SIAM J. OPTIM.
Vol. 18, No. 1, pp. 242–259

A BUNDLE METHOD FOR A CLASS OF BILEVEL NONSMOOTH CONVEX MINIMIZATION PROBLEMS*

MIKHAIL V. SOLODOV†

Abstract. We consider the bilevel problem of minimizing a nonsmooth convex function over the set of minimizers of another nonsmooth convex function. Standard convex constrained optimization is a particular case in this framework, corresponding to taking the lower level function as a penalty of the feasible set. We develop an explicit bundle-type algorithm for solving the bilevel problem, where each iteration consists of making one descent step for a weighted sum of the upper and lower level functions, after which the weight can be updated immediately. Convergence is shown under very mild assumptions. We note that in the case of standard constrained optimization, the method does not require iterative solution of any penalization subproblems—not even approximately—and does not assume any regularity of constraints (e.g., the Slater condition). We also present some computational experiments for minimizing a nonsmooth convex function over a set defined by linear complementarity constraints.

Key words. bilevel optimization, convex optimization, nonsmooth optimization, bundle methods, penalty methods

AMS subject classifications. 90C30, 65K05, 49D27

DOI. 10.1137/050647566

1. Introduction.

We consider a class of *bilevel problems* of the form

$$\begin{array}{ll} \text{minimize} & f_1(x) \\ \text{subject to} & x \in S_2 = \arg\min\{f_2(x) \mid x \in \Re^n\}, \end{array} \tag{1.1}$$

where $f_1 : \Re^n \to \Re$ and $f_2 : \Re^n \to \Re$ are convex functions, in general nondifferentiable.

The above is a special case of the *mathematical program with generalized equation (or equilibrium) constraint* [20, 7], which is

$$\begin{array}{ll} \text{minimize} & f_1(x) \\ \text{subject to} & x \in \{x \in \Re^n \mid 0 \in T(x)\}, \end{array}$$

where T is a set-valued mapping from $\Re^n$ to the subsets of $\Re^n$. The bilevel problem (1.1) is obtained by setting $T(x) = \partial f_2(x)$, $x \in \Re^n$. In the formulation of the problem considered here, there is only one (decision) variable $x \in \Re^n$, and we are interested in identifying specific solutions of the inclusion $0 \in T(x)$ (equivalently, of the lower level minimization problem in (1.1)); see [7]. Problems of the form of (1.1) are also sometimes referred to as *hierarchical optimization*; see, e.g., [12, 4].

Note that, as a special case, (1.1) contains the standard convex constrained optimization problem

$$\begin{array}{ll} \text{minimize} & f_1(x) \\ \text{subject to} & g_i(x) \le 0, \quad i = 1, \ldots, m, \end{array} \tag{1.2}$$

*Received by the editors December 14, 2005; accepted for publication (in revised form) October 24, 2006; published electronically April 3, 2007. This research is supported in part by CNPq grants 301508/2005-4, 490200/2005-2, and 550317/2005-8, by PRONEX-Optimization, and by FAPERJ grant E-26/151.942/2004.

http://www.siam.org/journals/siopt/18-1/64756.html

†Instituto de Matemática Pura e Aplicada, Estrada Dona Castorina 110, Jardim Botânico, Rio de Janeiro, RJ 22460-320, Brazil (solodov@impa.br).

where $g : \Re^n \to \Re^m$ is a (nonsmooth) convex function. Indeed, (1.2) is obtained from (1.1) by taking $f_2(x) = p(x)$, where $p : \Re^n \to \Re_+$ is some penalty of the constraints, e.g.,

$$f_2(x) = p(x) = \sum_{i=1}^{m} \max\{0,\, g_i(x)\}. \tag{1.3}$$

In this paper, we show that the bilevel problem (1.1) can be solved by a properly designed (proximal) bundle method [15, 11, 3], iteratively applied to the parametrized family of functions

$$F_\sigma(x) = \sigma f_1(x) + f_2(x), \quad \sigma > 0, \tag{1.4}$$

where σ varies along the iterations. Specifically, if $x^k \in \Re^n$ is the current iterate and $\sigma_k > 0$ is the current parameter, it is enough to make *just one* descent step for F_{σ_k} from the point x^k, after which the parameter σ_k can be immediately updated. We emphasize that at no iteration is the function F_{σ_k} minimized to any prescribed precision. Once the descent condition is achieved, the parameter can be updated immediately and we can start working with the new function $F_{\sigma_{k+1}}$. For convergence of the resulting algorithm to the solution set of (1.1), parameters $\{\sigma_k\}$ should be chosen is such a way that

$$\lim_{k\to\infty} \sigma_k = 0, \quad \sum_{k=0}^{\infty} \sigma_k = +\infty. \tag{1.5}$$

The requirement that σ_k must tend to zero is natural and indispensable, as can be seen from the case of standard optimization (1.2). To this end, it is interesting to comment on the relation between our method and the classical penalty approximation scheme [8, 23]. The penalty scheme consists of solving a sequence of unconstrained subproblems

$$\text{minimize } F_\sigma(x), \quad x \in \Re^n, \tag{1.6}$$

where F_σ is given by (1.4) with f_2 being a penalty term p, such as (1.3). (In the literature, it is more common to minimize $\sigma^{-1}F_\sigma(x) = f_1(x) + \sigma^{-1}p(x)$, but the resulting subproblem is clearly equivalent to (1.6).) As is well known, under mild assumptions optimal paths of solutions $x(\sigma)$ of penalized problems (1.6) tend to the solution set of (1.2) as $\sigma \to 0$. We emphasize that the requirement that penalty parameters should tend to zero is, in general, indispensable. To guarantee that a solution of (1.6) is a solution of the original problem (1.2) for some *fixed* $\sigma > 0$ (i.e., exactness of the penalty function), some regularity assumptions on constraints are needed (e.g., see [3, section 14.4]). No assumptions of this type are made in this paper. The fundamental issue is approximating $x(\sigma_k)$ for some sequence of parameters $\sigma_k \to 0$. It is clear that approximating $x(\sigma_k)$ with precision is computationally impractical. It is therefore attractive to trace the optimal path in a loose (and computationally cheap) manner, while still safeguarding convergence. In a sense, this is what our method does: instead of solving subproblems (1.6) to some prescribed accuracy, it makes just one descent step for F_{σ_k} from the current iterate x^k and immediately updates the parameter. We emphasize that this results in meaningful progress (and ultimately produces iterates converging to solutions of the problem) for arbitrary points x^k, and not just for points close to the optimal path, i.e., points close to $x(\sigma_k)$.

We therefore obtain an implementable algorithm for tracing optimal paths of penalty schemes.

We next discuss the relationship of our algorithm to the existing literature. For the bilevel setting of (1.1), we believe that our proposal is the first method which is completely *explicit*. In some ways, it is related to [4], where a proximal point method for (1.1) has been considered, and (1.5) is referred to as *slow control*. However, as any proximal method, the method of [4] is *implicit*: it requires solving nontrivial subproblems of minimizing regularizations of functions F_{σ_k} at every iteration, even if approximately. By contrast, the method proposed in this paper is completely explicit: each iteration is a serious (or descent) step for the current F_{σ_k}, constructed by a finite number of null steps in a way which is essentially standard in nonsmooth optimization.

The special case of standard optimization deserves some further comments. We next discuss bundle methods applicable to problems with nonlinear constraints, such as (1.2) above. When the problem admits exact penalization, one can solve the equivalent unconstrained problem of minimizing the exact penalty function; see [14, 18]. However, as already mentioned above, exact penalization requires regularity assumptions on constraints, such as the Slater condition (existence of some $x \in \Re^n$ such that $g_i(x) < 0$ for all $i = 1, \ldots, m$). We stress that no assumptions of this type are needed for our method. For example, our method is applicable to minimizing a nonsmooth function subject to (monotone linear) complementarity constraints

$$(Qx + q)_i \geq 0, \quad x_i \geq 0, \quad x_i(Qx + q)_i = 0, \quad i = 1, \ldots, n,$$

where Q is an $n \times n$ positive semidefinite matrix. Those constraints can be modeled in the form (1.2) as

$$-Qx - q \leq 0, \quad -x \leq 0, \quad \langle Qx + q, x \rangle \leq 0.$$

Complementarity constraints do not satisfy constraint qualifications, no matter how they are modeled, which makes this class of problems particularly difficult. We shall come back to problems with complementarity constraints in section 4, where some computational experiments are presented.

The methods in [21, 22] and [15, Chap. 5] do not use penalization but enforce feasibility of every serious iteration. In particular, they require a feasible starting point, which is a difficult computational task (in the case of nonlinear constraints). In addition, regularity of constraints is still needed for convergence. Bundle methods which do not use penalty functions and do not enforce feasibility are [19, 9, 24, 13]. The methods in [24, 13] share one feature in common with the one proposed here: they apply bundle techniques to a dynamically changing objective function, except that the function is different (underlying [24, 13] is the so-called *improvement function*, which goes back to [21, 15, 1]). The methods of [24, 13] require the Slater condition, while those in [19, 9] do not. However, [19, 9] (as well as [21, 17, 18]) need a priori boundedness assumptions on the iterates to prove convergence. For our method, we assume only that the solution set of the problem is bounded.

For the standard optimization setting (1.2), this paper is also somewhat related to [5], where interior penalty schemes are coupled with continuous-time steepest descent to produce a family of paths converging to a solution set. However, concrete numerical schemes in [5] arise from *implicit* discretization and, thus, result in implicit proximal-point iterations, just as in [4]. Nevertheless, it was conjectured in [5] that an economic algorithm performing a single iteration of some descent method for each value of σ_k could be enough to generate a sequence of iterates converging to a solution of the

problem. This is what the presented method does, although we use exterior rather than interior penalties and consider the more general nonsmooth setting (as well as the more general bilevel setting). A related explicit descent scheme for the smooth case has been developed in [25].

Our notation is quite standard. By $\langle x, y\rangle$ we denote the inner product of x and y, and by $\|\cdot\|$ the associated norm, where the space is always clear from the context. For a convex function $f : \Re^n \to \Re$, its ε-subdifferential at the point $x \in \Re^n$ is denoted by $\partial_\varepsilon f(x) = \{g \in \Re^n \mid f(y) \geq f(x) + \langle g, y - x\rangle - \varepsilon \text{ for all } y \in \Re^n\}$, where $\varepsilon \in \Re_+$. Then the subdifferential of f at x is given by $\partial f(x) = \partial_0 f(x)$. If S is a closed convex set in $\Re^n$, then $P_S(x)$ stands for the orthogonal projection of the point $x \in \Re^n$ onto S, and $\operatorname{dist}(x, S) = \|x - P_S(x)\|$ is the distance from x to S.

2. The algorithm. As already outlined above, the conceptual idea of the algorithm is quite simple. If $x^k \in \Re^n$ is the current approximation to a solution of (1.1) and $\sigma_k > 0$ is the current parameter defining the function F_{σ_k} in (1.4), an iteration of the method consists of making a descent step for F_{σ_k} relative to its value at x^k. After this, the value of σ_k can be changed immediately. Since the function F_{σ_k} is nonsmooth, the computationally implementable way to construct a descent step is the bundle technique [15, 11, 3]. We next introduce the notation necessary for stating our algorithm.

Bundle methods keep memory of the past in a *bundle* of information. Let x^k be the current approximation to a solution and let y^i, $i = 1, \ldots, \ell - 1$, be all the points that have been produced by the method so far, including the ones which have not been accepted as satisfactory (so-called "null steps"). Generally, $\{x^k\}$ is a particular subsequence of $\{y^i\}$. For an iteration index ℓ, we shall denote by $k(\ell)$ the index of the last iteration preceding the iteration ℓ at which x^k and σ_k have been modified. Whenever k and ℓ appear in the same expression, we mean that $k = k(\ell)$.

Let us denote the function and subgradient values of f_1 at the points y^i, $i = 1, \ldots, \ell - 1$, by $f_1^i = f_1(y^i)$, $g_1^i \in \partial f_1(y^i)$, and similarly for f_2. Since $(\sigma_k g_1^i + g_2^i) \in \partial F_{\sigma_k}(y^i)$, this information can be used to define a cutting-planes approximation Ψ_ℓ of the function F_{σ_k}, as follows:

$$\begin{aligned}\Psi_\ell(y) &:= \max_{i<\ell} \left\{\sigma_k f_1^i + f_2^i + \langle \sigma_k g_1^i + g_2^i, y - y^i\rangle\right\} \\ &= \sigma_k f_1(x^k) + f_2(x^k) \\ &\quad + \max_{i<\ell} \left\{-(\sigma_k e_1^{k,i} + e_2^{k,i}) + \langle \sigma_k g_1^i + g_2^i, y - x^k\rangle\right\}, \end{aligned} \tag{2.1}$$

where the second expression is centered at x^k and uses the linearization errors at y^i with respect to x^k:

$$e_p^{k,i} := f_p(x^k) - f_p^i - \langle g_p^i, x^k - y^i\rangle \geq 0, \quad p = 1, 2. \tag{2.2}$$

We note that the second representation of Ψ_ℓ in (2.1) is better suited for implementations, due to lower storage requirements. As is readily seen from the definition of ε-subgradients, it holds that

$$g_p^i \in \partial_{e_p^{k,i}} f_p(x^k), \quad p = 1, 2, \tag{2.3}$$

and

$$(\sigma_k g_1^i + g_2^i) \in \partial_{(\sigma_k e_1^{k,i} + e_2^{k,i})} F_{\sigma_k}(x^k). \tag{2.4}$$

The linearization errors in (2.2) have to be properly updated every time x^k changes.

Choosing a proximal parameter $\mu_\ell > 0$, we generate the next candidate point y^ℓ by solving a quadratic programming (QP) reformulation of the problem

$$\min_{y\in\Re^n}\left\{\Psi_\ell(y)+\frac{1}{2}\mu_\ell\|y-x^k\|^2\right\}. \tag{2.5}$$

We note that the resulting quadratic program possesses a certain special structure, for which efficient software has been developed [16, 10]. The iterate y^ℓ is considered good enough when $F_{\sigma_k}(y^\ell)$ is sufficiently smaller than $F_{\sigma_k}(x^k)$ (the so-called "serious step"; this will be made precise later). If y^ℓ is acceptable, then we set $x^{k+1} := y^\ell$, choose new σ_{k+1}, and proceed to construct a descent step for $F_{\sigma_{k+1}}$. Otherwise, a so-called "null step" is declared and the procedure continues for F_{σ_k}, using the enhanced approximation $\Psi_{\ell+1}$.

In order for the basic idea outlined above to be practical, some important details have to be incorporated into the design of the method, as discussed next.

The number of constraints in the QP reformulation of (2.5) is precisely the number of elements in the bundle. Obviously, one has to keep this number computationally manageable. Thus, the bundle has to be *compressed* whenever the number of elements reaches some chosen bound. Reducing the bundle amounts to replacing the cutting-planes model (2.1) with another function, defined with a smaller number of cutting planes, which we shall still denote by Ψ_ℓ. This has to be done without impairing convergence of the algorithm. For this purpose, the so-called *aggregate function* is fundamental [3, Chap. 9], which we shall introduce in what follows.

It is convenient to split the information kept at iteration ℓ into two separate parts. One is the "oracle" bundle containing subgradient values at (some of!) points y^i, $i = 1,\dots,\ell-1$, and the associated linearization errors (recall (2.3) and (2.2)):

$$\mathcal{B}_\ell^{oracle} \subset \bigcup_{i<\ell}\left\{\left(e_p^{k,i}\in\Re_+, g_p^i\in\partial_{e_p^{k,i}} f_p(x^k),\ p=1,2\right)\right\}.$$

Note that here, the bundle $\mathcal{B}_\ell^{oracle}$ is not required to contain information at all the previous points (this is reflected by the use of the inclusion, rather than equation, in the definition above). The other part is the "aggregate" bundle, obtained from solutions of the QP subproblems. This bundle contains certain special ε-subgradients at x^k, to be introduced in Lemma 2.1 below. For now, we formally set

$$\mathcal{B}_\ell^{agg} \subset \bigcup_{i<\ell}\left\{\left(\hat\varepsilon_p^{k,i}\in\Re_+, \hat g_p^i\in\partial_{\hat\varepsilon_p^{k,i}} f_p(x^k),\ p=1,2\right)\right\},$$

without specifying how exactly those objects are obtained. Note that here there may no longer exist any previous point y^i, $i<\ell$, for which $\hat g_p^i \in \partial f_p(y^i)$, $p = 1, 2$.

The information in $\mathcal{B}_\ell^{oracle}$ and $\mathcal{B}_\ell^{agg}$ defines a cutting-planes approximation of F_{σ_k} given by

$$\begin{aligned}\Psi_\ell(y) = {}& \sigma_k f_1(x^k) + f_2(x^k)\\ &+\max\left\{\max_{i\in\mathcal{B}_\ell^{oracle}}\left\{-(\sigma_k e_1^{k,i}+e_2^{k,i})+\langle\sigma_k g_1^i+g_2^i, y-x^k\rangle\right\},\right.\\ &\left.\max_{i\in\mathcal{B}_\ell^{agg}}\left\{-(\sigma_k\hat\varepsilon_1^{k,i}+\hat\varepsilon_2^{k,i})+\langle\sigma_k\hat g_1^i+\hat g_2^i, y-x^k\rangle\right\}\right\},\end{aligned} \tag{2.6}$$

where by $i \in \mathcal{B}_\ell^{oracle}$ we mean that there exists an element in the set $\mathcal{B}_\ell^{oracle}$ indexed by i; and similarly for $\mathcal{B}_\ell^{agg}$. Although this notation is formally improper (the bundles are not sets of indices), it does not lead to any confusion while simplifying the formulas.

We next discuss properties of the solution of QP subproblem (2.5) with Ψ_ℓ given by (2.6). The following characterization is an adaptation of [3, Lemma 9.8] for our setting.

LEMMA 2.1. *For the unique solution y^ℓ of (2.5) with Ψ_ℓ given by (2.6), it holds that*

(i) $y^\ell = x^k - \frac{1}{\mu_\ell}(\sigma_k \hat{g}_1^\ell + \hat{g}_2^\ell)$;
(ii) $\hat{g}_p^\ell = \sum_{i \in \mathcal{B}_\ell^{oracle}} \lambda_i^\ell g_p^i + \sum_{i \in \mathcal{B}_\ell^{agg}} \hat{\lambda}_i^\ell \hat{g}_p^i$, $p = 1, 2$,
where $\lambda^\ell \geq 0$, $\hat{\lambda}^\ell \geq 0$ and $\sum_{i \in \mathcal{B}_\ell^{oracle}} \lambda_i^\ell + \sum_{i \in \mathcal{B}_\ell^{agg}} \hat{\lambda}_i^\ell = 1$;
(iii) $(\sigma_k \hat{g}_1^\ell + \hat{g}_2^\ell) \in \partial \Psi_\ell(y^\ell)$;
(iv) $\hat{g}_p^\ell \in \partial_{\hat{\varepsilon}_p^{k,\ell}} f_p(x^k)$, *where* $\hat{\varepsilon}_p^{k,\ell} = \sum_{i \in \mathcal{B}_\ell^{oracle}} \lambda_i^\ell e_p^{k,i} + \sum_{i \in \mathcal{B}_\ell^{agg}} \hat{\lambda}_i^\ell \hat{\varepsilon}_p^{k,i}$, $p = 1, 2$;
(v) $(\sigma_k \hat{g}_1^\ell + \hat{g}_2^\ell) = \hat{g}^\ell \in \partial_{\hat{\varepsilon}^{k,\ell}} F_{\sigma_k}(x^k)$, *where* $\hat{\varepsilon}^{k,\ell} = \sigma_k \hat{\varepsilon}_1^{k,\ell} + \hat{\varepsilon}_2^{k,\ell}$;
(vi) $\hat{\varepsilon}^{k,\ell} = F_{\sigma_k}(x^k) - \Psi_\ell(y^\ell) - \frac{1}{\mu_\ell}\|\hat{g}^\ell\|^2 \geq 0$.

Proof. The assertions can be verified following the analysis in [3, Lemma 9.8], and taking into account the special structure of the function F_{σ_k} and of its approximation Ψ_ℓ. We omit the details. ∎

We note that λ^ℓ and $\hat{\lambda}^\ell$ in Lemma 2.1 are the Lagrange multipliers associated with y^ℓ in the quadratic program reformulation of (2.5) (or the problem variables, if one solves the dual of this quadratic program, as in [16]). In any case, λ^ℓ and $\hat{\lambda}^\ell$ are available as part of the solution to (2.5). The quantities $\hat{g}_p^\ell$, $\hat{\varepsilon}_p^{k,\ell}$, $p = 1, 2$, defined in Lemma 2.1 are precisely the ones that appear in the definition of $\mathcal{B}_\ell^{agg}$ (except that $\mathcal{B}_\ell^{agg}$ contains information computed at iterations previous to the ℓth; at the first iteration we formally set $\mathcal{B}_\ell^{agg} = \emptyset$). We are now ready to introduce the aggregate function, already mentioned above:

$$l_{k,\ell}(y) := \sigma_k f_1(x^k) + f_2(x^k) - (\sigma_k \hat{\varepsilon}_1^{k,\ell} + \hat{\varepsilon}_2^{k,\ell}) + \langle \sigma_k \hat{g}_1^\ell + \hat{g}_2^\ell, y - x^k \rangle,$$

where

$$\hat{g}_p^\ell \in \partial_{\hat{\varepsilon}_p^{k,\ell}} f_p(x^k), \quad p = 1, 2, \tag{2.7}$$

and consequently,

$$(\sigma_k \hat{g}_1^\ell + \hat{g}_2^\ell) \in \partial_{(\sigma_k \hat{\varepsilon}_1^{k,\ell} + \hat{\varepsilon}_2^{k,\ell})} F_{\sigma_k}(x^k). \tag{2.8}$$

As already noted above, this function is defined directly from the quantities available after solving (2.5).

As pointed out in [6, eqs. (4.7)–(4.9)], to guarantee that a bundle technique would be able to construct a descent step for F_{σ_k} with respect to its value at x^k (assuming x^k is not a minimizer of F_{σ_k}) one can actually use any cutting-planes models Ψ_ℓ satisfying (for all $y \in \Re^n$) the following three conditions:

$$\begin{aligned} \Psi_\ell(y) &\leq F_{\sigma_k}(y) && \text{for all } \ell \geq 1 \text{ and all } k, \\ l_{k,\ell}(y) &\leq \Psi_{\ell+1}(y) && \text{for those } \ell \text{ for which } y^\ell \text{ is a null step}, \\ \sigma_k f_1^\ell + f_2^\ell + \langle \sigma_k g_1^\ell + g_2^\ell, y - y^\ell \rangle &\leq \Psi_{\ell+1}(y) && \text{for those } \ell \text{ for which } y^\ell \text{ is a null step}. \end{aligned}$$

The last two conditions mean that when defining the new bundles, it is enough for $\mathcal{B}_{\ell+1}^{oracle}$ to contain the cutting plane computed at the new point y^ℓ (i.e., the subgradients g_1^ℓ, g_2^ℓ, and the associated linearization errors $e_1^{k,\ell}$, $e_2^{k,\ell}$) and for $\mathcal{B}_{\ell+1}^{agg}$ to contain

the last aggregate function $l_{k,\ell}$ (i.e., the ε-subgradients $\hat{g}_1^\ell$, $\hat{g}_2^\ell$, and the associated $\hat{\varepsilon}_1^{k,\ell}$, $\hat{\varepsilon}_2^{k,\ell}$). In particular, at any iteration, the bundle can contain as few elements as we wish (as long as the two specified above are included). This fact is crucial for effective control of the size of subproblems (2.5). Finally, to make sure that the first condition above holds for all k, the linearization and aggregate errors have to be properly updated every time x^k changes to x^{k+1} (in particular, to ensure the key relations (2.4) and (2.8)). As is readily seen, the following formulas do the job:

(2.9)
$$e_p^{k+1,i} = e_p^{k,i} + f_p(x^{k+1}) - f_p(x^k) + \langle g_p^i, x^k - x^{k+1}\rangle, \quad p = 1,2, \text{ for } i \in \mathcal{B}_{\ell+1}^{oracle},$$
$$\hat{\varepsilon}_p^{k+1,i} = \hat{\varepsilon}_p^{k,i} + f_p(x^{k+1}) - f_p(x^k) + \langle \hat{g}_p^i, x^k - x^{k+1}\rangle, \quad p = 1,2, \text{ for } i \in \mathcal{B}_{\ell+1}^{agg}.$$

We are now ready to formally state the algorithm.

ALGORITHM 2.1 (bilevel bundle method).

Step 0. `Initialization.`
Choose parameter $m \in (0,1)$ *and an integer* $|\mathcal{B}|_{max} \geq 2$.
Choose $x^0 \in \Re^n$ *and* $\sigma_0 > 0$, $\beta_0 > 0$. *Set* $y^0 := x^0$ *and compute* f_p^0, g_p^0, $p = 1,2$. *Set* $k = 0$, $\ell = 1$, $e_p^{0,0} := 0$, $p = 1,2$. *Define the starting bundles* $\mathcal{B}_1^{oracle} := \{(e_p^{0,0}, g_p^0,\ p = 1,2)\}$ *and* $\mathcal{B}_1^{agg} := \emptyset$.

Step 1. `QP subproblem.`
Choose $\mu_\ell > 0$ *and compute* y^ℓ *as the solution of* (2.5), *where* Ψ_ℓ *is defined by* (2.6). *Compute*

$$\hat{g}^\ell = \mu_\ell(x^k - y^\ell), \quad \hat{\varepsilon}^{k,\ell} = F_{\sigma_k}(x^k) - \Psi_\ell(y^\ell) - \frac{1}{\mu_\ell}\|\hat{g}^\ell\|^2, \quad \delta_\ell = \hat{\varepsilon}^{k,\ell} + \frac{1}{2\mu_\ell}\|\hat{g}^\ell\|^2.$$

Compute f_p^ℓ, g_p^ℓ, $p = 1,2$. *Compute* $e_p^{k,\ell}$, $p = 1,2$, *using* (2.2) *written with* $i = \ell$.

Step 2. `Descent test.` *If*

$$F_{\sigma_k}(y^\ell) \leq F_{\sigma_k}(x^k) - m\delta_\ell, \tag{2.10}$$

then declare a serious step. *Otherwise, declare a* null step.

Step 3. `Bundle management.`
Set $\mathcal{B}_{\ell+1}^{oracle} := \mathcal{B}_\ell^{oracle}$ *and* $\mathcal{B}_{\ell+1}^{agg} := \mathcal{B}_\ell^{agg}$. *If the bundle has reached the maximum size (i.e., if* $|\mathcal{B}_{\ell+1}^{oracle} \cup \mathcal{B}_{\ell+1}^{agg}| = |\mathcal{B}|_{max}$*), then delete at least two elements from* $\mathcal{B}_{\ell+1}^{oracle} \cup \mathcal{B}_{\ell+1}^{agg}$ *and append the aggregate information* $(\hat{\varepsilon}_p^{\ell,k}, \hat{g}_p^\ell,\ p = 1,2)$ *to* $\mathcal{B}_{\ell+1}^{agg}$.
In any case, append $(e_p^{k,\ell}, g_p^\ell,\ p = 1,2)$ *to* $\mathcal{B}_{\ell+1}^{oracle}$.

Step 4. *If* `Descent test` *was satisfied,*
set $x^{k+1} = y^\ell$ *and choose* $0 < \sigma_{k+1} \leq \sigma_k$ *and* $0 < \beta_{k+1} \leq \beta_k$.
Update the linearization and aggregate errors using (2.9).
Set $k = k + 1$ *and go to Step* 5.
If

$$\max\{\hat{\varepsilon}^{k,\ell}, \|\hat{g}^\ell\|\} \leq \beta_k\sigma_k, \tag{2.11}$$

choose $0 < \sigma_{k+1} < \sigma_k$ *and* $0 < \beta_{k+1} < \beta_k$.
Set $x^{k+1} = x^k$, $k = k + 1$ *and go to Step* 5.

Step 5. *Set* $\ell = \ell + 1$ *and go to Step* 1.

The role of checking condition (2.11) is to detect the situation when the point x^k happens to be a minimizer of the function F_{σ_k} (or is almost a minimizer; recall Lemma 2.1(v)). If it is so, we immediately update the parameter σ_k. This is reasonable, since we are not interested in minimizing F_{σ_k}. The case of x^k being a minimizer of F_{σ_k}, however, is very unlikely to occur, since for no iteration k the function F_{σ_k} is being minimized with any prescribed precision. This is also confirmed by our numerical experiments in section 4, where we ignored the safeguard (2.11) in our implementation.

The algorithm does not have an overall stopping test. In the unconstrained case, a reliable stopping test is one of the important advantages of bundle methods (as compared, for example, to subgradient methods). However, lack of a stopping test in our setting cannot be considered to be a drawback of the algorithm. Indeed, a bilevel problem does not admit an explicit optimality condition. Actually, the same is in general already true for constrained optimization without a regularity assumption on the constraints (except for some special cases, of course). As a result, there is no explicit way to measure violation/satisfaction of optimality in (1.1), and, consequently, lack of a stopping test is inherent in the nature of the problem.

We note that there is certain freedom in updating or not updating the parameter σ_k after every iteration. While our goal is to show that we can update it after a single descent step, note that, in principle, we are not obliged to do so ($\sigma_{k+1} = \sigma_k$ is allowed, unless (2.11) holds; in the latter case, x^k almost minimizes F_{σ_k} and it does not make sense to insist on further descent for this function). For convergence, it would be required that σ_k not go to zero too fast, in the sense of condition (1.5) stated above. In the case of the standard optimization problem (1.2), this condition allows a natural interpretation. In order to be able to trace the optimal penalty path $x(\sigma)$ with such a relaxed precision (making just one descent step for each penalized subproblem (1.6)), we should not be jumping too far from the target $x(\sigma_k)$ on the path to the next target $x(\sigma_{k+1})$ as we move along. On the other hand, if σ_k is kept constant over a few descent iterations, this allows for a more rapid change in the parameter for the next iteration, while still guaranteeing the second condition in (1.5). This is intuitively reasonable: if we get closer to the optimal path, then the target can be moved further. In our numerical experiments in section 4, we have used the simplest generic choice of $\sigma_k = \sigma_0/(k+1)$. We have experimented with some other options (for example, keeping the parameter unchanged for some iterations), but found that this does not make much difference (for our test problems). We shall discuss this further in section 4.

3. Convergence analysis. In our convergence analysis, we assume that the objective function f_1 is bounded below; i.e.,

$$-\infty < \bar{f}_1 = \inf\{f_1(x) \mid x \in \Re^n\}.$$

Since we also assume that the problem is solvable, the function f_2 is automatically bounded below, and we define

$$-\infty < \bar{f}_2 = \min\{f_2(x) \mid x \in \Re^n\}.$$

For the subsequent analysis, it is convenient to think of Algorithm 2.1 as "applied" to the shifted function

$$F_\sigma(x) = \sigma(f_1(x) - \bar{f}_1) + (f_2(x) - \bar{f}_2), \tag{3.1}$$

instead of the function F_σ given by (1.4), as stated originally. We can do this because Algorithm 2.1 would generate the same iterates whether F_σ were given by (3.1) or

(1.4). Indeed, the two functions have the same subgradients and the same difference for function values at any two points. Hence, the cutting-planes models (2.6) for the two functions would differ by a constant term (not dependent on y). This means that solutions y^ℓ of QP subproblems (2.5) would be the same, as well as the quantities $\hat{g}^\ell$ and $\hat{\varepsilon}^{k,\ell}$, which are defined by those solutions. Therefore, the relations in (2.10) and (2.11), which are guiding the algorithm, also do not change. From now on, we consider that the method is "applied" to function F_σ defined by (3.1) (even though the function from (1.4) is used in reality, of course). This is convenient for the subsequent analysis and should not lead to any confusion.

We proceed to prove convergence of the algorithm.

PROPOSITION 3.1. *Let f_1 and f_2 be convex functions.*

If for consecutive null steps it holds that $\bar{\mu} \geq \mu_{\ell+1} \geq \mu_\ell > 0$, then Algorithm 2.1 *is well defined and either* (2.10) *or* (2.11) *(or both) hold infinitely often. In particular, the parameter σ_k is updated infinitely often.*

Proof. Let k be any iteration index and consider the sequence of null steps applied to the current (fixed over those null steps) function F_{σ_k}. By properties of standard bundle methods (e.g., [3, Thm. 9.15]), it holds that either the descent test (2.10) is satisfied after a finite number of null steps, or x^k is a minimizer of F_{σ_k}. In the latter case, it further holds that $\delta_\ell \to 0$ as $\ell \to \infty$. Hence, $\hat{g}^\ell \to 0$ and $\hat{\varepsilon}^{k,\ell} \to 0$ as $\ell \to \infty$. This means that the condition (2.11) would be satisfied after a finite number of null steps.

We have therefore established that either (2.10) or (2.11) is guaranteed to be satisfied after a finite number of null steps. This shows that the method is well defined and updates σ_k infinitely often. ∎

We next prove that the generated sequence $\{x^k\}$ is bounded and its accumulation points are feasible for problem (1.1).

PROPOSITION 3.2. *Let f_1 and f_2 be convex functions such that f_1 is bounded below on $\Re^n$ and the solution set S_1 of problem* (1.1) *is nonempty and bounded.*

Suppose that $\bar{\mu} \geq \mu_\ell \geq \hat{\mu} > 0$ for all iterations ℓ, that $\mu_{\ell+1} \geq \mu_\ell$ on consecutive null steps, and that $\sigma_k \to 0$ as $k \to \infty$.

Then any sequence $\{x^k\}$ generated by Algorithm 2.1 *is bounded and all its accumulation points are feasible for problem* (1.1); *i.e., they belong to S_2.*

Proof. If the serious step descent test (2.10) is satisfied only a finite number of times, it is readily seen that there exists some iteration index k_0 such that $x^k = x^{k_0}$ for all $k \geq k_0$ (because x^k is changed only at serious steps, i.e., when (2.10) holds). Hence, in this case $\{x^k\}$ is trivially bounded.

Assume now that (2.10) is satisfied infinitely often. In what follows, we consider the subsequence of indices k at which (2.10) holds, i.e., at which x^k changes. But to simplify the notation, we shall not introduce this subsequence explicitly. Here, we can simply disregard all the iterations at which x^k remained fixed. We can do this within the current analysis of boundedness of $\{x^k\}$, because those iterations merely changed σ_k (and the only assumption for the latter used below is that it should be nonincreasing—the property which holds for any subsequence of $\{\sigma_k\}$ by the construction of the method).

For each k, let $\ell(k)$ be the index ℓ for which (2.10) was satisfied (in particular, $x^{k+1} = y^{\ell(k)}$). By (2.10), it holds that

$$\begin{aligned} m\delta_{\ell(k)} &\leq F_{\sigma_k}(x^k) - F_{\sigma_k}(x^{k+1}) \\ &= \sigma_k(f_1(x^k) - \bar{f}_1) - \sigma_k(f_1(x^{k+1}) - \bar{f}_1) \\ &\quad + (f_2(x^k) - \bar{f}_2) - (f_2(x^{k+1}) - \bar{f}_2). \end{aligned}$$

Summing up the latter inequalities for $k = 0, \dots, k_1$, we obtain that

$$\begin{aligned} m\sum_{k=0}^{k_1}\delta_{\ell(k)} &\le \sigma_0(f_1(x^0)-\bar f_1) + \sum_{k=0}^{k_1-1}(\sigma_{k+1}-\sigma_k)(f_1(x^{k+1})-\bar f_1) \\ &\quad - \sigma_{k_1}(f_1(x^{k_1+1})-\bar f_1) + (f_2(x^0)-\bar f_2) - (f_2(x^{k_1+1})-\bar f_2) \\ &\le \sigma_0(f_1(x^0)-\bar f_1) + (f_2(x^0)-\bar f_2), \end{aligned}$$

where we have used the facts that, for all k, $f_1(x^k) \ge \bar f_1$, $f_2(x^k) \ge \bar f_2$, and $0 < \sigma_{k+1} \le \sigma_k$. Letting $k_1 \to \infty$, we conclude that

$$\sum_{k=0}^{\infty}\delta_{\ell(k)} \le m^{-1}(\sigma_0(f_1(x^0)-\bar f_1) + (f_2(x^0)-\bar f_2)) < +\infty. \tag{3.2}$$

In particular,

$$\delta_{\ell(k)} \to 0 \quad \text{as } k \to \infty. \tag{3.3}$$

Take any $\bar x \in S_1 \ne \emptyset$. Using Lemma 2.1(i), we obtain that

$$\begin{aligned} \|x^{k+1}-\bar x\|^2 &= \|x^k-\bar x\|^2 - \frac{2}{\mu_{\ell(k)}}\langle \hat g^{\ell(k)}, x^k-\bar x\rangle + \frac{1}{\mu_{\ell(k)}^2}\|\hat g^{\ell(k)}\|^2 \\ &\le \|x^k-\bar x\|^2 + \frac{2}{\mu_{\ell(k)}}\left(F_{\sigma_k}(\bar x) - F_{\sigma_k}(x^k) + \hat\varepsilon^{k,\ell(k)} + \frac{1}{2\mu_{\ell(k)}}\|\hat g^{\ell(k)}\|^2\right) \\ &= \|x^k-\bar x\|^2 + \frac{2}{\mu_{\ell(k)}}\delta_{\ell(k)} \\ &\quad + \frac{2}{\mu_{\ell(k)}}\left(\sigma_k(f_1(\bar x)-f_1(x^k)) + f_2(\bar x) - f_2(x^k)\right) \\ &\le \|x^k-\bar x\|^2 + \frac{2}{\mu_{\ell(k)}}\delta_{\ell(k)} + \frac{2\sigma_k}{\mu_{\ell(k)}}\left(f_1(\bar x)-f_1(x^k)\right), \end{aligned} \tag{3.4}$$

where the first inequality is by Lemma 2.1(v), and the last is by the fact that $f_2(\bar x) \le f_2(x^k)$, since $\bar x \in S_1 \subset S_2$.

We next consider separately the following two possible cases:

Case 1. There exists k_2 such that $f_1(\bar x) \le f_1(x^k)$ for all $k \ge k_2$.

Case 2. For each k, there exists $k_3 \ge k$ such that $f_1(\bar x) > f_1(x^{k_3})$.

Case 1. For $k \ge k_2$, we obtain from (3.4) that

$$\|x^{k+1}-\bar x\|^2 \le \|x^k-\bar x\|^2 + \frac{2}{\hat\mu}\delta_{\ell(k)}. \tag{3.5}$$

Recalling (3.2), we conclude that $\{\|x^k-\bar x\|^2\}$ converges (see, e.g., [23, Lem. 2, p. 44]). Hence, $\{x^k\}$ is bounded.

Case 2. For each k, define

$$i_k = \max\{i \le k \mid f_1(\bar x) > f_1(x^i)\}.$$

In the case under consideration, it holds that $i_k \to \infty$ when $k \to \infty$.

We first show that $\{x^{i_k}\}$ is bounded. Observe that

$$\begin{aligned} S_1 &= \{x \in S_2 \mid f_1(x) \le f_1(\bar x)\} \\ &= \{x \in \Re^n \mid \max\{f_2(x)-\bar f_2, f_1(x)-f_1(\bar x)\} \le 0\}. \end{aligned}$$

By assumption, the set S_1 is nonempty and bounded. Therefore, the convex function

$$\phi : \Re^n \to \Re, \quad \phi(x) = \max\{f_2(x) - \bar{f}_2, f_1(x) - f_1(\bar{x})\}$$

has a particular level set $\{x \in \Re^n \mid \phi(x) \le 0\}$ which is nonempty and bounded. It follows that all level sets of ϕ are bounded (see, e.g., [2, Prop. 2.3.1]), i.e.,

$$L_\phi(c) = \{x \in \Re^n \mid \phi(x) \le c\}$$

is bounded for any $c \in \Re$.

Since $f_1(x) - \bar{f}_1 \ge 0$ for all $x \in \Re^n$ and $0 < \sigma_{k+1} \le \sigma_k$, it holds that

$$F_{\sigma_{k+1}}(x) \le F_{\sigma_k}(x) \qquad \text{for all } x \in \Re^n.$$

Hence,

$$0 \le F_{\sigma_{k+1}}(x^{k+1}) \le F_{\sigma_k}(x^{k+1}) \le F_{\sigma_k}(x^k),$$

where the third inequality follows from (2.10). The above relations show that $\{F_{\sigma_k}(x^k)\}$ is nonincreasing and bounded below. Hence, it converges. It then easily follows that $\{f_2(x^k) - \bar{f}_2\}$ is bounded (because both terms in $F_{\sigma_k}(x^k) = \sigma_k(f_1(x^k) - \bar{f}_1) + (f_2(x^k) - \bar{f}_2)$ are nonnegative).

Fix any $c \ge 0$ such that $f_2(x^k) - \bar{f}_2 \le c$ for all k. Since $f_1(x^{i_k}) - f_1(\bar{x}) < 0 \le c$ (by the definition of the index i_k), we have that $x^{i_k} \in L_\phi(c)$, which is a bounded set. This shows that $\{x^{i_k}\}$ is bounded.

By the definition of i_k, it further holds that

$$f_1(\bar{x}) \le f_1(x^i), \quad i = i_k + 1, \ldots, k \quad (\text{if } k > i_k).$$

Hence, from (3.4), we have that

$$\|x^{i+1} - \bar{x}\|^2 \le \|x^i - \bar{x}\|^2 + \frac{2}{\hat{\mu}}\delta_{\ell(i)}, \quad i = i_k + 1, \ldots, k.$$

Therefore, for any k, it holds that

$$\begin{aligned} \|x^k - \bar{x}\|^2 &\le \|x^{i_k} - \bar{x}\|^2 + \frac{2}{\hat{\mu}} \sum_{i=i_k+1}^{k-1} \delta_{\ell(i)} \\ &\le \|x^{i_k} - \bar{x}\|^2 + \frac{2}{\hat{\mu}} \sum_{i=i_k+1}^{\infty} \delta_{\ell(i)}. \end{aligned} \tag{3.6}$$

Recalling that $i_k \to \infty$, by (3.2) we have that

$$\sum_{i=i_k+1}^{\infty} \delta_{\ell(i)} \to 0 \quad \text{as } k \to \infty. \tag{3.7}$$

Taking also into account the boundedness of $\{x^{i_k}\}$, the relation (3.6) implies that the whole sequence $\{x^k\}$ is bounded.

We next show that all accumulation points of $\{x^k\}$ belong to S_2. For each k, either (2.10) or (2.11) holds. Regardless of whether both conditions hold infinitely often or only one does, it is easy to see that

$$\hat{g}^{\ell(k)} \to 0 \quad \text{and} \quad \hat{\varepsilon}^{k,\ell(k)} \to 0 \quad \text{as } k \to \infty, \tag{3.8}$$

where (3.3) is used if (2.10) holds infinitely often, and (2.11) is used directly.

Let $x \in \Re^n$ be arbitrary but fixed. By Lemma 2.1(v),

$$\sigma_k f_1(x) + f_2(x) \geq \sigma_k f_1(x^k) + f_2(x^k) + \langle \hat{g}^{\ell(k)}, x - x^k \rangle - \hat{\varepsilon}^{k,\ell(k)}. \tag{3.9}$$

Let x^∞ be any accumulation point of $\{x^k\}$. Using boundedness of $\{x^k\}$, the continuity of f_1 and f_2, the fact that $\sigma_k \to 0$ and (3.8), and passing onto the limit in (3.9) along the subsequence which converges to x^∞, we obtain that $f_2(x) \geq f_2(x^\infty)$, where $x \in \Re^n$ is arbitrary. Hence, $x^\infty \in S_2$. □

The rest of the proof is done separately for the following two cases: the number of serious steps when (2.10) is satisfied is either infinite or finite.

THEOREM 3.3. *Let f_1 and f_2 be convex functions such that f_1 is bounded below on $\Re^n$ and the solution set S_1 of problem* (1.1) *is nonempty and bounded.*

Suppose that $\bar{\mu} \geq \mu_\ell \geq \hat{\mu} > 0$ for all iterations ℓ, and that $\mu_{\ell+1} \geq \mu_\ell$ on consecutive null steps.

If serious step descent test (2.10) *is satisfied an infinite number of times and we choose $\{\sigma_k\}$ according to* (1.5) *and $\{\beta_k\} \to 0$ as $k \to \infty$, then* $\operatorname{dist}(x^k, S_1) \to 0$ *as $k \to \infty$, and all accumulation points of $\{x^k\}$ are solutions of* (1.1).

Proof. Take any $\bar{x} \in S_1$. We again consider separately the two possible cases introduced in the proof of Proposition 3.2:

Case 1. There exists k_2 such that $f_1(\bar{x}) \leq f_1(x^k)$ for all $k \geq k_2$.

Case 2. For each k, there exists $k_3 \geq k$ such that $f_1(\bar{x}) > f_1(x^{k_3})$.

Case 2. Recalling that $i_k = \max\{i \leq k \mid f_1(\bar{x}) > f_1(x^i)\}$ so that $f_1(x^{i_k}) < f_1(\bar{x})$, by the continuity of f_1 it holds that $f_1(x^\infty) \leq f_1(\bar{x})$ for any accumulation point x^∞ of $\{x^{i_k}\}$. Since all accumulation points of $\{x^k\}$ belong to S_2 (as established in Proposition 3.2), it must be the case that all accumulation points of $\{x^{i_k}\}$ are solutions of the problem. In particular,

$$\operatorname{dist}(x^{i_k}, S_1) \to 0 \quad \text{as } k \to \infty. \tag{3.10}$$

For each k, define $\bar{x}^k = P_{S_1}(x^{i_k})$. Using (3.6) with $\bar{x} = \bar{x}^k$ gives

$$\begin{aligned} \operatorname{dist}(x^k, S_1)^2 &\leq \|x^k - \bar{x}^k\|^2 \\ &\leq \operatorname{dist}(x^{i_k}, S_1)^2 + \frac{2}{\hat{\mu}} \sum_{i=i_k+1}^{\infty} \delta_{\ell(i)}. \end{aligned}$$

Passing onto the limit in the latter relation as $k \to \infty$, and using (3.7) and (3.10), we obtain that $\operatorname{dist}(x^k, S_1) \to 0$.

Case 1. As has been shown in Proposition 3.2, in this case the sequence $\{\|x^k - \bar{x}\|\}$ converges for any $\bar{x} \in S_1$. Therefore, if we establish that $\{x^k\}$ has an accumulation point $x^\infty \in S_1$, it would immediately follow that $\{\|x^k - x^\infty\|\} \to 0$; i.e., the whole sequence $\{x^k\}$ converges to $x^\infty \in S_1$.

Suppose first that (2.11) is satisfied only a finite number of times. Suppose further that there is no accumulation point of $\{x^k\}$ which solves (1.1). Since, by Proposition 3.2, all accumulation points are feasible for (1.1), the second assumption means that $\liminf_{k\to\infty} f_1(x^k) > f_1(\bar{x})$, where $\bar{x} \in S_1$. In particular, there exists $t > 0$ such that

$f_1(\bar{x}) \le f_1(x^k) - t$ for all $k \ge k_4$. We then obtain from (3.4) that for $k > k_4$, it holds that

$$\begin{aligned}\|x^{k+1} - \bar{x}\|^2 &\le \|x^k - \bar{x}\|^2 + \frac{2}{\hat{\mu}}\delta_{\ell(k)} - \frac{2t}{\bar{\mu}}\sigma_k \\ &\le \|x^{k_4} - \bar{x}\|^2 + \frac{2}{\hat{\mu}}\sum_{i=k_4-1}^{k}\delta_{\ell(i)} - \frac{2t}{\bar{\mu}}\sum_{i=k_4-1}^{k}\sigma_i.\end{aligned}$$

Passing onto the limit when $k \to \infty$ in the latter relation, we obtain

$$\frac{2t}{\bar{\mu}}\sum_{i=k_4-1}^{\infty}\sigma_i \le \|x^{k_4} - \bar{x}\|^2 + \frac{2}{\hat{\mu}}\sum_{i=k_4-1}^{\infty}\delta_{\ell(i)},$$

which is a contradiction, due to (3.2) and (1.5). Hence, $\liminf_{k\to\infty} f_1(x^k) = f_1(\bar{x})$. Since $\{x^k\}$ is bounded, it must have an accumulation point x^∞ such that $f_1(x^\infty) = f_1(\bar{x})$. As $x^\infty \in S_2$, this means that $x^\infty \in S_1$.

Finally, suppose that (2.11) is satisfied an infinite number of times. Consider the subsequence of indices k for which (2.11) holds (we shall not specify it explicitly) and let $\ell(k)$ denote the associated index ℓ in (2.11). We have that

$$\max\{\sigma_k^{-1}\hat{\varepsilon}^{k,\ell(k)}, \sigma_k^{-1}\|\hat{g}^{\ell(k)}\|\} \le \beta_k, \quad \beta_k \to 0. \tag{3.11}$$

Taking any $x \in S_2$ and using Lemma 2.1(v), we have that

$$\sigma_k f_1(x) + f_2(x) \ge \sigma_k f_1(x^k) + f_2(x^k) + \langle \hat{g}^{\ell(k)}, x - x^k\rangle - \hat{\varepsilon}^{k,\ell(k)}.$$

Since $f_2(x) \le f_2(x^k)$ for any $x \in S_2$, we obtain

$$f_1(x) \ge f_1(x^k) + \langle \sigma_k^{-1}\hat{g}^{\ell(k)}, x - x^k\rangle - \sigma_k^{-1}\hat{\varepsilon}^{k,\ell(k)}. \tag{3.12}$$

Hence, passing onto the limit in (3.12) as $k \to \infty$ along some subsequence converging to x^∞ and taking into account (3.11), we conclude that $f_1(x) \ge f_1(x^\infty)$ for any $x \in S_2$. Therefore, $x^\infty \in S_1$ also in this case, which concludes the proof. ∎

It remains to consider the case of a finite number of serious steps in Algorithm 2.1. As already discussed above, this is rather unlikely to occur. Actually, as the next result shows, it can happen only if we hit an exact solution of the problem, which is generally an exceptional situation.

THEOREM 3.4. *Let f_1 and f_2 be convex functions such that f_1 is bounded below on $\Re^n$ and the solution set S_1 of problem (1.1) is nonempty and bounded.*

Suppose that $\bar{\mu} \ge \mu_\ell \ge \hat{\mu} > 0$ for all iterations ℓ, and that $\mu_{\ell+1} \ge \mu_\ell$ on consecutive null steps.

If the serious step descent test (2.10) is satisfied a finite number of times and we choose $\{\sigma_k\} \to 0$ and $\{\beta_k\} \to 0$ as $k \to \infty$, then there exists an iteration index k_0 such that $x^k = x^{k_0}$ for all $k \ge k_0$ and $x^{k_0} \in S_1$.

Proof. Since x^k is changed only when (2.10) holds, it is readily seen that $x^k = x^{k_0}$ for all $k \ge k_0$. By Proposition 3.2, we have that $x^{k_0} \in S_2$.

By Proposition 3.1, we have that for all $k \ge k_0$, σ_k is updated when (2.11) holds. For each k, let $\ell(k)$ denote the index ℓ for which (2.11) is satisfied. We have that

$$\max\{\sigma_k^{-1}\hat{\varepsilon}^{k,\ell(k)}, \sigma_k^{-1}\|\hat{g}^{\ell(k)}\|\} \le \beta_k, \quad \beta_k \to 0. \tag{3.13}$$

Taking any $x \in S_2$ and using Lemma 2.1(v), we have that

$$\sigma_k f_1(x) + f_2(x) \geq \sigma_k f_1(x^{k_0}) + f_2(x^{k_0}) + \langle \hat{g}^{\ell(k)}, x - x^{k_0} \rangle - \hat{\varepsilon}^{k,\ell(k)}.$$

Since $f_2(x) = f_2(x^{k_0})$ for any $x \in S_2$, we obtain

$$f_1(x) \geq f_1(x^{k_0}) + \langle \sigma_k^{-1} \hat{g}^{\ell(k)}, x - x^{k_0} \rangle - \sigma_k^{-1} \hat{\varepsilon}^{k,\ell(k)}. \tag{3.14}$$

Hence, passing onto the limit in (3.14) as $k \to \infty$ and taking into account (3.13), we conclude that $f_1(x) \geq f_1(x^{k_0})$ for any $x \in S_2$. Therefore, $x^{k_0} \in S_1$, as claimed. ∎

4. Computational experiments. In this section, we report on some numerical experiments for the problem of minimizing a piecewise quadratic convex function over a set defined by monotone linear complementarity constraints. Specifically, we consider the problem

$$\begin{array}{ll} \text{minimize} & \max_{j=1,\ldots,l}\{\langle A^j x, x\rangle + \langle b^j, x\rangle + c^j\} \\ \text{subject to} & Qx + q \geq 0,\ x \geq 0,\ \langle x, Qx + q\rangle \leq 0, \end{array} \tag{4.1}$$

where Q and A^j, $j = 1, \ldots, l$, are $n \times n$ positive semidefinite matrices; q and b^j, $j = 1, \ldots, l$, are vectors in $\Re^n$; and $c^j \in \Re$, $j = 1, \ldots, l$. This problem is converted to the setting of the paper by choosing

$$f_1(x) = \max_{j=1,\ldots,l}\{\langle A^j x, x\rangle + \langle b^j, x\rangle + c^j\},$$

$$f_2(x) = \sum_{i=1}^{n} \max\{-x_i, 0\} + \sum_{i=1}^{n} \max\{-(Qx+q)_i, 0\} + \max\{\langle Qx + q, x\rangle, 0\}.$$

The code is written in MATLAB, essentially by making modifications to a more-or-less standard unconstrained proximal bundle code. Runs are performed under MATLAB Version 7.0.0.19901 (R14). The test problems were constructed by first generating a feasible point $\bar{x}$ of (4.1), and then a function f_1 for which $\bar{x}$ is optimal. Details are presented next.

The process starts with defining an $n \times n$ positive semidefinite matrix Q of rank $r < n$, whose entries are uniformly distributed in the interval $[-5, 5]$. We next generate a point $\bar{x}$, with each coordinate having equal probability of being zero or being uniformly distributed in $[0, 5]$. Finally, we define $q = -Q\bar{x} + \bar{y}$, where a coordinate of $\bar{y}$ is zero if the corresponding coordinate of $\bar{x}$ is positive, while other coordinates of $\bar{y}$ have equal probability of being zero or uniformly generated from $[0, 5]$. As can be easily seen, such $\bar{x}$ is a feasible point for problem (4.1). It does not satisfy strict complementarity and, typically, is not an isolated feasible point (here, it is important that Q is a degenerate matrix). Obviously, $\bar{x}$ is an unconstrained minimizer of the function f_2, i.e., $\bar{x} \in S_2$.

Next, we construct a function f_1 such that $\bar{x}$ is a minimizer of f_1 over S_2. As the constraints in (4.1) do not satisfy a constraint qualification, we can only overestimate the tangent cone $T_{S_2}(\bar{x})$ to S_2 at $\bar{x}$, which gives underestimation of its dual:

$$(T_{S_2}(\bar{x}))^* \supset K = \operatorname{cone}\left(\{-e^i \mid \bar{x}_i = 0\} \cup \{-Q_i \mid \bar{y}_i = 0\} \cup \{q + (Q + Q^\top)\bar{x}\}\right), \tag{4.2}$$

where e^i is the ith element of the canonical basis of $\Re^n$, Q_i is the ith row of the matrix Q, and $\operatorname{cone}(X)$ stands for the conic hull of the set X in $\Re^n$.

We shall construct the needed function f_1 by defining antigradients of pieces of f_1 active at $\bar{x}$ as some elements belonging to the right-hand side of (4.2). This would guarantee the optimality condition

$$0 \in \partial f_1(\bar{x}) + (T_{S_2}(\bar{x}))^*, \tag{4.3}$$

even though the set $(T_{S_2}(\bar{x}))^*$ is not fully known. First, we generate symmetric $n \times n$ positive semidefinite matrices A^j, $j = 1, \ldots, l$, with random entries distributed in $[-5, 5]$. Choosing the number $l_0 \leq l$ of pieces of f_1 active at $\bar{x}$, we next define

$$b^j = -2A^j\bar{x} - u^j, \;\; u^j \in K, \quad j = 1, \ldots, l_0,$$

where elements u^j of K are generated by taking random coefficients in $[0, 1]$ for all vectors in the right-hand side of (4.2). The elements b^j, $j = l_0+1, \ldots, l$, are generated randomly.

It remains to make sure that the first l_0 pieces in the definition of f_1 are active at $\bar{x}$. To this end, we compute

$$\bar{c} = 5 + \max_{j=1,\ldots,l}\{\langle A^j\bar{x}, \bar{x}\rangle + \langle b^j, \bar{x}\rangle\},$$

and set

$$c^j = \bar{c} - \langle A^j\bar{x}, \bar{x}\rangle - \langle b^j, \bar{x}\rangle, \quad j = 1, \ldots, l_0,$$

$$c^j = 0, \quad j = l_0 + 1, \ldots, l.$$

It can be seen that for the point $\bar{x}$, the maximum in the definition of f_1 is attained for indices $j = 1, \ldots, l_0$, and that $f_1(\bar{x}) = \bar{c}$. By the previous constructions, we have that (4.3) holds, and thus $\bar{x}$ is a solution of (4.1). Furthermore, the optimal value of this problem is $\bar{c}$.

Our code is a slightly simplified version of Algorithm 2.1, in particular in the following two details. First, instead of an aggregation technique to control the bundle, we use simple selection of active pieces; i.e., after every iteration we discard those cutting planes which correspond to zero multipliers in the solution of the QP subproblem. Second, we ignore the safeguard (2.11) that detects when the current point x^k is almost a minimizer of F_{σ_k}, and so σ_k needs to be reduced (even if a serious step has not yet been constructed). As already discussed above, since at no iteration F_{σ_k} is being minimized to any specific precision, this situation is unlikely to occur prematurely if σ_k is updated after each serious step. This intuition was confirmed by our experiments. We observed that optimality is achieved only asymptotically, and so the standard bundle stopping test,

$$\hat{\varepsilon}^{k,\ell} \leq t_1 \quad \text{and} \quad \|\hat{g}^\ell\|^2 \leq t_2, \tag{4.4}$$

can be used without any harm. But, of course, one has to be aware that this stopping test cannot be fully reliable in our setting. In our experiments, we set $t_1 = 10^{-2}$ and $t_2 = 10^{-4}$, as it is often difficult to get more precision from a nondifferentiable optimization code in a simple MATLAB implementation. We start with $x^0 = (2, \ldots, 2)$, and set $m = 10^{-1}$ in the descent test (2.10). The proximal parameter μ_ℓ in (2.5) is changed at serious steps only by the safeguarded version of the *reversal quasi-Newton* scalar update; see [3, section 9.3.3]. More precisely,

$$\mu_{k+1} = \min\left\{c_1, \max\{\tilde{\mu}_{k+1}, c_2\}\right\},$$

TABLE 4.1
Summary of numerical experiments.

	Convergence (out of 20)		"Failures" (out of 20)	
$n = 5$	18 cases	38.3 oracle calls	2 cases	100 oracle calls
$\operatorname{rank} Q = 4$	$R_1 = 2.2 * 10^{-5}$	$R_2 = 1.2 * 10^{-5}$	$R_1 = 4.1 * 10^{-4}$	$R_2 = 2.2 * 10^{-4}$
$n = 5$	19 cases	32.2 oracle calls	1 case	100 oracle calls
$\operatorname{rank} Q = 2$	$R_1 = 6.2 * 10^{-4}$	$R_2 = 8.1 * 10^{-5}$	$R_1 = 3.2 * 10^{-4}$	$R_2 = 1.1 * 10^{-5}$
$n = 10$	12 cases	109.5 oracle calls	8 cases	200 oracle calls
$\operatorname{rank} Q = 8$	$R_1 = 2.8 * 10^{-5}$	$R_2 = 1.4 * 10^{-5}$	$R_1 = 2.2 * 10^{-4}$	$R_2 = 3.3 * 10^{-5}$
$n = 10$	14 cases	89.9 oracle calls	6 cases	200 oracle calls
$\operatorname{rank} Q = 5$	$R_1 = 3.7 * 10^{-4}$	$R_2 = 4.2 * 10^{-5}$	$R_1 = 7.2 * 10^{-4}$	$R_2 = 5.3 * 10^{-4}$
$n = 10$	16 cases	60.6 oracle calls	4 cases	200 oracle calls
$\operatorname{rank} Q = 2$	$R_1 = 9.8 * 10^{-4}$	$R_2 = 5.4 * 10^{-6}$	$R_1 = 2 * 10^{-3}$	$R_2 = 3.1 * 10^{-6}$

where $\tilde{\mu}_{k+1}$ is the value prescribed by [3, section 9.3.3], and $c_1 = 10$, $c_2 = 10^{-1}$. Subproblems (2.5) are solved by applying the MATLAB QP routine `qp.m` to the dual formulation of (2.5).

For updating the weight parameter, we use the simple generic choice

$$\sigma_k = \sigma_0/(k+1). \tag{4.5}$$

For lower dimensions (say, $n = 5$), when fewer iterations are expected, we start with $\sigma_0 = 10$. For higher dimensions (say, $n = 10$), when more iterations are typically needed, we start with $\sigma_0 = 20$. We have experimented with other possibilities, like keeping σ_k fixed over some number of serious steps, as well as with some more involved strategies. While improvements are possible, at this time we did not find them significant enough, with respect to the simple (4.5), to warrant their description. Generally, our experiments are intended for merely verifying that the proposed algorithm works and in a reasonable way. We did not spend much time on tuning various parameters to obtain an efficient code. To achieve this, as a first step, one should dispense with the generic `qp.m` MATLAB QP solver, which is known to be problematic (and was observed to be a limitation for our experiments as well). Instead, some good specialized solver (e.g., based on [16, 10]) has to be employed.

Our results are summarized in Table 4.1. We report on problems of dimensions $n = 5$ and $n = 10$, with various degrees of degeneracy of matrix Q, i.e., for different values of rank $Q = r < n$. Note that the number of constraints in (4.1) is $2n + 1$. For each pair of n and r the results are averaged over 20 runs. For all the problems, $l = 5$ and $l_0 = 3$; i.e., f_1 is defined by a maximum of five quadratic functions, with three of them being active at $\bar{x}$. We found that moderate variations of l and l_0 do not change much of the average behavior of the method. We thus keep them fixed in our report, to simplify the table. We report the number of times (out of 20 runs) that convergence had been declared according to the stopping rule (4.4), and the number of times this did not happen (declared as a failure) after a maximum allowed number of calls to the oracle (i.e., evaluations of f_1, f_2, and of their subgradients). In the case of $n = 5$, the maximal number of oracle calls is 100, and in the case of $n = 10$, it is 200. For both outcomes, we report the average number of oracle calls at termination (which is redundant in the case of failures) and the average of the relative accuracies achieved with respect to the optimal value $\bar{c}$ of problem (4.1) and of the (in)feasibility measure (the optimal value of f_1, which is zero). Specifically, in Table 4.1, we denote

$$R_1 = |(f_1(x^k) - \bar{c})/(f_1(x^0) - \bar{c})|, \quad R_2 = f_2(x^k)/f_2(x^0),$$

where x^k is the last serious iterate before termination.

We note that even in the cases of "failure" the method actually makes reasonable progress to the solution of the problem, as evidenced by the values of R_1 and R_2 in Table 4.1. We believe that a more careful implementation, including a better QP solver, should improve the accuracy (especially in higher dimensions) and eliminate "failures" of nonsatisfaction of the stopping rule (4.4). To this end, we observed that in most cases, the values of R_1 and R_2 (which measure actual proximity to solution) are very satisfactory, and close to those reported at termination, well before the stopping rule (4.4) is activated or the maximum number of oracle calls is reached. To some extent, this is quite normal for bundle methods, as they have to generate enough information in order to "recognize" optimality of the current point. For example, even starting with $x^0 = \bar{x}$, about 20 oracle calls were required in our experiments before the method stopped according to (4.4). But in some cases, even when the values in the stopping test (4.4) are already quite close to the required tolerances relatively early, it proves difficult to get more precision and satisfy (4.4). As already stated, we believe that the QP solver used in our implementation is likely the main reason we are not able to progress to higher accuracy with respect to stopping test (4.4). In any case, we believe that Table 4.1 shows reasonable behavior of Algorithm 2.1, even in our simple implementation, on problems with complementarity constraints (which is a difficult class of problems). Finally, we observe that degeneracy of the matrix Q defining complementarity constraints is not a problem for our algorithm at all. Actually, problems with higher degeneracy of Q appear even easier to solve. We conjecture that the reason for this is that, in the case of high degeneracy of Q, the feasible set of (4.1) is larger and the function f_2 is easier to minimize. This may make the overall problem easier to deal with in our setting.

5. Concluding remarks. We have presented a bundle method for solving a nonsmooth convex bilevel problem, which includes standard nonsmooth constrained optimization as a special case. The attractive feature of the method is that it is completely explicit. In particular, it does not require an iterative solution (not even approximate) of any optimization subproblems with general structure. Moreover, in the case of optimization, no constraint qualifications are required for convergence.

Acknowledgment. The author thanks Claudia Sagastizábal for her MATLAB unconstrained bundle code, which served as the basis for the implementation of Algorithm 2.1.

REFERENCES

[1] A. Auslender, *Numerical methods for nondifferentiable convex optimization*, Math. Program. Stud., 30 (1987), pp. 102–126.
[2] D. P. Bertsekas, *Convex Analysis and Optimization*, Athena Scientific, Belmont, MA, 2003.
[3] J. F. Bonnans, J. Ch. Gilbert, C. Lemaréchal, and C. Sagastizábal, *Numerical Optimization: Theoretical and Practical Aspects*, Springer-Verlag, Berlin, 2003.
[4] A. Cabot, *Proximal point algorithm controlled by a slowly vanishing term: Applications to hierarchical minimization*, SIAM J. Optim., 15 (2005), pp. 555–572.
[5] R. Cominetti and M. Courdurier, *Coupling general penalty schemes for convex programming with the steepest descent and the proximal point algorithm*, SIAM J. Optim., 13 (2002), pp. 745–765.
[6] R. Correa and C. Lemaréchal, *Convergence of some algorithms for convex minimization*, Math. Program., 62 (1993), pp. 261–275.
[7] M. Kočvara and J. V. Outrata, *Optimization problems with equilibrium constraints and their numerical solution*, Math. Program., 101 (2004), pp. 119–149.
[8] A. V. Fiacco and G. P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley & Sons, New York, 1968.

[9] R. Fletcher and S. Leyffer, *A Bundle Filter Method for Nonsmooth Nonlinear Optimization*, Numerical Analysis Report NA/195, Department of Mathematcs, The University of Dundee, Scotland, 1999.
[10] A. Frangioni, *Solving semidefinite quadratic optimization problems within nonsmooth optimization problems*, Comput. Oper. Res., 23 (1996), pp. 1099–1118.
[11] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms*, Springer-Verlag, Berlin, 1993.
[12] V. V. Kalashnikov and N. I. Kalashnikova, *Solving two-level variational inequality. Hierarchical and bilevel programming*, J. Global Optim., 8 (1996), pp. 289–294.
[13] E. Karas, A. Ribeiro, C. Sagastizábal, and M. Solodov, *A bundle-filter method for nonsmooth convex constrained optimization*, Math. Program., to appear.
[14] K. C. Kiwiel, *An exact penalty function algorithm for nonsmooth convex constrained minimization problems*, IMA J. Numer. Anal., 5 (1985), pp. 111–119.
[15] K. C. Kiwiel, *Methods of Descent for Nondifferentiable Optimization*, Lecture Notes in Math. 1133, Springer-Verlag, Berlin, 1985.
[16] K. C. Kiwiel, *A method for solving certain quadratic programming problems arising in nonsmooth optimization*, IMA J. Numer. Anal., 6 (1986), pp. 137–152.
[17] K. C. Kiwiel, *A constraint linearization method for nondifferentiable convex minimization*, Numer. Math., 51 (1987), pp. 395–414.
[18] K. C. Kiwiel, *Exact penalty functions in proximal bundle methods for constrained convex nondifferentiable minimization*, Math. Programming, 52 (1991), pp. 285–302.
[19] C. Lemaréchal, A. Nemirovskii, and Yu. Nesterov, *New variants of bundle methods*, Math. Programming, 69 (1995), pp. 111–148.
[20] Z.-Q. Luo, J.-S. Pang, and D. Ralph, *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, Cambridge, UK, 1996.
[21] R. Mifflin, *An algorithm for constrained optimization with semismooth functions*, Math. Oper. Res., 2 (1977), pp. 191–207.
[22] R. Mifflin, *A modification and extension of Lemarechal's algorithm for nonsmooth minimization*, Math. Programming Stud., 17 (1982), pp. 77–90.
[23] B. T. Polyak, *Introduction to Optimization*, Optimization Software, Inc., Publications Division, New York, 1987.
[24] C. Sagastizábal and M. Solodov, *An infeasible bundle method for nonsmooth convex constrained optimization without a penalty function or a filter*, SIAM J. Optim., 16 (2005), pp. 146–169.
[25] M. V. Solodov, *An explicit descent method for bilevel convex optimization*, J. Convex Anal., 14 (2007), pp. 227–238.