Technical Report: TR-96-17, December 01, 1997

Applying Bundle Methods to the Optimization of Polyhedral Functions: An Applications-Oriented Development

Carraresi Paolo Frangioni Antonio Nonato Maddalena

Dipartimento di Informatica, Università di Pisa
corso Italia 40, 56125 Pisa, Italy

Abstract

Among practical methods for large-scale for Nondifferentiable Optimization (NDO), Bundle methods are widely recognized to play a relevant role; despite their potential, however, they are not often utilized for maximization of polyhedral functions, that appears in many different context such as Lagrangean Duals and decomposition algorithms, since simpler-to-program but less efficient approaches like subgradient methods are preferred. The aim of this work is to provide an applications-oriented survey of the theory of Bundle methods when applied to problems arising in continuous and combinatorial optimization, with an introduction to the several variants of Bundle approaches that can be built up by using a limited set of basic concepts and tools.

ACM–CR Subject Classification:

Keyword and phrases: Nondifferentiable Optimization, Bundle methods, Lagrangean Duals

During the last 20 years, Bundle methods for Nondifferentiable Optimization (NDO) have become an important research area: stemming from the pioneering works of Lemarechal [Le75] [Le77], a large number of papers and books devoted to "classical" Bundle methods have appeared, either for the unconstrained [Le89] [Ki90] [Le91] and/or constrained [LSB81] [Ki85a] [Ki85b] [BNS87] [Ki89] [Ki91a] [SZ92] [HL93] [Mo94] maximization, along with several variants [Mi82] [GM91] [Ki91c] [Mi91] [GM92] [Ki95] [LNN95]. It is nowadays widely accepted that Bundle methods are generally preferable to older subgradient algorithms [HWC74] [Po77], that however still receive some attention [KU93], and to conjugate subgradient methods [Wo75]: from the theoretical viewpoint, they also have nice and interesting connections with the emerging field of proximal point methods [GHV92] [GHV93] [IST94] [BGL95], and with smooth regularizations of nonsmooth functions [KR92] (see [HL93] for an up-do-date review of all such questions). From the applicative point of view, the use of Nondifferentiable Optimization approaches to tackle Lagrangean Duals and decomposition approaches [Ge69] is of well-known importance: among the many examples, we report the use of Dual Ascent approaches as upper bounding procedures for hard combinatorial problems [Gr72] [FNS75] [ZTT84] [Ba93] [BS93] [CGN95] [EGM95], and decomposition approaches to large-scale structured Linear Programs especially in connection with Multicommodity Flow Problems [Me90] [KR92] [LGM93] [Me94] [FG96].

Despite their known superiority, Bundle methods are not always preferred to other older and slower NDO methods, most notably subgradient methods: we claim that this is mainly due to the lack of reliable and flexible Bundle codes; this is in turn caused by the higher programming complexity of such an approach vs. the astounding simplicity of a subgradient code, that is much easier to write and modify in order to cope with the different needs arising in practice when used as a tool within larger projects. The aim of this paper is to present an applications-oriented survey of the theory of Bundle methods, to help practitioners and non-NDO experts to develop and write their own codes for NDO problems arising in practice within the field of combinatorial optimization: such problems are mainly characterized by the fact that nondifferentiable functions to be maximized are in fact polyhedral, thus simplifying some of the issues related to Bundle algorithm (most notably convergence and finite termination), but at the same time opening the interesting question of whether Bundle algorithms explicitly designed for this class of problems may perform better than general ones; after the presentation of the relevant theory in the field, we will show how a variety of a (possibly specialized) Bundle approaches can be constructed from a limited set of basic ideas and algorithmic tools.
Throughout the paper, notations will be those usually adopted: symbols will be written in *italic*, vectors are intended as columns and will be written in **bold**, scalar product between $\boldsymbol{v}$ and $\boldsymbol{w}$ will be denoted by $\boldsymbol{vw}$ or sometimes by $\boldsymbol{v}\cdot\boldsymbol{w}$, $\|\cdot\|$ will indicate the Euclidean norm.

1. Some theory of Bundle methods: the dual viewpoint

We start presenting some of the basic theory underlying Bundle methods: although almost all this material has appeared in some form in the literature in the past years, we think that a self-contained, applications-oriented resume of all of the following facts and their implications is still missing. To begin, let's illustrate the "classical" NDO view of Bundle methods: we are concerned about the numerical solution of the problem (that, for reasons to become clear in the following, we consider of

"dual" nature)

(D) $\max_y \{ \varphi(\boldsymbol{y}) : \boldsymbol{y} \in Y \}$

where $\varphi() : \Re^m \rightarrow \Re \cup [-\infty]$ is a continuous nondifferentiable concave function and $Y \subseteq \Re^m$.
Usually, some technical assumptions have to be made on $\varphi()$ (upper semicontinuity, $\varphi()$ not uniformly identical to $-\infty$) and Y (compactness), but in the cases of our interest $\varphi()$ will be a polyhedral function and Y some "simple" subset of $\Re^m$ such as a polyhedron, the nonnegative orthant or the whole $\Re^m$, so that we won't pay much attention to this sort of details: for sake of simplicity, in the first paragraphs we will restrict our attention to the case of $Y \equiv \Re^m$.
Theoretically speaking, nondifferentiable convex function appears not to be much harder to maximize than differentiable ones: in spite of the *gradient* $\nabla\varphi(\bar{\boldsymbol{y}})$, one just has to define the *subdifferential* $\partial\varphi(\bar{\boldsymbol{y}})$, as the set of all the *subgradients* $\boldsymbol{g}$ of $\varphi(\bar{\boldsymbol{y}})$ in $\bar{\boldsymbol{y}}$, where for now (we will describe later in detail their "primal" interpretation) a subgradient is just (the gradient of) a *supporting hyperplane* of the epigraph of $\varphi()$ in $\bar{\boldsymbol{y}}$ (Figure 1), i.e. such that

$$\varphi(\boldsymbol{y}) \leq \varphi(\bar{\boldsymbol{y}}) + \boldsymbol{g}\cdot(\bar{\boldsymbol{y}} - \boldsymbol{y}) \qquad \forall\, \boldsymbol{y}$$

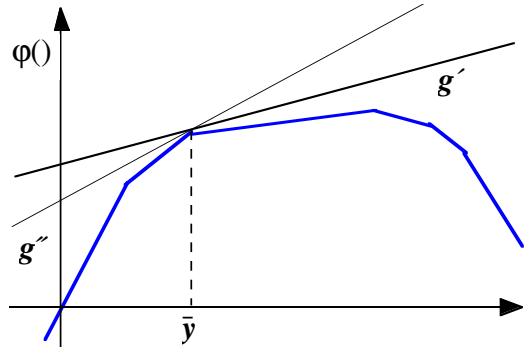

Figure 1: two subgradients $\boldsymbol{g}'$ and $\boldsymbol{g}''$ in a kink point

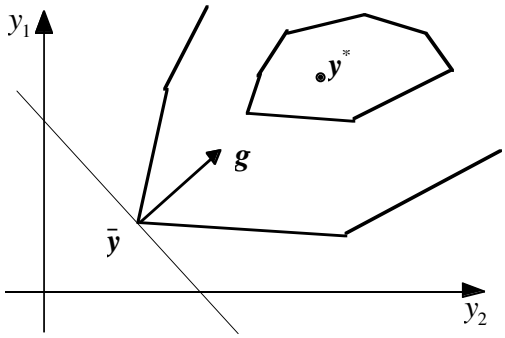

Figure 2: contour plots of $\varphi()$ and it optimal set $\boldsymbol{y}^*$

In other words, any subgradient $\boldsymbol{g}$ in $\bar{\boldsymbol{y}}$ has the property of $\nabla\varphi(\bar{\boldsymbol{y}})$, when existent, that the level set $\{ \boldsymbol{y} : \varphi(\boldsymbol{y}) \geq \varphi(\bar{\boldsymbol{y}}) \}$, and therefore the set of optimal solutions of (D) lies in the half-space pointed by $\boldsymbol{g}$ w.r.t. $\bar{\boldsymbol{y}}$ (Figure 2); actually, the point-to-set mapping $\partial\varphi(\bar{\boldsymbol{y}})$ generalizes the point-to-point mapping $\nabla\varphi(\bar{\boldsymbol{y}})$ under many aspects, some of the most important being listed below (for a proof see [Le77]):

- if $\varphi()$ is differentiable in $\bar{\boldsymbol{y}}$, then $\partial\varphi(\bar{\boldsymbol{y}}) = \{ \nabla\varphi(\bar{\boldsymbol{y}}) \}$
- concavity of $\varphi()$ implies the *existence* of the directional derivative $\varphi'(\bar{\boldsymbol{y}}, \boldsymbol{d})$ at any point $\bar{\boldsymbol{y}}$ and along any direction $\boldsymbol{d}$ (with unitary norm), plus *closedness* and *convexity* of $\partial\varphi(\bar{\boldsymbol{y}})$; then, the relation

  $$\varphi'(\bar{\boldsymbol{y}}, \boldsymbol{d}) = \nabla\varphi(\bar{\boldsymbol{y}})\cdot\boldsymbol{d}$$

  that is valid if $\varphi()$ is differentiable in $\bar{\boldsymbol{y}}$, extends to

  $$\varphi'(\bar{\boldsymbol{y}}, \boldsymbol{d}) = \min_v \{ \boldsymbol{v}\cdot\boldsymbol{d} : \boldsymbol{v} \in \partial\varphi(\bar{\boldsymbol{y}}) \}$$

- $\nabla\varphi(\bar{\boldsymbol{y}})$ (when existent) is the steepest ascent direction for $\varphi()$ in $\bar{\boldsymbol{y}}$, i.e.

  $$\nabla\varphi(\bar{\boldsymbol{y}}) = \operatorname{argmax}_d \{ \varphi'(\bar{\boldsymbol{y}}, \boldsymbol{d}) : \| \boldsymbol{d} \| = 1 \}$$

  but just using the facts listed at the previous point, one can show that more in general

$$\text{argmin}_g \{ \| g \| : g \in \partial\varphi(\bar{y}) \} = \text{argmax}_d \{ \varphi'(\bar{y}, d) : \| d \| = 1 \}$$

- finally, from the above relations it can be easily seen that $\bar{y}$ is optimal for $\varphi() \Leftrightarrow \mathbf{0} \in \partial\varphi(\bar{y}) \Leftrightarrow$

$$0 = \min_g \{ \| g \| : g \in \partial\varphi(\bar{y}) \}$$

In other words, first-order information about $\varphi()$ in $\bar{y}$ of (apparently) the same "strength" than the gradient can be obtained at the "only" cost of solving a convex optimization subproblem (*steepest ascent direction finding*): however, in most practical cases it is assumed that $\partial\varphi(\bar{y})$ is impossible or too costly to be calculated, so that $\varphi()$ is only known by means of a "black box" solver that inputs a point $\bar{y}$ and outputs the value of $\varphi(\bar{y})$ together with just *any subgradient* $g(\bar{y})$ of $\varphi()$ in $\bar{y}$.
This assumption makes things more difficult: it can be easily shown that any algorithm that works just by taking into account one subgradient at a time essentially lacks of both a guaranteed ascent direction (Figure 3) and an implementable stopping criterion. In fact, if $\varphi()$ is not differentiable in its optimum y^* (as it is almost always the case) the solver is not guaranteed to report $g(\bar{y}) = \mathbf{0}$ even for $\bar{y} = y^*$: some different theory is needed for the purpose.
The basic step is to further extend the concept of subgradient to that of *ε-subgradient* (of φ () in $\bar{y}$)

$$\varphi(y) \le \varphi(\bar{y}) + g \cdot (\bar{y} - y) + \varepsilon \qquad \forall\, y$$

(i.e. the gradient of any supporting hyperplane g of the epigraph of $\varphi()$ in $(\bar{y}, \varphi(\bar{y}) + \varepsilon)$, Figure 4) and $\partial\varphi(\bar{y})$ to the corresponding *ε-subdifferential* set $\partial_\varepsilon\varphi(\bar{y})$: all the previous results can be seen as the special case for $\varepsilon = 0$ of an even more general set of results, based on the notions of *ε-directional derivative* and *ε-ascent direction* (for $\varphi()$ in $\bar{y}$) that we are going to sketch in the following.

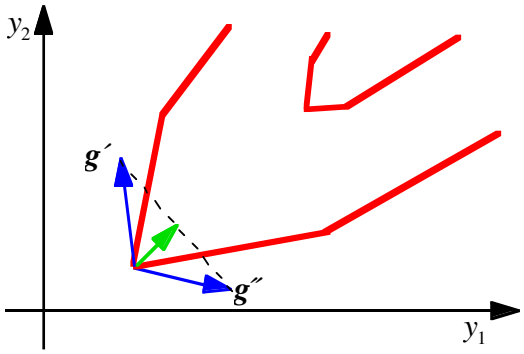

Figure 3: neither g' nor g'' are of ascent, while their minimum-norm convex combination is

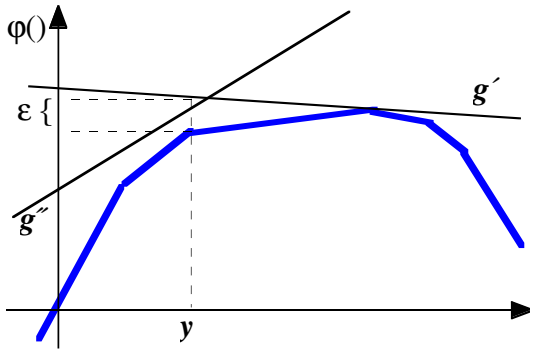

Figure 4: two ε-subgradients for $\varphi()$ in $\bar{y}$: g' gives the ε-directional derivative in $\bar{y}$

F

v is an *ε-ascent direction* for $\varphi()$ in $\bar{y}$ *iif* it permits (in theory) an ascent strictly larger than ε, i.e.

$$\exists t > 0 \text{ s.t. } \varphi(\bar{y} + tv) - \varphi(\bar{y}) > \varepsilon$$

A strictly related concept is that of *ε-directional derivative* of $\varphi()$ in $\bar{y}$ along d, that is, roughly speaking, the slope of the steepest line contained in the restriction of $\varphi()$ along d, passing from $(\bar{y}, \varphi(\bar{y}) + \varepsilon)$ and touching the epigraph of $\varphi()$ somewhere (Figure 4): more formally

$$\varphi'_\varepsilon(\bar{y}, d) = \sup_{t>0} \frac{\varphi(\bar{y} + tv) - \varphi(\bar{y}) - \varepsilon}{t}$$

(the formula justifies for the name), and it can be easily seen that d *is of ε-ascent iif* $\varphi'_\varepsilon(\bar{y}, d) > 0$.

Moreover, all the previously obtained results about subgradients gracefully extend to the new case

- $\varphi'_\varepsilon(\bar{y},\boldsymbol{d}) = \min_v\{\ \boldsymbol{v}\cdot\boldsymbol{d} : \boldsymbol{v} \in \partial_\varepsilon\varphi(\bar{y})\ \}$
- $\operatorname{argmin}_g\{\ \|\boldsymbol{g}\| : \boldsymbol{g} \in \partial_\varepsilon\varphi(\bar{y})\ \} = \operatorname{argmax}_d\{\ \varphi'_\varepsilon(\bar{y},\boldsymbol{d}) : \|\boldsymbol{d}\| = 1\ \}$
- $\bar{y}$ is ε-optimal for $\varphi() \Leftrightarrow \mathbf{0} \in \partial_\varepsilon\varphi(\bar{y}) \Leftrightarrow 0 = \min_g\{\ \|\boldsymbol{g}\| : \boldsymbol{g} \in \partial_\varepsilon\varphi(\bar{y})\ \}$

while ε-subgradients also have the so-called *information transport property*

$$\boldsymbol{g} \in \partial_\varepsilon\varphi(\bar{y}) \quad \Rightarrow \quad \boldsymbol{g} \in \partial_\alpha\varphi(\bar{y}) \qquad \forall\, \boldsymbol{y} \quad \text{and} \quad \alpha \geq \varphi(\bar{y}) + \boldsymbol{g}\cdot(\boldsymbol{y} - \bar{y}) - \varphi(\boldsymbol{y}) + \varepsilon$$

In plain words, a vector $\boldsymbol{g}$ that belongs to the ε-subdifferential of $\varphi()$ in some point $\bar{y}$ can be easily shown to belong to the α-subdifferential of $\varphi()$ in any other point $\boldsymbol{y}$, where $\alpha \geq 0$ (known as *linearization error*) can be calculated from $\boldsymbol{g}$ and $\varphi(\boldsymbol{y})$; hence, the first-order information about $\varphi()$ in $\bar{y}$ that we have at hand can be "translated" into information about the behavior of the function in a completely unrelated point $\boldsymbol{y}$, at the cost of "deteriorating" it of a known measure α. This naturally leads to the idea of Bundle methods, that are mainly characterized by keeping the first-order information about $\varphi()$ in a "disaggregated" form, in contrast with aggregated subgradient algorithms: in any such method, a (finite) sequence of points $\{\ \boldsymbol{y}_i\ \}$ is visited, and a subset β (Bundle) of the whole set of informations $\{\ <\ \boldsymbol{y}_i\ ,\ \varphi(\boldsymbol{y}_i)\ ,\ \boldsymbol{g}(\boldsymbol{y}_i)\ >\ \}$ that represents the "history" of the computations performed so far is used to compute a *tentative ascent direction* $\boldsymbol{d}$ along which the next trial point is choosen. Many known algorithms fall within this class, at least if such a mild definition is used, such as the standard subgradient method in which always $|\beta| = 1$, i.e. the latest subgradient is used as the trial direction: more interesting for what follows is that the *Cutting Plane method* can be seen as a Bundle method (in fact, we will show later on that Bundle algorithms can be seen as a modified CP).
The function

$$\varphi_{CP}(\boldsymbol{y}) = \min_i\{\ \varphi(\boldsymbol{y}_i)\ + \boldsymbol{g}(\boldsymbol{y}_i)\cdot(\boldsymbol{y} - \boldsymbol{y}_i) : i \in \beta\ \}$$

that is usually called *cutting plane model* (Figure 6) is a polyhedral upper approximation of $\varphi()$, formed of a (small) subset of its facets, that is tight at least in all the points $\boldsymbol{y}_i$.

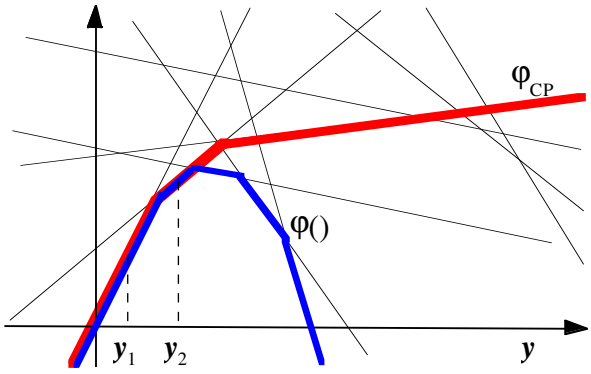

Figure 6: $\varphi_{CP}()$ is an upper approximation of $\varphi()$

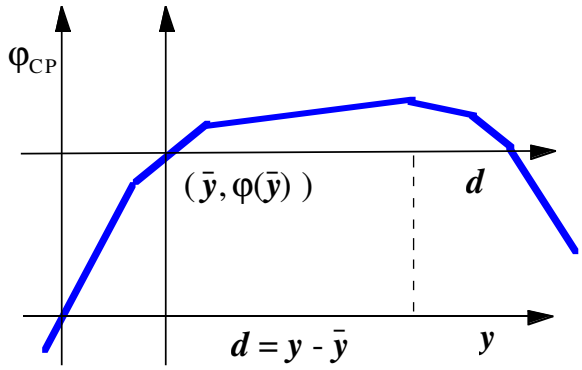

Figure 7: $\varphi_{CP}(\boldsymbol{d})$ is just a double translation of $\varphi_{CP}(\boldsymbol{y})$

Anticipating some standard terminology that will be useful a little later, we note that if any particular point $\bar{y}$ in the dual space is designated as the *current point*, the above formula can be rewritten as

$$\varphi_{CP}(\boldsymbol{d}) = \min_i\{\ \alpha_i + \boldsymbol{g}_i\cdot\boldsymbol{d} : i \in \beta\ \} + \varphi(\bar{y})$$

(Figure 7) where $\boldsymbol{g}_i = \boldsymbol{g}(\boldsymbol{y}_i)$, $\alpha_i = \varphi(\boldsymbol{y}_i) + \boldsymbol{g}(\boldsymbol{y}_i)\cdot(\bar{y} - \boldsymbol{y}_i) - \varphi(\bar{y}) \geq 0$ and $\boldsymbol{d}$ (not necessarily of unitary

norm) is to be interpreted as the displacement from the current point $\bar{\boldsymbol{y}}$; hence, a natural idea is to find the optimal solution $\boldsymbol{d}^*$ of the linear programming subproblem (that has a limited number of constraints, and therefore can be solved efficiently)

$$(D_\beta) \qquad \max_{\boldsymbol{d}}\{\ \varphi_{CP}(\boldsymbol{d})\ \} + \varphi(\bar{\boldsymbol{y}}) = \max_{d,v}\{\ v : v \le \boldsymbol{g}_i \cdot \boldsymbol{d} + \alpha_i \ \ i \in \beta\ \} + \varphi(\bar{\boldsymbol{y}})$$

and to use $\boldsymbol{y} = \bar{\boldsymbol{y}} + \boldsymbol{d}^*$ as the next trial point in which to evaluate the function: if $\varphi_{CP}(\boldsymbol{d}^*) = \varphi(\bar{\boldsymbol{y}} + \boldsymbol{d}^*)$ then $\boldsymbol{y}$ is optimal for the entire problem (D), otherwise a new cutting plane $\boldsymbol{g}(\boldsymbol{y})$ has been generated and can be added to β to obtain a better CP model. This simple approach suffers from a number of theoretical and practical problems: barring the fact that finite termination can be ensured for polyhedral functions only, since general nondifferentiable functions can be expressed as linear programs only at the cost of introducing an infinite number of constraints, cutting plane approaches are known to have a bad practical behavior; this is generally due to the fact that, following the vertices of (some approximations of) the epigraph of $\varphi()$, the trajectory followed by the current point is "unstable", and rather than "pointing towards" the true optimum it keeps "circling around" it for some time: this is especially true in earlier stages of the algorithm's operations, where (D_β) has *not* a finite solution, so that artificial (usually box) constraints have to be added in order to obtain a new point $\boldsymbol{y}$, that often turns out to be unreasonably "far away".

Although actually motivated by different reasons, that will be extensively presented later on, the other "standard component" of a Bundle algorithm can be also presented as a way to reduce the abovementioned "instability" of CP methods: rather than solving (D_β), the tentative ascent direction $\boldsymbol{d}$ is taken to be the optimal solution of the following Quadratic Problem

$$(\Pi_{\beta t}) \qquad \max_{\boldsymbol{d},v}\{\ v - 1/(2\,t)\|\,\boldsymbol{d}\,\|^2 : v \le \boldsymbol{g}_i \cdot \boldsymbol{d} + \alpha_i \ \ i \in \beta\ \}$$

that is in fact (D_β) (disregarding the constant term $\varphi(\bar{\boldsymbol{y}})$) plus a simple quadratic *penalty term* (sometimes called *stabilizing term*) that is introduced to discourage directions leading to "far away" points where $\varphi_{CP}()$ is probably a "bad" approximation of $\varphi()$. From this viewpoint, the *trust region parameter* t measures how far from the current point $\bar{\boldsymbol{y}}$ we "trust" our model $\varphi_{CP}()$, somehow resembling the *trust radius* of trust region methods for smooth optimization: note that, for sufficiently large values of t, $(\Pi_{\beta t})$ is equivalent to the "naive" (D_β) (Figure 8), provided that the latter problem has finite maximum (which is very often *not* the case), or otherwise phrased the (possibly unbounded) optimal solution of (D_β) is just the limit for $t \to +\infty$ of the optimal solution of $(\Pi_{\beta t})$.

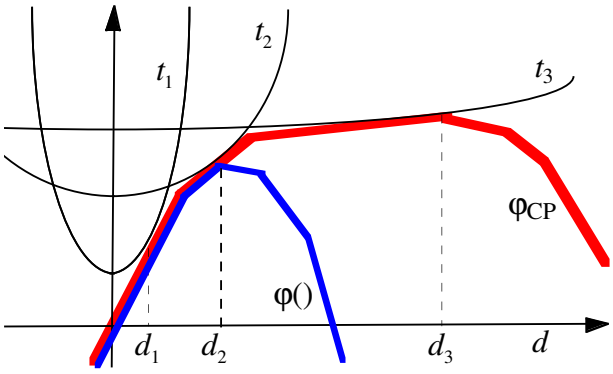

Figure 8: effect of the stabilizing term for $t_1 < t_2 < t_3$ - d_i are the optimal solutions

The role of t as "distance" parameter can be made apparent by considering the Quadratic Dual of $(\Pi_{\beta t})$, i.e.

$$(\Delta_{\beta t}) \qquad \min_\theta \{ (1/2)t \| \Sigma_i \boldsymbol{g}_i\theta_i \|^2 + \Sigma_i \alpha_i\theta_i : \theta \in \Theta \}$$

where $\Theta = \{ \theta : \Sigma_i \theta_i = 1 , \theta \geq \mathbf{0} \}$ is the unitary simplex in the $|\beta|$-dimensional space of dual multipliers: being θ^* the optimal solution of $(\Delta_{\beta t})$, $\boldsymbol{d}^*$ and v^* can be found as

$$\boldsymbol{d}^* = t\boldsymbol{z} \qquad \text{where} \qquad \boldsymbol{z} = \boldsymbol{z}_{\theta^*} = \Sigma_i \boldsymbol{g}_i\theta_i^*$$

$$v^* = t\| \boldsymbol{z} \|^2 + \sigma \qquad \text{where} \qquad \sigma = \sigma_{\theta^*} = \Sigma_i \alpha_i\theta_i^*$$

i.e. the new tentative point $\boldsymbol{y}$ can be found along *direction* $\boldsymbol{z}$ (a convex combination of the currently available subgradients), using t as *predefinite stepsize*; however, Bundle methods are *not* pure trust region algorithms for several reasons, the simplest being that t is not a trust radius since $\boldsymbol{z}$ is not in principle of unitary norm, and $\| \boldsymbol{z} \|$ depends on t in a "complicated" way so that the actual displacement $t\| \boldsymbol{z} \|$ is only known *a posteriori*.

Problem $(\Delta_{\beta t})$ has emerged from the very beginning [Le77] as the best candidate for direction finding subproblem within Bundle methods: in our opinion, this is mainly due to the fact that it can be viewed from a surprisingly high number of different viewpoints, each one shedding new light on some of its properties and possibly suggesting new ways of approaching some of the open questions in the development of a good overall algorithm; in fact, even though different problems have also been proposed for the same purpose, almost all of them are just minor modifications of $(\Delta_{\beta t})$ (for instance with t fixed to 1 or with $\| \boldsymbol{Hd} \|$ in the objective function for some simple scaling matrix $\boldsymbol{H}$), and they must "refer" to one or another of the "interpretations" of $(\Delta_{\beta t})$ to be justified. Actually, in the original development $(\Delta_{\beta t})$ was introduced before $(\Pi_{\beta t})$, and was motivated by some of the above ε-subgradients theory: to see the connection, we shall look at $(\Delta_{\beta t})$ as the Lagrangean relaxation of

$$(\Delta_\beta^\varepsilon) \qquad \min_\theta \{ (1/2) \| \Sigma_i \boldsymbol{g}_i\theta_i \|^2 : \Sigma_i \alpha_i\theta_i \leq \varepsilon , \theta \in \Theta \}$$

where a Lagrangean multiplier $1/t$ has been used to dualize the constraint $\Sigma_i \alpha_i\theta_i \leq \varepsilon$; such a relaxation

$$\min_\theta \{ (1/2) \| \Sigma_i \boldsymbol{g}_i\theta_i \|^2 + (1/t)\Sigma_i \alpha_i\theta_i : \theta \in \Theta \} \quad [- \varepsilon t]$$

actually differs from $(\Delta_{\beta t})$ for a multiplicative factor of t in the objective function, but it is easy to see (just writing the Khunn-Tucker conditions) that the two problems are essentially the same, with the precise meaning that they have the same optimal primal solution θ^* and the respective dual optimal solutions are scaled of the same factor of t.

The point is that $(\Delta_\beta^\varepsilon)$ can be in turn interpreted as a "smoothing" of

$$\min_{\boldsymbol{g}} \{ \| \boldsymbol{g} \| : \boldsymbol{g} \in \text{conv}(\{ \boldsymbol{g}_i : \alpha_i \leq \varepsilon \}) \}$$

that needs few comments once one realizes that $\text{conv}(\{ \boldsymbol{g}_i : \alpha_i \leq \varepsilon \}) \subseteq \partial_\varepsilon\varphi(\bar{\boldsymbol{y}})$; hence, with the language of NDO theory, $(\Delta_\beta^\varepsilon)$ is a modification of the problem of finding the minimum norm vector in (an inner approximation of) the ε-subdifferential of the current point, in which rather than erasing all the $\boldsymbol{g}_i$s not belonging to the ε-subgradient we set an upper bound ε on the "total approximation", ensuring that any $\boldsymbol{g}_i$ with a large α_i will only have a small "influence" (multiplier θ_i) on the direction $\boldsymbol{d}$. Historically, $(\Delta_\beta^\varepsilon)$ was considered the "basic" QP subproblem to be solved, and in fact it is possible to develop the whole theory of Bundle algorithms in terms of $(\Delta_\beta^\varepsilon)$ rather than of $(\Delta_{\beta t})$, see. e.g.

[HL93]: this choice had some practical rationale, since the constrained least square problem (Δ_β^ε) was considered easier to solve than the general QP problem with one constraint ($\Delta_{\beta t}$) but it is nowadays just a matter of tastes (see also [Ki85a], Introduction to Chap. 7 - in fact, ten years separate [Mi77] from [Ki86] in which specialized algorithms for the former and the latter respectively are presented). In our implementation of Bundle algorithms, we exploit the efficient algorithm [Fr95] for directly solving ($\Delta_{\beta t}$); for theoretical reasons we prefer to consider it the "basic" subproblem, but in the next paragraph it will be shown how (Δ_β^ε) can solved for any fixed ε by means of ($\Delta_{\beta t}$), so this point is not relevant, at least in theory.

The "dual" viewpoint is very important in describing Bundle algorithms and proving results: for instance, it is easy to show that for any $\theta \in \Theta$ the vector z_θ belongs to the σ_θ-subdifferential of the current point $\bar{y}$, and therefore the relation

$$\| z_\theta \| = 0 \qquad \Rightarrow \qquad \bar{y} \text{ is } \sigma_\theta\text{-optimal}$$

can be easily understood and proved in terms of the above ε-subgradients theory; conversely, the actual stopping conditions used by Bundle algorithms, i.e.

$$\| z_\theta \|^2 \le \varepsilon_Q \qquad\qquad \sigma_\theta \le \varepsilon_L$$

for two fixed tolerances ε_Q and ε_L can be better understood in terms of another interpretation, hereafter called "primal" (not to be confused with what it is called primal in textbooks, e.g. [HL93] Chap. XV), and that in our opinion is at least as much important to properly understand and design efficient Bundle methods for polyhedral functions.

2. Bundle methods and polyhedral functions: the primal viewpoint

There exist several applications in which NDO methods can be used as a tool for efficiently solving problems of different nature: stated in a very general form, let us consider the problem

$$\text{(H)} \qquad \min_x \{ \boldsymbol{cx} : \boldsymbol{x} \in X, \boldsymbol{Ax} = \boldsymbol{b} \}$$

where $X \subseteq \Re^n$ is a "simple" bounded set and $\boldsymbol{Ax} = \boldsymbol{b}$ are some "complicating" constraints, in the sense that

$$\text{(P}_\mu\text{)} \qquad \min_x \{ \mu\boldsymbol{x} : \boldsymbol{x} \in X \}$$

can be solved "efficiently" for any choice of μ while the whole (P) is significantly harder to solve.
We will focus on the solution of the convex program

$$\text{(P)} \qquad \min_x \{ \boldsymbol{cx} : \boldsymbol{x} \in \bar{X}, \boldsymbol{Ax} = \boldsymbol{b} \}$$

where $\bar{X} = \text{conv}(X')$, $X' = \{ \boldsymbol{x}_i \} \subset X$ being a finite set of its extreme points; actually, boundedness is assumed only for sake of simplicity, since otherwise the cone generated by its extreme rays should be added. This general framework covers several different cases, such as

- X is a structured polyhedron itself, such as a flow polytope and/or a separable one: hence, (H) ≡ (P) is a structured LP with "complicating" constraints, that we may to solve by some sort of decomposition approach;
- X is a discrete set corresponding to the combinatorial problem (P_μ), that is an "easier" relaxation of

the much harder (H): hence, (P_μ) can be solved either to optimality or with an heuristic, and (P) is some sort of relaxation that can be used to derive upper bounds on (H);

- X is a generic convex set that can (for the purpose of solving (H)) be finitely inner approximated;
- combinations of the above.

Following the classic Dantzig-Wolfe decomposition approach, we define the Master Problem

(M) $\quad \min_\theta \{ \boldsymbol{c}(\Sigma_i \boldsymbol{x}_i\theta_i) : \boldsymbol{A}(\Sigma_i \boldsymbol{x}_i\theta_i) = \boldsymbol{b} , \theta \in \Theta \}$

where as usual Θ is the unitary simplex in the $|X'|$-dimensional space: (M) is obviously equivalent to (P), but each $\boldsymbol{x} \in \bar{X}$ is explicitly described as a convex combination of the *vertices* $\boldsymbol{x}_i$, using (a possibly exponential number of) *combinators* θ; the fundamental strategy of the Dantzig-Wolfe algorithm is to solve (M) by generating only a (small) subset of the vertices, hence from now on by appending a subscript β to the name of a problem we restrict it to the subset $\bar{X}_\beta = \text{conv}(\{ \boldsymbol{x}_i : i \in \beta \})$, as

$$(M_\beta) \qquad \min_\theta \left\{ c\left(\sum_{i\in\beta} \boldsymbol{x}_i\theta_i\right) : A\left(\sum_{i\in\beta} x_i\theta_i\right) = \boldsymbol{b} , \theta \in \Theta \right\} \quad \equiv \quad \min_\theta \{ \boldsymbol{cx} : \boldsymbol{x} \in \bar{X}_\beta , \boldsymbol{Ax} = \boldsymbol{b} \}$$

(the dimensions of θ and Θ will be hereafter omitted, as they are always deducible from the context).
The linear dual problem of (M_β) is

(L_β) $\quad \max_{y,v} \{ \boldsymbol{yb} + v : v \le (\boldsymbol{c} - \boldsymbol{yA})\boldsymbol{x}_i \;\; i \in \beta \}$

and its dual optimal solutions $(v^*, \boldsymbol{y}^*)$ are used in the classic DW decomposition approach for the so called "pricing" operation, i.e. for obtaining an(y) optimal solution $\boldsymbol{x}^* = \boldsymbol{x}(\boldsymbol{y}^*)$ of the *pricing subproblem*

($P_{\boldsymbol{c}-\boldsymbol{y}^*\boldsymbol{A}}$) $\quad \min_{\boldsymbol{x}} \{ (\boldsymbol{c} - \boldsymbol{y}^*\boldsymbol{A})\boldsymbol{x} : \boldsymbol{x} \in X \}$

that gives a new "tentative optimal" vertex of X: if the corresponding constraint $v^* \le (\boldsymbol{c} - \boldsymbol{y}^*\boldsymbol{A})\boldsymbol{x}^*$ is satisfied, then $(v^*, \boldsymbol{y}^*)$ are clearly dual optimal solutions for the whole (L) and $\boldsymbol{x}_{\theta^*} = \Sigma_{i\in\beta} \boldsymbol{x}_i\theta_i^*$ is the optimal solution of (P), θ^* being the optimal solution of (M_β); otherwise $\boldsymbol{x}^*$ is added to β and the process is repeated.
It is a well-known fact that the DW approach is exactly equivalent to the CP approach applied to the Lagrangean Dual of (H) with respect to the complicating constraints $\boldsymbol{Ax} = \boldsymbol{b}$: in fact, by defining

$$\varphi(\boldsymbol{y}) = \boldsymbol{yb} + \min\{ (\boldsymbol{c} - \boldsymbol{yA})\boldsymbol{x} : \boldsymbol{x} \in X \}$$

we have that (L) can be written as the unconstrained maximization of (the concave polyhedral function) $\varphi()$, and $\boldsymbol{g}(\bar{\boldsymbol{y}}) = \boldsymbol{b} - \boldsymbol{Ax}(\bar{\boldsymbol{y}})$ is a subgradient for $\varphi()$ in $\bar{\boldsymbol{y}}$. The optimal dual prices $\boldsymbol{y}^*$ are used from a primal viewpoint to obtain, through the "oracle" ($P_{\boldsymbol{c}-\boldsymbol{y}^*\boldsymbol{A}}$), a new $\boldsymbol{x}_i = \boldsymbol{x}(\boldsymbol{y}_i)$ to enrich the inner approximation $\bar{X}_\beta$, and from the dual viewpoint to obtain a new $\boldsymbol{g}_i = \boldsymbol{g}(\boldsymbol{y}_i)$ to enrich the cutting plane model $\varphi_{CP}()$.
A simple notational observation may help in seeing the equivalence: for any dual vector $\boldsymbol{y}$, $\varphi(\boldsymbol{y})$ can be written as $\varphi(\boldsymbol{y}) = \boldsymbol{cx}(\boldsymbol{y}) + \boldsymbol{y}\cdot\boldsymbol{g}(\boldsymbol{y})$, and the linearization errors α_i of $\varphi()$ w.r.t. any current point $\bar{\boldsymbol{y}}$ are

$$\alpha_i = \varphi(\boldsymbol{y}_i) + \boldsymbol{g}_i\cdot(\bar{\boldsymbol{y}} - \boldsymbol{y}_i) - \varphi(\bar{\boldsymbol{y}}) = (\boldsymbol{c} - \bar{\boldsymbol{y}}\boldsymbol{A})(\boldsymbol{x}_i - \boldsymbol{x}(\bar{\boldsymbol{y}}))$$

so that for $\bar{\boldsymbol{y}} = \boldsymbol{0}$ we have $\varphi(\boldsymbol{0}) = \boldsymbol{cx}(\boldsymbol{0})$ and $\alpha_i = \boldsymbol{c}(\boldsymbol{x}_i - \boldsymbol{x}(\boldsymbol{0}))$; therefore, ($L_\beta$) can be rewritten as

(L_β) $\max_{y,v}\{\ v : v \le (\boldsymbol{c} - y\boldsymbol{A})\boldsymbol{x}_i + y\boldsymbol{b} - \boldsymbol{c}\boldsymbol{x}(\boldsymbol{0}) \quad i \in \beta\} + \varphi(\boldsymbol{0})$

i.e. after some simple algebra

$$\max_{y,v}\{\ v : v \le \boldsymbol{g}_i y + \alpha_i \ \ i \in \beta\ \} + \varphi(\boldsymbol{0})$$

which means that (L_β) is exactly the cutting plane subproblem (D_β) for the particular value $\bar{y} = \boldsymbol{0}$ of the current point (hence $\boldsymbol{d} = y - \boldsymbol{0} = y$): more in general, (L_β) and (D_β) are exactly the same problem modulo a proper translation w.r.t. the current point $\bar{y}$, and (M_β) can be rewritten as

(M_β) $\min_\theta\{\ \alpha\theta : \Sigma_i\ (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_i)\theta_i = \boldsymbol{0}\ ,\ \theta \in \Theta\ \} + \varphi(\bar{y})$

Now, just substituting $\boldsymbol{g}_i$ for $\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_i$ (and disregarding $\varphi(\bar{y})$) in the above formula, we immediately get the primal interpretation of $(\Delta_{\beta t})$: it is nothing but a Lagrangean relaxation (with Lagrangean multiplier t) of the following (nonlinear) relaxation of (M_β)

(M_β^ε) $\min_\theta\{\ \alpha\theta : (1/2)\| \boldsymbol{G}_\beta\theta \|^2 \le \varepsilon\ ,\ \theta \in \Theta\ \}$

(where $\boldsymbol{G}_\beta$ is the matrix having $\boldsymbol{g}_i$ $i \in \beta$ as columns). Here, the constraint $\boldsymbol{A}(\Sigma_i\, x_i\theta_i) = \boldsymbol{b}$, which requires the existence of a *feasible* point that can be written as convex combination of the available extreme points, has been relaxed in a *least-square fashion* to $\| \boldsymbol{A}(\Sigma_i\, x_i\theta_i) - \boldsymbol{b} \|^2 \le \varepsilon$: obviously, (M_β^0) is equivalent to (M_β), since their objective functions and feasible regions are exactly the same, and it is easy to see that (M_β^ε) is a family of relaxations of (M_β) that gets weaker as ε increases.

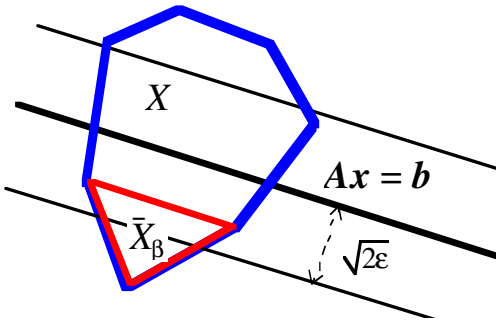

Figure 9: (M_β) can be empty while (M_β^ε) it is not

Even from a practical point of view, using (M_β^ε) instead of (M_β) may be reasonable, since it means extending the linear (affine) constraint region $\{\ \boldsymbol{x} : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\ \}$ to a "cylinder" of ray $\sqrt{2\varepsilon}$ around it, and it is always possible to find a large enough ε which makes (M_β^ε) feasible for every nonempty β (Figure 9), while (M_β) can stay unbounded until a large number of vertices is added: in this sense, the former can be viewed as a "stabilized" version of the latter, that might help in enhancing the poor practical performances of the classical DW method caused by "instability" problems analogous to those of the CP algorithm. It is instructive to remark that the same problem $(\Delta_{\beta t})$ can be seen as a Lagrangean relaxation of the two respectively primal and dual problems (M_β^ε) and $(\Delta_\beta^\varepsilon)$, with Lagrange multiplier t in the first case and $1/t$ in the second. The constraint relaxed from $(\Delta_\beta^\varepsilon)$ is aimed to ensure that all the information used to compute the direction is reliable, i.e. (almost) in the ε-subdifferential, while the one relaxed from (M_β^ε) is aimed to ensure (almost) feasibility; therefore, it is possible to see t as a "knob" which can be tuned to privilege either the dual aspect (searching for an ascent direction) or the primal one (searching for a feasible primal solution). In this sense, Bundle algorithms are close to Primal-Dual approaches, but this "balance" from two contrasting needs can also be seen in different ways:

finding a feasible $\boldsymbol{x}_\theta$ means in fact finding a null subgradient in the σ_θ-subdifferential of the current point, i.e. showing *global* σ_θ*-optimality* of $\bar{\boldsymbol{y}}$ by finding the (finite) maximum of $\varphi_{CP}()$; hence, t can be used to switch from a "local" to a "global" view of the function, that is if $\boldsymbol{d}$ has to be an ascent direction in a small neighborhood of $\bar{\boldsymbol{y}}$, or rather if $\boldsymbol{d}$ has to point towards the maximum of $\varphi_{CP}()$.

The primal view of Bundle algorithms is important for their full understanding: in particular, in the following we will discuss several properties that intimately link the usual stopping conditions of Bundle algorithms with "primal optimality"; from our experience, this can be significant in order to develop efficient Bundle codes. The main tool for describing the links between the two views is the "sensitivity" (concave, nondifferentiable) function

$$\psi(\gamma) = \min_x \{ (\boldsymbol{c} - \gamma)\boldsymbol{x} : \boldsymbol{x} \in X \} = \text{optval}((P_{c-\gamma}))$$

such that $\varphi(\boldsymbol{y}) = \psi(\boldsymbol{yA}) + \boldsymbol{yb}$: the following general theorem (that we prove as example) gives a "dual flavor" even to the vectors $\boldsymbol{x}$ of the primal space, showing how [ε-]optimality for $(P_{c-\gamma})$ is equivalent to the condition of being an [ε-]subgradient for $\psi()$.

Theorem 2.1 $\boldsymbol{x}$ *is an* ε*-optimal solution for* $(P_{c-\gamma}) \Leftrightarrow -\boldsymbol{x}$ *is an* ε*-subgradient for* $\psi()$ *in* γ.

Proof: [$\Rightarrow$] By ε-optimality of $\boldsymbol{x}$, $\psi(\gamma) + \varepsilon \geq (\boldsymbol{c} - \gamma)\boldsymbol{x}$: clearly, for any γ', $\boldsymbol{x}$ is in general suboptimal, i.e. $(\boldsymbol{c} - \gamma')\boldsymbol{x} \geq \psi(\gamma')$, hence

$$\psi(\gamma') \leq (\boldsymbol{c} - \gamma')\boldsymbol{x} + \gamma\boldsymbol{x} - \gamma\boldsymbol{x} = (\boldsymbol{c} - \gamma)\boldsymbol{x} + (\gamma - \gamma')\boldsymbol{x} \leq \psi(\gamma) + (\gamma' - \gamma)(-\boldsymbol{x}) + \varepsilon$$

[$\Leftarrow$] Conversely, suppose $\forall\ \gamma'\ \psi(\gamma') \leq \psi(\gamma) + (\gamma' - \gamma)(-\boldsymbol{x}) + \varepsilon$ holds, but $\psi(\gamma) < (\boldsymbol{c} - \gamma)\boldsymbol{x} - \varepsilon$ (i.e. $\boldsymbol{x}$ is not ε-optimal): then, by taking $\gamma' = \boldsymbol{c}$ in the previous relation, we get

$$0 = \psi(\boldsymbol{c}) \leq \psi(\gamma) + (\boldsymbol{c} - \gamma)(-\boldsymbol{x}) + \varepsilon < [\ (\boldsymbol{c} - \gamma)\boldsymbol{x} - \varepsilon\ ] - (\boldsymbol{c} - \gamma)\boldsymbol{x} + \varepsilon = 0 \qquad \Diamond$$

Now, take the linearization error $\alpha_i = \varphi(\boldsymbol{y}_i) + \boldsymbol{g}_i(\bar{\boldsymbol{y}} - \boldsymbol{y}_i) - \varphi(\bar{\boldsymbol{y}})$ of the i-th subgradient $\boldsymbol{g}_i = \boldsymbol{b} - \boldsymbol{Ax}_i$ w.r.t. any current point $\bar{\boldsymbol{y}}$: just rewriting things as

$$\alpha_i = \varphi(\boldsymbol{y}_i) + \boldsymbol{g}_i(\bar{\boldsymbol{y}} - \boldsymbol{y}_i) - \varphi(\bar{\boldsymbol{y}}) = [\ \psi(\boldsymbol{y}_i\boldsymbol{A}) + \boldsymbol{y}_i\boldsymbol{b}\ ] + (\bar{\boldsymbol{y}} - \boldsymbol{y}_i)(\boldsymbol{b} - \boldsymbol{Ax}_i) - [\ \psi(\bar{\boldsymbol{y}}\boldsymbol{A}) + \bar{\boldsymbol{y}}\boldsymbol{b}\ ] =$$

$$= \psi(\boldsymbol{y}_i\boldsymbol{A}) + (-\boldsymbol{x}_i)(\bar{\boldsymbol{y}}\boldsymbol{A} - \boldsymbol{y}_i\boldsymbol{A}) - \psi(\bar{\boldsymbol{y}}\boldsymbol{A}) = \psi(\gamma_i) + (\bar{\gamma} - \gamma_i)(-\boldsymbol{x}_i) - \psi(\bar{\gamma})$$

(where $\bar{\gamma} = \bar{\boldsymbol{y}}\boldsymbol{A}$ and $\gamma_i = \boldsymbol{y}_i\boldsymbol{A}$) it is easy to see that α_i is as well the linearization error of $-\boldsymbol{x}_i$ as a subgradient of $\psi()$ in γ_i w.r.t. $\bar{\gamma}$, or in other words that $\boldsymbol{x}_i$ *is an* α_i*-optimal solution for the pricing subproblem* $(P_{c-\bar{\gamma}})$ *corresponding to the current point* $\bar{\boldsymbol{y}}$. But this immediately implies that, for any vector of combinators $\theta \in \Theta$, $\boldsymbol{x}_\theta$ is a σ_θ-subgradient of $\psi()$ in $\bar{\gamma}$, i.e. that $\boldsymbol{x}_\theta$ *is* σ_θ*-optimal for* $(P_{c-\bar{\gamma}})$: hence, we can proof in a much simpler and more general form two theorems of [Me94] that can be used to try to measure the "quality" of the current solution.

Theorem 2.2 *let* $\boldsymbol{x}^*$ *be the optimal solution for(M) and* $z_\theta = \boldsymbol{b} - \boldsymbol{Ax}_\theta = \Sigma_i\ \boldsymbol{g}_i\theta_i$ *for any vector of combinators* $\theta \in \Theta$*: hence,* $\boldsymbol{cx}^* \geq \boldsymbol{cx}_\theta + \bar{\boldsymbol{y}}z_\theta - \sigma_\theta$.

Proof: By σ_θ-optimality of $\boldsymbol{x}_\theta$, we have $(\boldsymbol{c} - \bar{\boldsymbol{y}}\boldsymbol{A})\boldsymbol{x}^* + \sigma_\theta \geq (\boldsymbol{c} - \bar{\boldsymbol{y}}\boldsymbol{A})\boldsymbol{x}_\theta$, i.e.

$$\boldsymbol{cx}^* \geq \boldsymbol{cx}_\theta + \bar{\boldsymbol{y}}(\boldsymbol{Ax}^* - \boldsymbol{Ax}_\theta) - \sigma_\theta$$

but $\boldsymbol{x}^*$ is feasible, i.e. $\boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{b}$, and this completes the proof. ◊

Theorem 2.3 *for any vector of combinators* $\theta \in \Theta$, $\boldsymbol{c}\boldsymbol{x}_\theta - \varphi(\bar{\boldsymbol{y}}) \leq \sigma_\theta - \bar{\boldsymbol{y}}\boldsymbol{z}_\theta$.

Proof: By σ_θ-optimality of $\boldsymbol{x}_\theta$, $\varphi(\bar{\boldsymbol{y}}) = \psi(\bar{\boldsymbol{y}}\boldsymbol{A}) + \bar{\boldsymbol{y}}\boldsymbol{b} \geq (\boldsymbol{c} - \bar{\boldsymbol{y}}\boldsymbol{A})\boldsymbol{x}_\theta - \sigma_\theta + \bar{\boldsymbol{y}}\boldsymbol{b}$. ◊

However, the above theorems cannot be used in practice as stopping conditions: instead the above mentioned rules

$$\| \boldsymbol{z}_\theta \|^2 \leq \varepsilon_Q \qquad\qquad \sigma_\theta \leq \varepsilon_L$$

for two fixed tolerances ε_Q and ε_L are usually used. Since $\boldsymbol{z}_\theta = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_\theta$ and $\boldsymbol{x}_\theta$ is σ_θ-optimal for subproblem ($P_{\boldsymbol{c}-\bar{\boldsymbol{y}}}$), corresponding to the current point $\bar{\boldsymbol{y}}$, such conditions have a clear primal interpretation that can be stated as follows: stop when a point $\boldsymbol{x}_\theta$ is found such that constraints $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ are "slightly" violated ($\| \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_\theta \|^2 \leq \varepsilon_Q$) and $\boldsymbol{x}_\theta$ is "almost" optimal for the current pricing subproblem (note that this is in fact a generalization of the stopping condition of DW algorithm). This interpretation can be helpful in the critical task of choosing a suitable value for ε_Q, especially when this value is not just kept fixed, but rather dynamically adjusted during the algorithm as discussed in the next paragraph; however, for sake of completeness we want to show how the stopping conditions also make sense from the dual viewpoint, starting from the typical stopping condition that everyone would like to have at hand, i.e. classical ε-optimality: if $\boldsymbol{y}^*$ is the optimal solution of (D) (such that $\varphi(\boldsymbol{y}^*) = \boldsymbol{c}\boldsymbol{x}^*$), then the most reasonable stopping criterion is to stop when

$$\frac{\varphi(\boldsymbol{y}^*) - \varphi(\bar{\boldsymbol{y}})}{\varphi(\boldsymbol{y}^*)} \leq \varepsilon$$

for any fixed relative tolerance ε. In a slightly stronger from, this can also be stated as $\varphi(\boldsymbol{y}^*) - \varphi(\bar{\boldsymbol{y}}) \leq \varepsilon\varphi(\bar{\boldsymbol{y}})$, but since we have in principle no information about the value of $\varphi(\boldsymbol{y}^*)$, this is not an implementable rule to guarantee ε-optimality. We will show that this is a general problem, by writing down a sequence of stronger conditions (that always ensure ε-optimality) in which an unknown quantity always exists that prevents us from being able to use these conditions in practice, but that will end in a nice characterization of the only implementable stopping criterion.
The basic fact is that $\boldsymbol{z}_\theta$ is a σ_θ-subgradient of φ() in $\bar{\boldsymbol{y}}$ for any vector of combinators $\theta \in \Theta$ (typically, the latest optimal solution of ($\Delta_{\beta l}$) is used), so that $\varphi(\boldsymbol{y}^*) \leq \varphi(\bar{\boldsymbol{y}}) + (\boldsymbol{y}^* - \bar{\boldsymbol{y}})\boldsymbol{z}_\theta + \sigma_\theta$ and

$$\sigma_\theta + (\boldsymbol{y}^* - \bar{\boldsymbol{y}})\boldsymbol{z}_\theta \leq \varepsilon\varphi(\bar{\boldsymbol{y}})$$

can be used as a stopping rule; however, $\boldsymbol{y}^*$ is obviously unknown, so that one might define

$$t^*(\boldsymbol{y}, \boldsymbol{z}_\theta) = \| \boldsymbol{y}^* - \boldsymbol{y} \| / \| \boldsymbol{z}_\theta \|$$

(the distance from $\boldsymbol{y}$ and the optimum, taking $\| \boldsymbol{z}_\theta \|$ as the unitary distance) and resort to the even stronger condition

$$\sigma_\theta + t^*(\bar{\boldsymbol{y}}, \boldsymbol{z}_\theta)\| \boldsymbol{z}_\theta \| \leq \varepsilon\varphi(\bar{\boldsymbol{y}}).$$

This could in turn (in theory) be further strengthened to the "standard" form by letting $\varepsilon_L = \varepsilon\varphi(\bar{\boldsymbol{y}}) / 2$ and choosing at each step some $\varepsilon_Q \leq \varepsilon_L / t^*(\bar{\boldsymbol{y}}, \boldsymbol{z}_\theta)$. Obviously, the denominator is still an unknown quantity, but these formulas give us an idea on how ε_L and ε_Q should be set (other than the obvious

thing that ε_L has to be interpreted in a *relative* way) saying that ε_Q should be related to both ε_L and to some measure of the distance from the current point and the optimum measured "along" z_θ. Moreover, the above formulas suggest an alternative approach, i.e. to stop when

$$\sigma_\theta + \bar{t}\| z_\theta \| \le \varepsilon\varphi(\bar{y})$$

where $\bar{t}$ is an user-provided fixed upper approximation of $t^*()$: in our experience, a viable such $\bar{t}$ has always been easily deducted by looking at a few runs, and seems to be fairly stable for problems belonging to the same class; moreover, it is closely related to other important informations that are usually given to a Bundle algorithm, such as initial, minimal and maximal values for t, which are usually determined in the same way.

As a final example of the importance of the primal viewpoint, we briefly sketch how it would be possible to construct, by only making use of the basic tools introduced for Bundle methods, a (naive) "purely primal" algorithm closely resembling the DW approach that does not make any use of the typical concepts (such as the Null Step - Serious Step) of dual Bundle methods, to be discussed later in the next paragraph. We are not proposing it as an efficient algorithm, since it completely disregards all the informations on the value of the dual function $\varphi()$, but just as an illustration of how such a machinery can be used from a different point of view.
The basic observation is that we can use any (efficient) algorithm that solves $(\Delta_{\beta t})$ to approach the quadratically constrained problem (M^ε_β): in fact, the univariate function

$$\delta_\beta(t) = \text{optval}(\ (\Delta_{\beta t})\ )$$

is well known to be concave (being a pointwise minimum of an infinite set of linear functions), so that the Lagrangean Dual

$$(T^\varepsilon_\beta) \qquad \max_{t\ge 0}\{\ \delta_\beta(t) - \varepsilon t\ \}$$

can be solved by means of any line search procedure on t. It is also well known that (T^ε_β) is equivalent to (M^ε_β); it follows that the former has finite optimum if and only if the latter is nonempty, as we will illustrate in some detail in the following. It can be shown that the nonnegative halfline $\Re_+$ can be subdivided into a finite number of intervals $[t_j, t_{j+1}]$ (each one being the stability interval of an optimal *base* of $(\Delta_{\beta t})$, refer to [Fr95] for more details) such that for each $t \in [t_j, t_{j+1}]$ the optimal solutions of $(\Delta_{\beta t})$ and $(\Pi_{\beta t})$ are completely described by three numbers $p_j \ge 0$, q_j and $r_j \ge 0$ and two vectors θ'_j and θ''_j, in the sense that $\theta(t) = \theta'_j + (1/t)\theta''_j$ is the optimal solution of $(\Delta_{\beta t})$ with

$$\| \boldsymbol{G}_\beta\theta(t) \|^2 = \| z(t) \|^2 = p_j + (1/t)^2 r_j \qquad \text{and} \qquad \alpha\theta(t) = q_j - (1/t)r_j$$

while for $(\Pi_{\beta t})$

$$\| \boldsymbol{d}(t) \|^2 = \| tz(t) \|^2 = t^2 p_j + r_j \qquad \text{and} \qquad v(t) = tp_j + q_j$$

so that

$$\delta_j(t) = v(t) - (1/2t)\| \boldsymbol{d}(t) \|^2 = (1/2)[\ tp_j + 2q_j - (1/t)r_j\ ] = (1/2)t\| \boldsymbol{G}_\beta\theta(t) \|^2 + \alpha\theta(t)$$

is equal to $\delta_\beta(t)$ for each $t \in [t_j, t_{j+1}]$.
Being the sum of the linear functions $(1/2)tp_j$ and $2q_j$ and of the concave function $-(1/2t)r_j$, $\delta_j(t)$ is a (strictly, if $r_j > 0$) smooth concave upper approximation of $\delta_\beta(t)$, while the latter is a piecewise concave

possibly nonsmooth function (see Figure 10). Therefore

$$\frac{d}{dt}\delta_j(t) = (1/2)(p_j + (1/t)^2 r_j) = (1/2)\| z(t) \|^2 \geq 0$$

is the derivative of $\delta_\beta()$ for any $t \in (t_j, t_{j+1})$ and its left and right derivative in the (resp. right and left) extremes of the interval, hence $\delta_\beta()$ is a non decreasing function.

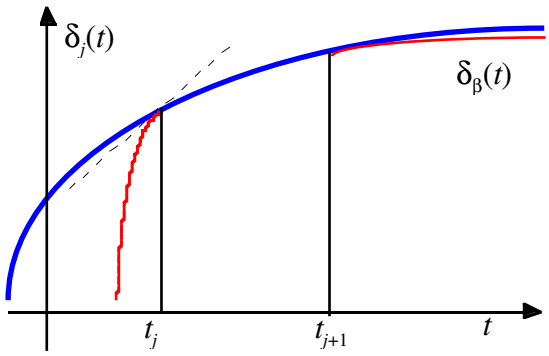

Figure 10: $\delta^+(t)$ is a smooth concave upper approximation of the nonsmooth $\delta_\beta(t)$

Therefore, if $\delta_\beta()$ happens to be differentiable in the optimal point $\bar{t}$, then $\| z(\bar{t}) \|^2 = 2\varepsilon$: in the same way, $\delta_\beta()$ is not differentiable in $\bar{t}$ if and only if $\bar{t} = t_i$ for some i, and two equivalent optimal solutions $\theta' = \theta_j(\bar{t})$ and $\theta'' = \theta_{j+1}(\bar{t})$ of $(\Delta_{\beta\bar{t}})$ exist such that

$$\varepsilon \geq (1/2)\| \boldsymbol{G}_\beta \theta' \|^2 \qquad \text{and} \qquad (1/2)\| \boldsymbol{G}_\beta \theta'' \|^2 \leq \varepsilon$$

where at least one of the two inequalities strictly holds. Hence, as the optimal solutions set of a convex problem is convex, $\mu\theta' + (1 - \mu)\theta''$ is still an optimal solution of $(\Delta_{\beta\bar{t}})$ for each $\mu \in [0, 1]$, and it is easy to show that it is always possible to (constructively) find a $\bar{\mu}$ such that

$$\| \boldsymbol{G}_\beta[\bar{\mu}\theta' + (1 - \bar{\mu})\theta''] \|^2 = \| \boldsymbol{G}_\beta \theta(\bar{t}) \|^2 = \| z(\bar{t}) \|^2 = 2\varepsilon$$

(as $\mu\| \theta' \|^2 + (1 - \mu)\| \theta'' \|^2 \geq \| \mu\theta' + (1 - \mu)\theta'' \|^2$ for each μ).
In other words, $\bar{t}$ is just the smallest value of t, if any, for which an optimal solution of $(\Delta_{\beta t})$ exists (and can be constructively found) that is feasible for (M^ε_β): this also shows why boundedness of (T^ε_β) implies nonemptyness of (M^ε_β) and vice-versa, since the existence of an $\bar{\boldsymbol{x}} \in \bar{X}_\beta$ such that $\| \boldsymbol{A}\bar{\boldsymbol{x}} - \boldsymbol{b} \|^2 \leq \varepsilon$ is equivalent to the existence of a $\bar{t}$ such that the derivative of $\delta_\beta(t) - \varepsilon t$ in $\bar{t}$ is ≤ 0.
All the above facts lead to the definition of a line-search procedure that can be sketched as follows: for given ε, β and t, $(\Delta_{\beta t})$ is solved and its sensitivity analysis parameters are used to find the new t or to detect unboundedness.

In fact:

- if $p_j < \varepsilon$, then $t' = \sqrt{ r_j / (\varepsilon - p_j) }$ is good value to test: in particular, if $t' \in [t_j, t_{j+1}]$ than it is optimal and we have explicit formulae for obtaining the optimal θ, v ... ; otherwise we move to the new interval. This ensures finite termination, since any interval (but the last one) cannot be visited more than once
- if $p_j \geq \varepsilon$, then $\| z(t) \|^2 > 2\varepsilon$ for each $t \in [t_j, t_{j+1}]$ and we must try some $t' > t_{j+1}$ to see whether a suitable interval exists: hence, if $t_{j+1} = +\infty$ we can conclude that (T^ε_β) is unbounded, and we return the current value of t and the optimal solutions of $(\Delta_{\beta t})$ to be used to obtain new informations

Hence, in a finite number of steps we can either find the optimal $\bar{t}$ (and the associated $\theta(\bar{t})$, $v(\bar{t})$ ...) or

detect unboundedness, reporting a t (and all the rest) that "demonstrates" it. Interestingly enough, this is *not* a line search on the space of dual variables $\boldsymbol{y}$: in fact $\varphi(\boldsymbol{y})$ is not evaluated on the sequence of points $\{\boldsymbol{y}^h\}$ obtained along the procedure (we are just interested in what the current model says), although one could imagine a procedure that does; moreover, the points $\boldsymbol{y}^h$ do not lie along the same line and this would eventually lead to a *curve-search* approach when seen from a dual viewpoint.
We can now state the algorithm:

Stabilized Dantzig-Wolfe

```
t̄ = t_0; ε = ε_0; β = { g(0) };                    /* initialization */
do
    ( Bounded , t̄ , v, y ) = SolveT( ε , β , t̄ );   /* the above line search on t */
    g = g(y); α = cx(y);                            /* solve the pricing subproblem */
    if( v > α + gy )                                /* "new information" obtained */

        β = β ∪ { g };                              /* exploit it (enlarge X̄_β) */

    else if( Bounded )                              /* (M^ε_β) equivalent to (M^ε) */
            ε = p_j + (1/t_{j+1})^2 r_j;            /* increase the required primal accuracy */
        else t̄ = 2t̄;
while( ε > 0 );
```

The proof of convergence is simple, and can be carried along without ever mentioning the "dual" aspects: when the new $\boldsymbol{g} = \boldsymbol{g}(\boldsymbol{y})$ gives no "new information", i.e. $v \le \alpha + \boldsymbol{g}\boldsymbol{y}$, the current value of $\delta_\beta(t)$ is equal to $\delta(\bar{t})$ [= optval($(\Delta_{\bar{t}})$)], since $(v, \boldsymbol{y})$ solve $(\Pi_{\bar{t}})$; in fact, being $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{y})$ the solution of the pricing subproblem, we have

$$(\boldsymbol{c} - \boldsymbol{y}\boldsymbol{A})\boldsymbol{x} \le (\boldsymbol{c} - \boldsymbol{y}\boldsymbol{A})\boldsymbol{x}' \qquad \forall\, \boldsymbol{x}' \in X$$

i.e. $$v \le \alpha + \boldsymbol{g}\boldsymbol{y} = \boldsymbol{c}\boldsymbol{x} + \boldsymbol{y}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}) \le \boldsymbol{c}\boldsymbol{x}' + \boldsymbol{y}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}') = \alpha' + \boldsymbol{g}'\boldsymbol{y} \qquad \forall\, \boldsymbol{g}'$$

so that $(v, \boldsymbol{y})$ is feasible for $(\Pi_{\bar{t}})$, $\theta(\bar{t})$ is feasible for $(\Delta_{\bar{t}})$ and complementary slackness hold. Therefore, provided that the original problem (P) has nonempty feasible region, for any fixed ε only a finite number of iterations is required in order to obtain a "Bounded" condition: in fact, at each step either β is enlarged or $\bar{t}$ is augmented, but there cannot be infinitely many consecutive $\bar{t}$ augments as this would imply that $\delta()$ goes to infinity, contradicting the boundedness of (T^0), and for any $\varepsilon \ge 0$ it exists a finite β such that (M^ε_β) is nonempty and (T^ε_β) is bounded. The latter argument also shows that there cannot be infinitely many ε decreases, since when β contains enough vertices it exists a t such that $\| z(t) \|^2 = p_j + (1/t)^2 r_j = 0$ $(\Rightarrow p_j = r_j = 0)$ that must be at some point reached (as the number of intervals is finite and don't depend on ε) forcing ε to become 0 - some subtle details, as how t_{j+1} as to be defined when $\bar{t}$ is a kink point of $\delta_\beta()$, can be omitted. Now, if the current $\bar{t}$ solves (T^ε_β), and $v \le \alpha + \boldsymbol{g}\boldsymbol{y}$, then $\theta(\bar{t})$ is optimal for the whole (M^ε), since it solves $(\Delta_{\bar{t}})$ and satisfies $\| \boldsymbol{G}_\beta \theta(\bar{t}) \|^2 = 2\varepsilon$, so that it also solves (T^ε), that is known to be equivalent to (M^ε): therefore, when the algorithm stops (with $\varepsilon = 0$) the obtained solution is optimal to (M^0), hence to (P).

Note that, in this very general setting, we haven't addressed several questions that might be of relevant

importance in practice, such as the "reset strategy" (how and when to eliminate items from β in order to keep it to a reasonable size) and the possibility of changing the current point from $\mathbf{0}$ to any general $\bar{\boldsymbol{y}}$, that can be obviously embedded in this framework: all we are interested in, in fact, is to make apparent all the deep relations between Bundle methods and Dantzig-Wolfe methods, and to clarify the role of the quadratic subproblems ($\Delta_{\beta t}$) and ($\Pi_{\beta t}$) as the "core" of both. In the next paragraph, we will focus on the details of implementing a Bundle algorithm for polyhedral functions based on the above theory.

3. A generic Bundle algorithm

Having described the main ideas under the Bundle approach, we can now turn to introduce what we view as a generic Bundle algorithm: we adopt a broad definition of the class, that seems to be quite accepted in the literature, but it is possible to find references (e.g. in [Ki85a]) in which the term "Bundle algorithm" is reserved only to a specific subclass of such methods; furthermore, despite its generality this definition does not cover all the Bundle-like methods proposed in the literature, most noticeably

- methods that try to extract some sort of second order information about the curvature of the function, for instance using a variable matrix-induced norm $\|\cdot\|_{\boldsymbol{H}}$ for the "stabilizing" term of ($\Pi_{\beta t}$) (the so-called *space dilation methods*, where $\boldsymbol{H}$ is some positive semi-definite matrix taking the place of $\nabla^2\varphi()$) [Mi82], [Mi91], [GM92] or in some different ways [GM91]: while it seems to be hardly possible to define a proper concept of curvature for polyhedral functions, complicating the structure of the quadratic subproblems may hide some relevant detail of our interpretation
- bundle methods for nonconvex nondifferentiable optimization [Ki91b], or for problems with "complicated" constraints [Ki91a], that seem not to be relevant for the typical applications we are interested in
- methods which use different forms of quadratic subproblems, like the "proximal level" algorithms presented in [Ki95] and [LNN95], in order not to loose the interpretations of the previous paragraphs

Generic Bundle Algorithm

$\bar{\boldsymbol{y}} = \boldsymbol{y}_0$; $t = t_0$; $\beta = \beta_0$; /* β_0 will contain at least $\boldsymbol{g}(\boldsymbol{y}_0)$ */
do
  ⟨ find a tentative direction $\boldsymbol{d}$ by solving ($\Delta_{\beta t}$) ⟩;
  if($\boldsymbol{d}$ is acceptable)
    ⟨ move along $\boldsymbol{d}$, generating some new $\boldsymbol{g}_i$ and a trial point $\boldsymbol{y}$ ⟩;
    if(a large enough improvement has been obtained)
      then $\bar{\boldsymbol{y}} = \boldsymbol{y}$;
    ⟨ update β, (possibly) including some new $\boldsymbol{g}_i$, deleting old ones and/or adding convex combination of the present ones (typically $\boldsymbol{z}$) ⟩;
  endif
  ⟨ update t, depending on history of the past iteration(s) ⟩;
while(⟨ some stopping condition ⟩);

At this point, we must clarify the scope and the aim of the present paper: this is not intended as a comprehensive review of the literature (the interested reader is referred to the excellent surveys in the bibliography), nor as a unifying scheme from which all the known methods can be formally derived, but simply as a way to convey some of the basic ideas beyond most of the proposed approach as seen from our viewpoint, in order to put up the basis for further development and discussion; furthermore, we will restrict our treatment to unconstrained optimization algorithms only: extensions to the constrained case will be discussed later on.
Convergence of such a generically stated method cannot be shown, since proper choices of the different (mutually dependent) update strategies are required: in fact, the actual methods are mainly characterized by different choices of the (many) unspecified operations in the algorithm, that can dramatically affect both convergence and practical performances. However, several known algorithms can be more or less formally fitted into the above framework, as those mentioned in the following

- the classic ***subgradient algorithm*** [Po77], that is still popular among researchers mainly because of its simplicity (see e.g. [KU93] and the references therein). In such an algorithm β is made of the last subgradient only, so that $(\Delta_{\beta t})$ vanishes and t plays no role; no measure of a "large enough improvement" is taken (movements to the trial point $\boldsymbol{y}$ are always performed) and the stepsize γ_k along the current direction $\boldsymbol{d} = \boldsymbol{g}_k$ at the k-th iteration is taken according to some predefined rule (e.g. $\gamma_k = \gamma_0 q^k$, $q \in (0, 1)$). Although the method can be slightly extended (i.e. by taking into account the last two subgradient or introducing some space dilation technique), its fundamental Markovian nature limits its efficiency (at the k-th step, we have forgotten almost everything that is happened before). Furthermore, convergence can be guaranteed in the limiting sense only, or it would require unrealistic assumptions like an a-priori knowledge of $\varphi(\boldsymbol{y}^*)$
- the ***cutting plane algorithm*** already discussed in §1: it can be obtained straightforward by letting t be constantly equal to 0 modulo the treatment of unbounded subproblems (that however is often left unspecified)
- the so-called ***conjugated subgradient*** methods [Wo75], that can be regarded as being somewhat "dual" to the cutting plane algorithms, since they can be obtained as the "limit for $t \to \infty$" of our scheme. In fact, in these methods a modified form of $(\Delta_{\beta t})$ is solved in which the linearization errors are neglected, (i.e. $\alpha_i = 0$ is forced for each i somehow pretending that the newly obtained subgradients all belong to $\partial\varphi(\bar{\boldsymbol{y}})$), and at each iteration a specialized (inexact) line search is performed along $\boldsymbol{d}$ that either finds a "significantly better" point $\boldsymbol{y}$ and moves to it, or stays at the current point producing a new subgradient $\boldsymbol{g}_i$ which is guaranteed to yield a "different" $\boldsymbol{d}$. According to the standard terminology, the first case is denoted as *Serious Step* (SS) and the typical condition required is that $\varphi(\boldsymbol{y}) \geq \varphi(\bar{\boldsymbol{y}}) + mv$ for some fixed $m \in (0, 1]$, while the latter case is known as a *Null Step* (NS). The main characteristic of such approaches is that at any Serious Step they need some sort of "reset rule" to avoid that outdated information (subgradients with a high α_i) interfere with the determination of a correct $\boldsymbol{d}$. Several such rules may be implemented, ranging from the coarsest in which the Bundle is cleaned leaving only the subgradient of the current point, to some "smoother" rules in which all the subgradients with a "small enough" α_i will survive a SS, or to *exponential smoothing* techniques in which the old $\boldsymbol{g}_i$s are multiplied by a factor $\beta^{A(i)}$, where $\beta < 1$ and $A(i)$ is the age of the subgradient (how many iterations ago it was generated)

- what has from the beginning been called "***Bundle method***" [Le77], [BNS87], [HL93] (Algorithm XIV.3.4.2), that can be seen as a first form of "smoothing" of the previous conjugated subgradient methods in which the reset rule is indirectly implemented by solving $(\Delta_\beta^\varepsilon)$ for some tolerance ε (in our framework this is obtained by a proper t adjusting scheme). Almost all the rest is the same indeed, i.e. an analogous inexact line search along the tentative ascent direction $\boldsymbol{d}$ is performed, resulting either in a SS or in a NS, with the noticeable characteristic that *no step larger than one is ever taken*, i.e. the next point is no further than $\bar{\boldsymbol{y}} + \boldsymbol{d} = \bar{\boldsymbol{y}} + t\boldsymbol{z}$: the critical part is that ε must be dynamically varied and the actual performance of the algorithm highly depends on it. Unfortunately no clear rules have been deviced so far for this task
- the most recent approaches, that can be seen as a further "smoothing" of the previous ones, in which $(\Delta_{\beta t})$ is solved for some t instead of solving $(\Delta_\beta^\varepsilon)$ [Ki90], [SZ92], [HL93] (Algorithm XV.3.3.4). These algorithms make no use of a line search as only $\varphi(\bar{\boldsymbol{y}} + \boldsymbol{d})$ is evaluated for each $\boldsymbol{d}$; therefore they need some dynamic adjustment of t in order to reach acceptable performances even though convergence is obtainable for any fixed t. The t management strategy is the critical part of any such algorithm and it is still unclear how it should really be done: different proposals have been made, (for example in [Ki90] t is chosen with an interpolation rule in which one assumes that $\varphi()$ is close to a quadratic function in a neighborhood of $\bar{\boldsymbol{y}}$), but the most appealing one, at least in theory, seems to be exploiting the "trust radius" nature of t that we discussed in the previous paragraphs. The resulting methods, usually denoted as ***Bundle/Trust Region approaches***, are mainly based on some further rules to decide on the updating of t : in case of a SS, besides moving to $\boldsymbol{y}$, t should be augmented whenever there is the chance that $\varphi_{CP}()$ be a "good" approximation of $\varphi()$ also outside the current trust region; on the other hand, in case of a NS the new $\boldsymbol{g}_i$, is included if the lack of improvement is due to a substantial lack of information of the cutting plane model (think to a small α_i), or to diminish t if it is suspected that $\varphi_{CP}()$ is substantially correct and a good improvement would have been possible within the neighbourhood of the current point.

Some features are common to almost all the above methods: for instance, it is always possible to keep the number of subgradients low by replacing any subset $\{ (\boldsymbol{g}_i, \alpha_i) \in \beta' \subseteq \beta \}$ with the convex combination $(\boldsymbol{g}_\theta, \alpha_\theta) = (\Sigma_i \boldsymbol{g}_i\theta_i, \Sigma_i \alpha_i\theta_i)$ for any $\theta \in \Theta$ (needles to say, usually $\boldsymbol{z}$ is selected among all possible combinations), since any such "artificial" $\boldsymbol{g}_\theta$ is in fact an α_θ-subgradient for $\varphi()$ at $\bar{\boldsymbol{y}}$, so that in the convergence proof of the algorithms one doesn't ever need to distinguish between the two provided that some other invariants hold; furthermore, the performances of all the actual implementations of the above algorithms are unfortunately strongly dependent on the choice of some *critical algorithmic parameters* (be t, ε or whatever) which are fundamentally related to $(\Delta_{\beta t})$ and whose value is admittedly hard to guess ([HL93], Remark XIV.4.2.1 and comments to Algorithm XV.3.3.4). With regard to this general difficulty, our claim is that *the fundamental deficiency of Bundle approaches is the lack of any* (at least *tentative*) *"long term" strategy*: likewise the inefficiency of subgradient algorithms, that can be directly related to their Markovian nature, the main problem of all the proposed Bundle approaches lies in their "myopic" nature, i.e. in the fact that all the fundamental decisions (on the critical algorithmic parameters) are taken just on the basis of the outcome of one iteration (the current one) rather than on the behavior of the algorithm during some longer period of time; to sum up, even tough the past information generated about $\varphi()$ is kept, Bundle methods suffer from a lack of information about their own past choices. This kind of problems are mostly visible in

the practical behavior of Bundle algorithms when used on tough and large sized instances: for example, in Trust Region algorithms t is usually monotonically nonincreasing, since the conditions required for a t augment are almost never satisfied; therefore, a very small t is quickly reached giving birth to a sequence of very short Serious Steps and requiring an unacceptable number of iterations to reach the optimal point. A disappointing observation is that, just keeping t fixed to some large value (that however still depends on the scaling of the function), the algorithm initially performs much better, although still requiring far too many iterations to converge.

Obviously, it is not easy to find a formalized notion of "long term behavior", and, most of all, one that could be used straightforward within an implementable algorithm. In the following we will introduce two different approaches to the problem and study their applicability to some of the above methods providing some additional details.

4. Variants to Bundle methods

We start by detailing the Bundle/Trust Region algorithm as introduced in [SZ92] in order to take it as a term of comparison with other possible proposals and as a basis for further extensions. We will not be specific about the stopping rules, being intended that anyone among the suitable ones discussed in §2 is used with the "scaled" stopping precision $\varepsilon_L\varphi(\bar{y})$.
The main ideas of the algorithm are simple: in order to accept the new point $\boldsymbol{y}$ as the current point, the *obtained* increase $\Delta\varphi$ must be at least a constant fraction (m_1) of the *promised* increase $v = \varphi_{CP}(\bar{\boldsymbol{y}} + \boldsymbol{d})$; if not, the newly obtained subgradient $\boldsymbol{g}$ is "trusted" as a correct first-order information about $\varphi()$ only if it belongs to the $(m_3\sigma_\theta)$-subdifferential of $\varphi()$ in $\bar{\boldsymbol{y}}$, where m_3 is a fixed constant and σ_θ is used the "average error" of the subgradients that have been used to obtain $\boldsymbol{d}$.

Bundle/Trust Region Algorithm

```
⟨ choose 0 < m1 < m2 < 1, 0 < m3 < 1, tM > 0 ⟩;
ȳ = y0; t = t0; β = { g(y0) };
do
 θ = Optsol( (Δβt) ); (v, d) = Optsol( (Πβt) );
 y = ȳ + d; g = g(y); Δφ = φ(y) - φ(ȳ); α = Δφ - g·d;
 if( Δφ ≥ m1v )                                   /* SS(i) */
      β = β ∪ { g };
      if( ( g·d ≤ m2v ) or ( t = tM ) )           /* SS(ii) */
           ȳ = y;
      else Increase( t );
 else                                             /* NS(i) = ~SS(i) */
      if( ( α ≤ m3σθ ) and ( Δφ ≤ || zθ ||² + σθ ) )   /* NS(ii) */
           β = β ∪ { g };
      else Decrease( t );
while( ~ Stop );
```

Some details are worth of discussion: condition SS(i), measuring the *relative* improvements, actually depends on how "demanding" one has been at the previous iteration, i.e. it depends on t. This

strategy, which is common to all the Bundle algorithms listed in the previous paragraph, has a potential drawback in that the *absolute* magnitude of $\Delta\varphi$ is completely disregarded, possibly leading to situations where a negligible improvement is considered important.

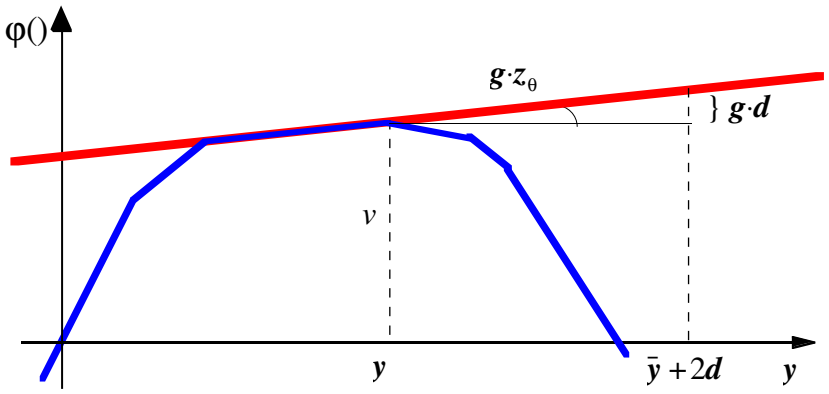

Figure 11: interpretation of SS(*ii*) rule in terms of expected improvement in $\bar{\boldsymbol{y}} + 2\boldsymbol{d}$

Condition SS(*ii*) tries to avoid this problem by detecting the situations in which a much better improvement could have been obtained by augmenting t, provided that t is still smaller of the fixed threshold t_M which is necessary for numerical purposes. Since moving to the trial point $\boldsymbol{y}$ would nullify the linearization error of newly obtained $\boldsymbol{g}$ ($\in \partial\varphi(\boldsymbol{y})$), then $\boldsymbol{g}\cdot\boldsymbol{d} + 0 \leq m_2 v < v$ ensures that the current $(v, \boldsymbol{d})$ will no more be optimal to $(\Pi_{\beta\cup\{\boldsymbol{g}\}t})$, i.e. the new information has seriously modified the cutting plane model. However, since $\boldsymbol{d} = tz_\theta$ and $\boldsymbol{g}\cdot z_\theta$ is the slope of the restriction of $\varphi()$ along z_θ, the above condition also implies that a further step of t along $\boldsymbol{d}$ can yield an improvement of at most $m_2 v$ (see Figure 11), i.e. this rule actually depends on the previous choice of t.

The NS(*ii*) condition is quite clear, even though its actual form is questionable (the extra condition $\Delta\varphi \leq \| z_\theta \|^2 + \sigma_\theta$ is not necessary when optimizing polyhedral functions and will be replaced in the following): a viable alternative might be requiring "strong" violation of the corresponding dual constraint, i.e. $m_3 v > \boldsymbol{g}\cdot\boldsymbol{d} + \alpha$, much in the spirit of the above discussion; however, the conditions for changing t altogether implements a *curve-search* procedure on t, in which a "good" value of the trust region parameter is searched for by sampling the space around $\bar{\boldsymbol{y}}$ in a pretty unusual and appealing way, i.e. not on a single fixed direction, but letting $\boldsymbol{d}$ change in response to the t variation.

Deciding about the actual Decrease() and Increase() of t is a delicate task, and nothing better than heuristic rules are known so far: the most popular one, used in [Ki90] for both increasing and decreasing t, is based on approximating the restriction of $\varphi()$ along z_θ with a quadratic function passing by $\varphi(\bar{\boldsymbol{y}})$ and $\varphi(\boldsymbol{y})$ and having $\boldsymbol{v}$ as derivative in 0, ending in

(*i*) $$t_{New} = 2tv(v - \Delta\varphi)$$

(in the original development, the formula is written for $u = 1/t$).

In the case of a polyhedral function, the rule can be questioned in that, when $v = \Delta\varphi$ (a case that is not unlikely to happen, especially if t is small) it gives no informations; moreover, it does not take into account the (known) derivative $\boldsymbol{g}\cdot z_\theta$ of the restriction at $\boldsymbol{y}$, therefore alternative methods such as

(*ii*) $$t_{New} = t\frac{t\boldsymbol{g}\cdot z_\theta - 2\Delta\varphi}{2(\boldsymbol{g}\cdot z_\theta - \Delta\varphi)} = t\frac{\alpha_i + \Delta\varphi}{2\alpha_i}$$

(*iii*) $$t_{New} = \frac{\alpha - 0}{(v / t) - \boldsymbol{g}\cdot\boldsymbol{z}_\theta}$$

may be more sensible when increasing or decreasing t respectively, even though some kind of a safeguard against degenerate cases is still necessary. Note that (*ii*) can be obtained as (*i*) by using the derivative in $\boldsymbol{y}$ rather than in $\bar{\boldsymbol{y}}$, and has the nice property that $t_{New} > t \Leftrightarrow \boldsymbol{g}\cdot\boldsymbol{d} > 0$; (*iii*) just gives the t where the linearizations in $\boldsymbol{y}$ and in $\bar{\boldsymbol{y}}$ (restricted to $\boldsymbol{z}_\theta$) intersects, and it is reasonable in view of the piecewise-linear form of the restriction itself.

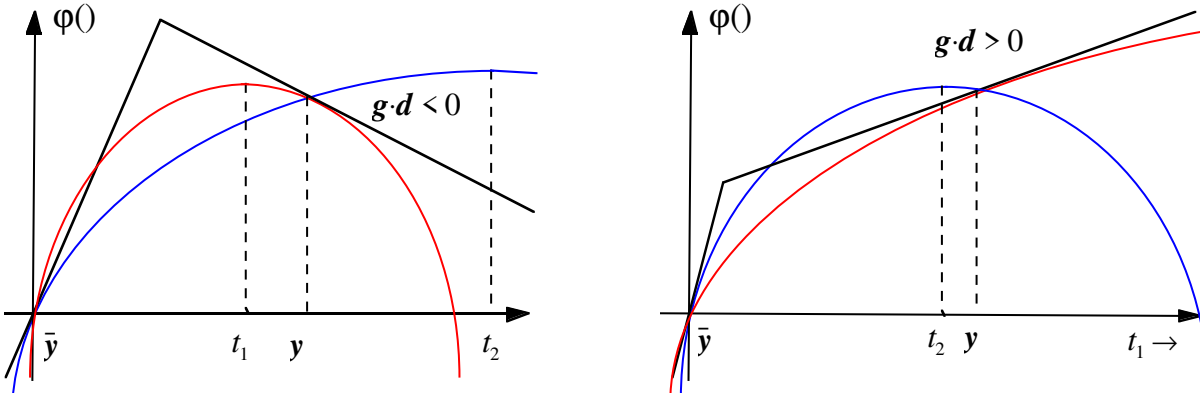

Figure 12: rulees (*i*) and (*ii*) can give very different values for t_1 and t_2 in different situations

However, in our experience simpler rules like multiplication by a constant value (i.e. bisection) have not been outperformed by the more sophisticated versions, in that none of the above has a real impact on the long term behavior of the algorithm (even though there may be relevant differences in the actual performances): as we already said, the major defect of BT algorithms is that their convergence rate rapidly decreases during the algorithm, up to the point where most of the overall execution time is taken by a long sequence of "short" Serious Steps with a small t yielding only a small fraction of the total improvement (with respect to the initial point).

We will now introduce our first attempt to solve this problem: one of its innovative characteristics, other than keeping trace of some parameters that give an idea of the status of the execution on a long term view, is that for the first time the sensitivity analysis informations of the ($\Delta_{\beta t}$) subproblem (w.r.t. t) are used to guide the decisions at the upper level.

Here, the parameter $\bar{\varepsilon}_L$ implements the long term "memory" of the method (note that using the "standard" stopping rule here would require two such parameters); we do not even consider a step to $\boldsymbol{y} = \bar{\boldsymbol{y}} + \boldsymbol{d}$ if the maximum obtainable improvement, measured by v, is not considered large enough, i.e. greater than $\bar{\varepsilon}_L$ in a relative sense; $\bar{\varepsilon}_L$ is decreased once no other increments of the current magnitude are expected. When no sufficient improvement is possible, a t increase is preferred, t being set to the smallest positive t that will produce a large enough v at the next iteration (note that $tp + q$ is an upper bound on such a v, so more than one consecutive short step can be performed): if such a t does not exist, implying that the CP model is bounded ($p = 0$), then t is set to $\bar{t}$ to reach (possibly global) convergence at the next iteration (since $\sigma_\theta \le v$). Note that this strategy has a clear interpretation from the primal viewpoint: when only negligible increments of the dual objective function value seem to be possible, trying to decrease $\| \boldsymbol{z}_\theta \|^2$ (going towards primal feasibility) is preferred.

Modified Bundle/Trust Region Algorithm

⟨ choose $0 < m_1 < m_2 < 1$, $0 < m_3 < 1$, $t_M > 0$, $\bar{\varepsilon}_L \geq \varepsilon_L$ ⟩;
$\bar{y} = y_0$; $t = t_0$; $\beta = \{ g(y_0) \}$;
do
 θ = Optsol($(\Delta_{\beta t})$); (v, d) = Optsol($(\Pi_{\beta t})$);
 $y = \bar{y} + d$; $g = g(y)$; $\Delta\varphi = \varphi(y) - \varphi(\bar{y})$; $\alpha = \Delta\varphi - g \cdot d$;
 if($\sigma_\theta + \bar{t} \| z_\theta \| \leq \bar{\varepsilon}_L \varphi(\bar{y})$) /* convergence for $\bar{\varepsilon}_L$ */
  $\bar{\varepsilon}_L = (\sigma_\theta + \bar{t} \| z_\theta \|) / 2$;
 if($v < \bar{\varepsilon}_L \varphi(\bar{y})$) /* useless step */
  if($p = 0$) /* p, q, r are those of §2 */
   $t = \bar{t}$;
  else $t = (\bar{\varepsilon}_L \varphi(\bar{y}) - q) / p$;
 else
  ⟨ ordinary SS(..), NS(..) part ⟩;

while($\sigma_\theta + \bar{t} \| z_\theta \| > \bar{\varepsilon}_L \varphi(\bar{y})$);

Whenever the short step is not detected, the original BT algorithm is followed: actually, the (useless) second part of NS(*ii*) condition must be replaced by "$v \approx \bar{\varepsilon}_L \varphi(\bar{y})$", to avoid a t increase being nullified by a t decrease due to the curved-search mechanism, which would cause a severe instability of t. With the aforementioned precautions, it is actually possible to adapt our modification also to other $(\Delta_{\beta t})$-based Bundle algorithms such as [Ki90]. A critical point of our modified scheme is the initial choice of $\bar{\varepsilon}_L$, that has an obvious impact on the global behavior of the algorithm: our preliminary computational experience (that will be discussed later on) suggests that the short step detection mechanism should not come into play too early, since it can interfere with the nice initial behavior of Bundle algorithms: therefore, a reasonable choice for $\bar{\varepsilon}_L$ may be some small multiple of the actual optimality parameter ε_L.

A somewhat different but related method can be described as follows: let LineSearch(z , t , $\bar{\varepsilon}_L$) = (y, τ) be any procedure performing a line search along z using t as initial step and stopping once a relative precision of $\bar{\varepsilon}_L$ is attained, i.e. when for the selected next step $\bar{\tau}$

$$(\varphi_{CP}(\bar{y} + \bar{\tau} z) - \varphi^*) / \varphi^* \leq \bar{\varepsilon}_L$$

holds, where φ^* is the best value of $\varphi()$ found so far; furthermore, let such procedure report $\tau > 0$ as the step relative to the best trial point visited and y as the optimal point along z; note that the two quantities refer to different points when $\varphi^* = \varphi(\bar{y}) > \varphi(\bar{y} + \tau z)$ (hence $y = \bar{y}$), i.e. no increasing step has ever been performed during all the line search. The rationale for such a method is that a correct t adjustment is the key of any Bundle algorithm, therefore it makes sense to spend some resources in order to gather more informations about t (in its trust radius interpretation): this leads to the idea that an inexact line search should be used to improve our knowledge of t rather than searching for the actual maximum of the function along a (potentially wrong) direction. Note that this is a pretty different viewpoint from the linesearch-based Bundle methods discussed in the previous paragraph: in fact, the line search used therein is bounded not to go further than t while ours is not constrained and

can yield an optimal step $\tau > t$.

Bundle/Line Search Algorithm

```
⟨ choose ε̄_L ≥ ε_L , μ ∈ (0, 1) ⟩;
ȳ = y_0; t = t_0; β = { g(y_0) };
do
    θ = Optsol( (Δ_βt) ); (v, d) = Optsol( (Π_βt) );
    if( v < ε̄_L φ(ȳ) )                          /* useless step */
        if( p = 0 )                              /* p, q, r are as in §2 */

            t = t̄;

        else  t = ( ε̄_L φ(ȳ) - q ) / p;
    else
        ( y , τ ) = LineSearch( z_θ , t , ε̄_L );
        t = τ;                                   /* t update */
        if( y = ȳ )                              /* a "null" step */
            ε̄_L = μ ε̄_L + ( 1 - μ ) ε_L;
        else                                     /* a "serious" step */
            ε̄_L = max{ ε̄_L , ( φ* - φ(ȳ) ) / φ* }; ȳ = y;
        Update( β );
while( σ_θ + t̄ || z_θ || > ε̄_L φ(ȳ) );
```

The variable $\bar{\varepsilon}_L$ still plays the role of the "long term memory" of the algorithm: its management is somewhat different, in that it is decreased at every "null" step to take into account the possibility that the failure of the step be caused by a "coarse" line search, while it may be augmented at some "serious" step to reflect the size of the real increase obtained; the parameter β reflects how reluctant the algorithm is in decreasing $\bar{\varepsilon}_L$, and should be set relatively near to 1 to avoid too many exact line searches; a good guess for the initial $\bar{\varepsilon}_L$, is not any more crucial since it will be automatically set after the first serious step.

Some more details are worth discussion: the intentionally generic Update(β) procedures is supposed to add to the Bundle a subset of the relevant information generated during the line search, among which there will certainly be the subgradient computed at $\bar{y} + \tau z_\theta$ (as it can be easily seen), using some of the rules previously discussed; note that the LineSearch() procedure ought to be specifically developed for polyhedral functions and can take advantage of all of the previously generated informations since the triplets $(\alpha_i, g_i \cdot z_\theta)$ for $i \in \beta$ exactly define the restriction of $\varphi_{CP}()$ along z_θ.

The above methods have to be intended as a general framework and several details can be modified within the actual implementations in order to enhance the computational performance: for example, in our MBT code the "short" step and the successive one are in fact performed at the same time, i.e. $\varphi(\bar{y} + d)$ is calculated, the newly obtained subgradient g is inserted in β and $(\Delta_{\beta t})$ is solved before t is changed in order to use more information for the delicate task of modifying t; in the same way, in our BLS code a t increase (using some heuristic rule) is issued whenever a "serious" step is done with $\min\{ \varphi^* - \varphi(\bar{y}) , \sigma_\theta \} < \varepsilon_L \varphi(\bar{y})$. Actually, even deeper modifications can be constructed: for instance, it is intuitively clear that a "pure" Trust Region method should be faster than a "pure" Line Search

methods, since the former can better exploit the information provided by the Bundle when computing the direction at each step; it is therefore possible to think of a mixed method in which the line search is substituted with a curved search with analogous features, i.e. in which the stopping condition is $(v - \varphi(\boldsymbol{y})) / \varphi(\bar{\boldsymbol{y}}) \leq \bar{\varepsilon}_L$, in such a way that the search provides an even more correct information on how far can we trust our model as we require the predicted value $v = \varphi_{CP}(\boldsymbol{y})$ being "near" to the actual one. The message we would like to convey is that, at least as long as polyhedral functions are concerned, a large number of possibly different Bundle methods can be constructed by combining the few basic elements that have been introduced so far; as different applications may benefit from specialized methods, as discussed later in this paper, only computational experience will determine which variant is the most sutable for a certain class of problems.

5. Implementational issues

To be effective in a real application, a Bundle algorithm has to be not only theoretically efficient but also must be carefully implemented: perhaps the first need of any good Bundle code is a very fast QP solver able to solve thousands of $(\Delta_{\beta t})$ problems with a reasonable computational cost. The main characteristic of any suitable QP solver is efficiency in reoptimization: actually, all the specialized algorithms known in the literature work on the dual problem $(\Delta_{\beta t})$ and employ the well-known active-set strategy which makes very simple reoptimizing after typical data updating as insertion/deletion of some $\boldsymbol{g}_i$, modification of the vector α and of t. Actually, all such algorithms bear strict resemblance to one another, from the very first [Mi77] that actually solves $(\Delta_{\beta}^{\varepsilon})$ to the first specialized algorithm designed for $(\Delta_{\beta t})$ [Ki86] and up to the recent [Ki94] and [Fr95] that efficiently solve the extended problem discussed below for the case of constrained maximization. Our implementation uses the efficient C++ code described in [Fr95] to which the interested reader is referred for more details: here, we only remark that computational experiences therein reported has shown how it outperforms non-specialized codes as [Po83] (that incidentally is used as $(\Delta_{\beta t})$-solver within the BT code of [SZ92]) and even modern interior-point based solvers [CLM93] of up to three orders of magnitude, making it evident how critical the solution of the QP subproblem is for the overall efficiency of the method. This is true under several aspects: for instance, a fast solver may allow larger bundles that, in turn, may help obtain good convergence ratio, but also the interface between the two codes must be carefully designed in order to make available all the relevant informations generated by the QP solver to the Bundle algorithm (think to the sensitivity analysis parameters). In fact, it is possible that the limited success of Bundle algorithms in practical applications where simpler but much slower subgradient methods have almost always been preferred, is probably due to the extra computational burden required by the solution of the $(\Delta_{\beta t})$ subproblem, i.e. in practice to the lack of available efficient specialized QP solvers: the latest generation of codes, if made widely available to the scientific community, would probably greatly help in filling this gap.

Several details of a Bundle code have a non negligible influence on its practical performances, but for space reasons can only be sketched here: we mention

- the so-called "reset rule", i.e. how and when old subgradients are discarded from B, and eventually aggregations are made: since it is well-known that polyhedral functions need large bundles to be dealt with, a good choice seems to be fixing a relatively large maximal dimension and to discard

only subgradients that have been redundant in the definition of the CP model (this information is naturally obtained from the QP solver) during the last k iterations, for some fixed k; from this viewpoint, aggregation is an "emergency" procedure that must be used only when the maximal dimension of the Bundle is reached and every subgradient is "relevant" for the CP model, even though this choice is questionable and aggregation could also be used to reduce the computational burden of the QP solution in case this would be the predominant on the calculation of $\varphi()$

- in some applications it is easy to obtain an upper bound $\bar{\varphi}$ to the optimal value $\varphi(\boldsymbol{y}^*)$ while calculating $\varphi(\boldsymbol{y})$: the Bundle code must be able to exploit this information, the best possible way being that of inserting in β a null subgradient with linearization error $\bar{\varphi} - \varphi(\bar{\boldsymbol{y}})$ (provided that the QP solver is capable of dealing with such a case, as it is the case of ours) making the CP model always bounded, but $\bar{\varphi}$ can be also taken into account to tune several critical parameters of the algorithm such as ε_{LS} or $\bar{\varepsilon}_L$
- calculating $\varphi(\boldsymbol{y})$ (i.e. solving $(P_{\boldsymbol{c}\text{-}\boldsymbol{yA}})$) may be hard, hence it may be necessary to resort to faster inexact algorithms that give approximate solutions: in real applications, it may be impossible even to know any constant estimate of the error, but however the function can still be "maximized" (in the sense of going towards a hopefully almost optimal solution) if enough care is taken in identifying the inconsistencies that have been introduced by the errors (typically, some α_i will at a certain point turn negative) and correcting them (scaling α to regain nonnegativity)
- in some applications it may be possible to produce multiple [ε-]subgradient (corresponding to different [ε-]optimal solutions of $(P_{\boldsymbol{c}\text{-}\boldsymbol{yA}})$) of $\varphi()$ for a single point $\boldsymbol{y}$: this usually happens when the calculation of $\varphi()$ is by far the most expensive task within the algorithm, hence it is be sensible to provide the Bundle with all the possible information
- an important (and often disregarded) characteristic of Bundle algorithms when applied to polyhedral functions, is that, together with the optimal dual solution $\boldsymbol{y}^*$, they can provide the optimal primal solution of (M) as $\boldsymbol{x}_{\theta^*} = \Sigma_{i \in B} \boldsymbol{x}_i \theta_i^*$, θ^* being the optimal solution of the last $(\Delta_{\beta t})$ and $\boldsymbol{x}_i$ being the primal solution associated to $\boldsymbol{g}_i$: since Bundle codes are usually constructed as "black boxes", in order to make this precious information available to the outside, some sort of "naming convention" (the indices i) must be established

Another important feature of a Bundle code is the capability of dealing with constraints on the dual ($\boldsymbol{y}$) space: actually, only very simple constraints (box constraints or even nonnegativity ones) are needed to cope with most of the real applications, such as Lagrangean relaxations of inequality constraints, hence one of the most viable strategies is to demand all the task to the QP solver, by making the constraints apparent to it. In other words, if the QP solver is capable of solving the extended problem

$$(\Pi'_{\beta t}) \qquad \max_{\boldsymbol{d},v} \{\, v - 1/(2t) \| \boldsymbol{d} \|^2 : v \le \boldsymbol{g}_i \cdot \boldsymbol{d} + \alpha_i \;\; i \in \beta \,,\, l \le d \,\}$$

then all the above methods without few modifications can be extended to solve

$$(D') \qquad \max_{\boldsymbol{y}} \{\, \varphi(\boldsymbol{y}) : \boldsymbol{y} \ge \boldsymbol{0} \,\}$$

since the constraints $l \le d$ may ensure that $\bar{\boldsymbol{y}} + \boldsymbol{d} \ge \boldsymbol{0}$: actually, the same is also possible if the QP solver is capable of (efficiently) deal with constraints of the form $0 \le d_i \; i \in \chi$, where χ is an arbitrary subset of the variables, since then an active-set strategy can be used; in both cases (but expecially in

the second) some care must be taken in choosing the stepsizes for the line search procedures and in other parts of the algorithm, but nothing really bad happens. Both alternatives have been used in our implementation: the [Fr95] code employs a two-level active-set strategy that has proven to be particularly effective for the kind of reoptimization arising within a Bundle code, using the set χ (that is usaully fairly stable) of "active" non-negativity constraints to diminish the dimensionality of the problem. However, constraints $\boldsymbol{y} \in Y$ can be accounted for in almost the same way, as long as Y is a polyhedral set and thq QP solver can deal with general linear constraints on $\boldsymbol{d}$, as is the case for [Fr95]; we remark that, although theoretically possible, dealing with "complex" sets Y may however be exceedingly expensive, since for all the specialized QP solvers every linear constraint of the set has the same "cost" as a subgradient, unless some smart active set strategy can be employed to keep the set of "active" constraints small.

6. Preliminary computational results and Conclusions

We briefly report some preliminary computational results to show that the techniques introduced in §4 may be used to enhance the performances of a Bundle algorithm when dealing with a polyhedral function: we have tested MBT and BLS against our implementations of the classical [SZ92] and [Ki90] algorithms, on instances coming out from real applications rather than from academic test examples, that according to our experience are representative of some typical situation that Bundle codes have to face in practice. The tests have been performed within two experimental Bundle codes for constrained and unconstrained optimization: the former has been developed in the context of the solution of Multicommodity Min Cost Flow Problems [FG96], while the latter is intended as a general "black box" for nondifferentiable optimization, and has already been successfully used within several larger projects, such as [MP95]; all the tested algorithms have been implemented as "slight" variants of the same generic Bundle approach, showing how a solid set of basic tools can help in rapidly building and testing different variants of the same approach. We stress the fact that those results are preliminary (a more extensive study on efficiency of Bundle methods will be the subject of a following paper), and are just intended to convey the idea that it is possible to construct efficient Bundle codes for optimization of polyhedral functions, by building on the top of the theory presented in the present paper: furthermore, we want to show that some of the ideas presented herein for assembling new variants of Bundle methods can have a real impact on the practical performances of the codes.

We start with the comparison of MBT with the algorithm of [SZ92] (called BT) and that of [Ki90] (called KB) for the solution of Lagrangean Duals in the context of finding upper bounds for the Quadratic Semi-Assignment Problem (QSAP): we have tested the three algorithms on instances of different sizes, and the results show one of the typical difficulties of Bundle codes. For both BT and KB, the main algorithmic parameters (m_1 , m_2 ...) were somehow hand-tuned, but are analogous to those of the original papers; the required stopping criteria were $\varepsilon_L=10^{-3}$ and $\varepsilon_Q=10^{-4}$, and MBT was run with exactly the same parameters as BT, so that the only difference between the two codes is the $\bar{\varepsilon}_L$ adjustment mechanism; each entry in the table corresponds to several problems of approximately the same size, and the results (expressed in terms of number of $\varphi()$ evaluations required for convergence) have been averaged, so that there actually exist instances in which MBT is (a little) slower than its competitors.

N. Vars.	MBT	BT	KB
51	184	188	213
97	297	307	403
455	853	1901	2000*
606	1175	2000*	2000*
873	1432	3600*	3600*
1224	2370	3600*	3600*

Table 1: Number of φ() evaluations for the unconstrained problems

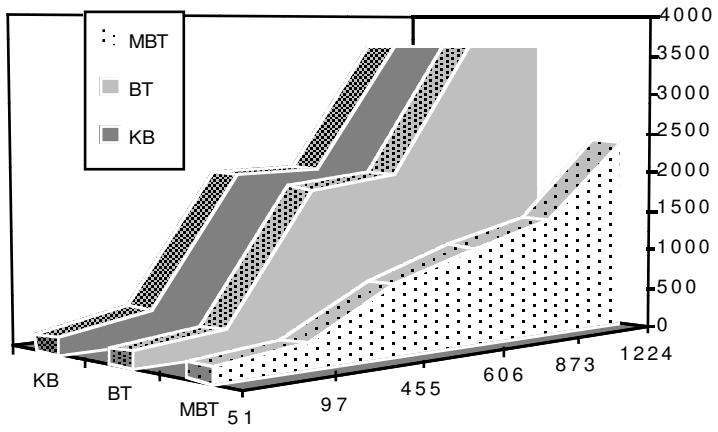

Chart 1: Results of Table 1

Some remarks are in order: in all of the entries of Table 1 marked with (*), the code terminated because of the extra stopping criterion based on a maximum fixed number of iterations, i.e. for problems over a certain size BT and KB were just unable to converge within that many iterations, even though they actually reached a value of φ() comparable (and even larger) than that obtained by MBT; this situation is typical of large problems, in which the algorithms spend a relevant fraction of their running time to certify the optimality, and rise several questions on what kind of result should be considered acceptable when a nondifferentiable function has to be maximized. In fact, there can be applications in which an optimal solution of (D) is not needed or practically impossible to reach, for instance because of the calculation of φ() being very expensive, so that a limited number of iterations is available and the best code is the one which reaches the higher value of φ(), no attempt to obtain a certified precision being made: for this purpose, all the three algorithms are comparable, and MBT can be also slower due to the $\bar{\varepsilon}_L$ adjustment mechanism; however, when convergence is important, for instance because a (quasi-)feasible primal solution is needed, the "long term memory" implemented in MBT really seems to play a relevant role. Some other qualitative observations can be made by looking more closely at the behavior of the algorithms: typically, very few *t* augments *and* decreases are done, both due to the curved-search mechanism and due to the interpolation heuristic, so that *t* is rather stable during the whole execution; this doesn't seem to be a nice feature, and we interpret this failure as an effect of the extremely "kinky" nature of those polyhedral functions, that can have billions of facets, contrasting with the "quasi-smooth" nature of nondifferentiable functions arising in different

applications and enforcing our claim that specialized approaches are needed for this class of functions.

The second comparison involved LBT and a modified version of BT capable of handling $\boldsymbol{y} \geq \mathbf{0}$ constraints (KB is supposed to be equivalent to BT), both used for the solution of a set of Multicommodity Min Cost Flow problems using a Bundle-based decomposition algorithm; although still of relevant dimensions, those problems are much easier than the previous ones, since many $y_i \geq 0$ constraints are active at $\boldsymbol{y}^*$, thus reducing the dimension of the problem.

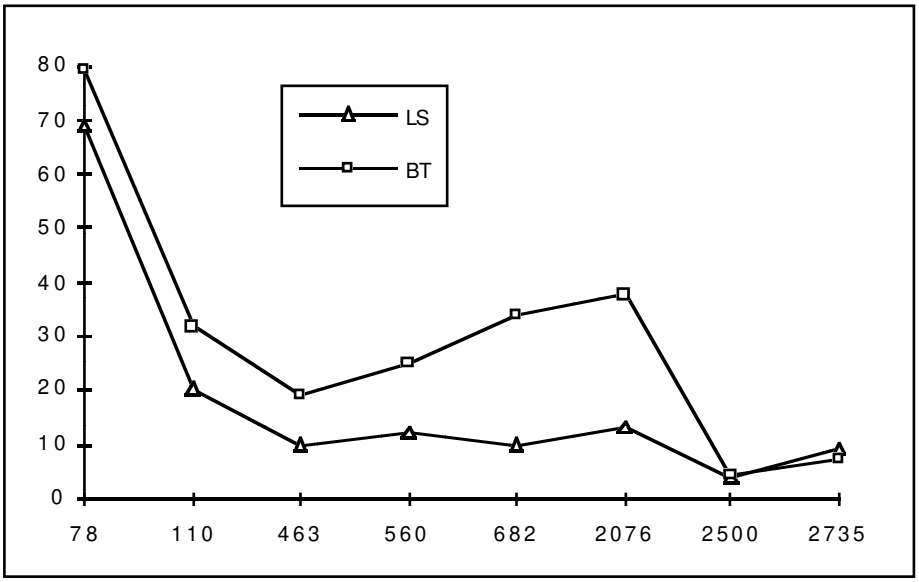

Chart 2: BT against LS φ() evaluations count

Here, the size of the problem is taken to be the number of arcs times the number of commodities (/100) of the MMCF problem, that should be a reasonable measure of the difficulty of the problem itself, and each point is obtained as the average of several subproblems of similar size; however, the actual difficulty of a problem also depends on how "tight" the generalized upper bound constraints are, and this probably produces the shape of Chart 2 above. These results show that, on some class of problems, LS can converge with (an average of) 2 to 3 times less φ() evaluations than BT, but they have to be taken with some care: in fact, even though the parameters of both the algorithms were hand-tuned for the class of instances of interest, and the stopping tolerances $\varepsilon_L = 10^{-6}$ were equal, it is not clear whether the superior performances of LS are due to its long term memory mechanism or to the fact that the line search mechanism helps in handling the constraints; furthermore, the implementation of BT we used was primarily designed to closely match the unconstrained version, and different choices may significantly affect the resulting performances.

Therefore, our computational experience seems to support our claim that specialized variants of Bundle algorithms can (and must) be constructed in order to efficiently cope with the maximization of polyhedral functions arising in many different applications within continuous and combinatorial optimization; in the last 20 years, a the large amount of research has been devoted to the theoretical and practical development of efficient Bundle methods for nondifferentiable optimization: we believe that time has come for making this body of knowledge available to the many practitioners and experts dealing with the optimization of polyhedral functions, under the form of efficient and flexible tools to be easily used and customized within larger optimization algorithms.

Acknowledgments

This paper is dedicated to the memory of Prof. Paolo Carraresi, recently died in his 48th.
We wish to acknowledge his invaluable contributions to this work and especially to our scientific and human preparation: we will always be in debt to his brilliant ideas, to his hard-worker example and especially to his high spirit and joy of life.

References

[Ba93] C. BARNHART "**Dual Ascent Methods for Large-Scale Multicommodity Flow Problems**" *Nav. Res. Log.* 40(3), p. 305-324, 1993

[BNS87] A. BIHAIN, V.H. NGUYEN AND J.J. STRODOT "**A Reduced Subgradient Algorithm**" *Math. Prog. Study* **30**, p. 127-149, 1987

[BGL95] J.F. BONNAIS, J.C. GILBERT, C. LEMARÉCHAL AND C.A. SAGASTIZÁBAL "**A Family of Variable Metric Proximal Methods**" *Math. Prog.* **68**, p. 56-94, 1995

[BS93] C. BARNHART, Y. SHEFFY "**A Network-Based Primal-Dual Heuristic for Multicommodity Network Flow Problems**" *Trans. Science* **27**(2), p. 15-47, 1993

[CGN95] P. CARRARESI, L. GIRARDI and M. NONATO **"Network Models, Lagrangean Relaxations and Subgradient Bundle Approach in Crew Scheduling Problems"** in *Computer Aided Scheduling of Public Transport* (J. Paixao ed.), Lecture Notes in Economical and Mathematical Systems, Springer-Verlag, p. 188-212, 1995

[CLM93] T.J CARPENTER, I.J. LUSTIG, J.M. MULVEY and D.F. SHANNO, **"Higher-order Predictor-Corrector Interior Point Methods with Application to Quadratic Objectives"**, *SIAM J. on Optimization*, **3**(4), p. 696-725, 1993

[DP95] P. DHARMA, J.A. SHAPIRO "**Augmented Lagrangeans for Linearly Constrained Optimization Using AMPL**" *CIV* **518** Final Report, 1995

[EGM95] L. EQUI, G. GALLO, S. MARZIALE and A. WEINTRAUB **"A Combined Transportation and Scheduling Problem"** *Technical Report* TR-**5/95**, Dipartimento di Informatica, Università di Pisa, 1995

[FG96] A. FRANGIONI, G. GALLO "**A Bundle type Dual-ascent Approach to Linear MultiCommodity Min Cost Flow Problems**" *Technical Report* **TR 01/96**, Dipartimento di Informatica, Università di Pisa, 1996

[FNS75] M.L. FISHER, W.D. NORTHUP AND J.F. SHAPIRO "**Using Duality to Solve Discrete Optimization Problems: Theory and Computational Experience**" *Math. Prog. Study* **3**, p. 56-94, 1975

[Fr95] A. FRANGIONI "**Solving Semidefinite Quadratic Problems Within Nonsmooth Optimization Algorithms**" *Technical Report* **TR 10/95**, Dipartimento di Informatica, Università di Pisa, 1995 (to appear on *Computers & O.R.)*

[Ge69] A.M. GEOFFRION "**Duality in Nonlinear Programming: a Simplified Applications-Oriented Development**" in *Prespective on Optimization: a collection of expository articles* (A.M. Geoffrion ed.), Addison-Welsey, 1972

[GHV92] J.L. GOFFIN, A. HAURIE AND J.P. VIAL "**Decomposition and Nondifferentiable Optimization with the Projective Algorithm**" *Mgmt. Sci.* **38**(2), p. 284-302, 1992

[GM91] M. GAUDIOSO, M.F. MONACO "**Quadratic Approximations in Convex Nondifferentiable Optimization**" *SIAM J. Control and Optimization* **29**(1), p. 58-70, 1991

[GM92] M. GAUDIOSO, M.F. MONACO "**Variants to the Cutting Plane Approach to Convex Nondifferentiable Optimization**" *Optimization* **25**, p. 66-75, 1992

[GM94] M. GAUDIOSO, M.F. MONACO "**The Newton direction and the cutting plane**" *EJOR* **73**, p. 172-174, 1994

[GHV93] J.L. GOFFIN, A. HAURIE, J.P. VIAL AND D.L. **ZHU** "**Using Central Prices in the Decomposition of Linear Programs**" *EJOR* **64**, p. 393-409, 1993

[Gr72] R.C. GRINOLD "**Steepest Ascent for Large Scale Linear Programs**" *SIAM Review* **14**(3), p. 447-464, 1972

[HL93] J-B. HIRIART-URRUTY, C. LEMARÉCHAL "**Convex Analysis and Minimization Algorithms II**" *A Series of Comprehensive Studies in Mathematics*, vol. **306**, Springer-Verlag, 1993

[HWC74] M. HELD, P. WOLFE AND H. CROWDER "**Validation of Subgradient Optimization**" *Math. Prog.* **6**, p. 62-88, 1974

[IST94] A.N. IUSEM, B.F. SVAITER AND M. TEBOULLE "**Entropy-like Proximal Methods in Convex Programming**" *Math. of O.R.* **19**(4), p. 790-814, 1994

[Ki85a] K.C. KIWIEL "**Methods of Descent for Nondifferentiable Optimization**" *Lecture Notes in Mathematics*, vol. **1133**, Springer-Verlag, 1985

[Ki85b] K.C. KIWIEL "**An Algorithm for Linearly Constrained Convex Nondifferentiable Minimization Problems**" *J. of Mathematical Analisys and Applications* **105**, p. 452-465, 1985

[Ki86] K.C. KIWIEL "**A Method for Solving Certain Quadratic Programming Problems Arising in Nonsmooth Optimization**" *IMA J. Numer. Anal.* **6**, p. 137-152, 1986

[Ki89] K.C. KIWIEL "**A Survey of Bundle Methods for Nondifferentiable Optimization**" *in Mathematical Programming and its Applications* (M. Iri and K. Tanabe eds.), KTK Scientific Publishers, 1989

[Ki90] K.C. KIWIEL "**Proximity Control in Bundle Methods for Convex Nondifferentiable Optimization**" *Math. Prog.* **46**, p. 105-122, 1990

[Ki91a] K.C. KIWIEL "**Exact Penalty Functions in Proximal Bundle Methods for Constrained Convex Nondifferentiable Optimization**" *Math. Prog.* **52**, p. 285-302, 1991

[Ki91b] K.C. KIWIEL "**A Restricted Step Proximal Bundle Method for Nonconvex Nondifferentiable Optimization**" *in Proc. Nonsmooth Optimization: Methods and Applications - Erice* (F. Gianessi ed.), Gordon and Breach Science Publishers, 1991

[Ki91c] K.C. KIWIEL "**A Tilted Cutting Plane Proximal Bundle Method for Convex Nondifferentiable Optimization**" *Op. Res. Letters* **10**, p. 75-81, 1991

[Ki94] K.C. KIWIEL "**A Cholesky Dual Method for Proximal PIecewise Linear Programming**" *Numer. Mat.*

68, p. 325-340, 1994

[Ki95] K.C. KIWIEL "**Proximal Level Bundle Methods for Convex Nondifferentiable Optimization, Saddle-Point Problems and Variational Inequalities**" *Mathematical Programming* **69**, p. 111-147, 1995

[KS77] J. KENNINGTON, M. SHALABY "**An Effective Subgradient Procedure for Minimal Cost Multicommodity Flow Problems**" *Mgmt. Sci.* **23**, p. 994-1004, 1977

[KR92] J. KREIMER, R.Y. RUBINSTEIN "**Nondifferentiable Optimization via Smooth Approximation: General Analytical Approach**" *Annals of O.R.* **39**, p. 97-119, 1992

[KU93] S. KIM, B. UM "**An Improved Subgradient Method for Constrained Nondifferentiable Optimization**" *Op. Res. Letters* **14**, p. 61-64, 1993

[Le75] C. LEMARÉCHAL "**Nondifferentiable Optimization: Subgradient and ε-Subgradient Methods**" *Lecture Notes in Optimization and Operations Research*, vol. **117**, Springer-Verlag, 1975

[Le77] C. LEMARÉCHAL "**Bundle Methods in Nonsmooth Optimization**" *in Nonsmooth Optimization*, vol. **3** of IIASA Proceedings Series (C. Lemaréchal, R. Mifflin eds.), Pergamon Press, 1977

[Le89] C. LEMARÉCHAL "**Nondifferentiable Optimization**" *in Handbooks in Opns. Res. and Mgmt. Sci.* vol. **1**: Optimization (G.L. Nehmauser, A.H.G. Rinnoy Kan and M.J. Todd eds.), North-Holland, 1989

[Le91] C. LEMARÉCHAL "**Lagrangean Decomposition and Nonsmooth Optimization: Bundle Algorithm, Prox Iteration, Augmented Lagrangean**" *in Proc. Nonsmooth Optimization: Methods and Applications - Erice* (F. Gianessi ed.), Gordon and Breach Science Publishers, 1991

[LGM93] R. DE LEONE, M. GAUDIOSO AND M.F. MONACO "**Nonsmooth Optimization Methods for Parallel Decomposition of Multicommodity Flow problems**" *Annals of O.R.* **44**, p. 299-311, 1993

[LNN95] C. LEMARÉCHAL, A. NEMIROVSKII AND Y. NESTEROV "**New Variants of Bundle Methods**" *Mathematical Programming* **69**, p. 111-147, 1995

[LSB81] C. LEMARÉCHAL, J.J. STRODIOT AND A. BIHAIN "**On A Bundle Algorithm for Nonsmooth Optimization**" *in Nonlinear Programming* **4**, (O.L. Mangasarian, R.R. Meyer and S.M. Robinson eds.), Academic Press, 1981

[Me90] D. MEDHI "**Parallel Bundle-Based Decomposition for Large-Scale Structured Mathematical Programming Problems**" *Annals of O.R.* **22**, p. 101-127, 1990

[Me94] D. MEDHI "**Bundle-Based Decomposition for Large-Scale Convex Optimization: Error Estimate and Application to Block-Angular Linear Programs**" *Math. Prog.* **66A**(1), p. 79-102, 1994

[Mi77] R. MIFFLIN "**A Feasible Descent Algorithm for Linearly Constrained Least Squares Problems**" *in Nonsmooth Optimization*, vol. **3** of IIASA Proceedings Series (C. Lemaréchal, R. Mifflin eds.), Pergamon Press, 1977

[Mi82] R. MIFFLIN "**A Modification and an Extension of Lemarechal's Algorithm for Nonsmooth Minimization**" *Math. Prog. Study* **17**, p. 77-90, 1982

[Mi91] R. MIFFLIN "**Ideas for Developing a Rapidly Convergent Algorithm for Nonsmooth Optimization**" *in Proc. Nonsmooth Optimization: Methods and Applications - Erice* (F. Gianessi ed.), Gordon and Breach

Science Publishers, 1991

[Mo94] M.F. MONACO "**Recenti Sviluppi nei Metodi Numerici per l'Ottimizzazione Nondifferenziabile**" *in Metodi di Ottimizzazione per le Decisioni* (G. Di Pillo ed.), Masson editoriale ESA, 1994

[MP95] F. MALUCELLI, D. PRETOLANI "**Lower Bounds for the Quadratic Semi-Assignment Problem**" *EJOR* **83**(2), 1995

[Po77] B.T. POLJAK "**Subgradient Methods: A Survey of Soviet Research**" *in Nonsmooth Optimization*, vol. **3** of IIASA Proceedings Series (C. Lemaréchal, R. Mifflin eds.), Pergamon Press, 1977

[Po83] M. POWELL "**ZQPCVX: A FORTRAN Subroutine for Convex Quadratic Programming**" *Report DAMTP* **1983/NA17**, University of Cambridge, 1983

[Ro70] R.T. ROCKAFELLAR "**Convex Analysis**" Princeton University Press, 1970

[SM91] G.L. SHULTZ, R.R. MEYER "**An Interior Point Method for Block-Angular Optimization**" *SIAM J. on Optimization*, **1**(4), 1991

[SZ92] H. SCHRAMM, J. ZOWE "**A Version of the Bundle Idea for Minimizing a Nonsmooth Function: Conceptual Idea, Convergence Analysis, Numerical Results**" *SIAM J. Optimization* **2**(1), p. 121-152, 1992

[Wo75] P. WOLFE "**A Method of Conjugate Subgradients for Minimizing Nondifferentiable Functions**" *Math. Prog. Study* **3**, p. 145-173, 1975

[ZTT84] A. ZAHORIK, L.J. THOMAS AND W.W. TRIGEIRO "**Network Programming Models for Production Scheduling in Multi-Stage, Multi-Item Capacitated Systems**" *Mgmt. Sci.* **30**, p. 308-325, 1984