UNIVERSITÉ DE NAMUR

Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche

researchportal.unamur.be

RESEARCH OUTPUTS / RÉSULTATS DE RECHERCHE

Numerical experience with a recursive trust-region method for multilevel nonlinear bound-constrained optimization

Gratton, Serge; Mouffe, Mélodie; Sartenaer, Annick; Toint, Philippe; Tomanos, Dimitri

Published in:
Optimization Methods and Software

DOI:
10.1080/10556780903239295

Publication date:
2010

Document Version
Early version, also known as pre-print

Link to publication

Citation for pulished version (HARVARD):
Gratton, S, Mouffe, M, Sartenaer, A, Toint, P & Tomanos, D 2010, 'Numerical experience with a recursive trust-region method for multilevel nonlinear bound-constrained optimization' Optimization Methods and Software, vol. 25, no. 3, pp. 359-386. https://doi.org/10.1080/10556780903239295

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 21. May. 2019

Numerical Experience with a Recursive Trust-Region Method for Multilevel Nonlinear Optimization

by S. Gratton[1], M. Mouffe[1], A. Sartenaer[2], Ph. L. Toint[2] and D. Tomanos[2]

Report 08/10 22 June 2008

[1] CERFACS,
av. G. Coriolis, Toulouse, France,
Emails serge.gratton@cerfacs.fr, mouffe@cerfacs.fr

[2] Department of Mathematics,
University of Namur,
61, rue de Bruxelles, B-5000 Namur, Belgium,
Emails: annick.sartenaer@fundp.ac.be, philippe.toint@fundp.ac.be,
dimitri.tomanos@fundp.ac.be

Numerical Experience with a Recursive Trust-Region Method for Multilevel Nonlinear Bound-Constrained Optimization

Serge Gratton, Mélodie Mouffe, Annick Sartenaer,
Philippe L. Toint and Dimitri Tomanos

22 June 2008

Abstract

We consider an implementation of the recursive multilevel trust-region algorithm proposed by Gratton, Mouffe, Toint and Weber-Mendonça (2008) for bound-constrained nonlinear problems, and provide numerical experience on multilevel test problems. A suitable choice of the algorithm's parameters is identified on these problems, yielding a satisfactory compromise between reliability and efficiency. The resulting default algorithm is then compared to alternative optimization techniques such as mesh refinement and direct solution of the fine-level problem.

Keywords: nonlinear optimization, bound-constrained problems, multilevel problems, simplified models, recursive algorithms, numerical performance

1 Introduction

The optimization of finite-dimensional discretizations of problems in infinite dimensional spaces has become a very important area for numerical computations in the last years. New interest in surface design, data assimilation for weather forecasting (Fisher, 1998) or in optimal control of systems described by partial-differential equations have been the main motivation of this challenging research trend, but other applications such as multi-dimensional scaling (Bronstein, Bronstein, Kimmel and Yavneh, 2005) or quantization schemes (Emilianenko, 2005) also give rise to similar questions. In such problems, one typically considers a fine discretization of the infinite-dimensional problem which provides a sufficiently good approximation for the solution of the discretized problem to be of interest. But coarser discretizations are often available that still describe the problem reasonably well, and can therefore be used to improve the efficiency of the numerical solution on the fine discretization. This observation has been widely used for linear problems and has spawned the important field of multigrid methods (see Briggs, Henson and McCormick, 2000, for an introduction to this field).

In a recent paper, Gratton et al. (2008) discuss preliminary experimental efficiency and convergence properties of a new recursive multilevel trust-region algorithm for unconstrained and bound-constrained optimization, which is partly inspired by multigrid techniques, and by similar ideas in linesearch-based optimization methods by Fisher (1998) or Nash (2000) and Lewis and Nash (2005). The main feature of the new method is to allow the exploitation, in a trust-region framework, of the fact that many large-scale optimization problems have a hierarchy of different descriptions, possibly involving different number of variables. The considered algorithm differs from the earlier proposal by Gratton, Sartenaer and Toint (2007) in that the infinity-norm is used to define the recursive trust regions, which results in substantial algorithmic simplifications when compared to

the earlier algorithm using the Euclidean norm[(1)]. It also allows the incorporation of bound constraints into the problem, a feature missing in the earlier version.

The purpose of our paper is to present the numerical experience gained so far with a particular implementation of this algorithm applied to a small collection of significant test problems. We determine suitable default values for the parameters of the method, compare it to other competing methods in this area and illustrate what we believe is the strong potential of methods of this type.

2 A recursive multilevel trust-region method

We consider the bound-constrained optimization problem

$$\min_{x\in\mathcal{F}} f(x), \tag{2.1}$$

where f is a twice-continuously differentiable objective function which maps $\mathbb{R}^n$ into $\mathbb{R}$ and is bounded below, where $\mathcal{F} = \{x \in \mathbb{R}^n \,|\, l \leq x \leq u\}$ is a set of bound constraints and where $l, u \in \mathbb{R}^n$ and are possibly infinite. The trust-region method that we consider here is iterative, in the sense that, given an initial point x_0, it produces a sequence $\{x_k\}$ of iterates. At each iterate, such a method builds a model of $f(x)$ around x_k which is assumed to be adequate in a *trust region* centered at x_k and defined by its radius $\Delta_k > 0$. A step s_k, feasible with respect to $\mathcal{F}$, is then computed that induces a sufficient reduction in the model inside the trust region. The objective function is calculated at the *trial point*, $x_k + s_k$, and this trial point is accepted as the next iterate if and only if ρ_k, the ratio of achieved reduction (in the objective function) to predicted reduction (in its local model), is reasonable (typically larger than a small positive constant η_1). The radius of the trust region is finally updated: it is decreased if the trial point is rejected and left unchanged or increased if ρ_k is sufficiently large. Algorithms of this type are known to be reliable and efficient; we refer the reader to Conn, Gould and Toint (2000) for a comprehensive coverage of this subject.

Many practical trust-region algorithms, including that presented here, use a *quadratic model*

$$m_k(x_k + s) = f(x_k) + \langle g_k, s\rangle + \tfrac{1}{2}\langle s, H_k s\rangle, \tag{2.2}$$

where $g_k \stackrel{\rm def}{=} \nabla f(x_k)$, H_k is a symmetric $n \times n$ approximation of $\nabla^2 f(x_k)$, and $\langle\cdot,\cdot\rangle$ is the Euclidean inner product. A sufficient decrease in this model inside the trust region is then obtained by (approximately) solving

$$\begin{array}{ll}\min & m_k(x_k + s).\\ \|s\|_\infty \leq \Delta_k & \\ x_k + s \in \mathcal{F} & \end{array} \tag{2.3}$$

The choice of the infinity norm in the trust-region description is natural in the context of bound-constrained problems, because the feasible set for problem (2.3) can then be fully represented by bound constraints.

As proposed in Gratton et al. (2007) and further explored in Gratton et al. (2008), we consider exploiting the knowledge of a hierarchy of descriptions for problem (2.1), if such a hierarchy is known. To be more specific, suppose that a collection of functions $\{f_i\}_{i=0}^r$ is available, each f_i being a twice-continuously differentiable function from $\mathbb{R}^{n_i}$ to $\mathbb{R}$ (with $n_i \geq n_{i-1}$). We assume that $n_r = n$ and $f_r(x) = f(x)$ for all $x \in \mathbb{R}^n$, giving back our original problem. We also make the assumption that f_i is "more costly" to minimize than f_{i-1} for each $i = 1, \ldots, r$. This is typically the case if the f_i represent increasingly finer discretizations of the same infinite-dimensional objective. To fix terminology, we will refer to a particular i as a *level*. We use the first subscript i in all subsequent subscripted symbols to denote a quantity corresponding to the i-th level, ranging from coarsest ($i = 0$)

(1) We refer the reader to Gratton et al. (2008) for a more complete discussion of these advantages.

to finest ($i = r$) (meaning in particular, if applied to a vector, that this vector belongs to $\mathbb{R}^{n_i}$). Some relation must exist between the variables of two successive functions of the collection set $\{f_i\}_{i=0}^r$. We thus assume that, for each $i = 1, \ldots, r$, there exist a full-rank linear operator R_i from $\mathbb{R}^{n_i}$ into $\mathbb{R}^{n_{i-1}}$ (the restriction) and another full-rank linear operator P_i from $\mathbb{R}^{n_{i-1}}$ into $\mathbb{R}^{n_i}$ (the prolongation) such that

$$\sigma_i P_i = R_i^T, \tag{2.4}$$

for some known constant $\sigma_i > 0$, where P_i and R_i are interpreted as restriction and prolongation between a fine and a coarse grid (see, for instance, Briggs et al., 2000 for an excellent introduction or to Trottenberg, Oosterlee and Schüller, 2001, for a more extensive coverage). We assume that the restriction operators are normalized to ensure that $\|R_i\|_\infty = 1$ and also that the entries of R_i and P_i are all non-negative.

The philosophy of our recursive algorithm is then to use the hierarchy of problem descriptions $\{f_i\}_{i=0}^{r-1}$ to efficiently construct minimizations steps. More precisely, we build, for each level i, a model leading to a *local bound-constrained minimization subproblem* at the coarse level $i-1$, and then compute a coarse step by *solving this subproblem using a trust-region algorithm*. The resulting coarse move is then prolongated (using P_i) into a trust-region step at level i. For this purpose, we first need to build an alternative *local lower-level model* h_{i-1} representing at level $i-1$ the function h_i to be minimized at level i (with $h_r = f_r = f$). We also need to define a set of bound constraints which represents both the *feasibility* with respect to the original (finest level) bounds and the *constraints on the stepsize* inherited from the trust regions at level i as well as at levels $i+1, \ldots, r$.

Consider first the construction of the local lower-level model h_{i-1} of h_i around $x_{i,k}$, the iterate at some iteration k at level i, say. If we restrict $x_{i,k}$ to level $i-1$ and define $x_{i-1,0} = R_i x_{i,k}$, the model h_{i-1} is then given by

$$h_{i-1}(x_{i-1,0} + s_{i-1}) \stackrel{\rm def}{=} f_{i-1}(x_{i-1,0} + s_{i-1}) + \langle v_{i-1}, s_{i-1}\rangle, \tag{2.5}$$

where $v_{i-1} = R_i g_{i,k} - \nabla f_{i-1}(x_{i-1,0})$ with $g_{i,k} \stackrel{\rm def}{=} \nabla h_i(x_{i,k})$. By convention, we set $v_r = 0$, such that, for all s_r,

$$h_r(x_{r,0} + s_r) = f_r(x_{r,0} + s_r) = f(x_{r,0} + s_r) \ \text{ and } \ g_{r,k} = \nabla h_r(x_{r,k}) = \nabla f(x_{r,k}).$$

The model h_{i-1} thus results from a *modification* of f_{i-1} by a linear term that enforces the relation $g_{i-1,0} = \nabla h_{i-1}(x_{i-1,0}) = R_i g_{i,k}$. This first-order modification[(2)] ensures that the first-order behaviours of h_i and h_{i-1} are similar in a neighbourhood of $x_{i,k}$ and $x_{i-1,0}$, respectively. Indeed, if s_i and s_{i-1} satisfy $s_i = P_i s_{i-1}$, we then have that

$$\langle g_{i,k}, s_i\rangle = \langle g_{i,k}, P_i s_{i-1}\rangle = \frac{1}{\sigma_i}\langle R_i g_{i,k}, s_{i-1}\rangle = \frac{1}{\sigma_i}\langle g_{i-1,0}, s_{i-1}\rangle, \tag{2.6}$$

where we have also used (2.4).

We next need to represent, at level $i-1$, feasibility with respect to the bound constraints. Because we aim at a description which is coherent across levels and because we wish to avoid general linear constraints, we choose this representation as a bound-constrained domain $\mathcal{F}_{i-1}$ defined recursively such that, for $i = 1, \ldots, r$,

$$x_{i,k} + P_i s_{i-1} \in \mathcal{F}_i \text{ for all } x_{i-1,0} + s_{i-1} \in \mathcal{F}_{i-1}, \tag{2.7}$$

with $\mathcal{F}_r = \mathcal{F}$, thereby ensuring that all iterates at the finest level remain feasible for the original bounds. In our algorithm, the specific choice of $\mathcal{F}_{i-1}$ is done using a Gelman-Mandel-like formula (see Gelman and Mandel, 1990, or Gratton et al., 2008), stating that

$$\mathcal{F}_{i-1} = \{x_{i-1} \in \mathbb{R}^{n_{i-1}} \ | \ l_{i-1} \le x_{i-1} \le u_{i-1}\}, \tag{2.8}$$

[(2)] The first-order modification (2.5) is usual in multigrid applications in the context of the "full approximation scheme", where it is usually called the "tau correction" (see, for instance, Chapter 3 of Briggs et al., 2000, or Hemker and Johnson, 1987).

where the bound vectors l_{i-1} and u_{i-1} are recursively defined componentwise by

$$[l_{i-1}]_j = [x_{i-1,0}]_j + \frac{1}{\|P_i\|_\infty} \max_{t=1,\ldots,n_i} [l_i - x_{i,k}]_t \tag{2.9}$$

and

$$[u_{i-1}]_j = [x_{i-1,0}]_j + \frac{1}{\|P_i\|_\infty} \min_{t=1,\ldots,n_i} [u_i - x_{i,k}]_t, \tag{2.10}$$

for $j = 1, \ldots, n_{i-1}$, with $l_r = l$ and $u_r = u$. We refer the reader to Lemma 4.3 in Gratton et al. (2008) for a proof of (2.7).

We then need to represent at the coarser level $i-1$ the constraints on the stepsize resulting from the trust region at level i,

$$\mathcal{B}_{i,k} = \{x_{i,k} + s_i \in \mathbb{R}^{n_i} \mid \|s_i\|_\infty \leq \Delta_{i,k}\},$$

associated with $x_{i,k}$, and also from the trust regions at levels higher than i. Let us denote by $\mathcal{A}_i$ the box representing these stepsize constraints inherited from higher levels. Then $\mathcal{B}_{i,k} \cap \mathcal{A}_i$ is also a box of the form $\{x_i \mid v_i \leq x_i \leq w_i\}$, where $v_i, w_i \in \mathbb{R}^{n_i}$. The set $\mathcal{A}_{i-1}$ is then defined by

$$\mathcal{A}_{i-1} = \{x_{i-1} \in \mathbb{R}^{n_{i-1}} \mid R_i v_i \leq x_{i-1} \leq R_i w_i\}. \tag{2.11}$$

For consistency, we set $\mathcal{A}_r = \mathbb{R}^n$. This definition is less restrictive than the Gelman-Mandel procedure used to define $\mathcal{F}_{i-1}$, but does not imply that $x_{i,k} + P_i s_{i-1} \in \mathcal{B}_{i,k} \cap \mathcal{A}_i$ for all $x_{i-1,0} + s_{i-1} \in \mathcal{A}_{i-1}$. This remains acceptable because the trust-region bounds need only be satisfied up to a constant factor to ensure global convergence (again see Gratton et al., 2008).

At level $i-1$, we finally consider the intersection of the domain corresponding to the original bounds on the problem and that resulting from the trust-region restrictions at higher level (if any). This intersection is given by $\mathcal{L}_i \stackrel{\text{def}}{=} \mathcal{F}_i \cap \mathcal{A}_i$ for $i = 0, \ldots, r$. The local subproblem to be solved at level $i-1$ is then given by

$$\min_{x_{i-1,0} + s_{i-1} \in \mathcal{L}_{i-1}} h_{i-1}(x_{i-1,0} + s_{i-1}).$$

As indicated above, we solve this subproblem using a trust-region method starting from $x_{i-1,0}$, whose ℓ-th iteration then involves the computation of

$$\min_{\substack{\|s_{i-1}\|_\infty \leq \Delta_{i-1,\ell} \\ x_{i-1,\ell} + s_{i-1} \in \mathcal{L}_{i-1}}} h_{i-1}(x_{i-1,\ell} + s_{i-1}). \tag{2.12}$$

In addition to the features already discussed, our recursive multilevel trust-region algorithm (see Algorithm RMTR$_\infty$ on page 6) crucially considers whether recurring to the local lower-level model is useful. This decision is made by comparing *criticality measures* at the current and lower levels. At level i, the criticality measure is computed as

$$\chi_{i,k} \stackrel{\text{def}}{=} \chi(x_{i,k}) = |\min_{\substack{x_{i,k} + d \in \mathcal{L}_i \\ \|d\|_\infty \leq 1}} \langle g_{i,k}, d\rangle|, \tag{2.13}$$

which can be interpreted as the maximal decrease of the linearized problem that can be achieved in the intersection of $\mathcal{L}_i$ and a box of radius one (see Conn, Gould, Sartenaer and Toint, 1993, for instance). We declare that recurring to the lower level is useful whenever this decrease at level $i-1$ is significant compared to that achievable at level i, which we formalize by the condition that

$$\frac{\chi_{i-1,0}}{\sigma_i} \geq \kappa_\chi \chi_{i,k}, \tag{2.14}$$

where $\kappa_\chi \in (0,1)$. The factor σ_i in the left-hand side results from (2.6) and the fact that the criticality measure (2.13) is a linear approximation of the decrease that can be achieved from $x_{i,k}$. If (2.14) does not hold, then the algorithm resorts to using the quadratic model (2.2) at level i, which we denote by $m_{i,k}(x_{i,k}+s_i)$. Otherwise, the choice between the two models remains open, allowing, as we discuss below, the efficient exploitation of multigrid techniques such as *smoothing iterations*.

We now turn to the more formal description of our algorithm and assume that the prolongations P_i and the restrictions R_i are known, as well as the functions $\{f_i\}_{i=0}^{r-1}$. We use the constants κ_χ, η_1, η_2, γ_1 and γ_2 satisfying the conditions $\kappa_\chi \in (0,1)$, $0 < \eta_1 \leq \eta_2 < 1$, and $0 < \gamma_1 \leq \gamma_2 < 1$. An initial trust-region radius for each level, $\Delta_{i,0} > 0$, is also defined. The algorithm's initial data consists of the level index i ($0 \leq i \leq r$), a starting point $x_{i,0}$, the gradient $g_{i,0}$ at this point and the corresponding criticality measure $\chi_{i,0}$, the description of the feasible sets $\mathcal{F}_i$ and $\mathcal{A}_i$, and a criticality tolerance $\epsilon_i^\chi \in (0,1)$.

Further motivation for this algorithm can be found in Gratton et al. (2008), together with a proof that, under reasonable assumptions, every limit point of the sequence of produced iterates must be a first-order critical point in the sense that $\lim_{k\to\infty} \chi_{r,k} = 0$. In particular, the functions f_i must have uniformly bounded Hessians for $i = 0, \ldots, r$. We produce a few additional useful comments :

1. The minimization of $f(x) = f_r(x_r) = h_r(x_r)$ (up to the critical tolerance $\epsilon_r^\chi < \chi_{r,0}$) is achieved by calling RMTR$_\infty$(r, $x_{r,0}$, $g_{r,0}$, $\chi_{r,0}$, $\mathcal{F}$, $\mathbb{R}^n$, ϵ_r^χ), for some starting point $x_{r,0}$. For coherence of notations, we thus view this call as being made from some (virtual) iteration 0 at level $r+1$.

2. The test for the value of i at the beginning of Step 1 is designed to identify the lowest level, at which no further recursion is possible. In this case, a Taylor's (i.e., non-recursive) iteration is the only possibility.

3. The set $\mathcal{W}_{i,k}$ represents the feasible domain of subproblem (2.12).

4. The formula for $\delta_{i,k}$ in Step 2 results from (2.5) and (2.6).

5. The "sufficient decrease" in the model (2.15) imposed in Step 3 means, as usual for trust-region methods, that the step $s_{i,k}$ must satisfy a specific condition (see Chapter 12 of Conn et al., 2000), known as the *Cauchy point condition*, and which imposes sufficient decrease relative to the local first-order behaviour of the objective function. It requires that

$$m_{i,k}(x_{i,k}) - m_{i,k}(x_{i,k}+s_{i,k}) \geq \kappa_{\rm red}\chi_{i,k} \min\left[\frac{\chi_{i,k}}{1+\|H_{i,k}\|_\infty}, \Delta_{i,k}, 1\right] \qquad (2.18)$$

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

6. Iteration k at level i is said to be *successful* if $\rho_{i,k} \geq \eta_1$.

3 A practical algorithm

Our algorithm description so far leaves a number of practical choices unspecified. It is the purpose of this section to provide the missing details for the particular implementations whose numerical performance is reported in this paper. These details are of course influenced by our focus on discretized problems, where the different levels correspond to different discretization grids, from coarser to finer.

Algorithm 2.1: RMTR$_\infty$ $(i, x_{i,0}, g_{i,0}, \chi_{i,0}, \mathcal{F}_i, \mathcal{A}_i, \epsilon_i^\chi)$

Step 0: Initialization. Compute $f_i(x_{i,0})$. Set $k = 0$ and

$$\mathcal{L}_i = \mathcal{F}_i \cap \mathcal{A}_i \quad \text{and} \quad \mathcal{W}_{i,0} = \mathcal{L}_i \cap \mathcal{B}_{i,0},$$

where $\mathcal{B}_{i,0} = \{x_{i,0} + s_i \in \mathbb{R}^{n_i} \mid \|s_i\|_\infty \leq \Delta_{i,0}\}$.

Step 1: Model choice. If $i = 0$, go to Step 3. Else, compute $R_i x_{i,k}$, $R_i g_{i,k}$, $\mathcal{F}_{i-1}$ from (2.8)-(2.10), $\mathcal{A}_{i-1}$ from (2.11) and $\chi_{i-1,0}$. If (2.14) fails, go to Step 3. Otherwise, choose to go to Step 2 or to Step 3.

Step 2: Recursive step computation. Call Algorithm

$$\mathbf{RMTR}_\infty(i-1, R_i x_{i,k}, R_i g_{i,k}, \chi_{i-1,0}, \mathcal{F}_{i-1}, \mathcal{A}_{i-1}, \epsilon_{i-1}^\chi),$$

yielding an approximate solution $x_{i-1,*}$ of (2.12). Then define $s_{i,k} = P_i(x_{i-1,*} - R_i x_{i,k})$, set $\delta_{i,k} = \frac{1}{\sigma_i}\big[h_{i-1}(R_i x_{i,k}) - h_{i-1}(x_{i-1,*})\big]$ and go to Step 4.

Step 3: Taylor step computation. Choose $H_{i,k}$ and compute a step $s_{i,k} \in \mathbb{R}^{n_i}$ that sufficiently reduces the model

$$m_{i,k}(x_{i,k} + s_i) = h_i(x_{i,k}) + \langle g_{i,k}, s_i \rangle + \tfrac{1}{2}\langle s_i, H_{i,k} s_i \rangle \tag{2.15}$$

and such that $x_{i,k} + s_{i,k} \in \mathcal{W}_{i,k}$. Set $\delta_{i,k} = m_{i,k}(x_{i,k}) - m_{i,k}(x_{i,k} + s_{i,k})$.

Step 4: Acceptance of the trial point. Compute $h_i(x_{i,k} + s_{i,k})$ and

$$\rho_{i,k} = \big[h_i(x_{i,k}) - h_i(x_{i,k} + s_{i,k})\big]/\delta_{i,k}. \tag{2.16}$$

If $\rho_{i,k} \geq \eta_1$, then define $x_{i,k+1} = x_{i,k} + s_{i,k}$; otherwise, define $x_{i,k+1} = x_{i,k}$.

Step 5: Termination. Compute $g_{i,k+1}$ and $\chi_{i,k+1}$. If $\chi_{i,k+1} \leq \epsilon_i^\chi$ or $x_{i,k+1} \notin \mathcal{A}_i$, then return with the approximate solution $x_{i,*} = x_{i,k+1}$.

Step 6: Trust-Region Update. Set

$$\Delta_{i,k+1} \in \begin{cases} [\Delta_{i,k}, +\infty) & \text{if } \rho_{i,k} \geq \eta_2, \\ [\gamma_2 \Delta_{i,k}, \Delta_{i,k}] & \text{if } \rho_{i,k} \in [\eta_1, \eta_2), \\ [\gamma_1 \Delta_{i,k}, \gamma_2 \Delta_{i,k}] & \text{if } \rho_{i,k} < \eta_1, \end{cases} \tag{2.17}$$

and $\mathcal{W}_{i,k+1} = \mathcal{L}_i \cap \mathcal{B}_{i,k+1}$ where

$$\mathcal{B}_{i,k+1} = \{x_{i,k+1} + s_i \in \mathbb{R}^{n_i} \mid \|s_i\|_\infty \leq \Delta_{i,k+1}\}.$$

Increment k by one and go to Step 1.

3.1 Taylor iterations: smoothing and solving

The most important issue is how to enforce sufficient decrease at Taylor iterations, that is, when Step 3 is executed. At the coarsest level ($i = 0$), the cost of fully minimizing (2.15) inside the trust region remains small, since the subproblem is of low dimension. We thus solve the subproblem using the PTCG (Projected Truncated Conjugate-Gradient) algorithm designed for the standard trust-region algorithm (see Conn, Gould and Toint, 1988, or Conn, Gould and Toint, 1992).

At finer levels ($i > 0$), we use an adaptation of multigrid smoothing techniques. The main characteristics of multigrid algorithms (see Briggs et al., 2000) are based on the observation that different *frequencies* are present in the initial error on the solution of the finest grid problem (or even of the infinite-dimensional one), and become only progressively visible in the hierarchy from coarse to fine grids. Low frequencies are visible from coarse grids and up, but higher ones can only be distinguished when the mesh-size of the grid becomes comparable to the frequency in question. In multigrid strategies, some algorithms, called *smoothers*, are known to very efficiently reduce the high frequency components of the error on a grid (that is, in most cases, the components whose "wavelength" is comparable to the grid's mesh-size). But these algorithms have little effect on the low frequency error components. It is observed however that such components on a fine grid *appear* more oscillatory on a coarser grid. They may thus be viewed as high frequency components on some coarser grid and be in turn reduced by a smoother. Moreover, this is done at a lower cost since computations on coarser grids are typically much cheaper than on finer ones. The multigrid strategy consists therefore in alternating between solving the problem on coarse grids, essentially annihilating low frequency components of the error, and on fine grids, where high frequency components are reduced (at a higher cost). This last operation is often called *smoothing* because the effect of reducing high frequency components without altering much the low frequency ones has a *smoothing effect* of the error's behaviour. We next adapt, in what follows, the multigrid smoothing technique to the computation of a Taylor step satisfying the requirements of Step 3 of Algorithm RMTR$_\infty$.

A very well-known multigrid smoothing technique is the Gauss-Seidel method, in which each equation of the Newton system is solved in succession[(3)]. To extend this procedure to our case, rather than successively solving equations, we perform successive one-dimensional bound-constrained minimizations of the model (2.15) along the coordinate axes, provided the curvature of this model along each axis is positive. More precisely, consider the minimization of (2.15) at level i along the j-th axis (starting each minimization from s such that $\nabla m_{i,k}(x_{i,k} + s) \stackrel{\rm def}{=} g$). Then, provided that the j-th diagonal entry of $H_{i,k}$ is positive, the j-th one-dimensional minimization then results in the updates

$$\alpha_j = \mathrm{Pr}_{\mathcal{W}_{i,k}}(-[g]_j/[H_{i,k}]_{jj}), \quad [s]_j \leftarrow [s]_j + \alpha_j \ \text{ and } \ g \leftarrow g + \alpha_j H_{i,k} e_{i,j}, \tag{3.19}$$

where $\mathrm{Pr}_{\mathcal{W}_{i,k}}(.)$ is the orthogonal projection on the intersection of all the constraints at level i, that is on $\mathcal{W}_{i,k} = \mathcal{F}_i \cap \mathcal{A}_i \cap \mathcal{B}_{i,k}$, where we denote by $[v]_j$ the j-th component of the vector v and by $[M]_{jj}$ the j-th diagonal entry of the matrix M, and where $e_{i,j}$ is the j-th vector of the canonical basis of $\mathbb{R}^{n_i}$. If, on the other hand, $[H_{i,k}]_{jj} \leq 0$, then a descent step is made along the j-th coordinate axis until the boundary of $\mathcal{W}_{i,k}$ is reached and the model gradient is updated accordingly. This process is the well-known Sequential Coordinate Minimization (SCM) (see, for instance, Ortega and Rheinboldt (1970), Section 14.6), which we adapted to handle bound constraints. In what follows, we refer to a set of n_i successive unidimensional minimizations as a *smoothing cycle*. A SCM *smoothing iteration* then consists of one or more of these cycles.

In order to enforce convergence to first-order points, we still have to ensure that a sufficient model decrease (2.18) has been obtained within the trust region after one or more complete smoothing cycles. To do so, we start the first smoothing cycle by selecting

[(3)] See Briggs et al., 2000, page 10, or Golub and Van Loan, 1989, page 510, or Ortega and Rheinboldt, 1970, page 214, amongst many others.

the axis corresponding to the index j_m such that

$$j_m = \operatorname*{argmin}_{j} \quad [g_{i,k}]_j [d_{i,k}]_j, \tag{3.20}$$

where

$$d_{i,k} = \operatorname*{argmin}_{\substack{x_{i,k}+d \in \mathcal{L}_i \\ \|d\|_\infty \leq 1}} \quad \langle g_{i,k}, d\rangle. \tag{3.21}$$

Indeed in this case the minimization of the model $m_{i,k}$ along $[d_{i,k}]_{j_m}$ within the trust region is guaranteed to yield a *Generalized Cauchy step* $\alpha_{j_m}[d_{i,k}]_{j_m}$ such that (2.18) holds (as is shown in Appendix A). Since the remaining minimizations in the first smoothing cycle (and the following ones, if any) only decrease the value of the model further, (2.18) thus also holds for the complete step $s_{i,k}$.

3.2 Linesearch

The implementation whose numerical performance is discussed in Section 4 uses a version that combines the traditional trust-region techniques with a linesearch, in the spirit of Toint (1983, 1987), Nocedal and Yuan (1998) and Gertz (1999) (see Conn et al., 2000, Section 10.3.2). More precisely, if $\rho_{i,k} < \eta_1$ in Step 4 of Algorithm RMTR$_\infty$ and the step is *gradient related* in the sense that

$$|\langle g_{i,k}, s_{i,k}\rangle| \geq \epsilon_{\rm gr} \|g_{i,k}\|_2 \, \|s_{i,k}\|_2$$

for some $\epsilon_{\rm gr} \in (0,1)$, the step corresponding to a new iteration and a smaller trust-region radius can be computed by backtracking along $s_{i,k}$, instead of recomputing a new one using SCM smoothing. On the other hand, if some iteration at the topmost level is successful and the minimizer of the quadratic model in the direction $s_{r,k}$ lies sufficiently far beyond the trust-region boundary, then a single doubling of the step is attempted to obtain further descent, a strategy reminiscent of the *internal doubling* procedure of Dennis and Schnabel (1983) (see Conn et al., 2000, Section 10.5.2), or the *magical step* technique of Conn, Vicente and Visweswariah (1999) and Conn et al. (2000), Section 10.4.1. The theoretical arguments developed in these references guarantee that global convergence of the modified algorithm to first-order critical points is not altered.

3.3 Second-order and Galerkin models

The gradient correction v_{i-1} in (2.5) ensures that h_i and h_{i-1} coincide at first order (up to the constant σ_i) in the range of the prolongation operator, since

$$\langle g_{i,k}, P_i s_{i-1}\rangle = \frac{1}{\sigma_i}\langle R_i g_{i,k}, s_{i-1}\rangle = \frac{1}{\sigma_i}\langle g_{i-1,0}, s_{i-1}\rangle.$$

We can also achieve coherence of the models at second order by choosing

$$h_{i-1}(x_{i-1,0}+s_{i-1}) = f_{i-1}(x_{i-1,0}+s_{i-1}) + \langle v_{i-1}, s_{i-1}\rangle + \tfrac{1}{2}\langle s_{i-1}, W_{i-1}s_{i-1}\rangle, \tag{3.22}$$

where $W_{i-1} = R_i\nabla^2 h_i(x_{i,k})P_i - \nabla^2 f_{i-1}(x_{i-1,0})$, since we then have that

$$\langle P_i s_{i-1}, \nabla^2 h_i(x_{i,k})P_i s_{i-1,}\rangle = \frac{1}{\sigma_i}\langle s_{i-1}, \nabla^2 h_{i-1}(x_{i-1,0})s_{i-1}\rangle.$$

The second-order model (3.22) is of course more costly, as the matrix W_{i-1} must be computed when starting the minimization at level $i-1$ and must also be used to update the gradient of h_{i-1} at each successful iteration at level $i-1$.

Another strategy consists to choose $f_{i-1}(x_{i-1,0}+s_{i-1}) = 0$ for all s_{i-1} in (3.22). This strategy amounts to consider the lower-level model as the "restricted" version of the

quadratic model at the upper level (this is known as the *Galerkin approximation*) and is interesting in that no evaluation of f_{i-1} is required. In the unconstrained case, when this model is strictly convex and the trust region is large enough, one minimization in Algorithm RMTR$_\infty$ (without premature termination) corresponds to applying a Galerkin multigrid linear solver on the associated Newton's equation. Note that this choice is allowed within the theory presented in Gratton et al. (2008), since the zero function is obviously twice-continuously differentiable, bounded below and has uniformly bounded Hessians.

3.4 Hessian of the models

Computing a model Hessian $H_{i,k}$ is often one of the heaviest task in Algorithm RMTR$_\infty$. Our choice in the experiments described in Section 4 is to use the exact second derivative matrix of the objective functions f_i. However, we have designed an automatic strategy that avoids recomputing the Hessian at each iteration when the gradient variations are still well predicted by the available $H_{i,k-1}$. More specifically, we choose to recompute the Hessian at the beginning of iteration (i,k) $(k>0)$ whenever the preceding iteration not successful enough (i.e. $\rho_{i,k-1} < \eta_H$) or when

$$\|g_{i,k} - g_{i,k-1} - H_{i,k-1}s_{i,k-1}\|_2 > \epsilon_H \|g_{i,k}\|_2,$$

where $\epsilon_H \in (0,1)$ is a small user-defined constant. Otherwise, we use $H_{i,k} = H_{i,k-1}$. Default value of $\epsilon_H = 0.15$ and $\eta_H = 0.5$ appear to give satisfactory results in most cases.

3.5 Prolongations and restrictions

We have chosen to define the prolongation and restriction operators P_i and R_i as follows. The prolongation is chosen as the *linear interpolation* operator, and the restriction is its transpose normalized to ensure that $\|R_i\|_\infty = 1$ and $\sigma_i = \|P_i\|_\infty^{-1}$ (see (2.4)). These operators are never assembled, but are rather applied locally for improved efficiency. Cubic interpolation could also be used in principle, but it produces denser Galerkin models and is very restrictive in the context of Gelman-Mandel restrictions. Moreover our experience is that the algorithm is computationally less efficient.

3.6 Free and fixed form recursions

An interesting feature of the RMTR$_\infty$ framework is that its convergence properties are preserved if the minimization at lower levels ($i = 0, \ldots, r-1$) is stopped after the *first successful iteration*. The flexibility of this allows to consider different recursion patterns, namely *fixed-form* and *free-form* ones. In a fixed form recursion pattern, a maximum number of successful iterations at each level is specified (like in V- and W-cycles in multigrid algorithms, see Briggs et al. (2000)). If no such premature termination is used but the minimization at each level is carried out until one of the classical termination conditions on the criticality measure and step size (see Step 5 of Algorithm RMTR$_\infty$) is satisfied, then the actual recursion pattern is uniquely determined by the progress of minimization at each level (hence yielding a free form recursion pattern).

In Section 4, we compare three recursion forms. In the first form, which we call V-form, the minimization at the lower levels consists in one successful smoothing iteration, followed by a successful recursive iteration, itself followed by a second successful smoothing iteration[(4)]. The second form is called W-form and is defined as a V-form to which is added one successful recursive iteration, and a final smoothing iteration. The third form is the free form recursion as explained above, in which we impose however that smoothing iterations and recursive (successful) iterations alternate at all levels but the coarsest. Indeed, during our experiments, we have found this alternance very fruitful (and rather

[(4)] A the coarsest level 0, smoothing iterations are skipped and recursion impossible.

natural in the interpretation of the algorithm as an alternance of high frequency reductions and low frequency removals).

Note that for each recursion form, any remaining iteration is skipped if one of the termination conditions in Step 5 of Algorithm RMTR$_\infty$ is satisfied.

3.7 Computing the starting point at the fine level

We also take advantage of the multilevel recursion idea to compute the starting point $x_{r,0}$ at the finest level by first restricting the user-supplied starting point to the lowest level and then applying Algorithm RMTR$_\infty$ successively at levels 0 up to $r-1$. In our experiments based on regular meshes (see Section 4), the accuracy on the criticality measure that is required for termination at level $i < r$ is given by

$$\epsilon_i^\chi = \epsilon_{i+1}^\chi \sigma_{i+1}, \tag{3.23}$$

where ϵ_r^χ is the user-supplied criticality requirement for the topmost level and σ_{i+1} is due to (2.6) and the definition (2.13) of the criticality measure. Once computed, the solution at level i is then prolongated to level $i+1$ using *cubic interpolation*. The criteria (3.23) comes from the fact that we want that the prolongation of our step stay critical for the upper level $i+1$ excepted for the highest frequencies of the error that are not visible at level i and only appear at level $i+1$ and finer levels.

3.8 Constants choice and recursive termination thresholds

We conclude the description of our practical algorithm by specifying our choice for the constants and the level-dependent criticality thresholds ϵ_i^χ. We set

$$\eta_1 = 0.01, \quad \eta_2 = 0.95, \quad \gamma_1 = 0.05 \text{ and } \gamma_2 = 1.00, \tag{3.24}$$

as this choice appears most often appropriate. The value 1 is also often satisfactory for the $\Delta_{i,0}$. We considered two possible expressions for the criticality thresholds. The first is related to the descent condition (2.14) and is given by

$$\epsilon_i^\chi = \kappa_\chi \chi_{i,k} \sigma_{i+1}. \tag{3.25}$$

We also considered using (3.23), but this was found to be unsuitable for recursive iterations. Indeed, it often prevented the effective use of coarse level computations because it was satisfied at $x_{0,i}$, resulting in an immediate return to the fine level. We thus considered an adaptation of this rule given by

$$\epsilon_i^\chi = \min\{\epsilon_{i+1}^\chi, \kappa_\chi \chi_{i,k}\} \sigma_{i+1}. \tag{3.26}$$

This adaptation was further motivated by the observation that the alternance between SCM smoothing and recursive iterations is very efficient in practice and we want thus to impose that at least one lower-level iteration is done if the descent condition (2.14) allows it.

4 Numerical tests

The algorithm described above has been coded in FORTRAN 95 and all experiments below were run on a 3.0 Ghz single-processor PC with 2 Gbytes of RAM.

4.1 Test problems

We have considered a suite of minimization problems in infinite-dimensional spaces, involving differential operators. These problems are detailed in Appendix B. The differential

operators are discretized on a hierarchy of regular grids such that the coarse grid at level $i-1$ is defined by taking every-other point in the grid at level i: the ratio between the grid spacing of two consecutive levels in each coordinate direction is therefore 2. The grid transfer operators P_i are defined as in classical geometric multigrid settings, using interpolation operators. The restriction operators R_i are such that (2.4) holds.

All experiments discussed below consider the solution of the test problem on the finest grid, whose size may be found in Table 4.1, together with other problems characteristics. The algorithms were terminated when the criticality measure (2.13) at the finest level was below 10^{-3} for all the test cases. Notice that requiring that $\chi_{r,k} \leq \epsilon_r = 10^{-3}$ is approximately the same as requiring the *scaled criticality measure* $\frac{\chi_{r,k}}{n_r}$, whose value is comparable, for example, with the infinity norm of the projected gradient $||Pr_{\mathcal{F}}(x_{r,k} - g_{r,k}) - x_{r,k}||_\infty$, to be such that $\frac{\chi_{r,k}}{n_r} \leq \frac{\epsilon_r}{n_r}$. This last tolerance is, for instance, $\frac{\epsilon_r}{n_r} \approx 10^{-9}$ in the case where $n_r = 1046529$ and $\frac{\epsilon_r}{n_r} \approx 10^{-8}$ if $n_r = 65025$.

Problem name	n_r	r	Comment
DNT	511	8	1-D, quadratic
P2D	1046529	9	2-D, quadratic
P3D	250047	5	3-D, quadratic
DEPT	1046529	9	2-D, quadratic, (Minpack 2)
DPJB	1046529	9	2-D, quadratic, with bound constraints, (Minpack 2)
DODC	65025	7	2-D, convex, (Minpack 2)
MINS-SB	1046529	9	2-D, convex, smooth boundary conds.
MINS-OB	65025	7	2-D, convex, oscillatory boundary conds.
MINS-DMSA	65025	7	2-D, convex, (Minpack 2)
IGNISC	65025	7	2-D, convex
DSSC	1046529	9	2-D, convex, (Minpack 2)
BRATU	1046529	9	2-D, convex, (Minpack 2)
MINS-BC	65025	7	2-D, convex, with bound constraints
MEMBR	393984	9	2-D, convex, free boundary, with bound constraints
NCCS	130050	7	2-D, nonconvex, smooth boundary conds.
NCCO	130050	7	2-D, nonconvex, oscillatory boundary conds.
MOREBV	1046529	9	2-D, nonconvex

Table 4.1: Test problem characteristics

Our testing strategy, which is discussed in the next paragraphs, is first to establish a good default value for the algorithmic parameters, and, in a second step, to compare the resulting method with other competing approaches.

4.2 In search of efficient default parameters

Given the relatively large number of parameters in our method, a complete discussion of all possible combinations is outside the scope of this paper. We have therefore adopted the following approach. We first fixed the parameters for which a reasonable consensus already exists, namely the trust-region parameters η_1, η_2, γ_1 and γ_2, which are set as in (3.24), in accordance with Conn et al. (2000) and Gould, Orban, Sartenaer and Toint (2005). The initial trust-region radii $\Delta_{i,0}$ are set to 1, as suggested in Section 17.2 of the first of these references. A second class of parameters was then isolated, containing algorithmic options with very marginal effect on the computational results. These are the choice of activating the linesearch mechanism (we allow for backtracking if the initial step is unsuccessful and at most one extrapolation evaluation if it is successful and gradient-related with $\epsilon_{\rm gr} = 0.01$), the parameters ϵ_H and η_H of the Hessian evaluation strategy (we chose $\eta_H = 0.5$ and $\epsilon_H = 0.15$), and the degree of the interpolation in the prolongation operator (linear interpolation is used within recursive iterations, and cubic interpolation

when prolongating the solution at a coarse level into a starting point at the next finer one). The remaining algorithmic parameters were either central in the definition of our method or found to alter the performance of the method significantly, and we focus the rest of our discussion on their choice.

We begin by determining the optimal combination of these paramaters. For this purpose, we ran a large number (192) of possible combinations of these options on our set of 17 test problems and report all results of the 3264 runs on a comet-shape graph representing a measure of the effort spent in function evaluations as a function of CPU-time. More precisely, we have first scaled, separately for each test problem, the number of function evaluations and CPU-time by dividing them by the best obtained for this problem by all algorithmic variants. We then plotted the averages of these scaled measures on all test problems for each algorithmic variant separately, after removing the variants for which the CPU limit of 1000 seconds was reached on at least one problem. In the first of these plots (Figures 4.1 and 4.2), we have used triangles for variants where the coarse Galerkin model is chosen at recursive iterations and stars for variants where the second-order model (3.22) is chosen instead[(5)].

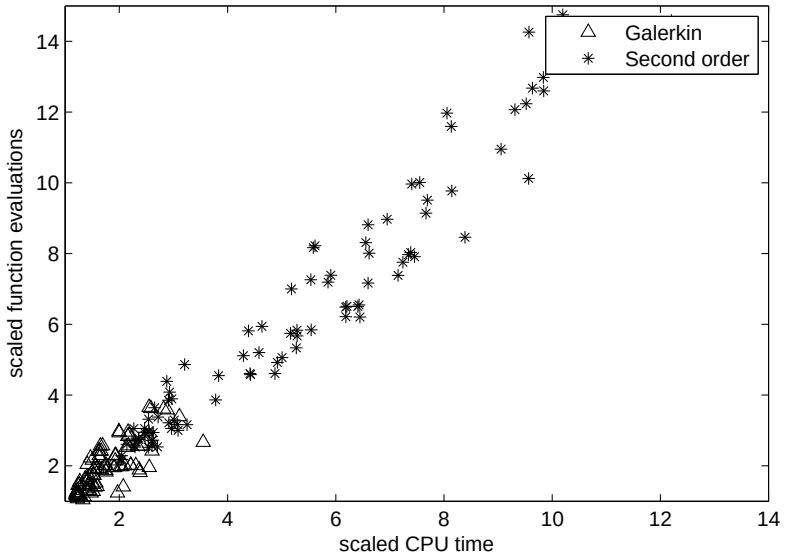

Figure 4.1: Average scaled function evaluations versus average scaled CPU-time for all algorithmic variants, distinguishing the type of model used.

We note a substantial spread of the results, with some options being up to fifteen times worse than others. The worst cases (in the top right corner) correspond to combinations of the quadratic model (3.22) with a single smoothing cycle and small values of κ_χ. On the other hand, the choice of the Galerkin model is very clearly the best. This is mainly due to the numerical cost of the alternative because it requires a function/Hessian evaluation and a matrix update for each model in (3.22). Even on the testcases for which this choice proves superior in number of iterations, the advantage is then lost in CPU-time. In view of this conclusion, we therefore select the the Galerkin model as our default and restrict further analysis to this case.

We now consider the number of smoothing cycles performed at each Taylor iteration (at a level $i > 0$) and illustrate our results in Figure 4.3. All algorithmic variants (with

[(5)] Notice that we did not represent the tests where the coarse model is defined as in (2.5) because preliminary tests showed that performing only a first-order correction is indisputably not competitive.

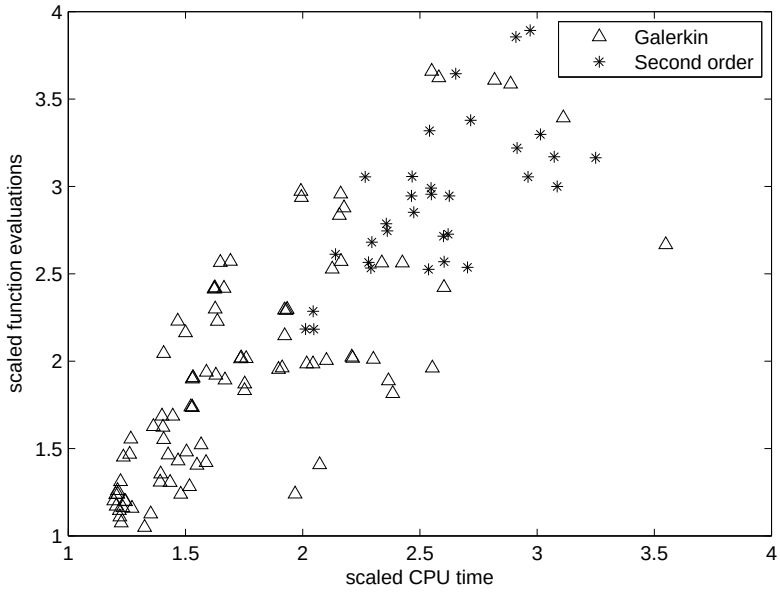

Figure 4.2: Detail of the lower left-hand corner of Figure 4.1.

the coarse Galerkin model) are again represented in a picture similar to Figure 4.1, where different symbols are used to isolate variants using different number of smoothing cycles.

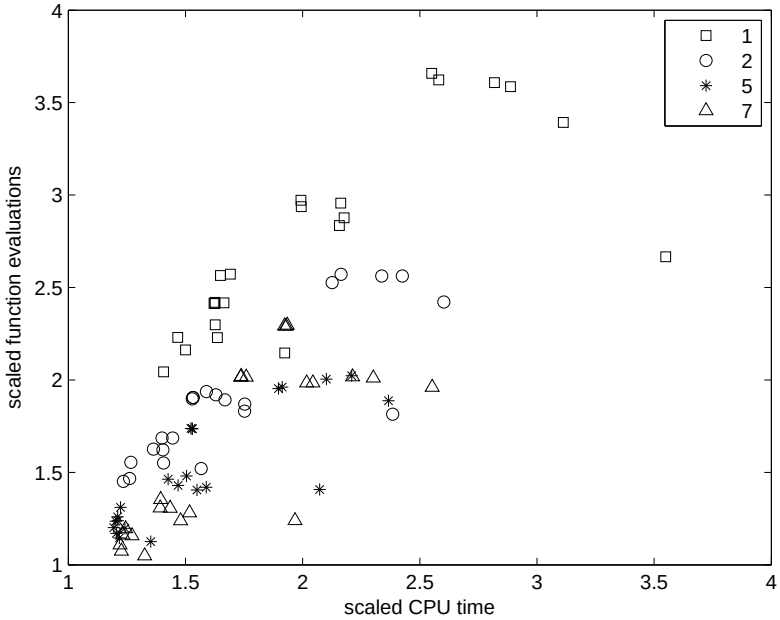

Figure 4.3: Average scaled function evaluations versus average scaled CPU-time for all algorithmic variants, distinguishing the number of smoothing cycles per Taylor iteration.

An important property of this option is that the number of function evaluations decreases as the number of cycles increases, because a single evaluation is exploited to a fuller

extent if more cycles are performed consecutively. This correlation is maintained up to a level (probably depending on the quadraticity of the objective function) beyond which the work of additional cycles is no longer effective. The correlation is much less clear when considering CPU-time, even if our result indicate that too few smoothing cycles is seldom the best option. Good choices seem to range between 2 and 7 cycles.

Choosing between the values for κ_χ is not easy. We have considered four possible values (1/2, 1/4, 1/8, 1/16). We first note that choosing κ_χ to be significantly larger that 1/2 results in a poor exploitation of the multilevel nature of the problem, since recursive iterations become much less frequent. On the other hand, values much smaller than 1/16 are also problematic because recursive iterations are then initiated for a too marginal benefit in optimality, although this strategy is closer to the unconditional recursive nature of multigrid algorithms for linear systems. In our tests the best threshold has been obtained for $\kappa_\chi = 1/4$ (see Figure 4.4, which is built on the same principle as the previous ones).

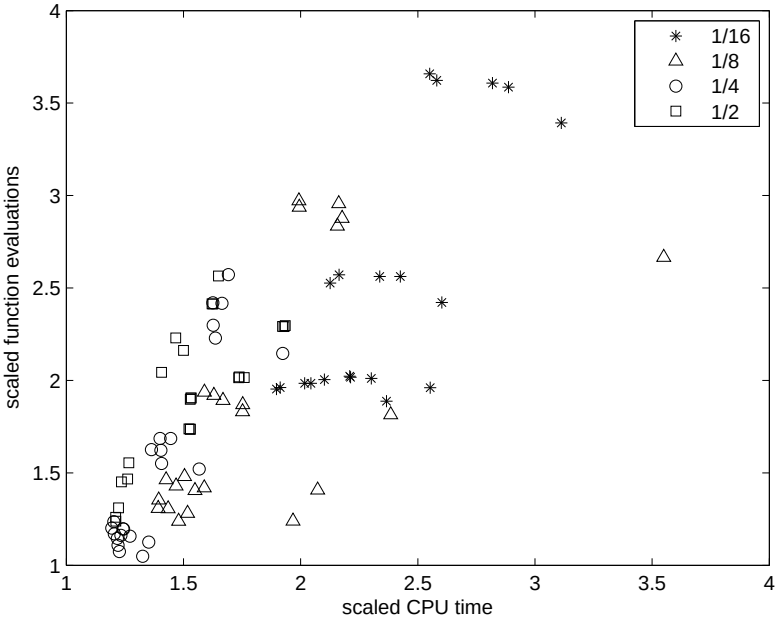

Figure 4.4: Average scaled function evaluations versus average scaled CPU-time for all algorithmic variants, distinguishing the values of κ_χ.

We now turn to the impact of the cycle types on performance, which is illustrated in Figure 4.5.

Remarkably, an excellent performance can be obtained with the three considered cycle styles, quite independently of the other algorithmic parameters. In particular, this indicates that the strategy for automatically adapting the cycle type to the problem at run-time is reasonably efficient. It is however slightly more complicated and the simpler V-form may often be prefered in practice.

Finally, Figure 4.6 shows the effect of the coarse criticality threshold choice between (3.25) (nomin) and (3.26) (min). It indicates that (3.26) is generally preferable, although the performance remains mixed.

As a conclusion of this analysis, we decided to select the defaults as the use of the Galerkin model, 7 smoothing cycles per Taylor iteration, a value of $\kappa_\chi = 1/4$, V-form iterations and the (3.26) termination rule.

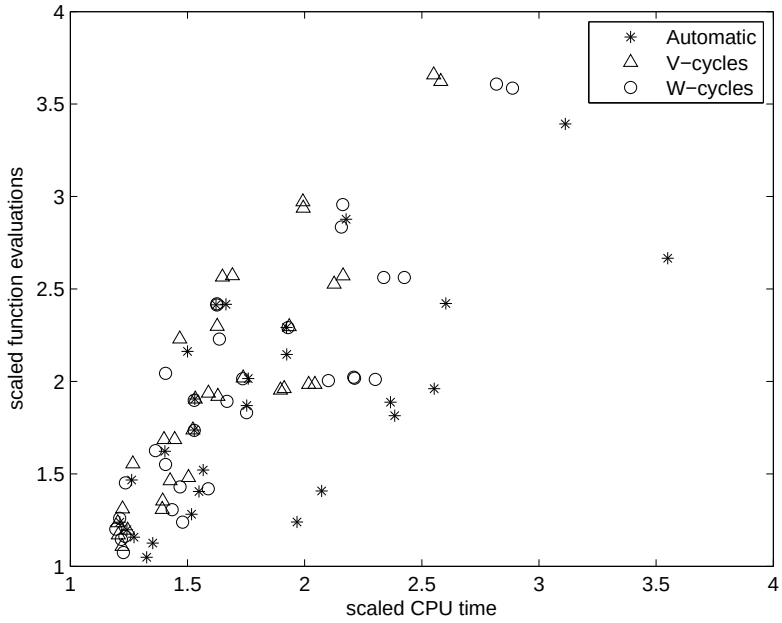

Figure 4.5: Average scaled function evaluations versus average scaled CPU-time for all algorithmic variants, distinguishing the type of recursive cycles.

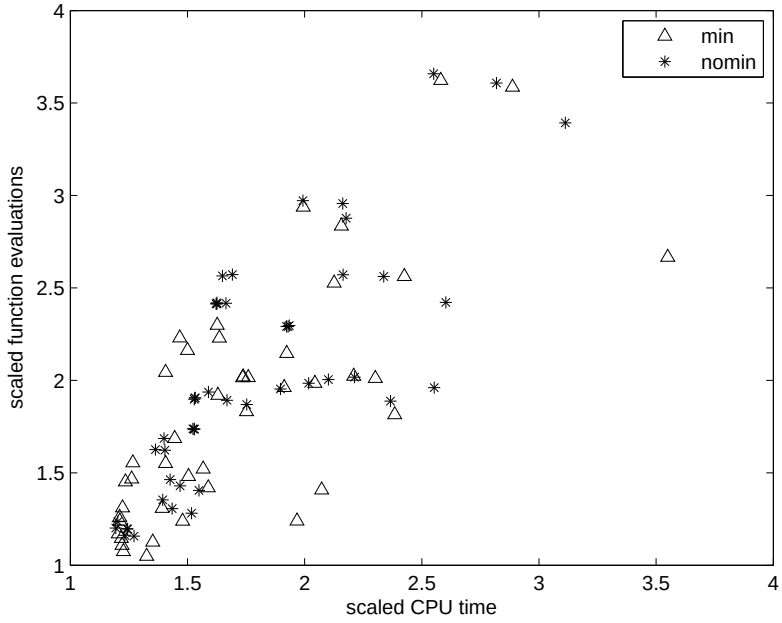

Figure 4.6: Average scaled function evaluations versus average scaled CPU-time for all algorithmic variants, distinguishing the type of lower level criticality threshold.

4.3 Performance of RMTR$_\infty$

We now analyse the performance of the resulting recursive trust-region algorithm in comparison with other approaches on our battery of 17 test problems. This analysis is conducted by comparing four algorithms:

- the *all on finest* (**AF**) algorithm, which is a standard Newton trust-region algorithm (with PTCG as subproblem solver) applied at the finest level, without recourse to coarse-level computations;
- the *mesh refinement technique* (**MR**), where the discretized problems are solved from the coarsest level (level 0) to the finest one (level r) successively, using the same standard Newton trust-region method, and where the starting point at level $i+1$ is obtained by prolongating (using P_{i+1}) the solution obtained at level i;
- the *multilevel on finest* (**MF**) method, where Algorithm RMTR$_\infty$ is applied directly on the finest level;
- the *full multilevel* (**FM**) algorithm where Algorithm RMTR$_\infty$ is applied successively on progressively finer discretizations (from coarsest to finest) and where the starting point at level $i+1$ is obtained by prolongating (using P_{i+1}) the solution obtained at level i.

A CPU-time performance profile (see Dolan and Moré, 2002) is presented in Figure 4.7 for all our test problems and these four variants. The first conclusion is that the full multilevel variant (FM) clearly outperforms all other variants. The second observation is that the AF variant is, as expected, by far the worst. The remaining two variants are surprisingly close, and the use of recursive iterations on the fine level appears to have an efficiency similar to that of optimizing on successively finer grids. These observations are confirmed by a detailed analysis of the complete numerical results presented in Appendix C.

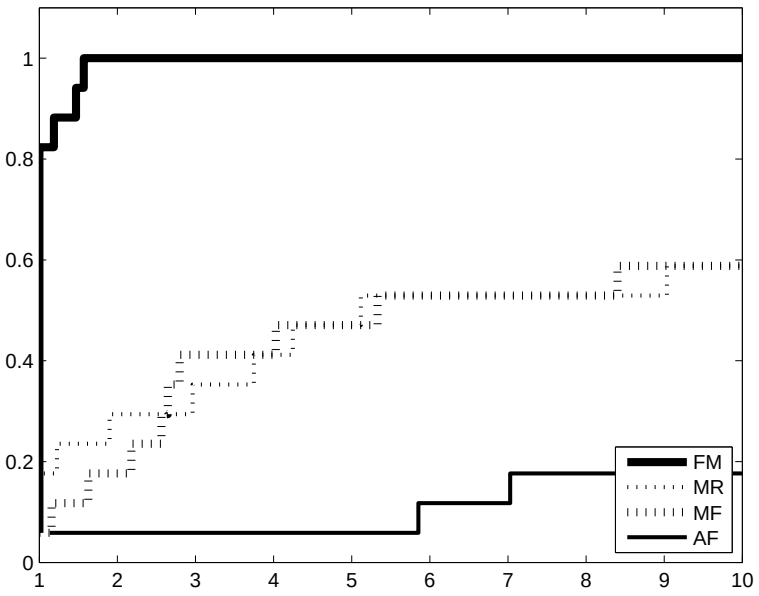

Figure 4.7: Performance profile for CPU time with variants AF, MF, MR and FM (17 test problems).

4.3.1 Unconstrained problems

We now turn to a more detailed comparison of the MR and FM variants on three specific unconstrained test problems (P2D, MINS-SB and NCCS), which we consider representative of the various problem classes mentioned in Table 4.1.

The performance of the algorithms is illustrated for each of these problems by a figure showing the history of the scaled criticality measure defined in Section 4.1 when the MR (thin line) and the FM (bold line) algorithms are used. In these figures, the dashed line represents the increase of the scaled criticality measure when a solution is prolongated during the application of a mesh refinement process. Moreover, and because iterations at coarse levels are considerably cheaper than those at higher ones, we have chosen to represent these histories as a function of the *equivalent number of finest iterations*, given by

$$q = \sum_{i=0}^{r} q_i \left(\frac{n_i}{n_r} \right), \tag{4.27}$$

where q_i is the number of iterations at level i.

We first consider the quadratic minimization problem P2D in Figure 4.8. Because this problem is equivalent to solving a linear system of equations, we expect algorithm FM to exhibit a multigrid-type behaviour. Looking at Figure 4.8, we see that this is effectively the case. We note that FM is considerably more efficient than MR (by a factor approaching 100). This last result confirms that our trust-region globalization is not hindering the known efficiency of the multigrid methods for this type of problems. Note that the significant increase of the scaled criticality measure when a lower level solution is prolongated to an upper level starting point is due to the fact that oscillatory components of the error cannot be represented on the coarser levels and therefore could not have been reduced at these levels.

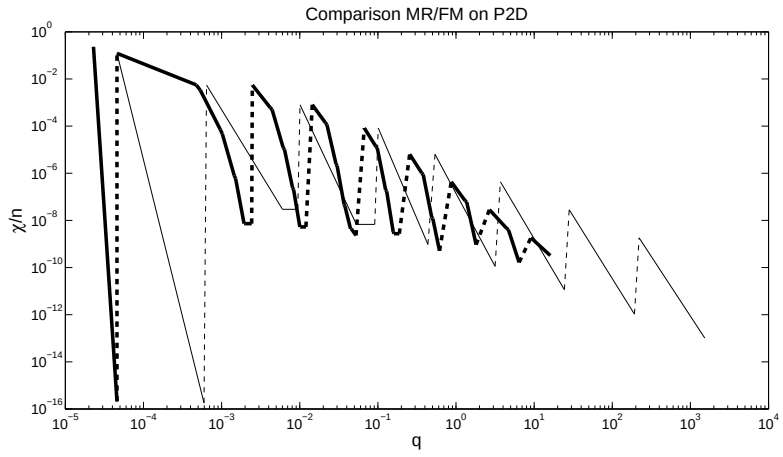

Figure 4.8: History of the scaled criticality measure on P2D. A small circle surrounds the iterations where the trust region is active. *Note that both axis are logarithmic.*

The same conclusions seem to apply when we consider Figures 4.9[(6)] and 4.10, where the same algorithms are tested on MINS-SB and NCCS, respectivelyx This is remarkable because the problems are now more general and do not correspond anymore to linear systems of equations (MINS-SB is nonquadratic) or elliptic problems (NCCS is non-convex).

An important feature of the classical trust-region algorithm is that its convergence is speeded up when the trust-region becomes inactive (because the algorithm then reduces to Newton's method and thus achieves quadratic convergence under the assumption that the

[(6)] Observe that the MR variant had to be stopped after 1 hour of computing on this problem.

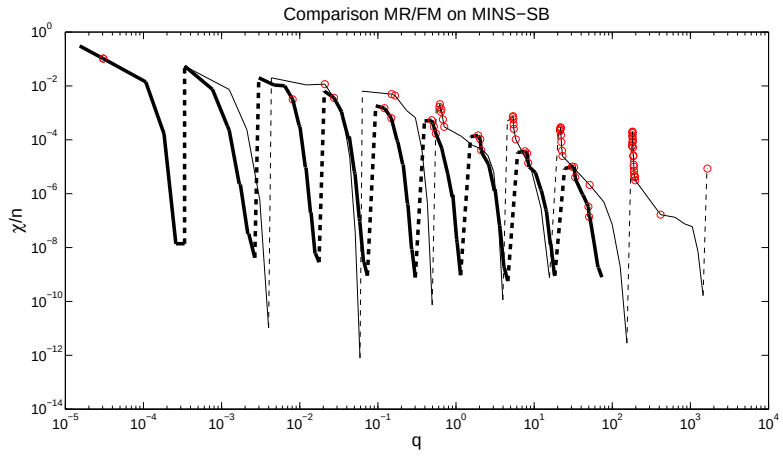

Figure 4.9: History of the scaled criticality measure on MINS-SB. A small circle surrounds the iterations where the trust region is active. *As above, both axis are logarithmic.*

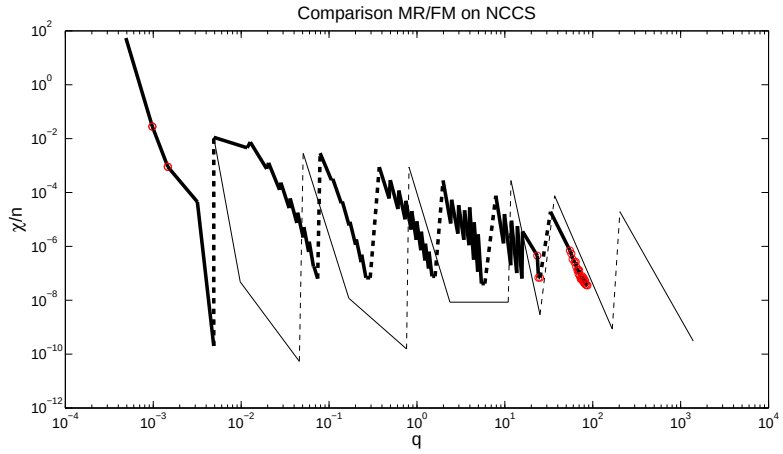

Figure 4.10: History of the scaled criticality measure on NCCS. A small circle surrounds the iterations trust region is active.*As above, both axis are logarithmic.*

second-order Taylor model (2.2) is chosen). Iterations where the trust-region is active have been indicated, in the above figures, by a small circle (observe that they often correspond to non-monotonic decrease of the scaled criticality). We note that no such iteration occurs for MR and FM on P2D, and also that convergence speeds up for all methods as soon as the trust region becomes inactive, even if the rate is at most linear for the multilevel methods.

4.3.2 Bound-constrained problems

We finally evaluate the RMTR$_\infty$ algorithm on the bound-constrained problems DPJB, MINS-BC and MEMBR. The results for these problems are presented in Figures 4.11 to 4.12.

We first note that the relative performance of the considered algorithms is very similar to that already analyzed for unconstrained problems, at least for DPJB[(7)] and MEMBR.

[(7)] We should note here that the Hessian of this quadratic problem is not supplied by the MINPACK

On this last problem, the figure indicates that further efficiency gains could be obtained by a finer tuning of the termination accuracy at levels 5, 6 and 7. On all three problems, a gain in CPU time of a factor exceeding 10 is typically obtained when considering the multilevel variant. Again, the trust-region constraint is mostly inactive on these examples. This is in sharp contrast with MINS-BC, where it plays an important role, except in the asymptotics (as expected from trust-region convergence theory).

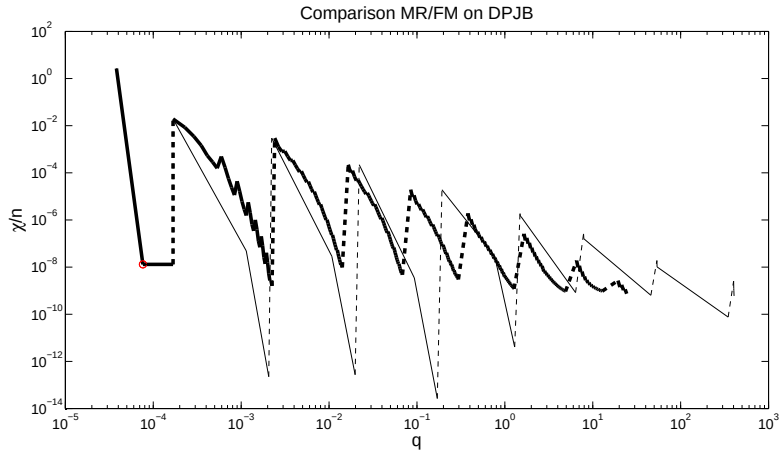

Figure 4.11: History of the scaled criticality measure on DPJB. *As above, 3both axis are logarithmic.*

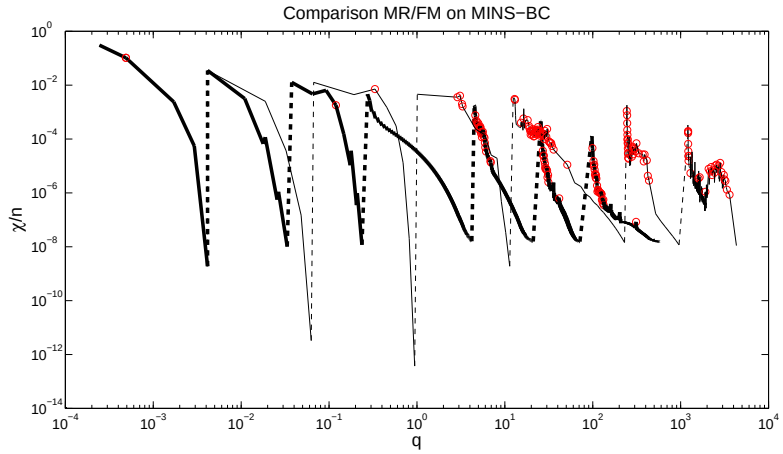

Figure 4.12: History of the scaled criticality measure on MINS-BC. A small circle surrounds the iterations where the trust region is active. *As above, both axis are logarithmic.*

5 Conclusion and perspectives

We have presented an implementation of the recursive multilevel trust-region algorithm for bound-constrained problems proposed by Gratton et al. (2008), as well as numerical

code and has been obtained once and for all at the beginning of the calculation by applying an optimized finite-difference scheme (see Powell and Toint, 1979).

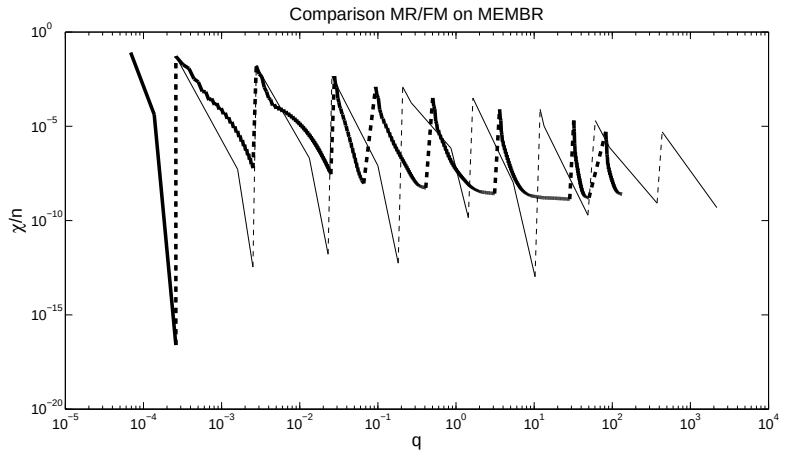

Figure 4.13: History of the scaled criticality measure on MEMBR. *As above, both axis are logarithmic.*

experience on multilevel test problems. A suitable choice of the algorithm's parameters has been identified on these problems, yielding a good compromise between reliability and efficiency. The resulting default algorithm has then be compared to alternative optimization techniques, such as mesh refinement and direct solution of the fine-level problem.

The authors are well aware that continued experimentation is needed on a larger spectrum of applications, but the numerical experience gained so far is very encouraging. Further comparison with other proposals, such as those by Kornhuber (1994,1996), is also desirable.

The extension of the method beyond geometric multigrid applications is also currently considered in conjunction with algebraic multigrid techniques. A more ambitious development involving the inclusion of constraints into the problem formulation is the object of ongoing research.

References

B. M. Averick and J. J. Moré. The Minpack-2 test problem collection. Technical Report ANL/MCS-TM-157, Mathematics and Computer Science, Argonne National Laboratory, Argonne, Illinois, USA, 1991.

A. Borzi and K. Kunisch. A globalisation strategy for the multigrid solution of elliptic optimal control problems. *Optimization Methods and Software*, **21**(3), 445–459, 2006.

W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM, Philadelphia, USA, 2nd edn, 2000.

M. M. Bronstein, A. M. Bronstein, R. Kimmel, and I. Yavneh. A multigrid approach for multi-dimensional scaling. Talk at the 12th Copper Mountain Conference on Multigrid Methods, 2005.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal on Numerical Analysis*, **25**(182), 433–460, 1988. See also same journal 26:764–767, 1989.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. LANCELOT*: a Fortran package for large-scale nonlinear optimization (Release A)*. Number 17 *in* 'Springer Series in Computational Mathematics'. Springer Verlag, Heidelberg, Berlin, New York, 1992.

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.

A. R. Conn, N. I. M. Gould, A. Sartenaer, and Ph. L. Toint. Global convergence of a class of trust region algorithms for optimization using inexact projections on convex constraints. *SIAM Journal on Optimization*, **3**(1), 164–221, 1993.

A. R. Conn, L. N. Vicente, and C. Visweswariah. Two-step algorithms for nonlinear optimization with structured applications. *SIAM Journal on Optimization*, **9**(4), 924–947, 1999.

J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1983. Reprinted as *Classics in Applied Mathematics 16*, SIAM, Philadelphia, USA, 1996.

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

M. Domorádová and Z. Dostál. Projector preconditioning for bound-constrained quadratic optimization. *Linear Algebra and its Applications*, **(to appear)**, 2007.

M. Emilianenko. A nonlinear energy-based multilevel quantization scheme. Talk at the 12th Copper Mountain Conference on Multigrid Methods, 2005.

M. Fisher. Minimization algorithms for variational data assimilation. *in* 'Recent Developments in Numerical Methods for Atmospheric Modelling', pp. 364–385. ECMWF, 1998.

E. Gelman and J. Mandel. On multilevel iterative methods for optimization problems. *Mathematical Programming*, **48**(1), 1–17, 1990.

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

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, second edn, 1989.

N. I. M. Gould, D. Orban, A. Sartenaer, and Ph. L. Toint. Sensitivity of trust-region algorithms on their parameters. *4OR, Quarterly Journal of the Italian, French and Belgian OR Societies*, **3**(3), 227–241, 2005.

S. Gratton, M. Mouffe, Ph. L. Toint, and M. Weber-Mendonça. A recursive trust-region method in infinity norm for bound-constrained nonlinear optimization. *IMA Journal of Numerical Analysis*, **(to appear)**, 2008.

S. Gratton, A. Sartenaer, and Ph. L. Toint. Numerical experience with a recursive trust-region method for multilevel nonlinear optimization. Technical Report 06/01, Department of Mathematics, University of Namur - FUNDP, Namur, Belgium, 2006.

S. Gratton, A. Sartenaer, and Ph. L. Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, **(to appear)**, 2007.

P. C. Hansen. Regularization tools: A Matlab package for analysis and solution of discrete ill-posed problems. *Numerical Algorithms*, **6**, 1–35, 1994.

P. W. Hemker and G. M. Johnson. Multigrid approach to Euler equations. *in* S. F. McCormick, ed., 'Multigrid methods', Vol. 3 of *Frontiers in Applied Mathematics*, pp. 57–72, Philadelphia, USA, 1987. SIAM.

R. Kornhuber. Monotone multigrid methods for elliptic variational inequalities I. *Numerische Mathematik*, **69**, 167–184, 1994.

R. Kornhuber. Monotone multigrid methods for elliptic variational inequalities II. *Numerische Mathematik*, **72**, 481–499, 1996.

M. Lewis and S. G. Nash. Model problems for the multigrid optimization of systems governed by differential equations. *SIAM Journal on Scientific Computing*, **26**(6), 1811–1837, 2005.

J. J. Moré, B. S. Garbow, and K. E. Hillstrom. Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, **7**(1), 17–41, 1981.

S. G. Nash. A multigrid approach to discretized optimization problems. *Optimization Methods and Software*, **14**, 99–116, 2000.

J. Nocedal and Y. Yuan. Combining trust region and line search techniques. *in* Y. Yuan, ed., 'Advances in Nonlinear Programming', pp. 153–176, Dordrecht, The Netherlands, 1998. Kluwer Academic Publishers.

J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, London, 1970.

M. J. D. Powell and Ph. L. Toint. On the estimation of sparse Hessian matrices. *SIAM Journal on Numerical Analysis*, **16**(6), 1060–1074, 1979.

Ph. L. Toint. VE08AD, a routine for partially separable optimization with bounded variables. *Harwell Subroutine Library*, **2**, 1983.

Ph. L. Toint. VE10AD, a routine for large scale nonlinear least squares. *Harwell Subroutine Library*, **2**, 1987.

U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Elsevier, Amsterdam, The Netherlands, 2001.

C. R. Vogel. *Computational Methods for Inverse Problems*, Vol. 23 of *Computational Frontiers in Applied Mathematics*. SIAM, Philadelphia, USA, 2002.

A The generalized Cauchy point

We give a proof of the generalized Cauchy condition mentioned in Section 3.1.

Lemma A.1 *Assume that the first unidimensional minimization in the first smoothing cycle at iteration (i,k) is performed along the j_m-th coordinate axis, where j_m is determined by (3.20) and (3.21), and results in a stepsize α_{j_m}. Then (2.18) holds for $s_{i,k} = \alpha_{j_m} e_{i,j_m}$.*

Proof. We drop the indexes i and k for simplicity. First note that

$$|g_{j_m}| \geq |g_{j_m} d_{j_m}| = |\min_j g_j d_j| \geq \frac{1}{n} \left| \sum_j g_j d_j \right| = \frac{1}{n} \chi, \tag{1.28}$$

where we have used (3.21) to derive the first inequality, (3.20) to derive the first equality and the fact that (3.21) implies that $g_j d_j \leq 0$ for all j to deduce the second inequality. Next observe that the line minimization along the j_m-th coordinate axis may terminate in three different situations. The first is when the minimum of the quadratic model is interior to $\mathcal{W}$, in which case we obtain that $H_{j_m,j_m} > 0$, that $\alpha_{j_m} = |g_{j_m}|/H_{j_m,j_m}$ and also that

$$m(x) - m(x + \alpha_{j_m} e_{j_m}) = \frac{|g_{j_m}|^2}{2H_{j_m,j_m}}.$$

Using now the bound $H_{j_m,j_m} \leq 1 + \|H\|_\infty$ and (1.28), we deduce that

$$m(x) - m(x + \alpha_{j_m} e_{j_m}) \geq \frac{\chi^2}{2n^2(1 + \|H\|_\infty)}. \tag{1.29}$$

The second situation is when the line minimizer is on the boundary of $\mathcal{B}$, in which case $\alpha_{j_m} = \Delta$ and thus

$$m(x) - m(x + \alpha_{j_m} e_{j_m}) \geq \frac{1}{2}|g_{j_m}|\Delta \geq \frac{1}{2n}\chi\Delta, \tag{1.30}$$

where we used (1.28) to obtain the last inequality. The third possibility is when the line minimizer is on the boundary of $\mathcal{L}$. In this case, we have that $|\alpha_{j_m}| \geq |d_{j_m}|$, where d is given by (3.21), and therefore, using (1.28) again, that

$$m(x) - m(x + \alpha_{j_m} e_{j_m}) \geq \frac{1}{2}|g_{j_m}\alpha_{j_m}| \geq \frac{1}{2}|g_{j_m} d_{j_m}| \geq \frac{1}{2n}\chi.$$

Combining this bound with (1.29) and (1.30), we thus obtain that (2.18) holds with $\kappa_{\rm red} = 1/2n^2$. □

B Test problems

We have build a suite of test problems as extensive as we could, from a variety of sources. We have kept the problems already discussed in Gratton, Sartenaer and Toint (2006) and have also used Lewis and Nash (2005) and the Minpack-2 collection (Averick and Moré, 1991) extensively. In what follows, we denote by S_2 and S_3 respectively the unit square and cube

$$S_2 = [0,1] \times [0,1] = \{(x,y), 0 \leq x \leq 1, 0 \leq y \leq 1\}$$

and

$$S_3 = [0,1] \times [0,1] \times [0,1] = \{(x,y,z), 0 \leq x \leq 1, 0 \leq y \leq 1, 0 \leq z \leq 1\}.$$

We also denote by $\mathcal{H}^1(\mathcal{D})$ the Hilbert space of all functions with compact support in the domain $\mathcal{D}$ such that v and $||\nabla v||^2$ belong to $\mathcal{L}^2(\mathcal{D})$, and by $\mathcal{H}^1_0(\mathcal{D})$ its subspace consisting of all function vanishing on the domain's boundary. For all problems, the starting value of the unknown function is chosen to be equal to one (at the finest level).

B.1 DNT: a Dirichlet-to-Neumann transfer problem

Let S be the square $[0,\pi] \times [0,\pi]$ and let Γ be its lower edge defined by $\{(x,y), 0 \leq x \leq \pi, y = 0\}$. The Dirichlet-to-Neumann transfer problem (Lewis and Nash, 2005) consists in finding the function $a(x)$ defined on $[0,\pi]$, that minimizes

$$\int_0^\pi \left(\frac{\partial u}{\partial y}(x,0) - f(x)\right)^2,$$

where $u(x,y)$ is the solution of the boundary value problem

$$\begin{array}{lcll} \Delta u & = & 0 & \text{in } S, \\ u(x,y) & = & a(x) & \text{on } \Gamma, \\ u(x,y) & = & 0 & \text{on } \partial S \backslash \Gamma, \end{array}$$

where Δ is the Laplacian operator. The problem is a 1D minimization problem, but the computations of the objective function, gradient and Hessian involve a partial differential equation in 2D. To introduce oscillatory components in the solution, we define $f(x) = \sum_{i=1}^{15} \sin(i\,x) + \sin(40\,x)$. The discretization of the problem is performed by finite differences with the same grid spacing in the two directions. The discretized problem is a linear least-squares problem.

B.2 P2D and P3D: two quadratic examples

We consider here the two-dimensional Poisson model problem P2D for multigrid solvers defined in S_2

$$\begin{aligned} -\Delta u(x) &= f(x) \text{ in } S_2 \\ u(x) &= 0 \text{ on } \partial S_2, \end{aligned}$$

where $f(x)$ is such that the analytical solution to this problem is $u(x) = 2x_2(1-x_2) + 2x_1(1-x_1)$. This problem is discretized using a 5-point finite-difference scheme. We consider the variational formulation of this problem, given by

$$\min_{x \in \mathbb{R}^{n_r}} \tfrac{1}{2} x^T A x - x^T b, \tag{2.31}$$

which is obviously equivalent to the linear system $Ax = b$, where A and b are the discretizations of the Laplacian and the right-hand side f, respectively. The main purpose of this example is to illustrate that our multilevel algorithm exhibits performances similar to traditional linear multigrid solvers on a quadratic model problem.

Problem P3D is a more nonlinear 3D version of P2D. We consider the differential equation

$$\begin{aligned} -(1+\sin^2(3\pi x_1))\Delta u(x) &= f(x) \quad \text{in } S_3, \\ u(x) &= 0 \qquad \text{on } \partial S_3. \end{aligned}$$

The right-hand side $f(x)$ is chosen such that $u(x) = x_1(1-x_1)x_2(1-x_2)x_3(1-x_3)$ is the desired solution. The Laplacian is discretized using the standard 7-point finite-difference approximation on a uniform 3D mesh. As for P2D, the solution algorithms are applied to the variational formulation (2.31).

B.3 MINS-SB, MINS-OB, MINS-BC and MINS-DMSA: four minimum surface problems

The domain of calculus of variation consists in finding stationary values v of integrals of the form $\int_a^b f(v, \dot{v}, x)\, dx$, where $\dot{v}$ is the first-order derivative of v. The multilevel trust-region algorithm can be applied to discretized versions of problems of this type. As representative of these, we consider several variants of the minimum surface problem

$$\min_{v \in \mathcal{K}} \int_{S_2} \sqrt{1 + \|\nabla_x v\|_2^2}$$

where $\mathcal{K} = \left\{ v \in H^1(S_2) \mid v(x) = v_0(x) \text{ on } \partial S_2 \right\}$. This convex problem is discretized using a finite-element basis defined using a uniform triangulation of S_2, with same grid spacing h along the two coordinate directions. The basis functions are the classical P1 functions which are linear on each triangle and take value 0 or 1 at each vertex. The boundary condition $v_0(x)$ is chosen as

$$v_0(x) = \begin{cases} f(x_1), & x_2 = 0, \quad 0 \le x_1 \le 1, \\ 0, & x_1 = 0, \quad 0 \le x_2 \le 1, \\ f(x_1), & x_2 = 1, \quad 0 \le x_1 \le 1, \\ 0, & x_1 = 1, \quad 0 \le x_2 \le 1, \end{cases}$$

where $f(x_1) = x_1(1-x_1)$ (for MINS-SB) or $f(x_1) = \sin(4\pi x_1) + \frac{1}{10}\sin(120\pi x_1)$ (for MINS-OB). To define problem MINS-BC, we introduce, in MINS-SB, the following lower bound constraint:

$$v(x) \ge \sqrt{2} \qquad \text{whenever} \quad \frac{4}{9} \le x_1, x_2 \le \frac{5}{9},$$

thereby creating an obstacle problem where the surface is constrained in the middle of the domain. The fourth variant of the minimum surface problem, MINS-DMSA, is the Enneper

problem proposed in Minpack-2, where the domain is now given by $\mathcal{D} = (-\frac{1}{2}, \frac{1}{2}) \times (-\frac{1}{2}, \frac{1}{2})$. The boundary condition is chosen on $\partial\mathcal{D}$ as

$$v_{\mathcal{D}}(x) = u^2 - v^2,$$

where u and v are the unique solutions to the equations

$$x_1 = u + uv^2 - \frac{1}{3}u^3, \quad x_2 = -v - u^2 v + \frac{1}{3}v^3.$$

B.4 MEMBR: a membrane problem

We consider the problem suggested by Domorádová and Dostál (2007) given by

$$\min_{u \in \mathcal{K}} \int_{S_2} \Big(||\nabla u(x)||_2^2 + u(x) \Big)$$

where the boundary of S_2 is composed of three parts: $\Gamma_u = \{0\} \times [0,1]$, $\Gamma_l = \{1\} \times [0,1]$ and $\Gamma_f = [0,1] \times \{0,1\}$ and where $\mathcal{K} = \{u \in \mathcal{H}^1(S_2) \mid u(x) = 0 \text{ on } \Gamma_u \text{ and } l \leq u(x) \text{ on } \Gamma_l\}$. The obstacle l on the boundary Γ_l is defined by the upper part of the circle with the radius one and center $S = (1; 0.5; -1.3)$.

The solution of this problem can be interpreted as the displacement of the membrane under the traction defined by the unit density. The membrane is fixed on Γ_u and is not allowed to penetrate the obstacle on Γ_l. We discretized the problem by piecewise linear finite elements using a regular triangular grid.

B.5 IGNISC, DSSC and BRATU: three combustion/Bratu problems

We first consider the following optimal-control problem (IGNISC), introduced by Borzi and Kunisch (2006), and related to the solid-ignition model:

$$\min_{u \in \mathcal{H}_0^1(S_2)} \left[\int_{S_2} \Big(u(x) - z \Big)^2 + \frac{\beta}{2} \int_{S_2} \Big(e^{u(x)} - e^z \Big)^2 + \frac{\nu}{2} \int_{S_2} \| \Delta u(x) - \delta e^{u(x)} \|_2^2 \right].$$

For the numerical tests, we chose $\nu = 10^{-5}$, $\delta = 6.8$, $\beta = 6.8$ and $z = \frac{1}{\pi^2}$.

The second problem of this type is the steady-state combustion problem DSSC of Minpack 2, stated as the infinite-dimensional optimization problem

$$\min_{u \in \mathcal{H}_0^1(S_2)} \int_{S_2} \Big(\tfrac{1}{2} ||\nabla u(x)||_2^2 - \lambda e^{u(x)} \Big)$$

and $\lambda = 5$ is a parameter. This problem is the variational formulation of the boundary value problem

$$\begin{aligned} -\Delta u(x) &= \lambda e^{u(x)}, & x \in S_2, \\ u(x) &= 0, & x \in \partial S_2. \end{aligned}$$

The third variant is a simple least-squares formulation of the same problem, where we solve

$$\min_{u \in \mathcal{H}_0^1(S_2)} \int_{S_2} \| \Delta u(x) + \lambda e^{u(x)} \|_2^2,$$

where $\lambda = 6.8$. For all these convex problems, we use standard 5-point finite differences on a uniform grid.

B.6 NCCS and NCCO: two nonconvex optimal control problems

We introduce the nonlinear least-squares problem

$$\min_{u,v\in\mathcal{H}_0^1(S_2)} \left[\int_{S_2}(u(x)-u_0(x))^2+\int_{S_2}(v(x)-v_0(x))^2+\int_{S_2}\|\Delta u(x)-v(x)u(x)+f_0(x)\|_2^2\right].$$

We distinguish two variants: the first with relatively smooth target functions and the second with more oscillatory ones. Theses functions $v_0(x)$ and $u_0(x)$ are defined on S_2 by

$$\begin{aligned} v_0(x) = u_0(x) &= \sin(6\pi x_1)\sin(2\pi x_2) \qquad \text{(for NCCS)}\\ v_0(x) = u_0(x) &= \sin(128\pi x_1)\sin(32\pi x_2) \qquad \text{(for NCCO)} \ . \end{aligned}$$

The function $f_0(x)$ is such that $-\Delta u_0(x)+v_0(x)u_0(x) = f_0(x)$ on S_2. This problem corresponds to a penalized version of a constrained optimal control problem, and is discretized using finite differences. The nonconvexity of the resulting discretized fine-grid problem has been assessed by a direct eigenvalue computation on the Hessian of the problem.

B.7 DPJB: pressure distribution in a journal bearing

The journal bearing problem arises in the determination of the pressure distribution in a thin film of lubricant between two circular cylinders. This problem is again proposed by Minpack 2, and is of the form

$$\min_{v\in\mathcal{K}} \tfrac{1}{2}\int_{\mathcal{D}}\Big(\, w_q(x)||\nabla v(x)||_2^2-\tfrac{1}{10}w_l(x)v(x)\,\Big)$$

where

$$w_q(x) = (1+\tfrac{1}{10}\cos x_1)^3 \quad \text{and} \quad w_l(x) = \tfrac{1}{10}\sin x_1$$

for some constant $\epsilon\in(0,1)$ and $\mathcal{D} = (0,2\pi)\times(0,20)$. The convex set $\mathcal{K}$ is defined by $\mathcal{K} = \{v\in\mathcal{H}_0^1(\mathcal{D}) \mid v(x)\geq 0 \ \text{ on } \ \mathcal{D}\}$. A finite-element approach of this problem is obtained by minimizing over the space of piecewise linear functions v with values $v_{i,j}$ at $z_{i,j}\in\mathbb{R}^2$ which are the vertices of the regular triangulations of $\mathcal{D}$.

B.8 DEPT: an elastic-plastic torsion problem

The elastic-plastic torsion problem DEPT from Minpack 2 arises from the determination of the stress field on an infinitely long cylindric bar. The infinite-dimensional version of this problem is of the form

$$\min_{v\in\mathcal{K}} \tfrac{1}{2}\int_{S_2}\Big(\, ||\nabla v(x)||_2^2-5v(x)\,\Big).$$

The convex set $\mathcal{K}$ is defined by $\mathcal{K} = \{v\in\mathcal{H}_0^1(S_2) \mid |v(x)|\leq \text{dist}(x,\partial S_2) \ \text{ on } \ S_2\}$, where $\text{dist}(.,\partial S_2)$ is the distance function to the boundary of S_2. A finite-element approach of this problem is obtained by minimizing over the space of piecewise linear functions v with values $v_{i,j}$ at $z_{i,j}\in\mathbb{R}^2$ which are the vertices of the regular triangulations of S_2.

B.9 DODC: an optimal design with composite materials

The Minpack 2 DODC optimal design problem is defined by

$$\min_{v\in\mathcal{H}_0^1(S_2)} \int_{\mathcal{D}}\Big(\, \psi_\lambda(||\nabla v(x)||_2)+v(x)\,\Big)$$

where

$$\psi_\lambda(t) = \begin{cases} \frac{1}{2}\mu_2 t^2, & 0\leq t\leq t_1,\\ \mu_2 t(t-\frac{1}{2}t_1), & t_1\leq t\leq t_2,\\ \frac{1}{2}\mu_1(t^2-t_2^2)+\mu_2 t_1(t_2-\frac{1}{2}t_1), & t_2\leq t, \end{cases}$$

with the breakpoints t_1 and t_2 defined by

$$t_1 = \sqrt{2\lambda\frac{\mu_1}{\mu_2}} \quad \text{and} \quad t_2 = \sqrt{2\lambda\frac{\mu_2}{\mu_1}},$$

and we choose $\lambda = 0.008$, $\mu_1 = 1$ and $\mu_2 = 2$. A finite-element approach of this problem is obtained by minimizing over the space of piecewise linear functions v with values $v_{i,j}$ at $z_{i,j} \in \mathbb{R}^2$ which are the vertices of the regular triangulations of S_2.

B.10 MOREBV: a nonlinear boundary value problem

The MOREBV problem is adapted (in infinite dimensions) from Moré, Garbow and Hillstrom (1981) and is described by

$$\min_{u \in \mathcal{H}_0^1(S_2)} \int \|\Delta u(x) - \tfrac{1}{2}[u(x) + \langle e, x\rangle + 1]^3\|_2^2,$$

where e is the vector of all ones. Once again, the problem is discretized by linear finite-elements on regular triangular grids.

C Complete numerical results

We give here the complete numerical results for all test problems and all variants. The columns of the following tables report CPU time (in seconds), the number of matrix-vector products or smoothing cycles and the number of objective function/gradient/Hessian evaluations.

P2D	CPU	Mv prods	Eval f	Eval g	eval H
FM	26.05	13.52	4.66	3.38	1.33
MR	569.72	1494.99	2.67	2.67	1.33
MF	72.85	52.93	10.00	10.00	1.00
AF	1122.83	3022.00	4.00	4.00	1.00
MINS-SB	CPU	Mv prods	Eval f	Eval g	eval H
FM	153.92	81.89	26.43	18.62	11.91
MR	3600.00	-	-	-	-
MF	3600.00	-	-	-	-
AF	3600.00	-	-	-	-
NCCS	CPU	Mv prods	Eval f	Eval g	eval H
FM	331.89	69.57	69.77	1100.27	0.00
MR	279.51	1342.26	2.68	57.50	0.00
MF	3600.00	-	-	-	-
AF	3600.00	-	-	-	-
DPJB	CPU	Mv prods	Eval f	Eval g	eval H
FM	83.61	11.17	16.98	28.98	0.00
MR	247.71	341.66	5.02	17.02	0.00
MF	1390.02	297.00	297.00	306.00	0.00
AF	3600.00	-	-	-	-
MEMBR	CPU	Mv prods	Eval f	Eval g	eval H
FM	153.96	76.73	98.43	98.43	1.33
MR	292.43	2103.35	3.00	3.00	1.33
MF	335.25	413.97	203.00	183.00	1.00
AF	1082.05	7423.00	43.00	43.00	1.00
MINS-BC	CPU	Mv prods	Eval f	Eval g	eval H
FM	140.02	402.25	551.00	540.88	31.64
MR	524.61	4055.91	413.59	400.60	47.15
MF	161.84	414.09	581.00	560.00	84.00
AF	2706.41	3935.00	1105.00	1001.00	1103.00
DNT	CPU	Mv prods	Eval f	Eval g	eval H
FM	6.73	33.62	9.33	7.33	1.33
MR	4.58	246.40	2.66	2.66	1.33
MF	24.41	131.82	37.00	28.00	1.00
AF	5.20	299.00	3.00	3.00	1.00
P3D	CPU	Mv prods	Eval f	Eval g	eval H
FM	28.78	39.38	8.92	8.64	1.33
MR	18.33	102.08	2.82	2.74	1.33
MF	47.47	64.75	12.00	12.00	1.00
AF	626.07	987.00	257.00	142.00	1.00
DEPT	CPU	Mv prods	Eval f	Eval g	eval H
FM	8.58	3.37	1.92	4.43	0.00
MR	95.44	206.38	1.66	4.25	0.00
MF	69.55	52.93	10.00	18.00	0.00
AF	1364.45	3019.00	4.00	12.00	0.00

DODC	CPU	Mv prods	Eval f	Eval g	eval H
FM	36.00	218.92	65.98	220.55	0.00
MR	184.23	4014.31	38.43	354.44	0.00
MF	58.58	282.99	93.00	399.00	0.00
AF	894.76	11472.00	493.00	4707.00	0.00
MINS-OB	CPU	Mv prods	Eval f	Eval g	eval H
FM	27.49	305.67	84.99	61.42	21.33
MR	116.73	1807.44	26.93	18.43	25.60
MF	70.44	564.15	261.00	185.00	69.00
AF	1545.63	5955.00	475.00	388.00	460.00
MINS-DMSA	CPU	Mv prods	Eval f	Eval g	eval H
FM	18.23	88.74	26.89	138.65	0.00
MR	289.64	2860.34	26.31	242.01	0.00
MF	73.41	200.25	137.00	591.00	0.00
AF	1196.81	5677.00	428.00	4116.00	0.00
IGNISC	CPU	Mv prods	Eval f	Eval g	eval H
FM	398.18	65.60	14.98	13.91	1.34
MR	488.22	1882.86	2.69	2.69	1.36
MF	398.34	257.11	60.00	46.00	1.00
AF	2330.42	11572.00	6.00	6.00	5.00
DSSC	CPU	Mv prods	Eval f	Eval g	eval H
FM	12.11	3.41	1.93	4.85	0.00
MR	122.32	211.51	1.67	4.68	0.00
MF	1051.56	760.65	165.00	134.00	0.00
AF	3183.85	6012.00	6.00	42.00	0.00
BRATU	CPU	Mv prods	Eval f	Eval g	eval H
FM	10.15	3.68	2.06	1.91	0.33
MR	91.71	203.00	1.67	1.67	0.33
MF	236.82	184.41	43.00	32.00	1.00
AF	2314.11	5458.00	6.00	6.00	4.00
NCCO	CPU	Mv prods	Eval f	Eval g	eval H
FM	224.20	44.01	35.33	791.37	0.00
MR	3589.62	17993.03	3.33	43.37	0.00
MF	3600.00	-	-	-	-
AF	3600.00	-	-	-	-
MOREBV	CPU	Mv prods	Eval f	Eval g	eval H
FM	41.73	12.83	4.54	3.60	0.33
MR	3600.00	-	-	-	-
MF	704.88	301.01	55.00	44.00	1.00
AF	3600.00	-	-	-	-