TECHNISCHE UNIVERSITÄT CHEMNITZ

Acceleration of the Level Method by Exploiting Recurrent Subgradients in Linear Programming Decomposition

T. Unger

Preprint 02-1

Fakultät für Mathematik

Acceleration of the Level Method by Exploiting Recurrent Subgradients in Linear Programming Decomposition

Thomas Unger[1]

Abstract

The level method is a special variant of the well-known cutting plane methods for solving nonsmooth convex optimization problems. It proved to be competitive with other methods in nonsmooth optimization.
It has been observed that during the solving process these methods compute cutting planes which are (nearly) parallel to each other. In this paper we investigate what effects appear together with this observation and how they may be exploited to speed up the level method algorithm for some classes of problems.

MSC: 90C25, 65K05, 90C05.

Keywords: nonsmooth programming, cutting plane methods, level method, piecewise linear functions.

1 Motivation

We describe a way to accelerate the performance of the level method applied to minimize the marginal function of a decomposed (large) linear program. The problem to be solved shall be given as

$$\begin{cases} \langle c, x\rangle + \langle d, y\rangle \to \min\limits_{x,y} \\ Ax + By = f \\ x, y \geq 0. \end{cases} \tag{1}$$

We suppose that (1) is solvable. Let y^* denote the y-part of an optimal solution; furthermore, let $u > 0$ such that $y^* \in Y := \{y : 0 \leq y_i \leq u \quad \forall\, i\}$.

Primal decomposition generates the convex marginal function

$$\varphi(y) = \langle d, y\rangle + \inf_x\{\langle c, x\rangle : Ax = f - By, x \geq 0\}. \tag{2}$$

[1]Faculty of Mathematics, TU Chemnitz, 09107 Chemnitz, Germany;
thomas.unger@mathematik.tu-chemnitz.de

It is well-known that the substitute

$$\left\{ \varphi(y) \to \min_{y \in Y} \right. \tag{3}$$

is equivalent to (1).

The level method [4], as any other cutting plane method, generates a sequence $y_1, \ldots, y_k$ of feasible iterates and uses the respective data, i.e. functionvalues and subgradients, to construct a new point y_{k+1} at the $k+1$-st iteration.

A so-called oracle serves for computing functionvalues and subgradients. Let $y \in Y$ be fixed. Then, by duality theory it holds

$$\varphi(y) = \langle d, y \rangle + \sup_z \{ \langle f - By, z \rangle : A^\tau z \leq c \}. \tag{4}$$

Suppose the oracle uses a simplex method to solve the inner program. Let $z^*(y)$ denote the optimal basic solution. Thus,

$$\varphi(y) = \langle d, y \rangle + \langle f - By, z^*(y) \rangle$$

and

$$g(y) := d - B^\tau z^*(y) \in \partial \varphi(y).$$

Obviously, the feasible set of the inner program (in its dual representation) is independent of the current value of y; thus, it has only a finite number of extreme points z^*. Hence, the oracle may deliver only a finite number of distinct subgradients $d - B^\tau z^*$. That means, some of them must repeat at different iterations. We will show next what consequences are connected with that and how these can be exploited to improve the performance of the algorithm. Let us briefly give the main steps of the level method in its traditional way.

2 Algorithm of the Level Method

Suppose the iterates $y_1, \ldots, y_k \in Y$ and their data $\varphi(y_1), \ldots, \varphi(y_k)$ and $g(y_1), \ldots, g(y_k)$ already computed. Let $\lambda \in (0,1)$ denote the level-parameter. The $k+1$-st iteration is as follows:

Algorithm: Level Method
Step 1: Determine the functionvalue-record

$$\varphi_k^* = \min_{1 \leq i \leq k} \{ \varphi(y_i) \} = \min \{ \varphi_{k-1}^*, \varphi(y_k) \}.$$

Step 2: Compute

$$\Delta_k = - \min_{y \in Y} \max_{1 \leq i \leq k} \left\{ \tfrac{1}{\|g(y_i)\|} \left(\varphi(y_i) + \langle g(y_i), y - y_i \rangle - \varphi_k^* \right) \right\}.$$

If $\Delta_k = 0$ stop.

Step 3: Compute y_{k+1} as projection of y_k onto the relaxed level set

$$Y_k(\lambda) = \left\{ y : \max_{1 \leq i \leq k} \left\{ \tfrac{1}{\|g(y_i)\|} \left(\varphi(y_i) + \langle g(y_i), y - y_i \rangle - \varphi_k^* \right) \leq -\lambda \Delta_k \right\} \right\} \cap Y.$$

Step 4: Request $\varphi(y_{k+1})$ and $g(y_{k+1}) \in \partial\varphi(y_{k+1})$ from the oracle.
Set $k = k + 1$; return to Step 1.

Remarks: 1. Due to normalizing the cutting planes, not the function $\varphi(y)$ itself is approximated but its level sets

$$Y_k = \left\{ y : \max_{1 \le i \le k} \{\varphi(y_i) + \langle g(y_i), y - y_i \rangle \le \varphi_k^*\} \right\}.$$

So, this version is sometimes referred to as level set method instead of level method.

2. By construction, Δ_k is the radius of the largest ball that is contained in Y_k and has center in Y.

3. Δ_k is the optimal value of the linear program

$$\begin{cases} t \to \max\limits_{t,y} \\ \langle g(y_i), y_i - y \rangle - \varphi(y_i) + \varphi_k^* \ge t\|g(y_i)\| \quad \forall i = 1, \dots, k \\ y \in Y \end{cases} \tag{5}$$

whereas computation of y_{k+1} consists in solving the quadratic program

$$\begin{cases} \|y - y_k\|^2 \to \min\limits_{y} \\ \langle g(y_i), y_i - y \rangle - \varphi(y_i) + \varphi_k^* \ge \lambda\Delta_k\|g(y_i)\| \quad \forall i = 1, \dots, k \\ y \in Y. \end{cases} \tag{6}$$

4. If $\lambda = 1$, the level method merges into the well-known Kelley cutting plane method [1] which is in case of linear programming decomposition finite but of exponential order. The level method finds an ε-optimal solution with polynomial complexity [4] and proved better in various tests [3].

3 Recurrent Subgradients

Since in general we may not expect finite termination of the algorithm (i.e., it generates an infinite sequence $y_1, y_2, \dots$ of iterates), and due to the fact that the oracle may deliver only a finite number of distinct subgradients, it will occur that they repeat. Throughout this section we suppose that $z^*(y_{i_1}) = z^*(y_{i_2}) := z^*$ whereat $i_2 > i_1$. Hence, $\varphi(y_{i_1}) = \langle d, y_{i_1} \rangle + \langle f - By_{i_1}, z^* \rangle$, $\varphi(y_{i_2}) = \langle d, y_{i_2} \rangle + \langle f - By_{i_2}, z^* \rangle$, and $g(y_{i_1}) = g(y_{i_2}) = d - B^\tau z^*$. Consequently, in the linear program for computing Δ_{i_2+1} we have for $i = i_1$ as well as for $i = i_2$ the constraint

$$\begin{aligned} t\|d - B^\tau z^*\| &\le \langle g(y_i), y_i - y \rangle - \varphi(y_i) + \varphi_{i_2}^* \\ &= \langle d - B^\tau z^*, y_i - y \rangle - \langle d, y_i \rangle - \langle f - By_i, z^* \rangle + \varphi_{i_2}^* \\ &= \langle B^\tau z^* - d, y \rangle - \langle f, z^* \rangle + \varphi_{i_2}^*, \end{aligned} \tag{7}$$

i.e. two identical constraints. Therefore, it is sensible to use this cognition for a possible improvement of the algorithm. The next observation motivates the changes to be considered in a refined method.

Lemma 1: It holds $\varphi^*_{i_2} \le \varphi^*_{i_2-1} - \lambda\Delta_{i_2-1}\|B^\tau z^* - d\|$.

Proof: Since y_{i_2} is optimal in

$$\begin{cases} \|y - y_{i_2-1}\|^2 \to \min\limits_{y\in Y} \\ \langle g(y_i), y_i - y\rangle - \varphi(y_i) + \varphi^*_{i_2-1} \ge \lambda\Delta_{i_2-1}\|g(y_i)\| \quad \forall\, i \le i_2 - 1, \end{cases}$$

due to (7) it holds for $i = i_1$

$$\langle B^\tau z^* - d, y_{i_2}\rangle - \langle f, z^*\rangle + \varphi^*_{i_2-1} \ge \lambda\Delta_{i_2-1}\|B^\tau z^* - d\|.$$

This leads to

$$\begin{aligned} \varphi^*_{i_2-1} - \varphi(y_{i_2}) &\ge \lambda\Delta_{i_2-1}\|B^\tau z^* - d\| + \langle f, z^*\rangle - \langle B^\tau z^* - d, y_{i_2}\rangle - \langle d, y_{i_2}\rangle - \langle f - By_{i_2}, z^*\rangle \\ &= \lambda\Delta_{i_2-1}\|B^\tau z^* - d\| > 0. \end{aligned}$$

Consequently, $\varphi(y_{i_2}) \le \varphi^*_{i_2-1} - \lambda\Delta_{i_2-1}\|B^\tau z^* - d\| < \varphi^*_{i_2-1}$ and $\varphi^*_{i_2} = \varphi(y_{i_2})$. □

Thus, a repeated subgradient belongs to a reduction in the functionvalue-record which is of the magnitude of the current optimality measure Δ.

Lemma 2: It holds

$$\frac{\Delta_{i_2}}{\Delta_{i_2-1}} \le 1 - \lambda\frac{\|B^\tau z^* - d\|}{\max\limits_{i\le i_2-1}\|g(y_i)\|}.$$

Proof: By definition of Δ_{i_2}, there exists $\bar{y} \in Y$ such that the ball $B_{\Delta_{i_2}}(\bar{y})$ centered in $\bar{y}$ with radius Δ_{i_2} satisfies

$$B_{\Delta_{i_2}}(\bar{y}) \subseteq \{y : \langle g(y_i), y_i - y\rangle - \varphi(y_i) + \varphi^*_{i_2} \ge 0 \quad \forall\, i\},$$

i.e., $\forall\, r : \|r\| \le 1$ it holds

$$\langle g(y_i), y_i - (\bar{y} + \Delta_{i_2} r)\rangle - \varphi(y_i) + \varphi^*_{i_2} \ge 0 \quad \forall\, i \le i_2, \tag{8}$$

which yields

$$\langle g(y_i), y_i - \bar{y}\rangle - \varphi(y_i) + \varphi^*_{i_2} \ge \Delta_{i_2}\|g(y_i)\| \quad \forall\, i \le i_2.$$

Then, for all $i \le i_2 - 1$ it holds due to Lemma 1 and (8)

$$\langle g(y_i), y_i - y\rangle - \varphi(y_i) + \varphi^*_{i_2-1} \ge \Delta_{i_2}\|g(y_i)\| + \lambda\Delta_{i_2-1}\|B^\tau z^* - d\| + \langle g(y_i), \bar{y} - y\rangle.$$

Thus, if $\|y - \bar{y}\| \le \Delta_{i_2} + \lambda\Delta_{i_2-1}\frac{\|B^\tau z^* - d\|}{\max_{i\le i_2-1}\|g(y_i)\|}$, then Cauchy-Schwarz-inequality yields

$$\langle g(y_i), y_i - y\rangle - \varphi(y_i) + \varphi^*_{i_2-1} \ge 0,$$

i.e., the set $\{y : \langle g(y_i), y_i - y\rangle - \varphi(y_i) + \varphi^*_{i_2-1} \ge 0\}$ contains the ball with center $\bar{y}$ and radius $\Delta_{i_2} + \lambda\Delta_{i_2-1}\frac{\|B^\tau z^* - d\|}{\max_{i\le i_2-1}\|g(y_i)\|}$. However, since Δ_{i_2-1} is the radius of the largest ball contained in this set with center in Y it follows $\Delta_{i_2-1} \ge \Delta_{i_2} + \lambda\Delta_{i_2-1}\frac{\|B^\tau z^* - d\|}{\max_{i\le i_2-1}\|g(y_i)\|}$ which is equivalent to

$$\Delta_{i_2} \le \left(1 - \lambda\frac{\|B^\tau z^* - d\|}{\max_{i\le i_2-1}\|g(y_i)\|}\right)\Delta_{i_2-1}.$$

□

This section should be closed with a note on the number N of possible distinct subgradients. In the first section it has been mentioned that this number is bounded by the number of extremal points of the polyedral set $\{z : A^{\tau} z \leq c\}$. Assuming that $A^{\tau} \in \mathbb{R}^{(\tilde{m}+\tilde{n})\times\tilde{n}}$ we immediately get $N \leq \binom{\tilde{m}+\tilde{n}}{\tilde{n}}$.
However, the main concept for dealing with primal decomposition is that the decomposed program splits into a series of independent programs. Assuming that A is blockdiagonal, i.e. $A = \operatorname{diag}(A_1, A_2, \ldots, A_K)$, we have to consider

$$\varphi(y) = \langle d, y\rangle + \sum_{k=1}^{K} \inf_{x^k}\{\langle c^k, x^k\rangle : A_k x^k = (f - By)^k, x^k \geq 0\} \tag{9}$$

resp. its dual representation

$$\varphi(y) = \langle d, y\rangle + \sum_{k=1}^{K} \sup_{z^k}\{\langle (f - By)^k, z^k\rangle : A_k^{\tau} z^k \leq c^k\}. \tag{10}$$

Thus, a subgradient is obtained through

$$g(y) := d - \sum_{k=1}^{K} B^{\tau} z^{k*}(y).$$

Hence, two subgradients $g(y_{i_1})$ and $g(y_{i_2})$ equal each other only in case that in all K subprograms identical basises are optimal at y_{i_1} and y_{i_2}.
At first glance, it seems that the number of different subgradients is increasing through decomposition, yet actually it decreases since the number of linear independent rows in a blockdiagonal matrix is not greater than in a general matrix of the same size.
For simplicity, we assume that all blocks A_k are of the same size: $A_k^{\tau} \in \mathbb{R}^{(m+n)\times n}$. Hence, the ratio of the number of possible basises in the decomposed version to the one with full matrix is

$$\frac{\binom{m+n}{n}^K}{\binom{K(m+n)}{Kn}} \ll 1.$$

We continue with an investigation regarding the effects of subgradients recurrance.

4 Longer Steps

Suppose again that z^* is optimal in $\sup_z\{\langle f - By, z\rangle : A^{\tau} z \leq c\}$ for $y = y_{i_1}$ as well as for $y = y_{i_2}$.

Lemma 3: φ is linear on $[y_{i_1}, y_{i_2}] = \{(1-\mu)y_{i_1} + \mu y_{i_2} : \mu \in [0,1]\}$.

Proof: By the assumption, there exist x_{i_1} and x_{i_2} such that for $i = i_1, i_2$ it holds

$$\begin{cases} Ax_i = f - By_i \\ x_i \geq 0 \\ A^{\tau} z^* \leq c \\ \langle x_i, A^{\tau} z^* - c\rangle = 0. \end{cases}$$

For any $\mu \in [0,1]$ obviously it is

$$\begin{cases} A((1-\mu)x_{i_1} + \mu x_{i_2}) = f - B((1-\mu)y_{i_1} + \mu y_{i_2}) \\ (1-\mu)x_{i_1} + \mu x_{i_2} \geq 0 \\ A^{\tau} z^* \leq c \\ \langle (1-\mu)x_{i_1} + \mu x_{i_2}, A^{\tau} z^* - c\rangle = 0 \end{cases}$$

and by linear programming theory z^* is an optimal solution of (4) for $y = (1-\mu)y_{i_1} + \mu y_{i_2}$. Hence, $\varphi(y) = \langle d, y\rangle + \langle f - By, z^*\rangle = \langle d - B^T z^*, y\rangle + \langle f, z^*\rangle$ for all $y \in [y_{i_1}, y_{i_2}]$. □

Because of $\varphi(y_{i_1}) \geq \varphi^*_{i_1} \geq \varphi^*_{i_2-1} > \varphi^*_{i_2}$ it holds $\langle d - B^T z^*, y_{i_2} - y_{i_1}\rangle < 0$, i.e. the directional derivative $\varphi'(y_{i_1}; y_{i_2} - y_{i_1})$ is negative. Since the level method suggests a search in this direction, too (remember $i_2 > i_1$), it seems sensible to take a longer step as far as the functionvalue decreases. Hence, we aim to solve the one-dimensional search problem

$$\begin{cases} \tilde{\varphi}(s) := \varphi(y_{i_2} + s(y_{i_2} - y_{i_1})) \to \min\limits_s \\ y_{i_2} + s(y_{i_2} - y_{i_1}) \in Y. \end{cases} \tag{11}$$

Due to our assumption the inequality $\varphi(y_{i_2}) < \varphi(y_{i_1})$ is valid; furthermore, φ and henceforth $\tilde{\varphi}$ are convex functions. Since $\tilde{\varphi}'(0;1) = \varphi'(y_{i_1}; y_{i_2} - y_{i_1}) < 0$ and Y is compact, the minimal value of (11) is attained at some $s^* \geq 0$. We describe a method to find this minimum by use of a simplex method.

For fixed y let β denote the optimal basis in

$$\sup_z\{\langle f - By, z\rangle : A^T z \leq c\} = \inf_x\{\langle c, x\rangle : Ax = f - By, x \geq 0\}.$$

By LP-theory, β is optimal for all y which it is primal and dual feasible for. The solutions connected to β are $z^*_\beta = (A^T_\beta)^{-1} c_\beta$ and $x^*_\beta = A^{-1}_\beta (f - By), x^*_{\beta^c} = 0$, where A_β denotes the matrix consisting of the columns of A belonging to β resp. x_β the components of x belonging to β, and β^c denotes the complement of β.

Now consider $y = y_{i_2} + s(y_{i_2} - y_{i_1})$ where β denotes the optimal basis for $y = y_{i_1}$ and $y = y_{i_2}$. Then, β is dual feasible for all $s \in \mathbb{R}$ and primal feasible for all s satisfying

$$x_\beta(s) = A^{-1}_\beta(f - B(y_{i_2} + s(y_{i_2} - y_{i_1})) = A^{-1}_\beta(f - By_{i_2}) - sA^{-1}_\beta B(y_{i_2} - y_{i_1}) \geq 0,$$

that means for

$$\begin{aligned} s \leq s_\beta &= \min\left\{ \frac{(A^{-1}_\beta(f - By_{i_2}))_j}{(A^{-1}_\beta B(y_{i_2} - y_{i_1}))_j} : (A^{-1}_\beta B(y_{i_2} - y_{i_1}))_j > 0 \right\} \\ &= \min\left\{ \frac{\langle (A^{-1}_\beta)_j, f - By_{i_2}\rangle}{\langle (A^{-1}_\beta)_j, B(y_{i_2} - y_{i_1})\rangle} : \langle (A^{-1}_\beta)_j, B(y_{i_2} - y_{i_1})\rangle > 0 \right\}. \end{aligned}$$

Let $s_Y = \max\{s \geq 0 : y_{i_2} + s(y_{i_2} - y_{i_1}) \in Y\} < \infty$. Then, β is optimal for (at least) $s \in [-1, \min\{s_\beta, s_Y\}]$.
If $\min\{s_\beta, s_Y\} = s_Y$ then we replace y_{i_2} by $y_{i_2} + s_Y(y_{i_2} - y_{i_1})$. Otherwise, $x_\beta(s_\beta)$ is primal degenerated, hence there is another basis optimal at $y_{i_2} + s_\beta(y_{i_2} - y_{i_1})$. Let $\tilde{\beta}$ denote any such basis. Then the directional derivative of $\tilde{\varphi}$ is given by

$$\tilde{\varphi}'(s_\beta; 1) = \varphi'(y_{i_2} + s_\beta(y_{i_2} - y_{i_1}); y_{i_2} - y_{i_1}) = \langle d - B^T(A^T_{\tilde{\beta}})^{-1} c_\beta, y_{i_2} - y_{i_1}\rangle.$$

If $\tilde{\varphi}'(s_\beta; 1) < 0$ we do the same considerations as above. This scheme is continued until we find $s^* = s_{\tilde{\beta}}$ such that $\tilde{\varphi}'(s_{\tilde{\beta}}; 1) \geq 0$ or $s^* = s_Y$ springs into action. The final s^* realizes $\min\limits_{s \geq 0}\{\tilde{\varphi}(s) : y_{i_2} + s(y_{i_2} - y_{i_1}) \in Y\}$. Replacing y_{i_2} by $y_{i_2} + s^*(y_{i_2} - y_{i_1})$ we start the next iteration of the level method.
Although this one-dimensional minimization may end at some basis which was optimal in a preceding iteration we do not suggest to do the same scheme again since the overall algorithm, namely the level method, gave no hint for doing that (besides, this strategy would degenerate to Kelley's method).

5 The Conceptual Refined Algorithm

In this section we describe the new features of a refined level method version which makes use of the considerations above. Again, we suppose that for $i = 1, \ldots, k$ the data y_i, $\varphi(y_i)$, $g(y_i)$ are known. In addition, let β_i denote some basis optimal in (2) for $y = y_i$.
In the algorithm of Section 2, Steps 3 and 4 are replaced by

Step 3': Compute $\hat{y}$ as projection of y_k onto the relaxed level set

$$Y_k(\lambda) = \left\{ y : \max_{1 \le i \le k} \left\{ \tfrac{1}{\|g(y_i)\|} \left(\varphi(y_i) + \langle g(y_i), y - y_i \rangle - \varphi_k^* \right) \le -\lambda \Delta_k \right\} \right\} \cap Y.$$

Step 4': Request $\varphi(\hat{y})$, $g(\hat{y}) \in \partial\varphi(\hat{y})$, and $\beta(\hat{y})$ optimal in (2) for $y = \hat{y}$ from the oracle.
Compare $\beta(\hat{y})$ with $\beta_k, \beta_{k-1}, \ldots, \beta_1$.
If $\beta(\hat{y}) \notin \{\beta_k, \beta_{k-1}, \ldots, \beta_1\}$, set $y_{k+1} = \hat{y}$.
Else, i.e. if $\beta(\hat{y}) = \beta_j$ for some $j \in \{k, \ldots, 1\}$, compute the optimal solution s^* of

$$\varphi(\hat{y} + s(\hat{y} - y_j)) \to \min_{s \ge 0,\, \hat{y} + s(\hat{y} - y_j) \in Y},$$

set $y_{k+1} = \hat{y} + s^*(\hat{y} - y_j)$.
Recall the oracle to gather $\varphi(y_{k+1}), g(y_{k+1}), \beta_{k+1}$.
Set $k = k + 1$ and return to Step 1.

It should be added that the technique described in Section 4 computes not only the optimal value s^* but all values (if any) at which the optimal basis of (2) changes along the ray $\hat{y} + s(\hat{y} - y_j)$. If at these kinks the oracle is accessorily called, the additional data may be used to further improve the approximation of Y^*.

6 Further Applications

Note that the objective function introduced by (2) is a piecewise linear function, and the considerations concerning longer steps – and consequently an acceleration of the level method – rely only on this piecewise linearity. Hence, the same target may be pursued if the primary problem consists in minimizing a piecewise linear convex function, which may be expressed as a finite max-function. Thus, all statements are transferable to

$$\begin{cases} \varphi(y) = \max_{p=1,\ldots,l} \{ \langle a_p, y \rangle + b_p \} \to \min_y \\ y \in Y = \{ y \in \mathbb{R}^m : -u \le y \le u \}. \end{cases} \tag{12}$$

Subgradients of φ are the vectors a_p (and in case that at some point y the max in (12) is attained at several indices $p_1, \ldots, p_q$, the subdifferential is the convex hull conv $\{a_{p_1}, \ldots, a_{p_q}\}$). However, there is only a finite number of extremal subgradients, namely $a_1, \ldots, a_l$ and recurrance is inevitable.

In our method, Step 4' of the foregoing section has to be replaced by

Step 4": Request $\varphi(\hat{y})$ and the active index $\hat{p}$ for $y = \hat{y}$ from the oracle.
Compare $\hat{p}$ with $p_k, p_{k-1}, \ldots, p_1$.
If $\hat{p} \notin \{p_k, p_{k-1}, \ldots, p_1\}$, set $y_{k+1} = \hat{y}$.
Else, i.e. if $\hat{p} = p_j$ for some $j \in \{k, \ldots, 1\}$, compute the optimal solution s^* of

$$\varphi(\hat{y} + s(\hat{y} - y_j)) \to \min_{s \ge 0,\, \hat{y} + s(\hat{y} - y_j) \in Y},$$

set $y_{k+1} = \hat{y} + s^*(\hat{y} - y_j)$.
Recall the oracle to gather $\varphi(y_{k+1}), p_{k+1}$.
Set $k = k + 1$ and return to Step 1.

In the following we describe how the main cause for a possible acceleration, namely subgradient recurrance, may be exploited even in nonlinear cases, i.e. for a general convex function φ. In this case, we may not expect exact repeating of some previously computed subgradient. However, it may happen that for a pair of indices $i_1 < i_2$ and some small $\varepsilon > 0$ it holds

$$\|g(y_{i_1}) - g(y_{i_2})\| < \varepsilon.$$

We will give a sufficient condition on the value of ε such that again $\varphi^*_{i_2} < \varphi^*_{i_1}$.

By construction, y_{i_2} is optimal in

$$\begin{cases} \|y - y_{i_2-1}\|^2 \to \min\limits_{y \in Y} \\ \langle g(y_i), y_i - y\rangle - \varphi(y_i) + \varphi^*_{i_2-1} \geq \lambda\Delta_{i_2-1}\|g(y_i)\| \quad \forall\, i \leq i_2 - 1. \end{cases}$$

Particularly, for $i = i_1$ it holds

$$\langle g(y_{i_1}), y_{i_1} - y_{i_2}\rangle - \varphi(y_{i_1}) + \varphi^*_{i_2-1} \geq \lambda\Delta_{i_2-1}\|g(y_{i_1})\|,$$

which together with the subgradient inequality

$$\varphi(y_{i_1}) \geq \varphi(y_{i_2}) + \langle \varphi(y_{i_2}), y_{i_1} - y_{i_2}\rangle$$

leads to

$$\begin{aligned} \varphi^*_{i_2-1} - \varphi(y_{i_2}) &\geq \lambda\Delta_{i_2-1}\|g(y_{i_1})\| + \langle g(y_{i_2}) - g(y_{i_1}), y_{i_1} - y_{i_2}\rangle \\ &\geq \lambda\Delta_{i_2-1}\|g(y_{i_1})\| - \|g(y_{i_2}) - g(y_{i_1})\|\|y_{i_1} - y_{i_2}\|. \end{aligned}$$

This difference is positive (and hence $\varphi^*_{i_2} < \varphi^*_{i_1}$) if $\frac{\|g(y_{i_2}) - g(y_{i_1})\|}{\|g(y_{i_1})\|} < \frac{\lambda}{D}\Delta_{i_2-1}$ where D denotes the diameter of Y.

Summarizing, if two subgradients are close to each other (compared to the size of the diameter Δ of the current approximation of the localization set), then the latest trial point is the best and again, a longer step in the direction given by these two trial points may cause a further decrease in the functionvalue record.

However, in contrast to the linear case, computation of an optimal steplength requires much more expenditure. A longer step (with a steplength suitably chosen) is therefore recommended mainly to gather more information. After all, even detecting the recurrance of a similarly computed subgradient requires more effort than in the linear case.

7 Computational Results

The above described longstep version of the level method has been tested for max-functions like given in (12) against the traditional level method described above. We solved series of 50 randomly generated problems each for different values of m (dimension of the variable y) and l (number of linear functions contributing to (12)).

The first table gives the average number of iterations needed to end in $\Delta < 10^{-5}$ in the original method (first row) resp. in the longstep version (second row, slanted):

l \ m	10	20	40
10	26.36 *15.46*	25.86 *17.78*	27.32 *17.40*
20	31.40 *26.56*	32.48 *26.50*	33.88 *28.92*
40	36.92 *35.74*	51.42 *43.12*	49.40 *49.06*

Number of Iterations: Original Level Method vs. *Longstep Level Method*

The next table gives the computing time for each series of 50 test instances in minutes:

l \ m	10	20	40
10	2:20 **1:06**	4:12 **2:10**	10:53 **6:20**
20	3:02 **2:15**	5:30 **3:49**	13:27 **9:13**
40	3:27 **3:02**	10:22 **8:40**	20:20 **18:50**

Computation Time: Original Level Method vs. **Longstep Level Method**

The last table summarizes the percentage of number of iterations (first row; slanted) resp. computation time (second row; boldface) of the longstep version compared to the original version:

l \ m	10	20	40
10	*58.6* **47.1**	*68.8* **51.6**	*63.7* **58.2**
20	*84.6* **74.2**	*81.6* **69.4**	*85.4* **68.5**
40	*96.8* **87.9**	*83.9* **83.6**	*99.3* **92.6**

Percentage of *average number of iterations* resp. **computation time** in the longstep version (original version =100%)

The results confirm that on average the longstep version beats the original one. The advantage is especially significant the smaller the ratio $\frac{l}{m}$. This seems to be natural since small values of l mean fewer different subgradients; hence, the effects of recurrent subgradients are stronger in these cases. For $m = l$, there is nearly no difference between the two versions. In these instances, there were nearly no recurrent subgradients observed. The few ones taken resulted mainly in very small stepsizes.

In the foregoing investigations we considered the minimization of convex piecewise-linear functions, hence, determination of functionvalues resp. subgradients consumed compared to computing the optimality gap Δ resp. a new trial point $\hat{y}$ just a small part of the computation time. It could be expected that in case of linear programming decomposition the comparison concerning computation time is even more distinct in favour of the longstep version.

Computation time could in both versions be reduced through a strategy of selection or aggregation of cuts as described in [2]. However, this was not part of our considerations.

Another aspect should be mentioned. As was said above, we used an optimality gap of 10^{-5} as stopping criterion. Most of the longstep test problems however ended in a gap of 10^{-12} to 10^{-24} which was caused by a last longstep. This gives reason to the assumption that the final iteration computes actually the minimal point (i.e., a Kelley step is taken at the end).

Literature

[1] J.E. Kelley: The cutting-plane method for solving convex programs; J. Soc. Ind. Appl. Math. 8 (1961); pp. 703-712.

[2] K. Kiwiel: Proximal level bundle methods for convex nondifferentiable optimization, saddle-point problems and variational inequalities; Math. Program. 69 (1995); pp. 89-109

[3] C. Lemaréchal: Numerical experiments in nonsmooth optimization; IIASA Collab. Proc. Ser. CP-82-S8 (1982), pp. 61-84.

[4] C. Lemaréchal, A. Nemirovskii, Y. Nesterov: New Variants of Bundle Methods; Math. Program. 69 (1995); pp. 111-147.