Variable metric method for minimization of partially separable nonsmooth functions

Ladislav Lukšan, Jan Vlček

Technical report No. 916

March 15, 2005
Variable metric method for minimization of partially separable nonsmooth functions

Ladislav Lukšan, Jan Vlček

March 15, 2005

Abstract:

In this report, we propose a new partitioned variable metric method for minimizing nonsmooth partially separable functions. After a short introduction, the complete algorithm is introduced and some implementation details are given. We prove that this algorithm is globally convergent under standard mild assumptions. Computational experiments given confirm efficiency and robustness of the new method.

Keywords:

Unconstrained optimization, large-scale optimization, nonsmooth optimization, bundle-type methods, variable metric methods, nonlinear least squares, partially separable problems, computational experiments.

---

1This work was supported by the Grant Agency of the Czech Academy of Sciences, project code IAA1030405. Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Praha 8 and Technical University of Liberec, Hálkova 6, 461 17 Liberec
1 Introduction

Nonsmooth optimization methods can be used efficiently in many areas of industrial design. The most frequently used nonsmooth objective functions have the following forms

\[ F(x) = \max_{1 \leq i \leq m} f_i(x), \]  
\[ F(x) = \sum_{i=1}^{m} f_i(x), \]

where \( f_i(x), 1 \leq i \leq m, \) are locally Lipschitz nonsmooth functions (usually absolute values of smooth functions). The first function, which correspond to \( l_\infty \) (minimax) approximation, can be used, e.g., for designing Chebyshev electrical filters. The second function, which corresponds to \( l_1 \) approximation, appears in image-restoration formulations as a term for recovering sharp edges. Both of these functions are locally Lipschitz and we are able to compute a (Clarke) subgradient \( g \in \partial F(x) \) at any point \( x \in \mathbb{R}^n \). Since a locally Lipschitz function is differentiable almost everywhere by the Rademacher theorem, then usually \( g = \nabla F(x) \). A special feature of nonsmooth problems is the fact that the gradient \( \nabla F(x) \) changes discontinuously and is not small in the neighborhood of a local extremum. Thus the standard optimization methods cannot be used efficiently.

The most commonly used approach for solving nonsmooth optimization problems is based on the bundle principle. In this case, values \( F(x_k), g(x_k) \in \partial F(x_k) \) at a single point \( x_k \) are replaced by a bundle of values \( F^j = F(y^j), g^j \in \partial F(y^j) \) obtained at trial points \( y^j, j \in J_k \subset \{1, \ldots, k\} \). This bundle of values serves for defining a piecewise quadratic function (with quadratic regularizing term), which is used for direction determination by solving a quadratic programming subproblem. The simplest proximal bundle methods use quadratic term with diagonal (usually scaled unit) matrix [7]. In this case, efficient methods require bundles containing approximately \( n \) elements, which is not practical in the large-scale case. Note that the quadratic programming subproblem corresponding to objective function (1) is sparse if functions \( f_i(x), 1 \leq i \leq m, \) have sparse subgradients. Thus the proximal bundle method can be used efficiently for function (1) if sparse quadratic programming solver is available. Unfortunately, it is not the case if objective function (2) is considered. In this case, the quadratic programming subproblem has dense constraints even if subgradients of functions \( f_i(x), 1 \leq i \leq m, \) are sparse.

To overcome difficulties concerning dense constraints, efficient nonsmooth variable metric methods [12],[14] were developed. In this case, variable metric updates accumulate information from previous iterations so that small-size bundles suffice for a rapid convergence. Thus we can use three-element bundles for an efficient implementation of the nonsmooth variable metric method and the solution of the corresponding quadratic programming subproblem can be obtained by simple formulas. Note that the nonsmooth variable metric method described in [14] uses three basic ideas of the bundle principle: quadratic programming subproblem with three bundle constraints, aggregation of subgradients and a special nonsmooth line search. These ideas will be mentioned below in connection with our new method.

The nonsmooth variable metric method described in [14] uses standard (dense) variable metric updates, which are not practical in the large-scale case. Therefore, additional possibilities motivated by the smooth case were studied. In [5], [6], efficient methods
utilizing limited-memory variable metric updates are described and their convergence is studied.

In this paper we focus our attention on partially separable functions of the form (2), where \(f_i(x), 1 \leq i \leq m\), are nonsmooth locally Lipschitz particular functions depending on \(n_i\) variables, where all numbers \(n_i\), \(1 \leq i \leq m\), are small comparing with \(n\). Partially separable functions are very popular in the smooth case, since efficient variable metric methods exist for seeking their minima [3]. Before description of the new method, we shortly describe the main ideas of variable metric methods for partially separable functions.

Let \(R_i^0 \subset R^n\) be the subspace defined by \(n_i\) variables appearing in \(f_i\) and \(Z_i \in R^{n \times n_i}\) be the matrix whose columns form the canonical orthonormal basis in \(R_i^0\) (i.e., they are columns of the unit matrix). Then we can define reduced gradients \(g_i = Z_i^T \nabla f_i\) and reduced Hessian matrices \(G_i = Z_i^T \nabla^2 f_i Z_i\). The \(k\)-th iteration of variable metric methods for partially separable functions starts in the point \(x^k\) with reduced gradients \(g_i^k\) and approximations of reduced Hessian matrices \(B_i^k, 1 \leq i \leq m\). Then gradient \(g^k\) and matrix \(B^k\) are constructed in such a way that

\[
g^k = \sum_{i=1}^{m} Z_i g_i^k, \quad B^k = \sum_{i=1}^{m} Z_i B_i^k Z_i^T
\]

and the direction vector \(d^k\) is computed by solving linear system \(B^k d^k = -g^k\). The new point \(x^{k+1} = x^k + \alpha^k d^k\) is determined by line search to satisfy the weak Wolfe conditions

\[
F(x^k + \alpha^k d^k) - F(x^k) \leq \varepsilon_1 \alpha^k (d^k)^T g^k, \\
(d^k)^T g(x^k + \alpha^k d^k) \geq \varepsilon_2 (d^k)^T g^k,
\]

with \(0 < \varepsilon_1 < 1/2\) and \(\varepsilon_1 < \varepsilon_2 < 1\). Finally, new reduced gradients \(g_i^{k+1}\) are computed and new approximations of reduced Hessian matrices \(B_i^{k+1}, 1 \leq i \leq m\) are obtained by variable metric updates using differences \(s_i^k = Z_i^T (x^{k+1} - x^k), y_i^k = g_i^{k+1} - g_i^k, 1 \leq i \leq m\).

We describe these updates in the next section.

The paper is organized as follows. In Section 2, we introduce a new variable metric method for minimizing partially separable nonsmooth functions and describe the corresponding algorithm in detail. In Section 3 we study theoretical properties of this partitioned nonsmooth variable metric method. Namely, we prove that this method is globally convergent under mild assumptions. Finally, in Section 4 we present results of our experiments confirming efficiency of the new method.

## 2 Description of the new method

The algorithm given below generates a sequence of basic points \(\{x^k\} \subset R^n\) which should converge to a minimizer of \(F : R^n \to R\) and a sequence of trial points \(\{y^k\} \subset R^n\) satisfying \(x^{k+1} = x^k + t_R^k d^k, y^{k+1} = x^k + t_L^k d^k\) for \(k \geq 1\) with \(y^1 = x^1\), where \(t_R^k > 0, t_R^k \geq t_L^k \geq 0\) are appropriately chosen stepsizes, \(B^k d^k = -\tilde{g}^k\) is a direction vector, \(\tilde{g}^k\) is an aggregate subgradient and the matrix \(B^k\) obtained by partitioned variable metric updates accumulates information about previous subgradients and represents an approximation of the Hessian matrix if function \(f\) is smooth. Stepsizes \(t_R^k\) and \(t_L^k\) are chosen by a special line-search procedure described in [14]. If the descent condition

\[
F(y^{k+1}) \leq F(x^k) - \varepsilon_L t_L^k w_k
\]

for some \(0 < \varepsilon_L < 1\), is fulfilled, we accept the trial point as a new basic point. The search direction is determined by

\[
d^k = \sum_{i=1}^{m} Z_i q_i^k,
\]

where \(q_i^k = g_i^k - Z_i^T Z_i q_i^{k-1}\). Then the sequence of trial points is generated by

\[
y^{k+1} = x^k + t_L^k d^k,
\]

and the sequence of stepsizes is chosen by a special line-search procedure described in [14].
is satisfied with suitable $t^k_L$, where $0 < \varepsilon_L < 1/2$ is fixed and $-w_k < 0$ represents the desirable amount of descent, then $x^{k+1} = y^{k+1}$ (descent step). In this case, the line-search procedure guarantees that either $t^k_L \geq \zeta$ or $\beta_{k+1} > \varepsilon_A w^k$, where

$$\beta_{k+1} = \max \left( \| F(x^k) - F(y^{k+1}) - (x^k - y^{k+1})^T g^{k+1} \|, \gamma \| x^k - y^{k+1} \| \right) \tag{5}$$

and $\zeta > 0$, $0 < \varepsilon_A < \varepsilon_R$ are fixed. Otherwise, if the condition

$$d^k)^T g^{k+1} \geq \beta^{k+1} - \varepsilon_R w^k \tag{6}$$

is satisfied, where $\varepsilon_L < \varepsilon_R < 1$ is fixed, then $x^{k+1} = x^k$ (null step). In this case, the line-search procedure guarantees that $\| y^{k+1} - z^{k+1} \| \leq \Delta$ where $\Delta$ is fixed and $z^{k+1}$ is a point such that $F(z^{k+1}) < F(x^k)$.

The aggregation procedure is very simple. Denoting by $l$ the lowest index satisfying $x^l = x^k$ (index of the iteration after the last descent step) and having the basic subgradient $g^l \in \partial F(x^l)$, the trial subgradient $g^{k+1} \in \partial F(y^{k+1})$ and the current aggregate subgradient $\tilde{g}^k$, we define $\tilde{g}^{k+1}$ as a convex combination

$$\tilde{g}^{k+1} = \lambda_1^{k} g^l + \lambda_2^{k} g^{k+1} + \lambda_3^{k} \tilde{g}^k,$$ \tag{7}

where multipliers $\lambda_1^{k}, \lambda_2^{k}, \lambda_3^{k}$ can be easily determined by minimization of a simple quadratic function (see Step 7 of Algorithm 1). This approach retains good convergence properties but eliminates the solution of the rather complicated quadratic programming subproblem that appears in standard bundle methods.

Matrices $B^k$ are generated by using partitioned variable metric updates [3]. After the null steps, symmetric rank one (SR1) update is used, since it gives a nondecreasing sequence of matrices as required for proving the global convergence. Because these properties are not necessary after descent steps, the standard BFGS update appears to be more suitable. Note that individual variable metric updates that could violate positive definiteness are skipped.

Efficiency of the algorithm is very sensitive to the initial stepsize selection, though it is not relevant for theoretical investigation. In fact, a bundle containing trial points and corresponding function values and subgradients is required for an efficient stepsize selection. Nevertheless, the initial stepsize selection does not require time-consuming operations (see Section 4 for details).

Now we are in a position to state the basic algorithm.

**Algorithm 1**

**Data:** A final accuracy tolerance $\varepsilon \geq 0$, restart parameters $\varepsilon_D \geq 0$, $\overline{f} > 0$, line search parameters $\varepsilon_A \geq 0$, $\varepsilon_L \geq 0$, $\varepsilon_R \geq 0$, stepsize bounds $\zeta > 0$, $\Delta > 0$, subgradient locality parameters $\gamma, \nu$ and a correction parameter $\rho \geq 0$.  

**Step 0:** *Initialization.* Choose the starting point $x^1 \in \mathbb{R}^n$ and positive definite reduced matrices $B^1_i$, $1 \leq i \leq m$ (e.g. $B^1_1 = I$, $1 \leq i \leq m$), set $y^1 = x^1$, $\alpha^1 = 0$ and compute $f^1_i = f_i(x^1)$, $g^1_i \in \partial f_i(x^1)$, $1 \leq i \leq m$ and $f^1 = F(x^1)$, $g^1 \in \partial F(x^1)$ (i.e. $g^1 = g^1_1 + \ldots + g^1_m$). Initialize iteration counter $k = 1$.

**Step 1:** *Descent step initiation.* Initialize the reduced aggregate subgradients $\tilde{g}^k_i = g^k_i$  

$1 \leq i \leq m$, the aggregate subgradient $\tilde{g}^k = \tilde{g}^k$, the aggregate subgradient locality measure $\tilde{\alpha}^k = 0$ and set $l = k$.  

3
Step 2: Direction determination. Determine $B_k$ from $B_k^i$, $1 \leq i \leq m$, and compute the Choleski decomposition $B^k = L^k D^k (L^k)^T$. Solve $L^k D^k (L^k)^T d^k = -g^k$ and set $d^k = d^k - \rho g^k$ and $w^k = -(\hat{g}^k)^T d^k + 2\hat{\alpha}^k$.

Step 3: Restart. If $k = l$ and either $-(d^k)^T g^k < \varepsilon_D \|d^k\|\|g^k\|$ or $\|d^k\| > \|\hat{g}\|\|g^k\|$, set $B^k = B^1$, $1 \leq i \leq m$ and go to Step 2.

Step 4: Stopping criterion. If $w^k \leq \varepsilon$, then stop.

Step 5: Line search. By the line search procedure given in [14] find stepsizes $t^k_l$ and $t^k_R$ and the corresponding quantities $x^{k+1} = x^k + t^k d^k$, $y^{k+1} = x^k + t^k d^k$, $f^{k+1} = f(x^{k+1})$, $g^{k+1} \in \partial F(y^{k+1})$, $1 \leq i \leq m$ and $f^{k+1} = F(x^k)$, $g^{k+1} \in \partial F(y^k)$ (i.e. $g^{k+1} = g_{k+1}^1 + \ldots + g_{m-1}^k$). Compute $\beta^{k+1}$ by (5). If $t^k_l > 0$, set $\alpha^{k+1} = 0$ (a descent step is taken), otherwise set $\alpha^{k+1} = \beta^{k+1}$ (a null step occurs).

Step 6: Update preparation. For $1 \leq i \leq m$, set $u^k_i = g^k_i - g^l_i$ and determine $s^k_i$ as a part of $s^k = t^k d^k$. If $t^k_l > 0$ (descent step), go to Step 9.

Step 7: Aggregation. Using the Choleski decomposition $B^k = L^k D^k (L^k)^T$, determine multipliers $\lambda^k_i \geq 0$, $i \in \{1, 2, 3\}$, $\lambda^k_1 + \lambda^k_2 + \lambda^k_3 = 1$, which minimize the function
\[
\varphi(\lambda_1, \lambda_2, \lambda_3) = (\lambda_1 g^l_1 + \lambda_2 g^k_1 + \lambda_3 \hat{g}_1^k)^T ((B^k)^{-1} + \rho I)(\lambda_1 g^l_1 + \lambda_2 g^k_1 + \lambda_3 \hat{g}_1^k) + 2(\lambda_2 \hat{\alpha}^{k+1} + \lambda_3 \hat{\alpha}^{k+1})
\]
Set $\hat{g}^{k+1} = \lambda_1 g^l_1 + \lambda_2 g^k_1 + \lambda_3 \hat{g}_1^k$, $1 \leq i \leq m$, $\hat{g}^{k+1} = \lambda_1 g^l_1 + \lambda_2 g^k_1 + \lambda_3 \hat{g}_1^k$, $\hat{\alpha}^{k+1} = \lambda_2 \hat{\alpha}^{k+1} + \lambda_3 \hat{\alpha}^{k+1}$.

Step 8: SR1 update. Let $v^k_i = u^k_i - B^k_i s^k_i$, $1 \leq i \leq m$. Set
\[
B^k_{i+1} = B^k_i + \frac{u^k_i (v^k_i)^T (s^k_i)^T v^k_i}{(s^k_i)^T v^k_i}, \quad (s^k_i)^T v^k_i > 0,
\]
\[
B^k_{i+1} = B^k_i, \quad (s^k_i)^T v^k_i \leq 0.
\]
Set $k = k + 1$ and go to Step 2.

Step 9: BFGS update. Set
\[
B^k_{i+1} = B^k_i + \frac{u^k_i (u^k_i)^T}{(s^k_i)^T u^k_i} - \frac{B^k_i s^k_i (B^k_i s^k_i)^T}{(s^k_i)^T B^k_i s^k_i}, \quad (s^k_i)^T u^k_i > 0,
\]
\[
B^k_{i+1} = B^k_i, \quad (s^k_i)^T u^k_i \leq 0.
\]
Set $k = k + 1$ and go to Step 1.

Conditions in Step 3 of the algorithm, guaranteeing that vectors $d^k$, $k \geq 1$, are uniformly bounded and eliminating badly conditioned cases, appear rarely and do not have influence on the efficiency of the algorithm. At the same time, corrections with $\rho > 0$ in Step 2
strongly affect efficiency of the method.

3 Properties of the new method

In this section, we prove under mild assumptions that the new method is globally convergent, which means that every cluster point \( x^* \) of the sequence \( \{x^k\} \subset \mathbb{R}^n \) is a stationary point of \( F \), i.e., \( 0 \in \partial F(x^*) \). For this purpose, we will assume that sequence \( \{x^k\} \) is infinite, i.e., \( \varepsilon = 0 \) in Algorithm 1.

**Assumption 1** Points \( x^k, y^k, k \geq 1 \), lie in the compact region \( \mathcal{D}(F) \) and functions \( f_i : \mathbb{R}^n \to \mathbb{R}, 1 \leq i \leq m \), are locally Lipschitz on \( \mathcal{D}(F) \).

**Remark 1** If the level set \( \mathcal{L}(F) = \{x \in \mathbb{R}^n : F(x) \leq F(x^1)\} \) is compact, then also set \( \mathcal{D}(F) = \mathcal{L}(F) + \Delta \mathcal{B}(0, 1) \) is compact (\( \mathcal{B}(0, 1) \) is the unit ball). Moreover, \( x^k \in \mathcal{L}(F) \) and \( y^k \in \mathcal{D}(F) \) for \( k \geq 1 \), since \( \|y^k - z^k\| \leq \Delta \) for some \( z^k \in \mathcal{L}(F) \) (it is assured by our line search procedure). If \( f_i, 1 \leq i \leq m \), are locally Lipschitz on \( \mathcal{D}(F) \), then also \( F \) is locally Lipschitz on \( \mathcal{D}(F) \). Thus subgradients \( g^k \in \partial F(y^k), k \geq 1 \), all their convex combinations and also values \( |F(x^k)|, k \geq 1 \) are uniformly bounded (see [1]). Conditions in Step 2 of Algorithm 1 assure that direction vectors \( d^k, k \geq 1 \), are uniformly bounded.

First we will investigate null steps. In the null steps, \( d^k = -H^k \tilde{g}^k \) and \( w^k = (\tilde{g}^k)^T H^k \tilde{g}^k + 2\tilde{\alpha}^k \) hold, where \( H^k = (B^k)^{-1} + \rho I \).

**Lemma 3.1** Let \( \rho > 0 \). Then matrices \( H^k, k \geq 1 \), are uniformly positive definite. Moreover matrices \( H^k \) are uniformly bounded and differences \( H^k - H^{k+1} \) are positive semidefinite in consecutive null steps (even if \( \rho = 0 \)).

**Proof.** Variable metric updates used in Steps 8 and 9 of Algorithm 1 guarantee that all matrices \( B^k_i, 1 \leq i \leq m, k \geq 1 \), are positive definite (see [10]). Since

\[
v^T B^k v = \sum_{i=1}^{m} v^T Z_i B^k_i Z_i v = \sum_{i=1}^{m} v^T B^k_i v_i
\]

and \( v_i = Z_i^T v \neq 0 \) for at least one index \( 1 \leq i \leq m \) if \( v \neq 0 \), also matrices \( B^k, (B^k)^{-1}, k \geq 1 \), are positive definite and if \( \rho > 0 \), matrices \( H^k = (B^k)^{-1} + \rho I, k \geq 1 \), are uniformly positive definite. In the null steps, rank 1 updates used in Step 8 of Algorithm 1 assure that differences \( B^{k+1}_i - B^k_i, 1 \leq i \leq m \), are positive semidefinite (since \( (s_i^k)^T v_i^k > 0 \), if the rank 1 update is not skipped). Thus \( B^{k+1}_i - B^k_i \) is positive semidefinite and, therefore, \( H^k - H^{k+1} = (B^k)^{-1} - (B^{k+1})^{-1} \) is positive semidefinite (the last fact is proved in [9]). Positive semidefiniteness of \( H^k - H^{k+1} \) implies that \( \|H^{k+1}\| \leq \|H^k\| \). Thus matrices \( H^k \) are uniformly bounded in consecutive null steps. \( \square \)

**Lemma 3.2** Let \( \rho > 0 \) and Assumption 1 holds. If the number of descent steps in Algorithm 1 is finite, then \( w^k \to 0 \).

**Proof.** Let \( x^l \) be the last point obtained by a descent step and \( k \geq l \). Denote

\[
\tilde{g}^k(\lambda) = \lambda g^{k+1} + (1 - \lambda)g^k, \\
\tilde{\alpha}^k(\lambda) = \lambda \alpha^{k+1} + (1 - \lambda)\alpha^k.
\]
where $0 \leq \lambda \leq 1$. Since matrix $H^k - H^{k+1}$ is positive semidefinite by Lemma 3.1, we can write

$$w^{k+1} = (\tilde{g}^{k+1})^T H^{k+1} \tilde{g}^{k+1} + 2\tilde{\alpha}^{k+1} \leq (\tilde{g}^{k+1})^T H^k \tilde{g}^{k+1} + 2\tilde{\alpha}^{k+1} \leq (\tilde{g}^k(\lambda))^T H^k \tilde{g}^k(\lambda) + 2\tilde{\alpha}_k(\lambda) = w_k(\lambda).$$

The last inequality follows from the fact that pair $(\tilde{g}^{k+1}, \tilde{\alpha}^{k+1})$ minimizes $\varphi(\lambda_1, \lambda_2, \lambda_3)$ (in Step 7 of Algorithm 1) over all convex combinations of pairs $(g_i, \alpha_i)$, $(g^{k+1}, \alpha^{k+1})$, $(\tilde{g}^k, \tilde{\alpha}^k)$. Furthermore, inequality

$$\alpha_{k+1} + (g^{k+1})^T H^k \tilde{g}^{k+1} \leq \varepsilon_R w_k$$

holds for $k \geq l$, since $\alpha_{k+1} = \beta_{k+1}$ in null steps. By successive arrangements, we obtain

$$w^k(\lambda) = (\tilde{g}^k(\lambda))^T H^k \tilde{g}^k(\lambda) + 2\tilde{\alpha}_k(\lambda) = (\tilde{g}^k)^T H^k \tilde{g}^k + 2\tilde{\alpha}_k + 2\lambda((g^{k+1})^T H^k \tilde{g}^{k+1} - (\tilde{g}^k)^T H^k \tilde{g}^k + \alpha^{k+1} - \tilde{\alpha}^k) + 2\lambda^2 (g^{k+1} - \tilde{g}^k)^T H^k (g^{k+1} - \tilde{g}^k) \leq w_k + 2\lambda \varepsilon_R w_k - 2\lambda w_k + 2\lambda^2 (g^{k+1} - \tilde{g}^k)^T H^k (g^{k+1} - \tilde{g}^k) \leq w_k + 2\lambda \varepsilon_R w_k - w_k - (1 - \varepsilon_R)^2 w_k^2 / M.$$ 

where the existence of constant $M$ follows from the boundedness of vectors $g^{k+1}$, $\tilde{g}^k$ and matrices $H^k$ (see Remark 1 and Lemma 3.1). The expression on the right hand side acquires the minimum for $\lambda = (1 - \varepsilon_R) w_k / M$ and its minimum value is equal to $w_k - (1 - \varepsilon_R)^2 w_k^2 / M$. Therefore, one has

$$w_{k+1} \leq w_k - \frac{(1 - \varepsilon_R)^2 w_k^2}{M}.$$ 

(8)

Now we can easily finish the proof. We show that $w_k \to 0$. If it were not true, constant $\delta > 0$ would have to exist such that $w_k \geq \delta, \forall k \geq l$ (since sequence $\{w_k\}$ is nonincreasing for $k \geq l$). Then we would obtain $w_{k+1} \leq w_k - (1 - \varepsilon_R)^2 \delta^2 / M \forall k \geq l$ from (8) so that $w_k < \delta$ would hold for sufficiently large indices, which is a contradiction.

**Theorem 3.1** Let the number of descent steps in Algorithm 1 be finite and $x^l$ be the last point obtained by a descent step. If $\rho > 0$ and Assumption 1 holds, then $x^l$ is a stationary point of $F$, i.e., $0 \in \partial F(x^l)$.

**Proof.** If $k = l$, then $\tilde{g}^k = g^k$ and $\tilde{\alpha}^k = 0$ (Step 1 of Algorithm 1). If $k > l$, then pair $(\tilde{g}^k, \tilde{\alpha}^k)$ is a convex combination of pairs $(g_i, \alpha_i)$, $(g^k, \alpha^k)$, $(g^{k-1}, \alpha^{k-1})$ (Step 7 of Algorithm 1) and, therefore, it is a convex combination of pairs $(g_i, \alpha_i)$, $l \leq i \leq k$, where $\alpha^l = \max \{ ||F(y^j) - F(x^l)|| - (y^j - x^l)^T g^l, \gamma|y^j - x^l|\gamma \} \geq \gamma|y^j - x^l|\gamma^l$ (9)

(with $\alpha^l = 0$, since $y^l = x^l$). By the Caratheodory theorem, there exist at most $n + 2$ pairs $(g^{k,i}, \alpha^{k,i}), g^{k,i} \in \partial f(y^{k,i}), (y^{k,i}, g^{k,i}, \alpha^{k,i}) \in \{(y^i, g^i, \alpha^i) : l \leq i \leq k\}$ such that

$$(\tilde{g}^k, \tilde{\alpha}^k) = \sum_{i=1}^{n+2} \lambda^{k,i} (g^{k,i}, \alpha^{k,i}),$$

(10)
where $\lambda^{k,i} \geq 0$, $1 \leq i \leq n+2$, $\lambda^{k,1} + \ldots + \lambda^{k,n+2} = 1$. Since vectors $y^{k,i}$, $g^{k,i}$, $1 \leq i \leq n+2$, are uniformly bounded, there is a subset $K$ such that $y^{k,i} \in K$, $g^{k,i} \in K$, $\lambda^{k,i} \in K$, $1 \leq i \leq n+2$. But $g^{*i} \in \partial f(y^{*i})$, $1 \leq i \leq n+2$ (see [1]) and (10) implies that $(\tilde{g}^k, \tilde{\alpha}^k) \to (\tilde{g}^*, \tilde{\alpha}^*)$, where

$$(\tilde{g}^*, \tilde{\alpha}^*) = \sum_{i=1}^{n+2} \lambda^{*,i}(g^{*,i}, s^{*,i})$$

and $\lambda^{*,i} \geq 0$, $1 \leq i \leq n+2$, $\lambda^{*,1} + \ldots + \lambda^{*,n+2} = 1$. Since $w^k \to 0$ by Lemma 3.2 and matrices $H^k$ are uniformly positive definite, one has $\tilde{g}^k \overset{K}{\to} 0$, $\tilde{\alpha}^k \overset{K}{\to} 0$, which implies

$$(0,0) = \sum_{i=1}^{n+2} \lambda^{*,i}(g^{*,i}, \alpha^{*,i}).$$

Assume without loss of generality that $\lambda^{*,i} > 0$, $1 \leq i \leq n+2$ (zero multipliers can be omitted). Since $\alpha^{*,i} \geq \gamma|y^{*,i} - x^*|^\nu \geq 0$ by (9), we obtain $\alpha^{*,i} = 0$ and $y^{*,i} = x^i$ for $1 \leq i \leq n+2$. Thus $g^i \in \partial F(y^i) = \partial F(x^i)$ and 0 = $\lambda^i_1 g^i + \ldots + \lambda^i_{n+2} g^i_{n+2} \in \partial F(x^i)$. □

**Theorem 3.2** Let $\rho > 0$ and Assumption 1 holds. Then every cluster point of the sequence $\{x^k\} \subset R^n$ obtained by Algorithm 1 is a stationary point of $F$.

**Proof.** If the number of descent steps in Algorithm 1 is finite, then there is a unique cluster point $x^*$ of the sequence $\{x^k\}$, which is a stationary point of $F$ by Theorem 3.1. Assume that the number of descent steps is infinite and $x^k \overset{K}{\to} x^*$. Since the current point is unchanged in null steps, we can assume without loss of generality that points $x^k \in K$ were chosen in such a way that the step $x^{k+1} = x^k + t^k L d^k$ is descent (note that point $x^{k+1}$ can lie outside $K$). Since sequence $\{F(x^k)\}$ is non-increasing and bounded from below by Assumption 1, one has $F(x^k) - F(x^{k+1}) \to 0$, which together with

$$0 \leq \epsilon_L t^k L w^k \leq F(x^k) - F(x^{k+1})$$

(see (4)) gives $t^k L w^k \overset{K}{\to} 0$. Let $K = K_1 \cup K_2$ where $K_1 = \{k \in K : t^k \geq \theta\}$ and $K_2 = \{k \in K : t^k \geq \delta\}$ (this partitioning is guaranteed by our line search procedure, see Section 2). If $K_1$ is infinite, then $t^k L w^k \overset{K_1}{\to} 0$ implies $w^k \overset{K_1}{\to} 0$, which together with $w^k = (g_k)^T H_k g^k$ (since $\tilde{g}^k = g^k$ and $\tilde{\alpha}^k = 0$ in descent steps) and uniform positive definiteness of matrices $H^k$ (Lemma 3.1) gives $g^k \overset{K_1}{\to} 0$. Thus $0 \in \partial F(x^*)$ (see [1]). If $K_1$ is finite, then $K_2$ is infinite. Assume first that subset $K_3 = \{k \in K_2 : w^k \geq \delta\}$ is infinite for some $\delta > 0$. Then $t^k L w^k \overset{K_3}{\to} 0$ implies $t^k L \overset{K_3}{\to} 0$ and since vectors $d^k$ are uniformly bounded (Remark 1), one has

$$\|x^{k+1} - x^k\| = t^k L \|d^k\| \overset{K_3}{\to} 0.$$ 

Since $g^{k+1} = x^{k+1}$ in descent steps, we obtain $\|y^{k+1} - x^k\| \overset{K_3}{\to} 0$, which together with (5) and continuity of $F$ gives $\beta^{k+1} \overset{K_3}{\to} 0$. Since $K_3 \subset K_2$ one has $0 \leq \epsilon_A w^k < \beta^{k+1}$ for all $k \in K_3$. Thus $w^k \overset{K_3}{\to} 0$, which is the contradiction with the definition of $K_3$ implying that $K_3$ is finite. In this way, we have proved that $w^k \overset{K_3}{\to} 0$. Thus $g^k \overset{K_3}{\to} 0$ and $0 \in \partial F(x^*)$ as in the case when $K_1$ is infinite. □
4 Implementation details

Algorithm 1 contains all important features of the partitioned nonsmooth variable metric method, which are necessary for the theoretical investigation. In this section we discuss some details concerning our implementation of the algorithm. These details are in fact the same as described in [14], i.e., Algorithm 1 is implemented in a similar way as Algorithm 2.1 in [14]. Therefore, we only mention main ideas, details can be found in [14].

Since quadratic programming subproblem used in Step 7 of Algorithm 1 is very simple, initial stepsize \( t = 1 \) need not be a good choice in connection with its solution. Therefore we store and use a bundle of values \( F^j = F(y^j), g^j \in \partial F(y^j) \) obtained at trial points \( y^j, j \in J_k = \{k - n_B + 1, \ldots, k\} \). These values serve for the construction of the piecewise linear function

\[
\varphi^k_P(t) = \max_{j \in J_k} \{ F(x^k + td^k) + t(d^k)^T g^j - \beta^k_j \},
\]

where

\[
\beta^k_j = \max(\|F^k - F(y^j) - (x^k - y^j)^T g^j\|, \gamma |x^k - y^j| \nu).
\]

After a descent step, we use quadratic approximation

\[
\varphi^k_Q(t) = F^k + t(d^k)^T g^k + \frac{1}{2} t^2 (d^k)^T (H^k)^{-1} d^k = F^k + (t - \frac{1}{2} t^2) (d^k)^T g^k
\]

and compute the initial stepsize by minimizing the function \( \varphi^k(t) = \max(\varphi^k_P(t), \varphi^k_Q(t)) \) in the interval \( 0 \leq t \leq 2 \). After a null step, the unit stepsize is mostly satisfactory. To utilize the bundle and improve the robustness and the efficiency of the method, we use the aggregate subgradient \( \tilde{g}_k \) to construct the linear approximation \( \varphi^k_L(t) = F^k + t(d^k)^T \tilde{g}_k \) of \( F(x^k + td^k) \) and compute the initial stepsize by minimizing the function

\[
\varphi^k(t) = \max(\psi^k_L(t), \psi^k_P(t)) + \frac{1}{2} t^2 (d^k)^T (H^k)^{-1} d^k = \max(\psi^k_L(t), \psi^k_P(t)) - \frac{1}{2} t^2 (d^k)^T g^k
\]

in the interval \( 0 \leq t \leq 1 \).

The second comment is related to the solution of the simple quadratic programming subproblem in Step 7 of Algorithm 1. This computation is not time consuming, but corresponding formulas are not simple because of the possible influence of round-off errors. More details are given in [14].

What concerns the termination criterion, the simple test \( w^k \leq \varepsilon \) is not always suitable, since it can lead to premature termination. Therefore additional conditions should be satisfied simultaneously. These conditions are discussed in [14] where a suitable termination criterion is introduced.

5 Computational experiments

Our partitioned nonsmooth variable metric method was tested by using the collection of relatively difficult problems with optional dimension chosen from [11], which can be downloaded (together with the above report) from \url{www.cs.cas.cz~luksan/}
test.html as Test 15. In [11], functions \( f_i(x), 1 \leq i \leq m \), are given, which serve for defining the objective function

\[ F(x) = \sum_{i=1}^{m} |f_i(x)|. \]

We have used parameters \( \varepsilon = 10^{-8}, \varepsilon_D = 10^{-6}, \varepsilon_A = 10^{-4}, \varepsilon_L = 10^{-4}, \varepsilon_R = 0.25, H = 10^{10}, t = 10^{-10}, \nu = 2, \rho = 10^{-12} \) and \( n_B = 20 \) (size of the bundle for the initial stepsize selection) in our tests. Parameters \( \Delta \) (the maximum stepsize) and \( \gamma \) (subgradient locality parameter) were carefully tuned for every method (including VBM and PBM).

Results of computational experiments are given in three tables, where P is the problem number, NEV is the number of function and also gradient evaluations and \( F \) is the function value reached. Note that problems used are relatively difficult, usually having more local solutions (values 8.000000, 64.000000 in Table 1 and 2.000000, 390.000000, 264.000000 in Table 2 seems to be local solutions different from the global ones). The last row of every table contains summary results: the total number of function evaluations and the total computational time. Tables 1 and 2 contain comparison of the new method PSVBM (Algorithm 1) with the nonsmooth variable metric method VBM described in [14] and the proximal bundle method PBM (see [7]) on small (n=50) and medium (n=200) size problems. Table 3 compares three versions (\( \rho = 0, \rho = 10^{-12} \) and \( \rho = 10^{-10} \)) of our PSVBM method on large scale partially separable problems with 1000 variables.

<table>
<thead>
<tr>
<th>( P )</th>
<th>NEV</th>
<th>( F )</th>
<th>NEV</th>
<th>( F )</th>
<th>NEV</th>
<th>( F )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>605</td>
<td>.102544E-08</td>
<td>5096</td>
<td>.545119E-13</td>
<td>6258</td>
<td>.387512E-07</td>
</tr>
<tr>
<td>2</td>
<td>235</td>
<td>.275184E-08</td>
<td>998</td>
<td>8.000000E00</td>
<td>1242</td>
<td>.156714E-07</td>
</tr>
<tr>
<td>3</td>
<td>70</td>
<td>.758451E-10</td>
<td>440</td>
<td>.532703E-08</td>
<td>2631</td>
<td>.211869E-07</td>
</tr>
<tr>
<td>4</td>
<td>72</td>
<td>29.10695109</td>
<td>338</td>
<td>29.10695109</td>
<td>169</td>
<td>34.10626848</td>
</tr>
<tr>
<td>5</td>
<td>242</td>
<td>.663320E-09</td>
<td>353</td>
<td>.473741E-08</td>
<td>133</td>
<td>.180701E-08</td>
</tr>
<tr>
<td>6</td>
<td>248</td>
<td>.323124E-08</td>
<td>447</td>
<td>.445267E-08</td>
<td>197</td>
<td>.887983E-08</td>
</tr>
<tr>
<td>7</td>
<td>296</td>
<td>583.9000724</td>
<td>1000</td>
<td>566.7522096</td>
<td>1175</td>
<td>583.9002160</td>
</tr>
<tr>
<td>8</td>
<td>356</td>
<td>.765773E-08</td>
<td>1320</td>
<td>.598195E-08</td>
<td>5476</td>
<td>.150995E-05</td>
</tr>
<tr>
<td>9</td>
<td>1821</td>
<td>134.8481786</td>
<td>688</td>
<td>132.6493773</td>
<td>3027</td>
<td>137.8278133</td>
</tr>
<tr>
<td>10</td>
<td>748</td>
<td>33.23330761</td>
<td>1051</td>
<td>32.36926315</td>
<td>3274</td>
<td>32.36926680</td>
</tr>
<tr>
<td>11</td>
<td>25887</td>
<td>.375443E-09</td>
<td>11198</td>
<td>.505268E-08</td>
<td>15915</td>
<td>.678975E-08</td>
</tr>
<tr>
<td>12</td>
<td>5310</td>
<td>150.3277379</td>
<td>3886</td>
<td>171.9263661</td>
<td>8489</td>
<td>150.3276924</td>
</tr>
<tr>
<td>13</td>
<td>440</td>
<td>709.6182298</td>
<td>806</td>
<td>709.6182298</td>
<td>10865</td>
<td>709.6182308</td>
</tr>
<tr>
<td>14</td>
<td>303</td>
<td>27.22786762</td>
<td>440</td>
<td>27.22786762</td>
<td>298</td>
<td>27.22786763</td>
</tr>
<tr>
<td>15</td>
<td>267</td>
<td>8.749955784</td>
<td>404</td>
<td>8.749955784</td>
<td>214</td>
<td>8.749955787</td>
</tr>
<tr>
<td>16</td>
<td>506</td>
<td>3.200000006</td>
<td>1214</td>
<td>3.200000000</td>
<td>1504</td>
<td>3.200000136</td>
</tr>
<tr>
<td>17</td>
<td>486</td>
<td>.283228E-08</td>
<td>571</td>
<td>.435820E-01</td>
<td>2704</td>
<td>.485198E-06</td>
</tr>
<tr>
<td>18</td>
<td>10119</td>
<td>.272617E-08</td>
<td>1565</td>
<td>18.55686116</td>
<td>3257</td>
<td>.368177E-07</td>
</tr>
<tr>
<td>19</td>
<td>462</td>
<td>.486099E-08</td>
<td>520</td>
<td>.438375E-08</td>
<td>1083</td>
<td>.208525E-05</td>
</tr>
<tr>
<td>20</td>
<td>443</td>
<td>.483206E-08</td>
<td>634</td>
<td>.423488E-08</td>
<td>9669</td>
<td>.330697E-07</td>
</tr>
<tr>
<td>21</td>
<td>665</td>
<td>64.00000000</td>
<td>316</td>
<td>63.98856687</td>
<td>562</td>
<td>64.000000001</td>
</tr>
<tr>
<td>22</td>
<td>385</td>
<td>143.3417786</td>
<td>4084</td>
<td>143.3786767</td>
<td>17004</td>
<td>143.3677713</td>
</tr>
</tbody>
</table>

\[ \Sigma \] 49920 \hspace{1cm} \text{TIME} = 5.83 \hspace{1cm} 37369 \hspace{1cm} \text{TIME} = 5.20 \hspace{1cm} 95146 \hspace{1cm} \text{TIME} = 48.82

Table 1: 22 problems with 50 variables
Table 2: 22 problems with 200 variables

<table>
<thead>
<tr>
<th></th>
<th>PSVBM NEV F</th>
<th>VBM NEV F</th>
<th>PBM NEV F</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>1</td>
<td>3134 .287703E-08</td>
<td>12782 2.000000022</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>305 .379422E-08</td>
<td>1471 .390.0000000</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>74 .226193E-09</td>
<td>1259 .807210E-07</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>51 126.8635489</td>
<td>89 126.8635489</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>282 .732927E-08</td>
<td>699 .883032E-07</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>344 .836329E-08</td>
<td>1269 .895936E-07</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>289 .732927E-07</td>
<td>1891 .2351.383667</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>616 .317243E-05</td>
<td>2571 .971029E-06</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>2516 552.3805506</td>
<td>4061 550.4463151</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>907 131.8884763</td>
<td>1581 131.0242899</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>17908 .654127E-02</td>
<td>20000 310.9025527</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>2043 621.1289465</td>
<td>20000 635.0621402</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>718 2940.509429</td>
<td>1352 2940.509413</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>348 112.3149539</td>
<td>1018 112.3149541</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>364 36.09356760</td>
<td>1896 36.09356759</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>380 268534E-01</td>
<td>1314 .929480E-07</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>5225 .8002196153</td>
<td>2219 .9441431850</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>4056 .565862E-08</td>
<td>1679 .963094E-07</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>701 .404661E-08</td>
<td>1845 .104635E-06</td>
</tr>
<tr>
<td></td>
<td>21</td>
<td>253 264.0000000</td>
<td>1122 264.0000001</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>1425 593.3607049</td>
<td>8914 593.3678758</td>
</tr>
<tr>
<td>Σ</td>
<td>48021 TIME = 28.78</td>
<td>90605 TIME = 88.83</td>
<td>186561 TIME = 1204.50</td>
</tr>
</tbody>
</table>

Results presented in the above tables imply several conclusions:

- Our partitioned nonsmooth variable metric method is competitive with standard nonsmooth methods VBM and PBM in small-size problems. It gives the best results for medium-size problems with 200 variables for which the proximal bundle method (that uses quadratic programming subproblems with large numbers of constraints) is unsuitable.

- Partitioned nonsmooth variable metric method successfully solves large-scale problems, which cannot be solved by standard nonsmooth methods utilizing dense matrices.

- A nonzero value of parameter $\rho$ has not only a theoretical significance, but it really improves efficiency and robustness of the method. Usually very small values $\rho = 10^{-12}$ or $\rho = 10^{-10}$ are sufficient. Greater values, e.g. $\rho = 10^{-8}$, decrease the efficiency of the method.
\[ \rho = 0 \]

\[ \rho = 10^{-12} \]

\[ \rho = 10^{-10} \]

<table>
<thead>
<tr>
<th>P</th>
<th>NEV</th>
<th>F</th>
<th>NEV</th>
<th>F</th>
<th>NEV</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>166</td>
<td>.147116E-03</td>
<td>114</td>
<td>.374913E-08</td>
<td>218</td>
<td>3.125593E-05</td>
</tr>
<tr>
<td>2</td>
<td>400</td>
<td>.481325E-05</td>
<td>600</td>
<td>.649530E-08</td>
<td>400</td>
<td>.649530E-08</td>
</tr>
<tr>
<td>3</td>
<td>106</td>
<td>.12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
</tr>
<tr>
<td>4</td>
<td>1428</td>
<td>4020.325073</td>
<td>1032</td>
<td>.680061E-04</td>
<td>1032</td>
<td>.680061E-04</td>
</tr>
<tr>
<td>5</td>
<td>295</td>
<td>.89.80742693</td>
<td>728</td>
<td>650</td>
<td>1376</td>
<td>66.53333334</td>
</tr>
<tr>
<td>6</td>
<td>555</td>
<td>658.0486676</td>
<td>128</td>
<td>648.2320706</td>
<td>128</td>
<td>648.2320706</td>
</tr>
<tr>
<td>7</td>
<td>7066</td>
<td>3135.352992</td>
<td>3754</td>
<td>992.932153</td>
<td>3754</td>
<td>992.932153</td>
</tr>
<tr>
<td>8</td>
<td>386</td>
<td>181.9261656</td>
<td>645</td>
<td>181.9261656</td>
<td>645</td>
<td>181.9261656</td>
</tr>
<tr>
<td>9</td>
<td>1428</td>
<td>181.9261656</td>
<td>114</td>
<td>181.9261656</td>
<td>114</td>
<td>181.9261656</td>
</tr>
<tr>
<td>10</td>
<td>377</td>
<td>181.9261656</td>
<td>650</td>
<td>181.9261656</td>
<td>650</td>
<td>181.9261656</td>
</tr>
<tr>
<td>11</td>
<td>1066</td>
<td>3135.352992</td>
<td>1032</td>
<td>.680061E-04</td>
<td>1032</td>
<td>.680061E-04</td>
</tr>
<tr>
<td>12</td>
<td>337</td>
<td>.296523E-01</td>
<td>728</td>
<td>648.2320706</td>
<td>728</td>
<td>648.2320706</td>
</tr>
<tr>
<td>13</td>
<td>528</td>
<td>12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
</tr>
<tr>
<td>14</td>
<td>582</td>
<td>12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
</tr>
<tr>
<td>15</td>
<td>78</td>
<td>.364876E-13</td>
<td>114</td>
<td>.374913E-08</td>
<td>114</td>
<td>.374913E-08</td>
</tr>
<tr>
<td>16</td>
<td>400</td>
<td>.481325E-05</td>
<td>600</td>
<td>.649530E-08</td>
<td>600</td>
<td>.649530E-08</td>
</tr>
<tr>
<td>17</td>
<td>1428</td>
<td>4020.325073</td>
<td>1032</td>
<td>.680061E-04</td>
<td>1032</td>
<td>.680061E-04</td>
</tr>
<tr>
<td>18</td>
<td>555</td>
<td>658.0486676</td>
<td>728</td>
<td>648.2320706</td>
<td>728</td>
<td>648.2320706</td>
</tr>
<tr>
<td>19</td>
<td>7066</td>
<td>3135.352992</td>
<td>3754</td>
<td>992.932153</td>
<td>3754</td>
<td>992.932153</td>
</tr>
<tr>
<td>20</td>
<td>337</td>
<td>.296523E-01</td>
<td>728</td>
<td>648.2320706</td>
<td>728</td>
<td>648.2320706</td>
</tr>
<tr>
<td>21</td>
<td>528</td>
<td>12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
<td>650</td>
<td>.12029.94285</td>
</tr>
<tr>
<td>22</td>
<td>78</td>
<td>.364876E-13</td>
<td>114</td>
<td>.374913E-08</td>
<td>114</td>
<td>.374913E-08</td>
</tr>
</tbody>
</table>

\[ \sum \text{NEV} = 25060 \quad \text{TIME} = 110.67 \]

\[ \sum \text{NEV} = 56990 \quad \text{TIME} = 296.86 \]

\[ \sum \text{NEV} = 60754 \quad \text{TIME} = 325.88 \]

Table 3: 22 problems with 1000 variables

References


