made available during the optimization process in order to vary the accuracy of the objective function computation. These methods typically appear in the context of a noisy objective function, where noise reduction can be achieved but at a significant cost. We therefore assume that the error can be estimated and consequently maintained under some acceptable threshold, while at the same time keeping the computational cost as low as possible. A first trust-region method with dynamic accuracy is described in Section 10.6 of Conn et al. (2000). The main idea there is to impose a model reduction larger than some multiple of the noise evaluated at both the current and candidate iterates. A cheaper nonmonotone approach has been developed in the context of nonlinear stochastic programming by Bastin, Cirillo and Toint (2006*a*), (see also Bastin, Cirillo and Toint, 2006*b*) more specifically for the minimization of sample average approximations (Shapiro 2003) relying on Monte-Carlo sampling, a method also known as sample-path optimization (Robinson, 1996). The main difference with respect to the work of Conn *et al.* is that it allows a reduction of the model smaller than the noise level. In both cases, the size of the model reduction is the main component to decide on the desired accuracy of the objective function: the adaptive mechanism is thus applied on the basis of past information, at the previous iterate, rather that at the current one. Our new proposal could therefore improve these techniques significantly because it uses the most relevant information on the model's quality at the current iterate instead of at the previous iterate, but this remains to be analyzed further.

Other applications of the same idea are also possible across the wide class of trust-region methods, constrained and unconstrained.

## References

F. Bastin, C. Cirillo, and Ph. L. Toint. An adaptive Monte-Carlo algorithm for computing mixed logit estimators. *Computational Management Science*, **3**(1), 55–80, 2006*a*.

F. Bastin, C. Cirillo, and Ph. L. Toint. Application of an adaptive Monte-Carlo algorithm to mixed logit estimation. *Transportation Research B*, **40**(7), 577–593, 2006*b*.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. Number 01 *in* 'MPS-SIAM Series on Optimization'. SIAM, Philadelphia, USA, 2000.

E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, **91**(2), 201–213, 2002.

N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr, a constrained and unconstrained testing environment, revisited. *ACM Transactions on Mathematical Software*, **29**(4), 373–394, 2003.

J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing*, **4**(3), 553–572, 1983.

S. M. Robinson. Analysis of sample-path optimization. *Mathematics of Operations Research*, **21**(3), 513–528, 1996.

A. Shapiro. Monte Carlo sampling methods. *in* A. Shapiro and A. Ruszczyński, eds, 'Stochastic Programming', Vol. 10 of *Handbooks in Operations Research and Management Science*, pp. 353–425. Elsevier, Amsterdam, The Netherlands, 2003.

## Appendix A

Here is the set of results from our tests. For each problem, we report its number of variables ($n$), the number of iterations ($iter$), the number of gradient evaluations ($\#g$) and the best objective function value found ($f$). The symbol $>$ indicates that the iteration limit (fixed at 100 000) was exceeded.

| | | BTR | | | RTR | | |
|---|---|---|---|---|---|---|---|
| Name | n | iter | #g | f | iter | #g | f |
| AKIVA | 2 | 6 | 7 | 6.1660e+00 | 6 | 7 | 6.1660e+00 |
| ALLINITU | 4 | 7 | 8 | 5.7444e+00 | 7 | 8 | 5.7444e+00 |
| ARGLINA | 200 | 5 | 6 | 2.0000e+02 | 5 | 6 | 2.0000e+02 |
| ARWHEAD | 100 | 5 | 6 | 6.5947e−14 | 5 | 6 | 6.5947e−14 |
| BARD | 3 | 9 | 9 | 8.2149e−03 | 9 | 9 | 8.2149e−03 |
| BDQRTIC | 100 | 10 | 11 | 3.7877e+02 | 10 | 11 | 3.7877e+02 |
| BEALE | 2 | 9 | 9 | 1.9232e−16 | 8 | 8 | 4.5813e−14 |
| BIGGS6 | 6 | 617 | 482 | 2.4268e−01 | 444 | 360 | 2.4269e−01 |
| BOX3 | 3 | 7 | 8 | 1.5192e−11 | 7 | 8 | 1.5192e−11 |

| | | BTR | | | RTR | | |
|---|---|---|---|---|---|---|---|
| Name | n | iter | #g | f | iter | #g | f |
| BRKMCC | 2 | 2 | 3 | 1.6904e−01 | 2 | 3 | 1.6904e−01 |
| BROWNAL | 200 | 27 | 23 | 2.4548e−13 | 25 | 21 | 7.9812e−23 |
| BROWNBS | 2 | 29 | 29 | 0.0000e+00 | 28 | 28 | 4.9304e−30 |
| BROWNDEN | 4 | 10 | 11 | 8.5822e+04 | 10 | 11 | 8.5822e+04 |
| BROYDN7D | 100 | 27 | 23 | 3.9739e+01 | 22 | 20 | 3.9867e+01 |
| BRYBND | 100 | 12 | 11 | 5.0820e−19 | 14 | 12 | 1.8559e−17 |
| CHAINWOO | 100 | 62 | 52 | 1.0000e+00 | 49 | 45 | 1.0000e+00 |
| CHNROSNB | 50 | 67 | 57 | 3.9250e−14 | 55 | 54 | 6.3584e−14 |
| CLIFF | 2 | 27 | 28 | 1.9979e−01 | 27 | 28 | 1.9979e−01 |
| COSINE | 100 | 6 | 7 | -9.9000e+01 | 6 | 7 | -9.9000e+01 |
| CRAGGLVY | 202 | 15 | 16 | 6.6741e+01 | 15 | 16 | 6.6741e+01 |
| CUBE | 2 | 40 | 33 | 8.0930e−16 | 35 | 30 | 1.3070e−14 |
| CURLY10 | 50 | 9 | 10 | -5.0158e+03 | 9 | 10 | -5.0158e+03 |
| CURLY20 | 50 | 8 | 9 | -5.0158e+03 | 8 | 9 | -5.0158e+03 |
| CURLY30 | 50 | 13 | 13 | -5.0158e+03 | 13 | 13 | -5.0158e+03 |
| DECONVU | 61 | 19 | 14 | 1.3203e−08 | 20 | 15 | 1.5318e−08 |
| DENSCHNA | 2 | 5 | 6 | 2.2139e−12 | 5 | 6 | 2.2139e−12 |
| DENSCHNB | 2 | 4 | 5 | 3.3850e−16 | 4 | 5 | 3.3850e−16 |
| DENSCHNC | 2 | 10 | 11 | 2.1777e−20 | 10 | 11 | 2.1777e−20 |
| DENSCHND | 3 | 40 | 34 | 6.5710e−09 | 33 | 29 | 9.2866e−08 |
| DENSCHNE | 3 | 9 | 10 | 8.7102e−19 | 9 | 10 | 8.7102e−19 |
| DENSCHNF | 2 | 6 | 7 | 6.5132e−22 | 6 | 7 | 6.5132e−22 |
| DIXMAANA | 150 | 7 | 8 | 1.0000e+00 | 7 | 8 | 1.0000e+00 |
| DIXMAANB | 150 | 14 | 13 | 1.0000e+00 | 14 | 13 | 1.0000e+00 |
| DIXMAANC | 150 | 10 | 10 | 1.0000e+00 | 10 | 10 | 1.0000e+00 |
| DIXMAAND | 150 | 12 | 11 | 1.0000e+00 | 12 | 11 | 1.0000e+00 |
| DIXMAANE | 150 | 17 | 15 | 1.0000e+00 | 14 | 13 | 1.0000e+00 |
| DIXMAANF | 150 | 18 | 15 | 1.0000e+00 | 15 | 13 | 1.0000e+00 |
| DIXMAANG | 150 | 16 | 14 | 1.0000e+00 | 16 | 14 | 1.0000e+00 |
| DIXMAANH | 150 | 20 | 17 | 1.0000e+00 | 15 | 14 | 1.0000e+00 |
| DIXMAANI | 150 | 14 | 13 | 1.0000e+00 | 14 | 13 | 1.0000e+00 |
| DIXMAANJ | 150 | 19 | 16 | 1.0000e+00 | 19 | 16 | 1.0000e+00 |
| DIXMAANK | 150 | 21 | 18 | 1.0000e+00 | 16 | 15 | 1.0000e+00 |
| DIXMAANL | 150 | 20 | 16 | 1.0000e+00 | 20 | 16 | 1.0000e+00 |
| DIXON3DQ | 100 | 4 | 5 | 1.1402e−29 | 4 | 5 | 1.1402e−29 |
| DJTL | 2 | 120 | 89 | -8.9515e+03 | 112 | 85 | -8.9515e+03 |
| DQDRTIC | 100 | 5 | 6 | 5.9658e−29 | 5 | 6 | 5.9658e−29 |
| DQRTIC | 100 | 29 | 30 | 2.8059e−08 | 29 | 30 | 2.8059e−08 |
| EDENSCH | 100 | 24 | 19 | 6.0328e+02 | 24 | 20 | 6.0328e+02 |
| EG2 | 100 | 3 | 4 | -9.8947e+01 | 3 | 4 | -9.8947e+01 |
| EIGENALS | 110 | 20 | 21 | 5.0766e−21 | 21 | 21 | 5.0350e−24 |
| EIGENBLS | 110 | 92 | 72 | 7.1205e−12 | 118 | 104 | 1.3990e−13 |
| ENGVAL1 | 100 | 9 | 10 | 1.0909e+02 | 9 | 10 | 1.0909e+02 |
| ENGVAL2 | 3 | 13 | 14 | 9.7152e−17 | 13 | 14 | 9.7152e−17 |
| ERRINROS | 50 | 47 | 43 | 3.9904e+01 | 48 | 45 | 3.9904e+01 |
| EXPFIT | 2 | 7 | 6 | 2.4051e−01 | 7 | 6 | 2.4051e−01 |
| EXTROSNB | 100 | 685 | 554 | 8.2403e−07 | 504 | 484 | 2.9412e−07 |
| FLETCBV2 | 100 | 2 | 3 | -5.1401e−01 | 2 | 3 | -5.1401e−01 |
| FLETCBV3 | 50 | > | > | -1.1173e+03 | > | > | -1.1072e+03 |
| FLETCHBV | 10 | 508 | 498 | -2.2561e+06 | 817 | 810 | -2.0573e+06 |
| FLETCHCR | 100 | 357 | 284 | 3.5522e−14 | 214 | 212 | 6.8069e−15 |
| FMINSRF2 | 121 | 26 | 23 | 1.0000e+00 | 25 | 22 | 1.0000e+00 |
| FMINSURF | 121 | 30 | 26 | 1.0000e+00 | 27 | 22 | 1.0000e+00 |
| FREUROTH | 100 | 9 | 10 | 1.1965e+04 | 9 | 10 | 1.1965e+04 |
| GENHUMPS | 10 | 10280 | 9680 | 3.5566e−12 | 11744 | 11060 | 3.4200e−15 |
| GENROSE | 100 | 123 | 99 | 1.0000e+00 | 107 | 96 | 1.0000e+00 |
| GENROSEB | 500 | 515 | 424 | 1.0000e+00 | 354 | 338 | 1.0000e+00 |
| GROWTHLS | 3 | 112 | 92 | 1.0040e+00 | 92 | 82 | 1.0040e+00 |
| GULF | 3 | 26 | 23 | 4.8518e−13 | 27 | 27 | 1.1954e−11 |
| HAIRY | 2 | 58 | 53 | 2.0000e+01 | 106 | 96 | 2.0000e+01 |
| HATFLDD | 3 | 20 | 20 | 6.6151e−08 | 20 | 20 | 6.6151e−08 |

| | | BTR | | | RTR | | |
|---|---|---|---|---|---|---|---|
| Name | n | iter | #g | f | iter | #g | f |
| HATFLDE | 3 | 21 | 21 | 5.1204e−07 | 21 | 21 | 5.1204e−07 |
| HEART6LS | 6 | 845 | 794 | 7.6247e−24 | 689 | 676 | 3.5331e−22 |
| HEART8LS | 8 | 105 | 84 | 2.9586e−15 | 86 | 74 | 1.0250e−18 |
| HELIX | 3 | 10 | 10 | 1.7058e−15 | 8 | 8 | 4.9599e−13 |
| HIELOW | 3 | 8 | 8 | 8.7417e+02 | 8 | 8 | 8.7417e+02 |
| HILBERTA | 2 | 3 | 4 | 8.2173e−33 | 3 | 4 | 8.2173e−33 |
| HILBERTB | 10 | 3 | 4 | 7.9218e−30 | 3 | 4 | 7.9218e−30 |
| HIMMELBB | 2 | 14 | 11 | 6.1873e−29 | 11 | 8 | 1.2467e−20 |
| HIMMELBF | 4 | 116 | 115 | 3.1857e+02 | 94 | 92 | 3.1857e+02 |
| HIMMELBG | 2 | 5 | 6 | 9.0327e−12 | 5 | 6 | 9.0327e−12 |
| HIMMELBH | 2 | 4 | 5 | -1.0000e+00 | 4 | 5 | -1.0000e+00 |
| HUMPS | 2 | 2466 | 2329 | 2.3594e−10 | 5536 | 5266 | 3.6245e−11 |
| JENSMP | 2 | 9 | 10 | 1.2436e+02 | 9 | 10 | 1.2436e+02 |
| KOWOSB | 4 | 10 | 9 | 3.0780e−04 | 10 | 9 | 3.0780e−04 |
| LIARWHD | 100 | 12 | 13 | 5.5677e−14 | 12 | 13 | 5.5677e−14 |
| LOGHAIRY | 2 | 2777 | 2730 | 1.8232e−01 | 9343 | 8523 | 1.8232e−01 |
| MANCINO | 100 | 14 | 15 | 1.5301e−21 | 16 | 15 | 1.8618e−21 |
| MARATOSB | 2 | 850 | 777 | -1.0000e+00 | 678 | 668 | -1.0000e+00 |
| MEXHAT | 2 | 31 | 30 | -4.0010e−02 | 31 | 30 | -4.0010e−02 |
| MEYER3 | 3 | > | > | 8.7946e+01 | > | > | 8.7946e+01 |
| MODBEALE | 200 | 10 | 11 | 7.8240e−21 | 10 | 11 | 7.8240e−21 |
| MOREBV | 100 | 1 | 2 | 7.8870e−10 | 1 | 2 | 7.8870e−10 |
| MSQRTALS | 100 | 21 | 17 | 2.3642e−12 | 19 | 17 | 2.7268e−19 |
| MSQRTBLS | 100 | 19 | 16 | 1.6184e−13 | 17 | 15 | 7.1379e−16 |
| NONCVXU2 | 100 | 47 | 42 | 2.3284e+02 | 51 | 40 | 2.3227e+02 |
| NONCVXUN | 100 | 45 | 39 | 2.3733e+02 | 38 | 31 | 2.3285e+02 |
| NONDIA | 100 | 6 | 7 | 1.4948e−18 | 6 | 7 | 1.4948e−18 |
| NONDQUAR | 100 | 15 | 16 | 2.6991e−09 | 15 | 16 | 2.6991e−09 |
| OSBORNEA | 5 | 38 | 32 | 5.4649e−05 | 35 | 30 | 5.4649e−05 |
| OSBORNEB | 11 | 18 | 17 | 4.0138e−02 | 17 | 16 | 4.0138e−02 |
| OSCIPATH | 8 | 2682 | 2252 | 1.7742e−05 | 2156 | 1954 | 1.5328e−05 |
| PALMER1C | 8 | 7 | 8 | 9.7605e−02 | 7 | 8 | 9.7605e−02 |
| PALMER1D | 7 | 7 | 8 | 6.5267e−01 | 7 | 8 | 6.5267e−01 |
| PALMER2C | 8 | 6 | 7 | 1.4369e−02 | 6 | 7 | 1.4369e−02 |
| PALMER3C | 8 | 6 | 7 | 1.9538e−02 | 6 | 7 | 1.9538e−02 |
| PALMER4C | 8 | 7 | 8 | 5.0311e−02 | 7 | 8 | 5.0311e−02 |
| PALMER5C | 6 | 5 | 6 | 2.1281e+00 | 5 | 6 | 2.1281e+00 |
| PALMER6C | 8 | 7 | 8 | 1.6387e−02 | 7 | 8 | 1.6387e−02 |
| PALMER7C | 8 | 9 | 10 | 6.0199e−01 | 9 | 10 | 6.0199e−01 |
| PALMER8C | 8 | 8 | 9 | 1.5977e−01 | 8 | 9 | 1.5977e−01 |
| PENALTY1 | 100 | 45 | 44 | 9.0249e−04 | 45 | 44 | 9.0249e−04 |
| PENALTY2 | 100 | 19 | 20 | 9.7096e+04 | 19 | 20 | 9.7096e+04 |
| PFIT1LS | 3 | 536 | 432 | 9.0700e−13 | 290 | 277 | 3.8991e−15 |
| PFIT2LS | 3 | 175 | 140 | 7.7708e−15 | 104 | 93 | 4.9517e−16 |
| PFIT3LS | 3 | 221 | 177 | 3.5764e−14 | 112 | 102 | 1.3794e−15 |
| PFIT4LS | 3 | 404 | 330 | 9.9629e−21 | 233 | 223 | 3.6519e−20 |
| POWELLSG | 4 | 15 | 16 | 4.6333e−09 | 15 | 16 | 4.6333e−09 |
| POWER | 100 | 24 | 25 | 1.1818e−09 | 24 | 25 | 1.1818e−09 |
| QUARTC | 100 | 29 | 30 | 2.8059e−08 | 29 | 30 | 2.8059e−08 |
| ROSENBR | 2 | 29 | 26 | 1.8013e−23 | 26 | 24 | 2.9201e−14 |
| S308 | 2 | 11 | 10 | 7.7320e−01 | 11 | 10 | 7.7320e−01 |
| SBRYBND | 100 | 50 | 40 | 8.1982e−18 | 50 | 40 | 8.1982e−18 |
| SCHMVETT | 100 | 4 | 5 | -2.9400e+02 | 4 | 5 | -2.9400e+02 |
| SCOSINE | 100 | > | > | -9.9000e+01 | 97 | 90 | -9.9000e+01 |
| SCURLY10 | 100 | 39 | 35 | -1.0032e+04 | 46 | 42 | -1.0032e+04 |
| SCURLY20 | 100 | 34 | 30 | -1.0032e+04 | 37 | 33 | -1.0032e+04 |
| SCURLY30 | 100 | 35 | 31 | -1.0032e+04 | 35 | 31 | -1.0032e+04 |
| SENSORS | 100 | 21 | 21 | -1.9668e+03 | 28 | 26 | -1.9668e+03 |
| SINEVAL | 2 | 54 | 48 | 1.7176e−28 | 54 | 50 | 1.1634e−31 |
| SINQUAD | 100 | 9 | 10 | -4.0056e+03 | 9 | 10 | -4.0056e+03 |
| SISSER | 2 | 12 | 13 | 1.0658e−08 | 12 | 13 | 1.0658e−08 |

| | | BTR | | | RTR | | |
|---|---|---|---|---|---|---|---|
| Name | n | iter | #g | f | iter | #g | f |
| SNAIL | 2 | 62 | 61 | 3.3170e−16 | 59 | 60 | 1.2117e−14 |
| SPARSINE | 100 | 37 | 27 | 3.0507e−14 | 41 | 29 | 2.0419e−15 |
| SPARSQUR | 100 | 16 | 17 | 1.4795e−08 | 16 | 17 | 1.4795e−08 |
| SPMSRTLS | 100 | 14 | 13 | 8.3824e−15 | 14 | 13 | 8.3824e−15 |
| SROSENBR | 100 | 6 | 7 | 8.8993e−28 | 6 | 7 | 8.8993e−28 |
| TOINTGOR | 50 | 9 | 10 | 1.3739e+03 | 9 | 10 | 1.3739e+03 |
| TOINTGSS | 100 | 11 | 10 | 1.0102e+01 | 11 | 10 | 1.0102e+01 |
| TOINTPSP | 50 | 24 | 21 | 2.2556e+02 | 17 | 15 | 2.2556e+02 |
| TQUARTIC | 100 | 11 | 11 | 1.1878e−14 | 13 | 13 | 3.0400e−17 |
| VARDIM | 200 | 29 | 30 | 2.6246e−24 | 29 | 30 | 2.6246e−24 |
| VAREIGVL | 50 | 16 | 14 | 1.5304e−09 | 16 | 14 | 1.5304e−09 |
| VIBRBEAM | 8 | 49 | 38 | 1.5645e−01 | 52 | 41 | 1.5645e−01 |
| WATSON | 12 | 14 | 14 | 8.1544e−07 | 12 | 13 | 7.2604e−08 |
| WOODS | 4 | 67 | 56 | 8.3449e−15 | 52 | 47 | 1.4369e−21 |
| YFITU | 3 | 51 | 44 | 6.6698e−13 | 48 | 44 | 6.6697e−13 |