Combining and scaling descent and negative curvature directions

**Catarina P. Avelino · Javier M. Moguerza ·
Alberto Olivares · Francisco J. Prieto**

Abstract The aim of this paper is the study of different approaches to combine and scale, in an efficient manner, descent information for the solution of unconstrained optimization problems. We consider the situation in which different directions are available in a given iteration, and we wish to analyze how to combine these directions in order to provide a method more efficient and robust than the standard Newton approach. In particular, we will focus on the scaling process that should be carried out before combining the directions. We derive some theoretical results regarding the conditions necessary to ensure the convergence of combination procedures following schemes similar to our proposals. Finally, we conduct some computational

Catarina P. Avelino was partially supported by Portuguese FCT postdoctoral grant SFRH/BPD/20453/2004 and by the Research Unit CM-UTAD of University of Trás-os-Montes e Alto Douro. Javier M. Moguerza and Alberto Olivares were partially supported by Spanish grant MEC MTM2006-14961-C05-05.
Francisco J. Prieto was partially supported by grant MTM2007-63140 of the Spanish Ministry of Education.

C. P. Avelino (✉)
Department of Mathematics, UTAD, Vila Real, Portugal
e-mail: cavelino@utad.pt

J. M. Moguerza · A. Olivares
Department of Statistics and Operational Research,
University Rey Juan Carlos, Madrid, Spain
e-mail: javier.moguerza@urjc.es

A. Olivares
e-mail: alberto.olivares@urjc.es

F. J. Prieto
Department of Statistics, University Carlos III de Madrid, Madrid, Spain
e-mail: franciscojavier.prieto@uc3m.es

experiments to compare these proposals with a modified Newton's method and other procedures in the literature for the combination of information.

Keywords Line search · Negative curvature · Nonconvex optimization

AMS Subject Classification 49M37 · 65K05 · 90C30

1 Introduction

We are interested in the study of algorithms to compute in an efficient manner solutions for unconstrained nonconvex problems of the form:

$$\min_x f(x), \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is at least twice continuously differentiable. This problem has been extensively studied in the literature (see for example Gill et al. (8) or Fletcher (5)), and different classes of algorithms have been proposed to compute local solutions of the problem. Most of these methods are based on the generation of a sequence of iterates, updated using the information derived from a single search direction. In most cases, the methods compute an approximation to the Newton direction based on second-order approximations to the problem, and ensure reasonable global convergence properties by adjusting the size of the direction through either a linesearch or a trust-region approach.

Nevertheless, it has been noted in the literature that in most practical cases it is possible to generate in each iteration additional information to update the iterates at a cost that is not significantly higher than that required by the classical Newton approach, see for example Moré and Sorensen (15), Fiacco and McCormick (4), or Forsgren and Murray (6). The previous references show the theoretical advantages of including directions of negative curvature as a part of an optimization algorithm. However, there are just a few works treating practical aspects regarding the use of this kind of information (see, for instance, Moguerza and Prieto (13,14)). Therefore it seems interesting to consider the potential improvement that the use of this information (for example descent and negative curvature directions) may imply for an unconstrained optimization algorithm, both in terms of its efficiency and its robustness. Also, for large-scale problems it may be computationally expensive to obtain an exact Newton direction in each iteration. The methods most commonly used are based on quasi-Newton approximations to the Hessian matrix, or in approximate solutions to the Newton system of equations. In these cases, the analysis of potential improvements based on combining several of these approaches seems particularly relevant, as no practical evidence suggests that any of the approximate Newton methods are in general superior to the others, and no clear evidence exists to support a strategy based on the a priori selection of one of them.

The use of combinations of directions has been analyzed in the literature for example in Moré and Sorensen (15), Mukai and Polak (16), Goldfarb (9), Moguerza and Prieto (13,14) or Olivares et al. (17). Although out of the scope of this paper, based on the conjugate gradient methodology there are linesearch procedures for unconstrained optimization which are useful for solving large scale problems. These methods have well-known convergence properties (see Hager and Zhang (12) and the references therein), and have been used in some practical engineering applications (see, for instance, Sun et al. (20)). There are some other works using directions of negative curvature within conjugate gradient schemes (see Sanmatías and Vercher (18) and the references therein). For instance, in Gould et al. (10), in each iteration the best direction is chosen and a standard linesearch is conducted. Another method based in the selection of directions is suggested by Sanmatías and Roma (19).

Our aim in this paper is to show that scaling the negative curvature directions and eventually other directions before performing a line search yields significant improvements in the efficiency of optimization algorithms. In addition, our methodology is general enough to be applied to an undetermined number of directions (this is important as most methods in the literature are able to combine a maximum of two directions). We will consider the particular case when the directions available are the Newton direction, the gradient and a negative curvature direction. Our approach is closely related to the standard Newton's method. Therefore it provides a more direct evaluation of the potential advantages of a combined information approach, when compared to the Newton algorithm. Also, the adjustments of the parameters to obtain efficient implementations of the combination algorithms are simpler within this setting. We implement some proposals in an efficient manner and study the conditions that must be imposed on these approaches to ensure reasonable global convergence properties to second-order critical points of problem (1). In addition, we compare their practical performance and effectiveness through a computational experiment based on a set of 119 small optimization problems from the CUTEr collection (11). The results and their analysis provide important insights both for practical applications in an improved Newton method setting and for possible extensions to large-scale problems.

The rest of the paper is organized as follows: in Sect. 2 we describe the algorithms proposed to compute local solutions for unconstrained optimization problems. Section 3 presents several global convergence results for the algorithms. Section 4 describes the computational experiment we have carried out. Finally, in Sect. 5 we present some final comments and conclusions.

2 General description of the algorithms

In this section we present and describe two algorithmic models for the solution of problem (1). Our description of the algorithms starts with a presentation of a common framework to both methods, and then introduces the different approaches specific to each algorithm. Given a set of p directions d_{ik}, computed in an iteration k of the optimization algorithm, our common approach for all the alternatives that we consider in this work is to define a combination of these directions to obtain a search direction d_k as

$$d_k = \sum_{i=1}^{p} \alpha_{ik} d_{ik}, \tag{2}$$

where $\alpha_{ik}, i = 1, \ldots, p$, are the coefficients to be determined by the proposed procedures at iteration k. Within this common framework, all the procedures compute a sequence of iterates $\{x_k\}$ from an initial approximation x_0, as

Table 1 General description of the algorithms

Main Algorithm	
Step 0	**[Initialization]** *Select* $x_0 \in \mathbb{R}^n$ *and constants* $\omega_* > 0, \sigma \in (0, 1/2)$. *Set* $k = 0$
Step 1	**[Test for convergence]** *If* $\|\nabla f(x_k)\| \leq \omega_*$ and $\lambda_{\min}(\nabla^2 f(x_k)) \geq -\omega_*$, *then stop*
Step 2	**[Computation of the search directions]** *Compute* d_{ik} *and* $\bar{d}_{ik}$
Step 3	**[Computation of steplengths]** *Compute* $\alpha_{ik}, \bar{\alpha}_{ik}$ *and define* $d_k = \sum_i \alpha_{ik} d_{ik} + \sum_i \bar{\alpha}_{ik} \bar{d}_{ik}$. *Perform a conventional linesearch to compute a steplength* ζ_k
Step 4	**[New iterate]** *Set* $x_{k+1} = x_k + \zeta_k d_k$, $k = k + 1$ *and go to Step 1*

$$x_{k+1} = x_k + \zeta_k d_k,$$

where ζ_k is a steplength computed through a standard linesearch procedure.

As already mentioned, an important and related task is the adequate scaling of the directions used in the algorithm (see Moguerza and Prieto (13,14)). While the standard Newton direction is well-scaled, particularly close to the solution, other alternative search directions may not be, implying a potential inefficiency in the resulting algorithm. Our proposal handles these problems by adjusting the scale of the available directions through the application of a few iterations of an optimization algorithm on a simplified local model for the problem. The computation of a set of values for α_{ik} on a smaller subspace as an optimization problem has already been treated by Byrd et al. (2). In particular, we propose and analyze two different algorithms to compute the coefficients in the combination: a Newton method that, given the directions d_{ik}, is applied to determine the initial values of α_{ik} for a linesearch; and a mixed approach that computes a set of values for α_{ik} by solving a trust-region subproblem, and then performing a linesearch to obtain the next iterate.

In what follows, and to improve the clarity of the presentation of convergence results in Sect. 3, we use a slightly modified notation for the directions combined in (2). We denote as d_{ik} those directions that are related to descent properties for problem (1), while we introduce the notation $\bar{d}_{ik}$ for the directions related to negative curvature properties of (1), if they are available at iteration k. In this way, d_k in (2) is obtained from the available directions as

$$d_k \equiv \sum_i \alpha_{ik} d_{ik} + \sum_i \bar{\alpha}_{ik} \bar{d}_{ik}, \tag{3}$$

for appropriate values of the scalars $\alpha_{ik}, \bar{\alpha}_{ik}$.

The general framework (the basic algorithm) is summarized in Table 1, where $\lambda_{\min}(H)$ denotes the smallest eigenvalue of a given matrix H. Note that we use the Newton direction whenever it satisfies a sufficient descent condition. The motivation for this choice is to ensure a quadratic local convergence rate for the algorithm. This choice works well in practice, and is consistent with the good local properties of the Newton direction close to a local solution. The methods differ only in the way the

Table 2 Computation of steplengths

Step 3	Computation of steplengths
Step 3a	**[Initialization]** *Set* $\hat{k} = 0$. *Select initial steplengths* α_{i0} *and* $\bar{\alpha}_{i0}$, $\xi \in (0, 1)$, $\sigma \in (0, 1/2)$ *and a positive integer* $\hat{k}_{\max}$
Step 3b	**[Test for convergence]** *Let* $d_{\hat{k}} = \sum_i \alpha_{i\hat{k}} d_{ik} + \sum_i \bar{\alpha}_{i\hat{k}} \bar{d}_{ik}$. *If* $\\|\nabla f(x_k + d_{\hat{k}})\\| \le \omega_*$ *or* $\hat{k} = \hat{k}_{\max}$, *then go to Step 3e*
Step 3c	**[Computation of $\alpha_{i\hat{k}}$ and $\bar{\alpha}_{i\hat{k}}$]** *Apply either Newton's method or a trust-region approach to update* $\alpha_{i\hat{k}}$ *and* $\bar{\alpha}_{i\hat{k}}$
Step 3d	**[New iterate]** *Set* $\hat{k} = \hat{k} + 1$ *and go to Step 3b*
Step 3e	**[Linesearch]** *Let* $\alpha_{ik} = \alpha_{i\hat{k}}$ *and* $\bar{\alpha}_{ik} = \bar{\alpha}_{i\hat{k}}$. *Also let* $d_k = \sum_i \alpha_{ik} d_{ik} + \sum_i \bar{\alpha}_{ik} \bar{d}_{ik}$. *Compute* $\zeta_k = \xi^l$, *where* l *is the smallest nonnegative integer such that* $f(x_k + \xi^l d_k) \le f(x_k) + \sigma \left(\xi^l \nabla f(x_k)^T d_k + \xi^{2l} \min\left(0, d_k^T \nabla^2 f(x_k) d_k\right)\right)$
Step 3f	**[Return]** *Return* d_k *and* ζ_k *to the Main Algorithm*

values for step lengths α_{ik} and $\bar{\alpha}_{ik}$ are computed, in particular, the way in which Step 3 is carried out. This step is described in Table 2. More specifically, our proposals are obtained by performing Step 3c in two different manners. This step corresponds to the scaling process that should be carried out before combining the directions. Next, we describe each one of the two proposed implementations for this step.

2.1 Newton-based scaling method (NSM)

Our first proposal is based on the application of Newton steps to obtain reasonable values for the steplengths. Once a reasonable set of values is obtained, either because the gradient of the objective function is small enough or because a sufficient number of Newton steps have been taken, the resulting values are further refined by imposing a sufficient descent condition and performing a linesearch to obtain a sufficient descent direction as a combination of the available directions in the current iteration. We will denote this algorithm as **NSM**.

Since the Newton direction is well-scaled, in our proposals we adjust only the steplengths associated with the gradient and the negative curvature directions, that is, we keep fixed the steplength associated with the original Newton direction. In this way, the scaling of these two directions will be adequate related to the Newton direction.

We now consider the different computations associated with Step 3c in Table 2 for algorithm **NSM**. For this method, the value for the final direction is obtained by adjusting the steps along the directions by computing Newton updates for the problem

$$\min_{(\alpha_{ik}, \bar{\alpha}_{ik})} \quad f\left(x_k + \sum_i \alpha_{ik} d_{ik} + \sum_i \bar{\alpha}_{ik} \bar{d}_{ik}\right) \tag{4}$$

Table 3 Step 3c performance for **NSM** algorithm

Computation of steplength-algorithm **NSM**	
Step 3c	**[Computation of $\alpha_{i\hat{k}}$ and $\bar{\alpha}_{i\hat{k}}$]** *Solve system* (5) *and obtain the updated steplengths from* (6)

by determining the solution for the system

$$\bar{H}_{k\hat{k}} \begin{pmatrix} \Delta\alpha_{i\hat{k}} \\ \Delta\bar{\alpha}_{i\hat{k}} \end{pmatrix} = -\bar{g}_{k\hat{k}}, \tag{5}$$

or a suitably modified system if the coefficient matrix has undesirable properties, where $\Delta\alpha_{i\hat{k}}$ and $\Delta\bar{\alpha}_{i\hat{k}}$ denote the changes in the corresponding steplengths, and

$$\bar{H}_{k\hat{k}} \equiv \begin{pmatrix} d_{ik}^T \\ \bar{d}_{ik}^T \end{pmatrix} \nabla^2 f\left(x_k + \sum_i \alpha_{i\hat{k}} d_{ik} + \sum_i \bar{\alpha}_{i\hat{k}} \bar{d}_{ik} \right) \left(d_{ik} \;\; \bar{d}_{ik} \right)$$

$$\bar{g}_{k\hat{k}} \equiv \begin{pmatrix} d_{ik}^T \\ \bar{d}_{ik}^T \end{pmatrix} \nabla f\left(x_k + \sum_i \alpha_{i\hat{k}} d_{ik} + \sum_i \bar{\alpha}_{i\hat{k}} \bar{d}_{ik} \right),$$

where $\nabla f(y)$ and $\nabla^2 f(y)$ denote, respectively the gradient vector and Hessian matrix of f with respect to y. That is, we project the Hessian and the gradient on the subspace determined by the search directions. Finally, we update

$$\alpha_{i(\hat{k}+1)} = \alpha_{i\hat{k}} + \Delta\alpha_{i\hat{k}}, \quad \bar{\alpha}_{i(\hat{k}+1)} = \bar{\alpha}_{i\hat{k}} + \Delta\bar{\alpha}_{i\hat{k}}. \tag{6}$$

In summary, we perform for Step 3c in **NSM** the operations indicated in Table 3.

2.2 Trust-region scaling method (TRSM)

A second, closely related proposal, computes updates for the steplengths using a trust-region approach, instead of the direct application of the Newton update. Nevertheless, it still carries out a linesearch once a sufficient number of steps have been taken, as in the preceding case. We will denote this algorithm as **TRSM**.

Consider now the implementation of Step 3c in Algorithm **TRSM**. Now we obtain the updates for the steplengths by computing a solution for the trust-region problem

$$\begin{aligned} &\min \; \bar{g}_{k\hat{k}}^T \begin{pmatrix} \delta\alpha_{i\hat{k}} \\ \delta\bar{\alpha}_{i\hat{k}} \end{pmatrix} + \tfrac{1}{2} \left(\delta\alpha_{i\hat{k}} \;\; \delta\bar{\alpha}_{i\hat{k}} \right) \bar{H}_{k\hat{k}} \begin{pmatrix} \delta\alpha_{i\hat{k}} \\ \delta\bar{\alpha}_{i\hat{k}} \end{pmatrix} \\ &\text{s.t} \quad \left\| \left(\delta\alpha_{i\hat{k}} \;\; \delta\bar{\alpha}_{i\hat{k}} \right) \right\| \le \Delta_k, \end{aligned} \tag{7}$$

and then updating the corresponding values using (6), as before. Step 3c in **TRSM** is implemented as indicated in Table 4.

Table 4 Step 3c performance for **TRSM** algorithm

Computation of steplength-algorithm **TRSM**	
Step 3c	**[Computation of** $\alpha_{i\hat{k}}$ **and** $\bar{\alpha}_{i\hat{k}}$**]** *Solve problem* (7) *and obtain the updated steplengths from* (6)

3 Convergence properties

In this section we study the global convergence properties of the algorithms described previously. The analysis is conducted on quite general versions of the algorithms, as we wish to provide theoretical insights beyond those obtained for the computational tests considered in Sect. 4 for more specific implementations of these algorithms.

Our goal is the computation of local solutions for unconstrained optimization problems of the form given in (1), where we assume for the results presented in this Section that $f : \mathbb{R}^n \to \mathbb{R}$ is a function with Lipschitz-continuous second derivatives on some compact set. These algorithms generate a sequence of iterates $\{x_k\}$ from an initial point x_0 and a combination of directions obtained in each iteration. These directions should capture the descent and negative curvature available in the iteration.

In order to establish convergence results for the sequence $\{x_k\}$ we need to assume that problem (1) and the initial point x_0 satisfy some regularity conditions. In particular, in what follows we will assume that the following properties hold:

A1 The level set of f at the initial point,

$$\mathcal{S}_0 \equiv \{x \in \mathbb{R}^n : f(x) \le f(x_0)\}$$

is compact.

A2 The function f has Lipschitz-continuous second derivatives with constant K_f on some open set containing $\mathcal{S}_0$, that is,

$$\|\nabla^2 f(y) - \nabla^2 f(x)\| \le K_f \|y - x\|,$$

for any $y, x \in \mathcal{S}_0$.

We use the 2-norm in all the results presented in this section, except where otherwise stated. Also, to simplify the notation in what follows we will use $g_k \equiv \nabla f(x_k)$, $g^* \equiv \nabla f(x^*)$, $H_k \equiv \nabla^2 f(x_k)$ and $H^* \equiv \nabla^2 f(x^*)$, where x^* denotes a limit point of the sequence $\{x_k\}$.

Before considering any detailed convergence analysis, note that the basic iteration step in either of the algorithms consists of the computation at x_k of the individual search directions d_{ik} and $\bar{d}_{ik}$ followed by two adjustments to determine the next iterate x_{k+1}:

- the computation of a combination of the directions, that is, the computation of steplengths α_{ik} and $\bar{\alpha}_{ik}$ associated to each search direction to define a combined direction d_k using (3), and

- the performance of a conventional (backtracking) linesearch on the resulting combined direction d_k to select a steplength ζ_k as the smallest nonnegative integer l such that $\zeta_k = \xi^l$ satisfies the descent condition

  $$f(x_k + \zeta_k d_k) \le f(x_k) + \sigma \left(\zeta_k g_k^T d_k + \zeta_k^2 \min \left(0, d_k^T H_k d_k \right) \right), \tag{8}$$

  where ξ is a positive constant and $0 < \sigma < 1/2$ is also a prespecified constant. The next iterate is then defined as $x_{k+1} = x_k + \zeta_k d_k$. From the definition of ζ_k it holds

  $$\zeta_k \le 1, \tag{9}$$

  although from a convergence point of view any positive upper bound would provide equivalent results.

The traditional approach to establish the convergence of the sequence $\{x^k\}$ is based on condition (8) and the satisfaction of some sufficient descent property by d_k. In the case of the proposed algorithms, and due to the use of negative curvature directions, a first version of the sufficient descent condition considered in the proofs is given by

$$\min \left(\zeta_k g_k^T d_k, \zeta_k^2 d_k^T H_k d_k \right) \le \hat{\gamma} \min \left(-\|g_k\|^2, \lambda_{\min}(H_k) \right), \tag{10}$$

for some positive constant $\hat{\gamma}$ and all k. The following result shows that these conditions, together with the requirement that the direction d_k is a descent direction,

$$g_k^T d_k \le 0, \tag{11}$$

are sufficient to ensure convergence to a second-order critical point.

Theorem 1 *Under conditions* (8), (10), (11) *and assumptions A1 and A2, the subspace search algorithm converges to second-order critical points of problem* (1).

Proof Consider the sequence of iterates $\{x_k\}$ generated by the algorithm; from Assumption *A1* and condition (8), this sequence contains convergent subsequences and from Assumption *A2* the values of $f(x_k)$ will be bounded below. Let $\mathcal{I}$ denote the sequence of iteration indexes corresponding to any one of these convergent subsequences, such that $x_k \to x^*$ for $k \in \mathcal{I}$.

From conditions (8), (10) and (11) it must hold that

$$\begin{aligned} f(x_k + \zeta d_k) - f(x_k) &\le \sigma \left(\zeta_k g_k^T d_k + \zeta_k^2 \min \left(0, d_k^T H_k d_k \right) \right) \\ &\le \sigma \min \left(\zeta_k g_k^T d_k, \zeta_k^2 d_k^T H_k d_k \right) \\ &\le \sigma \hat{\gamma} \min \left(-\|g_k\|^2, \lambda_{\min}(H_k) \right). \end{aligned}$$

But taking limits as $k \to \infty$ along $\mathcal{I}$ and using Assumptions *A1* and *A2*, the boundedness of $f(x_k)$ and the fact that the right-hand side in the preceding inequality is nonpositive, we obtain

$$\min\left(-\|g^*\|^2, \lambda_{\min}(H^*)\right) = 0 \Rightarrow \|g^*\| = 0,\ \lambda_{\min}(H^*) \geq 0.$$

□

From this basic result, the remaining proofs in this section are aimed at establishing that the directions we use and the combinations we compute to form d_k lead to the satisfaction of conditions (8), (10) and (11). Nevertheless, to attain these results some conditions must be imposed on the information used to form d_k, both on the search directions and the steplengths. Our main contribution in this section should be to identify a set of reasonably general conditions and to prove that the desired result follows from them.

The proofs proceed by establishing sufficient conditions on the combined direction d_k to ensure the satisfaction of the required convergence conditions (8), (10) and (11). The conditions we introduce are a generalization of the well-known conditions in the literature for the unconstrained case based on linesearches. We then show that the proposed algorithms **NSM** and **TRSM** compute directions d_k that satisfy these conditions. Note though that this proposed generalization of the conditions presents some complications. For example, in general a linear combination involving negative curvature directions has curvature properties that are not linear combinations of the properties for each individual direction.

We introduce the following conditions on d_k:

- We require having descent on the combined direction, that is, we continue assuming condition (11) introduced above holds.
- Furthermore, whenever $\|g_k\|^2 \geq |\min(0, \lambda_{\min}(H_k))|$, that is, we have more descent that negative curvature in an iteration, we must have

$$\|d_k\| \leq \kappa_g \|g_k\| \tag{12}$$
$$g_k^T d_k \leq -\gamma_g \|g_k\|^2, \tag{13}$$

  for positive constants κ_g, γ_g.
- Otherwise, if $\|g_k\|^2 < |\min(0, \lambda_{\min}(H_k))|$ then we must have

$$\|d_k\| \leq \kappa_H |\lambda_{\min}(H_k)|^{1/2} \tag{14}$$
$$g_k^T d_k + d_k^T H_k d_k \leq -\gamma_H |\lambda_{\min}(H_k)|, \tag{15}$$

  for positive constants κ_H, γ_H.

These conditions will be established below to hold for the combinations computed by both algorithms. The following Lemmas show that the preceding conditions are sufficient to ensure convergence, if ζ_k is chosen as indicated in (8).

Lemma 1 *Under conditions (13) and (15) it holds that*

$$\min\left(g_k^T d_k, d_k^T H_k d_k\right) \leq \hat{\gamma} \min\left(-\|g_k\|^2, \lambda_{\min}(H_k)\right), \tag{16}$$

for some positive constant $\hat{\gamma}$ and all k.

Proof Consider two possible cases regarding the amount of descent and negative curvature available in a given iteration k:

- If we have $\|g_k\|^2 \geq |\min(0, \lambda_{\min}(H_k))|$, from (13) it holds that

$$\begin{aligned}\min\left(g_k^T d_k, d_k^T H_k d_k\right) &\leq g_k^T d_k \leq -\gamma_g \|g_k\|^2 \\ &\leq \gamma_g \min\left(-\|g_k\|^2, \lambda_{\min}(H_k)\right).\end{aligned} \tag{17}$$

- Otherwise, assume that $\|g_k\|^2 < |\min(0, \lambda_{\min}(H_k))|$. In this case we have $\lambda_{\min}(H_k) < -\|g_k\|^2 \leq 0$, and from (15) it follows that

$$\begin{aligned}\min\left(g_k^T d_k, d_k^T H_k d_k\right) &\leq \tfrac{1}{2}\left(g_k^T d_k + d_k^T H_k d_k\right) \leq \tfrac{1}{2}\gamma_H \lambda_{\min}(H_k) \\ &\leq \tfrac{1}{2}\gamma_H \min\left(-\|g_k\|^2, \lambda_{\min}(H_k)\right).\end{aligned} \tag{18}$$

The desired result follows from (17) and (18). □

We can now proceed to prove the satisfaction of conditions (8), (10), and thus the convergence of the algorithm.

Theorem 2 *Under assumptions A1 and A2, and conditions* (11)–(15)*, the sequence* $\{x_k\}$ *converges to second-order critical points of problem* (1).

Proof We start by proving the existence of a lower bound on the steplength ζ_k for a search direction d_k satisfying (11)–(15). We prove that all sufficiently small values of ζ are such that ζd_k satisfies condition (8). We define a function of a real variable $\Psi_k(\zeta)$ as

$$\Psi_k(\zeta) \equiv f(x_k + \zeta d_k) - f(x_k) - \sigma\left(\zeta g_k^T d_k + \zeta^2 \min\left(0, d_k^T H_k d_k\right)\right). \tag{19}$$

Function Ψ satisfies

$$\Psi_k(0) = 0, \quad \Psi_k'(0) = (1-\sigma) g_k^T d_k, \quad \Psi_k''(0) = d_k^T H_k d_k - 2\sigma \min\left(0, d_k^T H_k d_k\right).$$

Note that any values of ζ such that $\Psi_k(\zeta) < 0$ satisfy descent condition (8).

From (11) we have that $\Psi_k'(0) = (1-\sigma) g_k^T d_k \leq 0$ and from (16) in Lemma 1 we also have $\min(\Psi_k'(0), \Psi_k''(0)) < 0$ unless we are at a second-order critical point.

Also, note that Assumption A2 implies for any ζ such that $x_k + \zeta d_k \in \mathcal{S}_0$,

$$|\Psi_k''(\zeta) - \Psi_k''(0)| = |d_k^T(\nabla^2 f(x_k + \zeta d_k) - H_k) d_k| \leq K_f \zeta \|d_k\| \|d_k\|^2.$$

As a consequence, we have that

$$\Psi_k''(\zeta) \leq \Psi_k''(0) + K_f \zeta \|d_k\|^3. \tag{20}$$

- Consider first those iterations where $\|g_k\|^2 \geq |\min(0, \lambda_{\min}(H_k))|$. From the second-order Taylor series expansion for Ψ, $\Psi(0) = 0$ and (20) we have,

  $$\Psi_k(\zeta) \leq \Psi_k'(0)\zeta + \tfrac{1}{2}\Psi_k''(0)\zeta^2 + \tfrac{1}{2}K_f\zeta^3\|d_k\|^3.$$

  Using the bound (12) in this expression we have

  $$\Psi_k(\zeta) \leq \Psi_k'(0)\zeta + \tfrac{1}{2}\Psi_k''(0)\zeta^2 + \tfrac{1}{2}K_f\zeta^3\kappa_g^3\|g_k\|^3. \tag{21}$$

  From Assumptions A1 and A2, there exist constants K_H and K_g such that $\|H_k\| \leq K_H$ and $\|g_k\| \leq K_g$ for all iterations k. Using these bounds on $\Psi_k''(0)$ together with (12) we have

  $$|\Psi_k''(0)| = |d_k^T H_k d_k - 2\sigma \min\left(0, d_k^T H_k d_k\right)| \leq K_H\|d_k\|^2 \leq K_H\kappa_g^2\|g_k\|^2. \tag{22}$$

  Replacing this bound in (21) and using $\|g_k\|^3 \leq K_g\|g_k\|^2$, $\Psi_k'(0) = (1-\sigma)g_k^T d_k \leq 0$, together with (13) and (9), we obtain

  $$\begin{aligned}\Psi_k(\zeta) &\leq (1-\sigma)g_k^T d_k\zeta + \tfrac{1}{2}K_H\kappa_g^2\|g_k\|^2\zeta^2 + \tfrac{1}{2}K_f K_g\kappa_g^3\|g_k\|^2\zeta^3\\ &\leq -(1-\sigma)\gamma_g\|g_k\|^2\zeta + \tfrac{1}{2}K_H\kappa_g^2\|g_k\|^2\zeta^2 + \tfrac{1}{2}K_f K_g\kappa_g^3\|g_k\|^2\zeta^2\\ &\leq \zeta\|g_k\|^2\left(-(1-\sigma)\gamma_g + \tfrac{1}{2}\kappa_g^2\left(K_H + K_f K_g\kappa_g\right)\zeta\right).\end{aligned}$$

  As a consequence, letting $\tilde{\zeta} = (2(1-\sigma)\gamma_g)/(\kappa_g^2(K_H + K_f K_g\kappa_g))$, the values of ζ in $[0, \tilde{\zeta}]$ satisfy (8) and if we use a backtracking search the value of ζ found in iteration k, ζ_k, will satisfy $\zeta_k \geq \tilde{\zeta}/2$.
- Otherwise, for those iterations k satisfying $-\|g_k\|^2 > \min(0, \lambda_{\min}(H_k)) = \lambda_{\min}(H_k)$, using (14) in (20),

  $$\Psi_k''(\zeta) \leq \Psi_k''(0) + K_f\zeta\kappa_H^3|\lambda_{\min}(H_k)|^{3/2}.$$

  As in the preceding case, from Assumptions A1 and A2 there exists a constant K_H such that $\|H_k\| \leq K_H$ and $|\lambda_{\min}(H_k)| \leq K_H$ for all iterations k. Thus, the previous bound implies also

  $$\Psi_k''(\zeta) \leq \Psi_k''(0) + K_f K_H^{1/2}\kappa_H^3|\lambda_{\min}(H_k)|\zeta. \tag{23}$$

  For the first and second derivatives of Ψ, letting

  $$\omega_k = \begin{cases} 1-2\sigma & \text{if } d_k^T H_k d_k \leq 0\\ 1 & \text{otherwise,}\end{cases}$$

  and noting that $1 \geq \omega_k \geq 1-2\sigma$, we have $\Psi'(0) = (1-\sigma)g_k^T d_k \leq 0$ and $\Psi_k''(0) = \omega_k d_k^T H_k d_k$. Replacing these values in the second-order Taylor series expansion for Ψ around zero,

$$\Psi_k(\zeta) \leq (1-\sigma) g_k^T d_k \zeta + \tfrac{1}{2}\omega_k d_k^T H_k d_k \zeta^2 + \tfrac{1}{2} K_f \kappa_H^3 |\lambda_{\min}(H_k)| \zeta^3. \qquad (24)$$

If we use $g_k^T d_k \leq 0$ from (11), (9) and $1-\sigma \geq \omega_k/2$, as $\sigma \leq 1/2$ and $\omega_k \geq 1-2\sigma$, we have

$$(1-\sigma)\zeta g_k^T d_k \leq \tfrac{1}{2}\omega_k \zeta g_k^T d_k \leq \tfrac{1}{2}\omega_k \zeta^2 g_k^T d_k,$$

and making use of (15),

$$\begin{aligned}(1-\sigma) g_k^T d_k \zeta + \tfrac{1}{2}\omega_k d_k^T H_k d_k \zeta^2 &\leq \tfrac{1}{2}\omega_k \zeta^2 \left(g_k^T d_k + d_k^T H_k d_k\right)\\ &\leq \tfrac{1}{2}(1-2\sigma)\zeta^2 \left(g_k^T d_k + d_k^T H_k d_k\right)\\ &\leq -\tfrac{1}{2}(1-2\sigma)\zeta^2 \gamma_H |\lambda_{\min}(H_k)|.\end{aligned}$$

Replacing this bound, $g_k^T d_k \leq 0$ and (23) in (24) we obtain

$$\Psi_k(\zeta) \leq \tfrac{1}{2}\zeta^2 |\lambda_{\min}(H_k)| \left(-(1-2\sigma)\gamma_H + K_f K_H^{1/2} \kappa_H^3 \zeta\right).$$

As a consequence, if in this case we let $\tilde{\zeta} = ((1-2\sigma)\gamma_H)/(K_f K_H^{1/2} \kappa_H^3)$, the values of ζ in $[0, \tilde{\zeta}]$ satisfy (8) and if we use a backtracking search the value of ζ found in iteration k, ζ_k, will satisfy $\zeta_k \geq \tilde{\zeta}/2$.

From the preceding results, (8) and (9), letting

$$\bar{\zeta} = \tfrac{1}{2}\min\left(\frac{2(1-\sigma)\gamma_g}{\kappa_g^2(K_H + K_f K_g \kappa_g)}, \frac{(1-2\sigma)\gamma_H}{K_f K_H^{1/2} \kappa_H^3}\right),$$

for all k the values of ζ_k satisfy

$$\zeta_k \geq \hat{\zeta} \equiv \min(1, \bar{\zeta}) > 0. \qquad (25)$$

Note that d_k satisfies (16) and ζ_k satisfies (25). As a consequence of these inequalities, $\min\left(g_k^T d_k, \zeta_k d_k^T H_k d_k\right) \leq 0$, and

$$\min\left(g_k^T(\zeta_k d_k), (\zeta_k d_k)^T H_k (\zeta_k d_k)\right) \leq \hat{\zeta} \min\left(g_k^T d_k, \zeta_k d_k^T H_k d_k\right). \qquad (26)$$

Also, from (11) and (25) implying $\hat{\zeta} \leq 1$, if $d_k^T H_k d_k \geq 0$,

$$\min\left(g_k^T d_k, \zeta_k d_k^T H_k d_k\right) = g_k^T d_k = \min\left(g_k^T d_k, d_k^T H_k d_k\right) \leq \hat{\zeta} \min\left(g_k^T d_k, d_k^T H_k d_k\right), \qquad (27)$$

while if $d_k^T H_k d_k < 0$,

$$\begin{aligned}\min\left(g_k^T d_k, \zeta_k d_k^T H_k d_k\right) &\leq \min\left(g_k^T d_k, \hat{\zeta} d_k^T H_k d_k\right) \leq \min\left(\hat{\zeta} g_k^T d_k, \hat{\zeta} d_k^T H_k d_k\right)\\ &\leq \hat{\zeta}\min\left(g_k^T d_k, d_k^T H_k d_k\right). \end{aligned} \tag{28}$$

Putting together (26), (27), (28) and (16) we obtain

$$\min\left(g_k^T(\zeta_k d_k), (\zeta_k d_k)^T H_k(\zeta_k d_k)\right) \leq \hat{\gamma}\hat{\zeta}^2 \min\left(-\|g_k\|^2, \lambda_{\min}(H_k)\right).$$

As a consequence, Theorem 1 applies and the desired convergence result follows. □

The next step in the proofs will be to prove that direction d_k, computed by the proposed algorithms, satisfies conditions (11)–(15). This result depends on both the properties of the descent and negative curvature directions used in the combination that defines d_k, (3), and on conditions on the choices of the steplengths α_{ik} and $\bar{\alpha}_{ik}$.

The conditions we impose on the directions are independent of the combination algorithm used, either **NSM** or **TRSM**. They are also fairly standard in the literature for those algorithms that use a single search direction, and are presented below.

We assume that the *descent* directions computed in iteration k, d_{ik}, satisfy the following conditions:

$$g_k^T d_{ik} \leq -\beta_1\|g_k\|^2, \quad \|d_{ik}\| \leq \beta_2\|g_k\|, \tag{29}$$

for some positive constants β_1, β_2, and whenever $\|g_k\| > 0$ there is at least one d_{ik} different from zero.

The *negative curvature* directions in iteration k, $\bar{d}_{ik}$, if they exist, are assumed to satisfy

$$\bar{d}_{ik}^T H_k \bar{d}_{ik} \leq \beta_3 \min(0, \lambda_{\min}(H_k)), \quad \bar{d}_{ik}^T g_k \leq 0, \tag{30}$$

$$\|\bar{d}_{ik}\| \leq \beta_4 |\min(0, \lambda_{\min}(H_k))|^{1/2}, \tag{31}$$

where $\lambda_{\min}(A)$ denotes the smallest eigenvalue of A, and whenever $\lambda_{\min}(H_k) < 0$ at least one $\bar{d}_{ik}$ is different from zero. The values β_3 and β_4 denote positive constants. Again, these conditions are quite standard, except perhaps for (31), but some boundedness condition needs to be imposed, as negative curvature directions have no inherent scale.

If no directions are available in a given iteration satisfying these conditions, we must be at a second-order critical point. To simplify the arguments in the proofs we impose the condition that all the directions we consider in a given iteration k, d_{ik} and $\bar{d}_{ik}$, are linearly independent. Otherwise, we remove some of them until this condition is attained, while satisfying the preceding conditions. In particular, whenever we have descent we must include at least one sufficient descent direction, and whenever we have negative curvature, we must include at least one direction of negative curvature (although both directions may be the same one, if it satisfies the required conditions).

This requirement implies the number of directions used in any iteration cannot be larger than n, the space dimension. Including any additional linear combinations of the chosen directions need not provide any particular advantage as the directions will be combined before a linesearch is performed in each iteration. As a particular case, in dimension two and in the presence of both descent and negative curvature we would choose at most two linearly independent directions: a descent direction and a negative curvature direction, and we would discard the rest. If both directions were available, any direction of interest could be obtained as a combination of those two directions.

A second set of required conditions are related to the steplengths α_{ik} and $\bar{\alpha}_{ik}$. These conditions are specific to each algorithm and we present them in separate subsections, together with the proofs that conditions (11)–(15) hold for both proposed algorithms.

3.1 The subspace search approach

We consider first the convergence details of algorithm **NSM**. In each iteration this algorithm finds a combination d_k of the available directions, defined in (3) in terms of nonegative scalars α_{ik} and $\bar{\alpha}_{ik}$, using a Newton iteration on the scalars. In this subsection we introduce conditions on these scalars, taking into account that there are many possible ways to conduct a modified Newton iteration to obtain the desired values. We have looked for sufficient conditions that are simple to impose and as unrestrictive as possible regarding the Newton implementation to use. In particular, note that the combination does not need to satisfy sufficent descent conditions, as the posterior linesearch will take care of that. These conditions are described below.

1. The steplengths must be nonnegative and bounded above in the algorithm, that is, there exists a positive constant $\kappa_\alpha \geq 1$ such that for all k,

$$0 \leq \alpha_{ik} \leq \kappa_\alpha, \quad 0 \leq \bar{\alpha}_{ik} \leq \kappa_\alpha. \tag{32}$$

   The nonnegativity of the steplengths ensures that the resulting step d_k will be a descent direction, as from (29) and (30) each one of the directions d_{ik} and $\bar{d}_{ik}$ is a descent direction. The upper bound is a safeguard against unreasonable implementation choices, in particular a reasonable and common choice would be to choose $\kappa_\alpha = 1$. This condition can be enforced by projecting onto the corresponding box.
2. If in a given iteration k we have significant descent compared to the negative curvature available in that iteration, that is, if no negative curvature exists ($\lambda_{\min}(H_k) \geq 0$) or

$$\|g_k\|^2 \geq |\min(0, \lambda_{\min}(H_k))| \tag{33}$$

   holds, then the steplength of at least one of the descent directions must be bounded below, that is, there must exist a positive constant $\bar{\gamma}$ such that for all these iterations,

$$\max_i \alpha_{ik} \geq \bar{\gamma}. \tag{34}$$

This condition is similar to the boundedness condition in the classical framework, at least for the descent directions. Again, if it does not hold it is enough to modify the largest value of α_{ik} to make it equal to $\bar{\gamma}$.

3. We also need conditions on the interaction of the different directions, in particular the negative curvature directions. If we have negative curvature and $\|g_k\|^2 < |\lambda_{\min}(H_k)|$ holds in iteration k, implying that we have more negative curvature than descent, then we require

$$\lambda_{\max}\left(D_k^T H_k D_k\right) \le \delta_1 \max_i \bar{d}_{ik}^T H_k \bar{d}_{ik}, \tag{35}$$

where D_k denotes the matrix having as its columns the directions d_{ik} and $\bar{d}_{ik}$ having positive values for their steplengths, and

$$\max_i \bar{\alpha}_{ik} \ge \bar{\gamma}, \tag{36}$$

to hold for some positive values $\delta_1 < 1$ and $\bar{\gamma}$, independent of the iteration,
A strategy to satisfy (35) could be to fix values of α_{ik} and $\bar{\alpha}_{ik}$ to zero until it is satisfied. Note that if just one value of $\bar{\alpha}_{ik}$ is different from zero the condition is trivially satisfied. Regarding (36), if it is not satisfied it would be enough to fix the largest value of $\bar{\alpha}_{ik}$ to $\bar{\gamma}$, for example.

The next Lemma shows that under the preceding conditions (11)–(15) are satisfied.

Lemma 2 *Under Assumptions A1, A2 and conditions* (34), (36) *and* (35), *for all iterations k of algorithm* ***NSM*** *conditions* (11)–(15) *hold.*

Proof From the definition of d_k, (3), conditions (29), (30) and the nonnegativity of α_{ik} and $\bar{\alpha}_{ik}$, we have

$$g_k^T d_k = \sum_i \alpha_{ik} g_k^T d_{ik} + \sum_i \bar{\alpha}_{ik} g_k^T \bar{d}_{ik} \le 0.$$

Thus, (11) holds.

Consider the case when $\|g_k\|^2 \ge |\min(0, \lambda_{\min}(H_k))|$. Again from the definition of d, (3), (29), (31) and (32),

$$\begin{aligned}
\|d_k\| &\le \sum_i \alpha_{ik}\|d_{ik}\| + \sum_i \bar{\alpha}_{ik}\|\bar{d}_{ik}\| \\
&\le \sum_i \kappa_\alpha \beta_2 \|g_k\| + \sum_i \kappa_\alpha \beta_4 \min\left(\|g_k\|, |\min(0, \lambda_{\min}(H_k))|^{1/2}\right) \\
&\le n\kappa_\alpha \beta_2 \|g_k\| + n\kappa_\alpha \beta_4 \|g_k\| \\
&\le n\kappa_\alpha (\beta_2 + \beta_4)\|g_k\|,
\end{aligned}$$

implying that (12) holds.

Also, from (3), (29), (30) and (34),

$$g_k^T d_k = \sum_i \alpha_{ik} g_k^T d_{ik} + \sum_i \bar{\alpha}_{ik} g_k^T \bar{d}_{ik} \le -\bar{\gamma}\|g_k\|^2,$$

implying now that (13) holds.

Consider the second case, that is, iterations k where $\|g_k\|^2 < |\min(0, \lambda_{\min}(H_k))|$. Note that for these iterations it also holds that $\|g_k\| < |\lambda_{\min}(H_k)|^{1/2}$.

For these iterations, from (3), (29), (31) and the condition $\|g_k\| < |\lambda_{\min}(H_k)|^{1/2}$ we have

$$\begin{aligned}\|d_k\| &\le \sum_i \alpha_{ik}\|d_{ik}\| + \sum_i \bar{\alpha}_{ik}\|\bar{d}_{ik}\| \\ &\le \sum_i \kappa_\alpha \beta_2 \|g_k\| + \sum_i \kappa_\alpha \beta_4 |\min(0, \lambda_{\min}(H_k))|^{1/2} \\ &\le n\kappa_\alpha(\beta_2 + \beta_4)|\lambda_{\min}(H_k)|^{1/2},\end{aligned}$$

and (14) holds.

Finally, note that from (11) we have

$$g_k^T d_k + d_k^T H_k d_k \le d_k^T H_k d_k = a_k^T D_k^T H_k D_k a_k \le \lambda_{\max}\left(D_k^T H_k D_k\right)\|a_k\|^2,$$

where a_k Denotes the vector of steplengths $(\alpha_{ik}, \bar{\alpha}_{ik})$ and D_k denotes the matrix having as columns the search directions $(d_{ik}, \bar{d}_{ik})$. From (36) we have $\|a_k\| \ge \bar{\gamma}$, and from (35), (30) and the preceding bounds

$$g_k^T d_k + d_k^T H_k d_k \le \delta_1 \max_i \bar{d}_{ik}^T H_k \bar{d}_{ik} \bar{\gamma}^2 \le \delta_1 \bar{\gamma}^2 \beta_3 \min(0, \lambda_{\min}(H_k)),$$

implying that (15) holds. □

From Lemma 2 and Theorems 2 and 1, under the preceding conditions (32)–(36), we have shown that algorithm **NSM** converges to second-order critical points of problem (1).

3.2 Trust-region and linesearch approach

Consider now the case of Algorithm **TRSM**, combining a trust-region approach and a linesearch on the direction obtained from the trust region. In iteration k of this algorithm, the descent and negative curvature directions are combined by solving a trust-region problem of the form

$$\begin{array}{ll}\min_a & g_k^T D_k a + \frac{1}{2} a^T D_k^T H_k D_k a \\ \text{s.t.} & \|a\| \le \Delta_k,\end{array} \tag{37}$$

where D_k is a matrix having as columns the descent and negative curvature directions d_{ik} and $\bar{d}_{ik}$ and Δ_k is a positive scalar.

We impose the condition that the value of Δ_k satisfies for all k

$$\bar{\Delta} \leq \Delta_k \leq \tilde{\Delta}, \tag{38}$$

where $\bar{\Delta}$ and $\tilde{\Delta}$ are two prespecified constants. Note that in this algorithm, due to the use of a linesearch to compute the final size of d_k, it should be possible to select the value of Δ_k (and those of $\bar{\Delta}$ and $\tilde{\Delta}$) with greater freedom than in the case of a pure trust-region algorithm without impacting its convergence properties, as the linesearch should correct for any deviation in the scale of d_k.

This algorithm provides a much more precise definition for d_k, compared to Algorithm **NSM**. As a consequence, we do not need additional conditions such as (34)–(36), used in the convergence proof of **NSM**.

Let a_k denote the optimal value for a in (37), and define our search direction as $d_k = D_k a_k$. Then, as in the preceding case, a standard linesearch in carried out along this direction to find a scalar ζ_k satisfying condition (8) for $\zeta_k d_k$.

The following result shows that conditions (11)–(15) also hold for algorithm **TRSM**.

Lemma 3 *Under condition* (38)*, for all iterations* k*, conditions* (11)–(15) *hold for algorithm* ***TRSM****.*

Proof Consider the three different cases for the solution of (37):

Case I	$\lambda_{\min}(H_k) > 0$, $\\|a_k\\| < \Delta_k$ and $a_k = -(D_k^T H_k D_k)^{-1} D_k^T g_k$.
Case II	$\\|a_k\\| = \Delta_k$ and $a_k = -(D_k^T H_k D_k + \kappa I)^{-1} D_k^T g_k$ for an appropriate positive value $\kappa > -\min(0, \lambda_{\min}(D_k^T H_k D_k))$.
Case III	$D_k^T g_k$ is orthogonal to all eigenvectors associated with $\lambda_{\min}(D_k^T H_k D_k) < 0$. For this case $\\|a_k\\| = \Delta_k$ and $a_k = -(D_k^T H_k D_k - \lambda_{\min}(D_k^T H_k D_k) I)^{+} D_k^T g_k + \eta v_{\min}$, where η is an appropriate scalar, $v_{\min}$ is an eigenvector associated with $\lambda_{\min}(D_k^T H_k D_k)$ and A^{+} denotes the generalized inverse of A (a matrix with the same eigenvectors as A, and eigenvalues equal to the inverses of the eigenvalues of A if different from zero, or zero otherwise).

Let

$$\tau_k = \begin{cases} 0 & \text{if Case I holds,} \\ \kappa & \text{if Case II holds,} \\ -\lambda_{\min}\left(D_k^T H_k D_k\right) & \text{if Case III holds.} \end{cases}$$

From $d_k = D_k a_k$ and the fact that for Cases I and II $D_k^T H_k D_k + \tau_k I$ is a positive definite matrix, we have $a_k = -(D_k^T H_k D_k + \tau_k I)^{-1} D_k^T g_k$ for those cases and

$$\begin{aligned} d_k &= -D_k \left(D_k^T H_k D_k + \tau_k I\right)^{-1} D_k^T g_k \\ g_k^T d_k &= -g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{-1} D_k^T g_k \le -\frac{\left\| D_k^T g_k \right\|^2}{\lambda_{\max}\left(D_k^T H_k D_k\right) + \tau_k}. \end{aligned} \tag{39}$$

Note that

$$A^{-1} B A^{-1} - A^{-1} = A^{-1}(B - A) A^{-1}.$$

Letting $A = D_k^T H_k D_k + \tau_k I$ and $B = D_k^T H_k D_k$ and using the preceding results we have

$$\begin{aligned} g_k^T d_k + d_k^T H_k d_k &= -g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{-1} D_k^T g_k \\ &\quad + g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{-1} D_k^T H_k D_k \left(D_k^T H_k D_k + \tau_k I\right)^{-1} D_k^T g_k \\ &= -\tau_k g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{-2} D_k^T g_k \\ &= -\tau_k \| \left(D_k^T H_k D_k + \tau_k I\right)^{-1} D_k^T g_k \|^2 \\ &= -\tau_k \|a_k\|^2. \end{aligned} \tag{40}$$

For Case III we have

$$d_k = -D_k \left(D_k^T H_k D_k + \tau_k I\right)^{+} D_k^T g_k + \eta D_k v_{\min}. \tag{41}$$

In this case it holds that $g_k^T D_k v_{\min} = 0$, and together with (41) we have

$$g_k^T d_k = -g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{+} D_k^T g_k \le -\frac{\left\| D_k^T g_k \right\|^2}{\lambda_{\max}\left(D_k^T H_k D_k\right) + \tau_k}, \tag{42}$$

as before.

Combining these results, (11) follows from (39) and (42).

For this case we also have

$$\begin{aligned} g_k^T d_k + d_k^T H_k d_k &= -g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{+} D_k^T g_k \\ &\quad + g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{+} D_k^T H_k D_k \left(D_k^T H_k D_k + \tau_k I\right)^{+} D_k^T g_k \\ &\quad - 2\eta g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^{+} D_k^T H_k D_k v_{\min} \\ &\quad + \eta^2 v_{\min}^T D_k^T H_k D_k v_{\min}. \end{aligned} \tag{43}$$

To simplify this expression, let $D_k^T H_k D_k = V \Lambda V^T$, the spectral decomposition of the matrix $D_k^T H_k D_k$, where we have omitted the iteration index k. Also, let v_i denote the i-th eigenvector of $D_k^T H_k D_k$ (the i-th column of V), λ_i be the i-th eigenvalue of the same matrix, and $u_i = v_i^T D_k g_k$. We have that $(D_k^T H_k D_k + \tau_k I)^+ = V(\Lambda + \tau_k I)^+ V^T$, and

$$\begin{aligned}
&-g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^+ D_k^T g_k \\
&\quad + g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^+ D_k^T H_k D_k \left(D_k^T H_k D_k + \tau_k I\right)^+ D_k^T g_k \\
&= -g_k^T D_k V(\Lambda + \tau_k I)^+ V^T D_k^T g_k + g_k^T D_k V(\Lambda + \tau_k I)^+ \Lambda (\Lambda + \tau_k I)^+ V^T D_k^T g_k \\
&= - \sum_{i:(\lambda_i + \tau_k) \neq 0} \frac{1}{\lambda_i + \tau_k} u_i^2 + \sum_{i:(\lambda_i + \tau_k) \neq 0} \frac{\lambda_i}{(\lambda_i + \tau_k)^2} u_i^2 = \sum_{i:(\lambda_i + \tau_k) \neq 0} \frac{-\tau_k}{(\lambda_i + \tau_k)^2} u_i^2 \\
&= -\tau_k g_k^T D_k \left(\left(D_k^T H_k D_k + \tau_k I\right)^+\right)^2 D_k^T g_k. \qquad (44)
\end{aligned}$$

Also, from the preceding definitions of $v_{\min}$ and τ_k,

$$D_k^T H_k D_k v_{\min} = -\tau_k v_{\min}, \quad \left(D_k^T H_k D_k + \tau_k I\right)^+ v_{\min} = 0, \qquad (45)$$

implying

$$\begin{aligned}
&-2\eta g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^+ D_k^T H_k D_k v_{\min} + \eta^2 v_{\min}^T D_k^T H_k D_k v_{\min} \\
&= -\tau_k \eta^2 v_{\min}^T v_{\min}. \qquad (46)
\end{aligned}$$

As a consequence, replacing (44) and (46) in (43) we have

$$g_k^T d_k + d_k^T H_k d_k = -\tau_k g_k^T D_k \left(\left(D_k^T H_k D_k + \tau_k I\right)^+\right)^2 D_k^T g_k - \tau_k \eta^2 v_{\min}^T v_{\min},$$

and using again (45),

$$\begin{aligned}
g_k^T d_k + d_k^T H_k d_k &= -\tau_k g_k^T D_k \left(\left(D_k^T H_k D_k + \tau_k I\right)^+\right)^2 D_k^T g_k \\
&\qquad + 2\tau_k \eta g_k^T D_k \left(D_k^T H_k D_k + \tau_k I\right)^+ v_{\min} - \tau_k \eta^2 v_{\min}^T v_{\min} \\
&= -\tau_k \left\| -\left(D_k^T H_k D_k + \tau_k I\right)^+ D_k^T g_k + \eta v_{\min} \right\|^2 = -\tau_k \|a_k\|^2
\end{aligned} \qquad (47)$$

From (40) and (47) we have

$$g_k^T d_k + d_k^T H_k d_k = -\tau_k \|a_k\|^2 \leq 0, \qquad (48)$$

where a_k denotes the solution of (37) and τ_k is a value satisfying $\tau_k \geq -\min(0, \lambda_{\min}(D_k^T H_k D_k))$.

We now consider the satisfaction of bounds (12) and (13) under the assumption that

$$\|g_k\|^2 \geq |\min(0, \lambda_{\min}(H_k))| \tag{49}$$

holds. From (29) and our condition that at least one descent direction must be selected in this case,

$$\exists i \quad d_{ik}^T g_k \leq -\beta_1 \|g_k\|^2 \Rightarrow \|D_k g_k\| \geq \beta_1 \|g_k\|^2. \tag{50}$$

Also from (49), (29) and (31),

$$\begin{aligned}
\|D_k\|^2 \leq \|D_k\|_F^2 &= \sum_i \|d_{ki}\|^2 + \sum_j \|\bar{d}_{kj}\|^2 \\
&\leq \sum_i \beta_2^2 \|g_k\|^2 + \sum_j \beta_4^2 \min(\|g_k\|^2, |\min(0, \lambda_{\min}(H_k))|) \\
&\leq n\beta_2^2 \|g_k\|^2 + n\beta_4^2 \|g_k\|^2 = n\left(\beta_2^2 + \beta_4^2\right)\|g_k\|^2,
\end{aligned} \tag{51}$$

where n, the space dimension, is a bound on the maximum number of directions used by the algorithm in a given iteration and $\|\cdot\|_F$ denotes the Frobenius norm. From this bound and $\|a_k\| \leq \Delta_k \leq \tilde{\Delta}_k$ we have

$$\|d_k\|^2 = \|D_k a_k\|^2 \leq \|D_k\|^2 \|a_k\|^2 \leq \tilde{\Delta}^2 n \left(\beta_2^2 + \beta_4^2\right)\|g_k\|^2,$$

establishing (12).

As in the preceding proofs, let K_H denote a positive constant such that $\|H_k\| \leq K_H$ for all k; its existence follows from Assumptions *A1* and *A2*. Using the already established bound (11), (50) and (51), we have

$$\begin{aligned}
g_k^T d_k &\leq -\frac{\left\|D_k^T g_k\right\|^2}{\lambda_{\max}\left(D_k^T H_k D_k\right) + \tau_k} \leq -\frac{\beta_1^2 \|g_k\|^4}{K_H \|D_k\|^2 + \tau_k} \\
&\leq -\frac{\beta_1^2}{n\left(\beta_2^2 + \beta_4^2\right) K_H + \tau_k / \|g_k\|^2} \|g_k\|^2.
\end{aligned} \tag{52}$$

To refine this bound further and to obtain (13), for a given iteration k we need to consider the three cases introduced in the first part of the proof.

- Case I: it holds that $\tau_k = 0$. Thus from (52)

$$g_k^T d_k \leq -\frac{\beta_1^2}{n\left(\beta_2^2 + \beta_4^2\right) K_H} \|g_k\|^2. \tag{53}$$

- Case II: from $\|a_k\| = \Delta_k, a_k = -(D_k^T H_k D_k + \tau_k I)^{-1} D_k^T g_k, \|H_k\| \le K_H$ and (51) we have

$$\begin{aligned}
\Delta_k &= \left\| \left(D_k^T H_k D_k + \tau_k I \right)^{-1} D_k^T g_k \right\| \le \frac{\| D_k^T g_k \|}{\lambda_{\min} \left(D_k^T H_k D_k \right) + \tau_k} \\
\Rightarrow \tau_k &\le \frac{\| D_k^T g_k \|}{\Delta_k} - \lambda_{\min} \left(D_k^T H_k D_k \right) \le \frac{\|D_k\| \|g_k\|}{\bar{\Delta}_k} + K_H \|D_k\|^2 \\
&\le \frac{\sqrt{n \left(\beta_2^2 + \beta_4^2\right)}}{\bar{\Delta}_k} \|g_k\|^2 + K_H n \left(\beta_2^2 + \beta_4^2 \right) \|g_k\|^2 \\
\Rightarrow \frac{\tau_k}{\|g_k\|^2} &\le \frac{\sqrt{n \left(\beta_2^2 + \beta_4^2\right)}}{\bar{\Delta}_k} + n K_H \left(\beta_2^2 + \beta_4^2 \right).
\end{aligned}$$

Replacing this bound in (52) we obtain

$$g_k^T d_k \le - \frac{\beta_1^2}{2n \left(\beta_2^2 + \beta_4^2\right) K_H + \sqrt{n \left(\beta_2^2 + \beta_4^2\right)} / \bar{\Delta}_k} \|g_k\|^2. \tag{54}$$

- Case III: we have $\tau_k = -\lambda_{\min}(H_k)$, and using (51) we have that

$$\begin{aligned}
\frac{\tau_k}{\|g_k\|^2} &= \frac{-\lambda_{\min} \left(D_k^T H_k D_k \right)}{\|g_k\|^2} \le \frac{\|H_k\| \|D_k\|^2}{\|g_k\|^2} \\
&\le \frac{K_H n \left(\beta_2^2 + \beta_4^2\right) \|g_k\|^2}{\|g_k\|^2} \le n K_H \left(\beta_2^2 + \beta_4^2 \right),
\end{aligned}$$

and replacing this bound in (52),

$$g_k^T d_k \le - \frac{\beta_1^2}{2n \left(\beta_2^2 + \beta_4^2\right) K_H} \|g_k\|^2. \tag{55}$$

The bound in (13) is a consequence of (53), (54) and (55).

For the last part of the proof, consider iterations where we have $\|g_k\|^2 < -\min(0, \lambda_{\min}(H_k))$. Note that in this case $\lambda_{\min}(H_k) < 0$. From these conditions, (29) and (31), and $\|a_k\| \le \Delta_k \le \tilde{\Delta}$ we have

$$\begin{aligned}
\|d_k\|^2 = \|D_k a_k\|^2 \le \|D_k\|^2 \|a_k\|^2 \le \tilde{\Delta}^2 \|D_k\|_F^2 &= \tilde{\Delta}^2 \left(\textstyle\sum_i \|d_{ki}\|^2 + \sum_j \|\bar{d}_{kj}\|^2 \right) \\
&\le \tilde{\Delta}^2 \left(\textstyle\sum_i \beta_2^2 \|g_k\|^2 + \sum_j \beta_4^2 |\lambda_{\min}(H_k)| \right) \\
&\le \tilde{\Delta}^2 \left(\textstyle\sum_i \beta_2^2 |\lambda_{\min}(H_k)| + \sum_j \beta_4^2 |\lambda_{\min}(H_k)| \right) \\
&\le \tilde{\Delta}^2 n \left(\beta_2^2 + \beta_4^2 \right) |\lambda_{\min}(H_k)|.
\end{aligned}$$

This result implies (14).

To prove (15), note that for those iterations satisfying $\|g_k\|^2 < -\min(0, \lambda_{\min}(H_k))$ we must have that either case II or III holds, implying that $\|a_k\| = \Delta_k \geq \bar{\Delta}$. Also, $\tau_k \geq -\min(0, \lambda_{\min}(D_k^T H_k D_k))$. As a consequence,

$$-\tau_k \|a_k\|^2 \leq \min\left(0, \lambda_{\min}\left(D_k^T H_k D_k\right)\right) \bar{\Delta}^2.$$

Furthermore, as the algorithm requires the use of a direction of negative curvature (at least) in that iteration, it must hold that some diagonal element of $D_k^T H_k D_k$ satisfies $\bar{d}_{ik}^T H_k \bar{d}_{ik} \leq \beta_3 \lambda_{\min}(H_k)$ from (30), implying $\lambda_{\min}(D_k^T H_k D_k)) \leq \beta_3 \lambda_{\min}(H_k) < 0$. Replacing these bounds in (48) we obtain

$$g_k^T d_k + d_k^T H_k d_k = -\tau_k \|a_k\|^2 \leq \bar{\Delta}^2 \beta_3 \lambda_{\min}(H_k),$$

that implies (15). □

Lemma 3 and Theorems 2 and 1, under condition (38), imply the convergence of algorithm **TRSM** to second-order critical points of problem (1).

In the following section we present a particular implementation of this approach that selects $d_{1k} = -H_k^{-1} g_k$, the Newton direction, in all iterations where it is defined. It also chooses $(a_k)_1 = 1$ and computes the remaining components of a_k solving the equivalent version of the trust-region problem (37). In this particular case, letting $a = \left(1 \ \bar{a}^T\right)^T$ and $D_k = \left(d_{1k} \ \bar{D}_k\right)$, where $\bar{D}_k$ denotes a matrix having as columns all search directions, except for the Newton direction, and $\bar{a}$ denotes the corresponding coefficients, this subproblem has the form

$$\begin{array}{ll} \min_{\bar{a}} & (g_k + H_k d_{1k})^T \bar{D}_k \bar{a} + \frac{1}{2} \bar{a}^T \bar{D}_k^T H_k \bar{D}_k \bar{a} \\ \text{s.t.} & \|\bar{a}\| \leq \bar{\Delta}_k, \end{array}$$

but as $g_k + H_k d_{1k} = 0$, we obtain the equivalent problem

$$\begin{array}{ll} \min_{\bar{a}} & \frac{1}{2} \bar{a}^T \bar{D}_k^T H_k \bar{D}_k \bar{a} \\ \text{s.t.} & \|\bar{a}\| \leq \bar{\Delta}_k. \end{array} \tag{56}$$

A solution of this problem is

- $\bar{a}_k = 0$ if $\lambda_{\min}(\bar{D}_k^T H_k \bar{D}_k) > 0$, or
- $\|\bar{a}_k\| = \bar{\Delta}_k$ and

$$\bar{D}_k^T H_k \bar{D}_k \bar{a}_k = \lambda_{\min}\left(\bar{D}_k^T H_k \bar{D}_k\right) \bar{a}_k,$$

  otherwise, that is, an eigenvector associated with the smallest eigenvalue of $\lambda_{\min}(\bar{D}_k^T H_k \bar{D}_k)$.

And as a consequence, the search direction for the linesearch part of the algorithm is given by $d_k = d_{1k} + \bar{D}_k \bar{a}_k = d_{1k} + \bar{d}_k$, where $\bar{d}_k$ is either equal to zero if the matrix H_k is positive definite, or a direction of negative curvature satisfying (30) and (31) otherwise, and with a size that is bounded below. Thus, the preceding conditions introduced for the combination of directions also hold in this case and the algorithm also converges to second-order critical points.

4 Implementation and numerical results

4.1 Implementation of the algorithms

The algorithms we have chosen to implement and test in this paper are particular versions of those described in Sect. 2, algorithms **NSM** and **TRSM**. In the preceding section we have studied the convergence properties of the more general versions of the algorithms, but these general versions must be specified in greater detail before they can be implemented. In the following paragraphs we describe the specific implementations we have used in our numerical experiments.

- Search directions. We compute just two descent directions, a Newton direction d_{1k} and the negative gradient d_{2k},

  $$d_{1k} = -B^{-1} \nabla f(x_k), \quad d_{2k} = -\nabla f(x_k),$$

  where $B = \nabla^2 f(x_k)$, if $\nabla^2 f(x_k)$ is positive definite, otherwise, B is a suitable approximation for $\nabla^2 f(x_k)$ to ensure that condition (29) is satisfied.
  We also use a single negative curvature direction, that we denote as $\bar{d}_{3k}$, whenever it is available at iteration k and satisfies conditions (30) and (31).
- Nonnegative steplengths. We introduce an additional step in algorithm **NSM** to ensure that the steplengths remain positive and bounded in all iterations. For a given positive constant κ_α we compute:

  Step 3c' **[Projection]** *Let* $\tilde{\alpha}_{2(\hat{k}+1)}$ *and* $\tilde{\alpha}_{3(\hat{k}+1)}$ *be the values obtained from Step 3c. Set* $\alpha_{2(\hat{k}+1)} = \min(\max(0, \tilde{\alpha}_{2(\hat{k}+1)}), \kappa_\alpha)$ *and* $\bar{\alpha}_{3(\hat{k}+1)} = \min(\max(0, \tilde{\alpha}_{3(\hat{k}+1)}), \kappa_\alpha)$.

  After this projection step, the resulting combination of directions d_k is still a descent direction, as it is a nonnegative linear combination of descent directions.
  From a theoretical point of view, this step ensures the satisfaction of condition (32) for the gradient and negative curvature direction.
- Unit Newton step. For both algorithms the value for the search direction in iteration k, d_k, has been obtained by fixing $\alpha_{1\hat{k}} = 1$, that is, the step taken along the Newton direction is set to 1 before the linesearch step (Step 3e) is carried out.
  If in addition to this, the initial values for the other steplengths, α_{20} and α_{30}, are taken to be equal to zero, close to the solution we may take Newton steps, and we may attain a quadratic rate of convergence.

Note that this choice of step for the Newton direction does not affect the convergence proofs. For algorithm **NSM** it ensures the satisfaction of condition (34). Regarding algorithm **TRSM**, the convergence for this case has been already discussed at the end of Sect. 3.

4.2 Computational experiments

We have carried out some computational experiments to compare the two proposals described in the preceding sections with an implementation of a modified linesearch Newton's method (**LSNM**), that is, an implementation of Newton's method including a modification of the Hessian matrix to compute the descent direction in those iterations where significant negative curvature is detected, as well as an implementation of a more sophisticated Newton's method using the Moré and Sorensen (15) approach (**MSNM**) that incorporates negative curvature through a quadratic curvilinear search.

Regarding the computation of the Newton direction and the negative curvature direction, an efficient method to compute both directions is the modified Cholesky factorization proposed by Gill and Murray (7). This procedure allows the computation of a negative curvature direction using the same factorization used for the determination of the modified Newton direction; in this manner, the computational cost to obtain the direction of negative curvature is almost null. An alternative factorization which may be used is the one proposed by Bunch and Parlett (1). In our computational experiments we have obtained similar results for both factorizations. In this section, we show the results obtained using the modified Cholesky factorization.

Numerical results have been obtained for a number of test problems from the CUTEr collection (11). We have included the 119 nonlinear unconstrained problems of dimension between 1 and 500, having continuous second derivatives, to ensure that the directions of interest were available for all of them.

The algorithms and the test problems have been implemented and executed using MATLAB 6.5 for Linux. In all cases we have considered the default starting points x_0 provided by CUTEr.

The algorithms introduced in Sect. 2 use some parameters. For our implementation, we have chosen the values $\sigma = 10^{-9}$ ($\sigma \in (0, 1/2)$), $\xi = 0.5$, $\delta = \Delta_0 = \widehat{k}_{max} = 1$; the convergence tolerance ω_* specifies the final accuracy requested by the user, and it has been set to the value 10^{-8}. The maximum number of iterations allowed was set to 1,000.

4.2.1 Analysis of the results

Tables 5 and 6 present the numerical results obtained for the different algorithms. In these tables we have used the following notation:

- n: Number of variables.
- iter: Iteration count.
- fgeval: Number of function and gradient evaluations.
- LSNM: Line search Newton method
- MSNM: Moré-Sorensen Newton method.

Table 5 Overall comparison of iteration and function evaluation counts

Problem	n	Iterations				Function evaluations			
		LSNM	MSNM	NSM	TRSM	LSNM	MSNM	NSM	TRSM
AKIVA	2	6	6	6	6	7	7	7	7
ALLINITU	4	6	9	7	6	9	12	21	10
ARGLINA	200	1	1	1	1	2	2	2	2
ARWHEAD	500	6	6	6	6	7	7	7	7
BARD	3	9	12	9	9	10	19	10	10
BDQRTIC	500	9	9	9	9	10	10	10	10
BEALE	2	6	6	6	6	39	23	41	41
BIGGS6	6	39	71	27	24	67	113	73	79
BOX3	3	8	8	8	8	9	9	9	9
BRKMCC	2	3	3	3	3	4	4	4	4
BROWNAL	200	15	32	26	13	148	522	398	111
BROWNBS	2	6	8	6	6	73	43	77	77
BROWNDEN	4	8	8	8	8	9	9	9	9
BROYDN7D	500	82	84	84	82	5,208	5,216	5,386	5,329
BRYBND	500	10	11	10	10	15	16	19	19
CHNROSNB	50	41	62	41	41	60	120	77	77
CLIFF	2	27	27	27	27	28	28	28	28
CRAGGLVY	500	14	14	14	14	15	15	15	15
CUBE	2	26	27	17	20	37	35	104	58
DECONVU	61	>1,000	179	>1,000	>1,000	–	1,757	–	–
DENSCHNA	2	6	6	6	6	7	7	7	7
DENSCHNB	2	6	6	6	6	35	21	37	37
DENSCHNC	2	10	10	10	10	11	11	11	11
DENSCHND	3	45	48	45	45	76	69	75	73
DENSCHNE	3	11	9	11	11	18	10	20	20
DENSCHNF	2	6	6	6	6	7	7	7	7
DIXMAANA	300	5	309	6	6	22	3,768	19	19
DIXMAANB	300	114	256	78	23	2,045	3,299	2,052	224
DIXMAANC	300	151	288	51	11	2,526	4,230	1,188	120
DIXMAAND	300	167	380	89	25	3,501	4,999	2,548	355
DIXMAANE	300	33	264	6	18	235	2,088	34	197
DIXMAANF	300	201	381	168	22	4,569	5,471	4,642	237
DIXMAANG	300	205	488	135	45	4,234	7,752	3,687	598
DIXMAANH	300	236	566	44	27	4,777	7,427	646	336
DIXMAANI	300	34	277	17	15	231	2,447	151	133
DIXMAANJ	300	245	404	154	273	4,347	6,740	3,566	9,332
DIXMAANK	300	253	598	203	179	4,263	8,270	5,697	4,304
DIXMAANL	300	285	603	107	70	5,519	9,716	1,672	1,383
DIXON3DQ	100	1	1	1	1	2	2	2	2

Table 5 continued

Problem	*n*	Iterations				Function evaluations			
		LSNM	MSNM	NSM	TRSM	LSNM	MSNM	NSM	TRSM
DJTL	2	>1,000	>1,000	>1,000	>1,000	–	–	–	–
DQDRTIC	500	1	1	1	1	2	2	2	2
DQRTIC	500	27	27	27	27	28	28	28	28
EDENSCH	36	12	12	12	12	13	13	13	13
ENGVAL2	3	15	14	15	15	21	17	20	20
ERRINROS	50	32	35	32	28	85	113	127	112
EXPFIT	2	5	9	5	5	8	14	10	10
FLETCHCR	100	157	298	148	150	184	585	259	271
FMINSURF	121	30	15	35	30	266	59	413	314
FREUROTH	500	37	42	8	8	4,646	5,426	595	595
GENROSE	500	624	>1,000	>1,000	>1,000	41,661	–	–	–
GROWTHLS	3	19	26	28	15	36	44	79	44
GULF	3	22	20	23	24	49	61	69	95
HAIRY	2	28	25	41	16	277	255	198	143
HATFLDD	3	17	23	17	20	21	31	23	30
HATFLDE	3	17	25	17	22	23	44	27	72
HEART6LS	6	499	399	703	434	1,693	1,480	3,646	2,369
HEART8LS	8	304	211	171	203	1,034	818	992	1,108
HELIX	3	15	12	15	15	23	25	31	31
HIELOW	3	5	6	4	4	7	10	8	8
HILBERTA	2	1	1	1	1	2	2	2	2
HILBERTB	10	1	1	1	1	2	2	2	2
HIMMELBB	2	11	9	12	13	13	11	16	17
HIMMELBF	4	>1,000	>1,000	>1,000	>1,000	–	–	–	–
HIMMELBG	2	5	6	5	5	8	9	14	14
HIMMELBH	2	4	4	4	4	33	19	35	35
HUMPS	2	75	160	82	144	419	1,307	484	813
JENSMP	2	9	9	9	9	10	10	10	10
KOWOSB	4	9	10	6	12	14	15	17	40
LIARWHD	500	11	11	11	11	12	12	12	12
LOGHAIRY	2	37	103	76	61	201	514	375	311
MANCINO	100	5	7	5	5	6	8	6	6
MARATOSB	2	671	734	362	220	993	1,074	2,495	1,746
MEXHAT	2	27	27	21	23	31	32	28	28
MEYER3	3	131	166	108	112	190	239	260	248
MOREBV	500	1	1	1	1	2	2	2	2
MSQRTALS	100	>1,000	>1,000	385	642	–	–	5,437	10,023
MSQRTBLS	100	>1,000	>1,000	>1,000	443	–	–	–	6,444
NONDIA	500	6	6	6	6	7	7	7	7
NONDQUAR	500	21	21	21	21	22	22	22	22

Table 5 continued

Problem	*n*	Iterations				Function evaluations			
		LSNM	MSNM	NSM	TRSM	LSNM	MSNM	NSM	TRSM
OSBORNEA	5	17	6	19	19	24	12	41	41
OSBORNEB	11	21	>1,000	16	16	69	–	58	58
PALMER1C	8	1	1	1	1	2	2	2	2
PALMER1D	7	1	1	1	1	2	2	2	2
PALMER2C	8	1	1	1	1	2	2	2	2
PALMER3C	8	1	1	1	1	2	2	2	2
PALMER4C	8	1	1	1	1	2	2	2	2
PALMER5C	6	1	1	1	1	2	2	2	2
PALMER6C	8	1	1	1	1	2	2	2	2
PALMER7C	8	1	1	1	1	2	2	2	2
PALMER8C	8	1	1	1	1	2	2	2	2
PENALTY1	500	40	40	35	35	44	43	40	40
PENALTY2	100	18	18	18	18	19	19	19	19
PENALTY3	100	287	13	184	205	427	18	736	1,141
POWELLSG	500	18	18	18	18	19	19	19	19
POWER	500	34	34	34	34	35	35	35	35
QUARTC	500	27	27	27	27	28	28	28	28
ROSENBR	2	21	22	18	18	29	29	30	30
S308	2	9	9	8	8	11	11	12	12
SCHMVETT	500	3	3	3	3	4	4	4	4
SINEVAL	2	41	44	27	29	65	63	104	86
SINQUAD	500	17	9	17	17	38	20	42	42
SISSER	2	18	18	18	18	19	19	19	19
SNAIL	2	62	105	42	48	112	219	116	215
SPMSRTLS	499	>1,000	>1,000	141	>1,000	–	–	3,525	–
SROSENBR	500	8	8	6	6	10	10	9	9
STRATEC	10	16	15	18	15	24	28	50	38
TOINTGOR	50	6	6	6	6	7	7	7	7
TOINTGSS	500	1	1	1	1	2	2	2	2
TOINTPSP	50	17	14	17	17	40	26	62	62
TOINTQOR	50	1	1	1	1	2	2	2	2
TQUARTIC	500	1	1	1	1	2	2	2	2
TRIDIA	500	1	1	1	1	2	2	2	2
VARDIM	200	28	28	28	28	29	29	29	29
VAREIGVL	50	29	26	29	29	34	47	38	38
VIBRBEAM	8	13	22	20	25	35	56	81	129
WATSON	12	23	>1,000	27	26	79	–	148	121
WOODS	100	39	76	32	33	62	190	66	60
YFITU	3	35	33	28	31	49	47	53	63
ZANGWIL2	2	1	1	1	1	2	2	2	2

Table 6 Overall comparison of the number of iterations in which negative curvature is detected

Problem	MSNM	NSM	TRSM
AKIVA	0	0	0
ALLINITU	2	2	2
ARGLINA	0	0	0
ARWHEAD	0	0	0
BARD	3	3	3
BDQRTIC	0	0	0
BEALE	0	0	0
BIGGS6	14	12	11
BOX3	1	1	1
BRKMCC	0	0	0
BROWNAL	25	21	7
BROWNBS	0	0	0
BROWNDEN	0	0	0
BROYDN7D	74	75	72
BRYBND	4	5	5
CHNROSNB	1	1	1
CLIFF	0	0	0
CRAGGLVY	0	0	0
CUBE	0	0	1
DECONVU	179	–	–
DENSCHNA	0	0	0
DENSCHNB	0	0	0
DENSCHNC	0	0	0
DENSCHND	5	4	6
DENSCHNE	3	8	8
DENSCHNF	0	0	0
DIXMAANA	301	3	3
DIXMAANB	250	74	20
DIXMAANC	279	46	7
DIXMAAND	374	85	21
DIXMAANE	256	3	10
DIXMAANF	376	163	18
DIXMAANG	478	130	40
DIXMAANH	562	40	23
DIXMAANI	267	12	11
DIXMAANJ	400	150	272
DIXMAANK	591	199	175
DIXMAANL	600	103	68
DIXON3DQ	0	0	0
DJTL	–	–	–
DQDRTIC	0	0	0
DQRTIC	0	0	0

Table 6 continued

Problem	MSNM	NSM	TRSM
EDENSCH	0	0	0
ENGVAL2	1	1	1
ERRINROS	13	11	12
EXPFIT	1	1	1
FLETCHCR	0	0	0
FMINSURF	1	11	8
FREUROTH	34	2	2
GENROSE	–	–	–
GROWTHLS	5	7	5
GULF	12	6	9
HAIRY	14	27	8
HATFLDD	1	2	2
HATFLDE	5	1	6
HEART6LS	386	696	425
HEART8LS	201	163	197
HELIX	5	4	4
HIELOW	1	1	1
HILBERTA	0	0	0
HILBERTB	0	0	0
HIMMELBB	8	10	11
HIMMELBF	–	–	–
HIMMELBG	1	1	1
HIMMELBH	0	0	0
HUMPS	113	65	121
JENSMP	0	0	0
KOWOSB	3	1	5
LIARWHD	0	0	0
LOGHAIRY	80	61	51
MANCINO	2	1	1
MARATOSB	1	59	23
MEXHAT	0	0	0
MEYER3	4	5	8
MOREBV	0	0	0
MSQRTALS	–	374	632
MSQRTBLS	–	–	430
NONDIA	0	0	0
NONDQUAR	0	0	0
OSBORNEA	4	3	3
OSBORNEB	–	3	3
PALMER1C	0	0	0
PALMER1D	0	0	0
PALMER2C	0	0	0

Table 6 continued

Problem	MSNM	NSM	TRSM
PALMER3C	0	0	0
PALMER4C	0	0	0
PALMER5C	0	0	0
PALMER6C	0	0	0
PALMER7C	0	0	0
PALMER8C	0	0	0
PENALTY1	0	0	0
PENALTY2	0	0	0
PENALTY3	4	19	46
POWELLSG	0	0	0
POWER	0	0	0
QUARTC	0	0	0
ROSENBR	0	0	0
S308	0	0	0
SCHMVETT	0	0	0
SINEVAL	0	0	0
SINQUAD	2	10	10
SISSER	0	0	0
SNAIL	3	3	1
SPMSRTLS	–	129	–
SROSENBR	0	0	0
STRATEC	5	5	4
TOINTGOR	0	0	0
TOINTGSS	0	0	0
TOINTPSP	0	0	0
TOINTQOR	0	0	0
TQUARTIC	0	0	0
TRIDIA	0	0	0
VARDIM	0	0	0
VAREIGVL	16	23	23
VIBRBEAM	10	10	19
WATSON	–	27	26
WOODS	25	4	3
YFITU	3	3	5
ZANGWIL2	0	0	0

- NSM: Newton-based Scaling method.
- TRSM: Trust-Region Scaling method.

If the maximum number of iterations is reached (iter $>$1,000), we do not consider the associated number of function and gradient evaluations in Table 5.

Table 7 provides a summary of the results. It is clear that the best results are obtained by the two proposals presented in this work. Both the average iteration counts and the

Table 7 Average numbers of iterations and function evaluations over the test problems, excluding those problems where an algorithm may have failed

	LSNM	MSNM	NSM	TRSM
Iter	49.27	76.25	37.79	31.61
Feval	488.68	780.55	403.21	309.74

average number of function evaluations are significantly lower for our proposals, compared to the two alternatives we have considered. In particular, the average reductions in the number of iterations obtained using the **NSM** and the **TRSM** methods with respect to the **LSNM** method are, respectively 23.30% and 35.84%. Regarding the reductions for the number of function evaluations the results are 17.49% and 36.62%, respectively.

With respect to the **LSNM** method, the reductions in iterations and function evaluations are far more marked than the increases. For the **NSM** method, the largest deterioration in the number of iterations amounted to 51.32% (problem LOGHAIRY), while the largest improvement was 81.82% (problem DIXMAANE). Regarding the **TRSM** algorithm, the largest deterioration in the number of iterations amounted to 48% (problem VIBRBEAM), while the largest improvement was 92.72% (problem DIXMAANC). Similar results are obtained when comparing the number of function evaluations.

Notice the large number of iterations and function evaluations for the **MSNM** algorithm when solving all the DIXMAAN problems, and problems BIGGS6, HUMPS or SNAIL. For instance, in Table 6, for the DIXMAAN problems, the number of iterations in which negative curvature is used is clearly larger when the **MSNM** method is used. In these cases the scaling of the descent directions seems to be specially relevant. For these problems our two proposals reduce drastically the number of iterations and moderately the number of function evaluations. It is important to remark that, excluding these problems, the **MSNM** algorithm provides similar results to those obtained using the **LSNM** algorithm. These problems are an example of the importance of an adequate scaling of the directions.

Regarding the impact of the use of negative curvature on the whole set of 119 problems, for those problems where negative curvature were detected, Table 6 shows the number of iterations where it was used. Negative curvature was detected for approximately 50% percent of the problems. For the **NSM** method, the average number of iterations per problem where a direction of negative curvature was used is 24.25, that is, 61.07% of the average number of iterations per problem. For the **TRSM** method this percentage is similar, 62.81%. For the **MSNM** method this percentage increases up to 70.14%, again because of the DIXMAAN problems. If we restrict the results only to those problems where negative curvature was detected, the average reductions in the number of iterations obtained using the **NSM** and the **TRSM** methods with respect to the **LSNM** method are, respectively 26.01% and 40.52%. Regarding the reductions for the number of function evaluations the results are 18.14% and 37.60%, respectively. Therefore, it is apparent the effect of the use of negative curvature in the successful performance of the methods.

Table 8 Approximate values for the probabilities $\rho_s^j(\tau)$, $j = 1, 2$, when $\tau = 1$, $\tau = 2$ and $\tau = 3$

	LSNM	MSNM	NSM	TRSM
$\rho_s^1(1)$	0.55	0.52	0.66	0.69
$\rho_s^1(2)$	0.84	0.76	0.87	0.93
$\rho_s^1(3)$	0.86	0.82	0.91	0.94
$\rho_s^2(1)$	0.64	0.55	0.48	0.55
$\rho_s^2(2)$	0.86	0.80	0.81	0.87
$\rho_s^2(3)$	0.86	0.81	0.88	0.91

4.2.2 Evaluation of the performance

In (3), Dolan and Moré have defined the performance profile of a solver as a (cumulative) distribution function for a given performance metric. We have evaluated and compared the performance of the solvers **LSNM** (Line search Newton method), **MSNM** (Moré-Sorensen Newton method), **NSM** (Newton-based Scaling method) and **TRSM** (Trust-Region Scaling method) on the set $\mathcal{P}$ of 119 test problems from the CUTEr collection.

We have considered two performance metrics, the number of iterations needed to attain the desired accuracy (since each iteration implies a considerable amount of work) and the corresponding number of function evaluations, both giving information on solvers robustness and efficiency.

Let p denote a particular problem of $\mathcal{P}$ and s a particular solver. The idea is to compare the performance of solver s on problem p with the best performance by any solver on this particular problem. As in (3), we define $\rho_s^j(\tau)$, $j = 1, 2$, as the probability that a performance ratio $r_{p,s}^j$ is within a factor of τ of the best possible ratio, where $j = 1$ refers to the number of iterations and $j = 2$ refers to the number of function evaluations. For $\tau = 1$, the probability $\rho_s^j(1)$ of a particular solver s is the probability that the value of the metric for the solver will be the best one among all solvers. The approximate values for the probabilities $\rho_s^j(\tau)$, $j = 1, 2$, when $\tau = 1, 2, 3$, are given in Table 8.

Regarding the number of iterations, from this table we see that **TRSM** has the highest probability of being the optimal solver. On the other hand, considering the number of function evaluations, we observe that **LSNM** has the highest probability of being the optimal solver.

In Figs. 1 and 2 we represent the performance profiles on a $\log_2$ scale , i.e., we plot τ (see (3) for the details). For large values of τ, the probability function $\rho_s^j(\tau)$ provides information about the total proportion of problems that a given code is able to solve. Thus, if we are interested in the probability that a solver will be successful, we should consider $\rho_s^j(\tau)$ for all solvers as τ becomes large (r_M^j). Figures 1 and 2 are plotted using a $\log_2$ scale to provide a more compact representation of the available data (if a linear scale were used, it would be necessary to include the whole interval [0, 1,024] to be able to include the largest values $r_{p,s}^j < r_M^j$, $j = 1, 2$).

Regarding the robustness of the methods, given that the problems are nonconvex in general, it could be the case that different methods converge to different local

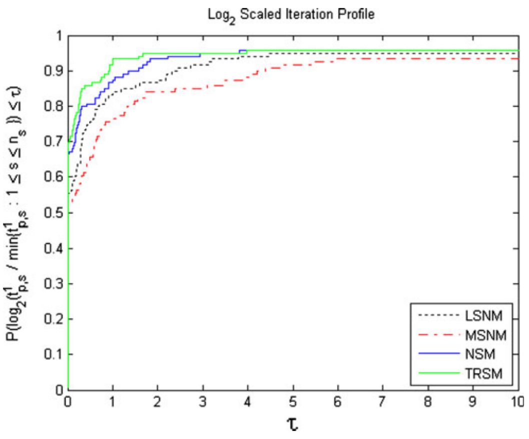

Fig. 1 Performance profile in a $\log_2$ scale-number of iterations

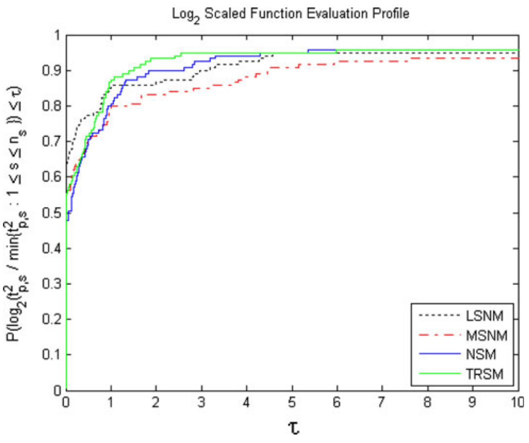

Fig. 2 Performance profile in a $\log_2$ scale-number of function evaluations

minimizers. In the set problem at hand, this happens for problems PENALTY3 and VIBRBEAM. For problem PENALTY3, methods **LSNM**, **NSM** and **TRSM** provide the same minimizer, while method **MSNM** improves the result obtained by the other three methods, not only with a lower objective function, but also with a decreased number of iterations and function evaluations. It is interesting to remark that the improvement obtained by method **MSNM** in the number of iterations and function evaluations is very large (more than 90% with respect to the best other method). For problem VIBRBEAM, each method provides a different result, been the best performance the one corresponding to **LSNM**, again with the lower objective function, iterations and function evaluations. In this case, the difference in the iteration count is small. For both problems, the best minimum is obtained by the method with less number of iterations and function evaluations, therefore not penalizing the overall performance of the method.

In Figs. 1 and 2, we see that both **NSM** and **TRSM** solve the highest number of problems (about 96%). In Fig. 1 we conclude that **TRSM** dominates all other solvers, since the performance profile for **TRSM** lies above all others for all performance ratios τ. Note that these results do not imply that **TRSM** solves every problem with a smaller number of iterations. For $\tau \geq 4$, solver **NSM** has the same performance as **TRSM**. In Fig. 2 we see that, for $1 \leq \tau \leq 4.3$, **TRSM** solves the largest number of problems requiring a number of function evaluations that is less than τ times that of any other solver. For $\tau \geq 6$, **NSM** is the only solver that attains the same best performance as **TRSM**. It is interesting to notice the good results of algorithm **LSNM** for the functions evaluation count. This is probably due to the fact that a standard backtracking search was implemented (which is specially efficient when a single direction is used). This shows that special care should be taken in the procedure to compute the final step length in the search to reduce the number of function evaluations when negative curvature and the gradient are used.

Considering all the previous facts, we conclude that the best results are obtained by the two proposals presented in this work.

5 Conclusions

In this paper we have studied different approaches to combine information in an efficient manner within an algorithm to solve unconstrained optimization problems. We show that negative curvature can be used successfully if some safeguards are taken into account. These safeguards refer mainly to the scaling problem that arises when dealing with descent directions of a different nature. In particular, we have proposed two procedures for the combination of the gradient, a modified Newton direction and a negative curvature direction. The procedures calculate, by performing a small number of iterations of a Newton's method or a trust-region method in a low dimensional subspace, the weights associated to the linear combination of the directions. In this way, this step can be viewed as a previous scaling of the directions. Then a linesearch procedure on the weights is carried out.

Our recommendation is that any algorithm using directions that arise from a different nature should include an scaling process before proceeding to the combination of the directions. This is the key point in the success of the algorithms proposed in this paper.

These results still require further work. In particular, the current environment of implementation does not allow the testing of the algorithms for large-scale problems. Nevertheless, given the success of the results for the problems in the test set, it seems to provide a promising starting point.

References

1. Bunch, J.R., Parlett, B N.: Direct methods for solving symmetric indefinite systems of linear equations. SIAM J. Numer. Anal. **8**, 639–655 (1971)
2. Byrd, R.H., Schnabel, R B., Shultz, G.A.: Approximate solution of the trust region problem by minimization over two-dimensional subspaces. Math. Program. **40**, 247–263 (1988)
3. Dolan, E., Moré, J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)
4. Fiacco, A.V., McCormick, G.P.: Nonlinear Programming: Sequential Unconstrained Minimization Techniques. Society for Industrial and Applied Mathematics, Philadelphia (1990)
5. Fletcher, R.: Practical Methods of Optimization, Volume 1, Unconstrained Optimization. Wiley, New York and Toronto (1980)
6. Forsgren, A., Murray, W.: Newton methods for large-scale linear equality-constrained minimization. SIAM J. Matrix Anal. Appl. **14**, 560–587 (1993)
7. Gill, P.E., Murray, W.: Newton type methods for unconstrained and linearly constrained optimization. Math. Program. **7**, 311–350 (1974)
8. Gill, P.E., Murray, W., Wright, M.H.: Practical Optimization. Academic Press, London and New York (1981)
9. Goldfarb, D.: Curvilinear path steplength algorithms for minimization which use directions of negative curvature. Math. Program. **18**, 31–40 (1980)
10. Gould, N.I M., Lucidi, S., Roma, M., Toint, Ph.L.: Exploiting negative curvature directions in linesearch methods for unconstrained optimization. Optim. Methods Softw. **14**, 75–98 (2000)
11. Gould, N.I.M., Orban, D., Toint, Ph.L.: CUTEr (and SifDec), a constrained and unconstrained testing environment, revisited. ACM Trans. Math. Softw. **29**, 373–394 (2003)
12. Hager, W.W., Zhang, H.: A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J. Optim. **16**, 170–192 (2005)
13. Moguerza, J M., Prieto, F.J.: An augmented Lagrangian interior-point method using directions of negative curvature. Math. Program. **95**, 573–616 (2003)
14. Moguerza, J.M., Prieto, F.J.: Combining search directions using gradient flows. Math. Program. **96**, 529–559 (2003)
15. Moré, J.J., Sorensen, D.C.: On the use of directions of negative curvature in a modified Newton method. Math. Program. **16**, 1–20 (1979)
16. Mukai, H., Polak, E.: A second-order method for the general nonlinear programming problem. J. Optim. Theory Appl. **26**, 515–532 (1978)
17. Olivares, A., Moguerza, J M., Prieto, F.J.: Nonconvex optimization using an adapted linesearch, Eur. J. Oper. Res. pages to be assigned, (2007)
18. Sanmatías, S., Vercher, E.: A generalized conjugate gradient algorithm. J. Optim. Theory Appl. **98**, 489–502 (1998)
19. Sanmatías, S., Roma, M.: Un método de búsqueda Lineal con Direcciones Combinadas Para La Optimización Irrestringida. Actas del XXVI Congreso Nacional de Estadística e Investigación Operativa, Úbeda, Spain (2001)
20. Sun, J., Yang, X., Chen, X.: Quadratic cost flow and the conjugate gradient method. Eur. J. Oper. Res. **164**, 104–114 (2005)