Math. Program., Ser. A (2009) 118:177–206
DOI 10.1007/s10107-007-0187-4

FULL LENGTH PAPER

An inexact bundle variant suited to column generation

K. C. Kiwiel · C. Lemaréchal

Received: 24 October 2006 / Accepted: 7 August 2007 / Published online: 15 September 2007

Abstract We give a bundle method for constrained convex optimization. Instead of using penalty functions, it shifts iterates towards feasibility, by way of a Slater point, assumed to be known. Besides, the method accepts an oracle delivering function and subgradient values with unknown accuracy. Our approach is motivated by a number of applications in column generation, in which constraints are positively homogeneous—so that zero is a natural Slater point—and an exact oracle may be time consuming. Finally, our convergence analysis employs arguments which have been little used so far in the bundle community. The method is illustrated on a number of cutting-stock problems.

Keywords Nondifferentiable optimization · Convex programming · Proximal bundle methods · Approximate subgradients · Column generation · Cutting-stock problem

Mathematics Subject Classification (2000) 65K05 · 90C25 · 90C27

Research supported by INRIA New Investigation Grant "Convex Optimization and Dantzig–Wolfe Decomposition".

K. C. Kiwiel
Newelska 6, 01-447 Warsaw, Poland
e-mail: kiwiel@ibspan.waw.pl

C. Lemaréchal (✉)
INRIA, 655 avenue de l'Europe, Montbonnot, 38334 St Ismier, France
e-mail: claude.lemarechal@inria.fr

1 Introduction

We consider the convex constrained minimization problem

$$\inf f(u), \quad u \in C, \quad h(u) \leqslant 0; \tag{1.1}$$

here C is a "simple" closed convex set in the Euclidean space $\mathbb{R}^m$ (typically a polyhedron); $f(\cdot)$ is a "simple" convex real-valued function (typically linear, or quadratic); $h(\cdot)$ is also a convex real-valued function,[1] but only known via an oracle which delivers appropriate information at any given $u \in C$.

The present paper relies upon the assumption that a *Slater point*

$$u^0 \in C \quad \text{such that} \quad h(u^0) < 0 \tag{1.2}$$

exists and *is available*; motivating applications are given in Sects. 3.2–3.3.

We are interested in algorithms of the *cutting-plane* type, whose building bricks are *linearizations* of $h(\cdot)$, i.e., affine functions $\ell(u) = ua - \gamma$ minorizing $h(u)$. At the current iteration k of such an algorithm, the oracle has been called at a number of trial points $u^1, \ldots, u^k$ in C, and has returned the corresponding couples $(h^1, a^1), \ldots, (h^k, a^k)$ in $\mathbb{R} \times \mathbb{R}^m$. Normally, $h^j = h(u^j)$ and $a^j \in \partial h(u^j)$ denote the (exact) constraint value and a subgradient at u^j. In this paper, the oracle is allowed to be noisy: we assume for all j

$$h^j = h(u^j) - \eta^j \quad \text{and} \quad a^j \in \partial_{\eta^j} h(u^j), \quad \text{with } \eta^j \geqslant 0, \tag{1.3}$$

where the inaccuracies η^j are unknown, and need not go to zero; Sect. 5.4 will specify the influence of large inaccuracies on the quality of the algorithm.

The above notation introduces the η-subdifferential[2]

$$\partial_\eta h(u) := \{a : h(\cdot) \geqslant h(u) - \eta + (\cdot - u)a\}. \tag{1.4}$$

As far as cutting planes are concerned, each (h^j, a^j) from the oracle defines the linearization

$$u \mapsto \bar{h}^j(u) := h^j + (u - u^j)a^j, \tag{1.5}$$

and the η^j-subgradient inequality gives for all $u \in \mathbb{R}^m$

$$h(u) \geqslant h(u^j) - \eta^j + (u - u^j)a^j = h^j + (u - u^j)a^j = \bar{h}^j(u). \tag{1.6}$$

In this context, the general bundle methodology [14, Sect. XV.3] maintains

[1] In this paper, we will systematically use notation such as $f(\cdot), h(\cdot), \ldots$ for *functions*, while $f, h, \ldots$ will be reserved to particular *values* of such functions.

[2] For reasons to come in Sect. 3 below, u and a are considered as row and column vectors respectively: a will be a column of an $m \times n$ constraint matrix A and u will be a multiplier vector.

- a model $\check{h}^k(\cdot)$ of $h(\cdot)$, which must satisfy

$$\check{h}^k(u) \leqslant h(u) \quad \text{for all } u \in C, \tag{1.7}$$

- a stability center $\hat{u}^k$,
- a stability parameter $t^k > 0$,

and the next reference point u^{k+1} is the optimal solution of

$$\inf f(u) + \tfrac{1}{2t^k}|u - \hat{u}^k|^2, \quad u \in C, \quad \check{h}^k(u) \leqslant 0. \tag{1.8}$$

In fact, $\check{h}(\cdot) := \check{h}^k(\cdot)$ is piecewise linear [so (1.8) is typically a quadratic programming problem]; as such, it can be written for some finite index set J^k:

$$\check{h}(u) = \max\{ua^j - \gamma^j : j \in J^k\}, \tag{1.9}$$

where each (γ^j, a^j) lies in $\mathbb{R} \times \mathbb{R}^m$; we will call *bundle* the data $\{(\gamma^j, a^j)\}_{j \in J^k}$ characterizing $\check{h}(\cdot)$. The affine functions in (1.9) are linearizations of $h(\cdot)$. They can be those of (1.5), with $j \in \{1, \ldots, k\}$ and $\gamma^j := u^j a^j - h^j$; note that (1.6) then guarantees (1.7). However, Sect. 2.3 below will introduce "exogeneous" linearizations, through the operation of *aggregation*.

Remark 1.1 We have introduced two ways for characterizing an affine function such as $\bar{h}^j(\cdot)$:

- (1.9) is the natural way; it uses the constant term γ^j, which will be useful for the applications in Sect. 3;
- (1.5) rather translates the origin to u^j, which is useful for the description and analysis of the algorithm; we will see in Sect. 2.4 that translating the origin to $\hat{u}$ is even more appropriate.

With the above notation, (1.8) can be more concretely written as

$$\inf f(u) + \tfrac{1}{2t^k}|u - \hat{u}^k|^2, \quad u \in C, \quad ua^j - \gamma^j \leqslant 0, \ \ j \in J^k. \tag{1.10}$$

Lemma 1.2 *Under assumption* (1.2), (1.8) *has a unique optimal solution* u^{k+1} *given by*

$$u^{k+1} = \hat{u}^k - t^k \hat{g}^k, \quad \textit{with} \quad \hat{g}^k := b^k + \mu^k \hat{a}^k + v^k, \tag{1.11}$$

where

- $b^k \in \mathbb{R}^m$ *is a subgradient of* f *at* u^{k+1},
- $\mu^k \geqslant 0$ *satisfies* $\mu^k \check{h}^k(u^{k+1}) = 0$,
- $\hat{a}^k \in \mathbb{R}^m$ *is a subgradient of* $\check{h}^k$ *at* u^{k+1},
- $v^k \in \mathbb{R}^m$ *lies in the normal cone* $\mathrm{N}_C(u^{k+1})$ *to* C *at* u^{k+1}.

With the explicit expression (1.9)*, we have in* (1.11)

$$\mu^k = \sum_{j\in J^k} \lambda^j \quad \textit{and} \quad \mu^k \hat{a}^k = \sum_{j\in J^k} \lambda^j a^j, \tag{1.12}$$

where the nonnegative multipliers λ^j *satisfy* $\lambda^j(u^{k+1}a^j - \gamma^j) = 0$.

Proof Because of (1.7), the Slater assumption is transmitted to (1.8), which has a unique optimal solution due to strong convexity of its objective. Then these statements are just the standard optimality conditions, see for example [34, Chap. 28]: a subgradient of the Lagrangian is opposite to the stated normal cone. Such a subgradient can be written $b + \frac{u-\hat{u}}{t} + \mu\hat{a}$ for (1.8) or $b + \frac{u-\hat{u}}{t} + \sum_j \lambda^j a^j$ for (1.10). □

This result reveals the crucial m-vectors $\hat{g}^k$ and $\hat{a}^k$. Up to the approximation $h(\cdot) \rightsquigarrow \check{h}^k(\cdot)$, $\hat{g}^k$ is a distinguished subgradient of the Lagrangian associated with (1.1) and the update formula $u^{k+1} = \hat{u}^k - t^k\hat{g}^k$ of (1.11) resembles a subgradient step with stepsize t^k, to minimize that Lagrangian. With respect to footnote 2, page 178, note that the subgradient $\hat{g}^k$ is a column; but $t^k\hat{g}^k$ should be viewed as a row. The whole business of convergence will be to drive $\hat{g}^k$ to 0. As for $\hat{a}^k$, it takes its importance for aggregation (Sect. 2.3), and also for Lagrangian relaxation, or rather column generation (Sect. 3.1).

The paper is organized as follows: Sect. 2 reviews the various points in the paper which make its originality; Sect. 3 is devoted to our motivating application: column generation; Sect. 4 states the algorithm, whose convergence is analyzed in Sect. 5 and interpreted in the primal space in Sect. 6; we conclude in Sect. 7 with numerical illustrations on cutting-stock problems.

2 Main ideas in the paper

We first proceed to outline the algorithm studied in this paper, by describing its current kth iteration. In this informal description, we will often drop the index k to alleviate notation; then the superscript "+" will stand for $k+1$.

2.1 Maintaining the stability center

The role of $\hat{u} := \hat{u}^k$ is to control a suitable balance between objective and constraint values. Our variant uses the Slater point (1.2) to take care of feasibility of each $\hat{u}$; as a result, the management of the stability center may disregard h-values and needs to check f-values only.

More precisely, having called the oracle at the new iterate u^+, we construct the *interpolated point*

$$\check{u}^k := u^0 + \check{\beta}^k(u^{k+1} - u^0) \quad \text{with} \quad \check{\beta}^k := \begin{cases} 1 & \text{if } h^{k+1} \leqslant 0, \\ \dfrac{-h^0}{h^{k+1} - h^0} & \text{otherwise.} \end{cases} \tag{2.1}$$

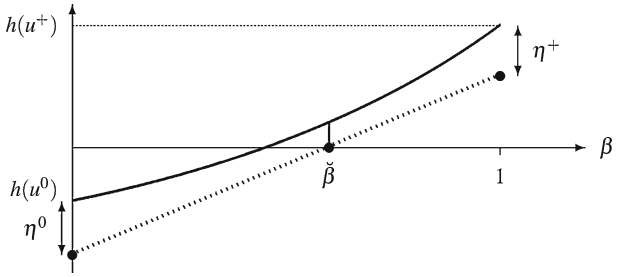

Fig. 1 Interpolation guarantees $h(\check{u}) \leqslant (1-\check{\beta})\eta^0 + \check{\beta}\eta^+$

Note here that $\check{\beta} \in [0, 1]$. The algorithm uses the (strictly negative) answer h^0 from the oracle, but a^0 need not be used. The next result is illustrated by Fig. 1.

Lemma 2.1 $h(\check{u}^k) \leqslant \check{\eta}^k := (1-\check{\beta}^k)\eta^0 + \check{\beta}^k\eta^{k+1} \leqslant \max\{\eta^0, \eta^{k+1}\}$.

Proof By convexity of $h(\cdot)$,

$$\begin{aligned} h(\check{u}) &\leqslant (1-\check{\beta})h(u^0) + \check{\beta}h(u^+) \\ &= (1-\check{\beta})(h^0+\eta^0) + \check{\beta}(h^+ + \eta^+) \\ &= h^0 + \check{\beta}(h^+ - h^0) + (1-\check{\beta})\eta^0 + \check{\beta}\eta^+, \end{aligned}$$

where we have used (1.3). Inspection of (2.1) shows that $h^0 + \check{\beta}(h^+ - h^0) \leqslant 0$ in either case, so the result follows. □

Thus, possible infeasibility of $\check{u}$ is controlled in the same way as the oracle's inaccuracy. In particular, $\check{u}$ is feasible in the case of an exact oracle.

Now let us assume for the moment that $\hat{u}$ is feasible in (1.8)—we will see that this is the case for an exact oracle. Then the *predicted decrease* $v := f(\hat{u}) - f(u^+)$ is positive (the case $v = 0$, i.e., $u^+ = \hat{u}$, is uninteresting; and Sect. 2.2 below will explain how to enforce positivity of v in the noisy case). As a result, the following strategy makes sense:

- Improve the current stability center if $f(\check{u})$ is "definitely smaller" than $f(\hat{u})$. More precisely, fix a coefficient $\kappa \in]0, 1[$ and set $\hat{u}^+ := \check{u}$ if $f(\hat{u}) - f(\check{u}) \geqslant \kappa v$; this is a *descent* step.
- If such is not the case, make a *null* step: $\hat{u}^+ := \hat{u}$.
- In either case, update $\check{h}(\cdot)$ and t and proceed to the next iteration.

The above interpolation idea is reminiscent of versions of the cutting-plane algorithm which also use points like u^0 and $\check{u}$; see [38] and the references therein. In these versions, however, the oracle is called at $\check{u}$, while our variant disregards $\check{u}$ for the oracle, which is called at u^+ only. However, Sect. 3.2 below will show that both approaches become closer in an important special case.

Except for the two recent filter methods [9,15], the existing bundle methods for constrained optimization require a merit function, for example an exact penalty $(f(u) + \pi \max\{0, h(u)\}$, as in SQP) or an "$F$-distance" $(\max\{f(u) - f(\hat{u}), h(u)\}$, as in the

method of centers). The earliest feasible-point methods of [32] and [18, Ch. 5], as well as the recent variant of [35], may converge slowly when their method-of-centers subproblems prevent approaching the constraint boundary fast. The penalty function methods of [19,20] tend to perform better; still, they require additionally that C be bounded, and may converge slowly when their penalty parameter estimates are too high. Finally, the level method of [29] (also see [24] and [2]) has good efficiency estimates when the set C is bounded, even if a Slater point does not exist; not surprisingly, therefore, it cannot benefit from the knowledge of a Slater point.

2.2 Coping with the noise

Suppose $t := t^k = +\infty$ in (1.8): there is no stabilizing term and (1.8) becomes a relaxation of (1.1), thanks to (1.7). If, in addition, we take $J^k = \{1, \ldots, k\}$, we obtain the pure cutting-plane algorithm[3] [5,16] used for standard column generation, see Sect. 3 below. This algorithm is little affected by inaccuracies: it just requires the oracle to provide linearizations satisfying (1.6). Accumulating linearizations eventually drives h^+ to 0; insofar as h^+ is close to $h(u^+)$ (depending on the noise in the oracle), a small h^+ implies that u^+ is approximately feasible, and therefore approximately optimal for (1.1).

This observation indicates that the noise can disturb our bundle algorithm only via the stabilizing term in (1.8). In fact, the new stability center $\hat{u}^+$ is constructed so as to be feasible in the *current* problem (1.8) (see Fig. 1). Nevertheless, $h(\hat{u}^+)$ may be positive and the property (1.6) need not guarantee $\hat{u}^+$ to stay feasible in *all* subsequent problems (1.8). When the stability center is not feasible, the predicted decrease may be negative: the algorithm is so much fooled that it seeks points worse than the stability center.

Our previous remark immediately suggests the cure, already proposed in [26]: just increase t in (1.8) in order to lessen the influence of the stabilizing term; do this until

- either v gives a safe descent test,
- or t is deemed large enough so that the whole algorithm can stop, just as the pure cutting-plane method would do.

Remark 2.2 We will see (end of Sects. 4 and 5.4) that more accurate answers from the oracle are required only at descent steps: large errors $h(u^j) - h^j$ at null steps do not deteriorate the final answer of the algorithm. □

To give a safe descent test, v should be "substantially" positive. Technically, it is convenient to require a decrease of the whole objective function in (1.8) from $\hat{u}$ to u^+: descent is tested only when

$$f(\hat{u}) - f(u^+) - \tfrac{1}{2t}|u^+ - \hat{u}|^2 \geqslant 0, \quad \text{i.e.,} \quad v \geqslant \tfrac{1}{2t}|u^+ - \hat{u}|^2; \tag{2.2}$$

otherwise t is simply increased and (1.8) is solved again, with the same $\hat{u}$ and $\check{h}(\cdot)$.

[3] When $t = +\infty$, (1.8) may have no solution; we skip this difficulty here.

Remark 2.3 (Bounding the objective) Let us mention here that no feasible $\hat{u}$ need ever be produced when the oracle is noisy; it may not be straightforward to bound from above the optimal value f^* of (1.1).

It is known that $f(u) + \pi \max\{0, h(u)\} \geqslant f^*$ for any $u \in C$ if π is large enough (larger than an optimal multiplier μ^*). Yet, such bounds assume some information about μ^*—and are corrupted by noise anyway.

However, assume that the oracle is also able to answer *upper* bounds, say $\tilde{h}^j \geqslant h(u^j)$. They can be inserted in the above exact penalty function, but better bounds can be obtained. In fact, introduce analogously to (2.1) the *upper interpolated point*

$$\tilde{u} := u^0 + \tilde{\beta}(u^+ - u^0) \quad \text{with} \quad \tilde{\beta} := \begin{cases} 1 & \text{if } \tilde{h}^+ \leqslant 0\,, \\ \dfrac{-\tilde{h}^0}{\tilde{h}^+ - \tilde{h}^0} & \text{otherwise} \end{cases}$$

and assume $\tilde{h}^0 < 0$. Then $\tilde{\beta} \in [0, 1]$ and $h(\tilde{u}) \leqslant 0$ by convexity, as in Lemma 2.1. This construction can be useful in applications, see Remark 3.5 below. □

Our algorithmic constructions and analysis of inaccuracies in the oracle extend to the constrained case the inexact linearization framework of [26,27]; for earlier related developments, see [13,17,22,33,37].

2.3 Managing the constraint model: the aggregate linearization

The management of $\check{h}(\cdot)$ should guarantee convergence in spite of possible non smoothness of $h(\cdot)$. To this aim the standard idea, which is used in the pure cutting-plane algorithm, is to accumulate information coming from the oracle (the "bundling" process): with the new linearization $\bar{h}^+(u) := h^+ + (u - u^+)a^+$—recall notation (1.5)—one sets $\check{h}^+(\cdot) := \max\{\check{h}(\cdot), \bar{h}^+(\cdot)\}$, i.e., $J^+ := J \cup \{+\}$. This results in storing all the (γ^j, a^j) in (1.10), which may become inconvenient or impossible when the iteration index k grows; the question is therefore: Which linearizations should $\check{h}^+(\cdot)$ be made from? To answer it, (1.7) should be kept in mind.

Naturally, the new couple (h^+, a^+) must appear in the new model: $J^+ \supset \{+\}$. As for information accumulation, it uses the set

$$\hat{J} := \{j \in J : u^+ a^j - \gamma^j = \check{h}(u^+)\} \tag{2.3}$$

of active linearizations at u^+. From standard convex analysis (see [14, Sect. VI.4.4 or Example VI.3.4] for example), the subdifferential of $\check{h}(\cdot)$ at u^+ is the convex hull of the corresponding slopes:

$$\partial\check{h}(u^+) = \left\{ \sum_{j\in\hat{J}} \alpha^j a^j \,:\, \alpha^j \geqslant 0,\ \sum_{j\in\hat{J}} \alpha^j = 1 \right\}. \tag{2.4}$$

By the definition of subgradient, the function $\ell(u) := \check{h}(u^+) + (u - u^+)a$ satisfies $\ell(\cdot) \leqslant \check{h}(\cdot)$ if $a \in \partial\check{h}(u^+)$. With reference to (1.9), this $\ell(\cdot)$ can be put in the form $\ell(u) = ua - \gamma$ and its constant term γ is easy to compute:

Lemma 2.4 *With $a \in \partial\check{h}(u^+)$, the above function $\ell(\cdot)$ has*

$$\begin{pmatrix}\gamma\\ a\end{pmatrix} = \sum_{j\in\hat{J}} \alpha^j \begin{pmatrix}\gamma^j\\ a^j\end{pmatrix} \quad \text{for some } \alpha \geqslant 0 \text{ with } \sum_{j\in\hat{J}} \alpha^j = 1\,. \tag{2.5}$$

Proof Because $u^+a^j - \gamma^j = \check{h}(u^+) = \ell(u^+) = u^+a - \gamma$ for all $j \in \hat{J}$, we obtain for any set α of convex multipliers

$$u^+\left(\sum_{i\in\hat{J}} \alpha^j a^j\right) - \sum_{i\in\hat{J}} \alpha^j\gamma^j = \check{h}(u^+) = \ell(u^+) = u^+a - \gamma.$$

This holds in particular for the α making up a—see (2.4). □

Then the bundling process distinguishes three cases:

(1) A descent step is made. Then the descent property is strong enough to imply convergence, even if J^+ is reduced to the singleton $\{+\}$.
(2) The constraint is not active in (1.8); more precisely, $\mu = 0$. Again, we may set $J^+ = \{+\}$ without impairing convergence.
(3) A null step is made and $\mu > 0$. Then Lemma 1.2 reveals the *aggregate linearization*

$$u \mapsto \bar{h}^-(u) = \bar{h}^{-k}(u) := \check{h}^k(u^{k+1}) + (u - u^{k+1})\hat{a}^k, \tag{2.6}$$

which satisfies $\bar{h}^-(\cdot) \leqslant \check{h}(\cdot)$. Indeed, $\bar{h}^-(\cdot)$ somehow gathers the whole information contained in the current bundle, entailing the memorization effect crucial for convergence; this is explained in [6, Sect. 4] for example.
Altogether we have $\max\{\bar{h}^-(u), \bar{h}^+(u)\} \leqslant h(u)$ for all u, which reveals a piecewise linear function satisfying (1.7): it is a valid candidate for the next model $\check{h}^+(\cdot)$. Taking this candidate as $\check{h}^+(\cdot)$ corresponds to the "minimal" set $\{-,+\}$ for J^+. A "maximal" J^+ would be $\{1,\ldots,k+1\}$, as in the pure cutting-plane algorithm. We therefore see that the new index set just has to satisfy

$$\{-k, k+1\} \subset J^{k+1} \subset \{-k, k+1\} \cup J^k. \tag{2.7}$$

No matter how J^+ is chosen as above, the result is a new model function satisfying—recall the notation (2.6), (1.5):

$$\max\{\bar{h}^{-k}(u), \bar{h}^{k+1}(u)\} \leqslant \check{h}^{k+1}(u) \leqslant h(u), \quad \text{for all } u \in C. \tag{2.8}$$

We conclude this section with a few remarks:

- A consequence of (2.5) is that the $\{-\}$-linearization is useless if J^+ already contains the whole of $\hat{J}$. When m is not too large, a reasonable choice is $J^+ = \hat{J} \cup \{+\}$. An even more sensible choice reduces $\hat{J}$ to the set of those j such that $\lambda^j > 0$ in Lemma 1.2; this is *linearization selection*, in which each J^k can be forced to have at most $m+1$ elements; see [18,27].
- In case (3), the software solving (1.10) usually provides the multiplier vector λ of (1.12), so $\hat{a}$ is readily available: just take $\alpha^j := \lambda^j/\mu$ in (2.5).
- Reducing J^+ to $\{+\}$ in cases (1), (2) is not recommended: the next iteration will be (close to) steepest descent, well known for its numerical inefficiency. Even when $\check{h}(u^+) < 0$ (in which case $\mu = 0$), bundling is probably worthwhile.
- This latter point suggests that aggregation might be desirable even if $\mu = 0$. For this, we can take any linearization of the form (2.6), where $\hat{a}$ is any convex combination of active a^j's at u^+.

2.4 Convergence analysis

For later use, we return in this section to the full k-notation. We start with elementary relations: the properties $b^k \in \partial f(u^{k+1})$, $\hat{a}^k \in \partial \check{h}(u^{k+1})$, $v^k \in \mathrm{N}_C(u^{k+1})$ in Lemma 1.2, and (1.7) give for all $u \in C$

$$f(u) \geqslant f(u^{k+1}) + (u - u^{k+1})b^k, \tag{2.9}$$

$$h(u) \geqslant \check{h}(u) \geqslant \check{h}(u^{k+1}) + (u - u^{k+1})\hat{a}^k, \tag{2.10}$$

$$0 \geqslant (u - u^{k+1})v^k. \tag{2.11}$$

Multiply (2.10) by $\mu^k \geqslant 0$, use complementarity slackness and sum up to obtain

$$\forall u \in C, \quad f(u) + \mu^k h(u) \geqslant f(u) + \mu^k \check{h}^k(u) \geqslant f(u^{k+1}) + (u - u^{k+1})\hat{g}^k. \tag{2.12}$$

In a bundle method, convergence of the "natural iterates" u^+ is not a relevant property: the actual candidates to solving (1.1) are rather the stability centers. This is why the optimality conditions of (1.8) or (1.10) are traditionally translated to $\hat{u}$:

Theorem 2.5 *With the notation of Lemma 1.2, set*

$$\begin{aligned} \hat{\varepsilon}^k &:= f(\hat{u}^k) - f(u^{k+1}) - (\hat{u}^k - u^{k+1})\hat{g}^k, \\ \hat{\delta}^k &:= \hat{\varepsilon}^k + \hat{u}^k \hat{g}^k. \end{aligned} \tag{2.13}$$

Then

$$\hat{\varepsilon}^k \geqslant -\mu^k h(\hat{u}^k). \tag{2.14}$$

Besides, for all u feasible in (1.8) [(e.g., feasible in (1.1))]:

$$f(u) \geqslant f(\hat{u}^k) - \hat{\varepsilon}^k + (u - \hat{u}^k)\hat{g}^k, \tag{2.15}$$

or equivalently

$$f(u) \geqslant f(\hat{u}^k) - \hat{\delta}^k + u\hat{g}^k. \tag{2.16}$$

Proof We use (2.12). First take $u = \hat{u}$ and arrange terms to obtain $\hat{\varepsilon} \geqslant -\mu\check{h}(\hat{u}) \geqslant -\mu h(\hat{u})$. Then take u feasible in (1.8): $\check{h}(u) \leqslant 0$ and $f(u) \geqslant f(u^+) + (u - u^+)\hat{g}$. Straightforward manipulations then give (2.15) and (2.16). Note that the feasible set of (1.8) contains the feasible set of (1.1) thanks to (1.7). □

Note that $\hat{\varepsilon}$ may be negative if $\hat{u}$ is not feasible in (1.8). In connection with Sects. 2.1 and 2.2, use (1.11) and (2.13): the predicted decrease is

$$v^k := f(\hat{u}^k) - f(u^{k+1}) = \hat{\varepsilon}^k + t^k|\hat{g}^k|^2. \tag{2.17}$$

As explained in Sect. 2.2, it is "substantially" positive when (2.2) holds, which has several equivalent expressions obtained by suitable manipulations based on (2.17):

$$v^k \geqslant \frac{1}{2t^k}|u^{k+1} - \hat{u}^k|^2, \quad v^k \geqslant -\hat{\varepsilon}^k, \quad \hat{\varepsilon}^k \geqslant -\frac{t^k}{2}|\hat{g}^k|^2. \tag{2.18}$$

When this holds, the descent test is

$$f(\hat{u}^k) - f(\check{u}^k) \geqslant \kappa v^k \tag{2.19}$$

with $\check{u}^k$ given by (2.1) and $\kappa \in]0, 1[$.

Traditional convergence analyses of bundle methods consist in showing that 0 is a cluster point of the sequence $\{\hat{\varepsilon}^k, \hat{g}^k\} \subset \mathbb{R} \times \mathbb{R}^m$; then (2.15) implies that $f(\hat{u}^k)$ is "asymptotically good". Here we use a slightly different argument, namely:

$$\text{The sequence } \{(\hat{\delta}^k, \hat{g}^k)\} \text{ has a cluster point } (\hat{\delta}, 0), \text{ with } \hat{\delta} \leqslant 0, \tag{2.20}$$

i.e., $\liminf_{k\to+\infty} \max\{\hat{\delta}^k, |\hat{g}^k|\} = 0$.

Remark 2.6 This argument goes back to [26] and has an interesting background in convex analysis. Call $\phi(\cdot)$ the actual objective function of (1.1) ($\phi(u) = f(u)$ if u is feasible, $+\infty$ otherwise) and admit that every $\hat{u}^k$ is feasible. Then write (2.16) as $u\hat{g}^k - \phi(u) \leqslant \hat{\delta}^k - \phi(\hat{u}^k)$ for all $u \in \mathbb{R}^m$ and take the supremum over u: $\phi^*(\hat{g}^k) \leqslant \hat{\delta}^k - \phi(\hat{u}^k)$ where $\phi^*(\cdot)$ is the convex conjugate of $\phi(\cdot)$. Finally take a subsequence stipulated by (2.20): knowing that $\phi^*(\cdot)$ is lower semicontinuous and that $\phi(\hat{u}^k)$ has a limit ($\phi(\hat{u}^k) = f(\hat{u}^k)$ is monotone),

$$\phi^*(0) \leqslant \liminf\ \phi^*(\hat{g}^k) \leqslant \limsup \phi^*(\hat{g}^k) \leqslant \hat{\delta} - \lim \phi(\hat{u}^k) \leqslant -\lim \phi(\hat{u}^k).$$

Remembering that $-\phi^*(0)$ is the infimum of $\phi(\cdot)$, this establishes that $\hat{u}^k$ is a minimizing sequence for (1.1).

With the traditional approach, the term $\hat{u}^k\hat{g}^k$ in (2.15) brings trouble if $\hat{u}^k$ is unbounded. □

Thus, f-values of the stability centers are assessed by (2.15) or (2.16). Their h-values are assessed by Lemma 2.1. Supposing that the latest descent step occurred at iteration K (i.e., after the $K+1$st oracle's call), feasibility of $\hat{u}$ depends on $\check{\eta}^K$, i.e., on η^0 and η^{K+1}; for example, $h(\hat{u}) \leqslant \eta^{K+1}$ if $h^{K+1} \leqslant 0$.

3 Motivation: column generation

Consider the following primal-dual pair of LP problems (c is an n-row, b is an m-column)

$$\min\ c\lambda, \quad A\lambda + b \geqslant 0, \quad \lambda \in \mathbb{R}^n_+, \tag{3.1}$$

$$\max\ -ub, \quad uA \leqslant c, \quad u \in \mathbb{R}^m_+, \tag{3.2}$$

where n is a huge number: then column generation is a method of choice. For $i \in \{1, \ldots, n\}$, let A_i denote column i of A. Setting $C := \mathbb{R}^m_+$, $f(u) := ub$ and

$$h(u) := \max_{i=1,\ldots,n} (uA_i - c_i) \tag{3.3}$$

clearly puts (3.2) in the form (1.1).

The possibility of an inaccurate oracle is useful in this framework. In fact (keeping Remark 2.2 in mind), the oracle in charge of solving (3.3) is allowed to compute an *arbitrary* $i = i^j$:

Proposition 3.1 *For given $u^j \in \mathbb{R}^m$ and $i^j \in \{1, \ldots, n\}$, set $h^j := u^j A_{ij} - c_{ij}$ and $a^j := A_{ij}$. Then*

$$\eta^j := h(u^j) - h^j \geqslant 0 \quad \textit{and} \quad a^j \in \partial_{\eta^j} h(u^j).$$

Proof The property $\eta^j \geqslant 0$ is obvious from the definition (3.3) of $h(u^j)$. Now consider the definition of $\partial_{\eta^j} h(u^j)$ in (1.4): for all u

$$h(u^j) - \eta^j + (u - u^j)a^j = h^j + (u - u^j)A_{ij} = u^j A_{ij} - c_{ij} + (u - u^j)A_{ij};$$

the last term is just $uA_{ij} - c_{ij}$, which is not larger than $h(u)$. □

As for the availability of a Slater point, it is application dependent; note that we may take $u^0 = 0$ in (1.2) if $c > 0$. A particularly interesting situation will be seen in Sect. 3.2 below.

In this section, we explain how our bundle method for the dual problem (3.2) can solve the primal problem (3.1). Again we drop the index k whenever possible.

3.1 Primal recovery

We proceed to show that the multiplier vector λ of Lemma 1.2 provides a good candidate for solving (3.1), once properly embedded in $\mathbb{R}^n$. In fact, the (γ^j, a^j)'s

in (1.9) or (1.10) connote the (c_i, A_i)'s in (3.3) or (3.2). The λ^j's of (1.12) define $\mu\hat{a}$ as well as

$$\mu\hat{\gamma} := \sum_{j\in J} \gamma^j \lambda^j; \tag{3.4}$$

then $\mu(\hat{\gamma}, \hat{a})$ connotes $(c\lambda, A\lambda)$ in (3.1).

A primal-dual optimal pair for (3.1), (3.2) is a $(\hat{\lambda}, \hat{u}) \in \mathbb{R}^n_+ \times \mathbb{R}^m_+$ satisfying

$$\hat{u}A \leqslant c, \quad A\hat{\lambda} + b \geqslant 0, \quad c\hat{\lambda} + \hat{u}b \leqslant 0, \tag{3.5}$$

where the last $\leqslant$-sign could just be replaced by $=$ (weak duality). In the construction of an optimal pair by our bundle method (remember from Sect. 2.4 that the candidate to dual optimality is the stability center $\hat{u}$), the next result deals with the second and third inequalities:

Lemma 3.2 *With* (3.4), (2.13) *and the notation of Lemma* 1.2, *there holds*

$$\mu\hat{\gamma} + \hat{u}b = \hat{\delta}, \tag{3.6}$$

$$\mu\hat{a} + b \geqslant \hat{g}. \tag{3.7}$$

Proof By the definition of normal cone, v in (1.11) is in complementarity with u^+: $0 \leqslant u^+ \perp (-v) \geqslant 0$, hence

$$b + \mu\hat{a} - \hat{g} \geqslant 0 \quad \text{and} \quad u^+(b + \mu\hat{a} - \hat{g}) = 0,$$

which gives (3.7) and $u^+(\mu\hat{a}) = u^+\hat{g} - u^+b$. Besides, complementarity slackness in (1.10) guarantees that $u^+a^j = \gamma^j$ if $\lambda^j > 0$, hence $u^+(\mu\hat{a}) = \mu\hat{\gamma}$; this gives $\mu\hat{\gamma} = u^+\hat{g} - u^+b$. Since $\hat{\delta} = (\hat{u} - u^+)b + u^+\hat{g}$ by (2.13), (3.6) follows. □

The role of the convergence property (2.20) for primal recovery is now clear: together with the constraint $\check{h}(u) \leqslant 0$, it aims at satisfying the three inequalities in (3.5). Assume for the moment that $J \subset \{1, \ldots, n\}$. Extending the λ^j's by zero gives $\hat{\lambda} \in \mathbb{R}^n$, which satisfies approximately the optimality conditions when the algorithm is close to convergence. Actually, J can be slightly more general:

Theorem 3.3 *Assume in* (1.9) *that the* (γ^j, a^j)*'s are linear combinations of the* (c_i, A_i)*'s in* (3.1):

$$\begin{pmatrix} \gamma^j \\ a^j \end{pmatrix} = \sum_{i=1}^n \alpha_i^j \begin{pmatrix} c_i \\ A_i \end{pmatrix} \quad \text{for each } j \in J\,. \tag{3.8}$$

With λ *of Lemma* 1.2, *define* $\hat{\lambda} \in \mathbb{R}^n$ *by*

$$\hat{\lambda}_i := \sum_{j\in J} \lambda^j \alpha_i^j \quad \text{for } i \in \{1, \ldots, n\}\,. \tag{3.9}$$

Then there holds

$$c\hat{\lambda} + \hat{u}b = \hat{\delta}, \quad A\hat{\lambda} + b \geqslant \hat{g}.$$

Proof Using successively the definition (3.9) of $\hat{\lambda}$, (3.8) and the definitions (1.12), (3.4) of $(\hat{\gamma}, \hat{a})$, we have

$$\begin{pmatrix} c\hat{\lambda} \\ A\hat{\lambda} \end{pmatrix} = \sum_{i=1}^{n} \hat{\lambda}_i \begin{pmatrix} c_i \\ A_i \end{pmatrix} = \sum_{j\in J} \lambda^j \sum_{i=1}^{n} \alpha_i^j \begin{pmatrix} c_i \\ A_i \end{pmatrix} = \sum_{j\in J} \lambda^j \begin{pmatrix} \gamma^j \\ a^j \end{pmatrix} = \mu \begin{pmatrix} \hat{\gamma} \\ \hat{a} \end{pmatrix}.$$

The result is then just Lemma 3.2. □

Naturally, α should be nonnegative to guarantee $\hat{\lambda} \geqslant 0$. In our framework, the α's actually form *convex* multipliers, coming into play when aggregating: in (2.7), $\hat{a} \in \partial\check{h}(u^+)$ is a convex combination of the a^j's making up $\check{h}(\cdot)$—see (2.4). The above result is therefore useful to deal with a bundle with negative indices. This will be seen more precisely in Sect. 6.

3.2 The positively homogeneous case

The main innovative feature of our method is the interpolation technique outlined in Sect. 2.1; arguably, its efficiency relies heavily on the choice of the Slater point: for example, would it not be a good idea to improve u^0 whenever a strictly better feasible point is found? Our technique, however, seems convenient in the particular instances of (3.1), (3.2) where c is the vector of all ones in R^n. Then the constraint in (1.1) has the form

$$h(u) = \sigma(u) - 1, \quad \text{where } \sigma(u) := \max_{i=1,\dots,n} uA_i. \tag{3.10}$$

Here come a few key observations:

(1) The above $\sigma(\cdot)$ is a positively homogeneous function of u: $\sigma(\beta u) = \beta\sigma(u)$ for all $\beta \geqslant 0$.
(2) An obvious Slater point is $u^0 = 0$, for which $h^0 := h(u^0) = -1$ is readily available.
(3) Assume $h^+ = h(u^+) \geqslant 0$ in (2.1) and use positive homogeneity ($\eta^0 = \eta^+ = 0$ and $h(\cdot)$ is affine with respect to β in Fig. 1, $\sigma(\cdot)$ is linear):

$$\sigma(\check{u}) = \sigma(u^0) + \check{\beta}\big(\sigma(u^+) - \sigma(u^0)\big) = \frac{1}{\sigma(u^+)}\sigma(u^+) = 1.$$

Thus, in the noiseless case, the candidate $\check{u}$ for the next stability center lies on the boundary of the feasible domain in (3.2).
(4) Of course, it is $\sigma(\cdot)$ which is computed by the oracle, and this computation can be inaccurate, as in the general case.

Property (3) above is very convenient and assesses the choice of $u^0 = 0$ as the interpolation center: first, optimal solutions of (3.2) should be sought on the boundary of its feasible domain; second, the stability centers are feasible (if the oracle is exact), so each $-\hat{u}b$ is a lower bound for the primal optimal value. These two features are a definite advantage of our approach, which can thus be called a "conic" variant.

Remark 3.4 Note an interesting consequence of positive homogeneity: suppose the oracle is called at a point βu^j, with $\beta > 0$. Then chances are that the oracle will find the same index i^j (see Proposition 3.1) that would be obtained at u^j: calling $(\sigma_\beta^j, a_\beta^j)$ its answer, we will have

$$\sigma_\beta^j = \beta\sigma^j \quad \text{and} \quad a_\beta^j = a^j.$$

Setting $\bar{\sigma}(\cdot) = \bar{h}(\cdot) + 1$ for $\bar{h}$ given in (1.5) and using $\sigma^j = u^j a^j$ gives

$$\begin{aligned}\bar{\sigma}_\beta^j(u) &= \sigma_\beta^j + (u - \beta u^j)a_\beta^j = \beta\sigma^j + (u - \beta u^j)a^j \\ &= \sigma^j + \beta\sigma^j + (u - u^j)a^j - \beta u^j a^j \\ &= \bar{\sigma}^j(u) + \beta\sigma^j - \beta\sigma^j = \bar{\sigma}^j(u).\end{aligned}$$

In other words: assuming that the oracle is reasonably deterministic, its answer at u^+ or $\check{u}$ of (2.1) produces the same linearization (1.5). □

3.3 Combinatorial applications

Linear programs with a constant cost row may not be so frequent. However, (3.1) may come from the Dantzig–Wolfe formulation of various combinatorial problems; λ is then an integer vector, and the constraint $\lambda \in \mathbb{N}^n$ is relaxed to $\lambda \geqslant 0$; see [42, Sect. 11.2]. The case $c_i \equiv 1$ occurs in the classical approach [12] of Gilmore and Gomory to the cutting-stock problem; Sect. 7 below gives an illustration. Then (3.10) is a knapsack problem, for which the possibility of an inexact oracle is particularly welcome. The same situation occurs in some relaxations of the graph-coloring problem [31], in which the oracle computes a maximum stable set. See also [4].

When c is a general positive vector, positive homogeneity can of course be recovered by modeling the constraint $uA \leqslant c$ as

$$h(u) := \max_{i=1,\ldots,n} \left(\frac{uA_i - c_i}{c_i}\right) = \max_{i=1,\ldots,n} \left(\frac{uA_i}{c_i}\right) - 1 \leqslant 0; \tag{3.11}$$

see [3,36]. This, however, implies that the oracle maximizing uA_i accommodates scaled columns of A.

Remark 3.5 Dual feasible points are useful to produce primal lower bounds. From this point of view, the $\check{u}$'s from (2.1) are useful if the oracle is exact; actually, $f(\check{u}) = -\check{u}b$ is just Farley's bound of [8], see also [1,7,30,39,40] (to realize this, compare (3.11) with the equation preceding the theorem in [8]).

If a branch and bound algorithm is used to approximate $h(u)$ in (3.11), upper bounds $\tilde{h}$ are available and convenient Farley-type bounds $f(\tilde{u}) = -\tilde{u}b$ can still be produced, via the interpolation mechanism mentioned in Remark 2.3. □

More generally, Dantzig–Wolfe formulations of combinatorial problems are

$$\min\ c\lambda + d\alpha, \quad A\lambda + B\alpha + b \geqslant 0, \quad \lambda \in \mathbb{N}^n,\ \alpha \in \{0,1\}^p$$

(see [40]). Their associate auxiliary problems (to maximize the Lagrangian, see [28]) are

$$\min_{\lambda \geqslant 0} (c - uA)\lambda + \min_{0 \leqslant \alpha \leqslant 1} (d - uB)\alpha - bu.$$

They result in a dual problem of the form (1.1), but where the objective function

$$f(u) = ub - \min_{0 \leqslant \alpha \leqslant 1} (d - uB)\alpha$$

is given through an oracle, just as the constraint $h(\cdot)$. Then (1.1) becomes a fully nonsmooth constrained optimization problem, for which there are a number of possibilities:

- It can be solved by standard versions of constrained bundle methods, as reviewed in Sect. 2.1.
- An additional variable can be introduced, say v, and (1.1) can be formulated as minimizing v, subject to the constraint $\max\{f(u) - v, h(u)\} \leqslant 0$.
- Our present variant can be tailored to this situation.

4 The inexact conic proximal bundle method

We now specify the algorithm outlined in Sect. 2. In our description, the model $\check{h}^k$ of (1.7) is abstract. It may have the particular form $\check{h}^k(\cdot) = \max_{j \in J^k} \bar{h}^j(\cdot)$, J^k being managed as described in Sect. 2.3; but this level of detail is not necessary in our description. The management of the stepsize t^k is also left vague. However we do describe the management of the stability center $\hat{u}^k$, as specified in Sect. 2.1.

The algorithm uses the Slater point u^0 of (1.2), a descent parameter $\kappa \in]0,1[$ and a lower bound $t_{\min} > 0$ for the stepsize; $K(\cdot)$ will mark descent iterations (at iteration k, the last descent iteration was the $K(k)$th one) and the flag NA will secure the noise-attenuation mechanism of Sect. 2.2 (during which a decrease of the stepsize is untimely).

Algorithm 4.1 An initial point $u^1 \in C$ and an initial stepsize $t^1 \geqslant t_{\min}$ are given.

Step 0 (*Initiation*). Call the oracle at u^1 to obtain h^1 and a^1 of (1.3). Choose a function $\check{h}^1(\cdot)$ satisfying (1.7). Compute $\breve{u}^0$ by (2.1) and set $\hat{u}^1 = \breve{u}^0$. Set $NA = 0$, $K(1) = 0$, $k = 1$.

Step 1 (*Trial point finding*). Find u^{k+1}, $\hat{a}^k$, μ^k, $\hat{g}^k$ as described by Lemma 1.2. Compute v^k of (2.17), $\hat{\varepsilon}^k$ and $\hat{\delta}^k$ of (2.13).

Substep 1' (*Stopping criterion*). If $\hat{g}^k = 0$ and $\hat{\delta}^k \leqslant 0$, stop.

Substep 1" (*Noise attenuation*). If (2.18) does not hold, set $t^k = 10t^k$, $NA = 1$ and loop back to Step 1.

Step 2 (*Oracle call*). Call the oracle at u^{k+1} to obtain h^{k+1} and a^{k+1}. Compute $\check{u}^k$ of (2.1).

Step 3 (*Step distribution*). Perform the following operations, depending on the descent test (2.19).

Descent-step: If (2.19) holds,	*Null-step*: If (2.19) does not hold,
set $\hat{u}^{k+1} = \check{u}^k$;	set $\hat{u}^{k+1} = \hat{u}^k$;
set $K(k+1) = k$, $NA = 0$.	set $K(k+1) = K(k)$.
Select $t^{k+1} \geqslant t_{\min}$.	If $NA = 1$, set $t^{k+1} = t^k$.
	If $NA = 0$, select $t^{k+1} \in [t_{\min}, t^k]$.

Step 4 (*Model updating*). Choose a function $\check{h}^{k+1}(\cdot)$ satisfying (2.8).

Step 5 (*Loop*). Increase k by 1 and go to Step 1. □

A few comments on the method are in order.

(1) The initial u^1 may be the Slater point itself; in this case, $\hat{u}^1 = \check{u}^0 = u^1 = u^0$.

(2) The simplest initial model is $\check{h}^1(\cdot) = \bar{h}^1(\cdot)$. However, the algorithm may be hot-started, with a nonempty initial bundle: J^1 in (1.9) will contain more than one index. Being higher, the model $\check{h}^1(\cdot)$ will thus be more accurate.

(3) Similarly, *multiple cuts* may be used at each iteration: the oracle may answer several values for h and a at a given u, each of which providing its linearization $\bar{h}(\cdot)$ satisfying (1.6). The main change in the algorithm is notational; we will not elaborate on this technique here.

(4) Step 1 may use the QP method of [21] or [10], which can solve efficiently sequences of subproblems (1.10) when C and f are polyhedral. The same method can also handle a quadratic f.

(5) Section 5 below will establish that either $f(\hat{u}^k) \to -\infty$ or the convergence property (2.20) holds. An additional stopping criterion could accordingly be inserted in Substep 1': stop if $f(\hat{u}^k)$ is deemed small enough.
Along the same lines, tolerances may be inserted in Substep 1': one can stop when $|\hat{g}^k| \leqslant \underline{\rho}$ and $\hat{\delta}^k \leqslant \underline{\hat{\delta}}$, with $\underline{\rho} > 0$ and $\underline{\hat{\delta}} > 0$. Admitting that $f(\hat{u}^k)$ is bounded from below, (2.20) will guarantee that the algorithm stops anyway. Note that $\underline{\hat{\delta}}$ is homogeneous to f-values; as for $\underline{\rho}$, (3.7) shows that it is a constraint residual in (3.1).

(6) Substep 1" may of course use extrapolation formulae more sophisticated than just multiplying t^k by 10. The only important thing is to drive t^k to $+\infty$ in case of an infinite loop within Step 1.
Step 3 may likewise use sophisticated updating formulae. Note that the stepsize may not increase after a null step.

(7) As mentioned in Sect. 2.3, the property $\check{h}^{k+1}(\cdot) \geqslant \bar{h}^{-k}(\cdot)$ in Step 4 is (always recommended but) only necessary when a null step is made and $\mu^k > 0$.

It should be clear that (2.20) is *the* desirable convergence property. With relation to our comment (5) above, let the tolerances $\hat{\underline{\delta}}$ and $\underline{\rho}$ stop the algorithm at some iteration $\underline{k}$, the last descent step having been performed at iteration $\underline{K} := K(\underline{k})$. Then (see Lemma 2.1) $\hat{u}^{\underline{k}}$ is feasible within $\check{\eta}^{\underline{K}} \leqslant \max\{\eta^0, \eta^{\underline{K}+1}\}$. As for objective values, assume that (1.1) has an optimal solution at finite distance, say $|u^*| \leqslant R$. Using the Cauchy–Schwarz inequality in (2.16), $f(\hat{u}^{\underline{k}}) \leqslant f(u^*) + \hat{\underline{\delta}} + R\underline{\rho}$.

5 Convergence

Convergence of a bundle method is usually split into two cases: either there are infinitely many descent steps, or the stability center stops. Here, the latter case splits in turn into several subcases, due to possible loops within Step 1.

We first establish relations coming from the noise in the oracle.

Lemma 5.1 *Let k be such that (2.18) does not hold. Then, u^0 being the Slater point (1.2),*

$$-\mu^k h(u^0) \leqslant f(u^0) - f(\hat{u}^k) + \frac{1}{2t^k}|u^0 - \hat{u}^k|^2, \tag{5.1}$$

$$-\mu^k h(\hat{u}^k) \leqslant \hat{\varepsilon}^k < -\frac{t^k}{2}|\hat{g}^k|^2 \quad \text{and} \quad |\hat{g}^k|^2 \leqslant 2h(\hat{u}^k)\frac{\mu^k}{t^k}. \tag{5.2}$$

Proof Set $u = u^0$ in (2.12) and use "not (2.18)":

$$\begin{aligned}\mu^k h(u^0) \geqslant \mu^k \check{h}^k(u^0) &\geqslant f(u^{k+1}) + (u^0 - u^{k+1})\hat{g}^k - f(u^0)\\ &= f(\hat{u}^k) - \hat{\varepsilon}^k + (u^0 - \hat{u}^k)\hat{g}^k - f(u^0)\\ &\geqslant f(\hat{u}^k) + \tfrac{t^k}{2}|\hat{g}^k|^2 + (u^0 - \hat{u}^k)\hat{g}^k - f(u^0).\end{aligned}$$

This gives[4]

$$\mu^k h(u^0) \geqslant f(\hat{u}^k) - \frac{1}{2t^k}|u^0 - \hat{u}^k|^2 - f(u^0),$$

which is (5.1). The first inequality in (5.2) comes easily from (2.14) and "not (2.18)"; the second inequality is a consequence. □

5.1 Infinite loop within Step 1

This section considers the case where k stops: the algorithm solves (1.8) repeatedly, without visiting Steps 2–5. Then the model $\check{h} = \check{h}^k$ and the stability center $\hat{u} = \hat{u}^k$ are fixed, only $t = t^k$ varies (increasingly). We therefore drop the misleading superscript k and use more appropriate notation μ_t, $\hat{g}_t$, u_t^+, etc. It is convenient to introduce the

[4] Just develop the square $|\sqrt{t}\hat{g} + (u^0 - \hat{u})/\sqrt{t}|^2 \geqslant 0$.

pure cutting-plane problem

$$\min f(u), \quad u \in C, \quad \check{h}(u) \leqslant 0. \tag{5.3}$$

Indeed, (1.8) amounts to computing the Moreau–Yosida regularization at $\hat{u}$ ([14, Sect. XV.4.1]) of the actual objective function in (5.3) (equal to $f(u)$ if u feasible, $+\infty$ otherwise).

Proposition 5.2 *Suppose the loop within Step 1 is infinite at some iteration k—so that $t \to +\infty$; call $\check{f}^*$ the optimal value of the pure cutting-plane problem (5.3). Then*

(1) $\limsup \hat{\delta}_t \leqslant 0$ *and* $\hat{g}_t \to 0$;
(2) $\check{f}^* > -\infty$, $\check{h}(\hat{u}) > 0$ *and* $f(\hat{u}) \leqslant \check{f}^*$;
(3) $f(u_t^+) \to \check{f}^*$; *and if (5.3) has a nonempty set of optimal solutions, then u_t^+ tends to the projection of $\hat{u}$ onto that set.*

Proof With the present notation, (5.1) reads

$$\mu_t \leqslant \frac{f(u^0) - f(\hat{u})}{-h(u^0)} + \frac{|u^0 - \hat{u}|^2}{-2th(u^0)},$$

which is bounded from above since t increases.

Then write (5.2): $-\mu_t h(\hat{u}) \leqslant \hat{\varepsilon}_t < -\frac{t}{2}|\hat{g}_t|^2$ and $|\hat{g}_t|^2 \leqslant 2h(\hat{u})\frac{\mu_t}{t}$; this implies $\hat{\varepsilon}_t < 0$ and $\hat{g}_t \to 0$. It follows that $\hat{\delta}_t \leqslant \hat{\varepsilon}_t + |\hat{u}|\,|\hat{g}_t|$ [see (2.13)] cannot have a positive cluster point; (1) is proved.

If $\hat{u}$ were feasible in (1.8), we could set $u = \hat{u}$ in (2.15), entailing the contradiction $\hat{\varepsilon}_t \geqslant 0$. Next, (2.16) shows with (1) that $f(\hat{u}) \leqslant f(u)$ for all u feasible in (1.8), hence in (5.3); in particular, (5.3) has a finite optimal value: this proves (2).

Finally, fix u feasible in (5.3), hence in (1.8):

$$f(u_t^+) + \frac{1}{2t}|u_t^+ - \hat{u}|^2 \leqslant f(u) + \frac{1}{2t}|u - \hat{u}|^2 \tag{5.4}$$

and pass to the limit: $\{u_t^+\}$ is a minimizing sequence for (5.3). Besides, set u in (5.4) to an optimal solution of (5.3): u is feasible in (1.8), $f(u_t^+) \geqslant f(u)$ and we can write

$$\frac{1}{2t}|u_t^+ - \hat{u}|^2 \leqslant \frac{1}{2t}|u - \hat{u}|^2;$$

this completes the proof of (3). □

It is known that the sequences $\{f(u_t^+)\}$ and $\{|u_t^+ - \hat{u}|\}$ are actually monotone; see for example [23]. Because $\hat{f}^*$ is not larger than the optimal value of (1.1), (2) shows that $\hat{u}$ has a very good f-value—but a blatantly bad h-value, although the latter is assessed by Lemma 2.1. During a loop within Step 1, trial points u_t^+ rely upon the deceiving point $\hat{u}$; they may be driven toward uninteresting regions of C, without even consulting the oracle to check. If f and C (and $\check{h}$) are polyhedral, u_t^+ solves (5.3) for t large enough: see [14, Proposition XV.4.2.5]. This case may be discovered by a

parametric QP method such as [23], which is thus useful to shorten such potentially fruitless loops.

5.2 Finitely many descent steps

This section is devoted to the case where the stability center stops, say at iteration $\bar{k}$; accordingly, we denote by $\hat{u}$ the stability center(s) $\hat{u}^k = \hat{u}^{\bar{k}}$ for $k \geqslant \bar{k}$.

First of all, a rather simple situation, similar to that of the previous section, is when there are infinitely many loops within Step 1. Then $NA = 1$ forever, and t^k is never decreased in Step 3; in fact, $t^k \to +\infty$.

Proposition 5.3 *Suppose there are only null steps after iteration $\bar{k}$. Denote by $\mathcal{K}$ the set of $k \geqslant \bar{k}$ for which at least one loop within Step 1 occurs. If $\mathcal{K}$ is an infinite set, then* (2.20) *holds*: *indeed* $\lim_{k\in\mathcal{K}} \hat{g}^k = 0$ *and* $\limsup_{k\in\mathcal{K}} \hat{\delta}^k \leqslant 0$.

Proof This is essentially Proposition 5.2(1): write (5.1), (5.2) for $k \in \mathcal{K}$ and let t^k tend to $+\infty$ for $k \in \mathcal{K}$. □

The other situation is when (2.18) holds for all k large enough. From then on, t^k may only decrease (or stay fixed forever). Besides, (2.17) implies $\hat{\varepsilon}^k \leqslant v^k$; so we clearly have

$$|\hat{\varepsilon}^k| \leqslant v^k \quad \text{and} \quad t^k|\hat{g}^k|^2 \leqslant 2v^k \quad \text{if (2.18) holds}. \tag{5.5}$$

Introduce the Lagrangian associated with (1.8)

$$C \ni u \mapsto L^k(u) := f(u) + \mu^k \check{h}^k(u) + \frac{1}{2t^k}|u - \hat{u}|^2. \tag{5.6}$$

We start with properties linking successive null steps, which are crucial for establishing convergence of bundle-type methods. They explain the importance of the aggregate linearization (2.6).

Lemma 5.4 *After a null step issued from $\hat{u}$, the Lagrangian* (5.6) *satisfies*

(1) $L^k(u^{k+1}) + \frac{1}{2t^k}|u^{k+1} - \hat{u}|^2 \leqslant L^k(\hat{u})$.

(2) $L^k(u^{k+1}) + \frac{1}{2t^k}|u^{k+2} - u^{k+1}|^2 \leqslant L^{k+1}(u^{k+2})$; *this relies upon the properties $\hat{u}^{k+1} = \hat{u}$ and $t^{k+1} \leqslant t^k$.*

Proof Linearize the non-quadratic part of L^k to obtain the quadratic function

$$C \ni u \mapsto \overline{L}^k(u) := f(u^{k+1}) + (u - u^{k+1})\hat{g}^k + \frac{1}{2t^k}|u - \hat{u}|^2. \tag{5.7}$$

In view of (2.12), we do have $\overline{L}^k \leqslant L^k$ over C.

Developing the square $|u - u^{k+1} + u^{k+1} - \hat{u}|^2$, direct calculations using (1.11) give

$$\overline{L}^k(u) = \overline{L}^k(u^{k+1}) + \frac{1}{2t^k}|u - u^{k+1}|^2 = L^k(u^{k+1}) + \frac{1}{2t^k}|u - u^{k+1}|^2, \tag{5.8}$$

so that

$$L^k(u^{k+1}) + \frac{1}{2t^k}|u - u^{k+1}|^2 = \bar{L}^k(u) \leqslant L^k(u) \quad \text{for all } u \in C$$

and (1) follows by setting $u = \hat{u}$.

Now we claim that

$$f(u) \geqslant f(u^{k+1}) + (u - u^{k+1})\hat{g}^k \quad \text{for all } u \text{ such that } \check{h}^{k+1}(u) \leqslant 0. \tag{5.9}$$

If $\mu^k = 0$, this is clear from (2.12), so assume $\mu^k > 0$: then $\check{h}^k(u^{k+1}) = 0$ (complementarity slackness). The definitions (2.8) and (2.6) of $\check{h}^{k+1}(\cdot)$ and $\bar{h}^-(\cdot)$ then give

$$0 \geqslant \check{h}^{k+1}(u) \geqslant \bar{h}^-(u) = (u - u^{k+1})\hat{a}^k.$$

Multiply by $\mu^k > 0$, sum up with (2.9), (2.11) and use (1.11); our claim is proved.

Plug (5.9) into (5.7): for all $u \in C$ such that $\check{h}^{k+1}(u) \leqslant 0$,

$$\overline{L}^k(u) \leqslant f(u) + \frac{1}{2t^k}|u - \hat{u}|^2 \leqslant f(u) + \frac{1}{2t^{k+1}}|u - \hat{u}|^2,$$

because $t^{k+1} \leqslant t^k$. Take in particular $u = u^{k+2}$: because $\hat{u}^{k+1} = \hat{u}$ and because of complementarity slackness, the righthand side is $L^{k+1}(u^{k+2})$, which is therefore bigger than $\overline{L}^k(u^{k+2})$; (2) then follows from the expression (5.8) of $\overline{L}^k(u^{k+2})$. □

We remark that the above result is really due to the strong convexity of $L^k(\cdot)$, which is minimized over C at u^{k+1}; see [14, Theorem VI.6.1.2]. We now turn to the situation where trouble due to a noisy oracle eventually ceases.

Proposition 5.5 *Suppose that, after some iteration, only null steps occur and t^k never increases (no loop within Step 1 occurs). Suppose also that the oracle inaccuracies η^k are bounded from above.*

Then $\hat{\delta}^k$, $\hat{g}^k$ tend to 0, as well as v^k; and u^k tends to $\hat{u}$.

Proof First we bound μ^k. Fix $b^0 \in \partial f(u^0)$ and plug the subgradient inequality into (2.12) with $u = u^0$:

$$f(u^0) + \mu^k h(u^0) \geqslant f(u^0) + (u^{k+1} - u^0)b^0 + (u^0 - u^{k+1})\hat{g}^k.$$

Single out $\hat{u}$ and use (1.11):

$$\begin{aligned} 0 \geqslant \mu^k h(u^0) &\geqslant (u^0 - \hat{u} - t^k b^0)\hat{g}^k + t^k|\hat{g}^k|^2 + (\hat{u} - u^0)b^0 \\ &\geqslant 2\hat{g}^k w^k + t_{\min}|\hat{g}^k|^2 - M, \end{aligned}$$

where we have set $w^k := (u^0 - \hat{u} - t^k b^0)/2$ and M is a constant. This implies (see footnote 4, page 193) $-\mu^k h(u^0) \leqslant |w^k|^2/t_{\min} + M$; because t^k does not increase, w^k is bounded and so is μ^k; say $\mu^k \leqslant \bar{\mu}$.

Then Lemma 5.4(1) and (1.7) give

$$L^k(u^{k+1}) \leqslant L^k(\hat{u}) \leqslant f(\hat{u}) + \bar{\mu}\max\{h(\hat{u}), 0\}.$$

Because Lemma 5.4(2) implies that the sequence $\{L^k(u^{k+1})\}$ is increasing, we see from Lemma 5.4(1) that $|u^{k+1} - \hat{u}|$ is bounded: u^k is bounded; and η^k is also bounded by assumption. Knowing that the η-subdifferential of (1.3) is locally bounded ([14, Prop.XI.4.1.2]),

$$\text{the sequence } \{a^k\} \text{ is bounded .} \tag{5.10}$$

Besides, $L^k(u^{k+1})$ has a finite limit, hence

$$\text{the sequence } \{|u^{k+2} - u^{k+1}|\} \text{ tends to } 0 \text{ .} \tag{5.11}$$

Now consider the linearization $\bar{h}^{k+1}(\cdot)$ of (1.5). Note that $h^{k+1} > 0$: otherwise $\check{u}^k$ of (2.1) would be equal to u^{k+1} and the descent test (2.19) would be passed, just by the definition (2.17) of v^k. Note also that $\check{h}^{k+1}(u^{k+2}) \leqslant 0$, hence $\bar{h}^{k+1}(u^{k+2}) \leqslant 0$ from (2.8). Then we have by the Cauchy-Schwarz inequality

$$\begin{aligned} 0 < h^{k+1} = \bar{h}^{k+1}(u^{k+1}) &= \bar{h}^{k+1}(u^{k+2}) + (u^{k+1} - u^{k+2})a^{k+1} \\ &\leqslant |u^{k+1} - u^{k+2}|\,|a^{k+1}|, \end{aligned}$$

so that $h^{k+1} \to 0$ from (5.10), (5.11). In (2.1), $\check{\beta}^k \to 1$ so $\check{\beta}^k \geqslant \kappa$ for k large enough. Start from "not (2.19)": $-\kappa v^k < f(\check{u}^k) - f(\hat{u}^k)$ and write

$$\begin{aligned} (1-\kappa)v^k &< f(\check{u}^k) - f(u^{k+1}) \\ &\leqslant (1-\check{\beta}^k)[f(u^0) - f(u^{k+1})] \\ &= (1-\check{\beta}^k)[f(u^0) - f(\hat{u}) + v^k]. \end{aligned}$$

[add $v = f(\hat{u}) - f(u^+)$]

[convexity of f between u^0 and u^+]

Take k so large that $\check{\beta}^k - \kappa > 0$ and obtain $v^k \leqslant \dfrac{f(u^0) - f(\hat{u})}{\check{\beta}^k - \kappa}(1 - \check{\beta}^k)$; hence $v^k \to 0$.

To finish the proof, observe from (5.5) that $\hat{g}^k \to 0$ because $t^k \geqslant t_{\min}$, and also $\hat{\varepsilon}^k \to 0$. So from (2.13), $\hat{\delta}^k = \hat{\varepsilon}^k + \hat{u}\hat{g}^k \to 0$. Finally observe from (2.18) that $|u^{k+1} - \hat{u}|$ tends to 0 just as v^k, since t^k does not increase. □

5.3 Case of infinitely many descent steps

The last situation is when the algorithm "looks like" an ordinary optimization algorithm, consisting of a series of descent iterations.

Proposition 5.6 *Suppose that the set $\mathcal{K} \subset \mathbb{N}$ of descent iterations is infinite. Either $f(\hat{u}^k) \to -\infty$ or the convergence property (2.20) holds in the sense that*

$$\lim_{k\in\mathcal{K}} \hat{g}^k = 0 \quad \textit{and} \quad \liminf_{k\in\mathcal{K}} \hat{\delta}^k \leqslant 0.$$

Proof Assuming that the monotonic sequence $\{f(\hat{u}^k)\}$ has a finite limit, let k_1 and k_2 be two successive indices in $\mathcal{K}$.

Because of (5.5), $v^k \geqslant 0$ for $k \in \mathcal{K}$; besides, $\hat{u}^{k_2} = \hat{u}^{k_1+1}$ and the descent test (2.19) gives $\kappa v^{k_2} \leqslant f(\hat{u}^{k_2+1}) - f(\hat{u}^{k_1+1})$. Summing up: $\sum_{k\in\mathcal{K}} v^k < +\infty$; the subsequence $\{v^k\}_{\mathcal{K}}$ therefore tends to 0; and remembering $t^k \geqslant t_{\min}$:

$$\lim_{k\in\mathcal{K}} t^k |g^k|^2 = 0, \quad \lim_{k\in\mathcal{K}} \hat{g}^k = 0, \quad \lim_{k\in\mathcal{K}} \hat{\varepsilon}^k = 0. \tag{5.12}$$

At the descent iteration $k = k_2$, $\hat{u}^{k+1} = \check{u}^k$ of (2.1), with $|\check{u}^k - u^0| \leqslant |u^{k+1} - u^0|$; using (1.11), we then write

$$\begin{aligned} |\hat{u}^{k+1} - u^0|^2 - |\hat{u}^k - u^0|^2 &\leqslant |\hat{u}^k - t^k\hat{g}^k - u^0|^2 - |\hat{u}^k - u^0|^2 \\ &= t^k(t^k\hat{g}^k + 2(u^0 - \hat{u}^k))\hat{g}^k. \end{aligned}$$

Using again the fact that $\hat{u}^k = \hat{u}^{k_2} = \hat{u}^{k_1+1}$, we sum these inequalities over $\mathcal{K}$ to obtain

$$-\infty < \sum_{k\in\mathcal{K}} t^k(t^k\hat{g}^k + 2(u^0 - \hat{u}^k))\hat{g}^k.$$

If there existed $\varepsilon > 0$ such that $(t^k\hat{g}^k + 2(u^0 - \hat{u}^k))\hat{g}^k \leqslant -\varepsilon$ for all $k \in \mathcal{K}$, then we would have $\sum_{\mathcal{K}} t^k < +\infty$, which is impossible. Therefore, using (5.12),

$$0 \leqslant \limsup_{k\in\mathcal{K}} [t^k|\hat{g}^k|^2 + 2u^0\hat{g}^k - 2\hat{u}^k\hat{g}^k] = \limsup_{k\in\mathcal{K}} [-2\hat{u}^k\hat{g}^k].$$

Then plug (5.12) into (2.13): $\liminf_{k\in\mathcal{K}} \hat{\delta}^k \leqslant 0$ and the proof is complete. □

5.4 Synthesis

The above study of the various possible cases clarifies the convergence properties of the algorithm. The present section summarizes these properties; it also studies boundedness of the multiplier μ, which is important in the primal-dual framework of Sect. 3.

Recall from the rules of the algorithm that $K(k)$ indexes the last descent iteration prior to k; and an important number for feasibility is the *asymptotic oracle inaccuracy*

$$\check{\eta}^\infty := \limsup_{k\to\infty} \check{\eta}^{K(k)}. \tag{5.13}$$

First we fix the case of bounded objective values. This does not depend much on the actual construction of $\{\hat{u}^k\}$ but rather on the properties of (1.1) itself.

Proposition 5.7 *Let the optimal objective value f^* of (1.1) be finite. Then*:

(1) *there exists $\mu^* \geqslant 0$ such that $\inf_{u\in C} f(u) + \mu^* h(u) = f^*$;*
(2) *with the notation (5.13), $\liminf f(\hat{u}^k) \geqslant f^* - \mu^* \check{\eta}^\infty$, so that $\{f(\hat{u}^k)\}$ is bounded from below if $\check{\eta}^\infty < +\infty$.*

Proof Statement (1) is just [34, Cor. 28.2.1]. To obtain (2), use Lemma 2.1 and write $f^* \leqslant f(\hat{u}^k) + \mu^* h(\hat{u}^k) \leqslant f(\hat{u}^k) + \mu^* \check{\eta}^{K(k)}$ for all k; then pass to the limit. □

Then convergence of the algorithm is as follows:

Theorem 5.8 *Suppose that Algorithm 4.1 neither terminates nor loops infinitely in Step 1 (so that $k \to \infty$), and the oracle inaccuracies η^k are bounded. Call f^* the optimal value of (1.1) and f^∞ the limit of $f(\hat{u}^k)$. Then*:

(1) *The convergence property (2.20) holds if $f^* > -\infty$.*
(2) *In this case, let $\mathcal{K}$ be an index set such that $\lim_{k\in\mathcal{K}} \max\{\hat{\delta}^k, |\hat{g}^k|\} = 0$. Then the corresponding subsequence $\{\mu^k\}$ is bounded*:

$$\limsup_{k\in\mathcal{K}} \mu^k \leqslant \frac{f(u^0) - f^\infty}{-h(u^0)}.$$

(3) *In any case, $f^\infty \leqslant f^*$ and $\limsup h(\hat{u}^k) \leqslant \check{\eta}^\infty$ of (5.13).*

Proof Propositions 5.3, 5.5 and 5.6 guarantee (1).

Now the first relation below is obtained by setting $u = u^0$ in (2.12); the second is direct from (2.13):

$$\begin{aligned} f(u^0) - f(u^+) - u^0\hat{g} + u^+\hat{g} &\geqslant -\mu h(u^0) \\ f(u^+) - u^+\hat{g} &= f(\hat{u}) - \hat{\delta}. \end{aligned}$$

Sum up, divide by $-h(u^0) > 0$ and pass to the limit to prove (2).

The second statement in (3) follows directly from Lemma 2.1. For the first, assume $f^\infty > -\infty$ (otherwise the proof is finished); write (2.16) with an arbitrary u feasible in (1.1) and pass to the limit to obtain $f^* \geqslant f^\infty$. □

6 Convergence in the primal

When the algorithm is used in the framework of Sect. 3, attention must be paid to the primal candidate $\hat{\lambda}$ introduced in Sect. 3.1. The model $\check{h}$ has the piecewise linear form (1.9) and several strategies are possible for the management of the bundle. To be specific, we will assume that Step 4 of Algorithm 4.1

- arbitrarily destroys indices from J^k,
- appends if necessary the aggregate column defined by (2.6),

- then appends the new column coming from u^{k+1},

so that the resulting J^{k+1} satisfies (2.7).

Lemma 6.1 *With this strategy, the columns* $\begin{pmatrix}\gamma^j\\ a^j\end{pmatrix}$, $j \in J^k$ *are convex combinations of the columns* $\begin{pmatrix}c_i\\ A_i\end{pmatrix}$ *in* (3.1).

Proof Two sorts of columns make up the bundle at a given iteration:

- The "natural" columns, which have $\gamma^j = c_{ij}$, $a^j = a_{ij}$ with $j > 0$, computed at the jth call of the oracle by a (possibly inaccurate) resolution of (3.3) for $u = u^j$.
- The aggregate columns, with negative indices; consider the aggregate column $-k<0$, constructed at the kth iteration:
  - from (2.4), $a^{-k} = \hat{a}^k$ of Proposition 1.2 is a convex combination of the a^j's in J^k,
  - from Lemma 2.4, $\gamma^{-k} = u^{k+1}\hat{a}^k - \check{h}^k(u^{k+1})$ is the same convex combination of the γ^j's.

Thus, the very first aggregation during the algorithm introduces a convex combination of natural columns. The subsequent aggregations introduce further convex combinations. Now taking convex combinations is a transitive operation, so these are again convex combinations of the original columns. □

To construct $\hat{\lambda}^k$ of (3.9) from λ of (1.12) is then a matter of computer programming, using appropriately the history of the successive aggregations. In (3.8), we have $\alpha_i^j \geqslant 0$ and $\sum_{i=1}^n \alpha_i^j = 1$ for each j; hence

$$\hat{\lambda}^k \geqslant 0 \quad \text{and} \quad \sum_{i=1}^n \hat{\lambda}_i^k = \sum_{j\in J^k} \lambda^j \sum_{i=1}^n \alpha_i^j = \sum_{j\in J^k} \lambda^j = \mu^k, \tag{6.1}$$

while Theorem 3.3 gives

$$c\hat{\lambda}^k + \hat{u}^k b = \hat{\delta}^k, \tag{6.2}$$

$$A\hat{\lambda}^k + b \geqslant \hat{g}^k. \tag{6.3}$$

With these premises, the convergence properties of $\hat{\lambda}^k$ follow naturally from Theorem 5.8:

Theorem 6.2 *Let the primal problem* (3.1) *have a feasible point, so that* (3.1) *and* (3.2) *have a finite common optimal value* z^* *and* (3.2) *has a multiplier* μ^*, *as stated in Proposition* 5.7.

Suppose that the algorithm neither terminates nor loops infinitely in Step 1 (*so that* $k \to \infty$), *and the oracle inaccuracies* η^k *are bounded. Define the* asymptotic primal error $\varepsilon^* := \mu^*\check{\eta}^\infty$ *from* (5.13) *and let* $\mathcal{K} \subset \mathbb{N}$ *be an index set as described in Theorem* 5.8(2). *Then*:

(1) *Each cluster point* $\hat\lambda^\infty$ *of the bounded sequence* $\{\hat\lambda^k\}_{k\in\mathcal{K}}$ *lies in the* ε^*-optimal primal solution set

$$\Lambda_{\varepsilon^*} := \{\lambda \in \mathbb{R}^n_+ : c\lambda \leqslant z^* + \varepsilon^*,\ A\lambda + b \geqslant 0\}. \tag{6.4}$$

(2) *The distance* $d(\hat\lambda^k) := \min_{\lambda\in\Lambda_{\varepsilon^*}} |\hat\lambda^k - \lambda|$ *from* $\hat\lambda^k$ *to the* ε^**-optimal set satisfies* $\lim_{k\in\mathcal{K}} d(\hat\lambda^k) = 0$.

Proof When both (3.1) and (3.2) are feasible, their respective optimal solution sets are nonempty and there is no duality gap. Comparing (3.2) with (1.1), their common optimal value z^* is $-f^*$ of Theorem 5.8.

Use Proposition 5.7(2) and Theorem 5.8(3): $-z^* \geqslant f^\infty \geqslant -z^* - \varepsilon^*$. Besides, Theorem 5.8(2) gives $\limsup_{k\in\mathcal{K}} \mu^k < \infty$: from (6.1), $\{\hat\lambda^k\}_{k\in\mathcal{K}}$ is bounded.

Let $\hat\lambda^\infty$ be a cluster point of $\{\hat\lambda^k\}_{k\in\mathcal{K}}$. Write (6.2) as $c\hat\lambda^k = \hat\delta^k - \hat u^k b$ and pass to the limit:

$$c\hat\lambda^\infty \leqslant \limsup_{k\in\mathcal{K}} \hat\delta^k - f^\infty \leqslant 0 - f^* + \varepsilon^* = z^* + \varepsilon^*.$$

Pass likewise to the limit in (6.3): $A\hat\lambda^\infty + b \geqslant 0$. This proves (1); then (2) follows from the continuity of the distance function $d(\cdot)$. □

7 Numerical illustrations

We conclude this paper with a brief account of our conic variant in practice, on the application that really motivated it: cutting-stock problems in the formulation of Gilmore–Gomory [12]; see also [30].

7.1 The cutting-stock problem

Recall that the problem is to minimize the number of stock pieces of width W, used to meet demands $d^1, \dots, d^m$, for m items to be cut at their widths $w^1, \dots, w^m$.

Hereafter, $w \in \mathbb{R}^m$ denotes the vector of widths. Let x_i denote the number of units of item i cut in a given roll. This makes up a vector $x \in \mathbb{R}^m$ which characterizes a *cut pattern*, and which is feasible if $wx \leqslant W$; let n be the (huge) number of feasible cut patterns. Then let λ_i be the number of rolls cut according to pattern i; relaxing the integrality constraint on λ, we obtain the formulation

$$\min \sum_{i=1}^n \lambda_i, \quad \sum_{i=1}^n \lambda_i x_i \geqslant d \in \mathbb{R}^m, \quad \lambda \geqslant 0.$$

This is (3.1), where $c \in \mathbb{R}^n$ is the vector of all ones and the feasible cut patterns make up the columns of A. The dual is to maximize ud over $\mathbb{R}^m_+$, subject to the constraint

$\sigma(u) := \max_{i=1}^{n} ux_i \leqslant 1$ and the oracle computing σ has to solve the knapsack problem

$$\max\ ux, \quad wx \leqslant W, \quad x \in \mathbb{N}^m. \tag{7.1}$$

7.2 Data sets

To save space, we give results only for the following randomly generated instances employed in [25]: the 4,000 instances of Wäscher and Gau [41] and the 3,360 instances of Degraeve and Peeters [7].

The instances of [41] are constructed by the CUTGEN1 generator of [11], using the following parameter values:

- Number of items $m = 10, 20, 30, 40, 50$.
- Width of the wide rolls $W = 10{,}000$.
- Interval fraction $c = 0.25, 0.5, 0.75, 1$; the widths w_i are uniformly distributed integers between 1 and cW.
- Average demand $\bar{d} = 10, 50$; with m uniform random numbers $R_1, \ldots, R_m \in (0, 1)$, the demands are $d_i := \lfloor \frac{R_i m\bar{d}}{R_1+\cdots+R_m} \rfloor$ for $i < m$, and $d_m := m\bar{d} - \sum_{i<m} d_i$ (in fact slightly more complicated formulas are used by [11]).

Duplicate widths are aggregated by summing their demands. Combining the different values for m, c and $\bar{d}$ results in 40 classes; in each class, 100 random instances are generated for a total of 4,000.

The *small-item-size* instances of [7] are generated similarly for $m = 10, 20, 30, 40, 50, 75, 100$, $c = 0.25, 0.5, 0.75, 1$ and $\bar{d} = 10, 50, 100$, except that $R_1, \ldots, R_m \in (0.1, 0.9)$ for the demand distribution. In the *medium-item-size* instances of [7], only $\bar{d} = 50$ is used and the widths are uniformly distributed on $[w_{\min}, cW]$, where $w_{\min} = 500,\ 1{,}000,\ 1{,}500$. Both cases have 84 data classes, and 20 random instances are generated in each class for a total of $2\times$ 1,680.

7.3 Implementation

Our testing environment uses a notebook PC (Pentium M 755 2 GHz, 1.5 GB RAM) under MS Windows XP, and Fortran 77.

In order to emphasize the primal-dual aspect of the algorithm, we report on the simultaneous generation of feasible primal solutions, along with the dual iterates u^k. These solutions are obtained by various heuristics, as described in [25].

We use the QP solver of [21] for (1.10). For the dual algorithm and primal heuristics, the knapsack problems (7.1) are solved by Martello–Toth's procedure MT1R, with an early termination test inserted (see [25, Sect. 2.2]): the branch and bound procedure is terminated when it obtains a feasible knapsack which is optimal within $\eta_r = 10^{-5}$ of relative accuracy.

A relative accuracy of $\underline{\varepsilon} = 10^{-9}$ is required from the conic algorithm. More precisely, Algorithm 4.1 is stopped when either v^k from (2.17) or $|\hat{g}^k| + \hat{\varepsilon}^k$ from (1.11), (2.13) is smaller than $\underline{\varepsilon}(1 + \hat{u}^k d)$, with (2.18) holding in the former case.

Table 1 Small-item-size instances of Degraeve and Peeters (240 instances per row)

m	k_{av}	k_{mx}	t_{av}	t_{mx}	n_e	n_g
10	14.92	32	0.00	0.01	108	0
20	32.66	61	0.01	0.04	110	0
30	53.05	97	0.06	10.63	115	1
40	71.61	140	0.04	0.32	124	0
50	93.20	171	0.09	0.68	139	0
75	145.80	259	0.26	1.89	140	1
100	192.05	338	0.46	4.07	147	0

Table 2 Medium-item-size instances of Degraeve and Peeters (240 instances per row)

m	k_{av}	k_{mx}	t_{av}	t_{mx}	n_e	n_g
10	17.33	27	0.00	0.01	54	0
20	34.92	58	0.01	0.08	63	0
30	53.43	86	0.02	0.14	83	0
40	70.73	123	0.04	0.61	68	0
50	90.10	164	0.07	0.89	69	1
75	139.22	236	0.36	8.28	80	1
100	191.29	300	1.46	59.67	78	0

Table 3 CSP instances of Wäscher and Gau (800 instances per row)

m	k_{av}	k_{mx}	t_{av}	t_{mx}	n_e	n_g
10	14.24	31	0.00	0.02	425	0
20	31.10	63	0.02	13.13	461	0
30	48.95	110	0.01	0.15	475	0
40	66.34	139	0.04	0.33	513	2
50	86.68	171	0.07	0.58	530	1

Besides, "early" termination occurs if the heuristic discovers a primal-optimal solution (this implies that the dual problem is solved as well, but the algorithm need not know it yet).

7.4 Results

Tables 1, 2, and 3 give the statistics for the three series of problems in Sect. 7.2; in these tables,

- k_{av} and k_{mx} are respectively the average and maximum numbers of iterations for the corresponding series of experiments;
- t_{av} and t_{mx} are likewise running times in wall-clock seconds;
- n_e is the number of "early" terminations due to the discovery of an optimal primal solution;

Table 4 Number of oracle calls compared with standard column generation on small- and medium-size instances

	small-size		medium-size	
m	k_{av}	n_{SP}	k_{av}	n_{SP}
10	15	16	17	18
20	33	42	35	50
30	53	63	53	96
40	72	100	71	128
50	93	138	90	174
75	146	232	139	332
100	192	318	191	551

- n_g is the number of instances with a nonzero final *gap* between the incumbent primal value and the dual bound rounded up to the next integer; we stress that this gap, which is 0 most of the time, *never* exceeds one unit.

These results demonstrate the validity of the method. Actually, they are quite similar to those reported in [25]. The latter concerns a highly elaborate bundle implementation to solve (3.2) by exact penalty, with a very smart choice of the penalty parameter via the FFD heuristic. By contrast, our implemented conic variant is quite simple. At present, its heuristics perform slightly worse on the instances of [7] (on 3 360 runs, 4 nonzero gaps instead of 3); their improvement is left for future work.

A real assessment of our method should involve comparisons with standard column generation. To be fair, however, such comparisons are not easy to perform: two (very) different and rather sophisticated pieces of software must be run by the same person, on the same computer, and under the same computing environment. Interested readers might want to consult [4,7,25,39–41]. Here we briefly mention some points not related to CPU time.

- It is commonly admitted that convergence of standard column generation is often hard to obtain. This motivates for example the hybrid mechanism of [7], with periodic switches to subgradient steps. By contrast, Algorithm 4.1 is "homogeneous", as it consistently uses the single cutting-plane paradigm.
- A comparison can be sketched concerning the number k_{av} of oracle calls, which is reported in [7] for small- and medium-size instances. Table 4 reproduces side-by-side column k_{av} of Tables 1 and 2 and column n_{SP} of Tables 2, 3, and 4 from [7] (lines *all*); numbers have been rounded.
  Beware that this comparison is not entirely meaningful since k_{av} refers to one single solution of (1.1)—or rather (3.2)—while n_{SP} includes several such solutions, at several nodes of the branch-and-bound tree.
- Although it is not directly related with solving (1.1), let us recall the striking fact that a proven optimal integer solution is found *at the root node*, in quasi all of our tests; by contrast, the branching strategy is an important ingredient in [7].
- At this point, observe that the number n_e of early terminations is fairly big. However, the influence of these early terminations is marginal: roughly speaking, k_{av} and k_{mx}

increase by some 10% if optimal primal solutions are just ignored by the dual optimization algorithm.

Acknowledgments We are indebted to an anonymous referee, whose lucid comments helped us to improve an earlier version of this paper.

References

1. Ben Amor, H., Valério de Carvalho, J.M.: Cutting stock problems. In: Desaulniers, G., Desrosiers, J., Salomon, M.M. (eds.) Column Generation, pp. 131–161. Springer, New York (2005)
2. Ben-Tal, A., Nemirovski, A.: Non-Euclidean restricted memory level method for large-scale convex optimization. Math. Program. **102**, 407–456 (2005)
3. Bixby, R.E., Gregory, J.W., Lustig, I.J., Marsten, R.E., Shanno, D.F.: Very large-scale linear programming: a case study in combining interior point and simplex methods. Oper. Res. **40**, 885–897 (1992)
4. Briant, O., Lemaréchal, C., Meurdesoif, Ph., Michel, S., Perrot, N., Vanderbeck, F.: Comparison of bundle and classical column generation. Math. Program. (2007, to appear). doi:10.1007/s10107-006-0079-z
5. Cheney, E.W., Goldstein, A.A.: Newton's method for convex programming and Tchebycheff approximation. Numer. Math. **1**, 253–268 (1959)
6. Correa, R., Lemaréchal, C.: Convergence of some algorithms for convex minimization. Math. Program. **62**, 261–275 (1993)
7. Degraeve, Z., Peeters, M.: Optimal integer solutions to industrial cutting stock problems. Part 2: Benchmark results. INFORMS J. Comput. **15**, 58–81 (2003)
8. Farley, A.A.: A note on bounding a class of linear programming problems, including cutting stock problems. Oper. Res. **38**, 922–923 (1990)
9. Fletcher, R., Leyffer, S.: A bundle filter method for nonsmooth nonlinear optimization. Numerical Analysis Report NA/195, Dept. of Math., Univ. of Dundee, Dundee (1999)
10. Frangioni, A.: Solving semidefinite quadratic problems within nonsmooth optimization algorithms. Comput. Oper. Res. **23**(11), 1099–1118 (1996)
11. Gau, T., Wäscher, G.: CUTGEN1: a problem generator for the standard one-dimensional cutting stock problem. Eur. J. Oper. Res. **84**, 572–579 (1995)
12. Gilmore, P.C., Gomory, R.E.: A linear programming approach to the cutting-stock problem. Oper. Res. **9**, 849–859 (1961)
13. Hintermüller, M.: A proximal bundle method based on approximate subgradients. Comput. Optim. Appl. **20**, 245–266 (2001)
14. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms. Springer, Berlin (1993)
15. Karas, E., Ribeiro, A., Sagastizábal, C., Solodov, M.: A bundle-filter method for nonsmooth convex constrained optimization. Math. Program. (2007, to appear). doi:10.1007/s10107-007-0123-7
16. Kelley, J.E.: The cutting-plane method for solving convex programs. J. Soc. Ind. Appl. Math. **8**, 703–712 (1960)
17. Kiwiel, K.C.: An algorithm for nonsmooth convex minimization with errors. Math. Comp. **45**, 173–180 (1985)
18. Kiwiel, K.C.: Methods of Descent for Nondifferentiable Optimization. Lecture Notes in Mathematics 1133. Springer, Berlin (1985)
19. Kiwiel, K.C.: A constraint linearization method for nondifferentiable convex minimization. Numer. Math. **51**, 395–414 (1987)
20. Kiwiel, K.C.: Exact penalty functions in proximal bundle methods for constrained convex nondifferentiable minimization. Math. Program. **52**, 285–302 (1991)
21. Kiwiel, K.C.: A Cholesky dual method for proximal piecewise linear programming. Numer. Math. **68**, 325–340 (1994)
22. Kiwiel, K.C.: Approximations in proximal bundle methods and decomposition of convex programs. J. Optim. Theory Appl. **84**, 529–548 (1995)
23. Kiwiel, K.C.: Finding normal solutions in piecewise linear programming. Appl. Math. Optim. **32**, 235–254 (1995)

24. Kiwiel, K.C.: Proximal level bundle methods for convex nondifferentiable optimization, saddle-point problems and variational inequalities. Math. Program. **69**, 89–109 (1995)
25. Kiwiel, K.C.: An inexact bundle approach to cutting stock problems. Tech. rep., Systems Research Institute, Warsaw (2005)
26. Kiwiel, K.C.: A proximal bundle method with approximate subgradient linearizations. SIAM J. Optim. **16**, 1007–1023 (2006)
27. Kiwiel, K.C.: A proximal-projection bundle method for Lagrangian relaxation, including semidefinite programming. SIAM J. Optim. **17**, 1015–1034 (2006)
28. Lemaréchal, C.: Lagrangian relaxation. In: Jünger, M., Naddef, D. (eds.) Computational Combinatorial Optimization, Lecture Notes in Computer Science 2241, pp. 112–156. Springer, Berlin (2001)
29. Lemaréchal, C., Nemirovskii, A.S., Nesterov Yu., E.: New variants of bundle methods. Math. Program. **69**, 111–147 (1995)
30. Lübbecke, M.E., Desrosiers, J.: Selected topics in column generation. Oper. Res. **53**, 1007–1023 (2005)
31. Mehrotra, A., Trick, M.: A column generation approach to graph coloring. INFORMS J. on Comput. **8**(4), 344–354 (1996)
32. Mifflin, R.: An algorithm for constrained optimization with semismooth functions. Math. Oper. Res. **2**, 191–207 (1977)
33. Miller, S.A.: An inexact bundle method for solving large structured linear matrix inequalities. Ph.D. thesis, Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA (2001)
34. Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1970)
35. Sagastizábal, C., Solodov, M.V.: An infeasible bundle method for nonsmooth convex constrained optimization without a penalty function or a filter. SIAM J. Optim. **16**, 146–169 (2005)
36. Savelsbergh, M.W.P.: A branch-and-price algorithm for the generalized assignment problem. Oper. Res. **45**, 831–841 (1997)
37. Solodov, M.V.: On approximations with finite precision in bundle methods for nonsmooth optimization. J. Optim. Theory Appl. **119**, 151–165 (2003)
38. Topkis, D.M.: A cutting-plane algorithm with linear and geometric rates of convergence. J. Optim. Theory Appl. **36**, 1–22 (1982)
39. Vanderbeck, F.: Computational study of a column generation algorithm for bin packing and cutting stock problems. Math. Program. **86**, 565–594 (1999)
40. Vanderbeck, F.: On Dantzig–Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. Oper. Res. **48**, 111–128 (2000)
41. Wäscher, G., Gau, T.: Heuristics for the integer one-dimensional cutting stock problem. OR Spectr. **18**, 131–144 (1996)
42. Wolsey, L.A.: Integer Programming. Wiley, New York (1998)